Document Title	SWS_EthernetSwitchDriver: Complete Change Documentation 4.3.0 - 4.3.1
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	695

Document Status	Final
Part of AUTOSAR Standard	Classic Platform
Part of Standard Release	4.3.1

Table of Contents

1 SWS_EthernetSwitchDriver

1.1 Specification Item ECUC_EthSwt_00003

Trace References:

none

Content:

Container Name	EthSwtGeneralEthSwtGeneral
Description	General configuration of Ethernet Switch Driver module.
Configuration Parameters	

Included parameters:

Included Parameters	
Parameter Name	SWS Item ID
EthSwtDevErrorDetect	ECUC_EthSwt_00002
EthSwtEnableVlanApi	ECUC_EthSwt_00055
EthSwtGetArlTableApi	ECUC_EthSwt_00052
EthSwtGetBaudRateApi	ECUC_EthSwt_00121
EthSwtGetBufferLevelApi	ECUC_EthSwt_00040
EthSwtGetCfgDataRawDone	ECUC_EthSwt_00124
EthSwtGetCfgHexDumpApi	ECUC_EthSwt_00093
EthSwtGetCfgHexDumpLengthApi	ECUC_EthSwt_00094
EthSwtGetCfgRaw	ECUC_EthSwt_00123
EthSwtGetDropCountApi	ECUC_EthSwt_00053
EthSwtGetDuplexModeApi	ECUC_EthSwt_00122
EthSwtGetLinkStateApi	ECUC_EthSwt_00120
EthSwtGetMacLearningModeApi	ECUC_EthSwt_00061
EthSwtGetPortCableDiagnosticsResultApi	ECUC_EthSwt_00092
EthSwtGetPortIdentifierApi	ECUC_EthSwt_00083
EthSwtGetPortMacAddrApi	ECUC_EthSwt_00051
EthSwtGetPortMirrorStateApi	ECUC_EthSwt_00087
EthSwtGetPortSignalQualityApi	ECUC_EthSwt_00082
EthSwtGetRxStatsApi	ECUC_EthSwt_00065
EthSwtGetSwitchIdentifierApi	ECUC_EthSwt_00084
EthSwtGetSwitchPortModeApi	ECUC_EthSwt_00118
EthSwtGetSwitchRegApi	ECUC_EthSwt_00066

Included Parameters	
Parameter Name	SWS Item ID
EthSwtGetTxErrorCounterValuesApi	ECUC_EthSwt_00100
EthSwtGetTxStatsApi	ECUC_EthSwt_00099
EthSwtGlobalTimeSupportApi	ECUC_EthSwt_00107
EthSwtIndex	ECUC_EthSwt_00033
EthSwtLinkDownCallout	ECUC_EthSwt_00115
EthSwtLinkDownUser	ECUC_EthSwt_00048
EthSwtLinkUpCallout	ECUC_EthSwt_00116
EthSwtLinkUpUser	ECUC_EthSwt_00068
EthSwtLowPowerModeSupport	ECUC_EthSwt_00102
EthSwtMainFunctionPeriod	ECUC_EthSwt_00071
EthSwtManagementSupportApi	ECUC_EthSwt_00108
EthSwtMgmtInfoIndicationTimeout	ECUC_EthSwt_00109
EthSwtPersistentConfigurationResult	ECUC_EthSwt_00062
EthSwtPersistentConfigurationResultUser	ECUC_EthSwt_00063
EthSwtPublicCddHeaderFile	ECUC_EthSwt_00064
EthSwtReadPortMirrorConfigurationApi	ECUC_EthSwt_00086
EthSwtReadTrcvRegisterApi	ECUC_EthSwt_00069
EthSwtResetConfigurationApi	ECUC_EthSwt_00049
EthSwtSetForwardingModeApi	ECUC_EthSwt_00104
EthSwtSetMacLearningModeApi	ECUC_EthSwt_00060
EthSwtSetPortLoopbackModeApi	ECUC_EthSwt_00090
EthSwtSetPortMirrorStateApi	ECUC_EthSwt_00088
EthSwtSetPortTestModeApi	ECUC_EthSwt_00089
EthSwtSetPortTxModeApi	ECUC_EthSwt_00091
EthSwtSetSwitchPortModeApi	ECUC_EthSwt_00117
EthSwtSetSwitchRegApi	ECUC_EthSwt_00067
EthSwtStartSwitchPortAutoNegotiationApi	ECUC_EthSwt_00119
EthSwtStoreConfigurationApi	ECUC_EthSwt_00050
EthSwtVerifyConfigApi	ECUC_EthSwt_00105
EthSwtVersionInfoApi	ECUC_EthSwt_00031
EthSwtWritePortMirrorConfigurationApi	ECUC_EthSwt_00085
EthSwtWriteTrcvRegisterApi	ECUC_EthSwt_00070

Included containers:

No Included Containers

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76531: [EthSwt] Syntax for callback functions <user>_LinkUp and <user>_LinkDown might be wrong

  Problem description:

  The callback function for <User>_LinkUp and <User>_LinkDown to inform the user about a Link state change might be wrong
  Currently the parameter of these both functions are defined as Pointer to uint8
  This makes no sense for me to use here pointers and not a uint 8 variable.

  Agreed solution:

  Chapter 8.6.3 Configurable Interfaces
  add following intro:
  In this chapter all interfaces are listed where the target function could be configured. The names of these kind of interfaces are not fixed because they are configurable.

  Add Subchapter with interface
  move SWS_EthSwt_00117 here and change to:
  Service name: <EthSwtLinkDownCallout>
  syntax: void <EthSwtLinkDownCallout>(uint8 SwitchIdx, uint8 PortIdx)
  Service ID: 0x19
  Sync/Async:Synchronous
  Reentrancy: Non Reentrant
  Parameters (in): SwitchIdx: Index of the switch within the context of the Ethernet Switch Driver
  PortIdx: Index of the port at the addressed switch
  Parameters (in/out): none
  Parameters (out): none
  Return value: none
  Description: Is called, if a link which is configured goes down.

  ~[SWS_EthSwt_00118]The function <EthSwtLinkDownCallout> shall be called if a link, which is configured, goes down. The function provides the Switch index and the Port index, such that the port which went down can be identified.

  -[SWS_EthSwt_00119]

  add second subchapter:

  move SWS_EthSwt_00203 here and change to:
  Service name: <EthSwtLinkUpCallout>
  syntax: void <EthSwtLinkUpCallout>(uint8 SwitchIdx, uint8 PortIdx)
  Service ID: 0x1a
  Sync/Async:Synchronous

Reentrancy: Non Reentrant
Parameters (in): SwitchIdx: Index of the switch within the context of the Ethernet Switch Driver
PortIdx: Index of the port at the addressed switch
Parameters (in/out): none
Parameters (out): none
Return value: none
Description: Is called, if a link which is configured goes up

~[SWS_EthSwt_00204]The function <EthSwtLinkUpCallout> shall be called if a link which is configured goes up. The function provides the Switch index and the Port index, such that the port which went up can be identified.

-[SWS_EthSwt_00205] This is no clear as it is a configurable interface now

~ECUC_EthSwt_00048 set EthSwtLinkDownUser to obsolete
+ add optional function name parameter with same attributes named EthSwtLinkDownCallout
change description to Defines the function name for <EthSwtLinkDownCallout>.

~ECUC_EthSwt_00068 set EthSwtLinkUpUser to obsolete
+add optional function name parameter with same attributes named EthSwtLinkUpCallout
change description to Defines the function name for <EthSwtLinkUpCallout>.

set ECUC_EthSwt_00047 EthSwtPortEnableLinkDownCallback to obsolete.

Header File name is defined by ECUC_EthSwt_00064 EthSwtPublicCddHeaderFile.
–Last change on issue 76531 comment 54–

BW-C-Level:

Application	Specification	Bus
1	4	1

- RfC #77018: [EthSwt] Clarification about the use case regarding API "EthSwt_GetCfgHexDump"

  Problem description:

  We discussed the use case regarding the API "EthSwt_GetCfgHexDump". Daimler want to read out the register memory of an Ethernet switch as a memory dump. The memory dump is transmitted via an diagnostic service over the network consecutively to an diagnostic tester. The memory dump is used to verify and compare to the expected / required values. It is not clear if the API is sufficient for this use case.

Agreed solution:

Capter 7.1.2.13 exchange in listing of APIS EthSwt_GetCfgHexDump with EthSwt_GetCfgDataRaw and EthSwt_GetCfgHexDumpLength with EthSwt_GetCfgDataInfo.

Chapter 8
-SWS_EthSwt_91026,
-SWS_EthSwt_00352
-SWS_EthSwt_00353
-SWS_EthSwt_00355
-ECUC_EthSwt_00093
-SWS_EthSwt_91027
-SWS_EthSwt_00356
-SWS_EthSwt_00357
-SWS_EthSwt_00358
-SWS_EthSwt_00359
Set APIs EthSwt_GetCfgHexDump and EthSwt_GetCfgHexDumpLength and config parameters EthSwtGetCfgHexDumpApi and EthSwt_GetCfgHexDumpLengthApi to deprecated or delete.

Chapter 10
Add parameter to enable disable APIs with name EthSwtGetCfgRaw to EthSwtGeneral-Container
Description Disable /Enable support of reading raw data from switch memory.
Multiplicity: 1
Type: EcucBooleanParamDef
Default: false
Post build variant false
Config class Pre-compile time for all variants.
scope: local

Introduce following new APIs to function: definitions
EthSwt_GetCfgDataRaw(
uint8 SwitchIdx,
uint32 Offset,
uint16 Length,
uint8 *BufferPtr
)

[attributes
- asynchronous
- non-reentrant
parameters IN: SwitchIdx:Index of the Ethernet switch within the context of the Ethernet Switch Driver
Offset: Offset of the Ethernet switch memory from where the reading starts
Length: Length of data in bytes that shall be copied.
parameters OUT: BufferPtr: Pointer to the location where the data shall be copied.
Std_ReturnType: E_OK: the data read was triggered successfully. E_NOT_OK: the data read was not triggered successfully. (i.e. indexed Ethernet switch is not available)
Description:
Retrieves the data in memory of the indexed Ethernet switch in variable length.]

+[SWS_EthSwt_xxxxx1] The function EthSwt_GetCfgDataRaw() shall only be available if parameter EthSwtGetCfgRaw is set to TRUE.(SRS_BSW_00171)
+[SWS_EthSwt_xxxxx2] When calling the function EthSwt_GetCfgDataRaw(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.()

EthSwt_GetCfgDataInfo(
uint8 SwitchIdx,
uint32 *DataSizePtr,
uint32 *DataAdressPtr
)
[attributes
- synchronous
- reentrant
parameters IN: SwitchIdx:Index of the Ethernet switch within the context of the Ethernet Switch Driver
parameters OUT: DataSizePtr: Pointer to the location where the total size of the configuration data shall be copied.
DataAddressPtr: Pointer to the location where the start address of the configuration registers shall be copied.
Std_ReturnType: E_OK: the data was obtained successfully. E_NOT_OK: the data was not obtained successfully. (i.e. indexed Ethernet switch is not available)
Description:
Retrieves the total size of data and the memory start address of the indexed Ethernet Switch.]

+[SWS_EthSwt_xxxxx3] The function EthSwt_GetCfgDataInfo() shall only be available if parameter EthSwtGetCfgRaw is set to TRUE.(SRS_BSW_00171)
+[SWS_EthSwt_xxxxx4] When calling the function EthSwt_GetCfgDataInfo(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.()

add following API to chapter "8.6.3 Configurable interfaces"
<GetCfgDataRawDone>(
uint8 SwitchIdx
)
[attributes:
synchronous
reentrant
parameters IN: SwitchIdx: Index of the Ethernet switch where the Configuration is read.
Return value : void
Description: The call of the function EthSwt_GetCfgDataRaw() triggers a asynchrony read of a certain memory section of the Ethernet switch driver. If the read is done, the configured callout function <GetCfgDataRawDone> shall be called]

Add a parameter to EthSwtGeneral-Container
GetCfgDataRawDone
Decription: Defines the function name for <GetCfgDataRawDone>
Multplicity: 0 ..1
Type EcucFuntionNameDef
Variant and Variant Multiplicity false
Config class Pre-compile for all variants
scope: local
dependency: The function GetCfgDataRawDone shall only be configured if parameter EthSwtGetCfgRaw is set to TRUE.

Header file name parameter is defined in ECUC_EthSwt_00064
–Last change on issue 77018 comment 32–

BW-C-Level:

Application	Specification	Bus
1	4	1

- RfC #77043: [EthSwt] EthSwt APIs shouldnt be precompile ON/OFF based on EthTrcv parameters

  Problem description:

  For the APIs EthSwt_SetSwitchPortMode, EthSwt_GetSwitchPortMode, EthSwt_StartSwitchPortAutoNegotiation, EthSwt_GetLinkState, EthSwt_GetBaudRate and EthSwt_GetDuplexMode there is a requirement that states that the API should be configurable On/Off based on a EthTrcv parameter. For example for EthSwt_StartSwitchPortAutoNegotiation the requirement(SWS_EthSwt_00035) is: The function EthSwt_StartSwitchPortAutoNegotiation shall be pre-compile time configurable On/Off by the configuration parameter: EthTrcvStartAutoNegotiationApi.

  However, for other similar APIs a configuration parameter was defined in the switch driver. The functions are: EthSwt_GetPortSignalQuality, EthSwt_GetPortIdentifier, EthSwt_SetPortLoopbackMode, EthSwt_SetPortTxMode, EthSwt_GetPortCableDiagnosticsResult.

  Also, if multiple EthTrcv drivers are used, how will the precompile parameters be defined in the switch?

  Agreed solution:

  add parameters
  EthSwtSetSwitchPortModeApi, EthSwtGetSwitchPortModeApi, EthSwtStartSwitchPortAutoNegotiationApi, EthSwtGetLinkStateApi, EthSwtGetBaudRateApi and EthSwtGetDuplexModeApi
  Description: Enables / disables the EthSwt_SetSwitchPortMode / EthSwt_GetSwitchPortMode / EthSwt_StartSwitchPortAutoNegotiation / EthSwt_GetLinkState/ EthSwt_GetBaudRate / EthSwt_GetDuplexMode
  Multiplicity: 1
  Type EcucBooleanParamDef
  Default: False
  Variant false
  Config Class Pre-compile for all variants

  ~[SWS_EthSwt_00022]
  replace EthTrcvSetTransceiverModeApi with EthSwtSetSwitchPortModeApi.
  ~SWS_EthSwt_00029
  replace EthTrcvGetTransceiverModeApi with EthSwtGetSwitchPortModeApi.
  ~SWS_EthSwt_00035
  replace EthTrcvStartAutoNegotiationApi with EthSwtStartSwitchPortAutoNegotiationApi
  ~[SWS_EthSwt_00042]

replace EthTrcvGetLinkStateApi with EthSwtGetLinkStateApi
~SWS_EthSwt_00049
replace EthTrcvGetBaudRateApi with EthSwtGetBaudRateApi
~SWS_EthSwt_00056
replace EthTrcvGetDuplexModeApi with EthSwtGetDuplexModeApi
–Last change on issue 77043 comment 2–

BW-C-Level:

Application	Specification	Bus
1	3	1

1.2 Specification Item ECUC_EthSwt_00005

Trace References:

none

Content:

Container Name	EthSwtPortEthSwtPort		
Description	Configuration of one Ethernet Switch Port.		
Post-Build Variant Multiplicity	true		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Configuration Parameters			

Included parameters:

Included Parameters	
Parameter Name	SWS Item ID
EthSwtPortEnableLinkDownCallback	ECUC_EthSwt_00047
EthSwtPortIdx	ECUC_EthSwt_00013
EthSwtPortMacLayerSpeed	ECUC_EthSwt_00114
EthSwtPortMacLayerSubType	ECUC_EthSwt_00113
EthSwtPortMacLayerType	ECUC_EthSwt_00072
EthSwtPortPhysicalLayerType	ECUC_EthSwt_00054
EthSwtPortPredefinedMacAddresses	ECUC_EthSwt_00032
EthSwtPortRole	ECUC_EthSwt_00101
EthSwtPortTimeStampSupport	ECUC_EthSwt_00112
EthSwtPortTrcvRef	ECUC_EthSwt_00041

Included containers:

Included Containers		
Container Name	Multiplicity	Scope / Dependency
EthSwtPortEgress	1	Configuration of one Ethernet Switch Port Egress behavior.
EthSwtPortIngress	1	Configuration of one Ethernet Switch Port ingress behavior.
EthSwtPortVlanMembership	0..4095	Description Determines the membership of this port to the virtual network, i.e. frames with this VID can be received and transmitted via this port.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76343: Configuration parameters for Ethernet MAC layer types is incomplete

  Problem description:

  With https://www.autosar.org/bugzilla/show_bug.cgi?id=73074 the configuration of the MAC layer type was separated from the physical layer type. The configuration parameter EthSwtPortMacLayerType allows for the configuration of "families" of MAC layer types, namely xMII, xGMII, and xxGMII.

  The concrete MAC layer type (e.g., SGMII vs. RGMII) however cannot be configure by this parameter. - Available switches (e.g., Broadcom's "Leo") however require the configuration of a *concrete* MAC layer type (and not only the configuration of a "family" of MAC layer types) for the individual port.

  The same holds probably true for the Ethernet Driver and the Ethernet Transceiver driver.

  One possibility to solve this is to introduce an additional config parameter named something like EthSwtPortMacLayerSubType with the following enum values "standard", "reduced", "reversed", "serial", "universal serial". - Thus for example the combination of EthSwtPortMacLayerType=xGMII and EthSwtPortMacLayerSubType="reversed" would yield RvGMII.

  Agreed solution:

  EthSwt:
  add config parameter to container EthSwtPort named EthSwtPortMacLayerSubType with the following enum values "STANDARD: standard media-independent interface", "REDUCED: Reduced media-independent interface", "REVERSED: reversed media-independent interface (to provide direct connection between two Ethernet MACs)", "SERIAL: low-power and low pin-count serial 8b/10b-coded

media-independent interface", "UNIVERSAL SERIAL": Universal low-power and low pin-count serial 8b/10b-coded media-independent interface".

add config parameter to container EthSwtPort named EthSwtPortMacLayerSpeed with the following enum values "ETH_MAC_LAYER_SPEED_10M", "ETH_MAC_LAYER_SPEED_100M", "ETH_MAC_LAYER_SPEED_1G", "ETH_MAC_LAYER_SPEED_10G".
All parameters shall have the same Multplicity, Variant and config class as EthSwtPortMacLayerType.

add description for EthSwt_PortMacLayerSubTypes:
Defines the MAC layer subtype of a switch port

add description for EthSwt_PortMacLayerSpeed:
Defines the baud rate of the MAC layer

Eth:
add config parameter to container EthCtrlConfig named EthCtrlMacLayerSubType with the following enum values "STANDARD", "REDUCED", "REVERSED", "SERIAL", "UNIVERSAL SERIAL". - Thus for example the combination of EthCtrlMacLayerType=xGMII and EthCtrlMacLayerSubType ="REVERSED" would yield RvGMII.
add config parameter to container EthCtrlConfig named EthCtrlMacLayerSpeed with the following enum values "ETH_MAC_LAYER_SPEED_10M", "ETH_MAC_LAYER_SPEED_100M", "ETH_MAC_LAYER_SPEED_1G", "ETH_MAC_LAYER_SPEED_10G".

add description for Eth_PortMacLayerSubTypes:
Defines the MAC layer subtype of a switch port

add description for Eth_PortMacLayerSpeed:
Defines the baud rate of the MAC layer

EthTrcv:
add config parameter to container EthTrcvCtrlConfig named EthTrcvPortMacLayerSubType with the following enum values "STANDARD", "REDUCED", "REVERSED", "SERIAL", "UNIVERSAL SERIAL". - Thus for example the combination of EthTrcvPortMacLayerType=xGMII and EthTrcvPortMacLayerSubType="REVERSED" would yield RvGMII.
add config parameter to container EthTrcvCtrlConfig named EthTrcvPortMacLayerSpeed with the following enum values "ETH_MAC_LAYER_SPEED_10M", "ETH_MAC_LAYER_SPEED_100M", "ETH_MAC_LAYER_SPEED_1G",

"ETH_MAC_LAYER_SPEED_10G".

add description for EthTrcv_PortMacLayerSubTypes:
Defines the MAC layer subtype of a switch port

add description for EthTrcv_PortMacLayerSpeed:
Defines the baud rate of the MAC layer

All parameters shall have the same Multplicity, Variant and config class as EthSwtPortMacLayerType.

No upstream Mapping as this decision is made on ECU-Configuration.
–Last change on issue 76343 comment 30–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.3 Specification Item ECUC_EthSwt_00016

Trace References:

none

Content:

Container Name	EthSwtDemEventParameterRefsEthSwtDemEventParameterRefs		
Description	Container for the references to DemEventParameter elements which shall be invoked using the API Dem_SetEventStatus in case the corresponding error occurs. The EventId is taken from the referenced DemEventParameter's DemEventId symbolic value. The standardized errors are provided in this container and can be extended by vendor-specific error references.		
Post-Build Variant Multiplicity	true		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Configuration Parameters			

Included parameters:

Included Parameters	
Parameter Name	SWS Item ID
ETHSWT_E_ACCESS	ECUC_EthSwt_00006
ETHSWT_E_SYNCPORT2PHY	ECUC_EthSwt_00125

Included containers:

No Included Containers

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77250: [EthIf][EthSwt][EthTrcv] EthSwt_SetSwitchPortMode() broken for ports without Ethernet transceiver

  Problem description:

  SWS_EthSwt_00019 states that EthSwt_SetSwitchPortMode() shall (unconditionally) invoke EthTrcv_SetTransceiverMode() of the Ethernet Transceiver Driver.

  This is bogus for the following two reasons:
  1) a particular switch port does not necessarily have an associated Ethernet transceiver. - In that case EthSwtPortTrcvRef won't be present. This is probably the case for MAC2MAC links if EthSwtPortRole is present an either set to ETHSWT_HOST_PORT or to ETHSWT_UP_LINK_PORT.
  2) Even if the particular switch port has an associated Ethernet transceiver, SWS_EthTrcv_00043 states that the EthTrcv shall invoke EthIf_TrcvModeIndication() as a response to EthTrcv_SetTransceiverMode().

  IMHO 1) should be solved by extending SWS_EthSwt_00019 distinguishing between ports which are associated with a transceiver (i.e., EthSwtPortTrcvRef present) - in that case EthTrcv_SetTransceiverMode() should be called - and ports which are not associated with a transceiver (i.e., EthSwtPortTrcvRef is not present) - in that case something else has to happen and this needs to be specified (probably the switch port shall be internally just disabled - maybe this needs to happen in the first case as well?)

  2) leads to the following call chain: EthIf_SwitchPortGroupRequestMode() -> EthSwt_SetSwitchPortMode() -> EthTrcv_SetTransceiverMode() -> EthIf_TrcvModeIndication(). Thus the indication actually bypasses the EthSwt which IMHO is extremely ugly. Additionally for switch ports without associated transceiver, the EthIf will never get a EthIf_TrcvModeIndication() ...

  Agreed solution:

  === System Template ===
  + Add upstream mapping of EthSwtPortTrcvRef to CouplingPort.physicalLayerType

  === EthSwt ===

~ch. 7.2.4 Production Errors
move SWS_EthSwt_00113 to 7.2.5 Extended production Errors
~SWS_EthSwt_00113 change detection criteria to
Fail:When access to the Ethernet Switch fails the module shall report the extended production error with event status DEM_EVENT_STATUS_PREFAILED to DEM.
Pass:When access to the Ethernet Switch succeeds the module shall report the extended production error with event status DEM_EVENT_STATUS_PREPASSED to DEM.

+ SWS_EthSwt_xxxxx new ext. production Error
Error Name: ETHSWT_E_SYNCPORT2PHY
Short Description: Ethernet switch port and the referenced Ethernet transceiver are in contradicting modes.
Long description: While getting the Ethernet switch port mode, the Ethernet switch driver detected an inconsistent state between Ethernet switch port and the referenced Ethernet transceiver Mode.
Detection Criteria: Fail: When getting the Ethernet switch port mode together with the Ethernet transceiver mode and the mode of the two referenced modules was found
inconsistent the module shall report the extended production error with event status DEM_EVENT_STATUS_PREFAILED to DEM.
Pass:When getting the Ethernet switch port mode together with the Ethernet transceiver mode and the mode of the two referenced modules was found
consistent the module shall report the extended production error with event status DEM_EVENT_STATUS_PREPASSED to DEM.
all other attributer: N/A

~ ch. 8.3.2 EthSwt_SwitchInit
~[SWS_EthSwt_00016] change "production error" to "extended production error"

~ch. 8.3.3 EthSwt_SetSwitchPortMode
~[SWS_EthSwt_00019]The function EthSwt_SetSwitchPortMode() shall put the indexed port of the switch into the specified mode. If EthSwtPort references an EthTrcv then the function EthTrcv_SetTransceiverMode() of the Ethernet Transceiver Driver shall additionally be called with the corresponding transceiver mode.

+[SWS_EthSwt_xxxxx] When calling the function EthSwt_SetSwitchPortMode() with mode ETHTRCV_MODE_DOWN, the EthSwt shall disable the Ethernet switch port directly for reduction of power consumption, if it is possible.

+[SWS_EthSwt_xxxxx] When calling the function EthSwt_SetSwitchPortMode(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.

+[SWS_EthSwt_xxxxxx] If EthSwtPort does not references an EthTrcv, EthSwt shall indicate a mode of the port by the API EthIf_SwitchPortModeIndication latest during the next EthSwt_MainFunction.(SRS_ETH_00118)

~ch. 10.1.3 EthSwtDemEventParameterRefs
+ SWS Item ECUC_EthSwt_xxxxx1 : Name ETHSWT_E_SYNCPORT2PHY
Parent Container EthSwtDemEventParameterRefs
Description Reference to the DemEventParameter which shall be issued when the error "Ethernet switch port and the referenced Ethernet transceiver are in contradicting modes" has occurred.
Multiplicity 0..1
Type Symbolic name reference to [DemEventParameter]
Post-Build Variant Multiplicity true
Post-Build Variant Value true
Multiplicity Configuration
Class Pre-compile time X VARIANT-PRE-COMPILE
Link time X VARIANT-LINK-TIME
Post-build time X VARIANT-POST-BUILD
Value Configuration Class
Pre-compile time X VARIANT-PRE-COMPILE
Link time X VARIANT-LINK-TIME
Post-build time X VARIANT-POST-BUILD
Scope / Dependency
scope: local

~ch. 10.1.5 EthSwtPort
~ECUC_EthSwt_00041 EthSwtPortTrcvRef change following attributes:
Multiplicity Configuration Class | Link time | VARIANT-LINK-TIME, VARIANT-POST-BUILD
| Post build | –
add to dependency: If EthSwtPortPhysicalLayerType is defined, then EthSwtPortTrcvRef holds the reference to the corresponding EthTrcv.

~ECUC_EthSwt_00054 add dependency to EthSwtPortPhysicalLayerType: If a EthSwtPort has an EthSwtPortPhysicalLayerType then EthSwtPort shall reference an EthTrcv.

=== EthIf ===

Add to chapter 8.4 (callback notifications):
+ EthIf_SwitchPortModeIndication
Syntax: void EthIf_SwitchPortModeIndication(uint8 SwitchIdx, uint8 SwitchPortIdx, EthSwt_ModeType PortMode)
Sevice ID: pick a free one
Sync/Async: Synchronous
Reentrancy: Non Reentrant
Parmaters in: SwitchIdx: Index of the switch within the context of the Ethernet Switch Driver
SwitchPortIdx: Index of the port at the addressed switch
PortMode: notified Ethernet Switch port mode.
Description: The EthIf shall determine the expected notifications based on the EthSwtPort configuration. In case the EthSwtPort references an EthTrcv the EthIf expects a notification from the EthTrcv via API EthIf_TrcvModeIndication(). Otherwise the EthIf expects a notification from the EthSwt via API EthIf_SwitchPortModeIndication().

~SWS_EthIf_00232
change from
Called asynchronously when mode has been read out. Triggered by previous
EthTrcv_SetTransceiverMode call. Can directly be called within the trigger
functions.
To
Called asynchronously when a mode change has been read out. If the function is triggered by previous call of
EthTrcv_SetTransceiverMode it can directly be called within the trigger function.
–Last change on issue 77250 comment 50–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.4 Specification Item ECUC_EthSwt_00020

Trace References:

none

Content:

Name	EthSwtPortSchedulerPredecessorOrder

Description	Defines the order of the scheduler predecessors. This value has to be understood as a relative value, i.e. the value shows only the relative ordering of the elements. The highest value has the highest priority and gaps are allowed (not dense based). The values need to be unique within one EthSwtPortScheduler.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 18446744073709551615		
Default value	–		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: ECU		

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #74858: Clarification on EthSwtPortSchedulerPredecessor

  Problem description:

  Description of Parameter ' EthSwtPortSchedulerPredecessor' is not significant.

  Clarification of this parameter is required:
  * Is 0 highest priority or the lowest priority ?
  * Is this parameter zero dense based?

  Agreed solution:

  add to end of implementation note in chapter 7.1.2.2 Configuration of Egress Port structure
  EthSwtPortSchedulerPredecessorOrder could be used to define the service weight of a Fifo queue if EthSwtPortSchedulerAlgorithm is configured to ETHSWT_SCHEDULER_DEFICIT_ROUND_ROBIN or ETHSWT_SCHEDULER_STRICT_PRIORITY. In case of EthSwtPortSchedulerAlgorithm is ETHSWT_SCHEDULER_STRICT_PRIORITY the EthSwtPortSchedulerPredecessorOrder defines the order of the schedulers,
  The fifo queue which is referenced (via Shaper or directly) by the Scheduler with the highest value of EthSwtPortSchedulerPredecessorOrder is the Fifo queue with the highest priority.

  ECUC_EthSwt_00020 EthSwtPortSchedulerPredecessorOrder:

  Description: Defines the order of the scheduler predecessors.
  This value has to be understood as a relative value, i.e. the values shows only the relative ordering of the elements.

The highest value has the highest priority and gaps are allowed (not dense based). The values need to be unique within one EthSwtPortScheduler.
–Last change on issue 74858 comment 10–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.5 Specification Item ECUC_EthSwt_00041

Trace References:

none

Content:

Name	EthSwtPortTrcvRefEthSwtPort.EthSwtPortTrcvRef		
Parent Container	EthSwtPort		
Description	Reference to the Ethernet transceiver driver this EthSwtPort is connected with.		
Multiplicity	0..1		
Type	Symbolic name reference to [EthTrcvConfig]		
Post-Build Variant Multiplicity	true		
Post-Build Variant Value	true		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	X –	VARIANT-POST-BUILD
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: ECU dependency: If EthSwtPortPhysicalLayerType is defined, then EthSwtPortTrcvRef holds the reference to the corresponding EthTrcv.		

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77250: [EthIf][EthSwt][EthTrcv] EthSwt_SetSwitchPortMode() broken for ports without Ethernet transceiver

  Problem description:

  SWS_EthSwt_00019 states that EthSwt_SetSwitchPortMode() shall (unconditionally) invoke EthTrcv_SetTransceiverMode() of the Ethernet Transceiver Driver.

  This is bogus for the following two reasons:

1) a particular switch port does not necessarily have an associated Ethernet transceiver. - In that case EthSwtPortTrcvRef won't be present. This is probably the case for MAC2MAC links if EthSwtPortRole is present an either set to ETHSWT_HOST_PORT or to ETHSWT_UP_LINK_PORT.
2) Even if the particular switch port has an associated Ethernet transceiver, SWS_EthTrcv_00043 states that the EthTrcv shall invoke EthIf_TrcvModeIndication() as a response to EthTrcv_SetTransceiverMode().

IMHO 1) should be solved by extending SWS_EthSwt_00019 distinguishing between ports which are associated with a transceiver (i.e., EthSwtPortTrcvRef present) - in that case EthTrcv_SetTransceiverMode() should be called - and ports which are not associated with a transceiver (i.e., EthSwtPortTrcvRef is not present) - in that case something else has to happen and this needs to be specified (probably the switch port shall be internally just disabled - maybe this needs to happen in the first case as well?)

2) leads to the following call chain: EthIf_SwitchPortGroupRequestMode() -> EthSwt_SetSwitchPortMode() -> EthTrcv_SetTransceiverMode() -> EthIf_TrcvModeIndication(). Thus the indication actually bypasses the EthSwt which IMHO is extremely ugly. Additionally for switch ports without associated transceiver, the EthIf will never get a EthIf_TrcvModeIndication() ...

Agreed solution:

=== System Template ===
+ Add upstream mapping of EthSwtPortTrcvRef to CouplingPort.physicalLayerType

=== EthSwt ===

~ch. 7.2.4 Production Errors
move SWS_EthSwt_00113 to 7.2.5 Extended production Errors
~SWS_EthSwt_00113 change detection criteria to
Fail:When access to the Ethernet Switch fails the module shall report the extended production error with event status DEM_EVENT_STATUS_PREFAILED to DEM.
Pass:When access to the Ethernet Switch succeeds the module shall report the extended production error with event status DEM_EVENT_STATUS_PREPASSED to DEM.

+ SWS_EthSwt_xxxxx new ext. production Error
Error Name: ETHSWT_E_SYNCPORT2PHY
Short Description: Ethernet switch port and the referenced Ethernet transceiver are in contradicting modes.
Long description: While getting the Ethernet switch port mode, the Ethernet switch driver detected an inconsistent state between Ethernet switch port and the

referenced Ethernet transceiver Mode.
Detection Criteria: Fail: When getting the Ethernet switch port mode together with the Ethernet transceiver mode and the mode of the two referenced modules was found
inconsistent the module shall report the extended production error with event status DEM_EVENT_STATUS_PREFAILED to DEM.
Pass:When getting the Ethernet switch port mode together with the Ethernet transceiver mode and the mode of the two referenced modules was found
consistent the module shall report the extended production error with event status DEM_EVENT_STATUS_PREPASSED to DEM.
all other attributer: N/A

~ ch. 8.3.2 EthSwt_SwitchInit
~[SWS_EthSwt_00016] change "production error" to "extended production error"

~ch. 8.3.3 EthSwt_SetSwitchPortMode
~[SWS_EthSwt_00019]The function EthSwt_SetSwitchPortMode() shall put the indexed port of the switch into the specified mode. If EthSwtPort references an EthTrcv then the function EthTrcv_SetTransceiverMode() of the Ethernet Transceiver Driver shall additionally be called with the corresponding transceiver mode.

+[SWS_EthSwt_xxxxx] When calling the function EthSwt_SetSwitchPortMode() with mode ETHTRCV_MODE_DOWN, the EthSwt shall disable the Ethernet switch port directly for reduction of power consumption, if it is possible.

+[SWS_EthSwt_xxxxx] When calling the function EthSwt_SetSwitchPortMode(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.

+[SWS_EthSwt_xxxxxx] If EthSwtPort does not references an EthTrcv, EthSwt shall indicate a mode of the port by the API EthIf_SwitchPortModeIndication latest during the next EthSwt_MainFunction.(SRS_ETH_00118)

~ch. 10.1.3 EthSwtDemEventParameterRefs
+ SWS Item ECUC_EthSwt_xxxxx1 : Name ETHSWT_E_SYNCPORT2PHY
Parent Container EthSwtDemEventParameterRefs
Description Reference to the DemEventParameter which shall be issued when the error "Ethernet switch port and the referenced Ethernet transceiver are in contradicting modes" has occurred.

Multiplicity 0..1
Type Symbolic name reference to [DemEventParameter]
Post-Build Variant Multiplicity true
Post-Build Variant Value true
Multiplicity Configuration
Class Pre-compile time X VARIANT-PRE-COMPILE
Link time X VARIANT-LINK-TIME
Post-build time X VARIANT-POST-BUILD
Value Configuration Class
Pre-compile time X VARIANT-PRE-COMPILE
Link time X VARIANT-LINK-TIME
Post-build time X VARIANT-POST-BUILD
Scope / Dependency
scope: local

~ch. 10.1.5 EthSwtPort
~ECUC_EthSwt_00041 EthSwtPortTrcvRef change following attributes:
Multiplicity Configuration Class | Link time | VARIANT-LINK-TIME, VARIANT-POST-BUILD
| Post build | –
add to dependency: If EthSwtPortPhysicalLayerType is defined, then EthSwtPortTrcvRef holds the reference to the corresponding EthTrcv.

~ECUC_EthSwt_00054 add dependency to EthSwtPortPhysicalLayerType: If a EthSwtPort has an EthSwtPortPhysicalLayerType then EthSwtPort shall reference an EthTrcv.

=== EthIf ===
Add to chapter 8.4 (callback notifications):
+ EthIf_SwitchPortModeIndication
Syntax: void EthIf_SwitchPortModeIndication(uint8 SwitchIdx, uint8 SwitchPortIdx, EthSwt_ModeType PortMode)
Sevice ID: pick a free one
Sync/Async: Synchronous
Reentrancy: Non Reentrant
Parmaters in: SwitchIdx: Index of the switch within the context of the Ethernet Switch Driver
SwitchPortIdx: Index of the port at the addressed switch
PortMode: notified Ethernet Switch port mode.
Description: The EthIf shall determine the expected notifications based on the EthSwtPort configuration. In case the EthSwtPort references an EthTrcv the EthIf expects a notification from the EthTrcv via API EthIf_TrcvModeIndication(). Otherwise the EthIf expects a notification from the EthSwt via API

EthIf_SwitchPortModeIndication().

~SWS_EthIf_00232
change from
Called asynchronously when mode has been read out. Triggered by previous EthTrcv_SetTransceiverMode call. Can directly be called within the trigger functions.
To
Called asynchronously when a mode change has been read out. If the function is triggered by previous call of
EthTrcv_SetTransceiverMode it can directly be called within the trigger function.
–Last change on issue 77250 comment 50–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.6 Specification Item ECUC_EthSwt_00047

Trace References:

none

Content:

Name	EthSwtPortEnableLinkDownCallbackEthSwtPort.EthSwtPortEnableLinkDownCallback		
Parent Container	EthSwtPort		
Description	Enables the callback <User>_LinkDown for this EthSwtPort if an IEEE802.1X link loss is detected. <User> is defined by EthSwtLinkDownUser. Tags: atp.Status=obsolete atp.StatusRevisionBegin=4.3.1		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76531: [EthSwt] Syntax for callback functions <user>_LinkUp and <user>_LinkDown might be wrong

  Problem description:

  The callback function for <User>_LinkUp and <User>_LinkDown to inform the user about a Link state change might be wrong
  Currently the parameter of these both functions are defined as Pointer to uint8
  This makes no sense for me to use here pointers and not a uint 8 variable.

  Agreed solution:

  Chapter 8.6.3 Configurable Interfaces
  add following intro:
  In this chapter all interfaces are listed where the target function could be configured. The names of these kind of interfaces are not fixed because they are configurable.

  Add Subchapter with interface
  move SWS_EthSwt_00117 here and change to:
  Service name: <EthSwtLinkDownCallout>
  syntax: void <EthSwtLinkDownCallout>(uint8 SwitchIdx, uint8 PortIdx)
  Service ID: 0x19
  Sync/Async:Synchronous
  Reentrancy: Non Reentrant
  Parameters (in): SwitchIdx: Index of the switch within the context of the Ethernet Switch Driver
  PortIdx: Index of the port at the addressed switch
  Parameters (in/out): none
  Parameters (out): none
  Return value: none
  Description: Is called, if a link which is configured goes down.

  ~[SWS_EthSwt_00118]The function <EthSwtLinkDownCallout> shall be called if a link, which is configured, goes down. The function provides the Switch index and the Port index, such that the port which went down can be identified.

  -[SWS_EthSwt_00119]

  add second subchapter:

  move SWS_EthSwt_00203 here and change to:
  Service name: <EthSwtLinkUpCallout>
  syntax: void <EthSwtLinkUpCallout>(uint8 SwitchIdx, uint8 PortIdx)
  Service ID: 0x1a
  Sync/Async:Synchronous

Reentrancy: Non Reentrant
Parameters (in): SwitchIdx: Index of the switch within the context of the Ethernet Switch Driver
PortIdx: Index of the port at the addressed switch
Parameters (in/out): none
Parameters (out): none
Return value: none
Description: Is called, if a link which is configured goes up

~[SWS_EthSwt_00204]The function <EthSwtLinkUpCallout> shall be called if a link which is configured goes up. The function provides the Switch index and the Port index, such that the port which went up can be identified.

-[SWS_EthSwt_00205] This is no clear as it is a configurable interface now

~ECUC_EthSwt_00048 set EthSwtLinkDownUser to obsolete
+ add optional function name parameter with same attributes named EthSwtLinkDownCallout
change description to Defines the function name for <EthSwtLinkDownCallout>.

~ECUC_EthSwt_00068 set EthSwtLinkUpUser to obsolete
+add optional function name parameter with same attributes named EthSwtLinkUpCallout
change description to Defines the function name for <EthSwtLinkUpCallout>.

set ECUC_EthSwt_00047 EthSwtPortEnableLinkDownCallback to obsolete.

Header File name is defined by ECUC_EthSwt_00064 EthSwtPublicCddHeaderFile.
–Last change on issue 76531 comment 54–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.7 Specification Item ECUC_EthSwt_00048

Trace References:

none

Content:

Name	EthSwtLinkDownUserEthSwtGeneral.EthSwtLinkDownUser		
Parent Container	EthSwtGeneral		
Description	Defines the <User> function name for the <User>_LinkDown callback. Tags: atp.Status=obsolete atp.StatusRevisionBegin=4.3.1		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default value	–		
maxLength	–		
minLength	–		
regularExpression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76531: [EthSwt] Syntax for callback functions <user>_LinkUp and <user>_LinkDown might be wrong

  Problem description:

  The callback function for <User>_LinkUp and <User>_LinkDown to inform the user about a Link state change might be wrong
  Currently the parameter of these both functions are defined as Pointer to uint8
  This makes no sense for me to use here pointers and not a uint 8 variable.

  Agreed solution:

  Chapter 8.6.3 Configurable Interfaces
  add following intro:
  In this chapter all interfaces are listed where the target function could be configured.
  The names of these kind of interfaces are not fixed because they are configurable.

  Add Subchapter with interface
  move SWS_EthSwt_00117 here and change to:
  Service name: <EthSwtLinkDownCallout>
  syntax: void <EthSwtLinkDownCallout>(uint8 SwitchIdx, uint8 PortIdx)

Service ID: 0x19
Sync/Async:Synchronous
Reentrancy: Non Reentrant
Parameters (in): SwitchIdx: Index of the switch within the context of the Ethernet Switch Driver
PortIdx: Index of the port at the addressed switch
Parameters (in/out): none
Parameters (out): none
Return value: none
Description: Is called, if a link which is configured goes down.

~[SWS_EthSwt_00118]The function <EthSwtLinkDownCallout> shall be called if a link, which is configured, goes down. The function provides the Switch index and the Port index, such that the port which went down can be identified.

-[SWS_EthSwt_00119]

add second subchapter:

move SWS_EthSwt_00203 here and change to:
Service name: <EthSwtLinkUpCallout>
syntax: void <EthSwtLinkUpCallout>(uint8 SwitchIdx, uint8 PortIdx)
Service ID: 0x1a
Sync/Async:Synchronous
Reentrancy: Non Reentrant
Parameters (in): SwitchIdx: Index of the switch within the context of the Ethernet Switch Driver
PortIdx: Index of the port at the addressed switch
Parameters (in/out): none
Parameters (out): none
Return value: none
Description: Is called, if a link which is configured goes up

~[SWS_EthSwt_00204]The function <EthSwtLinkUpCallout> shall be called if a link which is configured goes up. The function provides the Switch index and the Port index, such that the port which went up can be identified.

-[SWS_EthSwt_00205] This is no clear as it is a configurable interface now

~ECUC_EthSwt_00048 set EthSwtLinkDownUser to obsolete
+ add optional function name parameter with same attributes named Eth-

SwtLinkDownCallout
change description to Defines the function name for <EthSwtLinkDownCallout>.

~ECUC_EthSwt_00068 set EthSwtLinkUpUser to obsolete
+add optional function name parameter with same attributes named EthSwtLinkUpCallout
change description to Defines the function name for <EthSwtLinkUpCallout>.

set ECUC_EthSwt_00047 EthSwtPortEnableLinkDownCallback to obsolete.

Header File name is defined by ECUC_EthSwt_00064 EthSwtPublicCddHeaderFile.
–Last change on issue 76531 comment 54–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.8 Specification Item ECUC_EthSwt_00054

Trace References:

none

Content:

Name	EthSwtPortPhysicalLayerTypeEthSwtPort.EthSwtPortPhysicalLayerType	
Parent Container	EthSwtPort	
Description	Defines the physical layer type of this EthSwtPort.	
Multiplicity	0..1	
Type	EcucEnumerationParamDef	
Range	ETHSWT_PORT_1000BASE_T EthSwtPort.EthSwtPortPhysicalLayerType.ETHSWT_PORT_1000BASE_T	Physical layer interface 1000BASE-T (1Gbit/s, 4 pairs). Used for consumer electronic.
	ETHSWT_PORT_1000BASE_T1 EthSwtPort.EthSwtPortPhysicalLayerType.ETHSWT_PORT_1000BASE_T1	Physical layer interface 1000BASE-T1 (1Gbit/s, 1 pair). Used for automotive.
	ETHSWT_PORT_100BASE_T1 EthSwtPort.EthSwtPortPhysicalLayerType.ETHSWT_PORT_100BASE_T1	Physical layer interface 100BASE-T1 (100Mbit/s, 1 pair). Used for automotive.
	ETHSWT_PORT_100BASE_TX EthSwtPort.EthSwtPortPhysicalLayerType.ETHSWT_PORT_100BASE_TX	Physical layer interface 100BASE-TX (100Mbit/s, 2 pairs). Used for consumer electronic.

Post-Build Variant Multiplicity	true		
Post-Build Variant Value	true		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	–	
Scope / Dependency	scope: ECU dependency: If a EthSwtPort has an EthSwtPortPhysicalLayerType then EthSwt Port shall reference an EthTrcv.		

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77250: [EthIf][EthSwt][EthTrcv] EthSwt_SetSwitchPortMode() broken for ports without Ethernet transceiver

  Problem description:

  SWS_EthSwt_00019 states that EthSwt_SetSwitchPortMode() shall (unconditionally) invoke EthTrcv_SetTransceiverMode() of the Ethernet Transceiver Driver.

  This is bogus for the following two reasons:
  1) a particular switch port does not necessarily have an associated Ethernet transceiver. - In that case EthSwtPortTrcvRef won't be present. This is probably the case for MAC2MAC links if EthSwtPortRole is present an either set to ETHSWT_HOST_PORT or to ETHSWT_UP_LINK_PORT.
  2) Even if the particular switch port has an associated Ethernet transceiver, SWS_EthTrcv_00043 states that the EthTrcv shall invoke EthIf_TrcvModeIndication() as a response to EthTrcv_SetTransceiverMode().

  IMHO 1) should be solved by extending SWS_EthSwt_00019 distinguishing between ports which are associated with a transceiver (i.e., EthSwtPortTrcvRef present) - in that case EthTrcv_SetTransceiverMode() should be called - and ports which are not associated with a transceiver (i.e., EthSwtPortTrcvRef is not present) - in that case something else has to happen and this needs to be specified (probably the switch port shall be internally just disabled - maybe this needs to happen in the first case as well?)

  2) leads to the following call chain: EthIf_SwitchPortGroupRequestMode() -> EthSwt_SetSwitchPortMode() -> EthTrcv_SetTransceiverMode() -> EthIf_TrcvModeIndication(). Thus the indication actually bypasses the EthSwt

which IMHO is extremely ugly. Additionally for switch ports without associated transceiver, the EthIf will never get a EthIf_TrcvModeIndication() ...

Agreed solution:

=== System Template ===
+ Add upstream mapping of EthSwtPortTrcvRef to CouplingPort.physicalLayerType

=== EthSwt ===

~ch. 7.2.4 Production Errors
move SWS_EthSwt_00113 to 7.2.5 Extended production Errors
~SWS_EthSwt_00113 change detection criteria to
Fail:When access to the Ethernet Switch fails the module shall report the extended production error with event status DEM_EVENT_STATUS_PREFAILED to DEM.
Pass:When access to the Ethernet Switch succeeds the module shall report the extended production error with event status DEM_EVENT_STATUS_PREPASSED to DEM.

+ SWS_EthSwt_xxxxx new ext. production Error
Error Name: ETHSWT_E_SYNCPORT2PHY
Short Description: Ethernet switch port and the referenced Ethernet transceiver are in contradicting modes.
Long description: While getting the Ethernet switch port mode, the Ethernet switch driver detected an inconsistent state between Ethernet switch port and the referenced Ethernet transceiver Mode.
Detection Criteria: Fail: When getting the Ethernet switch port mode together with the Ethernet transceiver mode and the mode of the two referenced modules was found
inconsistent the module shall report the extended production error with event status DEM_EVENT_STATUS_PREFAILED to DEM.
Pass:When getting the Ethernet switch port mode together with the Ethernet transceiver mode and the mode of the two referenced modules was found
consistent the module shall report the extended production error with event status DEM_EVENT_STATUS_PREPASSED to DEM.
all other attributer: N/A

~ ch. 8.3.2 EthSwt_SwitchInit
~[SWS_EthSwt_00016] change "production error" to "extended production error"

~ch. 8.3.3 EthSwt_SetSwitchPortMode
~[SWS_EthSwt_00019]The function EthSwt_SetSwitchPortMode() shall put the indexed port of the switch into the specified mode. If EthSwtPort references an

EthTrcv then the function EthTrcv_SetTransceiverMode() of the Ethernet Transceiver Driver shall additionally be called with the corresponding transceiver mode.

+[SWS_EthSwt_xxxxx] When calling the function EthSwt_SetSwitchPortMode() with mode ETHTRCV_MODE_DOWN, the EthSwt shall disable the Ethernet switch port directly for reduction of power consumption, if it is possible.

+[SWS_EthSwt_xxxxx] When calling the function EthSwt_SetSwitchPortMode(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.

+[SWS_EthSwt_xxxxxx] If EthSwtPort does not references an EthTrcv, EthSwt shall indicate a mode of the port by the API EthIf_SwitchPortModeIndication latest during the next EthSwt_MainFunction.(SRS_ETH_00118)

~ch. 10.1.3 EthSwtDemEventParameterRefs
+ SWS Item ECUC_EthSwt_xxxxx1 : Name ETHSWT_E_SYNCPORT2PHY
Parent Container EthSwtDemEventParameterRefs
Description Reference to the DemEventParameter which shall be issued when the error "Ethernet switch port and the referenced Ethernet transceiver are in contradicting modes" has occurred.
Multiplicity 0..1
Type Symbolic name reference to [DemEventParameter]
Post-Build Variant Multiplicity true
Post-Build Variant Value true
Multiplicity Configuration
Class Pre-compile time X VARIANT-PRE-COMPILE
Link time X VARIANT-LINK-TIME
Post-build time X VARIANT-POST-BUILD
Value Configuration Class
Pre-compile time X VARIANT-PRE-COMPILE
Link time X VARIANT-LINK-TIME
Post-build time X VARIANT-POST-BUILD
Scope / Dependency
scope: local

~ch. 10.1.5 EthSwtPort
~ECUC_EthSwt_00041 EthSwtPortTrcvRef change following attributes:
Multiplicity Configuration Class | Link time | VARIANT-LINK-TIME, VARIANT-POST-BUILD

| Post build | –
add to dependency: If EthSwtPortPhysicalLayerType is defined, then EthSwtPortTrcvRef holds the reference to the corresponding EthTrcv.

~ECUC_EthSwt_00054 add dependency to EthSwtPortPhysicalLayerType: If a EthSwtPort has an EthSwtPortPhysicalLayerType then EthSwtPort shall reference an EthTrcv.

=== EthIf ===
Add to chapter 8.4 (callback notifications):
+ EthIf_SwitchPortModeIndication
Syntax: void EthIf_SwitchPortModeIndication(uint8 SwitchIdx, uint8 SwitchPortIdx, EthSwt_ModeType PortMode)
Sevice ID: pick a free one
Sync/Async: Synchronous
Reentrancy: Non Reentrant
Parmaters in: SwitchIdx: Index of the switch within the context of the Ethernet Switch Driver
SwitchPortIdx: Index of the port at the addressed switch
PortMode: notified Ethernet Switch port mode.
Description: The EthIf shall determine the expected notifications based on the EthSwtPort configuration. In case the EthSwtPort references an EthTrcv the EthIf expects a notification from the EthTrcv via API EthIf_TrcvModeIndication(). Otherwise the EthIf expects a notification from the EthSwt via API EthIf_SwitchPortModeIndication().

~SWS_EthIf_00232
change from
Called asynchronously when mode has been read out. Triggered by previous
EthTrcv_SetTransceiverMode call. Can directly be called within the trigger
functions.
To
Called asynchronously when a mode change has been read out. If the function is triggered by previous call of
EthTrcv_SetTransceiverMode it can directly be called within the trigger function.
–Last change on issue 77250 comment 50–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.9 Specification Item ECUC_EthSwt_00068

Trace References:

none

Content:

Name	EthSwtLinkUpUserEthSwtGeneral.EthSwtLinkUpUser		
Parent Container	EthSwtGeneral		
Description	Defines the <User> function name for the <User>_LinkUp callback. Tags: atp.Status=obsolete atp.StatusRevisionBegin=4.3.1		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default value	–		
maxLength	–		
minLength	–		
regularExpression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76531: [EthSwt] Syntax for callback functions <user>_LinkUp and <user>_LinkDown might be wrong

  Problem description:

  The callback function for <User>_LinkUp and <User>_LinkDown to inform the user about a Link state change might be wrong
  Currently the parameter of these both functions are defined as Pointer to uint8
  This makes no sense for me to use here pointers and not a uint 8 variable.

  Agreed solution:

  Chapter 8.6.3 Configurable Interfaces
  add following intro:
  In this chapter all interfaces are listed where the target function could be configured.
  The names of these kind of interfaces are not fixed because they are configurable.

Add Subchapter with interface
move SWS_EthSwt_00117 here and change to:
Service name: <EthSwtLinkDownCallout>
syntax: void <EthSwtLinkDownCallout>(uint8 SwitchIdx, uint8 PortIdx)
Service ID: 0x19
Sync/Async:Synchronous
Reentrancy: Non Reentrant
Parameters (in): SwitchIdx: Index of the switch within the context of the Ethernet Switch Driver
PortIdx: Index of the port at the addressed switch
Parameters (in/out): none
Parameters (out): none
Return value: none
Description: Is called, if a link which is configured goes down.

~[SWS_EthSwt_00118]The function <EthSwtLinkDownCallout> shall be called if a link, which is configured, goes down. The function provides the Switch index and the Port index, such that the port which went down can be identified.

-[SWS_EthSwt_00119]

add second subchapter:

move SWS_EthSwt_00203 here and change to:
Service name: <EthSwtLinkUpCallout>
syntax: void <EthSwtLinkUpCallout>(uint8 SwitchIdx, uint8 PortIdx)
Service ID: 0x1a
Sync/Async:Synchronous
Reentrancy: Non Reentrant
Parameters (in): SwitchIdx: Index of the switch within the context of the Ethernet Switch Driver
PortIdx: Index of the port at the addressed switch
Parameters (in/out): none
Parameters (out): none
Return value: none
Description: Is called, if a link which is configured goes up

~[SWS_EthSwt_00204]The function <EthSwtLinkUpCallout> shall be called if a link which is configured goes up. The function provides the Switch index and the Port index, such that the port which went up can be identified.

-[SWS_EthSwt_00205] This is no clear as it is a configurable interface now

~ECUC_EthSwt_00048 set EthSwtLinkDownUser to obsolete
+ add optional function name parameter with same attributes named EthSwtLinkDownCallout
change description to Defines the function name for <EthSwtLinkDownCallout>.

~ECUC_EthSwt_00068 set EthSwtLinkUpUser to obsolete
+add optional function name parameter with same attributes named EthSwtLinkUpCallout
change description to Defines the function name for <EthSwtLinkUpCallout>.

set ECUC_EthSwt_00047 EthSwtPortEnableLinkDownCallback to obsolete.

Header File name is defined by ECUC_EthSwt_00064 EthSwtPublicCddHeaderFile.
–Last change on issue 76531 comment 54–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.10 Specification Item ECUC_EthSwt_00093

Trace References:

none

Content:

Name	EthSwtGetCfgHexDumpApiEthSwtGeneral.EthSwtGetCfgHexDumpApi		
Parent Container	EthSwtGeneral		
Description	Enables / Disables EthSwt_GetCfgHexDump API Tags: atp.Status=obsolete atp.StatusRevisionBegin=4.3.1		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77018: [EthSwt] Clarification about the use case regarding API "EthSwt_GetCfgHexDump"

  Problem description:

  We discussed the use case regarding the API "EthSwt_GetCfgHexDump". Daimler want to read out the register memory of an Ethernet switch as a memory dump. The memory dump is transmitted via an diagnostic service over the network consecutively to an diagnostic tester. The memory dump is used to verify and compare to the expected / required values. It is not clear if the API is sufficient for this use case.

  Agreed solution:

  Capter 7.1.2.13 exchange in listing of APIS EthSwt_GetCfgHexDump with EthSwt_GetCfgDataRaw and EthSwt_GetCfgHexDumpLength with EthSwt_GetCfgDataInfo.

  Chapter 8
  -SWS_EthSwt_91026,
  -SWS_EthSwt_00352
  -SWS_EthSwt_00353
  -SWS_EthSwt_00355
  -ECUC_EthSwt_00093
  -SWS_EthSwt_91027
  -SWS_EthSwt_00356
  -SWS_EthSwt_00357
  -SWS_EthSwt_00358
  -SWS_EthSwt_00359
  Set APIs EthSwt_GetCfgHexDump and EthSwt_GetCfgHexDumpLength and config parameters EthSwtGetCfgHexDumpApi and EthSwt_GetCfgHexDumpLengthApi to deprecated or delete.

  Chapter 10
  Add parameter to enable disable APIs with name EthSwtGetCfgRaw to EthSwtGeneral-Container
  Description Disable /Enable support of reading raw data from switch memory.
  Multiplicity: 1
  Type: EcucBooleanParamDef
  Default: false
  Post build variant false
  Config class Pre-compile time for all variants.
  scope: local

Introduce following new APIs to function: definitions
EthSwt_GetCfgDataRaw(
uint8 SwitchIdx,
uint32 Offset,
uint16 Length,
uint8 *BufferPtr
)
[attributes
- asynchronous
- non-reentrant
parameters IN: SwitchIdx:Index of the Ethernet switch within the context of the Ethernet Switch Driver
Offset: Offset of the Ethernet switch memory from where the reading starts
Length: Length of data in bytes that shall be copied.
parameters OUT: BufferPtr: Pointer to the location where the data shall be copied.
Std_ReturnType: E_OK: the data read was triggered successfully. E_NOT_OK: the data read was not triggered successfully. (i.e. indexed Ethernet switch is not available)
Description:
Retrieves the data in memory of the indexed Ethernet switch in variable length.]

+[SWS_EthSwt_xxxxx1] The function EthSwt_GetCfgDataRaw() shall only be available if parameter EthSwtGetCfgRaw is set to TRUE.(SRS_BSW_00171)
+[SWS_EthSwt_xxxxx2] When calling the function EthSwt_GetCfgDataRaw(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.()

EthSwt_GetCfgDataInfo(
uint8 SwitchIdx,
uint32 *DataSizePtr,
uint32 *DataAdressPtr
)
[attributes
- synchronous
- reentrant
parameters IN: SwitchIdx:Index of the Ethernet switch within the context of the Ethernet Switch Driver
parameters OUT: DataSizePtr: Pointer to the location where the total size of the configuration data shall be copied.

DataAddressPtr: Pointer to the location where the start address of the configuration registers shall be copied.
Std_ReturnType: E_OK: the data was obtained successfully. E_NOT_OK: the data was not obtained successfully. (i.e. indexed Ethernet switch is not available)
Description:
Retrieves the total size of data and the memory start address of the indexed Ethernet Switch.]

+[SWS_EthSwt_xxxxx3] The function EthSwt_GetCfgDataInfo() shall only be available if parameter EthSwtGetCfgRaw is set to TRUE.(SRS_BSW_00171)
+[SWS_EthSwt_xxxxx4] When calling the function EthSwt_GetCfgDataInfo(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.()

add following API to chapter "8.6.3 Configurable interfaces"
<GetCfgDataRawDone>(
uint8 SwitchIdx
)
[attributes:
synchronous
reentrant
parameters IN: SwitchIdx: Index of the Ethernet switch where the Configuration is read.
Return value : void
Description: The call of the function EthSwt_GetCfgDataRaw() triggers a asynchrony read of a certain memory section of the Ethernet switch driver. If the read is done, the configured callout function <GetCfgDataRawDone> shall be called]

Add a parameter to EthSwtGeneral-Container
GetCfgDataRawDone
Decription: Defines the function name for <GetCfgDataRawDone>
Multplicity: 0 ..1
Type EcucFuntionNameDef
Variant and Variant Multiplicity false
Config class Pre-compile for all variants
scope: local
dependency: The function GetCfgDataRawDone shall only be configured if parameter EthSwtGetCfgRaw is set to TRUE.

Header file name parameter is defined in ECUC_EthSwt_00064
–Last change on issue 77018 comment 32–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.11 Specification Item ECUC_EthSwt_00094

Trace References:

none

Content:

Name	EthSwtGetCfgHexDumpLengthApiEthSwtGeneral.EthSwtGetCfgHexDumpLengthApi		
Parent Container	EthSwtGeneral		
Description	Enables / Disables EthSwt_GetCfgHexDumpLength API Tags: atp.Status=obsolete atp.StatusRevisionBegin=4.3.1		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77018: [EthSwt] Clarification about the use case regarding API "EthSwt_GetCfgHexDump"

  Problem description:

  We discussed the use case regarding the API "EthSwt_GetCfgHexDump". Daimler want to read out the register memory of an Ethernet switch as a memory dump. The memory dump is transmitted via an diagnostic service over the network consecutively to an diagnostic tester. The memory dump is used to verify and compare to the expected / required values. It is not clear if the API is sufficient for this use case.

  Agreed solution:

Capter 7.1.2.13 exchange in listing of APIS EthSwt_GetCfgHexDump with EthSwt_GetCfgDataRaw and EthSwt_GetCfgHexDumpLength with EthSwt_GetCfgDataInfo.

Chapter 8
-SWS_EthSwt_91026,
-SWS_EthSwt_00352
-SWS_EthSwt_00353
-SWS_EthSwt_00355
-ECUC_EthSwt_00093
-SWS_EthSwt_91027
-SWS_EthSwt_00356
-SWS_EthSwt_00357
-SWS_EthSwt_00358
-SWS_EthSwt_00359
Set APIs EthSwt_GetCfgHexDump and EthSwt_GetCfgHexDumpLength and config parameters EthSwtGetCfgHexDumpApi and EthSwt_GetCfgHexDumpLengthApi to deprecated or delete.

Chapter 10
Add parameter to enable disable APIs with name EthSwtGetCfgRaw to EthSwtGeneral-Container
Description Disable /Enable support of reading raw data from switch memory.
Multiplicity: 1
Type: EcucBooleanParamDef
Default: false
Post build variant false
Config class Pre-compile time for all variants.
scope: local

Introduce following new APIs to function: definitions
EthSwt_GetCfgDataRaw(
uint8 SwitchIdx,
uint32 Offset,
uint16 Length,
uint8 *BufferPtr
)
[attributes
- asynchronous
- non-reentrant
parameters IN: SwitchIdx:Index of the Ethernet switch within the context of the Ethernet Switch Driver

Offset: Offset of the Ethernet switch memory from where the reading starts
Length: Length of data in bytes that shall be copied.
parameters OUT: BufferPtr: Pointer to the location where the data shall be copied.
Std_ReturnType: E_OK: the data read was triggered successfully. E_NOT_OK: the data read was not triggered successfully. (i.e. indexed Ethernet switch is not available)
Description:
Retrieves the data in memory of the indexed Ethernet switch in variable length.]

+[SWS_EthSwt_xxxxx1] The function EthSwt_GetCfgDataRaw() shall only be available if parameter EthSwtGetCfgRaw is set to TRUE.(SRS_BSW_00171)
+[SWS_EthSwt_xxxxx2] When calling the function EthSwt_GetCfgDataRaw(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.()

EthSwt_GetCfgDataInfo(
uint8 SwitchIdx,
uint32 *DataSizePtr,
uint32 *DataAdressPtr
)
[attributes
- synchronous
- reentrant
parameters IN: SwitchIdx:Index of the Ethernet switch within the context of the Ethernet Switch Driver
parameters OUT: DataSizePtr: Pointer to the location where the total size of the configuration data shall be copied.
DataAddressPtr: Pointer to the location where the start address of the configuration registers shall be copied.
Std_ReturnType: E_OK: the data was obtained successfully. E_NOT_OK: the data was not obtained successfully. (i.e. indexed Ethernet switch is not available)
Description:
Retrieves the total size of data and the memory start address of the indexed Ethernet Switch.]

+[SWS_EthSwt_xxxxx3] The function EthSwt_GetCfgDataInfo() shall only be available if parameter EthSwtGetCfgRaw is set to TRUE.(SRS_BSW_00171)
+[SWS_EthSwt_xxxxx4] When calling the function EthSwt_GetCfgDataInfo(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return

E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.()

add following API to chapter "8.6.3 Configurable interfaces"
<GetCfgDataRawDone>(
uint8 SwitchIdx
)
[attributes:
synchronous
reentrant
parameters IN: SwitchIdx: Index of the Ethernet switch where the Configuration is read.
Return value : void
Description: The call of the function EthSwt_GetCfgDataRaw() triggers a asynchrony read of a certain memory section of the Ethernet switch driver. If the read is done, the configured callout function <GetCfgDataRawDone> shall be called]

Add a parameter to EthSwtGeneral-Container
GetCfgDataRawDone
Decription: Defines the function name for <GetCfgDataRawDone>
Multplicity: 0 ..1
Type EcucFuntionNameDef
Variant and Variant Multiplicity false
Config class Pre-compile for all variants
scope: local
dependency: The function GetCfgDataRawDone shall only be configured if parameter EthSwtGetCfgRaw is set to TRUE.

Header file name parameter is defined in ECUC_EthSwt_00064
–Last change on issue 77018 comment 32–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.12 Specification Item ECUC_EthSwt_00113

Trace References:

none

Content:

Name	EthSwtPortMacLayerSubTypeEthSwtPort.EthSwtPortMacLayerSubType		
Description	Defines the MAC layer subtype of this EthSwtPort.		
Multiplicity	0..1		
Type	EcucEnumerationParamDef		
Range	REDUCEDEthSwtPort.EthSwtPortMacLayerSubType.REDUCED	Reduced media-independent interface	
	REVERSEDEthSwtPort.EthSwtPortMacLayerSubType.REVERSED	reversed media-independent interface (to provide direct connection between two Ethernet MACs)	
	SERIALEthSwtPort.EthSwtPortMacLayerSubType.SERIAL	low-power and low pin-count serial 8b/10b-coded media-independent interface	
	STANDARDEthSwtPort.EthSwtPortMacLayerSubType.STANDARD	standard media-independent interface	
	UNIVERSAL_SERIALEthSwtPort.EthSwtPortMacLayerSubType.UNIVERSAL_SERIAL	Universal low-power and low pin-count serial 8b/10b-coded media-independent interface	
Post-Build Variant Multiplicity	true		
Post-Build Variant Value	true		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	–	
Scope / Dependency	scope: ECU		

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76343: Configuration parameters for Ethernet MAC layer types is incomplete

  Problem description:

  With https://www.autosar.org/bugzilla/show_bug.cgi?id=73074 the configuration of the MAC layer type was separated from the physical layer type. The configuration parameter EthSwtPortMacLayerType allows for the configuration of "families" of MAC layer types, namely xMII, xGMII, and xxGMII.

The concrete MAC layer type (e.g., SGMII vs. RGMII) however cannot be configure by this parameter. - Available switches (e.g., Broadcom's "Leo") however require the configuration of a *concrete* MAC layer type (and not only the configuration of a "family" of MAC layer types) for the individual port.

The same holds probably true for the Ethernet Driver and the Ethernet Transceiver driver.

One possibility to solve this is to introduce an additional config parameter named something like EthSwtPortMacLayerSubType with the following enum values "standard", "reduced", "reversed", "serial", "universal serial". - Thus for example the combination of EthSwtPortMacLayerType=xGMII and EthSwtPortMacLayerSubType="reversed" would yield RvGMII.

Agreed solution:

EthSwt:
add config parameter to container EthSwtPort named EthSwtPortMacLayerSubType with the following enum values "STANDARD: standard media-independent interface", "REDUCED: Reduced media-independent interface", "REVERSED: reversed media-independent interface (to provide direct connection between two Ethernet MACs)", "SERIAL: low-power and low pin-count serial 8b/10b-coded media-independent interface", "UNIVERSAL SERIAL": Universal low-power and low pin-count serial 8b/10b-coded media-independent interface".

add config parameter to container EthSwtPort named EthSwtPortMacLayerSpeed with the following enum values "ETH_MAC_LAYER_SPEED_10M", "ETH_MAC_LAYER_SPEED_100M", "ETH_MAC_LAYER_SPEED_1G", "ETH_MAC_LAYER_SPEED_10G".
All parameters shall have the same Multplicity, Variant and config class as EthSwtPortMacLayerType.

add description for EthSwt_PortMacLayerSubTypes:
Defines the MAC layer subtype of a switch port

add description for EthSwt_PortMacLayerSpeed:
Defines the baud rate of the MAC layer

Eth:
add config parameter to container EthCtrlConfig named EthCtrlMacLayerSub-

Type with the following enum values "STANDARD", "REDUCED", "REVERSED", "SERIAL", "UNIVERSAL SERIAL". - Thus for example the combination of EthCtrlMacLayerType=xGMII and EthCtrlMacLayerSubType ="REVERSED" would yield RvGMII.
add config parameter to container EthCtrlConfig named EthCtrlMacLayerSpeed with the following enum values "ETH_MAC_LAYER_SPEED_10M", "ETH_MAC_LAYER_SPEED_100M", "ETH_MAC_LAYER_SPEED_1G", "ETH_MAC_LAYER_SPEED_10G".

add description for Eth_PortMacLayerSubTypes:
Defines the MAC layer subtype of a switch port

add description for Eth_PortMacLayerSpeed:
Defines the baud rate of the MAC layer

EthTrcv:
add config parameter to container EthTrcvCtrlConfig named EthTrcvPortMacLayerSubType with the following enum values "STANDARD", "REDUCED", "REVERSED", "SERIAL", "UNIVERSAL SERIAL". - Thus for example the combination of EthTrcvPortMacLayerType=xGMII and EthTrcvPortMacLayerSubType="REVERSED" would yield RvGMII.
add config parameter to container EthTrcvCtrlConfig named EthTrcvPortMacLayerSpeed with the following enum values "ETH_MAC_LAYER_SPEED_10M", "ETH_MAC_LAYER_SPEED_100M", "ETH_MAC_LAYER_SPEED_1G", "ETH_MAC_LAYER_SPEED_10G".

add description for EthTrcv_PortMacLayerSubTypes:
Defines the MAC layer subtype of a switch port

add description for EthTrcv_PortMacLayerSpeed:
Defines the baud rate of the MAC layer

All parameters shall have the same Multplicity, Variant and config class as EthSwtPortMacLayerType.

No upstream Mapping as this decision is made on ECU-Configuration.
–Last change on issue 76343 comment 30–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.13 Specification Item ECUC_EthSwt_00114

Trace References:

none

Content:

Name	EthSwtPortMacLayerSpeedEthSwtPort.EthSwtPortMacLayerSpeedin container EthSwtPort		
Description	Defines the baud rate of the MAC layer.		
Multiplicity	0..1		
Type	EcucEnumerationParamDef		
Range	ETH_MAC_LAYER_SPEED_100M	–	EthSwtPort.EthSwtPortMacLayerSpeed.ETH_MAC_LAYER_SPEED_100M
	ETH_MAC_LAYER_SPEED_10G	–	EthSwtPort.EthSwtPortMacLayerSpeed.ETH_MAC_LAYER_SPEED_10G
	ETH_MAC_LAYER_SPEED_10M	–	EthSwtPort.EthSwtPortMacLayerSpeed.ETH_MAC_LAYER_SPEED_10M
	ETH_MAC_LAYER_SPEED_1G	–	EthSwtPort.EthSwtPortMacLayerSpeed.ETH_MAC_LAYER_SPEED_1G
Post-Build Variant Multiplicity	true		
Post-Build Variant Value	true		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	–	
Scope / Dependency	scope: ECU		

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76343: Configuration parameters for Ethernet MAC layer types is incomplete

  Problem description:

  With https://www.autosar.org/bugzilla/show_bug.cgi?id=73074 the configuration of the MAC layer type was separated from the physical layer type. The configuration parameter EthSwtPortMacLayerType allows for the configuration of "families" of MAC layer types, namely xMII, xGMII, and xxGMII.

  The concrete MAC layer type (e.g., SGMII vs. RGMII) however cannot be configure by this parameter. - Available switches (e.g., Broadcom's "Leo") however require the configuration of a *concrete* MAC layer type (and not only the configuration of a "family" of MAC layer types) for the individual port.

  The same holds probably true for the Ethernet Driver and the Ethernet Transceiver driver.

  One possibility to solve this is to introduce an additional config parameter named something like EthSwtPortMacLayerSubType with the following enum values "standard", "reduced", "reversed", "serial", "universal serial". - Thus for example the combination of EthSwtPortMacLayerType=xGMII and EthSwtPortMacLayerSubType="reversed" would yield RvGMII.

  Agreed solution:

  EthSwt:
  add config parameter to container EthSwtPort named EthSwtPortMacLayerSubType with the following enum values "STANDARD: standard media-independent interface", "REDUCED: Reduced media-independent interface", "REVERSED: reversed media-independent interface (to provide direct connection between two Ethernet MACs)", "SERIAL: low-power and low pin-count serial 8b/10b-coded media-independent interface", "UNIVERSAL SERIAL": Universal low-power and low pin-count serial 8b/10b-coded media-independent interface".

  add config parameter to container EthSwtPort named EthSwtPortMacLayerSpeed with the following enum values "ETH_MAC_LAYER_SPEED_10M", "ETH_MAC_LAYER_SPEED_100M", "ETH_MAC_LAYER_SPEED_1G", "ETH_MAC_LAYER_SPEED_10G".
  All parameters shall have the same Multplicity, Variant and config class as EthSwtPortMacLayerType.

  add description for EthSwt_PortMacLayerSubTypes:
  Defines the MAC layer subtype of a switch port

add description for EthSwt_PortMacLayerSpeed:
Defines the baud rate of the MAC layer

Eth:
add config parameter to container EthCtrlConfig named EthCtrlMacLayerSubType with the following enum values "STANDARD", "REDUCED", "REVERSED", "SERIAL", "UNIVERSAL SERIAL". - Thus for example the combination of EthCtrlMacLayerType=xGMII and EthCtrlMacLayerSubType ="REVERSED" would yield RvGMII.
add config parameter to container EthCtrlConfig named EthCtrlMacLayerSpeed with the following enum values "ETH_MAC_LAYER_SPEED_10M", "ETH_MAC_LAYER_SPEED_100M", "ETH_MAC_LAYER_SPEED_1G", "ETH_MAC_LAYER_SPEED_10G".

add description for Eth_PortMacLayerSubTypes:
Defines the MAC layer subtype of a switch port

add description for Eth_PortMacLayerSpeed:
Defines the baud rate of the MAC layer

EthTrcv:
add config parameter to container EthTrcvCtrlConfig named EthTrcvPortMacLayerSubType with the following enum values "STANDARD", "REDUCED", "REVERSED", "SERIAL", "UNIVERSAL SERIAL". - Thus for example the combination of EthTrcvPortMacLayerType=xGMII and EthTrcvPortMacLayerSubType="REVERSED" would yield RvGMII.
add config parameter to container EthTrcvCtrlConfig named EthTrcvPortMacLayerSpeed with the following enum values "ETH_MAC_LAYER_SPEED_10M", "ETH_MAC_LAYER_SPEED_100M", "ETH_MAC_LAYER_SPEED_1G", "ETH_MAC_LAYER_SPEED_10G".

add description for EthTrcv_PortMacLayerSubTypes:
Defines the MAC layer subtype of a switch port

add description for EthTrcv_PortMacLayerSpeed:
Defines the baud rate of the MAC layer

All parameters shall have the same Multplicity, Variant and config class as EthSwtPortMacLayerType.

No upstream Mapping as this decision is made on ECU-Configuration.
–Last change on issue 76343 comment 30–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.14 Specification Item ECUC_EthSwt_00115

Trace References:

none

Content:

Name	EthSwtLinkDownCalloutEthSwtGeneral.EthSwtLinkDownCallout		
Parent Container	EthSwtGeneral		
Description	Defines the function name for the <EthSwtLinkDownCallout> callout.		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default value	–		
maxLength	–		
minLength	–		
regularExpression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76531: [EthSwt] Syntax for callback functions <user>_LinkUp and <user>_LinkDown might be wrong

  Problem description:

  The callback function for <User>_LinkUp and <User>_LinkDown to inform the user about a Link state change might be wrong
  Currently the parameter of these both functions are defined as Pointer to uint8
  This makes no sense for me to use here pointers and not a uint 8 variable.

  Agreed solution:

  Chapter 8.6.3 Configurable Interfaces
  add following intro:
  In this chapter all interfaces are listed where the target function could be configured.
  The names of these kind of interfaces are not fixed because they are configurable.

  Add Subchapter with interface
  move SWS_EthSwt_00117 here and change to:
  Service name: <EthSwtLinkDownCallout>
  syntax: void <EthSwtLinkDownCallout>(uint8 SwitchIdx, uint8 PortIdx)
  Service ID: 0x19
  Sync/Async:Synchronous
  Reentrancy: Non Reentrant
  Parameters (in): SwitchIdx: Index of the switch within the context of the Ethernet Switch Driver
  PortIdx: Index of the port at the addressed switch
  Parameters (in/out): none
  Parameters (out): none
  Return value: none
  Description: Is called, if a link which is configured goes down.

  ~[SWS_EthSwt_00118]The function <EthSwtLinkDownCallout> shall be called if a link, which is configured, goes down. The function provides the Switch index and the Port index, such that the port which went down can be identified.

  -[SWS_EthSwt_00119]

  add second subchapter:

  move SWS_EthSwt_00203 here and change to:
  Service name: <EthSwtLinkUpCallout>
  syntax: void <EthSwtLinkUpCallout>(uint8 SwitchIdx, uint8 PortIdx)
  Service ID: 0x1a
  Sync/Async:Synchronous

Reentrancy: Non Reentrant
Parameters (in): SwitchIdx: Index of the switch within the context of the Ethernet Switch Driver
PortIdx: Index of the port at the addressed switch
Parameters (in/out): none
Parameters (out): none
Return value: none
Description: Is called, if a link which is configured goes up

~[SWS_EthSwt_00204]The function <EthSwtLinkUpCallout> shall be called if a link which is configured goes up. The function provides the Switch index and the Port index, such that the port which went up can be identified.

-[SWS_EthSwt_00205] This is no clear as it is a configurable interface now

~ECUC_EthSwt_00048 set EthSwtLinkDownUser to obsolete
+ add optional function name parameter with same attributes named EthSwtLinkDownCallout
change description to Defines the function name for <EthSwtLinkDownCallout>.

~ECUC_EthSwt_00068 set EthSwtLinkUpUser to obsolete
+add optional function name parameter with same attributes named EthSwtLinkUpCallout
change description to Defines the function name for <EthSwtLinkUpCallout>.

set ECUC_EthSwt_00047 EthSwtPortEnableLinkDownCallback to obsolete.

Header File name is defined by ECUC_EthSwt_00064 EthSwtPublicCddHeaderFile.
–Last change on issue 76531 comment 54–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.15 Specification Item ECUC_EthSwt_00116

Trace References:

none

Content:

Name	EthSwtLinkUpCalloutEthSwtGeneral.EthSwtLinkUpCallout		
Parent Container	EthSwtGeneral		
Description	Defines the function name for the <EthSwtLinkUpCallout> callout.		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default value	–		
maxLength	–		
minLength	–		
regularExpression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76531: [EthSwt] Syntax for callback functions <user>_LinkUp and <user>_LinkDown might be wrong

  Problem description:

  The callback function for <User>_LinkUp and <User>_LinkDown to inform the user about a Link state change might be wrong
  Currently the parameter of these both functions are defined as Pointer to uint8
  This makes no sense for me to use here pointers and not a uint 8 variable.

  Agreed solution:

  Chapter 8.6.3 Configurable Interfaces
  add following intro:
  In this chapter all interfaces are listed where the target function could be configured.
  The names of these kind of interfaces are not fixed because they are configurable.

Add Subchapter with interface
move SWS_EthSwt_00117 here and change to:
Service name: <EthSwtLinkDownCallout>
syntax: void <EthSwtLinkDownCallout>(uint8 SwitchIdx, uint8 PortIdx)
Service ID: 0x19
Sync/Async:Synchronous
Reentrancy: Non Reentrant
Parameters (in): SwitchIdx: Index of the switch within the context of the Ethernet Switch Driver
PortIdx: Index of the port at the addressed switch
Parameters (in/out): none
Parameters (out): none
Return value: none
Description: Is called, if a link which is configured goes down.

~[SWS_EthSwt_00118]The function <EthSwtLinkDownCallout> shall be called if a link, which is configured, goes down. The function provides the Switch index and the Port index, such that the port which went down can be identified.

-[SWS_EthSwt_00119]

add second subchapter:

move SWS_EthSwt_00203 here and change to:
Service name: <EthSwtLinkUpCallout>
syntax: void <EthSwtLinkUpCallout>(uint8 SwitchIdx, uint8 PortIdx)
Service ID: 0x1a
Sync/Async:Synchronous
Reentrancy: Non Reentrant
Parameters (in): SwitchIdx: Index of the switch within the context of the Ethernet Switch Driver
PortIdx: Index of the port at the addressed switch
Parameters (in/out): none
Parameters (out): none
Return value: none
Description: Is called, if a link which is configured goes up

~[SWS_EthSwt_00204]The function <EthSwtLinkUpCallout> shall be called if a link which is configured goes up. The function provides the Switch index and the Port index, such that the port which went up can be identified.

-[SWS_EthSwt_00205] This is no clear as it is a configurable interface now

~ECUC_EthSwt_00048 set EthSwtLinkDownUser to obsolete
+ add optional function name parameter with same attributes named EthSwtLinkDownCallout
change description to Defines the function name for <EthSwtLinkDownCallout>.

~ECUC_EthSwt_00068 set EthSwtLinkUpUser to obsolete
+add optional function name parameter with same attributes named EthSwtLinkUpCallout
change description to Defines the function name for <EthSwtLinkUpCallout>.

set ECUC_EthSwt_00047 EthSwtPortEnableLinkDownCallback to obsolete.

Header File name is defined by ECUC_EthSwt_00064 EthSwtPublicCddHeaderFile.
–Last change on issue 76531 comment 54–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.16 Specification Item ECUC_EthSwt_00117

Trace References:

none

Content:

Name	EthSwtSetSwitchPortModeApiEthSwtGeneral.EthSwtSetSwitchPortModeApi
Parent Container	EthSwtGeneral
Description	Enables / Disables EthSwt_SetSwitchPortMode API
Multiplicity	1
Type	EcucBooleanParamDef
Default value	–
Post-Build Variant Value	false

Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77043: [EthSwt] EthSwt APIs shouldnt be precompile ON/OFF based on EthTrcv parameters

  Problem description:

  For the APIs EthSwt_SetSwitchPortMode, EthSwt_GetSwitchPortMode, EthSwt_StartSwitchPortAutoNegotiation, EthSwt_GetLinkState, EthSwt_GetBaudRate and EthSwt_GetDuplexMode there is a requirement that states that the API should be configurable On/Off based on a EthTrcv parameter. For example for EthSwt_StartSwitchPortAutoNegotiation the requirement(SWS_EthSwt_00035) is: The function EthSwt_StartSwitchPortAutoNegotiation shall be pre-compile time configurable On/Off by the configuration parameter: EthTrcvStartAutoNegotiationApi.

  However, for other similar APIs a configuration parameter was defined in the switch driver. The functions are: EthSwt_GetPortSignalQuality, EthSwt_GetPortIdentifier, EthSwt_SetPortLoopbackMode, EthSwt_SetPortTxMode, EthSwt_GetPortCableDiagnosticsResult.

  Also, if multiple EthTrcv drivers are used, how will the precompile parameters be defined in the switch?

  Agreed solution:

  add parameters
  EthSwtSetSwitchPortModeApi, EthSwtGetSwitchPortModeApi, EthSwtStartSwitchPortAutoNegotiationApi, EthSwtGetLinkStateApi, EthSwtGetBaudRateApi and EthSwtGetDuplexModeApi
  Description: Enables / disables the EthSwt_SetSwitchPortMode / EthSwt_GetSwitchPortMode / EthSwt_StartSwitchPortAutoNegotiation / EthSwt_GetLinkState/ EthSwt_GetBaudRate / EthSwt_GetDuplexMode
  Multiplicity: 1
  Type EcucBooleanParamDef
  Default: False
  Variant false

Config Class Pre-compile for all variants

~[SWS_EthSwt_00022]
replace EthTrcvSetTransceiverModeApi with EthSwtSetSwitchPortModeApi.
~SWS_EthSwt_00029
replace EthTrcvGetTransceiverModeApi with EthSwtGetSwitchPortModeApi.
~SWS_EthSwt_00035
replace EthTrcvStartAutoNegotiationApi with EthSwtStartSwitchPortAutoNegotiationApi
~[SWS_EthSwt_00042]
replace EthTrcvGetLinkStateApi with EthSwtGetLinkStateApi
~SWS_EthSwt_00049
replace EthTrcvGetBaudRateApi with EthSwtGetBaudRateApi
~SWS_EthSwt_00056
replace EthTrcvGetDuplexModeApi with EthSwtGetDuplexModeApi
–Last change on issue 77043 comment 2–

BW-C-Level:

Application	Specification	Bus
1	3	1

1.17 Specification Item ECUC_EthSwt_00118

Trace References:

none

Content:

Name	EthSwtGetSwitchPortModeApiEthSwtGeneral.EthSwtGetSwitchPortModeApi		
Parent Container	EthSwtGeneral		
Description	Enables / Disables EthSwt_GetSwitchPortMode API		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	

Scope / Dependency	scope: local

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77043: [EthSwt] EthSwt APIs shouldnt be precompile ON/OFF based on EthTrcv parameters

  Problem description:

  For the APIs EthSwt_SetSwitchPortMode, EthSwt_GetSwitchPortMode, EthSwt_StartSwitchPortAutoNegotiation, EthSwt_GetLinkState, EthSwt_GetBaudRate and EthSwt_GetDuplexMode there is a requirement that states that the API should be configurable On/Off based on a EthTrcv parameter. For example for EthSwt_StartSwitchPortAutoNegotiation the requirement(SWS_EthSwt_00035) is: The function EthSwt_StartSwitchPortAutoNegotiation shall be pre-compile time configurable On/Off by the configuration parameter: EthTrcvStartAutoNegotiationApi.

  However, for other similar APIs a configuration parameter was defined in the switch driver. The functions are: EthSwt_GetPortSignalQuality, EthSwt_GetPortIdentifier, EthSwt_SetPortLoopbackMode, EthSwt_SetPortTxMode, EthSwt_GetPortCableDiagnosticsResult.

  Also, if multiple EthTrcv drivers are used, how will the precompile parameters be defined in the switch?

  Agreed solution:

  add parameters
  EthSwtSetSwitchPortModeApi, EthSwtGetSwitchPortModeApi, EthSwtStartSwitchPortAutoNegotiationApi, EthSwtGetLinkStateApi, EthSwtGetBaudRateApi and EthSwtGetDuplexModeApi
  Description: Enables / disables the EthSwt_SetSwitchPortMode / EthSwt_GetSwitchPortMode / EthSwt_StartSwitchPortAutoNegotiation / EthSwt_GetLinkState/ EthSwt_GetBaudRate / EthSwt_GetDuplexMode
  Multiplicity: 1
  Type EcucBooleanParamDef
  Default: False
  Variant false
  Config Class Pre-compile for all variants

  ~[SWS_EthSwt_00022]
  replace EthTrcvSetTransceiverModeApi with EthSwtSetSwitchPortModeApi.

~SWS_EthSwt_00029
replace EthTrcvGetTransceiverModeApi with EthSwtGetSwitchPortModeApi.
~SWS_EthSwt_00035
replace EthTrcvStartAutoNegotiationApi with EthSwtStartSwitchPortAutoNegotiationApi
~[SWS_EthSwt_00042]
replace EthTrcvGetLinkStateApi with EthSwtGetLinkStateApi
~SWS_EthSwt_00049
replace EthTrcvGetBaudRateApi with EthSwtGetBaudRateApi
~SWS_EthSwt_00056
replace EthTrcvGetDuplexModeApi with EthSwtGetDuplexModeApi
–Last change on issue 77043 comment 2–

BW-C-Level:

Application	Specification	Bus
1	3	1

1.18 Specification Item ECUC_EthSwt_00119

Trace References:

none

Content:

Name	EthSwtStartSwitchPortAutoNegotiationApiEthSwtGeneral.EthSwtStartSwitchPortAutoNegotiationApi		
Parent Container	EthSwtGeneral		
Description	Enables / Disables EthSwt_StartSwitchPortAutoNegotiation API		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77043: [EthSwt] EthSwt APIs shouldnt be precompile ON/OFF based on EthTrcv parameters

  Problem description:

  For the APIs EthSwt_SetSwitchPortMode, EthSwt_GetSwitchPortMode, EthSwt_StartSwitchPortAutoNegotiation, EthSwt_GetLinkState, EthSwt_GetBaudRate and EthSwt_GetDuplexMode there is a requirement that states that the API should be configurable On/Off based on a EthTrcv parameter. For example for EthSwt_StartSwitchPortAutoNegotiation the requirement(SWS_EthSwt_00035) is: The function EthSwt_StartSwitchPortAutoNegotiation shall be pre-compile time configurable On/Off by the configuration parameter: EthTrcvStartAutoNegotiationApi.

  However, for other similar APIs a configuration parameter was defined in the switch driver. The functions are: EthSwt_GetPortSignalQuality, EthSwt_GetPortIdentifier, EthSwt_SetPortLoopbackMode, EthSwt_SetPortTxMode, EthSwt_GetPortCableDiagnosticsResult.

  Also, if multiple EthTrcv drivers are used, how will the precompile parameters be defined in the switch?

  Agreed solution:

  add parameters
  EthSwtSetSwitchPortModeApi, EthSwtGetSwitchPortModeApi, EthSwtStartSwitchPortAutoNegotiationApi, EthSwtGetLinkStateApi, EthSwtGetBaudRateApi and EthSwtGetDuplexModeApi
  Description: Enables / disables the EthSwt_SetSwitchPortMode / EthSwt_GetSwitchPortMode / EthSwt_StartSwitchPortAutoNegotiation / EthSwt_GetLinkState/ EthSwt_GetBaudRate / EthSwt_GetDuplexMode
  Multiplicity: 1
  Type EcucBooleanParamDef
  Default: False
  Variant false
  Config Class Pre-compile for all variants

  ~[SWS_EthSwt_00022]
  replace EthTrcvSetTransceiverModeApi with EthSwtSetSwitchPortModeApi.
  ~SWS_EthSwt_00029
  replace EthTrcvGetTransceiverModeApi with EthSwtGetSwitchPortModeApi.
  ~SWS_EthSwt_00035
  replace EthTrcvStartAutoNegotiationApi with EthSwtStartSwitchPortAutoNegotiationApi
  ~[SWS_EthSwt_00042]

replace EthTrcvGetLinkStateApi with EthSwtGetLinkStateApi
~SWS_EthSwt_00049
replace EthTrcvGetBaudRateApi with EthSwtGetBaudRateApi
~SWS_EthSwt_00056
replace EthTrcvGetDuplexModeApi with EthSwtGetDuplexModeApi
–Last change on issue 77043 comment 2–

BW-C-Level:

Application	Specification	Bus
1	3	1

1.19 Specification Item ECUC_EthSwt_00120

Trace References:

none

Content:

Name	EthSwtGetLinkStateApiEthSwtGeneral.EthSwtGetLinkStateApi		
Parent Container	EthSwtGeneral		
Description	Enables / Disables EthSwt_GetLinkState API		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77043: [EthSwt] EthSwt APIs shouldnt be precompile ON/OFF based on EthTrcv parameters

  Problem description:

  For the APIs EthSwt_SetSwitchPortMode, EthSwt_GetSwitchPortMode, EthSwt_StartSwitchPortAutoNegotiation, EthSwt_GetLinkState, EthSwt_GetBaudRate and EthSwt_GetDuplexMode there is a requirement that states that the API should be configurable On/Off based on a EthTrcv parameter. For example for EthSwt_StartSwitchPortAutoNegotiation the requirement(SWS_EthSwt_00035) is: The function EthSwt_StartSwitchPortAutoNegotiation shall be pre-compile time configurable On/Off by the configuration parameter: EthTrcvStartAutoNegotiationApi.

However, for other similar APIs a configuration parameter was defined in the switch driver. The functions are: EthSwt_GetPortSignalQuality, EthSwt_GetPortIdentifier, EthSwt_SetPortLoopbackMode, EthSwt_SetPortTxMode, EthSwt_GetPortCableDiagnosticsResult.

Also, if multiple EthTrcv drivers are used, how will the precompile parameters be defined in the switch?

Agreed solution:

add parameters
EthSwtSetSwitchPortModeApi, EthSwtGetSwitchPortModeApi, EthSwtStartSwitchPortAutoNegotiationApi, EthSwtGetLinkStateApi, EthSwtGetBaudRateApi and EthSwtGetDuplexModeApi
Description: Enables / disables the EthSwt_SetSwitchPortMode / EthSwt_GetSwitchPortMode / EthSwt_StartSwitchPortAutoNegotiation / EthSwt_GetLinkState/ EthSwt_GetBaudRate / EthSwt_GetDuplexMode
Multiplicity: 1
Type EcucBooleanParamDef
Default: False
Variant false
Config Class Pre-compile for all variants

~[SWS_EthSwt_00022]
replace EthTrcvSetTransceiverModeApi with EthSwtSetSwitchPortModeApi.
~SWS_EthSwt_00029
replace EthTrcvGetTransceiverModeApi with EthSwtGetSwitchPortModeApi.
~SWS_EthSwt_00035
replace EthTrcvStartAutoNegotiationApi with EthSwtStartSwitchPortAutoNegotiationApi
~[SWS_EthSwt_00042]
replace EthTrcvGetLinkStateApi with EthSwtGetLinkStateApi
~SWS_EthSwt_00049
replace EthTrcvGetBaudRateApi with EthSwtGetBaudRateApi
~SWS_EthSwt_00056
replace EthTrcvGetDuplexModeApi with EthSwtGetDuplexModeApi
–Last change on issue 77043 comment 2–

BW-C-Level:

Application	Specification	Bus
1	3	1

1.20 Specification Item ECUC_EthSwt_00121

Trace References:

none

Content:

Name	EthSwtGetBaudRateApiEthSwtGeneral.EthSwtGetBaudRateApi		
Parent Container	EthSwtGeneral		
Description	Enables / Disables EthSwt_GetBaudRate API		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77043: [EthSwt] EthSwt APIs shouldnt be precompile ON/OFF based on EthTrcv parameters

  Problem description:

  For the APIs EthSwt_SetSwitchPortMode, EthSwt_GetSwitchPortMode, EthSwt_StartSwitchPortAutoNegotiation, EthSwt_GetLinkState, EthSwt_GetBaudRate and EthSwt_GetDuplexMode there is a requirement that states that the API should be configurable On/Off based on a EthTrcv parameter. For example for EthSwt_StartSwitchPortAutoNegotiation the requirement(SWS_EthSwt_00035) is: The function EthSwt_StartSwitchPortAutoNegotiation shall be pre-compile time configurable On/Off by the configuration parameter: EthTrcvStartAutoNegotiationApi.

However, for other similar APIs a configuration parameter was defined in the switch driver. The functions are: EthSwt_GetPortSignalQuality, EthSwt_GetPortIdentifier, EthSwt_SetPortLoopbackMode, EthSwt_SetPortTxMode, EthSwt_GetPortCableDiagnosticsResult.

Also, if multiple EthTrcv drivers are used, how will the precompile parameters be defined in the switch?

Agreed solution:

add parameters
EthSwtSetSwitchPortModeApi, EthSwtGetSwitchPortModeApi, EthSwtStartSwitchPortAutoNegotiationApi, EthSwtGetLinkStateApi, EthSwtGetBaudRateApi and EthSwtGetDuplexModeApi
Description: Enables / disables the EthSwt_SetSwitchPortMode / EthSwt_GetSwitchPortMode / EthSwt_StartSwitchPortAutoNegotiation / EthSwt_GetLinkState/ EthSwt_GetBaudRate / EthSwt_GetDuplexMode
Multiplicity: 1
Type EcucBooleanParamDef
Default: False
Variant false
Config Class Pre-compile for all variants

~[SWS_EthSwt_00022]
replace EthTrcvSetTransceiverModeApi with EthSwtSetSwitchPortModeApi.
~SWS_EthSwt_00029
replace EthTrcvGetTransceiverModeApi with EthSwtGetSwitchPortModeApi.
~SWS_EthSwt_00035
replace EthTrcvStartAutoNegotiationApi with EthSwtStartSwitchPortAutoNegotiationApi
~[SWS_EthSwt_00042]
replace EthTrcvGetLinkStateApi with EthSwtGetLinkStateApi
~SWS_EthSwt_00049
replace EthTrcvGetBaudRateApi with EthSwtGetBaudRateApi
~SWS_EthSwt_00056
replace EthTrcvGetDuplexModeApi with EthSwtGetDuplexModeApi
–Last change on issue 77043 comment 2–

BW-C-Level:

Application	Specification	Bus
1	3	1

1.21 Specification Item ECUC_EthSwt_00122

Trace References:

none

Content:

Name	EthSwtGetDuplexModeApiEthSwtGeneral.EthSwtGetDuplexModeApi		
Parent Container	EthSwtGeneral		
Description	Enables / Disables EthSwt_GetDuplexMode API		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77043: [EthSwt] EthSwt APIs shouldnt be precompile ON/OFF based on EthTrcv parameters

  Problem description:

  For the APIs EthSwt_SetSwitchPortMode, EthSwt_GetSwitchPortMode, EthSwt_StartSwitchPortAutoNegotiation, EthSwt_GetLinkState, EthSwt_GetBaudRate and EthSwt_GetDuplexMode there is a requirement that states that the API should be configurable On/Off based on a EthTrcv parameter. For example for EthSwt_StartSwitchPortAutoNegotiation the requirement(SWS_EthSwt_00035) is: The function EthSwt_StartSwitchPortAutoNegotiation shall be pre-compile time configurable On/Off by the configuration parameter: EthTrcvStartAutoNegotiationApi.

  However, for other similar APIs a configuration parameter was defined in the switch driver. The functions are: EthSwt_GetPortSignalQuality, EthSwt_GetPortIdentifier, EthSwt_SetPortLoopbackMode, EthSwt_SetPortTxMode, EthSwt_GetPortCableDiagnosticsResult.

Also, if multiple EthTrcv drivers are used, how will the precompile parameters be defined in the switch?

Agreed solution:

add parameters
EthSwtSetSwitchPortModeApi, EthSwtGetSwitchPortModeApi, EthSwtStartSwitchPortAutoNegotiationApi, EthSwtGetLinkStateApi, EthSwtGetBaudRateApi and EthSwtGetDuplexModeApi
Description: Enables / disables the EthSwt_SetSwitchPortMode / EthSwt_GetSwitchPortMode / EthSwt_StartSwitchPortAutoNegotiation / EthSwt_GetLinkState/ EthSwt_GetBaudRate / EthSwt_GetDuplexMode
Multiplicity: 1
Type EcucBooleanParamDef
Default: False
Variant false
Config Class Pre-compile for all variants

~[SWS_EthSwt_00022]
replace EthTrcvSetTransceiverModeApi with EthSwtSetSwitchPortModeApi.
~SWS_EthSwt_00029
replace EthTrcvGetTransceiverModeApi with EthSwtGetSwitchPortModeApi.
~SWS_EthSwt_00035
replace EthTrcvStartAutoNegotiationApi with EthSwtStartSwitchPortAutoNegotiationApi
~[SWS_EthSwt_00042]
replace EthTrcvGetLinkStateApi with EthSwtGetLinkStateApi
~SWS_EthSwt_00049
replace EthTrcvGetBaudRateApi with EthSwtGetBaudRateApi
~SWS_EthSwt_00056
replace EthTrcvGetDuplexModeApi with EthSwtGetDuplexModeApi
–Last change on issue 77043 comment 2–

BW-C-Level:

Application	Specification	Bus
1	3	1

1.22 Specification Item ECUC_EthSwt_00123

Trace References:

none

Content:

Name	EthSwtGetCfgRawEthSwtGeneral.EthSwtGetCfgRaw		
Parent Container	EthSwtGeneral		
Description	Disable /Enable support of reading raw data from switch memory		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77018: [EthSwt] Clarification about the use case regarding API "EthSwt_GetCfgHexDump"

  Problem description:

  We discussed the use case regarding the API "EthSwt_GetCfgHexDump". Daimler want to read out the register memory of an Ethernet switch as a memory dump. The memory dump is transmitted via an diagnostic service over the network consecutively to an diagnostic tester. The memory dump is used to verify and compare to the expected / required values. It is not clear if the API is sufficient for this use case.

  Agreed solution:

  Capter 7.1.2.13 exchange in listing of APIS EthSwt_GetCfgHexDump with EthSwt_GetCfgDataRaw and EthSwt_GetCfgHexDumpLength with EthSwt_GetCfgDataInfo.

  Chapter 8
  -SWS_EthSwt_91026,
  -SWS_EthSwt_00352
  -SWS_EthSwt_00353
  -SWS_EthSwt_00355
  -ECUC_EthSwt_00093
  -SWS_EthSwt_91027
  -SWS_EthSwt_00356

-SWS_EthSwt_00357
-SWS_EthSwt_00358
-SWS_EthSwt_00359
Set APIs EthSwt_GetCfgHexDump and EthSwt_GetCfgHexDumpLength and config parameters EthSwtGetCfgHexDumpApi and EthSwt_GetCfgHexDumpLengthApi to deprecated or delete.

Chapter 10
Add parameter to enable disable APIs with name EthSwtGetCfgRaw to EthSwtGeneral-Container
Description Disable /Enable support of reading raw data from switch memory.
Multiplicity: 1
Type: EcucBooleanParamDef
Default: false
Post build variant false
Config class Pre-compile time for all variants.
scope: local

Introduce following new APIs to function: definitions
EthSwt_GetCfgDataRaw(
uint8 SwitchIdx,
uint32 Offset,
uint16 Length,
uint8 *BufferPtr
)
[attributes
- asynchronous
- non-reentrant
parameters IN: SwitchIdx:Index of the Ethernet switch within the context of the Ethernet Switch Driver
Offset: Offset of the Ethernet switch memory from where the reading starts
Length: Length of data in bytes that shall be copied.
parameters OUT: BufferPtr: Pointer to the location where the data shall be copied.
Std_ReturnType: E_OK: the data read was triggered successfully. E_NOT_OK: the data read was not triggered successfully. (i.e. indexed Ethernet switch is not available)
Description:
Retrieves the data in memory of the indexed Ethernet switch in variable length.]

+[SWS_EthSwt_xxxxx1] The function EthSwt_GetCfgDataRaw() shall only be available if parameter EthSwtGetCfgRaw is set to TRUE.(SRS_BSW_00171)
+[SWS_EthSwt_xxxxx2] When calling the function EthSwt_GetCfgDataRaw(), the

function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.()

EthSwt_GetCfgDataInfo(
uint8 SwitchIdx,
uint32 *DataSizePtr,
uint32 *DataAdressPtr
)
[attributes
- synchronous
- reentrant
parameters IN: SwitchIdx:Index of the Ethernet switch within the context of the Ethernet Switch Driver
parameters OUT: DataSizePtr: Pointer to the location where the total size of the configuration data shall be copied.
DataAddressPtr: Pointer to the location where the start address of the configuration registers shall be copied.
Std_ReturnType: E_OK: the data was obtained successfully. E_NOT_OK: the data was not obtained successfully. (i.e. indexed Ethernet switch is not available)
Description:
Retrieves the total size of data and the memory start address of the indexed Ethernet Switch.]

+[SWS_EthSwt_xxxxx3] The function EthSwt_GetCfgDataInfo() shall only be available if parameter EthSwtGetCfgRaw is set to TRUE.(SRS_BSW_00171)
+[SWS_EthSwt_xxxxx4] When calling the function EthSwt_GetCfgDataInfo(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.()

add following API to chapter "8.6.3 Configurable interfaces"
<GetCfgDataRawDone>(
uint8 SwitchIdx
)
[attributes:
synchronous
reentrant
parameters IN: SwitchIdx: Index of the Ethernet switch where the Configuration is

read.
Return value : void
Description: The call of the function EthSwt_GetCfgDataRaw() triggers a asynchrony read of a certain memory section of the Ethernet switch driver. If the read is done, the configured callout function <GetCfgDataRawDone> shall be called]

Add a parameter to EthSwtGeneral-Container
GetCfgDataRawDone
Decription: Defines the function name for <GetCfgDataRawDone>
Multplicity: 0 ..1
Type EcucFuntionNameDef
Variant and Variant Multiplicity false
Config class Pre-compile for all variants
scope: local
dependency: The function GetCfgDataRawDone shall only be configured if parameter EthSwtGetCfgRaw is set to TRUE.

Header file name parameter is defined in ECUC_EthSwt_00064
–Last change on issue 77018 comment 32–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.23 Specification Item ECUC_EthSwt_00124

Trace References:

none

Content:

Name	EthSwtGetCfgDataRawDoneEthSwtGeneral.EthSwtGetCfgDataRawDone
Parent Container	EthSwtGeneral
Description	Defines the function name for <GetCfgDataRawDone>
Multiplicity	0..1
Type	EcucFunctionNameDef
Default value	–
maxLength	–

minLength	–		
regularExpression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local dependency: The function GetCfgDataRawDone shall only be configured if parameter EthSwtGetCfgRaw is set to TRUE.		

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77018: [EthSwt] Clarification about the use case regarding API "EthSwt_GetCfgHexDump"

  Problem description:

  We discussed the use case regarding the API "EthSwt_GetCfgHexDump". Daimler want to read out the register memory of an Ethernet switch as a memory dump. The memory dump is transmitted via an diagnostic service over the network consecutively to an diagnostic tester. The memory dump is used to verify and compare to the expected / required values. It is not clear if the API is sufficient for this use case.

  Agreed solution:

  Capter 7.1.2.13 exchange in listing of APIS EthSwt_GetCfgHexDump with EthSwt_GetCfgDataRaw and EthSwt_GetCfgHexDumpLength with EthSwt_GetCfgDataInfo.

  Chapter 8
  -SWS_EthSwt_91026,
  -SWS_EthSwt_00352
  -SWS_EthSwt_00353
  -SWS_EthSwt_00355
  -ECUC_EthSwt_00093
  -SWS_EthSwt_91027
  -SWS_EthSwt_00356

-SWS_EthSwt_00357
-SWS_EthSwt_00358
-SWS_EthSwt_00359
Set APIs EthSwt_GetCfgHexDump and EthSwt_GetCfgHexDumpLength and config parameters EthSwtGetCfgHexDumpApi and EthSwt_GetCfgHexDumpLengthApi to deprecated or delete.

Chapter 10
Add parameter to enable disable APIs with name EthSwtGetCfgRaw to EthSwtGeneral-Container
Description Disable /Enable support of reading raw data from switch memory.
Multiplicity: 1
Type: EcucBooleanParamDef
Default: false
Post build variant false
Config class Pre-compile time for all variants.
scope: local

Introduce following new APIs to function: definitions
EthSwt_GetCfgDataRaw(
uint8 SwitchIdx,
uint32 Offset,
uint16 Length,
uint8 *BufferPtr
)
[attributes
- asynchronous
- non-reentrant
parameters IN: SwitchIdx:Index of the Ethernet switch within the context of the Ethernet Switch Driver
Offset: Offset of the Ethernet switch memory from where the reading starts
Length: Length of data in bytes that shall be copied.
parameters OUT: BufferPtr: Pointer to the location where the data shall be copied.
Std_ReturnType: E_OK: the data read was triggered successfully. E_NOT_OK: the data read was not triggered successfully. (i.e. indexed Ethernet switch is not available)
Description:
Retrieves the data in memory of the indexed Ethernet switch in variable length.]

+[SWS_EthSwt_xxxxx1] The function EthSwt_GetCfgDataRaw() shall only be available if parameter EthSwtGetCfgRaw is set to TRUE.(SRS_BSW_00171)
+[SWS_EthSwt_xxxxx2] When calling the function EthSwt_GetCfgDataRaw(), the

function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.()

EthSwt_GetCfgDataInfo(
uint8 SwitchIdx,
uint32 *DataSizePtr,
uint32 *DataAdressPtr
)
[attributes
- synchronous
- reentrant
parameters IN: SwitchIdx:Index of the Ethernet switch within the context of the Ethernet Switch Driver
parameters OUT: DataSizePtr: Pointer to the location where the total size of the configuration data shall be copied.
DataAddressPtr: Pointer to the location where the start address of the configuration registers shall be copied.
Std_ReturnType: E_OK: the data was obtained successfully. E_NOT_OK: the data was not obtained successfully. (i.e. indexed Ethernet switch is not available)
Description:
Retrieves the total size of data and the memory start address of the indexed Ethernet Switch.]

+[SWS_EthSwt_xxxxx3] The function EthSwt_GetCfgDataInfo() shall only be available if parameter EthSwtGetCfgRaw is set to TRUE.(SRS_BSW_00171)
+[SWS_EthSwt_xxxxx4] When calling the function EthSwt_GetCfgDataInfo(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.()

add following API to chapter "8.6.3 Configurable interfaces"
<GetCfgDataRawDone>(
uint8 SwitchIdx
)
[attributes:
synchronous
reentrant
parameters IN: SwitchIdx: Index of the Ethernet switch where the Configuration is

read.
Return value : void
Description: The call of the function EthSwt_GetCfgDataRaw() triggers a asynchrony read of a certain memory section of the Ethernet switch driver. If the read is done, the configured callout function <GetCfgDataRawDone> shall be called]

Add a parameter to EthSwtGeneral-Container
GetCfgDataRawDone
Decription: Defines the function name for <GetCfgDataRawDone>
Multplicity: 0 ..1
Type EcucFuntionNameDef
Variant and Variant Multiplicity false
Config class Pre-compile for all variants
scope: local
dependency: The function GetCfgDataRawDone shall only be configured if parameter EthSwtGetCfgRaw is set to TRUE.

Header file name parameter is defined in ECUC_EthSwt_00064
–Last change on issue 77018 comment 32–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.24 Specification Item ECUC_EthSwt_00125

Trace References:

none

Content:

Name	ETHSWT_E_SYNCPORT2PHYEthSwtDemEventParameterRefs.ETHSWT_E_SYNCPORT2PHY
Parent Container	EthSwtDemEventParameterRefs
Description	Reference to the DemEventParameter which shall be issued when the error "Ethernet switch port and the referenced Ethernet transceiver are in contradicting modes" has occurred.
Multiplicity	0..1
Type	Symbolic name reference to [DemEventParameter]
Post-Build Variant Multiplicity	true

Post-Build Variant Value	true		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77250: [EthIf][EthSwt][EthTrcv] EthSwt_SetSwitchPortMode() broken for ports without Ethernet transceiver

  Problem description:

  SWS_EthSwt_00019 states that EthSwt_SetSwitchPortMode() shall (unconditionally) invoke EthTrcv_SetTransceiverMode() of the Ethernet Transceiver Driver.

  This is bogus for the following two reasons:
  1) a particular switch port does not necessarily have an associated Ethernet transceiver. - In that case EthSwtPortTrcvRef won't be present. This is probably the case for MAC2MAC links if EthSwtPortRole is present an either set to ETHSWT_HOST_PORT or to ETHSWT_UP_LINK_PORT.
  2) Even if the particular switch port has an associated Ethernet transceiver, SWS_EthTrcv_00043 states that the EthTrcv shall invoke EthIf_TrcvModeIndication() as a response to EthTrcv_SetTransceiverMode().

  IMHO 1) should be solved by extending SWS_EthSwt_00019 distinguishing between ports which are associated with a transceiver (i.e., EthSwtPortTrcvRef present) - in that case EthTrcv_SetTransceiverMode() should be called - and ports which are not associated with a transceiver (i.e., EthSwtPortTrcvRef is not present) - in that case something else has to happen and this needs to be specified (probably the switch port shall be internally just disabled - maybe this needs to happen in the first case as well?)

  2) leads to the following call chain: EthIf_SwitchPortGroupRequestMode() -> EthSwt_SetSwitchPortMode() -> EthTrcv_SetTransceiverMode() -> EthIf_TrcvModeIndication(). Thus the indication actually bypasses the EthSwt

which IMHO is extremely ugly. Additionally for switch ports without associated transceiver, the EthIf will never get a EthIf_TrcvModeIndication() ...

Agreed solution:

=== System Template ===
+ Add upstream mapping of EthSwtPortTrcvRef to CouplingPort.physicalLayerType

=== EthSwt ===

~ch. 7.2.4 Production Errors
move SWS_EthSwt_00113 to 7.2.5 Extended production Errors
~SWS_EthSwt_00113 change detection criteria to
Fail:When access to the Ethernet Switch fails the module shall report the extended production error with event status DEM_EVENT_STATUS_PREFAILED to DEM.
Pass:When access to the Ethernet Switch succeeds the module shall report the extended production error with event status DEM_EVENT_STATUS_PREPASSED to DEM.

+ SWS_EthSwt_xxxxx new ext. production Error
Error Name: ETHSWT_E_SYNCPORT2PHY
Short Description: Ethernet switch port and the referenced Ethernet transceiver are in contradicting modes.
Long description: While getting the Ethernet switch port mode, the Ethernet switch driver detected an inconsistent state between Ethernet switch port and the referenced Ethernet transceiver Mode.
Detection Criteria: Fail: When getting the Ethernet switch port mode together with the Ethernet transceiver mode and the mode of the two referenced modules was found
inconsistent the module shall report the extended production error with event status DEM_EVENT_STATUS_PREFAILED to DEM.
Pass:When getting the Ethernet switch port mode together with the Ethernet transceiver mode and the mode of the two referenced modules was found
consistent the module shall report the extended production error with event status DEM_EVENT_STATUS_PREPASSED to DEM.
all other attributer: N/A

~ ch. 8.3.2 EthSwt_SwitchInit
~[SWS_EthSwt_00016] change "production error" to "extended production error"

~ch. 8.3.3 EthSwt_SetSwitchPortMode
~[SWS_EthSwt_00019]The function EthSwt_SetSwitchPortMode() shall put the indexed port of the switch into the specified mode. If EthSwtPort references an

EthTrcv then the function EthTrcv_SetTransceiverMode() of the Ethernet Transceiver Driver shall additionally be called with the corresponding transceiver mode.

+[SWS_EthSwt_xxxxx] When calling the function EthSwt_SetSwitchPortMode() with mode ETHTRCV_MODE_DOWN, the EthSwt shall disable the Ethernet switch port directly for reduction of power consumption, if it is possible.

+[SWS_EthSwt_xxxxx] When calling the function EthSwt_SetSwitchPortMode(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.

+[SWS_EthSwt_xxxxxx] If EthSwtPort does not references an EthTrcv, EthSwt shall indicate a mode of the port by the API EthIf_SwitchPortModeIndication latest during the next EthSwt_MainFunction.(SRS_ETH_00118)

~ch. 10.1.3 EthSwtDemEventParameterRefs
+ SWS Item ECUC_EthSwt_xxxxx1 : Name ETHSWT_E_SYNCPORT2PHY
Parent Container EthSwtDemEventParameterRefs
Description Reference to the DemEventParameter which shall be issued when the error "Ethernet switch port and the referenced Ethernet transceiver are in contradicting modes" has occurred.
Multiplicity 0..1
Type Symbolic name reference to [DemEventParameter]
Post-Build Variant Multiplicity true
Post-Build Variant Value true
Multiplicity Configuration
Class Pre-compile time X VARIANT-PRE-COMPILE
Link time X VARIANT-LINK-TIME
Post-build time X VARIANT-POST-BUILD
Value Configuration Class
Pre-compile time X VARIANT-PRE-COMPILE
Link time X VARIANT-LINK-TIME
Post-build time X VARIANT-POST-BUILD
Scope / Dependency
scope: local

~ch. 10.1.5 EthSwtPort
~ECUC_EthSwt_00041 EthSwtPortTrcvRef change following attributes:
Multiplicity Configuration Class | Link time | VARIANT-LINK-TIME, VARIANT-POST-BUILD

| Post build | –
add to dependency: If EthSwtPortPhysicalLayerType is defined, then EthSwtPortTrcvRef holds the reference to the corresponding EthTrcv.

~ECUC_EthSwt_00054 add dependency to EthSwtPortPhysicalLayerType: If a EthSwtPort has an EthSwtPortPhysicalLayerType then EthSwtPort shall reference an EthTrcv.

=== EthIf ===
Add to chapter 8.4 (callback notifications):
+ EthIf_SwitchPortModeIndication
Syntax: void EthIf_SwitchPortModeIndication(uint8 SwitchIdx, uint8 SwitchPortIdx, EthSwt_ModeType PortMode)
Sevice ID: pick a free one
Sync/Async: Synchronous
Reentrancy: Non Reentrant
Parmaters in: SwitchIdx: Index of the switch within the context of the Ethernet Switch Driver
SwitchPortIdx: Index of the port at the addressed switch
PortMode: notified Ethernet Switch port mode.
Description: The EthIf shall determine the expected notifications based on the EthSwtPort configuration. In case the EthSwtPort references an EthTrcv the EthIf expects a notification from the EthTrcv via API EthIf_TrcvModeIndication(). Otherwise the EthIf expects a notification from the EthSwt via API EthIf_SwitchPortModeIndication().

~SWS_EthIf_00232
change from
Called asynchronously when mode has been read out. Triggered by previous
EthTrcv_SetTransceiverMode call. Can directly be called within the trigger
functions.
To
Called asynchronously when a mode change has been read out. If the function is triggered by previous call of
EthTrcv_SetTransceiverMode it can directly be called within the trigger function.
–Last change on issue 77250 comment 50–

BW-C-Level:

Application	**Specification**	**Bus**
1	4	1

1.25 Specification Item noname

Trace References:

SRS_BSW_00406 00171

Content:

If default error detection is enabled: the The function EthSwt_PortEnableTimeStampGetCfgDataRaw() shall check that the service only be available if parameter EthSwt_Init() was previously called.

If the check fails, the function EthSwt_PortEnableTimeStamp() shall raise the development error ETHIF_E_NOT_INITIALIZED and return E_NOT_OKGetCfgRaw is set to TRUE.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77018: [EthSwt] Clarification about the use case regarding API "EthSwt_GetCfgHexDump"

  Problem description:

  We discussed the use case regarding the API "EthSwt_GetCfgHexDump". Daimler want to read out the register memory of an Ethernet switch as a memory dump. The memory dump is transmitted via an diagnostic service over the network consecutively to an diagnostic tester. The memory dump is used to verify and compare to the expected / required values. It is not clear if the API is sufficient for this use case.

  Agreed solution:

  Capter 7.1.2.13 exchange in listing of APIS EthSwt_GetCfgHexDump with EthSwt_GetCfgDataRaw and EthSwt_GetCfgHexDumpLength with EthSwt_GetCfgDataInfo.

  Chapter 8
  -SWS_EthSwt_91026,
  -SWS_EthSwt_00352
  -SWS_EthSwt_00353
  -SWS_EthSwt_00355
  -ECUC_EthSwt_00093
  -SWS_EthSwt_91027
  -SWS_EthSwt_00356
  -SWS_EthSwt_00357
  -SWS_EthSwt_00358
  -SWS_EthSwt_00359
  Set APIs EthSwt_GetCfgHexDump and EthSwt_GetCfgHexDumpLength and config parameters EthSwtGetCfgHexDumpApi and EthSwt_GetCfgHexDumpLengthApi to

deprecated or delete.

Chapter 10
Add parameter to enable disable APIs with name EthSwtGetCfgRaw to EthSwtGeneral-Container
Description Disable /Enable support of reading raw data from switch memory.
Multiplicity: 1
Type: EcucBooleanParamDef
Default: false
Post build variant false
Config class Pre-compile time for all variants.
scope: local

Introduce following new APIs to function: definitions
EthSwt_GetCfgDataRaw(
uint8 SwitchIdx,
uint32 Offset,
uint16 Length,
uint8 *BufferPtr
)
[attributes
- asynchronous
- non-reentrant
parameters IN: SwitchIdx:Index of the Ethernet switch within the context of the Ethernet Switch Driver
Offset: Offset of the Ethernet switch memory from where the reading starts
Length: Length of data in bytes that shall be copied.
parameters OUT: BufferPtr: Pointer to the location where the data shall be copied.
Std_ReturnType: E_OK: the data read was triggered successfully. E_NOT_OK: the data read was not triggered successfully. (i.e. indexed Ethernet switch is not available)
Description:
Retrieves the data in memory of the indexed Ethernet switch in variable length.]

+[SWS_EthSwt_xxxxx1] The function EthSwt_GetCfgDataRaw() shall only be available if parameter EthSwtGetCfgRaw is set to TRUE.(SRS_BSW_00171)
+[SWS_EthSwt_xxxxx2] When calling the function EthSwt_GetCfgDataRaw(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.()

EthSwt_GetCfgDataInfo(
uint8 SwitchIdx,
uint32 *DataSizePtr,
uint32 *DataAdressPtr
)
[attributes
- synchronous
- reentrant
parameters IN: SwitchIdx:Index of the Ethernet switch within the context of the Ethernet Switch Driver
parameters OUT: DataSizePtr: Pointer to the location where the total size of the configuration data shall be copied.
DataAddressPtr: Pointer to the location where the start address of the configuration registers shall be copied.
Std_ReturnType: E_OK: the data was obtained successfully. E_NOT_OK: the data was not obtained successfully. (i.e. indexed Ethernet switch is not available)
Description:
Retrieves the total size of data and the memory start address of the indexed Ethernet Switch.]

+[SWS_EthSwt_xxxxx3] The function EthSwt_GetCfgDataInfo() shall only be available if parameter EthSwtGetCfgRaw is set to TRUE.(SRS_BSW_00171)
+[SWS_EthSwt_xxxxx4] When calling the function EthSwt_GetCfgDataInfo(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.()

add following API to chapter "8.6.3 Configurable interfaces"
<GetCfgDataRawDone>(
uint8 SwitchIdx
)
[attributes:
synchronous
reentrant
parameters IN: SwitchIdx: Index of the Ethernet switch where the Configuration is read.
Return value : void
Description: The call of the function EthSwt_GetCfgDataRaw() triggers a asynchrony read of a certain memory section of the Ethernet switch driver. If the read is done, the configured callout function <GetCfgDataRawDone> shall be called]

Add a parameter to EthSwtGeneral-Container
GetCfgDataRawDone
Decription: Defines the function name for <GetCfgDataRawDone>
Multplicity: 0 ..1
Type EcucFuntionNameDef
Variant and Variant Multiplicity false
Config class Pre-compile for all variants
scope: local
dependency: The function GetCfgDataRawDone shall only be configured if parameter EthSwtGetCfgRaw is set to TRUE.

Header file name parameter is defined in ECUC_EthSwt_00064
–Last change on issue 77018 comment 32–

BW-C-Level:

Application	**Specification**	**Bus**
1	4	1

1.26 Specification Item SWS_EthSwt_00001

Trace References:

SRS_BSW_00385

Content:

Type or error	Related error code	Value [hex]
Invalid switch index	ETHSWT_E_INV_SWITCH_IDX	0x01
EthSwt module was not initialized	ETHSWT_E_NOT_INITIALIZED UNINIT	0x02
Invalid pointer in parameter list	ETHSWT_E_INVPARAM_POINTER	0x03
Invalid API which is not available by another module	ETHSWT_E_INV_API	0x05
Invalid switch port index	ETHSWT_E_INV_SWITCHPORT_IDX	0x06
Invalid Controller Index	ETHSWT_E_INV_CTRL_IDX	0x07
Invalid input parameter	ETHSWT_E_INV_PARAM	0x08
Invalid configuration	ETHSWT_E_INIT_FAILED	0x09

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

Problem description:

[SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

In the above requirement -
1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with

input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.27 Specification Item SWS_EthSwt_00004

Trace References:

none

Content:

The types specified in SWS_EthernetSwitchDriver shall be declared in Eth_GeneralTypes.h

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76197: [EthSwt] Requirement stating that all types of module are specified in Eth_General makes no sence

  Problem description:

  SWS_EthSwt_00004 shall be deleted. It is completely not needed to specify internal types to Eth_General. All public types were moved to Eth_General in Release 4.3-!

Agreed solution:

Remove SWS_EthSwt_00004

BW-C-Level:

Application	Specification	Bus
1	1	1

1.28 Specification Item SWS_EthSwt_00006

Trace References:

SRS_ETH_00086 BSW_00101

Content:

Service name:	EthSwt_InitEthSwt_Init	
Syntax:	void EthSwt_Init(const EthSwt_ConfigType* CfgPtr)	
Service ID[hex]:	0x01	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	CfgPtrEthSwt_Init.CfgPtr	Points to the implementation specific structure
Parameters (inout):	None	
Parameters (out):	None	
Return value:	None	
Description:	Initializes the Ethernet Switch Driver	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76248: [EthSwt]Remove references to non existing requirements (SRS, SWS)

  Problem description:

  Some review findings brought up references to non existing requirements. Please check and remove the references and if needed reformulate the requirements.

  Agreed solution:

  SWS_EthSwt_00123 –> SRS_BSW_00406
  SWS_EthSwt_00165 -> SRS_BSW_00395
  SWS_EthSwt_00227 -> SRS_ETH_00087
  SWS_EthSwt_91017, SWS_EthSwt_91020, –> SRS_ETH_00123
  SWS_EthSwt_00006 –> SRS_BSW_00101

SWS_EthSwt_00010 left blank as SRS doe not yet provide a requirement here but maybe with RFC # 75170 AUTOSAR will introduce an SRS-item after all.
SWS_EthSwt_00376, SWS_EthSwt_00374, SWS_EthSwt_00375 –> SRS-Missing! same as for SWS_EThSwt_00010
SWS_EthSwt_00018, SRS_EthSwt_00025 –> SRS_ETH_00118
SWS_EthSwt_00031 –> SRS_ETH_00087
SWS_EthSwt_00037 –> SRS_ETH_00119
SWS_EthSwt_00044, SWS_EthSwt_00051 –> SRS_ETH_00118
SWS_EthSwt_00060 –> SRS_ETH_00087
SWS_EthSwt_00111 –> erase SRS_ETH_00086
SWS_EthSwt00079 –>SRS_ETH_00119
SWS_EthSwt_00206, SWS_EthSwt_00211, SWS_EthSwt_00216, SWS_EthSwt_00221 –>SRS_Eth_00120
SWS_EthSwt_00172 –>SRS_ETH_00121, SRS_ETH_00114
SWS_EthSwt_00086, SWS_EthSwt_00091, SWS_EthSwt_00182, SWS_EthSwt_00125 –> SRS_ETH_00087, SRS_ETH_00122
SWS_EthSwt_00187 –> SRS_ETH_00087
SWS_EthSwt_00058 –> SRS_BSW_00171
SWS_EthSwt_91012, SWS_EthSwt_91013 –> SRS_ETH_00126
SWS_EthSwt_91014, SWS_EthSwt_91015, SWS_EthSwt_91016, SWS_EthSwt_91018, SWS_EthSwt_91019, SWS_EthSwt_91021, SWS_EthSwt_91022, SWS_EthSwt_91022, SWS_EthSwt_91023, SWS_EthSwt_91024, SWS_EthSwt_91025, SWS_EthSwt_91026, SWS_EthSwt_91027 –> SRS_Eth_00123
SWS_EthSwt_00117, SWS_EthSwt_00203 –> SRS_ETH_00119, SRS_ETH_00087
SWS_EthSwt_00193 –> SRS_ETH_00122, SRS_ETH_00087
SWS_EthSwt_00114 –> SRS_BSW_00433
SWS_EthSwt_00098 –> SRS_Eth_00122, SRS_ETH_00118, SRS_ETH_00119, SRS_ETH_00120, SRS_ETH_00087, SRS_ETH_00125, SRS_BSW_00375
–Last change on issue 76248 comment 13–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.29 Specification Item SWS_EthSwt_00008

Trace References:

SRS_BSW_00101

Content:

The function EthSwt_Init shall change the state of the component all switches controlled by this Switch Driver from ETHSWT_STATE_UNINIT to ETHSWT_STATE_INIT.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76887: [EthSwt] EthSwt_SwitchInit shall change the state of a switch to ACTIVE, not the state of the component.

  Problem description:

  Requirement SWS_EthSwt_00012 states that EthSwt_SwitchInit shall change the state of the component from ETHSWT_STATE_INIT to ETHSWT_STATE_ACTIVE.This translates into the fact that if at least one switch is active, we assume that all switches are active. The subsequent call of a function for a certain switch for which EthSwt_SwitchInit was not called, will assume that the switch is active.

  EthSwt_SwithcInit shall change the state of a switch to ACTIVE, not the state of the component.

  Agreed solution:

  [SWS_EthSwt_00008]
  The function EthSwt_Init shall change the state of all switches controlled by this Switch Driver from
  ETHSWT_STATE_UNINIT to ETHSWT_STATE_INIT.

  [SWS_EthSwt_00012]
  EthSwt_SwitchInit shall change the state of the indexed switch from
  ETHSWT_STATE_INIT to ETHSWT_STATE_ACTIVE.

  SWS_EthSwt_00123
  change description of range to
  ETHSWT_STATE_UNINIT 0x00 Switch is not yet configured
  ETHSWT_STATE_INIT 0x01 Switch is configured
  ETHSWT_STATE_ACTIVE 0x02 Switch is active
  –Last change on issue 76887 comment 12–

  BW-C-Level:

Application	Specification	Bus
1	4	1

1.30 Specification Item SWS_EthSwt_00009

Trace References:

SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled: , the function EthSwt_Init shall check the parameter CfgPtr for being valid, i. e. not Null pointer. . If the check fails, the function EthSwt_Init shall raise the development error ETHSWT_E_INV_POINTER and return E_NOT_OK. In INIT_FAILED. Note: Please note that in case of variant pre-compile , NULL_PTR is allowed.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76886: [EthSwt] SWS_EthSwt_00009 contradicts SWS_EthSwt_00006

  Problem description:

  The SWS_EthSwt_00006 defines the prototype for EthSwt_Init function.
  Service name:EthSwt_Init
  Syntax: void EthSwt_Init(const EthSwt_ConfigType* CfgPtr)
  [...]

  Requirement SWS_EthSwt_00009 states that the functions shall return E_NOT_OK in case the parameter check fails.

  The "and return E_NOT_OK." should be removed from the requirement text

  Agreed solution:

  The "and return E_NOT_OK." should be removed from the SWS_EthSwt_00009 text to be compliant with SWS_EthSwt_00006

  BW-C-Level:

Application	Specification	Bus
1	1	1

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -

1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevEr-

rorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.31 Specification Item SWS_EthSwt_00010

Trace References:

SRS_ETH_00086

Content:

Service name:	EthSwt_SwitchInitEthSwt_SwitchInit	
Syntax:	Std_ReturnType EthSwt_SwitchInit(uint8 SwitchIdx)	
Service ID[hex]:	0x02	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	SwitchIdxEthSwt_SwitchInit.SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: success E_NOT_OK: switch could not be initialized
Description:	Initializes the indexed swtich with a given configuration for the switch index	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76879: [EthTrcv] Add EthSwt API's to optional interfaces

  Problem description:

Optional interfaces are missing, in case the Ethernet transceiver is connected to a Ethernet switch

Agreed solution:

add the following API's to optional interfaces:
EthSwt_ReadTrcvRegister
EthSwt_WriteTrcvRegister

move the following API's from mandatory interfaces to optional interfaces:
Eth_ReadMii
Eth_WriteMii
–Last change on issue 76879 comment 1–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.32 Specification Item SWS_EthSwt_00011

Trace References:

SRS_BSW_00101

Content:

EthSwt_SwitchInit shall:

- Configure if necessary configure all configuration parameters (e.g. port structure, VLAN configuration, ...) at all ports of the switch and the switch itself.
- Perform if necessary perform a soft reset, i.e. resetting the switch via register setting not via a reset pin. This is hardware dependent and might not be supported by all switch devices.
- After resetting the switch and EthSwtLowPowerModeSupport set to TRUE, the Ethernet switch shall enter an inactive or low power mode. If EthSwtLowPowerMode Support is not defined or set to FALSE the Ethernet switch shall enter an active state

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #75170: [EthSwt] Handling of EthSwt_SwitchInit

  Problem description:

As result of RfC # 73706, the handling of the EthSwt_SwitchInit is not clear:
- Which modul is calling the EthSwt_SwitchInit?
- Why is the function not asynchron since the configuration of the ethernet switch driver may delay the start up?

Agreed solution:

~SWS_EthSwt_00011:
EthSwt_SwitchInit shall:
-if necessary configure all configuration parameters (e.g. port structure, VLAN configuration, ...) at all ports of the switch and the switch itself.
-if necessary perform a soft reset, i.e. resetting the switch via register setting not via a reset pin. This is hardware dependent and might not be supported by all switch devices.
-after resetting the switch and EthSwtLowPowerModeSupport is set to TRUE, the Ethernet switch shall enter an inactive or low power mode. If EthSwtLowPowerModeSupport is not defined or set to FALSE the Ethernet switch shall enter an active state

NOTE: Please Note that this function can take a very long time to complete and shall in this case can not be called by EcuM or BswM. Instead it e.g. could be called from a background task.
–Last change on issue 75170 comment 24–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.33 Specification Item SWS_EthSwt_00012

Trace References:

SRS_BSW_00101

Content:

EthSwt_SwitchInit shall change the state of the component indexed switch from ETHSWT_STATE_INIT to ETHSWT_STATE_ACTIVE.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76887: [EthSwt] EthSwt_SwitchInit shall change the state of a switch to ACTIVE, not the state of the component.

Problem description:

Requirement SWS_EthSwt_00012 states that EthSwt_SwitchInit shall change the state of the component from ETHSWT_STATE_INIT to ETHSWT_STATE_ACTIVE.This translates into the fact that if at least one switch is active, we assume that all switches are active. The subsequent call of a function for a certain switch for which EthSwt_SwitchInit was not called, will assume that the switch is active.

EthSwt_SwithcInit shall change the state of a switch to ACTIVE, not the state of the component.

Agreed solution:

[SWS_EthSwt_00008]
The function EthSwt_Init shall change the state of all switches controlled by this Switch Driver from
ETHSWT_STATE_UNINIT to ETHSWT_STATE_INIT.

[SWS_EthSwt_00012]
EthSwt_SwitchInit shall change the state of the indexed switch from
ETHSWT_STATE_INIT to ETHSWT_STATE_ACTIVE.

SWS_EthSwt_00123
change description of range to
ETHSWT_STATE_UNINIT 0x00 Switch is not yet configured
ETHSWT_STATE_INIT 0x01 Switch is configured
ETHSWT_STATE_ACTIVE 0x02 Switch is active
–Last change on issue 76887 comment 12–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.34 Specification Item SWS_EthSwt_00013

Trace References:

SRS_BSW_00406

Content:

If development error detection is enabled: EthSwt_SwitchInit shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_NOT_INITIALIZED and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

  For Error Detection following general rules apply:
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with

input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.35 Specification Item SWS_EthSwt_00014

Trace References:

SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchIdx is not valid,EthSwt_SwitchInit shall raise the development error ETHSWT_E_INV_SWITCH_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

  For Error Detection following general rules apply:
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
  [SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.

[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.36 Specification Item SWS_EthSwt_00016

Trace References:

SRS_BSW_00386

Content:

The function EthSwt_SwitchInit shall check the access to the Ethernet controllerSwitch hardware, i.e. by trying to read or write registers during the configuration of the switch. If the access to the registers fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76888: [EthSwt] Clarification for ETHSWT_E_ACCESS error

  Problem description:

The SWS_EthSwt_00016 requirement states that "The function EthSwt_SwitchInit shall check the access to the Ethernet controller, i.e. by trying to read or write registers during the configuration of the switch. If the access to the registers fails, the function shall raise the production error ETHSWT_E_ACCESS and return E_NOT_OK".

What is the Eth controller that this requirement is referring to? IF it is the interface used to configure the switch register, the requirement is too restrictive, as certain switch devices can be accessed using the Spi Driver.

Agreed solution:

- Change requirement SWS_EthSwt_00016 to:
[SWS_EthSwt_00016]
The function EthSwt_SwitchInit shall check the access to the Ethernet Switch hardware, i.e. by trying to read or write registers during the configuration of the switch. If the access to the registers fails, the function shall raise the production error ETHSWT_E_ACCESS and return E_NOT_OK.(SRS_BSW_00386)
- Add a note below SWS_EthSwt_00016:
Note: Access to the Ethernet Switch hardware is device dependent, e.g. access through the Ethernet Controller Mii, access through SPI, ... etc.
–Last change on issue 76888 comment 3–

BW-C-Level:

Application	Specification	Bus
1	4	1

- RfC #77250: [EthIf][EthSwt][EthTrcv] EthSwt_SetSwitchPortMode() broken for ports without Ethernet transceiver

  Problem description:

  SWS_EthSwt_00019 states that EthSwt_SetSwitchPortMode() shall (unconditionally) invoke EthTrcv_SetTransceiverMode() of the Ethernet Transceiver Driver.

  This is bogus for the following two reasons:
  1) a particular switch port does not necessarily have an associated Ethernet transceiver. - In that case EthSwtPortTrcvRef won't be present. This is probably the case for MAC2MAC links if EthSwtPortRole is present an either set to ETHSWT_HOST_PORT or to ETHSWT_UP_LINK_PORT.
  2) Even if the particular switch port has an associated Ethernet transceiver, SWS_EthTrcv_00043 states that the EthTrcv shall invoke EthIf_TrcvModeIndication() as a response to EthTrcv_SetTransceiverMode().

  IMHO 1) should be solved by extending SWS_EthSwt_00019 distinguishing between ports which are associated with a transceiver (i.e., EthSwtPortTrcvRef present) - in that case EthTrcv_SetTransceiverMode() should be called - and ports which are not associated with a transceiver (i.e., EthSwtPortTrcvRef is not present) - in that case something else has to happen and this needs to be specified (probably the switch port shall be internally just disabled - maybe this needs to happen in the first case as well?)

2) leads to the following call chain: EthIf_SwitchPortGroupRequestMode() -> EthSwt_SetSwitchPortMode() -> EthTrcv_SetTransceiverMode() -> EthIf_TrcvModeIndication(). Thus the indication actually bypasses the EthSwt which IMHO is extremely ugly. Additionally for switch ports without associated transceiver, the EthIf will never get a EthIf_TrcvModeIndication() ...

Agreed solution:

=== System Template ===
+ Add upstream mapping of EthSwtPortTrcvRef to CouplingPort.physicalLayerType

=== EthSwt ===

~ch. 7.2.4 Production Errors
move SWS_EthSwt_00113 to 7.2.5 Extended production Errors
~SWS_EthSwt_00113 change detection criteria to
Fail:When access to the Ethernet Switch fails the module shall report the extended production error with event status DEM_EVENT_STATUS_PREFAILED to DEM.
Pass:When access to the Ethernet Switch succeeds the module shall report the extended production error with event status DEM_EVENT_STATUS_PREPASSED to DEM.

+ SWS_EthSwt_xxxxx new ext. production Error
Error Name: ETHSWT_E_SYNCPORT2PHY
Short Description: Ethernet switch port and the referenced Ethernet transceiver are in contradicting modes.
Long description: While getting the Ethernet switch port mode, the Ethernet switch driver detected an inconsistent state between Ethernet switch port and the referenced Ethernet transceiver Mode.
Detection Criteria: Fail: When getting the Ethernet switch port mode together with the Ethernet transceiver mode and the mode of the two referenced modules was found
inconsistent the module shall report the extended production error with event status DEM_EVENT_STATUS_PREFAILED to DEM.
Pass:When getting the Ethernet switch port mode together with the Ethernet transceiver mode and the mode of the two referenced modules was found
consistent the module shall report the extended production error with event status

DEM_EVENT_STATUS_PREPASSED to DEM.
all other attributer: N/A

~ ch. 8.3.2 EthSwt_SwitchInit
~[SWS_EthSwt_00016] change "production error" to "extended production error"

~ch. 8.3.3 EthSwt_SetSwitchPortMode
~[SWS_EthSwt_00019]The function EthSwt_SetSwitchPortMode() shall put the indexed port of the switch into the specified mode. If EthSwtPort references an EthTrcv then the function EthTrcv_SetTransceiverMode() of the Ethernet Transceiver Driver shall additionally be called with the corresponding transceiver mode.

+[SWS_EthSwt_xxxxx] When calling the function EthSwt_SetSwitchPortMode() with mode ETHTRCV_MODE_DOWN, the EthSwt shall disable the Ethernet switch port directly for reduction of power consumption, if it is possible.

+[SWS_EthSwt_xxxxx] When calling the function EthSwt_SetSwitchPortMode(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.

+[SWS_EthSwt_xxxxxx] If EthSwtPort does not references an EthTrcv, EthSwt shall indicate a mode of the port by the API EthIf_SwitchPortModeIndication latest during the next EthSwt_MainFunction.(SRS_ETH_00118)

~ch. 10.1.3 EthSwtDemEventParameterRefs
+ SWS Item ECUC_EthSwt_xxxxx1 : Name ETHSWT_E_SYNCPORT2PHY
Parent Container EthSwtDemEventParameterRefs
Description Reference to the DemEventParameter which shall be issued when the error "Ethernet switch port and the referenced Ethernet transceiver are in contradicting modes" has occurred.
Multiplicity 0..1
Type Symbolic name reference to [DemEventParameter]
Post-Build Variant Multiplicity true
Post-Build Variant Value true
Multiplicity Configuration
Class Pre-compile time X VARIANT-PRE-COMPILE
Link time X VARIANT-LINK-TIME
Post-build time X VARIANT-POST-BUILD
Value Configuration Class

Pre-compile time X VARIANT-PRE-COMPILE
Link time X VARIANT-LINK-TIME
Post-build time X VARIANT-POST-BUILD
Scope / Dependency
scope: local

~ch. 10.1.5 EthSwtPort
~ECUC_EthSwt_00041 EthSwtPortTrcvRef change following attributes:
Multiplicity Configuration Class | Link time | VARIANT-LINK-TIME, VARIANT-POST-BUILD
| Post build | –
add to dependency: If EthSwtPortPhysicalLayerType is defined, then EthSwtPortTrcvRef holds the reference to the corresponding EthTrcv.

~ECUC_EthSwt_00054 add dependency to EthSwtPortPhysicalLayerType: If a EthSwtPort has an EthSwtPortPhysicalLayerType then EthSwtPort shall reference an EthTrcv.

=== EthIf ===
Add to chapter 8.4 (callback notifications):
+ EthIf_SwitchPortModeIndication
Syntax: void EthIf_SwitchPortModeIndication(uint8 SwitchIdx, uint8 SwitchPortIdx, EthSwt_ModeType PortMode)
Sevice ID: pick a free one
Sync/Async: Synchronous
Reentrancy: Non Reentrant
Parmaters in: SwitchIdx: Index of the switch within the context of the Ethernet Switch Driver
SwitchPortIdx: Index of the port at the addressed switch
PortMode: notified Ethernet Switch port mode.
Description: The EthIf shall determine the expected notifications based on the EthSwtPort configuration. In case the EthSwtPort references an EthTrcv the EthIf expects a notification from the EthTrcv via API EthIf_TrcvModeIndication(). Otherwise the EthIf expects a notification from the EthSwt via API EthIf_SwitchPortModeIndication().

~SWS_EthIf_00232
change from
Called asynchronously when mode has been read out. Triggered by previous
EthTrcv_SetTransceiverMode call. Can directly be called within the trigger
functions.
To
Called asynchronously when a mode change has been read out. If the function is

triggered by previous call of
EthTrcv_SetTransceiverMode it can directly be called within the trigger function.
–Last change on issue 77250 comment 50–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.37 Specification Item SWS_EthSwt_00018

Trace References:

SRS_ETH_00086 00118

Content:

Service name:	EthSwt_SetSwitchPortModeEthSwt_SetSwitchPortMode	
Syntax:	Std_ReturnType EthSwt_SetSwitchPortMode(uint8 SwitchIdx, uint8 SwitchPortIdx, EthTrcv_ModeType PortMode)	
Service ID[hex]:	0x03	
Sync/Async:	Synchronous /Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	SwitchIdxEthSwt_SetSwitchPort Mode.SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	SwitchPortIdxEthSwt_SetSwitchPort Mode.SwitchPortIdx	Index of the port at the addressed switch
	PortModeEthSwt_SetSwitchPort Mode.PortMode	ETHTRCV_MODE_DOWN: disable the addressed port at the switch ETHTRCV_MODE_ACTIVE: enable the addressed port at the switch
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: success E_NOT_OK: The indexed switch port could not be set to PortMode
Description:	Enables/disables the indexed switch port	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76248: [EthSwt]Remove references to non existing requirements (SRS, SWS)

  Problem description:

Some review findings brought up references to non existing requirements. Please check and remove the references and if needed reformulate the requirements.

Agreed solution:

SWS_EthSwt_00123 –> SRS_BSW_00406
SWS_EthSwt_00165 -> SRS_BSW_00395
SWS_EthSwt_00227 -> SRS_ETH_00087
SWS_EthSwt_91017, SWS_EthSwt_91020, –> SRS_ETH_00123
SWS_EthSwt_00006 –> SRS_BSW_00101
SWS_EthSwt_00010 left blank as SRS doe not yet provide a requirement here but maybe with RFC # 75170 AUTOSAR will introduce an SRS-item after all.
SWS_EthSwt_00376, SWS_EthSwt_00374, SWS_EthSwt_00375 –> SRS-Missing! same as for SWS_EThSwt_00010
SWS_EthSwt_00018, SRS_EthSwt_00025 –> SRS_ETH_00118
SWS_EthSwt_00031 –> SRS_ETH_00087
SWS_EthSwt_00037 –> SRS_ETH_00119
SWS_EthSwt_00044, SWS_EthSwt_00051 –> SRS_ETH_00118
SWS_EthSwt_00060 –> SRS_ETH_00087
SWS_EthSwt_00111 –> erase SRS_ETH_00086
SWS_EthSwt00079 –>SRS_ETH_00119
SWS_EthSwt_00206, SWS_EthSwt_00211, SWS_EthSwt_00216, SWS_EthSwt_00221 –>SRS_Eth_00120
SWS_EthSwt_00172 –>SRS_ETH_00121, SRS_ETH_00114
SWS_EthSwt_00086, SWS_EthSwt_00091, SWS_EthSwt_00182, SWS_EthSwt_00125 –> SRS_ETH_00087, SRS_ETH_00122
SWS_EthSwt_00187 –> SRS_ETH_00087
SWS_EthSwt_00058 –> SRS_BSW_00171
SWS_EthSwt_91012, SWS_EthSwt_91013 –> SRS_ETH_00126
SWS_EthSwt_91014, SWS_EthSwt_91015, SWS_EthSwt_91016, SWS_EthSwt_91018, SWS_EthSwt_91019, SWS_EthSwt_91021, SWS_EthSwt_91022, SWS_EthSwt_91022, SWS_EthSwt_91023, SWS_EthSwt_91024, SWS_EthSwt_91025, SWS_EthSwt_91026, SWS_EthSwt_91027 –> SRS_Eth_00123
SWS_EthSwt_00117, SWS_EthSwt_00203 –> SRS_ETH_00119, SRS_ETH_00087
SWS_EthSwt_00193 –> SRS_ETH_00122, SRS_ETH_00087
SWS_EthSwt_00114 –> SRS_BSW_00433
SWS_EthSwt_00098 –> SRS_Eth_00122, SRS_ETH_00118, SRS_ETH_00119, SRS_ETH_00120, SRS_ETH_00087, SRS_ETH_00125, SRS_BSW_00375
–Last change on issue 76248 comment 13–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.38 Specification Item SWS_EthSwt_00019

Trace References:

SRS_ETH_00118

Content:

The function EthSwt_SetSwitchPortMode() shall put the indexed port of the switch in into the specified modeby calling . If EthSwtPort references an EthTrcv then the function EthTrcv_SetTransceiverMode() of the Ethernet Transceiver Driver shall additionally be called with the corresponding transceiver mode.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77250: [EthIf][EthSwt][EthTrcv] EthSwt_SetSwitchPortMode() broken for ports without Ethernet transceiver

  Problem description:

  SWS_EthSwt_00019 states that EthSwt_SetSwitchPortMode() shall (unconditionally) invoke EthTrcv_SetTransceiverMode() of the Ethernet Transceiver Driver.

  This is bogus for the following two reasons:
  1) a particular switch port does not necessarily have an associated Ethernet transceiver. - In that case EthSwtPortTrcvRef won't be present. This is probably the case for MAC2MAC links if EthSwtPortRole is present an either set to ETHSWT_HOST_PORT or to ETHSWT_UP_LINK_PORT.
  2) Even if the particular switch port has an associated Ethernet transceiver, SWS_EthTrcv_00043 states that the EthTrcv shall invoke EthIf_TrcvModeIndication() as a response to EthTrcv_SetTransceiverMode().

  IMHO 1) should be solved by extending SWS_EthSwt_00019 distinguishing between ports which are associated with a transceiver (i.e., EthSwtPortTrcvRef present) - in that case EthTrcv_SetTransceiverMode() should be called - and ports which are not associated with a transceiver (i.e., EthSwtPortTrcvRef is not present) - in that case something else has to happen and this needs to be specified (probably the switch port shall be internally just disabled - maybe this needs to happen in the first case as well?)

  2) leads to the following call chain: EthIf_SwitchPortGroupRequestMode()

-> EthSwt_SetSwitchPortMode() -> EthTrcv_SetTransceiverMode() -> EthIf_TrcvModeIndication(). Thus the indication actually bypasses the EthSwt which IMHO is extremely ugly. Additionally for switch ports without associated transceiver, the EthIf will never get a EthIf_TrcvModeIndication() ...

Agreed solution:

=== System Template ===
+ Add upstream mapping of EthSwtPortTrcvRef to CouplingPort.physicalLayerType

=== EthSwt ===

~ch. 7.2.4 Production Errors
move SWS_EthSwt_00113 to 7.2.5 Extended production Errors
~SWS_EthSwt_00113 change detection criteria to
Fail:When access to the Ethernet Switch fails the module shall report the extended production error with event status DEM_EVENT_STATUS_PREFAILED to DEM.
Pass:When access to the Ethernet Switch succeeds the module shall report the extended production error with event status DEM_EVENT_STATUS_PREPASSED to DEM.

+ SWS_EthSwt_xxxxx new ext. production Error
Error Name: ETHSWT_E_SYNCPORT2PHY
Short Description: Ethernet switch port and the referenced Ethernet transceiver are in contradicting modes.
Long description: While getting the Ethernet switch port mode, the Ethernet switch driver detected an inconsistent state between Ethernet switch port and the referenced Ethernet transceiver Mode.
Detection Criteria: Fail: When getting the Ethernet switch port mode together with the Ethernet transceiver mode and the mode of the two referenced modules was found
inconsistent the module shall report the extended production error with event status DEM_EVENT_STATUS_PREFAILED to DEM.
Pass:When getting the Ethernet switch port mode together with the Ethernet transceiver mode and the mode of the two referenced modules was found
consistent the module shall report the extended production error with event status DEM_EVENT_STATUS_PREPASSED to DEM.
all other attributer: N/A

~ ch. 8.3.2 EthSwt_SwitchInit
~[SWS_EthSwt_00016] change "production error" to "extended production error"

~ch. 8.3.3 EthSwt_SetSwitchPortMode

~[SWS_EthSwt_00019]The function EthSwt_SetSwitchPortMode() shall put the indexed port of the switch into the specified mode. If EthSwtPort references an EthTrcv then the function EthTrcv_SetTransceiverMode() of the Ethernet Transceiver Driver shall additionally be called with the corresponding transceiver mode.

+[SWS_EthSwt_xxxxx] When calling the function EthSwt_SetSwitchPortMode() with mode ETHTRCV_MODE_DOWN, the EthSwt shall disable the Ethernet switch port directly for reduction of power consumption, if it is possible.

+[SWS_EthSwt_xxxxx] When calling the function EthSwt_SetSwitchPortMode(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.

+[SWS_EthSwt_xxxxxx] If EthSwtPort does not references an EthTrcv, EthSwt shall indicate a mode of the port by the API EthIf_SwitchPortModeIndication latest during the next EthSwt_MainFunction.(SRS_ETH_00118)

~ch. 10.1.3 EthSwtDemEventParameterRefs
+ SWS Item ECUC_EthSwt_xxxxx1 : Name ETHSWT_E_SYNCPORT2PHY
Parent Container EthSwtDemEventParameterRefs
Description Reference to the DemEventParameter which shall be issued when the error "Ethernet switch port and the referenced Ethernet transceiver are in contradicting modes" has occurred.
Multiplicity 0..1
Type Symbolic name reference to [DemEventParameter]
Post-Build Variant Multiplicity true
Post-Build Variant Value true
Multiplicity Configuration
Class Pre-compile time X VARIANT-PRE-COMPILE
Link time X VARIANT-LINK-TIME
Post-build time X VARIANT-POST-BUILD
Value Configuration Class
Pre-compile time X VARIANT-PRE-COMPILE
Link time X VARIANT-LINK-TIME
Post-build time X VARIANT-POST-BUILD
Scope / Dependency
scope: local

~ch. 10.1.5 EthSwtPort
~ECUC_EthSwt_00041 EthSwtPortTrcvRef change following attributes:

Multiplicity Configuration Class | Link time | VARIANT-LINK-TIME, VARIANT-POST-BUILD
| Post build | –
add to dependency: If EthSwtPortPhysicalLayerType is defined, then EthSwtPortTrcvRef holds the reference to the corresponding EthTrcv.

~ECUC_EthSwt_00054 add dependency to EthSwtPortPhysicalLayerType: If a EthSwtPort has an EthSwtPortPhysicalLayerType then EthSwtPort shall reference an EthTrcv.

=== EthIf ===
Add to chapter 8.4 (callback notifications):
+ EthIf_SwitchPortModeIndication
Syntax: void EthIf_SwitchPortModeIndication(uint8 SwitchIdx, uint8 SwitchPortIdx, EthSwt_ModeType PortMode)
Sevice ID: pick a free one
Sync/Async: Synchronous
Reentrancy: Non Reentrant
Parmaters in: SwitchIdx: Index of the switch within the context of the Ethernet Switch Driver
SwitchPortIdx: Index of the port at the addressed switch
PortMode: notified Ethernet Switch port mode.
Description: The EthIf shall determine the expected notifications based on the EthSwtPort configuration. In case the EthSwtPort references an EthTrcv the EthIf expects a notification from the EthTrcv via API EthIf_TrcvModeIndication(). Otherwise the EthIf expects a notification from the EthSwt via API EthIf_SwitchPortModeIndication().

~SWS_EthIf_00232
change from
Called asynchronously when mode has been read out. Triggered by previous
EthTrcv_SetTransceiverMode call. Can directly be called within the trigger
functions.
To
Called asynchronously when a mode change has been read out. If the function is triggered by previous call of
EthTrcv_SetTransceiverMode it can directly be called within the trigger function.
–Last change on issue 77250 comment 50–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.39 Specification Item SWS_EthSwt_00020

Trace References:

SRS_BSW_00406

Content:

If development error detection is enabled: the function EthSwt_SetSwitchPortMode shall check that the service EthSwt_SwitchInit was previously called. If the check fails, the function shall raise the development error ETHSWT_E_NOT_INITIALIZED and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.40 Specification Item SWS_EthSwt_00021

Trace References:

SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchIdx is not valid,EthSwt_SetSwitchPortMode shall raise the development error ETHSWT_E_INV_SWITCH_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

  For Error Detection following general rules apply:
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with in-

put parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.41 Specification Item SWS_EthSwt_00022

Trace References:

SRS_BSW_00171

Content:

The function EthSwt_SetSwitchPortMode shall be pre compile time configurable On/Off by the configuration parameter: EthTrcvSwtSetTransceiverSwitchPortModeApi.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77043: [EthSwt] EthSwt APIs shouldnt be precompile ON/OFF based on EthTrcv parameters

  Problem description:

  For the APIs EthSwt_SetSwitchPortMode, EthSwt_GetSwitchPortMode, EthSwt_StartSwitchPortAutoNegotiation, EthSwt_GetLinkState, EthSwt_GetBaudRate and EthSwt_GetDuplexMode there is a requirement that states that the API should be configurable On/Off based on a EthTrcv parameter. For example for EthSwt_StartSwitchPortAutoNegotiation the requirement(SWS_EthSwt_00035) is: The function EthSwt_StartSwitchPortAutoNegotiation shall be pre-compile time configurable On/Off by the configuration parameter: EthTrcvStartAutoNegotiationApi.

  However, for other similar APIs a configuration parameter was defined in the switch driver. The functions are: EthSwt_GetPortSignalQuality, EthSwt_GetPortIdentifier, EthSwt_SetPortLoopbackMode, EthSwt_SetPortTxMode, EthSwt_GetPortCableDiagnosticsResult.

  Also, if multiple EthTrcv drivers are used, how will the precompile parameters be defined in the switch?

  Agreed solution:

  add parameters
  EthSwtSetSwitchPortModeApi, EthSwtGetSwitchPortModeApi, EthSwtStartSwitchPortAutoNegotiationApi, EthSwtGetLinkStateApi, EthSwtGetBaudRateApi and EthSwtGetDuplexModeApi
  Description: Enables / disables the EthSwt_SetSwitchPortMode / EthSwt_GetSwitchPortMode / EthSwt_StartSwitchPortAutoNegotiation / EthSwt_GetLinkState/ EthSwt_GetBaudRate / EthSwt_GetDuplexMode
  Multiplicity: 1
  Type EcucBooleanParamDef
  Default: False
  Variant false
  Config Class Pre-compile for all variants

  ~[SWS_EthSwt_00022]
  replace EthTrcvSetTransceiverModeApi with EthSwtSetSwitchPortModeApi.
  ~SWS_EthSwt_00029
  replace EthTrcvGetTransceiverModeApi with EthSwtGetSwitchPortModeApi.
  ~SWS_EthSwt_00035
  replace EthTrcvStartAutoNegotiationApi with EthSwtStartSwitchPortAutoNegotia-

tionApi
~[SWS_EthSwt_00042]
replace EthTrcvGetLinkStateApi with EthSwtGetLinkStateApi
~SWS_EthSwt_00049
replace EthTrcvGetBaudRateApi with EthSwtGetBaudRateApi
~SWS_EthSwt_00056
replace EthTrcvGetDuplexModeApi with EthSwtGetDuplexModeApi
–Last change on issue 77043 comment 2–

BW-C-Level:

Application	Specification	Bus
1	3	1

1.42 Specification Item SWS_EthSwt_00025

Trace References:

SRS_ETH_00086 00118

Content:

Service name:	EthSwt_GetSwitchPortModeEthSwt_GetSwitchPortMode	
Syntax:	Std_ReturnType EthSwt_GetSwitchPortMode(uint8 SwitchIdx, uint8 SwitchPortIdx, EthTrcv_ModeType* SwitchModePtr)	
Service ID[hex]:	0x04	
Sync/Async:	Synchronous /Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	SwitchIdxEthSwt_GetSwitchPort Mode.SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	SwitchPortIdxEthSwt_GetSwitchPort Mode.SwitchPortIdx	Index of the port at the addressed switch
Parameters (inout):	None	
Parameters (out):	SwitchModePtrEthSwt_GetSwitchPort Mode.SwitchModePtr	ETHTRCV_MODE_DOWN: the port of the switch is disabled ETHTRCV_MODE_ACTIVE: the port of the switch is enabled
Return value:	Std_ReturnType	E_OK: success E_NOT_OK: The mode of the indexed switch port could not be obtained.
Description:	Obtains the mode of the indexed switch port	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76879: [EthTrcv] Add EthSwt API's to optional interfaces

  Problem description:

  Optional interfaces are missing, in case the Ethernet transceiver is connected to a Ethernet switch

  Agreed solution:

  add the following API's to optional interfaces:
  EthSwt_ReadTrcvRegister
  EthSwt_WriteTrcvRegister

  move the following API's from mandatory interfaces to optional interfaces:
  Eth_ReadMii
  Eth_WriteMii
  –Last change on issue 76879 comment 1–

  BW-C-Level:

Application	Specification	Bus
1	1	1

1.43 Specification Item SWS_EthSwt_00026

Trace References:

SRS_ETH_00118

Content:

The function EthSwt_GetSwitchPortMode

shall read the mode of the indexed port of the switchby calling . If EthSwtPort references an EthTrcv then the function shall additionally call the corresponding function EthTrcv_GetTransceiverMode of the Ethernet Transceiver Driver.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77497: [EthIf][EthSwt][EthTrcv] EthSwt_GetSwitchPortMode() broken for ports without Ethernet transceiver

  Problem description:

  RfC # 77250 introduce the correction for the API EthSwt_SetSwitchPortMode(). Thus, also the EthSwt_GetSwitchPortMode)() has to be revised.

  Agreed solution:

change [SWS_EthSwt_00026] to
The function EthSwt_GetSwitchPortMode shall read the mode of the indexed port of the switch. If EthSwtPort references an EthTrcv then the function shall additionally call the corresponding function EthTrcv_GetTransceiverMode of the Ethernet Transceiver Driver.

+[SWS_EthSwt_xxxxx] If the obtained modes of the EthSwtPort and the EthTrcv are not aligned, the function EthSwt_GetSwitchPortMode shall raise the extended production error ETHSWT_E_SYNCPORT2PHY and return E_NOT_OK.
If EthTrcv_GetTransceiverMode returns E_NOT_OK, the EthSwt_GetSwitchPortMode shall also return E_NOT_OK without raising an error.

+[SWS_EthSwt_xxxxx] If the function EthSwt_GetSwitchPortMode() is called, the function shall check the access to the Ethernet Switch Driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the production error ETHSWT_E_ACCESS and return E_OK.
–Last change on issue 77497 comment 12–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.44 Specification Item SWS_EthSwt_00027

Trace References:

SRS_BSW_00406

Content:

If development error detection is enabled: the function EthSwt_GetSwitchPortMode shall check that the service EthSwt_SwitchInit was previously called. If the check fails, the function shall raise the development error ETHSWT_E_NOT_INITIALIZED and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return

E_NOT_OK if the driver is not yet initialized. ()

In the above requirement -
1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being

valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.45 Specification Item SWS_EthSwt_00028

Trace References:

SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchIdx is not valid,EthSwt_GetSwitchPortMode shall raise the development error ETHSWT_E_INV_SWITCH_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API

of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.46 Specification Item SWS_EthSwt_00029

Trace References:

SRS_BSW_00171

Content:

The function EthSwt_GetSwitchPortMode shall be pre compile time configurable On/Off by the configuration parameter: EthTrcvSwtGetTransceiverSwitchPortModeApi.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77043: [EthSwt] EthSwt APIs shouldnt be precompile ON/OFF based on EthTrcv parameters

  Problem description:

  For the APIs EthSwt_SetSwitchPortMode, EthSwt_GetSwitchPortMode, EthSwt_StartSwitchPortAutoNegotiation, EthSwt_GetLinkState, EthSwt_GetBaudRate and EthSwt_GetDuplexMode there is a requirement that states that the API should be configurable On/Off based on a EthTrcv parameter. For example for EthSwt_StartSwitchPortAutoNegotiation the requirement(SWS_EthSwt_00035) is: The function EthSwt_StartSwitchPortAutoNegotiation shall be pre-compile time configurable On/Off by the configuration parameter: EthTrcvStartAutoNegotiationApi.

  However, for other similar APIs a configuration parameter was defined in the switch driver. The functions are: EthSwt_GetPortSignalQuality, EthSwt_GetPortIdentifier, EthSwt_SetPortLoopbackMode, EthSwt_SetPortTxMode, EthSwt_GetPortCableDiagnosticsResult.

Also, if multiple EthTrcv drivers are used, how will the precompile parameters be defined in the switch?

Agreed solution:

add parameters
EthSwtSetSwitchPortModeApi, EthSwtGetSwitchPortModeApi, EthSwtStartSwitchPortAutoNegotiationApi, EthSwtGetLinkStateApi, EthSwtGetBaudRateApi and EthSwtGetDuplexModeApi
Description: Enables / disables the EthSwt_SetSwitchPortMode / EthSwt_GetSwitchPortMode / EthSwt_StartSwitchPortAutoNegotiation / EthSwt_GetLinkState/ EthSwt_GetBaudRate / EthSwt_GetDuplexMode
Multiplicity: 1
Type EcucBooleanParamDef
Default: False
Variant false
Config Class Pre-compile for all variants

~[SWS_EthSwt_00022]
replace EthTrcvSetTransceiverModeApi with EthSwtSetSwitchPortModeApi.
~SWS_EthSwt_00029
replace EthTrcvGetTransceiverModeApi with EthSwtGetSwitchPortModeApi.
~SWS_EthSwt_00035
replace EthTrcvStartAutoNegotiationApi with EthSwtStartSwitchPortAutoNegotiationApi
~[SWS_EthSwt_00042]
replace EthTrcvGetLinkStateApi with EthSwtGetLinkStateApi
~SWS_EthSwt_00049
replace EthTrcvGetBaudRateApi with EthSwtGetBaudRateApi
~SWS_EthSwt_00056
replace EthTrcvGetDuplexModeApi with EthSwtGetDuplexModeApi
–Last change on issue 77043 comment 2–

BW-C-Level:

Application	Specification	Bus
1	3	1

1.47 Specification Item SWS_EthSwt_00031

Trace References:

SRS_ETH_00086 00087

Content:

Service name:	EthSwt_StartSwitchPortAutoNegotiationEthSwt_StartSwitchPortAutoNegotiation	
Syntax:	Std_ReturnType EthSwt_StartSwitchPortAutoNegotiation(uint8 SwitchIdx, uint8 SwitchPortIdx)	
Service ID[hex]:	0x05	
Sync/Async:	Asynchronous /Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	SwitchIdxEthSwt_StartSwitchPortAutoNegotiation.SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	SwitchPortIdxEthSwt_StartSwitchPortAutoNegotiation.SwitchPortIdx	Index of the port at the addressed switch
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: success E_NOT_OK: Automatic negotiation could not be started for the indexed switch port.
Description:	Starts the auto-negotiation of the indexed switch port	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76248: [EthSwt]Remove references to non existing requirements (SRS, SWS)

  Problem description:

  Some review findings brought up references to non existing requirements. Please check and remove the references and if needed reformulate the requirements.

  Agreed solution:

  SWS_EthSwt_00123 –> SRS_BSW_00406
  SWS_EthSwt_00165 -> SRS_BSW_00395
  SWS_EthSwt_00227 -> SRS_ETH_00087
  SWS_EthSwt_91017, SWS_EthSwt_91020, –> SRS_ETH_00123
  SWS_EthSwt_00006 –> SRS_BSW_00101
  SWS_EthSwt_00010 left blank as SRS doe not yet provide a requirement here but maybe with RFC # 75170 AUTOSAR will introduce an SRS-item after all.
  SWS_EthSwt_00376, SWS_EthSwt_00374, SWS_EthSwt_00375 –> SRS-Missing! same as for SWS_EThSwt_00010
  SWS_EthSwt_00018, SRS_EthSwt_00025 –> SRS_ETH_00118
  SWS_EthSwt_00031 –> SRS_ETH_00087
  SWS_EthSwt_00037 –> SRS_ETH_00119
  SWS_EthSwt_00044, SWS_EthSwt_00051 –> SRS_ETH_00118

SWS_EthSwt_00060 –> SRS_ETH_00087
SWS_EthSwt_00111 –> erase SRS_ETH_00086
SWS_EthSwt00079 –>SRS_ETH_00119
SWS_EthSwt_00206, SWS_EthSwt_00211, SWS_EthSwt_00216, SWS_EthSwt_00221 –>SRS_Eth_00120
SWS_EthSwt_00172 –>SRS_ETH_00121, SRS_ETH_00114
SWS_EthSwt_00086, SWS_EthSwt_00091, SWS_EthSwt_00182, SWS_EthSwt_00125 –> SRS_ETH_00087, SRS_ETH_00122
SWS_EthSwt_00187 –> SRS_ETH_00087
SWS_EthSwt_00058 –> SRS_BSW_00171
SWS_EthSwt_91012, SWS_EthSwt_91013 –> SRS_ETH_00126
SWS_EthSwt_91014, SWS_EthSwt_91015, SWS_EthSwt_91016, SWS_EthSwt_91018, SWS_EthSwt_91019, SWS_EthSwt_91021, SWS_EthSwt_91022, SWS_EthSwt_91022, SWS_EthSwt_91023, SWS_EthSwt_91024, SWS_EthSwt_91025, SWS_EthSwt_91026, SWS_EthSwt_91027 –> SRS_Eth_00123
SWS_EthSwt_00117, SWS_EthSwt_00203 –> SRS_ETH_00119, SRS_ETH_00087
SWS_EthSwt_00193 –> SRS_ETH_00122, SRS_ETH_00087
SWS_EthSwt_00114 –> SRS_BSW_00433
SWS_EthSwt_00098 –> SRS_Eth_00122, SRS_ETH_00118, SRS_ETH_00119, SRS_ETH_00120, SRS_ETH_00087, SRS_ETH_00125, SRS_BSW_00375
–Last change on issue 76248 comment 13–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.48 Specification Item SWS_EthSwt_00032

Trace References:

SRS_ETH_00087

Content:

The function EthSwt_StartSwitchPortAutoNegotiation shall restart the automatic negotiation of the transmission parameters used used transmission parameters of the referenced Ethernet transceiver driver by calling the API function EthTrcv_StartAutoNegotiationby the indexed transceiver().

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77664: [EthSwt] Behaviour of certain APIs for ports without EthSwtPortTrcvRef (part II)

  Problem description:

  The following APIs have no description regarding the handling of ports without EthSwtPortTrcvRef:
  EthSwt_GetLinkState
  EthSwt_GetBaudRate
  EthSwt_GetDuplexMode

  The description should be adjusted and harmonized with related requirements.

  I would also add the following function:
  EthSwt_StartSwitchPortAutoNegotiation
  I rate this function not to be used for ports without a EthSwtPortTrcvRef. For the sake of completeness I would also reformulate some requirements.
  –Last change on issue 77664 comment 10–

  Agreed solution:

  === EthSwt ===
  — EthSwt_GetLinkState —
  ~[SWS_EthSwt_00038]
  The function EthSwt_GetLinkState shall read the current (link) state of the indexed switch port. If the indexed Ethernet port references an Ethernet transceiver, the link state shall be obtained by calling the function EthTrcv_GetLinkState() of the Ethernet Transceiver Driver. If the indexed Ethernet Switch port does not reference an Ethernet transceiver, the state shall be obtained from the MAC interface of the Switch port.

  — EthSwt_GetBaudRate —
  ~[SWS_EthSwt_00045]
  The function EthSwt_GetBaudRate() shall read the current baud rate of the indexed switch port. If the indexed Ethernet port reference an Ethernet transceiver, the baud rate shall be obtained by the function EthTrcv_GetBaudRate() of the Ethernet Transceiver Driver. If the indexed Ethernet Switch port does not reference an Ethernet transceiver, the baud rate shall be obtained from the MAC interface of the Switch port.
  -[SWS_EthSwt_00153] as # 77349 introduces a general req to this.

  — EthSwt_GetDuplexMode —
  ~[SWS_EthSwt_00052]
  The function EthSwt_GetDuplexMode() shall read the current duplex mode of the

indexed switch port. If the indexed Ethernet port reference an Ethernet transceiver, the duplex mode shall be obtained by calling the function EthTrcv_ GetDuplexMode() of the Ethernet Transceiver Driver. If the indexed Ethernet Switch port does not reference an Ethernet transceiver, the duplex mode shall be obtained from the MAC interface of the Switch port.
-[SWS_EthSwt_00155] as # 77349 introduces a general req to this.

— EthSwt_StartSwitchPortAutoNegotiation —
~[SWS_EthSwt_00032]
The function EthSwt_StartSwitchPortAutoNegotiation shall restart the automatic negotiation of the used transmission parameters of the referenced Ethernet transceiver driver by calling the function EthTrcv_StartAutoNegotiation().
-[SWS_EthSwt_00158] as # 77349 introduces a general req to this.
–Last change on issue 77664 comment 26–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.49 Specification Item SWS_EthSwt_00033

Trace References:

SRS_BSW_00406

Content:

If development error detection is enabled: the function EthSwt_StartSwitchPortAutoNegotiation shall check that the service EthSwt_SwitchInit was previously called. If the check fails, the function shall raise the development error ETHSWT_E_NOT_INITIALIZED and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most

probably should be development error detection.

Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain

the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.50 Specification Item SWS_EthSwt_00034

Trace References:

SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchIdx is not valid,EthSwt_StartSwitchPortAutoNegotiation shall raise the development error ETHSWT_E_INV_SWITCH_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to

If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.51 Specification Item SWS_EthSwt_00035

Trace References:

SRS_BSW_00171

Content:

The function EthSwt_StartSwitchPortAutoNegotiation shall be pre compile time configurable On/Off by the configuration parameter: EthTrcvStartSwtStartSwitchPortAutoNegotiationApi.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77043: [EthSwt] EthSwt APIs shouldnt be precompile ON/OFF based on EthTrcv parameters

  Problem description:

  For the APIs EthSwt_SetSwitchPortMode, EthSwt_GetSwitchPortMode, EthSwt_StartSwitchPortAutoNegotiation, EthSwt_GetLinkState, EthSwt_GetBaudRate and EthSwt_GetDuplexMode there is a requirement that states that the API should be configurable On/Off based on a EthTrcv parameter. For example for EthSwt_StartSwitchPortAutoNegotiation the requirement(SWS_EthSwt_00035) is: The function EthSwt_StartSwitchPortAutoNegotiation shall be pre-compile time configurable On/Off by the configuration parameter: EthTrcvStartAutoNegotiationApi.

  However, for other similar APIs a configuration parameter was defined in the switch driver. The functions are: EthSwt_GetPortSignalQuality, EthSwt_GetPortIdentifier, EthSwt_SetPortLoopbackMode, EthSwt_SetPortTxMode, EthSwt_GetPortCableDiagnosticsResult.

Also, if multiple EthTrcv drivers are used, how will the precompile parameters be defined in the switch?

Agreed solution:

add parameters
EthSwtSetSwitchPortModeApi, EthSwtGetSwitchPortModeApi, EthSwtStartSwitchPortAutoNegotiationApi, EthSwtGetLinkStateApi, EthSwtGetBaudRateApi and EthSwtGetDuplexModeApi
Description: Enables / disables the EthSwt_SetSwitchPortMode / EthSwt_GetSwitchPortMode / EthSwt_StartSwitchPortAutoNegotiation / EthSwt_GetLinkState/ EthSwt_GetBaudRate / EthSwt_GetDuplexMode
Multiplicity: 1
Type EcucBooleanParamDef
Default: False
Variant false
Config Class Pre-compile for all variants

~[SWS_EthSwt_00022]
replace EthTrcvSetTransceiverModeApi with EthSwtSetSwitchPortModeApi.
~SWS_EthSwt_00029
replace EthTrcvGetTransceiverModeApi with EthSwtGetSwitchPortModeApi.
~SWS_EthSwt_00035
replace EthTrcvStartAutoNegotiationApi with EthSwtStartSwitchPortAutoNegotiationApi
~[SWS_EthSwt_00042]
replace EthTrcvGetLinkStateApi with EthSwtGetLinkStateApi
~SWS_EthSwt_00049
replace EthTrcvGetBaudRateApi with EthSwtGetBaudRateApi
~SWS_EthSwt_00056
replace EthTrcvGetDuplexModeApi with EthSwtGetDuplexModeApi
–Last change on issue 77043 comment 2–

BW-C-Level:

Application	Specification	Bus
1	3	1

1.52 Specification Item SWS_EthSwt_00037

Trace References:

SRS_ETH_00086 00119

Content:

Service name:	EthSwt_GetLinkStateEthSwt_GetLinkState	
Syntax:	Std_ReturnType EthSwt_GetLinkState(uint8 SwitchIdx, uint8 SwitchPortIdx, EthTrcv_LinkStateType* LinkStatePtr)	
Service ID[hex]:	0x06	
Sync/Async:	Synchronous /Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	SwitchIdxEthSwt_GetLinkState.SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	SwitchPortIdxEthSwt_GetLinkState.SwitchPortIdx	Index of the port at the addressed switch
Parameters (inout):	None	
Parameters (out):	LinkStatePtrEthSwt_GetLinkState.LinkStatePtr	ETHSWT_LINK_STATE_DOWN: Switch port is disconnected ETHSWT_LINK_STATE_ACTIVE: Switch port is connected
Return value:	Std_ReturnType	E_OK: success E_NOT_OK: Link state of the indexed switch port could not be obtained
Description:	Obtains the link state of the indexed switch port	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76248: [EthSwt]Remove references to non existing requirements (SRS, SWS)

  Problem description:

  Some review findings brought up references to non existing requirements. Please check and remove the references and if needed reformulate the requirements.

  Agreed solution:

  SWS_EthSwt_00123 –> SRS_BSW_00406
  SWS_EthSwt_00165 -> SRS_BSW_00395
  SWS_EthSwt_00227 -> SRS_ETH_00087
  SWS_EthSwt_91017, SWS_EthSwt_91020, –> SRS_ETH_00123
  SWS_EthSwt_00006 –> SRS_BSW_00101
  SWS_EthSwt_00010 left blank as SRS doe not yet provide a requirement here but maybe with RFC # 75170 AUTOSAR will introduce an SRS-item after all.
  SWS_EthSwt_00376, SWS_EthSwt_00374, SWS_EthSwt_00375 –> SRS-Missing! same as for SWS_EThSwt_00010
  SWS_EthSwt_00018, SRS_EthSwt_00025 –> SRS_ETH_00118
  SWS_EthSwt_00031 –> SRS_ETH_00087
  SWS_EthSwt_00037 –> SRS_ETH_00119

SWS_EthSwt_00044, SWS_EthSwt_00051 –> SRS_ETH_00118
SWS_EthSwt_00060 –> SRS_ETH_00087
SWS_EthSwt_00111 –> erase SRS_ETH_00086
SWS_EthSwt00079 –>SRS_ETH_00119
SWS_EthSwt_00206, SWS_EthSwt_00211, SWS_EthSwt_00216, SWS_EthSwt_00221 –>SRS_Eth_00120
SWS_EthSwt_00172 –>SRS_ETH_00121, SRS_ETH_00114
SWS_EthSwt_00086, SWS_EthSwt_00091, SWS_EthSwt_00182, SWS_EthSwt_00125 –> SRS_ETH_00087, SRS_ETH_00122
SWS_EthSwt_00187 –> SRS_ETH_00087
SWS_EthSwt_00058 –> SRS_BSW_00171
SWS_EthSwt_91012, SWS_EthSwt_91013 –> SRS_ETH_00126
SWS_EthSwt_91014, SWS_EthSwt_91015, SWS_EthSwt_91016, SWS_EthSwt_91018, SWS_EthSwt_91019, SWS_EthSwt_91021, SWS_EthSwt_91022, SWS_EthSwt_91022, SWS_EthSwt_91023, SWS_EthSwt_91024, SWS_EthSwt_91025, SWS_EthSwt_91026, SWS_EthSwt_91027 –> SRS_Eth_00123
SWS_EthSwt_00117, SWS_EthSwt_00203 –> SRS_ETH_00119, SRS_ETH_00087
SWS_EthSwt_00193 –> SRS_ETH_00122, SRS_ETH_00087
SWS_EthSwt_00114 –> SRS_BSW_00433
SWS_EthSwt_00098 –> SRS_Eth_00122, SRS_ETH_00118, SRS_ETH_00119, SRS_ETH_00120, SRS_ETH_00087, SRS_ETH_00125, SRS_BSW_00375
–Last change on issue 76248 comment 13–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.53 Specification Item SWS_EthSwt_00038

Trace References:

SRS_ETH_00118, SRS_ETH_00119

Content:

The function EthSwt_GetLinkState shall read the current link (link) state of the indexed switch port. If the indexed Ethernet port references an Ethernet transceiver, the link state shall be obtained by calling the corresponding function EthTrcv_GetLinkState() of the Ethernet Transceiver Driver. If the indexed Ethernet Switch port does not reference an Ethernet transceiver, the state shall be obtained from the MAC interface of the Switch port.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77664: [EthSwt] Behaviour of certain APIs for ports without EthSwtPortTrcvRef (part II)

  Problem description:

  The following APIs have no description regarding the handling of ports without EthSwtPortTrcvRef:
  EthSwt_GetLinkState
  EthSwt_GetBaudRate
  EthSwt_GetDuplexMode

  The description should be adjusted and harmonized with related requirements.

  I would also add the following function:
  EthSwt_StartSwitchPortAutoNegotiation
  I rate this function not to be used for ports without a EthSwtPortTrcvRef. For the sake of completeness I would also reformulate some requirements.
  –Last change on issue 77664 comment 10–

  Agreed solution:

  === EthSwt ===
  — EthSwt_GetLinkState —
  ~[SWS_EthSwt_00038]
  The function EthSwt_GetLinkState shall read the current (link) state of the indexed switch port. If the indexed Ethernet port references an Ethernet transceiver, the link state shall be obtained by calling the function EthTrcv_GetLinkState() of the Ethernet Transceiver Driver. If the indexed Ethernet Switch port does not reference an Ethernet transceiver, the state shall be obtained from the MAC interface of the Switch port.

  — EthSwt_GetBaudRate —
  ~[SWS_EthSwt_00045]
  The function EthSwt_GetBaudRate() shall read the current baud rate of the indexed switch port. If the indexed Ethernet port reference an Ethernet transceiver, the baud rate shall be obtained by the function EthTrcv_GetBaudRate() of the Ethernet Transceiver Driver. If the indexed Ethernet Switch port does not reference an Ethernet transceiver, the baud rate shall be obtained from the MAC interface of the Switch port.
  -[SWS_EthSwt_00153] as # 77349 introduces a general req to this.

  — EthSwt_GetDuplexMode —

~[SWS_EthSwt_00052]
The function EthSwt_GetDuplexMode() shall read the current duplex mode of the indexed switch port. If the indexed Ethernet port reference an Ethernet transceiver, the duplex mode shall be obtained by calling the function EthTrcv_ GetDuplexMode() of the Ethernet Transceiver Driver. If the indexed Ethernet Switch port does not reference an Ethernet transceiver, the duplex mode shall be obtained from the MAC interface of the Switch port.
-[SWS_EthSwt_00155] as # 77349 introduces a general req to this.

— EthSwt_StartSwitchPortAutoNegotiation —
~[SWS_EthSwt_00032]
The function EthSwt_StartSwitchPortAutoNegotiation shall restart the automatic negotiation of the used transmission parameters of the referenced Ethernet transceiver driver by calling the function EthTrcv_StartAutoNegotiation().
-[SWS_EthSwt_00158] as # 77349 introduces a general req to this.
–Last change on issue 77664 comment 26–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.54 Specification Item SWS_EthSwt_00039

Trace References:

SRS_BSW_00406

Content:

If development error detection is enabled: the function EthSwt_GetLinkState shall check that the service EthSwt_SwitchInit was previously called. If the check fails, the function shall raise the development error ETHSWT_E_NOT_INITIALIZED and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

In the above requirement -
1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.

[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.55 Specification Item SWS_EthSwt_00040

Trace References:

SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchIdx is not valid, EthSwt_GetLinkState shall raise the development error ETHSWT_E_INV_SWITCH_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not de-

fault errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.56 Specification Item SWS_EthSwt_00042

Trace References:

SRS_BSW_00171

Content:

The function EthSwt_GetLinkState shall be pre compile time configurable On/Off by the configuration parameter: EthTrcvSwtGetLinkStateApi.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77043: [EthSwt] EthSwt APIs shouldnt be precompile ON/OFF based on EthTrcv parameters

  Problem description:

  For the APIs EthSwt_SetSwitchPortMode, EthSwt_GetSwitchPortMode, EthSwt_StartSwitchPortAutoNegotiation, EthSwt_GetLinkState, EthSwt_GetBaudRate and EthSwt_GetDuplexMode there is a requirement that states that the API should be configurable On/Off based on a EthTrcv parameter. For example for EthSwt_StartSwitchPortAutoNegotiation the requirement(SWS_EthSwt_00035) is: The function EthSwt_StartSwitchPortAutoNegotiation shall be pre-compile time configurable On/Off by the configuration parameter: EthTrcvStartAutoNegotiationApi.

  However, for other similar APIs a configuration parameter was defined in the switch driver. The functions are: EthSwt_GetPortSignalQuality, EthSwt_GetPortIdentifier, EthSwt_SetPortLoopbackMode, EthSwt_SetPortTxMode, EthSwt_GetPortCableDiagnosticsResult.

Also, if multiple EthTrcv drivers are used, how will the precompile parameters be defined in the switch?

Agreed solution:

add parameters
EthSwtSetSwitchPortModeApi, EthSwtGetSwitchPortModeApi, EthSwtStartSwitchPortAutoNegotiationApi, EthSwtGetLinkStateApi, EthSwtGetBaudRateApi and EthSwtGetDuplexModeApi
Description: Enables / disables the EthSwt_SetSwitchPortMode / EthSwt_GetSwitchPortMode / EthSwt_StartSwitchPortAutoNegotiation / EthSwt_GetLinkState/ EthSwt_GetBaudRate / EthSwt_GetDuplexMode
Multiplicity: 1
Type EcucBooleanParamDef
Default: False
Variant false
Config Class Pre-compile for all variants

~[SWS_EthSwt_00022]
replace EthTrcvSetTransceiverModeApi with EthSwtSetSwitchPortModeApi.
~SWS_EthSwt_00029
replace EthTrcvGetTransceiverModeApi with EthSwtGetSwitchPortModeApi.
~SWS_EthSwt_00035
replace EthTrcvStartAutoNegotiationApi with EthSwtStartSwitchPortAutoNegotiationApi
~[SWS_EthSwt_00042]
replace EthTrcvGetLinkStateApi with EthSwtGetLinkStateApi
~SWS_EthSwt_00049
replace EthTrcvGetBaudRateApi with EthSwtGetBaudRateApi
~SWS_EthSwt_00056
replace EthTrcvGetDuplexModeApi with EthSwtGetDuplexModeApi
–Last change on issue 77043 comment 2–

BW-C-Level:

Application	Specification	Bus
1	3	1

1.57 Specification Item SWS_EthSwt_00044

Trace References:

SRS_ETH_00086 00118

Content:

Service name:	EthSwt_GetBaudRateEthSwt_GetBaudRate	
Syntax:	Std_ReturnType EthSwt_GetBaudRate(uint8 SwitchIdx, uint8 SwitchPortIdx, EthTrcv_BaudRateType* BaudRatePtr)	
Service ID[hex]:	0x07	
Sync/Async:	Synchronous /Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	SwitchIdxEthSwt_GetBaudRate.SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	SwitchPortIdxEthSwt_GetBaudRate.SwitchPortIdx	Index of the port at the addressed switch
Parameters (inout):	None	
Parameters (out):	BaudRatePtrEthSwt_GetBaudRate.BaudRatePtr	ETHTRCV_BAUD_RATE_10MBIT: 10MBit connection ETHTRCV_BAUD_RATE_100MBIT: 100MBit connection ETHTRCV_BAUD_RATE_1000MBIT: 1000MBit connection
Return value:	Std_ReturnType	E_OK: success E_NOT_OK: Baud rate of the indexed switch port could not be obtained
Description:	Obtains the baud rate of the indexed switch port	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76248: [EthSwt]Remove references to non existing requirements (SRS, SWS)

  Problem description:

  Some review findings brought up references to non existing requirements. Please check and remove the references and if needed reformulate the requirements.

  Agreed solution:

  SWS_EthSwt_00123 –> SRS_BSW_00406
  SWS_EthSwt_00165 -> SRS_BSW_00395
  SWS_EthSwt_00227 -> SRS_ETH_00087
  SWS_EthSwt_91017, SWS_EthSwt_91020, –> SRS_ETH_00123
  SWS_EthSwt_00006 –> SRS_BSW_00101
  SWS_EthSwt_00010 left blank as SRS doe not yet provide a requirement here but maybe with RFC # 75170 AUTOSAR will introduce an SRS-item after all.
  SWS_EthSwt_00376, SWS_EthSwt_00374, SWS_EthSwt_00375 –> SRS-Missing! same as for SWS_EThSwt_00010
  SWS_EthSwt_00018, SRS_EthSwt_00025 –> SRS_ETH_00118
  SWS_EthSwt_00031 –> SRS_ETH_00087

SWS_EthSwt_00037 –> SRS_ETH_00119
SWS_EthSwt_00044, SWS_EthSwt_00051 –> SRS_ETH_00118
SWS_EthSwt_00060 –> SRS_ETH_00087
SWS_EthSwt_00111 –> erase SRS_ETH_00086
SWS_EthSwt00079 –>SRS_ETH_00119
SWS_EthSwt_00206, SWS_EthSwt_00211, SWS_EthSwt_00216, SWS_EthSwt_00221 –>SRS_Eth_00120
SWS_EthSwt_00172 –>SRS_ETH_00121, SRS_ETH_00114
SWS_EthSwt_00086, SWS_EthSwt_00091, SWS_EthSwt_00182, SWS_EthSwt_00125 –> SRS_ETH_00087, SRS_ETH_00122
SWS_EthSwt_00187 –> SRS_ETH_00087
SWS_EthSwt_00058 –> SRS_BSW_00171
SWS_EthSwt_91012, SWS_EthSwt_91013 –> SRS_ETH_00126
SWS_EthSwt_91014, SWS_EthSwt_91015, SWS_EthSwt_91016, SWS_EthSwt_91018, SWS_EthSwt_91019, SWS_EthSwt_91021, SWS_EthSwt_91022, SWS_EthSwt_91022, SWS_EthSwt_91023, SWS_EthSwt_91024, SWS_EthSwt_91025, SWS_EthSwt_91026, SWS_EthSwt_91027 –> SRS_Eth_00123
SWS_EthSwt_00117, SWS_EthSwt_00203 –> SRS_ETH_00119, SRS_ETH_00087
SWS_EthSwt_00193 –> SRS_ETH_00122, SRS_ETH_00087
SWS_EthSwt_00114 –> SRS_BSW_00433
SWS_EthSwt_00098 –> SRS_Eth_00122, SRS_ETH_00118, SRS_ETH_00119, SRS_ETH_00120, SRS_ETH_00087, SRS_ETH_00125, SRS_BSW_00375
–Last change on issue 76248 comment 13–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.58 Specification Item SWS_EthSwt_00045

Trace References:

SRS_ETH_00118

Content:

The function EthSwt_GetBaudRate() shall read the current baud rate of the indexed switch portby calling the corresponding . If the indexed Ethernet port reference an Ethernet transceiver, the baud rate shall be obtained by the function EthTrcv_GetBaudRate() of the Ethernet Transceiver Driver. If the indexed Ethernet Switch port does not reference an

Ethernet transceiver, the baud rate shall be obtained from the MAC interface of the Switch port.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77664: [EthSwt] Behaviour of certain APIs for ports without EthSwtPortTrcvRef (part II)

  Problem description:

  The following APIs have no description regarding the handling of ports without EthSwtPortTrcvRef:
  EthSwt_GetLinkState
  EthSwt_GetBaudRate
  EthSwt_GetDuplexMode

  The description should be adjusted and harmonized with related requirements.

  I would also add the following function:
  EthSwt_StartSwitchPortAutoNegotiation
  I rate this function not to be used for ports without a EthSwtPortTrcvRef. For the sake of completeness I would also reformulate some requirements.
  –Last change on issue 77664 comment 10–

  Agreed solution:

  === EthSwt ===
  — EthSwt_GetLinkState —
  ~[SWS_EthSwt_00038]
  The function EthSwt_GetLinkState shall read the current (link) state of the indexed switch port. If the indexed Ethernet port references an Ethernet transceiver, the link state shall be obtained by calling the function EthTrcv_GetLinkState() of the Ethernet Transceiver Driver. If the indexed Ethernet Switch port does not reference an Ethernet transceiver, the state shall be obtained from the MAC interface of the Switch port.

  — EthSwt_GetBaudRate —
  ~[SWS_EthSwt_00045]
  The function EthSwt_GetBaudRate() shall read the current baud rate of the indexed switch port. If the indexed Ethernet port reference an Ethernet transceiver, the baud rate shall be obtained by the function EthTrcv_GetBaudRate() of the Ethernet Transceiver Driver. If the indexed Ethernet Switch port does not reference an Ethernet transceiver, the baud rate shall be obtained from the MAC interface of the Switch port.

-[SWS_EthSwt_00153] as # 77349 introduces a general req to this.

— EthSwt_GetDuplexMode —
~[SWS_EthSwt_00052]
The function EthSwt_GetDuplexMode() shall read the current duplex mode of the indexed switch port. If the indexed Ethernet port reference an Ethernet transceiver, the duplex mode shall be obtained by calling the function EthTrcv_ GetDuplexMode() of the Ethernet Transceiver Driver. If the indexed Ethernet Switch port does not reference an Ethernet transceiver, the duplex mode shall be obtained from the MAC interface of the Switch port.
-[SWS_EthSwt_00155] as # 77349 introduces a general req to this.

— EthSwt_StartSwitchPortAutoNegotiation —
~[SWS_EthSwt_00032]
The function EthSwt_StartSwitchPortAutoNegotiation shall restart the automatic negotiation of the used transmission parameters of the referenced Ethernet transceiver driver by calling the function EthTrcv_StartAutoNegotiation().
-[SWS_EthSwt_00158] as # 77349 introduces a general req to this.
–Last change on issue 77664 comment 26–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.59 Specification Item SWS_EthSwt_00046

Trace References:

SRS_BSW_00118

Content:

If development error detection is enabled: the function EthSwt_GetBaudRate shall check that the service EthSwt_SwitchInit was previously called. If the check fails, the function shall raise the development error ETHSWT_E_NOT_INITIALIZED and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

[SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

In the above requirement -
1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.

[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.60 Specification Item SWS_EthSwt_00047

Trace References:

SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchIdx is not valid, EthSwt_GetBaudRate shall raise the development error ETHSWT_E_INV_SWITCH_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -

1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevEr-

rorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.61 Specification Item SWS_EthSwt_00049

Trace References:

SRS_BSW_00171

Content:

The function EthSwt_GetBaudRate shall be pre compile time configurable On/Off by the configuration parameter: EthTrcvSwtGetBaudRateApi.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77043: [EthSwt] EthSwt APIs shouldnt be precompile ON/OFF based on EthTrcv parameters

  Problem description:

  For the APIs EthSwt_SetSwitchPortMode, EthSwt_GetSwitchPortMode, EthSwt_StartSwitchPortAutoNegotiation, EthSwt_GetLinkState, EthSwt_GetBaudRate and EthSwt_GetDuplexMode there is a requirement that states that the API should be configurable On/Off based on a EthTrcv parameter. For example for EthSwt_StartSwitchPortAutoNegotiation the requirement(SWS_EthSwt_00035) is: The function EthSwt_StartSwitchPortAutoNegotiation shall be pre-compile time configurable On/Off by the configuration parameter: EthTrcvStartAutoNegotiationApi.

  However, for other similar APIs a configuration parameter was defined in the switch driver. The functions are: EthSwt_GetPortSignalQuality, EthSwt_GetPortIdentifier, EthSwt_SetPortLoopbackMode, EthSwt_SetPortTxMode, EthSwt_GetPortCableDiagnosticsResult.

Also, if multiple EthTrcv drivers are used, how will the precompile parameters be defined in the switch?

Agreed solution:

add parameters
EthSwtSetSwitchPortModeApi, EthSwtGetSwitchPortModeApi, EthSwtStartSwitchPortAutoNegotiationApi, EthSwtGetLinkStateApi, EthSwtGetBaudRateApi and EthSwtGetDuplexModeApi
Description: Enables / disables the EthSwt_SetSwitchPortMode / EthSwt_GetSwitchPortMode / EthSwt_StartSwitchPortAutoNegotiation / EthSwt_GetLinkState/ EthSwt_GetBaudRate / EthSwt_GetDuplexMode
Multiplicity: 1
Type EcucBooleanParamDef
Default: False
Variant false
Config Class Pre-compile for all variants

~[SWS_EthSwt_00022]
replace EthTrcvSetTransceiverModeApi with EthSwtSetSwitchPortModeApi.
~SWS_EthSwt_00029
replace EthTrcvGetTransceiverModeApi with EthSwtGetSwitchPortModeApi.
~SWS_EthSwt_00035
replace EthTrcvStartAutoNegotiationApi with EthSwtStartSwitchPortAutoNegotiationApi
~[SWS_EthSwt_00042]
replace EthTrcvGetLinkStateApi with EthSwtGetLinkStateApi
~SWS_EthSwt_00049
replace EthTrcvGetBaudRateApi with EthSwtGetBaudRateApi
~SWS_EthSwt_00056
replace EthTrcvGetDuplexModeApi with EthSwtGetDuplexModeApi
–Last change on issue 77043 comment 2–

BW-C-Level:

Application	Specification	Bus
1	3	1

1.62 Specification Item SWS_EthSwt_00051

Trace References:

SRS_ETH_00086 00118

Content:

Service name:	EthSwt_GetDuplexModeEthSwt_GetDuplexMode	
Syntax:	Std_ReturnType EthSwt_GetDuplexMode(uint8 SwitchIdx, uint8 SwitchPortIdx, EthTrcv_DuplexModeType* DuplexModePtr)	
Service ID[hex]:	0x08	
Sync/Async:	Synchronous /Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	SwitchIdxEthSwt_GetDuplex Mode.SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	SwitchPortIdxEthSwt_GetDuplex Mode.SwitchPortIdx	Index of the port at the addressed switch
Parameters (inout):	None	
Parameters (out):	DuplexModePtrEthSwt_GetDuplex Mode.DuplexModePtr	ETHTRCV_DUPLEX_MODE_HALF: half duplex connections ETHTRCV_DUPLEXMODE_FULL: full duplex connection
Return value:	Std_ReturnType	E_OK: success E_NOT_OK: duplex mode of the indexed switch port could not be obtained
Description:	Obtains the duplex mode of the indexed switch port	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76248: [EthSwt]Remove references to non existing requirements (SRS, SWS)

  Problem description:

  Some review findings brought up references to non existing requirements. Please check and remove the references and if needed reformulate the requirements.

  Agreed solution:

  SWS_EthSwt_00123 –> SRS_BSW_00406
  SWS_EthSwt_00165 -> SRS_BSW_00395
  SWS_EthSwt_00227 -> SRS_ETH_00087
  SWS_EthSwt_91017, SWS_EthSwt_91020, –> SRS_ETH_00123
  SWS_EthSwt_00006 –> SRS_BSW_00101
  SWS_EthSwt_00010 left blank as SRS doe not yet provide a requirement here but maybe with RFC # 75170 AUTOSAR will introduce an SRS-item after all.

SWS_EthSwt_00376, SWS_EthSwt_00374, SWS_EthSwt_00375 –> SRS-Missing! same as for SWS_EThSwt_00010
SWS_EthSwt_00018, SRS_EthSwt_00025 –> SRS_ETH_00118
SWS_EthSwt_00031 –> SRS_ETH_00087
SWS_EthSwt_00037 –> SRS_ETH_00119
SWS_EthSwt_00044, SWS_EthSwt_00051 –> SRS_ETH_00118
SWS_EthSwt_00060 –> SRS_ETH_00087
SWS_EthSwt_00111 –> erase SRS_ETH_00086
SWS_EthSwt00079 –>SRS_ETH_00119
SWS_EthSwt_00206, SWS_EthSwt_00211, SWS_EthSwt_00216, SWS_EthSwt_00221 –>SRS_Eth_00120
SWS_EthSwt_00172 –>SRS_ETH_00121, SRS_ETH_00114
SWS_EthSwt_00086, SWS_EthSwt_00091, SWS_EthSwt_00182, SWS_EthSwt_00125 –> SRS_ETH_00087, SRS_ETH_00122
SWS_EthSwt_00187 –> SRS_ETH_00087
SWS_EthSwt_00058 –> SRS_BSW_00171
SWS_EthSwt_91012, SWS_EthSwt_91013 –> SRS_ETH_00126
SWS_EthSwt_91014, SWS_EthSwt_91015, SWS_EthSwt_91016, SWS_EthSwt_91018, SWS_EthSwt_91019, SWS_EthSwt_91021, SWS_EthSwt_91022, SWS_EthSwt_91022, SWS_EthSwt_91023, SWS_EthSwt_91024, SWS_EthSwt_91025, SWS_EthSwt_91026, SWS_EthSwt_91027 –> SRS_Eth_00123
SWS_EthSwt_00117, SWS_EthSwt_00203 –> SRS_ETH_00119, SRS_ETH_00087
SWS_EthSwt_00193 –> SRS_ETH_00122, SRS_ETH_00087
SWS_EthSwt_00114 –> SRS_BSW_00433
SWS_EthSwt_00098 –> SRS_Eth_00122, SRS_ETH_00118, SRS_ETH_00119, SRS_ETH_00120, SRS_ETH_00087, SRS_ETH_00125, SRS_BSW_00375
–Last change on issue 76248 comment 13–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.63 Specification Item SWS_EthSwt_00052

Trace References:

SRS_ETH_00118

Content:

The function EthSwt_GetDuplexMode() shall read the current duplex mode of the indexed switch port. If the indexed Ethernet port reference an Ethernet transceiver, the duplex mode shall be obtained by calling the function EthTrcv_ GetDuplexMode() of the Ethernet Transceiver Driver. If the indexed Ethernet Switch port does not reference an Ethernet transceiver, the duplex mode shall be obtained from the MAC interface of the Switch port.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77664: [EthSwt] Behaviour of certain APIs for ports without EthSwtPortTrcvRef (part II)

  Problem description:

  The following APIs have no description regarding the handling of ports without EthSwtPortTrcvRef:
  EthSwt_GetLinkState
  EthSwt_GetBaudRate
  EthSwt_GetDuplexMode

  The description should be adjusted and harmonized with related requirements.

  I would also add the following function:
  EthSwt_StartSwitchPortAutoNegotiation
  I rate this function not to be used for ports without a EthSwtPortTrcvRef. For the sake of completeness I would also reformulate some requirements.
  –Last change on issue 77664 comment 10–

  Agreed solution:

  === EthSwt ===
  — EthSwt_GetLinkState —
  ~[SWS_EthSwt_00038]
  The function EthSwt_GetLinkState shall read the current (link) state of the indexed switch port. If the indexed Ethernet port references an Ethernet transceiver, the link state shall be obtained by calling the function EthTrcv_GetLinkState() of the Ethernet Transceiver Driver. If the indexed Ethernet Switch port does not reference an Ethernet transceiver, the state shall be obtained from the MAC interface of the Switch port.

  — EthSwt_GetBaudRate —
  ~[SWS_EthSwt_00045]
  The function EthSwt_GetBaudRate() shall read the current baud rate of the indexed switch port. If the indexed Ethernet port reference an Ethernet transceiver, the baud rate shall be obtained by the function EthTrcv_GetBaudRate() of the Ethernet

Transceiver Driver. If the indexed Ethernet Switch port does not reference an Ethernet transceiver, the baud rate shall be obtained from the MAC interface of the Switch port.
-[SWS_EthSwt_00153] as # 77349 introduces a general req to this.

— EthSwt_GetDuplexMode —
~[SWS_EthSwt_00052]
The function EthSwt_GetDuplexMode() shall read the current duplex mode of the indexed switch port. If the indexed Ethernet port reference an Ethernet transceiver, the duplex mode shall be obtained by calling the function EthTrcv_ GetDuplexMode() of the Ethernet Transceiver Driver. If the indexed Ethernet Switch port does not reference an Ethernet transceiver, the duplex mode shall be obtained from the MAC interface of the Switch port.
-[SWS_EthSwt_00155] as # 77349 introduces a general req to this.

— EthSwt_StartSwitchPortAutoNegotiation —
~[SWS_EthSwt_00032]
The function EthSwt_StartSwitchPortAutoNegotiation shall restart the automatic negotiation of the used transmission parameters of the referenced Ethernet transceiver driver by calling the function EthTrcv_StartAutoNegotiation().
-[SWS_EthSwt_00158] as # 77349 introduces a general req to this.
–Last change on issue 77664 comment 26–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.64 Specification Item SWS_EthSwt_00053

Trace References:

SRS_BSW_00406

Content:

If development error detection is enabled: the function EthSwt_GetDuplexMode shall check that the service EthSwt_SwitchInit was previously called. If the check fails, the function shall raise the development error ETHSWT_E_NOT_INITIALIZED and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

Problem description:

[SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

In the above requirement -
1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with

input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.65 Specification Item SWS_EthSwt_00054

Trace References:

SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchIdx is not valid,EthSwt_GetDuplexMode shall raise the development error ETHSWT_E_INV_SWITCH_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

In the above requirement -
1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETH-

SWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.66 Specification Item SWS_EthSwt_00056

Trace References:

SRS_BSW_00171

Content:

The function EthSwt_GetDuplexMode shall be pre compile time configurable On/Off by the configuration parameter: EthTrcvSwtGetDuplexModeApi.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77043: [EthSwt] EthSwt APIs shouldnt be precompile ON/OFF based on EthTrcv parameters

  Problem description:

  For the APIs EthSwt_SetSwitchPortMode, EthSwt_GetSwitchPortMode, EthSwt_StartSwitchPortAutoNegotiation, EthSwt_GetLinkState, EthSwt_GetBaudRate and EthSwt_GetDuplexMode there is a requirement that states that the API should be configurable On/Off based on a EthTrcv parameter. For example for EthSwt_StartSwitchPortAutoNegotiation the requirement(SWS_EthSwt_00035) is: The function EthSwt_StartSwitchPortAutoNegotiation shall be pre-compile time configurable On/Off by the configuration parameter: EthTrcvStartAutoNegotiationApi.

However, for other similar APIs a configuration parameter was defined in the switch driver. The functions are: EthSwt_GetPortSignalQuality, EthSwt_GetPortIdentifier, EthSwt_SetPortLoopbackMode, EthSwt_SetPortTxMode, EthSwt_GetPortCableDiagnosticsResult.

Also, if multiple EthTrcv drivers are used, how will the precompile parameters be defined in the switch?

Agreed solution:

add parameters
EthSwtSetSwitchPortModeApi, EthSwtGetSwitchPortModeApi, EthSwtStartSwitchPortAutoNegotiationApi, EthSwtGetLinkStateApi, EthSwtGetBaudRateApi and EthSwtGetDuplexModeApi
Description: Enables / disables the EthSwt_SetSwitchPortMode / EthSwt_GetSwitchPortMode / EthSwt_StartSwitchPortAutoNegotiation / EthSwt_GetLinkState/ EthSwt_GetBaudRate / EthSwt_GetDuplexMode
Multiplicity: 1
Type EcucBooleanParamDef
Default: False
Variant false
Config Class Pre-compile for all variants

~[SWS_EthSwt_00022]
replace EthTrcvSetTransceiverModeApi with EthSwtSetSwitchPortModeApi.
~SWS_EthSwt_00029
replace EthTrcvGetTransceiverModeApi with EthSwtGetSwitchPortModeApi.
~SWS_EthSwt_00035
replace EthTrcvStartAutoNegotiationApi with EthSwtStartSwitchPortAutoNegotiationApi
~[SWS_EthSwt_00042]
replace EthTrcvGetLinkStateApi with EthSwtGetLinkStateApi
~SWS_EthSwt_00049
replace EthTrcvGetBaudRateApi with EthSwtGetBaudRateApi
~SWS_EthSwt_00056
replace EthTrcvGetDuplexModeApi with EthSwtGetDuplexModeApi
–Last change on issue 77043 comment 2–

BW-C-Level:

Application	**Specification**	**Bus**
1	3	1

1.67 Specification Item SWS_EthSwt_00058

Trace References:

SRS_ETH_00086 BSW_00171

Content:

Service name:	EthSwt_GetVersionInfoEthSwt_GetVersionInfo	
Syntax:	void EthSwt_GetVersionInfo(Std_VersionInfoType* VersionInfoPtr)	
Service ID[hex]:	0x18	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	None	
Parameters (inout):	None	
Parameters (out):	VersionInfoPtrEthSwt_GetVersionInfo.VersionInfoPtr	Pointer to where to store the version information of this module.
Return value:	None	
Description:	Returns the version information of this module.	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76248: [EthSwt]Remove references to non existing requirements (SRS, SWS)

  Problem description:

  Some review findings brought up references to non existing requirements. Please check and remove the references and if needed reformulate the requirements.

  Agreed solution:

  SWS_EthSwt_00123 –> SRS_BSW_00406
  SWS_EthSwt_00165 -> SRS_BSW_00395
  SWS_EthSwt_00227 -> SRS_ETH_00087
  SWS_EthSwt_91017, SWS_EthSwt_91020, –> SRS_ETH_00123
  SWS_EthSwt_00006 –> SRS_BSW_00101
  SWS_EthSwt_00010 left blank as SRS doe not yet provide a requirement here but maybe with RFC # 75170 AUTOSAR will introduce an SRS-item after all.
  SWS_EthSwt_00376, SWS_EthSwt_00374, SWS_EthSwt_00375 –> SRS-Missing! same as for SWS_EThSwt_00010
  SWS_EthSwt_00018, SRS_EthSwt_00025 –> SRS_ETH_00118
  SWS_EthSwt_00031 –> SRS_ETH_00087
  SWS_EthSwt_00037 –> SRS_ETH_00119
  SWS_EthSwt_00044, SWS_EthSwt_00051 –> SRS_ETH_00118
  SWS_EthSwt_00060 –> SRS_ETH_00087

SWS_EthSwt_00111 –> erase SRS_ETH_00086
SWS_EthSwt00079 –>SRS_ETH_00119
SWS_EthSwt_00206, SWS_EthSwt_00211, SWS_EthSwt_00216, SWS_EthSwt_00221 –>SRS_Eth_00120
SWS_EthSwt_00172 –>SRS_ETH_00121, SRS_ETH_00114
SWS_EthSwt_00086, SWS_EthSwt_00091, SWS_EthSwt_00182, SWS_EthSwt_00125 –> SRS_ETH_00087, SRS_ETH_00122
SWS_EthSwt_00187 –> SRS_ETH_00087
SWS_EthSwt_00058 –> SRS_BSW_00171
SWS_EthSwt_91012, SWS_EthSwt_91013 –> SRS_ETH_00126
SWS_EthSwt_91014, SWS_EthSwt_91015, SWS_EthSwt_91016, SWS_EthSwt_91018, SWS_EthSwt_91019, SWS_EthSwt_91021, SWS_EthSwt_91022, SWS_EthSwt_91022, SWS_EthSwt_91023, SWS_EthSwt_91024, SWS_EthSwt_91025, SWS_EthSwt_91026, SWS_EthSwt_91027 –> SRS_Eth_00123
SWS_EthSwt_00117, SWS_EthSwt_00203 –> SRS_ETH_00119, SRS_ETH_00087
SWS_EthSwt_00193 –> SRS_ETH_00122, SRS_ETH_00087
SWS_EthSwt_00114 –> SRS_BSW_00433
SWS_EthSwt_00098 –> SRS_Eth_00122, SRS_ETH_00118, SRS_ETH_00119, SRS_ETH_00120, SRS_ETH_00087, SRS_ETH_00125, SRS_BSW_00375
–Last change on issue 76248 comment 13–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.68 Specification Item SWS_EthSwt_00060

Trace References:

SRS_ETH_00086, SRS_ETH_00087

Content:

Service name:	EthSwt_GetPortMacAddrEthSwt_GetPortMacAddr
Syntax:	Std_ReturnType EthSwt_GetPortMacAddr(const uint8* MacAddrPtr, const uint8* SwitchIdxPtr, uint8* PortIdxPtr)
Service ID[hex]:	0x09
Sync/Async:	Synchronous /Asynchronous
Reentrancy:	Non Reentrant

Parameters (in):	MacAddrPtrEthSwt_GetPortMacAddr.MacAddrPtr	MAC-address for which a switch port is searched over which the node with this MAC-address can be reached.
	SwitchIdxPtrEthSwt_GetPortMacAddr.SwitchIdxPtr	Pointer to the switch index
Parameters (inout):	None	
Parameters (out):	PortIdxPtrEthSwt_GetPortMacAddr.PortIdxPtr	Pointer to the port index
Return value:	Std_ReturnType	E_OK: success E_NOT_OK: multiple ports were found
Description:	Obtains the port over which this MAC-address at the indexed switch can be reached. The result might be used for a DHCP-server which will need the port/MAC-resolution. If for the PortIdxPtr the maximal possible value (255) is returned the given MAC address cannot be reached via a port of this switch. If multiple ports were found the API returns E_NOT_OK.	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76248: [EthSwt]Remove references to non existing requirements (SRS, SWS)

  Problem description:

  Some review findings brought up references to non existing requirements. Please check and remove the references and if needed reformulate the requirements.

  Agreed solution:

  SWS_EthSwt_00123 –> SRS_BSW_00406
  SWS_EthSwt_00165 -> SRS_BSW_00395
  SWS_EthSwt_00227 -> SRS_ETH_00087
  SWS_EthSwt_91017, SWS_EthSwt_91020, –> SRS_ETH_00123
  SWS_EthSwt_00006 –> SRS_BSW_00101
  SWS_EthSwt_00010 left blank as SRS doe not yet provide a requirement here but maybe with RFC # 75170 AUTOSAR will introduce an SRS-item after all.
  SWS_EthSwt_00376, SWS_EthSwt_00374, SWS_EthSwt_00375 –> SRS-Missing! same as for SWS_EThSwt_00010
  SWS_EthSwt_00018, SRS_EthSwt_00025 –> SRS_ETH_00118
  SWS_EthSwt_00031 –> SRS_ETH_00087
  SWS_EthSwt_00037 –> SRS_ETH_00119
  SWS_EthSwt_00044, SWS_EthSwt_00051 –> SRS_ETH_00118
  SWS_EthSwt_00060 –> SRS_ETH_00087
  SWS_EthSwt_00111 –> erase SRS_ETH_00086
  SWS_EthSwt00079 –>SRS_ETH_00119
  SWS_EthSwt_00206, SWS_EthSwt_00211, SWS_EthSwt_00216, SWS_EthSwt_00221 –>SRS_Eth_00120
  SWS_EthSwt_00172 –>SRS_ETH_00121, SRS_ETH_00114
  SWS_EthSwt_00086, SWS_EthSwt_00091, SWS_EthSwt_00182,

SWS_EthSwt_00125 –> SRS_ETH_00087, SRS_ETH_00122
SWS_EthSwt_00187 –> SRS_ETH_00087
SWS_EthSwt_00058 –> SRS_BSW_00171
SWS_EthSwt_91012, SWS_EthSwt_91013 –> SRS_ETH_00126
SWS_EthSwt_91014, SWS_EthSwt_91015, SWS_EthSwt_91016, SWS_EthSwt_91018, SWS_EthSwt_91019, SWS_EthSwt_91021, SWS_EthSwt_91022, SWS_EthSwt_91022, SWS_EthSwt_91023, SWS_EthSwt_91024, SWS_EthSwt_91025, SWS_EthSwt_91026, SWS_EthSwt_91027 –> SRS_Eth_00123
SWS_EthSwt_00117, SWS_EthSwt_00203 –> SRS_ETH_00119, SRS_ETH_00087
SWS_EthSwt_00193 –> SRS_ETH_00122, SRS_ETH_00087
SWS_EthSwt_00114 –> SRS_BSW_00433
SWS_EthSwt_00098 –> SRS_Eth_00122, SRS_ETH_00118, SRS_ETH_00119, SRS_ETH_00120, SRS_ETH_00087, SRS_ETH_00125, SRS_BSW_00375
–Last change on issue 76248 comment 13–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.69 Specification Item SWS_EthSwt_00062

Trace References:

SRS_BSW_00406

Content:

If development error detection is enabled: the function EthSwt_GetPortMacAddr shall check that the service EthSwt_SwitchInit was previously called. If the check fails, the function shall raise the development error ETHSWT_E_NOT_INITIALIZED and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

In the above requirement -
1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.

[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.70 Specification Item SWS_EthSwt_00064

Trace References:

SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter MacAddrPtr is a NULL pointer, EthSwt_GetPortMacAddr shall raise the development error ETHSWT_E_INV_POINTER and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not de-

fault errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.71 Specification Item SWS_EthSwt_00079

Trace References:

SRS_ETH_00086 00119

Content:

Service name:	EthSwt_GetBufferLevelEthSwt_GetBufferLevel	
Syntax:	Std_ReturnType EthSwt_GetBufferLevel(uint8 SwitchIdx, uint32* SwitchBufferLevelPtr)	
Service ID[hex]:	0x0b	
Sync/Async:	Synchronous /Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	SwitchIdxEthSwt_GetBufferLevel.SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
Parameters (inout):	None	
Parameters (out):	SwitchBufferLevelPtrEthSwt_GetBufferLevel.SwitchBufferLevelPtr	The interpretation of this value is switch dependent
Return value:	Std_ReturnType	E_OK: success E_NOT_OK: buffer level could not be obtained
Description:	Reads the buffer level of the corresponding switch. Whether this buffer level is one value for the entire switch (shared memory) or one value for each port at a switch is technology dependent. This API will be called, e.g. by a CDD	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76248: [EthSwt]Remove references to non existing requirements (SRS, SWS)

  Problem description:

Some review findings brought up references to non existing requirements. Please check and remove the references and if needed reformulate the requirements.

Agreed solution:

SWS_EthSwt_00123 –> SRS_BSW_00406
SWS_EthSwt_00165 -> SRS_BSW_00395
SWS_EthSwt_00227 -> SRS_ETH_00087
SWS_EthSwt_91017, SWS_EthSwt_91020, –> SRS_ETH_00123
SWS_EthSwt_00006 –> SRS_BSW_00101
SWS_EthSwt_00010 left blank as SRS doe not yet provide a requirement here but maybe with RFC # 75170 AUTOSAR will introduce an SRS-item after all.
SWS_EthSwt_00376, SWS_EthSwt_00374, SWS_EthSwt_00375 –> SRS-Missing! same as for SWS_EThSwt_00010
SWS_EthSwt_00018, SRS_EthSwt_00025 –> SRS_ETH_00118
SWS_EthSwt_00031 –> SRS_ETH_00087
SWS_EthSwt_00037 –> SRS_ETH_00119
SWS_EthSwt_00044, SWS_EthSwt_00051 –> SRS_ETH_00118
SWS_EthSwt_00060 –> SRS_ETH_00087
SWS_EthSwt_00111 –> erase SRS_ETH_00086
SWS_EthSwt00079 –>SRS_ETH_00119
SWS_EthSwt_00206, SWS_EthSwt_00211, SWS_EthSwt_00216, SWS_EthSwt_00221 –>SRS_Eth_00120
SWS_EthSwt_00172 –>SRS_ETH_00121, SRS_ETH_00114
SWS_EthSwt_00086, SWS_EthSwt_00091, SWS_EthSwt_00182, SWS_EthSwt_00125 –> SRS_ETH_00087, SRS_ETH_00122
SWS_EthSwt_00187 –> SRS_ETH_00087
SWS_EthSwt_00058 –> SRS_BSW_00171
SWS_EthSwt_91012, SWS_EthSwt_91013 –> SRS_ETH_00126
SWS_EthSwt_91014, SWS_EthSwt_91015, SWS_EthSwt_91016, SWS_EthSwt_91018, SWS_EthSwt_91019, SWS_EthSwt_91021, SWS_EthSwt_91022, SWS_EthSwt_91022, SWS_EthSwt_91023, SWS_EthSwt_91024, SWS_EthSwt_91025, SWS_EthSwt_91026, SWS_EthSwt_91027 –> SRS_Eth_00123
SWS_EthSwt_00117, SWS_EthSwt_00203 –> SRS_ETH_00119, SRS_ETH_00087
SWS_EthSwt_00193 –> SRS_ETH_00122, SRS_ETH_00087
SWS_EthSwt_00114 –> SRS_BSW_00433
SWS_EthSwt_00098 –> SRS_Eth_00122, SRS_ETH_00118, SRS_ETH_00119, SRS_ETH_00120, SRS_ETH_00087, SRS_ETH_00125, SRS_BSW_00375
–Last change on issue 76248 comment 13–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.72 Specification Item SWS_EthSwt_00081

Trace References:

SRS_BSW_00406

Content:

If development error detection is enabled: the function EthSwt_GetBufferLevel shall check that the service EthSwt_SwitchInit was previously called. If the check fails, the function shall raise the development error ETHSWT_E_NOT_INITIALIZED and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER

add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.73 Specification Item SWS_EthSwt_00082

Trace References:

SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchIdx is not valid,EthSwt_GetBufferLevel shall raise the development error ETHSWT_E_INV_SWITCH_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

  For Error Detection following general rules apply:

[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.74 Specification Item SWS_EthSwt_00086

Trace References:

SRS_ETH_0008600087, SRS_ETH_00087 00122

Content:

Service name:	EthSwt_StoreConfigurationEthSwt_StoreConfiguration	
Syntax:	Std_ReturnType EthSwt_StoreConfiguration(uint8 SwitchIdx)	
Service ID[hex]:	0x13	
Sync/Async:	Synchronous /Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	SwitchIdxEthSwt_Store Configuration.SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: success E_NOT_OK: Configuration could not be persistently stored
Description:	Stores the configuration of the learned MAC/Port tables of a switch in a persistent manner and will be used by e.g. CDD.	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76248: [EthSwt]Remove references to non existing requirements (SRS, SWS)

  Problem description:

  Some review findings brought up references to non existing requirements. Please check and remove the references and if needed reformulate the requirements.

  Agreed solution:

  SWS_EthSwt_00123 –> SRS_BSW_00406
  SWS_EthSwt_00165 -> SRS_BSW_00395
  SWS_EthSwt_00227 -> SRS_ETH_00087
  SWS_EthSwt_91017, SWS_EthSwt_91020, –> SRS_ETH_00123
  SWS_EthSwt_00006 –> SRS_BSW_00101
  SWS_EthSwt_00010 left blank as SRS doe not yet provide a requirement here but maybe with RFC # 75170 AUTOSAR will introduce an SRS-item after all.
  SWS_EthSwt_00376, SWS_EthSwt_00374, SWS_EthSwt_00375 –> SRS-Missing! same as for SWS_EThSwt_00010
  SWS_EthSwt_00018, SRS_EthSwt_00025 –> SRS_ETH_00118
  SWS_EthSwt_00031 –> SRS_ETH_00087
  SWS_EthSwt_00037 –> SRS_ETH_00119
  SWS_EthSwt_00044, SWS_EthSwt_00051 –> SRS_ETH_00118
  SWS_EthSwt_00060 –> SRS_ETH_00087
  SWS_EthSwt_00111 –> erase SRS_ETH_00086
  SWS_EthSwt00079 –>SRS_ETH_00119
  SWS_EthSwt_00206, SWS_EthSwt_00211, SWS_EthSwt_00216,

SWS_EthSwt_00221 –>SRS_Eth_00120
SWS_EthSwt_00172 –>SRS_ETH_00121, SRS_ETH_00114
SWS_EthSwt_00086, SWS_EthSwt_00091, SWS_EthSwt_00182, SWS_EthSwt_00125 –> SRS_ETH_00087, SRS_ETH_00122
SWS_EthSwt_00187 –> SRS_ETH_00087
SWS_EthSwt_00058 –> SRS_BSW_00171
SWS_EthSwt_91012, SWS_EthSwt_91013 –> SRS_ETH_00126
SWS_EthSwt_91014, SWS_EthSwt_91015, SWS_EthSwt_91016, SWS_EthSwt_91018, SWS_EthSwt_91019, SWS_EthSwt_91021, SWS_EthSwt_91022, SWS_EthSwt_91022, SWS_EthSwt_91023, SWS_EthSwt_91024, SWS_EthSwt_91025, SWS_EthSwt_91026, SWS_EthSwt_91027 –> SRS_Eth_00123
SWS_EthSwt_00117, SWS_EthSwt_00203 –> SRS_ETH_00119, SRS_ETH_00087
SWS_EthSwt_00193 –> SRS_ETH_00122, SRS_ETH_00087
SWS_EthSwt_00114 –> SRS_BSW_00433
SWS_EthSwt_00098 –> SRS_Eth_00122, SRS_ETH_00118, SRS_ETH_00119, SRS_ETH_00120, SRS_ETH_00087, SRS_ETH_00125, SRS_BSW_00375
–Last change on issue 76248 comment 13–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.75 Specification Item SWS_EthSwt_00088

Trace References:

SRS_BSW_00406

Content:

If development error detection is enabled: the function EthSwt_StoreConfiguration shall check that the service EthSwt_SwitchInit was previously called. If the check fails, the function shall raise the development error ETHSWT_E_NOT_INITIALIZED and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

[SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

In the above requirement -
1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.

[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.76 Specification Item SWS_EthSwt_00089

Trace References:

SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchIdx is not valid,EthSwt_StoreConfiguration shall raise the development error ETHSWT_E_INV_SWITCH_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -

1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevEr-

rorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	**Specification**	**Bus**
1	4	1

1.77 Specification Item SWS_EthSwt_00091

Trace References:

SRS_ETH_0008600087, SRS_ETH_00087 00122

Content:

Service name:	EthSwt_ResetConfigurationEthSwt_ResetConfiguration	
Syntax:	Std_ReturnType EthSwt_ResetConfiguration(uint8 SwitchIdx)	
Service ID[hex]:	0x14	
Sync/Async:	Synchronous /Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	SwitchIdxEthSwt_Reset Configuration.SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: success E_NOT_OK: configuration could be persistently reset
Description:	Resets the configuration of the learned MAC/Port tables of a switch in a persistent manner and will be used by e.g. CDD. The statically configured entries shall still remain.	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76248: [EthSwt]Remove references to non existing requirements (SRS, SWS)

Problem description:

Some review findings brought up references to non existing requirements. Please check and remove the references and if needed reformulate the requirements.

Agreed solution:

SWS_EthSwt_00123 –> SRS_BSW_00406
SWS_EthSwt_00165 -> SRS_BSW_00395
SWS_EthSwt_00227 -> SRS_ETH_00087
SWS_EthSwt_91017, SWS_EthSwt_91020, –> SRS_ETH_00123
SWS_EthSwt_00006 –> SRS_BSW_00101
SWS_EthSwt_00010 left blank as SRS doe not yet provide a requirement here but maybe with RFC # 75170 AUTOSAR will introduce an SRS-item after all.
SWS_EthSwt_00376, SWS_EthSwt_00374, SWS_EthSwt_00375 –> SRS-Missing! same as for SWS_EThSwt_00010
SWS_EthSwt_00018, SRS_EthSwt_00025 –> SRS_ETH_00118
SWS_EthSwt_00031 –> SRS_ETH_00087
SWS_EthSwt_00037 –> SRS_ETH_00119
SWS_EthSwt_00044, SWS_EthSwt_00051 –> SRS_ETH_00118
SWS_EthSwt_00060 –> SRS_ETH_00087
SWS_EthSwt_00111 –> erase SRS_ETH_00086
SWS_EthSwt00079 –>SRS_ETH_00119
SWS_EthSwt_00206, SWS_EthSwt_00211, SWS_EthSwt_00216, SWS_EthSwt_00221 –>SRS_Eth_00120
SWS_EthSwt_00172 –>SRS_ETH_00121, SRS_ETH_00114
SWS_EthSwt_00086, SWS_EthSwt_00091, SWS_EthSwt_00182, SWS_EthSwt_00125 –> SRS_ETH_00087, SRS_ETH_00122
SWS_EthSwt_00187 –> SRS_ETH_00087
SWS_EthSwt_00058 –> SRS_BSW_00171
SWS_EthSwt_91012, SWS_EthSwt_91013 –> SRS_ETH_00126
SWS_EthSwt_91014, SWS_EthSwt_91015, SWS_EthSwt_91016, SWS_EthSwt_91018, SWS_EthSwt_91019, SWS_EthSwt_91021, SWS_EthSwt_91022, SWS_EthSwt_91022, SWS_EthSwt_91023, SWS_EthSwt_91024, SWS_EthSwt_91025, SWS_EthSwt_91026, SWS_EthSwt_91027 –> SRS_Eth_00123
SWS_EthSwt_00117, SWS_EthSwt_00203 –> SRS_ETH_00119, SRS_ETH_00087
SWS_EthSwt_00193 –> SRS_ETH_00122, SRS_ETH_00087
SWS_EthSwt_00114 –> SRS_BSW_00433
SWS_EthSwt_00098 –> SRS_Eth_00122, SRS_ETH_00118, SRS_ETH_00119, SRS_ETH_00120, SRS_ETH_00087, SRS_ETH_00125, SRS_BSW_00375
–Last change on issue 76248 comment 13–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.78 Specification Item SWS_EthSwt_00093

Trace References:

SRS_BSW_00406

Content:

If development error detection is enabled: the function EthSwt_ResetConfiguration shall check that the service EthSwt_SwitchInit was previously called. If the check fails, the function shall raise the development error ETHSWT_E_NOT_INITIALIZED and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER

add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.79 Specification Item SWS_EthSwt_00094

Trace References:

SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchIdx is not valid,EthSwt_ResetConfiguration shall raise the development error ETHSWT_E_INV_SWITCH_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

  For Error Detection following general rules apply:

[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.80 Specification Item SWS_EthSwt_00098

Trace References:

SRS_Eth_00122, SRS_ETH_00118, SRS_ETH_00119, SRS_ETH_00120, SRS_ETH_00087, SRS_ETH_00125, SRS_BSW_00375

Content:

API function	Description
Dem_SetEventStatus	Called by SW-Cs or BSW modules to report monitor status information to the Dem. BSW modules calling Dem_SetEvent Status can safely ignore the return value.
Det_ReportError	Service to report development errors.
Eth_ReadMii	Reads a transceiver register
Eth_WriteMii	Configures a transceiver register or triggers a function offered by the receiver
EthIf_SwitchEgressTimeStampIndication	Returns an egress timestamp value out of the Switch. If the HW resolution is lower than the Eth_TimeStampType resolution resp. range, than the remaining bits will be filled with 0.
EthIf_SwitchIngressTimeStampIndication	Returns an ingress timestamp value out of the Switch. If the HW resolution is lower than the Eth_TimeStampType resolution resp. range, than the remaining bits will be filled with 0.
EthIf_SwitchMgmtInfoIndication	Ingress Switch management info indication redirected call to upper layers who registered for the call.
EthTrcv_GetBaudRate	Obtains the baud rate of the indexed transceiver
EthTrcv_GetDuplexMode	Obtains the duplex mode of the indexed transceiver
EthTrcv_GetLinkState	Obtains the link state of the indexed transceiver
EthTrcv_GetTransceiverMode	Obtains the state of the indexed transceiver
EthTrcv_SetTransceiverMode	Enables / disables the indexed transceiver
EthTrcv_StartAutoNegotiation	Restarts the negotiation of the transmission parameters used by the indexed transceiver
NvM_GetErrorStatus	Service to read the block dependent error/status information.
NvM_ReadBlock	Service to copy the data of the NV block to its corresponding RAM block.
NvM_WriteBlock	Service to copy the data of the RAM block to its corresponding NV block.
Spi_AsyncTransmit	Service to transmit data on the SPI bus.
Spi_Cancel	Service cancels the specified on-going sequence transmission.
Spi_ReadIB	Service for reading synchronously one or more data from an IB SPI Handler/Driver Channel specified by parameter.
Spi_SetAsyncMode	Service to set the asynchronous mechanism mode for SPI busses handled asynchronously.
Spi_SetupEB	Service to setup the buffers and the length of data for the EB SPI Handler/Driver Channel specified.
Spi_SyncTransmit	Service to transmit data on the SPI bus
Spi_WriteIB	Service for writing one or more data to an IB SPI Handler/Driver Channel specified by parameter.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76404: [Det] Clarifications on runtime errors

Problem description:

There are several uncertainties/problems in the SWS DET:
1. According to SWS_Det_00180, the callouts should have the same signatures as the corresponding DET functions, but they are void(void) (SWS_Det_00181, SWS_Det_00184, SWS_Det_00187).
2. Section 8.2.3.1 does not describe how the instance ID is passed to DET.
3. Configuration of header files for all three error type callouts are missing.
4. Why does the development error callout reside in DetNotification, while the other two callouts reside in DetGeneral?
5. The limitation in section 4.1 regarding "supervisor mode" does not really make sense. It is assumed that the DET is ignorant regarding the call context, and the software receiving DET callbacks (like DLT or the implementers of the callouts) need to take care of resolving the calling context, if necessary (e.g. in multi-core environments).
6. SWS_Det_00302 defines several runtime errors. But apart from DET_E_CANNOT_REPORT, it is unclear in which situation these errors could be reported by DET: For errors reported by BSW, the DET has no means to validate anything that could lead to such an error. And for SWCs, the modeling already takes care that DET_E_WRONG_MODULE and DET_E_WRONG_INSTANCE cannot occur, while the other two errors can also not be checked by DET without further configuration.
7. Det_ReportTransientFault (SWS_Det_01003) shall return the return value of a configured callout. But what shall happen if more than one callout exisits, and the return different values?
8. SWS_Det_00052: The only API that can result in DET_E_PARAM_POINTER is Det_GetVersionInfo (as the error description mentions correctly). Please reformulate this requirement and move it to section 8.1.3.6 "Det_GetVersionInfo".
–Last change on issue 76404 comment 13–

Agreed solution:

1.
~change SWS_Det_00181/184/187 such that signatures match the APIs
~Figures 3,5, and 7 to be corrected (return missing)
5. remove from 4.1. the sentence: "It is assumed that the whole Basic Software runs in supervisor mode or the switch to supervisor mode is done by a system call within the error reporting function of the DET module."
6. remove SWS_Det_00302 and SWS_Det_00303 and all included errors
7. change SWS_Det_01003 (Return Value-Part only): "Std_ReturnType" If no callout exists it shall return E_OK, otherwise it shall return the value of the configured callout. In case several callouts are configured the logical or (sum) of the callout return values shall be returned. Rationale: since E_OK=0, E_OK will be only returned if all are E_OK, and for multiple error codes there is a good chance to detect several

of them.
8. change SWS_Det_00052 from "in case a null pointer error occurs." to "in case a null pointer error occurs in Det_GetVersionInfo." Do not move the requirement, since otherwise the section 7.7 would be empty, but add the following sentence to 8.1.3.6: "In case a null pointer is passed, DET_E_PARAM_POINTER is returned, see SWS_Det_00052."
–Last change on issue 76404 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

- RfC #76879: [EthTrcv] Add EthSwt API's to optional interfaces

  Problem description:

  Optional interfaces are missing, in case the Ethernet transceiver is connected to a Ethernet switch

  Agreed solution:

  add the following API's to optional interfaces:
  EthSwt_ReadTrcvRegister
  EthSwt_WriteTrcvRegister

  move the following API's from mandatory interfaces to optional interfaces:
  Eth_ReadMii
  Eth_WriteMii
  –Last change on issue 76879 comment 1–

  BW-C-Level:

Application	Specification	Bus
1	1	1

1.81 Specification Item SWS_EthSwt_00107

Trace References:

SRS_BSW_00406

Content:

If development error detection is enabled: the function EthSwt_ GetCounterValues shall check that the service EthSwt_SwitchInit was previously called. If the check fails, the function shall raise the development error ETHSWT_E_NOT_INITIALIZED and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

  For Error Detection following general rules apply:
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being

valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.82 Specification Item SWS_EthSwt_00108

Trace References:

SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchIdx is not valid,EthSwt_GetCounterValues shall raise the development error ETHSWT_E_INV_SWITCH_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

  For Error Detection following general rules apply:
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
  [SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.

[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.83 Specification Item SWS_EthSwt_00111

Trace References:

SRS_ETH_00086, SRS_ETH_00087

Content:

Service name:	EthSwt_GetArlTableEthSwt_GetArlTable	
Syntax:	Std_ReturnType EthSwt_GetArlTable(uint8 switchIdx, uint16* numberOfElements, Eth_MacVlanType* arlTableListPointer)	
Service ID[hex]:	0x0a	
Sync/Async:	Synchronous /Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	switchIdxEthSwt_GetArlTable.switchIdx	Index of the switch within the context of the Ethernet Switch Driver

Parameters (inout):	numberOfElementsEthSwt_GetArlTable.numberOfElements	In: Maximum number of elements which can be written into the arlTable Out: Number of elements which are currently available in the EthSwitch module.
Parameters (out):	arlTableListPointerEthSwt_GetArlTable.arlTableListPointer	Returns a pointer to the memory where the ARL table of the switch consisting of a list of structs with MAC-address, VLAN-ID and port shall be stored.
Return value:	Std_ReturnType	E_OK: success E_NOT_OK: requested switchIdx is not valid or inactive
Description:	Obtains the address resolution table of a switch and copies the list into a user provided buffer. The function will copy all or numberOfElements into the output list. If input value of numberOfElements is 0 the function will not copy any data but only return the number of valid entries in the cache. arlTableListPointer may be NULL_PTR in this case.	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76248: [EthSwt]Remove references to non existing requirements (SRS, SWS)

  Problem description:

  Some review findings brought up references to non existing requirements. Please check and remove the references and if needed reformulate the requirements.

  Agreed solution:

  SWS_EthSwt_00123 –> SRS_BSW_00406
  SWS_EthSwt_00165 -> SRS_BSW_00395
  SWS_EthSwt_00227 -> SRS_ETH_00087
  SWS_EthSwt_91017, SWS_EthSwt_91020, –> SRS_ETH_00123
  SWS_EthSwt_00006 –> SRS_BSW_00101
  SWS_EthSwt_00010 left blank as SRS doe not yet provide a requirement here but maybe with RFC # 75170 AUTOSAR will introduce an SRS-item after all.
  SWS_EthSwt_00376, SWS_EthSwt_00374, SWS_EthSwt_00375 –> SRS-Missing! same as for SWS_EThswt_00010
  SWS_EthSwt_00018, SRS_EthSwt_00025 –> SRS_ETH_00118
  SWS_EthSwt_00031 –> SRS_ETH_00087
  SWS_EthSwt_00037 –> SRS_ETH_00119
  SWS_EthSwt_00044, SWS_EthSwt_00051 –> SRS_ETH_00118
  SWS_EthSwt_00060 –> SRS_ETH_00087
  SWS_EthSwt_00111 –> erase SRS_ETH_00086
  SWS_EthSwt00079 –>SRS_ETH_00119
  SWS_EthSwt_00206, SWS_EthSwt_00211, SWS_EthSwt_00216, SWS_EthSwt_00221 –>SRS_Eth_00120
  SWS_EthSwt_00172 –>SRS_ETH_00121, SRS_ETH_00114
  SWS_EthSwt_00086, SWS_EthSwt_00091, SWS_EthSwt_00182,

SWS_EthSwt_00125 –> SRS_ETH_00087, SRS_ETH_00122
SWS_EthSwt_00187 –> SRS_ETH_00087
SWS_EthSwt_00058 –> SRS_BSW_00171
SWS_EthSwt_91012, SWS_EthSwt_91013 –> SRS_ETH_00126
SWS_EthSwt_91014, SWS_EthSwt_91015, SWS_EthSwt_91016, SWS_EthSwt_91018, SWS_EthSwt_91019, SWS_EthSwt_91021, SWS_EthSwt_91022, SWS_EthSwt_91022, SWS_EthSwt_91023, SWS_EthSwt_91024, SWS_EthSwt_91025, SWS_EthSwt_91026, SWS_EthSwt_91027 –> SRS_Eth_00123
SWS_EthSwt_00117, SWS_EthSwt_00203 –> SRS_ETH_00119, SRS_ETH_00087
SWS_EthSwt_00193 –> SRS_ETH_00122, SRS_ETH_00087
SWS_EthSwt_00114 –> SRS_BSW_00433
SWS_EthSwt_00098 –> SRS_Eth_00122, SRS_ETH_00118, SRS_ETH_00119, SRS_ETH_00120, SRS_ETH_00087, SRS_ETH_00125, SRS_BSW_00375
–Last change on issue 76248 comment 13–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.84 Specification Item SWS_EthSwt_00112

Trace References:

SRS_BSW_00406

Content:

If development error detection is enabled: the function EthSwt_GetArlTable shall check that the service EthSwt_SwitchInit was previously called. If the check fails, the function shall raise the development error ETHSWT_E_NOT_INITIALIZED and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

In the above requirement -
1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.

[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.85 Specification Item SWS_EthSwt_00113

Trace References:

SRS_BSW_00385

Content:

Error Name:	ETHSWT_E_ACCESS	
Short Description:	Ethernet Switch Access Failure	
Long Description:	This production error shall be issued when the switch is not accessible.	
Recommended DTC:	N/A	
Detection Criteria:	Fail	If during initialization the switch cannot be configured and a ETHSWT_E_ACCESS error is reported by the API call. Before the initialization of the switch hardware is executed this condition can be reseted. When access to the Ethernet Switch fails the module shall report the extended production error with event status DEM_EVENT_STATUS_PREFAILED to DEM.
	Pass	If no ETHSWT_E_ACCESS is reportedWhen access to the Ethernet Switch succeeds the module shall report the extended production error with event status DEM_EVENT_STATUS_PREPASSED to DEM.
Secondary Parameters:	N/A	

Time Required:	N/A
Monitor Frequency	N/A
MIL illumination:	N/A

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77250: [EthIf][EthSwt][EthTrcv] EthSwt_SetSwitchPortMode() broken for ports without Ethernet transceiver

  Problem description:

  SWS_EthSwt_00019 states that EthSwt_SetSwitchPortMode() shall (unconditionally) invoke EthTrcv_SetTransceiverMode() of the Ethernet Transceiver Driver.

  This is bogus for the following two reasons:
  1) a particular switch port does not necessarily have an associated Ethernet transceiver. - In that case EthSwtPortTrcvRef won't be present. This is probably the case for MAC2MAC links if EthSwtPortRole is present an either set to ETHSWT_HOST_PORT or to ETHSWT_UP_LINK_PORT.
  2) Even if the particular switch port has an associated Ethernet transceiver, SWS_EthTrcv_00043 states that the EthTrcv shall invoke EthIf_TrcvModeIndication() as a response to EthTrcv_SetTransceiverMode().

  IMHO 1) should be solved by extending SWS_EthSwt_00019 distinguishing between ports which are associated with a transceiver (i.e., EthSwtPortTrcvRef present) - in that case EthTrcv_SetTransceiverMode() should be called - and ports which are not associated with a transceiver (i.e., EthSwtPortTrcvRef is not present) - in that case something else has to happen and this needs to be specified (probably the switch port shall be internally just disabled - maybe this needs to happen in the first case as well?)

  2) leads to the following call chain: EthIf_SwitchPortGroupRequestMode() -> EthSwt_SetSwitchPortMode() -> EthTrcv_SetTransceiverMode() -> EthIf_TrcvModeIndication(). Thus the indication actually bypasses the EthSwt which IMHO is extremely ugly. Additionally for switch ports without associated transceiver, the EthIf will never get a EthIf_TrcvModeIndication() ...

  Agreed solution:

  === System Template ===
  + Add upstream mapping of EthSwtPortTrcvRef to CouplingPort.physicalLayerType

  === EthSwt ===

~ch. 7.2.4 Production Errors
move SWS_EthSwt_00113 to 7.2.5 Extended production Errors
~SWS_EthSwt_00113 change detection criteria to
Fail:When access to the Ethernet Switch fails the module shall report the extended production error with event status DEM_EVENT_STATUS_PREFAILED to DEM.
Pass:When access to the Ethernet Switch succeeds the module shall report the extended production error with event status DEM_EVENT_STATUS_PREPASSED to DEM.

+ SWS_EthSwt_xxxxx new ext. production Error
Error Name: ETHSWT_E_SYNCPORT2PHY
Short Description: Ethernet switch port and the referenced Ethernet transceiver are in contradicting modes.
Long description: While getting the Ethernet switch port mode, the Ethernet switch driver detected an inconsistent state between Ethernet switch port and the referenced Ethernet transceiver Mode.
Detection Criteria: Fail: When getting the Ethernet switch port mode together with the Ethernet transceiver mode and the mode of the two referenced modules was found
inconsistent the module shall report the extended production error with event status DEM_EVENT_STATUS_PREFAILED to DEM.
Pass:When getting the Ethernet switch port mode together with the Ethernet transceiver mode and the mode of the two referenced modules was found
consistent the module shall report the extended production error with event status DEM_EVENT_STATUS_PREPASSED to DEM.
all other attributer: N/A

~ ch. 8.3.2 EthSwt_SwitchInit
~[SWS_EthSwt_00016] change "production error" to "extended production error"

~ch. 8.3.3 EthSwt_SetSwitchPortMode
~[SWS_EthSwt_00019]The function EthSwt_SetSwitchPortMode() shall put the indexed port of the switch into the specified mode. If EthSwtPort references an EthTrcv then the function EthTrcv_SetTransceiverMode() of the Ethernet Transceiver Driver shall additionally be called with the corresponding transceiver mode.

+[SWS_EthSwt_xxxxx] When calling the function EthSwt_SetSwitchPortMode() with mode ETHTRCV_MODE_DOWN, the EthSwt shall disable the Ethernet switch port directly for reduction of power consumption, if it is possible.

+[SWS_EthSwt_xxxxx] When calling the function EthSwt_SetSwitchPortMode(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.

+[SWS_EthSwt_xxxxxx] If EthSwtPort does not references an EthTrcv, EthSwt shall indicate a mode of the port by the API EthIf_SwitchPortModeIndication latest during the next EthSwt_MainFunction.(SRS_ETH_00118)

~ch. 10.1.3 EthSwtDemEventParameterRefs
+ SWS Item ECUC_EthSwt_xxxxx1 : Name ETHSWT_E_SYNCPORT2PHY
Parent Container EthSwtDemEventParameterRefs
Description Reference to the DemEventParameter which shall be issued when the error "Ethernet switch port and the referenced Ethernet transceiver are in contradicting modes" has occurred.
Multiplicity 0..1
Type Symbolic name reference to [DemEventParameter]
Post-Build Variant Multiplicity true
Post-Build Variant Value true
Multiplicity Configuration
Class Pre-compile time X VARIANT-PRE-COMPILE
Link time X VARIANT-LINK-TIME
Post-build time X VARIANT-POST-BUILD
Value Configuration Class
Pre-compile time X VARIANT-PRE-COMPILE
Link time X VARIANT-LINK-TIME
Post-build time X VARIANT-POST-BUILD
Scope / Dependency
scope: local

~ch. 10.1.5 EthSwtPort
~ECUC_EthSwt_00041 EthSwtPortTrcvRef change following attributes:
Multiplicity Configuration Class | Link time | VARIANT-LINK-TIME, VARIANT-POST-BUILD
| Post build | –
add to dependency: If EthSwtPortPhysicalLayerType is defined, then EthSwtPortTrcvRef holds the reference to the corresponding EthTrcv.

~ECUC_EthSwt_00054 add dependency to EthSwtPortPhysicalLayerType: If a EthSwtPort has an EthSwtPortPhysicalLayerType then EthSwtPort shall reference an EthTrcv.

=== EthIf ===
Add to chapter 8.4 (callback notifications):
+ EthIf_SwitchPortModeIndication
Syntax: void EthIf_SwitchPortModeIndication(uint8 SwitchIdx, uint8 SwitchPortIdx, EthSwt_ModeType PortMode)
Sevice ID: pick a free one
Sync/Async: Synchronous
Reentrancy: Non Reentrant
Parmaters in: SwitchIdx: Index of the switch within the context of the Ethernet Switch Driver
SwitchPortIdx: Index of the port at the addressed switch
PortMode: notified Ethernet Switch port mode.
Description: The EthIf shall determine the expected notifications based on the EthSwtPort configuration. In case the EthSwtPort references an EthTrcv the EthIf expects a notification from the EthTrcv via API EthIf_TrcvModeIndication(). Otherwise the EthIf expects a notification from the EthSwt via API EthIf_SwitchPortModeIndication().

~SWS_EthIf_00232
change from
Called asynchronously when mode has been read out. Triggered by previous
EthTrcv_SetTransceiverMode call. Can directly be called within the trigger
functions.
To
Called asynchronously when a mode change has been read out. If the function is triggered by previous call of
EthTrcv_SetTransceiverMode it can directly be called within the trigger function.
–Last change on issue 77250 comment 50–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.86 Specification Item SWS_EthSwt_00114

Trace References:

SRS_ETH_00086 BSW_00433

Content:

Service name:	EthSwt_MainFunctionEthSwt_MainFunction

Syntax:	void EthSwt_MainFunction(void)
Service ID[hex]:	0x1c
Sync/Async:	Synchronous
Reentrancy:	Non Reentrant
Parameters (in):	None
Parameters (inout):	None
Parameters (out):	None
Return value:	None
Description:	Service to support asynchronous behavior of API calls

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76248: [EthSwt]Remove references to non existing requirements (SRS, SWS)

  Problem description:

  Some review findings brought up references to non existing requirements. Please check and remove the references and if needed reformulate the requirements.

  Agreed solution:

  SWS_EthSwt_00123 –> SRS_BSW_00406
  SWS_EthSwt_00165 -> SRS_BSW_00395
  SWS_EthSwt_00227 -> SRS_ETH_00087
  SWS_EthSwt_91017, SWS_EthSwt_91020, –> SRS_ETH_00123
  SWS_EthSwt_00006 –> SRS_BSW_00101
  SWS_EthSwt_00010 left blank as SRS doe not yet provide a requirement here but maybe with RFC # 75170 AUTOSAR will introduce an SRS-item after all.
  SWS_EthSwt_00376, SWS_EthSwt_00374, SWS_EthSwt_00375 –> SRS-Missing! same as for SWS_EThSwt_00010
  SWS_EthSwt_00018, SRS_EthSwt_00025 –> SRS_ETH_00118
  SWS_EthSwt_00031 –> SRS_ETH_00087
  SWS_EthSwt_00037 –> SRS_ETH_00119
  SWS_EthSwt_00044, SWS_EthSwt_00051 –> SRS_ETH_00118
  SWS_EthSwt_00060 –> SRS_ETH_00087
  SWS_EthSwt_00111 –> erase SRS_ETH_00086
  SWS_EthSwt00079 –>SRS_ETH_00119
  SWS_EthSwt_00206, SWS_EthSwt_00211, SWS_EthSwt_00216, SWS_EthSwt_00221 –>SRS_Eth_00120
  SWS_EthSwt_00172 –>SRS_ETH_00121, SRS_ETH_00114
  SWS_EthSwt_00086, SWS_EthSwt_00091, SWS_EthSwt_00182, SWS_EthSwt_00125 –> SRS_ETH_00087, SRS_ETH_00122

SWS_EthSwt_00187 –> SRS_ETH_00087
SWS_EthSwt_00058 –> SRS_BSW_00171
SWS_EthSwt_91012, SWS_EthSwt_91013 –> SRS_ETH_00126
SWS_EthSwt_91014, SWS_EthSwt_91015, SWS_EthSwt_91016, SWS_EthSwt_91018, SWS_EthSwt_91019, SWS_EthSwt_91021, SWS_EthSwt_91022, SWS_EthSwt_91022, SWS_EthSwt_91023, SWS_EthSwt_91024, SWS_EthSwt_91025, SWS_EthSwt_91026, SWS_EthSwt_91027 –> SRS_Eth_00123
SWS_EthSwt_00117, SWS_EthSwt_00203 –> SRS_ETH_00119, SRS_ETH_00087
SWS_EthSwt_00193 –> SRS_ETH_00122, SRS_ETH_00087
SWS_EthSwt_00114 –> SRS_BSW_00433
SWS_EthSwt_00098 –> SRS_Eth_00122, SRS_ETH_00118, SRS_ETH_00119, SRS_ETH_00120, SRS_ETH_00087, SRS_ETH_00125, SRS_BSW_00375
–Last change on issue 76248 comment 13–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.87 Specification Item SWS_EthSwt_00117

Trace References:

SRS_ETH_0008600119, SRS_ETH_00087

Content:

Service name:	<User>_EthSwtLinkDown<UserCallout>_<EthSwtLinkDownCallout>	
Syntax:	void <User>_EthSwtLinkDownCallout>(uint8* uint8 SwitchIdxPtr, uint8* uint8 PortIdx Ptr)	
Service ID[hex]:	0x19	
Sync/Async:	Synchronous /Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	None	
	Parameters (inout):	
	Parameters (out):	SwitchIdxPtr<User>_EthSwtLinkDown Callout>.SwitchIdx Ptr
PortIdxPtr<User>_EthSwtLinkDown Callout>.PortIdx Ptr		Pointer to the port index Index of the port at the addressed switch
Parameters (inout):	None	

Parameters (out):	None
Return value:	None
Description:	Shall be Is called, if a link which is configured for .1X goes down(link loss) goes down.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76248: [EthSwt]Remove references to non existing requirements (SRS, SWS)

  Problem description:

  Some review findings brought up references to non existing requirements. Please check and remove the references and if needed reformulate the requirements.

  Agreed solution:

  SWS_EthSwt_00123 –> SRS_BSW_00406
  SWS_EthSwt_00165 -> SRS_BSW_00395
  SWS_EthSwt_00227 -> SRS_ETH_00087
  SWS_EthSwt_91017, SWS_EthSwt_91020, –> SRS_ETH_00123
  SWS_EthSwt_00006 –> SRS_BSW_00101
  SWS_EthSwt_00010 left blank as SRS doe not yet provide a requirement here but maybe with RFC # 75170 AUTOSAR will introduce an SRS-item after all.
  SWS_EthSwt_00376, SWS_EthSwt_00374, SWS_EthSwt_00375 –> SRS-Missing! same as for SWS_EThSwt_00010
  SWS_EthSwt_00018, SRS_EthSwt_00025 –> SRS_ETH_00118
  SWS_EthSwt_00031 –> SRS_ETH_00087
  SWS_EthSwt_00037 –> SRS_ETH_00119
  SWS_EthSwt_00044, SWS_EthSwt_00051 –> SRS_ETH_00118
  SWS_EthSwt_00060 –> SRS_ETH_00087
  SWS_EthSwt_00111 –> erase SRS_ETH_00086
  SWS_EthSwt00079 –>SRS_ETH_00119
  SWS_EthSwt_00206, SWS_EthSwt_00211, SWS_EthSwt_00216, SWS_EthSwt_00221 –>SRS_Eth_00120
  SWS_EthSwt_00172 –>SRS_ETH_00121, SRS_ETH_00114
  SWS_EthSwt_00086, SWS_EthSwt_00091, SWS_EthSwt_00182, SWS_EthSwt_00125 –> SRS_ETH_00087, SRS_ETH_00122
  SWS_EthSwt_00187 –> SRS_ETH_00087
  SWS_EthSwt_00058 –> SRS_BSW_00171
  SWS_EthSwt_91012, SWS_EthSwt_91013 –> SRS_ETH_00126
  SWS_EthSwt_91014, SWS_EthSwt_91015, SWS_EthSwt_91016, SWS_EthSwt_91018, SWS_EthSwt_91019, SWS_EthSwt_91021, SWS_EthSwt_91022, SWS_EthSwt_91022, SWS_EthSwt_91023,

SWS_EthSwt_91024, SWS_EthSwt_91025, SWS_EthSwt_91026, SWS_EthSwt_91027 –> SRS_Eth_00123
SWS_EthSwt_00117, SWS_EthSwt_00203 –> SRS_ETH_00119, SRS_ETH_00087
SWS_EthSwt_00193 –> SRS_ETH_00122, SRS_ETH_00087
SWS_EthSwt_00114 –> SRS_BSW_00433
SWS_EthSwt_00098 –> SRS_Eth_00122, SRS_ETH_00118, SRS_ETH_00119, SRS_ETH_00120, SRS_ETH_00087, SRS_ETH_00125, SRS_BSW_00375
–Last change on issue 76248 comment 13–

BW-C-Level:

Application	Specification	Bus
1	1	1

- RfC #76531: [EthSwt] Syntax for callback functions <user>_LinkUp and <user>_LinkDown might be wrong

Problem description:

The callback function for <User>_LinkUp and <User>_LinkDown to inform the user about a Link state change might be wrong
Currently the parameter of these both functions are defined as Pointer to uint8
This makes no sense for me to use here pointers and not a uint 8 variable.

Agreed solution:

Chapter 8.6.3 Configurable Interfaces
add following intro:
In this chapter all interfaces are listed where the target function could be configured.
The names of these kind of interfaces are not fixed because they are configurable.

Add Subchapter with interface
move SWS_EthSwt_00117 here and change to:
Service name: <EthSwtLinkDownCallout>
syntax: void <EthSwtLinkDownCallout>(uint8 SwitchIdx, uint8 PortIdx)
Service ID: 0x19
Sync/Async:Synchronous
Reentrancy: Non Reentrant
Parameters (in): SwitchIdx: Index of the switch within the context of the Ethernet Switch Driver
PortIdx: Index of the port at the addressed switch
Parameters (in/out): none
Parameters (out): none
Return value: none
Description: Is called, if a link which is configured goes down.

~[SWS_EthSwt_00118]The function <EthSwtLinkDownCallout> shall be called if a link, which is configured, goes down. The function provides the Switch index and the Port index, such that the port which went down can be identified.

-[SWS_EthSwt_00119]

add second subchapter:

move SWS_EthSwt_00203 here and change to:
Service name: <EthSwtLinkUpCallout>
syntax: void <EthSwtLinkUpCallout>(uint8 SwitchIdx, uint8 PortIdx)
Service ID: 0x1a
Sync/Async:Synchronous
Reentrancy: Non Reentrant
Parameters (in): SwitchIdx: Index of the switch within the context of the Ethernet Switch Driver
PortIdx: Index of the port at the addressed switch
Parameters (in/out): none
Parameters (out): none
Return value: none
Description: Is called, if a link which is configured goes up

~[SWS_EthSwt_00204]The function <EthSwtLinkUpCallout> shall be called if a link which is configured goes up. The function provides the Switch index and the Port index, such that the port which went up can be identified.

-[SWS_EthSwt_00205] This is no clear as it is a configurable interface now

~ECUC_EthSwt_00048 set EthSwtLinkDownUser to obsolete
+ add optional function name parameter with same attributes named EthSwtLinkDownCallout
change description to Defines the function name for <EthSwtLinkDownCallout>.

~ECUC_EthSwt_00068 set EthSwtLinkUpUser to obsolete
+add optional function name parameter with same attributes named EthSwtLinkUpCallout
change description to Defines the function name for <EthSwtLinkUpCallout>.

set ECUC_EthSwt_00047 EthSwtPortEnableLinkDownCallback to obsolete.

Header File name is defined by ECUC_EthSwt_00064 EthSwtPublicCddHeaderFile.
–Last change on issue 76531 comment 54–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.88 Specification Item SWS_EthSwt_00118

Trace References:

SRS_ETH_00119, SRS_ETH_00087

Content:

The function <User>_EthSwtLinkDownCallout> shall be called if a link, which is configuredfor .1X , goes down (link loss). The function returns provides the Switch index and the Port index, such that the port which went down can be identified.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76531: [EthSwt] Syntax for callback functions <user>_LinkUp and <user>_LinkDown might be wrong

  Problem description:

  The callback function for <User>_LinkUp and <User>_LinkDown to inform the user about a Link state change might be wrong
  Currently the parameter of these both functions are defined as Pointer to uint8
  This makes no sense for me to use here pointers and not a uint 8 variable.

  Agreed solution:

  Chapter 8.6.3 Configurable Interfaces
  add following intro:
  In this chapter all interfaces are listed where the target function could be configured.
  The names of these kind of interfaces are not fixed because they are configurable.

  Add Subchapter with interface
  move SWS_EthSwt_00117 here and change to:
  Service name: <EthSwtLinkDownCallout>
  syntax: void <EthSwtLinkDownCallout>(uint8 SwitchIdx, uint8 PortIdx)
  Service ID: 0x19
  Sync/Async:Synchronous

Reentrancy: Non Reentrant
Parameters (in): SwitchIdx: Index of the switch within the context of the Ethernet Switch Driver
PortIdx: Index of the port at the addressed switch
Parameters (in/out): none
Parameters (out): none
Return value: none
Description: Is called, if a link which is configured goes down.

~[SWS_EthSwt_00118]The function <EthSwtLinkDownCallout> shall be called if a link, which is configured, goes down. The function provides the Switch index and the Port index, such that the port which went down can be identified.

-[SWS_EthSwt_00119]

add second subchapter:

move SWS_EthSwt_00203 here and change to:
Service name: <EthSwtLinkUpCallout>
syntax: void <EthSwtLinkUpCallout>(uint8 SwitchIdx, uint8 PortIdx)
Service ID: 0x1a
Sync/Async:Synchronous
Reentrancy: Non Reentrant
Parameters (in): SwitchIdx: Index of the switch within the context of the Ethernet Switch Driver
PortIdx: Index of the port at the addressed switch
Parameters (in/out): none
Parameters (out): none
Return value: none
Description: Is called, if a link which is configured goes up

~[SWS_EthSwt_00204]The function <EthSwtLinkUpCallout> shall be called if a link which is configured goes up. The function provides the Switch index and the Port index, such that the port which went up can be identified.

-[SWS_EthSwt_00205] This is no clear as it is a configurable interface now

~ECUC_EthSwt_00048 set EthSwtLinkDownUser to obsolete
+ add optional function name parameter with same attributes named EthSwtLinkDownCallout
change description to Defines the function name for <EthSwtLinkDownCallout>.

~ECUC_EthSwt_00068 set EthSwtLinkUpUser to obsolete
+add optional function name parameter with same attributes named EthSwtLinkUpCallout
change description to Defines the function name for <EthSwtLinkUpCallout>.

set ECUC_EthSwt_00047 EthSwtPortEnableLinkDownCallback to obsolete.

Header File name is defined by ECUC_EthSwt_00064 EthSwtPublicCddHeaderFile.
–Last change on issue 76531 comment 54–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.89 Specification Item SWS_EthSwt_00119

Trace References:

SRS_BSW_00171

Content:

The function <User>_LinkDown shall be pre compile time configurable by the <user> with content of EthSwtLinkDownUser.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76531: [EthSwt] Syntax for callback functions <user>_LinkUp and <user>_LinkDown might be wrong

  Problem description:

  The callback function for <User>_LinkUp and <User>_LinkDown to inform the user about a Link state change might be wrong
  Currently the parameter of these both functions are defined as Pointer to uint8
  This makes no sense for me to use here pointers and not a uint 8 variable.

  Agreed solution:

  Chapter 8.6.3 Configurable Interfaces
  add following intro:
  In this chapter all interfaces are listed where the target function could be configured.
  The names of these kind of interfaces are not fixed because they are configurable.

Add Subchapter with interface
move SWS_EthSwt_00117 here and change to:
Service name: <EthSwtLinkDownCallout>
syntax: void <EthSwtLinkDownCallout>(uint8 SwitchIdx, uint8 PortIdx)
Service ID: 0x19
Sync/Async:Synchronous
Reentrancy: Non Reentrant
Parameters (in): SwitchIdx: Index of the switch within the context of the Ethernet Switch Driver
PortIdx: Index of the port at the addressed switch
Parameters (in/out): none
Parameters (out): none
Return value: none
Description: Is called, if a link which is configured goes down.

~[SWS_EthSwt_00118]The function <EthSwtLinkDownCallout> shall be called if a link, which is configured, goes down. The function provides the Switch index and the Port index, such that the port which went down can be identified.

-[SWS_EthSwt_00119]

add second subchapter:

move SWS_EthSwt_00203 here and change to:
Service name: <EthSwtLinkUpCallout>
syntax: void <EthSwtLinkUpCallout>(uint8 SwitchIdx, uint8 PortIdx)
Service ID: 0x1a
Sync/Async:Synchronous
Reentrancy: Non Reentrant
Parameters (in): SwitchIdx: Index of the switch within the context of the Ethernet Switch Driver
PortIdx: Index of the port at the addressed switch
Parameters (in/out): none
Parameters (out): none
Return value: none
Description: Is called, if a link which is configured goes up

~[SWS_EthSwt_00204]The function <EthSwtLinkUpCallout> shall be called if a link which is configured goes up. The function provides the Switch index and the Port index, such that the port which went up can be identified.

-[SWS_EthSwt_00205] This is no clear as it is a configurable interface now

~ECUC_EthSwt_00048 set EthSwtLinkDownUser to obsolete
+ add optional function name parameter with same attributes named EthSwtLinkDownCallout
change description to Defines the function name for <EthSwtLinkDownCallout>.

~ECUC_EthSwt_00068 set EthSwtLinkUpUser to obsolete
+add optional function name parameter with same attributes named EthSwtLinkUpCallout
change description to Defines the function name for <EthSwtLinkUpCallout>.

set ECUC_EthSwt_00047 EthSwtPortEnableLinkDownCallback to obsolete.

Header File name is defined by ECUC_EthSwt_00064 EthSwtPublicCddHeaderFile.
–Last change on issue 76531 comment 54–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.90 Specification Item SWS_EthSwt_00122

Trace References:

none

Content:

The EthSwt_MainFunction shall call the API EthSwt_GetCounterValues and shall check each single value of referenced by CounterPtr:

1. If the first value is greater than zero, the function shall raise the development error ETHSWT_E_BUFFEROVERRUN
2. If the second value is greater than zero, the function shall raise the development error ETHSWT_E_CRC
3. If the third value is greater than zero, the function shall raise the development error ETHSWT_E_UNDERSIZEPCKT

4. If the forth value is greater than zero, the function shall raise the development error ETHSWT_E_OVERSIZEPCKT
5. If the fifth value is greater than zero, the function shall raise the development error ETHSWT_E_ALIGNMENT
6. If the sixth value is greater than zero, the function shall raise the development error ETHSWT_E_SQETEST
7. If the seventh value is greater than zero, the function shall raise the development error ETHSWT_E_INDISCARD
8. If the eighth value is greater than zero, the function shall raise the development error ETHSWT_E_INERROR
9. If the ninth value is greater than zero, the function shall raise the development error ETHSWT_E_OUTDISCARD
10. If the tenth value is greater than zero, the function shall raise the development error ETHSWT_E_OUTERROR
11. If the 11th value is greater than zero, the function shall raise the development error ETHSWT_E_SINGLECOLLISION
12. If the 12th value is greater than zero, the function shall raise the development error ETHSWT_E_MULTIPLECOLLISION
13. If the 13th value is greater than zero, the function shall raise the development error ETHSWT_E_DEFFEREDTRANSMISSION
14. If the 14th value is greater than zero, the function shall raise the development error ETHSWT_E_LATECOLLISION
15. If the eleventh value is greater than zero, the function shall raise the development error ETHSWT_E_DROPCOUNTER

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76392: [EthSwt] Extended production Errors set by Main Function using an own interface function is strange approach

  Problem description:

  In chapter 8.5 SWS_EthSwt_00122 states to call the API EthSwt_GetCountervaValues in the cyclic Main-function and report production errors for every single counter.
  this is really strange.
  1st: Why shall the main function call an external interface. It could simply fetch the data directly from HW

2nd: Not every Counter of this API directly implies an error on the bus. It simply gives a statistic of events on the bus which than carefully interpreted are resulting in a diagnostic result of the bus. Setting an error on values greater zero will lead to a unnecessary errors counted here.
3rd: calling this API cyclically with fetching about 15 or more bytes from HW on every main cycle, may lead to poor performance of the driver.

Agreed solution:

Take out SWS_EThSwt_00122 and all related extended production error. SWS_EthSwt_00137, SWS_EthSwt_00138, SWS_EthSwt_00139, SWS_EthSwt_00141,SWS_EthSwt_00142 to SWS_EthSwt_00152.

Add following text in 7.2.5:
There are no extended production errors.
–Last change on issue 76392 comment 5–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.91 Specification Item SWS_EthSwt_00123

Trace References:

SRS_ETH_00086 BSW_00406

Content:

Name:	EthSwt_StateTypeEthSwt_StateType		
Type:	Enumeration		
Range:	ETHSWT_STATE_UNINITEth Swt_State Type.ETHSWT_STATE_UNINIT	0x00	Driver Switch is not yet configured
	ETHSWT_STATE_INITEth Swt_State Type.ETHSWT_STATE_INIT	0x01	Driver Switch is configured
	ETHSWT_STATE_ACTIVEEth Swt_State Type.ETHSWT_STATE_ACTIVE	0x02	Driver Switch is active
Description:	Status supervision used for Development Error Detection. The state shall be available for debugging.		

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76248: [EthSwt]Remove references to non existing requirements (SRS, SWS)

  Problem description:

  Some review findings brought up references to non existing requirements. Please check and remove the references and if needed reformulate the requirements.

  Agreed solution:

  SWS_EthSwt_00123 –> SRS_BSW_00406
  SWS_EthSwt_00165 -> SRS_BSW_00395
  SWS_EthSwt_00227 -> SRS_ETH_00087
  SWS_EthSwt_91017, SWS_EthSwt_91020, –> SRS_ETH_00123
  SWS_EthSwt_00006 –> SRS_BSW_00101
  SWS_EthSwt_00010 left blank as SRS doe not yet provide a requirement here but maybe with RFC # 75170 AUTOSAR will introduce an SRS-item after all.
  SWS_EthSwt_00376, SWS_EthSwt_00374, SWS_EthSwt_00375 –> SRS-Missing! same as for SWS_EThSwt_00010
  SWS_EthSwt_00018, SRS_EthSwt_00025 –> SRS_ETH_00118
  SWS_EthSwt_00031 –> SRS_ETH_00087
  SWS_EthSwt_00037 –> SRS_ETH_00119
  SWS_EthSwt_00044, SWS_EthSwt_00051 –> SRS_ETH_00118
  SWS_EthSwt_00060 –> SRS_ETH_00087
  SWS_EthSwt_00111 –> erase SRS_ETH_00086
  SWS_EthSwt00079 –>SRS_ETH_00119
  SWS_EthSwt_00206, SWS_EthSwt_00211, SWS_EthSwt_00216, SWS_EthSwt_00221 –>SRS_Eth_00120
  SWS_EthSwt_00172 –>SRS_ETH_00121, SRS_ETH_00114
  SWS_EthSwt_00086, SWS_EthSwt_00091, SWS_EthSwt_00182, SWS_EthSwt_00125 –> SRS_ETH_00087, SRS_ETH_00122
  SWS_EthSwt_00187 –> SRS_ETH_00087
  SWS_EthSwt_00058 –> SRS_BSW_00171
  SWS_EthSwt_91012, SWS_EthSwt_91013 –> SRS_ETH_00126
  SWS_EthSwt_91014, SWS_EthSwt_91015, SWS_EthSwt_91016, SWS_EthSwt_91018, SWS_EthSwt_91019, SWS_EthSwt_91021, SWS_EthSwt_91022, SWS_EthSwt_91022, SWS_EthSwt_91023, SWS_EthSwt_91024, SWS_EthSwt_91025, SWS_EthSwt_91026, SWS_EthSwt_91027 –> SRS_Eth_00123
  SWS_EthSwt_00117, SWS_EthSwt_00203 –> SRS_ETH_00119, SRS_ETH_00087
  SWS_EthSwt_00193 –> SRS_ETH_00122, SRS_ETH_00087
  SWS_EthSwt_00114 –> SRS_BSW_00433
  SWS_EthSwt_00098 –> SRS_Eth_00122, SRS_ETH_00118, SRS_ETH_00119, SRS_ETH_00120, SRS_ETH_00087, SRS_ETH_00125, SRS_BSW_00375
  –Last change on issue 76248 comment 13–

BW-C-Level:

Application	Specification	Bus
1	1	1

- RfC #76887: [EthSwt] EthSwt_SwitchInit shall change the state of a switch to ACTIVE, not the state of the component.

  Problem description:

  Requirement SWS_EthSwt_00012 states that EthSwt_SwitchInit shall change the state of the component from ETHSWT_STATE_INIT to ETHSWT_STATE_ACTIVE.This translates into the fact that if at least one switch is active, we assume that all switches are active. The subsequent call of a function for a certain switch for which EthSwt_SwitchInit was not called, will assume that the switch is active.

  EthSwt_SwithcInit shall change the state of a switch to ACTIVE, not the state of the component.

  Agreed solution:

  [SWS_EthSwt_00008]
  The function EthSwt_Init shall change the state of all switches controlled by this Switch Driver from
  ETHSWT_STATE_UNINIT to ETHSWT_STATE_INIT.

  [SWS_EthSwt_00012]
  EthSwt_SwitchInit shall change the state of the indexed switch from
  ETHSWT_STATE_INIT to ETHSWT_STATE_ACTIVE.

  SWS_EthSwt_00123
  change description of range to
  ETHSWT_STATE_UNINIT 0x00 Switch is not yet configured
  ETHSWT_STATE_INIT 0x01 Switch is configured
  ETHSWT_STATE_ACTIVE 0x02 Switch is active
  –Last change on issue 76887 comment 12–

  BW-C-Level:

Application	Specification	Bus
1	4	1

1.92 Specification Item SWS_EthSwt_00125

Trace References:

SRS_ETH_0008600087, SRS_ETH_00087 00122

Content:

Service name:	EthSwt_NvmSingleBlockCallbackEthSwt_NvmSingleBlockCallback	
Syntax:	Std_ReturnType EthSwt_NvmSingleBlockCallback(uint8 ServiceId, NvM_RequestResultType JobResult)	
Service ID[hex]:	0x17	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	ServiceIdEthSwt_NvmSingleBlock Callback.ServiceId	Unique Service ID of NVRAM manager service
	JobResultEthSwt_NvmSingleBlock Callback.JobResult	Covers the job result of the previous processed single block job.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: success E_NOT_OK: Callback function has not been processed successfully
Description:	Function will be called by the NVRAMManager after the switch configuration has been stored or resetted.	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76248: [EthSwt]Remove references to non existing requirements (SRS, SWS)

  Problem description:

  Some review findings brought up references to non existing requirements. Please check and remove the references and if needed reformulate the requirements.

  Agreed solution:

  SWS_EthSwt_00123 –> SRS_BSW_00406
  SWS_EthSwt_00165 -> SRS_BSW_00395
  SWS_EthSwt_00227 -> SRS_ETH_00087
  SWS_EthSwt_91017, SWS_EthSwt_91020, –> SRS_ETH_00123
  SWS_EthSwt_00006 –> SRS_BSW_00101
  SWS_EthSwt_00010 left blank as SRS doe not yet provide a requirement here but maybe with RFC # 75170 AUTOSAR will introduce an SRS-item after all.
  SWS_EthSwt_00376, SWS_EthSwt_00374, SWS_EthSwt_00375 –> SRS-Missing! same as for SWS_EThSwt_00010

SWS_EthSwt_00018, SRS_EthSwt_00025 –> SRS_ETH_00118
SWS_EthSwt_00031 –> SRS_ETH_00087
SWS_EthSwt_00037 –> SRS_ETH_00119
SWS_EthSwt_00044, SWS_EthSwt_00051 –> SRS_ETH_00118
SWS_EthSwt_00060 –> SRS_ETH_00087
SWS_EthSwt_00111 –> erase SRS_ETH_00086
SWS_EthSwt00079 –>SRS_ETH_00119
SWS_EthSwt_00206, SWS_EthSwt_00211, SWS_EthSwt_00216, SWS_EthSwt_00221 –>SRS_Eth_00120
SWS_EthSwt_00172 –>SRS_ETH_00121, SRS_ETH_00114
SWS_EthSwt_00086, SWS_EthSwt_00091, SWS_EthSwt_00182, SWS_EthSwt_00125 –> SRS_ETH_00087, SRS_ETH_00122
SWS_EthSwt_00187 –> SRS_ETH_00087
SWS_EthSwt_00058 –> SRS_BSW_00171
SWS_EthSwt_91012, SWS_EthSwt_91013 –> SRS_ETH_00126
SWS_EthSwt_91014, SWS_EthSwt_91015, SWS_EthSwt_91016, SWS_EthSwt_91018, SWS_EthSwt_91019, SWS_EthSwt_91021, SWS_EthSwt_91022, SWS_EthSwt_91022, SWS_EthSwt_91023, SWS_EthSwt_91024, SWS_EthSwt_91025, SWS_EthSwt_91026, SWS_EthSwt_91027 –> SRS_Eth_00123
SWS_EthSwt_00117, SWS_EthSwt_00203 –> SRS_ETH_00119, SRS_ETH_00087
SWS_EthSwt_00193 –> SRS_ETH_00122, SRS_ETH_00087
SWS_EthSwt_00114 –> SRS_BSW_00433
SWS_EthSwt_00098 –> SRS_Eth_00122, SRS_ETH_00118, SRS_ETH_00119, SRS_ETH_00120, SRS_ETH_00087, SRS_ETH_00125, SRS_BSW_00375
–Last change on issue 76248 comment 13–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.93 Specification Item SWS_EthSwt_00137

Trace References:

SRS_BSW_00385

Content:

Error Name:	ETHSWT_E_BUFFEROVERRUN
Short Description:	Dropped packet due to buffer overrun in switch

Long Description:	Dropped packet due to buffer overrun in switch	
Recommended DTC:	N/A	
Detection Criteria:	Fail	If main function detects that the corresponding counter value is greater than zero, this error will be reported
	Pass	If no such error is reported.
Secondary Parameters:	N/A	
Time Required:	N/A	
Monitor Frequency	N/A	
MIL illumination:	N/A	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76392: [EthSwt] Extended production Errors set by Main Function using an own interface function is strange approach

  Problem description:

  In chapter 8.5 SWS_EthSwt_00122 states to call the API EthSwt_GetCountervaValues in the cyclic Main-function and report production errors for every single counter.
  this is really strange.
  1st: Why shall the main function call an external interface. It could simply fetch the data directly from HW
  2nd: Not every Counter of this API directly implies an error on the bus. It simply gives a statistic of events on the bus which than carefully interpreted are resulting in a diagnostic result of the bus. Setting an error on values greater zero will lead to a unnecessary errors counted here.
  3rd: calling this API cyclically with fetching about 15 or more bytes from HW on every main cycle, may lead to poor performance of the driver.

  Agreed solution:

  Take out SWS_EThSwt_00122 and all related extended production error. SWS_EthSwt_00137, SWS_EthSwt_00138, SWS_EthSwt_00139, SWS_EthSwt_00141,SWS_EthSwt_00142 to SWS_EthSwt_00152.

  Add following text in 7.2.5:
  There are no extended production errors.
  –Last change on issue 76392 comment 5–

  BW-C-Level:

Application	Specification	Bus
1	4	1

1.94 Specification Item SWS_EthSwt_00138

Trace References:

SRS_BSW_00385

Content:

Error Name:	ETHSWT_E_CRC	
Short Description:	Dropped packet due to CRC error detected in switch	
Long Description:	Dropped packet due to CRC error detected in switch	
Recommended DTC:	N/A	
Detection Criteria:	Fail	If main function detects that the corresponding counter value is greater than zero, this error will be reported
	Pass	If no such error is reported.
Secondary Parameters:	N/A	
Time Required:	N/A	
Monitor Frequency	N/A	
MIL illumination:	N/A	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76392: [EthSwt] Extended production Errors set by Main Function using an own interface function is strange approach

  Problem description:

  In chapter 8.5 SWS_EthSwt_00122 states to call the API EthSwt_GetCountervaValues in the cyclic Main-function and report production errors for every single counter.
  this is really strange.
  1st: Why shall the main function call an external interface. It could simply fetch the data directly from HW
  2nd: Not every Counter of this API directly implies an error on the bus. It simply gives a statistic of events on the bus which than carefully interpreted are resulting in a diagnostic result of the bus. Setting an error on values greater zero will lead to a unnecessary errors counted here.

3rd: calling this API cyclically with fetching about 15 or more bytes from HW on every main cycle, may lead to poor performance of the driver.

Agreed solution:

Take out SWS_EThSwt_00122 and all related extended production error. SWS_EthSwt_00137, SWS_EthSwt_00138, SWS_EthSwt_00139, SWS_EthSwt_00141,SWS_EthSwt_00142 to SWS_EthSwt_00152.

Add following text in 7.2.5:
There are no extended production errors.
–Last change on issue 76392 comment 5–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.95 Specification Item SWS_EthSwt_00139

Trace References:

SRS_BSW_00385

Content:

Error Name:	ETHSWT_E_DROPCOUNT	
Short Description:	Dropped packet due to other reason than buffer overrun or CRC error	
Long Description:	Dropped packet due to other reason than buffer overrun or CRC error	
Recommended DTC:	N/A	
Detection Criteria:	Fail	If main function detects that the corresponding counter value is greater than zero, this error will be reported
	Pass	If no such error is reported.
Secondary Parameters:	N/A	
Time Required:	N/A	
Monitor Frequency	N/A	
MIL illumination:	N/A	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76392: [EthSwt] Extended production Errors set by Main Function using an own interface function is strange approach

  Problem description:

  In chapter 8.5 SWS_EthSwt_00122 states to call the API EthSwt_GetCountervaValues in the cyclic Main-function and report production errors for every single counter.
  this is really strange.
  1st: Why shall the main function call an external interface. It could simply fetch the data directly from HW
  2nd: Not every Counter of this API directly implies an error on the bus. It simply gives a statistic of events on the bus which than carefully interpreted are resulting in a diagnostic result of the bus. Setting an error on values greater zero will lead to a unnecessary errors counted here.
  3rd: calling this API cyclically with fetching about 15 or more bytes from HW on every main cycle, may lead to poor performance of the driver.

  Agreed solution:

  Take out SWS_EThSwt_00122 and all related extended production error. SWS_EthSwt_00137, SWS_EthSwt_00138, SWS_EthSwt_00139, SWS_EthSwt_00141,SWS_EthSwt_00142 to SWS_EthSwt_00152.

  Add following text in 7.2.5:
  There are no extended production errors.
  –Last change on issue 76392 comment 5–

  BW-C-Level:

Application	Specification	Bus
1	4	1

1.96 Specification Item SWS_EthSwt_00141

Trace References:

SRS_BSW_00385

Content:

Error Name:	ETHSWT_E_UNDERSIZEPCKT
Short Description:	An undersized packet occurred

Long Description:	An error due to the occurrence undersized packets which were less than 64 octets long (excluding framing bits, but including FCS octets) and were otherwise well formed. (see IETF RFC 1757)	
Recommended DTC:	N/A	
Detection Criteria:	Fail	If main function detects that the corresponding counter value is greater than zero, this error will be reported
	Pass	If no such error is reported.
Secondary Parameters:	N/A	
Time Required:	N/A	
Monitor Frequency	N/A	
MIL illumination:	N/A	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76392: [EthSwt] Extended production Errors set by Main Function using an own interface function is strange approach

  Problem description:

  In chapter 8.5 SWS_EthSwt_00122 states to call the API EthSwt_GetCountervaValues in the cyclic Main-function and report production errors for every single counter.
  this is really strange.
  1st: Why shall the main function call an external interface. It could simply fetch the data directly from HW
  2nd: Not every Counter of this API directly implies an error on the bus. It simply gives a statistic of events on the bus which than carefully interpreted are resulting in a diagnostic result of the bus. Setting an error on values greater zero will lead to a unnecessary errors counted here.
  3rd: calling this API cyclically with fetching about 15 or more bytes from HW on every main cycle, may lead to poor performance of the driver.

  Agreed solution:

  Take out SWS_EThSwt_00122 and all related extended production error. SWS_EthSwt_00137, SWS_EthSwt_00138, SWS_EthSwt_00139, SWS_EthSwt_00141,SWS_EthSwt_00142 to SWS_EthSwt_00152.

  Add following text in 7.2.5:
  There are no extended production errors.
  –Last change on issue 76392 comment 5–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.97 Specification Item SWS_EthSwt_00142

Trace References:

SRS_BSW_00385

Content:

Error Name:	ETHSWT_E_OVERSIZEPCKT	
Short Description:	An undersized packet occurred	
Long Description:	An error due to the occurrence oversized packets which are longer than 1518 octets (excluding framing bits, but including FCS octets) and were otherwise well formed. (see IETF RFC 1757)	
Recommended DTC:	N/A	
Detection Criteria:	Fail	If main function detects that the corresponding counter value is greater than zero, this error will be reported
	Pass	If no such error is reported.
Secondary Parameters:	N/A	
Time Required:	N/A	
Monitor Frequency	N/A	
MIL illumination:	N/A	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76392: [EthSwt] Extended production Errors set by Main Function using an own interface function is strange approach

  Problem description:

  In chapter 8.5 SWS_EthSwt_00122 states to call the API EthSwt_GetCountervaValues in the cyclic Main-function and report production errors for every single counter.
  this is really strange.
  1st: Why shall the main function call an external interface. It could simply fetch the data directly from HW
  2nd: Not every Counter of this API directly implies an error on the bus. It simply

gives a statistic of events on the bus which than carefully interpreted are resulting in a diagnostic result of the bus. Setting an error on values greater zero will lead to a unnecessary errors counted here.
3rd: calling this API cyclically with fetching about 15 or more bytes from HW on every main cycle, may lead to poor performance of the driver.

Agreed solution:

Take out SWS_EThSwt_00122 and all related extended production error. SWS_EthSwt_00137, SWS_EthSwt_00138, SWS_EthSwt_00139, SWS_EthSwt_00141,SWS_EthSwt_00142 to SWS_EthSwt_00152.

Add following text in 7.2.5:
There are no extended production errors.
–Last change on issue 76392 comment 5–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.98 Specification Item SWS_EthSwt_00143

Trace References:

SRS_BSW_00385

Content:

Error Name:	ETHSWT_E_ALIGNMENT	
Short Description:	Alignment error of an Ethernet frame	
Long Description:	Alignment errors occur if packets are received and are not an integral number of octets in length and do not pass the CRC.	
Recommended DTC:	N/A	
Detection Criteria:	Fail	If main function detects that the corresponding counter value is greater than zero, this error will be reported
	Pass	If no such error is reported.
Secondary Parameters:	N/A	
Time Required:	N/A	
Monitor Frequency	N/A	
MIL illumination:	N/A	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76392: [EthSwt] Extended production Errors set by Main Function using an own interface function is strange approach

  Problem description:

  In chapter 8.5 SWS_EthSwt_00122 states to call the API EthSwt_GetCountervaValues in the cyclic Main-function and report production errors for every single counter.
  this is really strange.
  1st: Why shall the main function call an external interface. It could simply fetch the data directly from HW
  2nd: Not every Counter of this API directly implies an error on the bus. It simply gives a statistic of events on the bus which than carefully interpreted are resulting in a diagnostic result of the bus. Setting an error on values greater zero will lead to a unnecessary errors counted here.
  3rd: calling this API cyclically with fetching about 15 or more bytes from HW on every main cycle, may lead to poor performance of the driver.

  Agreed solution:

  Take out SWS_EThSwt_00122 and all related extended production error. SWS_EthSwt_00137, SWS_EthSwt_00138, SWS_EthSwt_00139, SWS_EthSwt_00141,SWS_EthSwt_00142 to SWS_EthSwt_00152.

  Add following text in 7.2.5:
  There are no extended production errors.
  –Last change on issue 76392 comment 5–

  BW-C-Level:

Application	Specification	Bus
1	4	1

1.99 Specification Item SWS_EthSwt_00144

Trace References:

SRS_BSW_00385

Content:

Error Name:	ETHSWT_E_SQETEST
Short Description:	SQE test error

Long Description:	SQE test error according to IETF RFC1643 dot3StatsSQETestErrors	
Recommended DTC:	N/A	
Detection Criteria:	Fail	If main function detects that the corresponding counter value is greater than zero, this error will be reported
	Pass	If no such error is reported.
Secondary Parameters:	N/A	
Time Required:	N/A	
Monitor Frequency	N/A	
MIL illumination:	N/A	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76392: [EthSwt] Extended production Errors set by Main Function using an own interface function is strange approach

  Problem description:

  In chapter 8.5 SWS_EthSwt_00122 states to call the API EthSwt_GetCountervaValues in the cyclic Main-function and report production errors for every single counter.
  this is really strange.
  1st: Why shall the main function call an external interface. It could simply fetch the data directly from HW
  2nd: Not every Counter of this API directly implies an error on the bus. It simply gives a statistic of events on the bus which than carefully interpreted are resulting in a diagnostic result of the bus. Setting an error on values greater zero will lead to a unnecessary errors counted here.
  3rd: calling this API cyclically with fetching about 15 or more bytes from HW on every main cycle, may lead to poor performance of the driver.

  Agreed solution:

  Take out SWS_EThSwt_00122 and all related extended production error. SWS_EthSwt_00137, SWS_EthSwt_00138, SWS_EthSwt_00139, SWS_EthSwt_00141,SWS_EthSwt_00142 to SWS_EthSwt_00152.

  Add following text in 7.2.5:
  There are no extended production errors.
  –Last change on issue 76392 comment 5–

  BW-C-Level:

Application	Specification	Bus
1	4	1

1.100 Specification Item SWS_EthSwt_00145

Trace References:

SRS_BSW_00385

Content:

Error Name:	ETHSWT_E_INDISCARD	
Short Description:	Discard of inbound packets	
Long Description:	This error occurs if inbound packets were chosen to be discarded even though no errors had been detected to prevent their being deliverable to a higher-layer protocol. One possible reason for discarding such a packet could be to free up buffer space. (see IETF RFC 2233 ifInDiscards)	
Recommended DTC:	N/A	
Detection Criteria:	Fail	If main function detects that the corresponding counter value is greater than zero, this error will be reported
	Pass	If no such error is reported.
Secondary Parameters:	N/A	
Time Required:	N/A	
Monitor Frequency	N/A	
MIL illumination:	N/A	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76392: [EthSwt] Extended production Errors set by Main Function using an own interface function is strange approach

  Problem description:

  In chapter 8.5 SWS_EthSwt_00122 states to call the API EthSwt_GetCountervaValues in the cyclic Main-function and report production errors for every single counter.
  this is really strange.
  1st: Why shall the main function call an external interface. It could simply fetch the data directly from HW
  2nd: Not every Counter of this API directly implies an error on the bus. It simply gives a statistic of events on the bus which than carefully interpreted are resulting in

a diagnostic result of the bus. Setting an error on values greater zero will lead to a unnecessary errors counted here.
3rd: calling this API cyclically with fetching about 15 or more bytes from HW on every main cycle, may lead to poor performance of the driver.

Agreed solution:

Take out SWS_EThSwt_00122 and all related extended production error. SWS_EthSwt_00137, SWS_EthSwt_00138, SWS_EthSwt_00139, SWS_EthSwt_00141,SWS_EthSwt_00142 to SWS_EthSwt_00152.

Add following text in 7.2.5:
There are no extended production errors.
–Last change on issue 76392 comment 5–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.101 Specification Item SWS_EthSwt_00146

Trace References:

SRS_BSW_00385

Content:

Error Name:	ETHSWT_E_INERROR	
Short Description:	Discard of inbound packets	
Long Description:	This error occurs if the total number of erroneous inbound packets is greater than zero	
Recommended DTC:	N/A	
Detection Criteria:	Fail	If main function detects that the corresponding counter value is greater than zero, this error will be reported
	Pass	If no such error is reported.
Secondary Parameters:	N/A	
Time Required:	N/A	
Monitor Frequency	N/A	
MIL illumination:	N/A	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76392: [EthSwt] Extended production Errors set by Main Function using an own interface function is strange approach

  Problem description:

  In chapter 8.5 SWS_EthSwt_00122 states to call the API EthSwt_GetCountervaValues in the cyclic Main-function and report production errors for every single counter.
  this is really strange.
  1st: Why shall the main function call an external interface. It could simply fetch the data directly from HW
  2nd: Not every Counter of this API directly implies an error on the bus. It simply gives a statistic of events on the bus which than carefully interpreted are resulting in a diagnostic result of the bus. Setting an error on values greater zero will lead to a unnecessary errors counted here.
  3rd: calling this API cyclically with fetching about 15 or more bytes from HW on every main cycle, may lead to poor performance of the driver.

  Agreed solution:

  Take out SWS_EThSwt_00122 and all related extended production error. SWS_EthSwt_00137, SWS_EthSwt_00138, SWS_EthSwt_00139, SWS_EthSwt_00141,SWS_EthSwt_00142 to SWS_EthSwt_00152.

  Add following text in 7.2.5:
  There are no extended production errors.
  –Last change on issue 76392 comment 5–

  BW-C-Level:

Application	Specification	Bus
1	4	1

1.102 Specification Item SWS_EthSwt_00147

Trace References:

SRS_BSW_00385

Content:

Error Name:	ETHSWT_E_OUTDISCARD
Short Description:	Discard of inbound packets

Long Description:	This error occurs if outbound packets were chosen to be discarded even though no errors had been detected to prevent their being transmitted. One possible reason for discarding such a packet could be to free up buffer space. (see IETF RFC 2233 ifOut Discards)	
Recommended DTC:	N/A	
Detection Criteria:	Fail	If main function detects that the corresponding counter value is greater than zero, this error will be reported
	Pass	If no such error is reported.
Secondary Parameters:	N/A	
Time Required:	N/A	
Monitor Frequency	N/A	
MIL illumination:	N/A	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76392: [EthSwt] Extended production Errors set by Main Function using an own interface function is strange approach

  Problem description:

  In chapter 8.5 SWS_EthSwt_00122 states to call the API EthSwt_GetCountervaValues in the cyclic Main-function and report production errors for every single counter.
  this is really strange.
  1st: Why shall the main function call an external interface. It could simply fetch the data directly from HW
  2nd: Not every Counter of this API directly implies an error on the bus. It simply gives a statistic of events on the bus which than carefully interpreted are resulting in a diagnostic result of the bus. Setting an error on values greater zero will lead to a unnecessary errors counted here.
  3rd: calling this API cyclically with fetching about 15 or more bytes from HW on every main cycle, may lead to poor performance of the driver.

  Agreed solution:

  Take out SWS_EThSwt_00122 and all related extended production error. SWS_EthSwt_00137, SWS_EthSwt_00138, SWS_EthSwt_00139, SWS_EthSwt_00141,SWS_EthSwt_00142 to SWS_EthSwt_00152.

  Add following text in 7.2.5:
  There are no extended production errors.
  –Last change on issue 76392 comment 5–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.103 Specification Item SWS_EthSwt_00148

Trace References:

SRS_BSW_00385

Content:

Error Name:	ETHSWT_E_OUTERROR	
Short Description:	Discard of inbound packets	
Long Description:	This error occurs if the total number of erroneous outbound packets is greater than zero	
Recommended DTC:	N/A	
Detection Criteria:	Fail	If main function detects that the corresponding counter value is greater than zero, this error will be reported
	Pass	If no such error is reported.
Secondary Parameters:	N/A	
Time Required:	N/A	
Monitor Frequency	N/A	
MIL illumination:	N/A	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76392: [EthSwt] Extended production Errors set by Main Function using an own interface function is strange approach

  Problem description:

  In chapter 8.5 SWS_EthSwt_00122 states to call the API EthSwt_GetCountervaValues in the cyclic Main-function and report production errors for every single counter.
  this is really strange.
  1st: Why shall the main function call an external interface. It could simply fetch the data directly from HW
  2nd: Not every Counter of this API directly implies an error on the bus. It simply gives a statistic of events on the bus which than carefully interpreted are resulting in

a diagnostic result of the bus. Setting an error on values greater zero will lead to a unnecessary errors counted here.
3rd: calling this API cyclically with fetching about 15 or more bytes from HW on every main cycle, may lead to poor performance of the driver.

Agreed solution:

Take out SWS_EThSwt_00122 and all related extended production error. SWS_EthSwt_00137, SWS_EthSwt_00138, SWS_EthSwt_00139, SWS_EthSwt_00141,SWS_EthSwt_00142 to SWS_EthSwt_00152.

Add following text in 7.2.5:
There are no extended production errors.
–Last change on issue 76392 comment 5–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.104 Specification Item SWS_EthSwt_00149

Trace References:

SRS_BSW_00385

Content:

Error Name:	ETHSWT_E_SINGLECOLLISION	
Short Description:	Number of packets with a single collision	
Long Description:	Single collision frames: A count of successfully transmitted frames on a particular interface for which transmission is inhibited by exactly one collision. (see IETF RFC1643 dot3StatsSingleCollisionFrames)	
Recommended DTC:	N/A	
Detection Criteria:	Fail	If main function detects that the corresponding counter value is greater than zero, this error will be reported
	Pass	If no such error is reported.
Secondary Parameters:	N/A	
Time Required:	N/A	
Monitor Frequency	N/A	
MIL illumination:	N/A	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76392: [EthSwt] Extended production Errors set by Main Function using an own interface function is strange approach

  Problem description:

  In chapter 8.5 SWS_EthSwt_00122 states to call the API EthSwt_GetCountervaValues in the cyclic Main-function and report production errors for every single counter.
  this is really strange.
  1st: Why shall the main function call an external interface. It could simply fetch the data directly from HW
  2nd: Not every Counter of this API directly implies an error on the bus. It simply gives a statistic of events on the bus which than carefully interpreted are resulting in a diagnostic result of the bus. Setting an error on values greater zero will lead to a unnecessary errors counted here.
  3rd: calling this API cyclically with fetching about 15 or more bytes from HW on every main cycle, may lead to poor performance of the driver.

  Agreed solution:

  Take out SWS_EThSwt_00122 and all related extended production error. SWS_EthSwt_00137, SWS_EthSwt_00138, SWS_EthSwt_00139, SWS_EthSwt_00141,SWS_EthSwt_00142 to SWS_EthSwt_00152.

  Add following text in 7.2.5:
  There are no extended production errors.
  –Last change on issue 76392 comment 5–

  BW-C-Level:

Application	Specification	Bus
1	4	1

1.105 Specification Item SWS_EthSwt_00150

Trace References:

SRS_BSW_00385

Content:

Error Name:	ETHSWT_E_MULTIPLECOLLISION
Short Description:	Number of packets with multiple collisions

Long Description:	Multiple collision frames: A count of successfully transmitted frames on a particular interface for which transmission is inhibited by more than one collision. (see IETF RFC1643 dot3StatsMultipleCollisionFrames)	
Recommended DTC:	N/A	
Detection Criteria:	Fail	If main function detects that the corresponding counter value is greater than zero, this error will be reported
	Pass	If no such error is reported.
Secondary Parameters:	N/A	
Time Required:	N/A	
Monitor Frequency	N/A	
MIL illumination:	N/A	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76392: [EthSwt] Extended production Errors set by Main Function using an own interface function is strange approach

  Problem description:

  In chapter 8.5 SWS_EthSwt_00122 states to call the API EthSwt_GetCountervaValues in the cyclic Main-function and report production errors for every single counter.
  this is really strange.
  1st: Why shall the main function call an external interface. It could simply fetch the data directly from HW
  2nd: Not every Counter of this API directly implies an error on the bus. It simply gives a statistic of events on the bus which than carefully interpreted are resulting in a diagnostic result of the bus. Setting an error on values greater zero will lead to a unnecessary errors counted here.
  3rd: calling this API cyclically with fetching about 15 or more bytes from HW on every main cycle, may lead to poor performance of the driver.

  Agreed solution:

  Take out SWS_EThSwt_00122 and all related extended production error. SWS_EthSwt_00137, SWS_EthSwt_00138, SWS_EthSwt_00139, SWS_EthSwt_00141,SWS_EthSwt_00142 to SWS_EthSwt_00152.

  Add following text in 7.2.5:
  There are no extended production errors.
  –Last change on issue 76392 comment 5–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.106 Specification Item SWS_EthSwt_00151

Trace References:

SRS_BSW_00385

Content:

Error Name:	ETHSWT_E_DEFFEREDTRANSMISSION	
Short Description:	Number of packets which are deffered	
Long Description:	Number of deferred transmission: A count of frames for which the first transmission attempt on a particular interface is delayed because the medium is busy. (see IETF RFC1643 dot3StatsDeferredTransmissions)	
Recommended DTC:	N/A	
Detection Criteria:	Fail	If main function detects that the corresponding counter value is greater than zero, this error will be reported
	Pass	If no such error is reported.
Secondary Parameters:	N/A	
Time Required:	N/A	
Monitor Frequency	N/A	
MIL illumination:	N/A	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76392: [EthSwt] Extended production Errors set by Main Function using an own interface function is strange approach

  Problem description:

  In chapter 8.5 SWS_EthSwt_00122 states to call the API EthSwt_GetCountervaValues in the cyclic Main-function and report production errors for every single counter.
  this is really strange.
  1st: Why shall the main function call an external interface. It could simply fetch the data directly from HW
  2nd: Not every Counter of this API directly implies an error on the bus. It simply

gives a statistic of events on the bus which than carefully interpreted are resulting in a diagnostic result of the bus. Setting an error on values greater zero will lead to a unnecessary errors counted here.
3rd: calling this API cyclically with fetching about 15 or more bytes from HW on every main cycle, may lead to poor performance of the driver.

Agreed solution:

Take out SWS_EThSwt_00122 and all related extended production error. SWS_EthSwt_00137, SWS_EthSwt_00138, SWS_EthSwt_00139, SWS_EthSwt_00141,SWS_EthSwt_00142 to SWS_EthSwt_00152.

Add following text in 7.2.5:
There are no extended production errors.
–Last change on issue 76392 comment 5–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.107 Specification Item SWS_EthSwt_00152

Trace References:

SRS_BSW_00385

Content:

Error Name:	ETHSWT_E_LATECOLLISION	
Short Description:	Number of packets with a late collision	
Long Description:	Number of late collisions: The number of times that a collision is detected on a particular interface later than 512 bit-times into the transmission of a packet. (see IETF RFC1643 dot3StatsLateCollisions)	
Recommended DTC:	N/A	
Detection Criteria:	Fail	If main function detects that the corresponding counter value is greater than zero, this error will be reported
	Pass	If no such error is reported.
Secondary Parameters:	N/A	
Time Required:	N/A	
Monitor Frequency	N/A	
MIL illumination:	N/A	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76392: [EthSwt] Extended production Errors set by Main Function using an own interface function is strange approach

  Problem description:

  In chapter 8.5 SWS_EthSwt_00122 states to call the API EthSwt_GetCountervaValues in the cyclic Main-function and report production errors for every single counter.
  this is really strange.
  1st: Why shall the main function call an external interface. It could simply fetch the data directly from HW
  2nd: Not every Counter of this API directly implies an error on the bus. It simply gives a statistic of events on the bus which than carefully interpreted are resulting in a diagnostic result of the bus. Setting an error on values greater zero will lead to a unnecessary errors counted here.
  3rd: calling this API cyclically with fetching about 15 or more bytes from HW on every main cycle, may lead to poor performance of the driver.

  Agreed solution:

  Take out SWS_EThSwt_00122 and all related extended production error. SWS_EthSwt_00137, SWS_EthSwt_00138, SWS_EthSwt_00139, SWS_EthSwt_00141,SWS_EthSwt_00142 to SWS_EthSwt_00152.

  Add following text in 7.2.5:
  There are no extended production errors.
  –Last change on issue 76392 comment 5–

  BW-C-Level:

Application	Specification	Bus
1	4	1

1.108 Specification Item SWS_EthSwt_00153

Trace References:

SRS_ETH_00118, SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369

Content:

The function EthSwt_GetBaudRate shall check whether the EthTrcv_GetBaudRate() API of the indexed transceiver driver is available by checking whether for this SwitchPortIdx the corresponding EthTrcv API is available. If this is not the case, the function shall return E_NOT_OK . If development error tracing is activated by EthSwtDevErrorDetect, EthSwt_GetBaudRate shall raise the development error ETHSWT_E_INV_API .

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77664: [EthSwt] Behaviour of certain APIs for ports without EthSwtPortTrcvRef (part II)

  Problem description:

  The following APIs have no description regarding the handling of ports without EthSwtPortTrcvRef:
  EthSwt_GetLinkState
  EthSwt_GetBaudRate
  EthSwt_GetDuplexMode

  The description should be adjusted and harmonized with related requirements.

  I would also add the following function:
  EthSwt_StartSwitchPortAutoNegotiation
  I rate this function not to be used for ports without a EthSwtPortTrcvRef. For the sake of completeness I would also reformulate some requirements.
  –Last change on issue 77664 comment 10–

  Agreed solution:

  === EthSwt ===
  — EthSwt_GetLinkState —
  ~[SWS_EthSwt_00038]
  The function EthSwt_GetLinkState shall read the current (link) state of the indexed switch port. If the indexed Ethernet port references an Ethernet transceiver, the link state shall be obtained by calling the function EthTrcv_GetLinkState() of the Ethernet Transceiver Driver. If the indexed Ethernet Switch port does not reference an Ethernet transceiver, the state shall be obtained from the MAC interface of the Switch port.

  — EthSwt_GetBaudRate —
  ~[SWS_EthSwt_00045]
  The function EthSwt_GetBaudRate() shall read the current baud rate of the indexed switch port. If the indexed Ethernet port reference an Ethernet transceiver, the baud rate shall be obtained by the function EthTrcv_GetBaudRate() of the Ethernet Transceiver Driver. If the indexed Ethernet Switch port does not reference an Ethernet transceiver, the baud rate shall be obtained from the MAC interface of the Switch port.

-[SWS_EthSwt_00153] as # 77349 introduces a general req to this.

— EthSwt_GetDuplexMode —
~[SWS_EthSwt_00052]
The function EthSwt_GetDuplexMode() shall read the current duplex mode of the indexed switch port. If the indexed Ethernet port reference an Ethernet transceiver, the duplex mode shall be obtained by calling the function EthTrcv_ GetDuplexMode() of the Ethernet Transceiver Driver. If the indexed Ethernet Switch port does not reference an Ethernet transceiver, the duplex mode shall be obtained from the MAC interface of the Switch port.
-[SWS_EthSwt_00155] as # 77349 introduces a general req to this.

— EthSwt_StartSwitchPortAutoNegotiation —
~[SWS_EthSwt_00032]
The function EthSwt_StartSwitchPortAutoNegotiation shall restart the automatic negotiation of the used transmission parameters of the referenced Ethernet transceiver driver by calling the function EthTrcv_StartAutoNegotiation().
-[SWS_EthSwt_00158] as # 77349 introduces a general req to this.
–Last change on issue 77664 comment 26–

BW-C-Level:

Application	**Specification**	**Bus**
1	4	1

1.109 Specification Item SWS_EthSwt_00155

Trace References:

SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369, SRS_ETH_00118

Content:

The function EthSwt_GetDuplexMode shall check whether the EthTrcv_GetDuplexMode() API of the indexed transceiver driver is available by checking whether for this SwitchPort Idx the corresponding EthTrcv API is available. If this is not the case, the function shall return E_NOT_OK and if development error tracing is activated by EthSwtDevErrorDetect the ETHSWT_E_INV_API shall be raised.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77664: [EthSwt] Behaviour of certain APIs for ports without EthSwtPortTrcvRef (part II)

Problem description:

The following APIs have no description regarding the handling of ports without EthSwtPortTrcvRef:
EthSwt_GetLinkState
EthSwt_GetBaudRate
EthSwt_GetDuplexMode

The description should be adjusted and harmonized with related requirements.

I would also add the following function:
EthSwt_StartSwitchPortAutoNegotiation
I rate this function not to be used for ports without a EthSwtPortTrcvRef. For the sake of completeness I would also reformulate some requirements.
–Last change on issue 77664 comment 10–

Agreed solution:

=== EthSwt ===
— EthSwt_GetLinkState —
~[SWS_EthSwt_00038]
The function EthSwt_GetLinkState shall read the current (link) state of the indexed switch port. If the indexed Ethernet port references an Ethernet transceiver, the link state shall be obtained by calling the function EthTrcv_GetLinkState() of the Ethernet Transceiver Driver. If the indexed Ethernet Switch port does not reference an Ethernet transceiver, the state shall be obtained from the MAC interface of the Switch port.

— EthSwt_GetBaudRate —
~[SWS_EthSwt_00045]
The function EthSwt_GetBaudRate() shall read the current baud rate of the indexed switch port. If the indexed Ethernet port reference an Ethernet transceiver, the baud rate shall be obtained by the function EthTrcv_GetBaudRate() of the Ethernet Transceiver Driver. If the indexed Ethernet Switch port does not reference an Ethernet transceiver, the baud rate shall be obtained from the MAC interface of the Switch port.
-[SWS_EthSwt_00153] as # 77349 introduces a general req to this.

— EthSwt_GetDuplexMode —
~[SWS_EthSwt_00052]
The function EthSwt_GetDuplexMode() shall read the current duplex mode of the indexed switch port. If the indexed Ethernet port reference an Ethernet transceiver, the duplex mode shall be obtained by calling the function EthTrcv_ GetDuplexMode()

of the Ethernet Transceiver Driver. If the indexed Ethernet Switch port does not reference an Ethernet transceiver, the duplex mode shall be obtained from the MAC interface of the Switch port.
-[SWS_EthSwt_00155] as # 77349 introduces a general req to this.

— EthSwt_StartSwitchPortAutoNegotiation —
~[SWS_EthSwt_00032]
The function EthSwt_StartSwitchPortAutoNegotiation shall restart the automatic negotiation of the used transmission parameters of the referenced Ethernet transceiver driver by calling the function EthTrcv_StartAutoNegotiation().
-[SWS_EthSwt_00158] as # 77349 introduces a general req to this.
–Last change on issue 77664 comment 26–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.110 Specification Item SWS_EthSwt_00158

Trace References:

SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369, SRS_ETH_00087

Content:

The function EthSwt_StartSwitchPortAutonegotiation shall check whether the EthTrcv_StartAutoNegotiation() API of the indexed transceiver driver is available by checking whether for this SwitchPortIdx the corresponding EthTrcv API is available. If this is not the case, the function shall return E_NOT_OK and if development error tracing is activated by EthSwtDevErrorDetect the ETHSWT_E_INV_API shall be raised.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77664: [EthSwt] Behaviour of certain APIs for ports without EthSwtPortTrcvRef (part II)

  Problem description:

  The following APIs have no description regarding the handling of ports without EthSwtPortTrcvRef:
  EthSwt_GetLinkState
  EthSwt_GetBaudRate
  EthSwt_GetDuplexMode

The description should be adjusted and harmonized with related requirements.

I would also add the following function:
EthSwt_StartSwitchPortAutoNegotiation
I rate this function not to be used for ports without a EthSwtPortTrcvRef. For the sake of completeness I would also reformulate some requirements.
–Last change on issue 77664 comment 10–

Agreed solution:

=== EthSwt ===
— EthSwt_GetLinkState —
~[SWS_EthSwt_00038]
The function EthSwt_GetLinkState shall read the current (link) state of the indexed switch port. If the indexed Ethernet port references an Ethernet transceiver, the link state shall be obtained by calling the function EthTrcv_GetLinkState() of the Ethernet Transceiver Driver. If the indexed Ethernet Switch port does not reference an Ethernet transceiver, the state shall be obtained from the MAC interface of the Switch port.

— EthSwt_GetBaudRate —
~[SWS_EthSwt_00045]
The function EthSwt_GetBaudRate() shall read the current baud rate of the indexed switch port. If the indexed Ethernet port reference an Ethernet transceiver, the baud rate shall be obtained by the function EthTrcv_GetBaudRate() of the Ethernet Transceiver Driver. If the indexed Ethernet Switch port does not reference an Ethernet transceiver, the baud rate shall be obtained from the MAC interface of the Switch port.
-[SWS_EthSwt_00153] as # 77349 introduces a general req to this.

— EthSwt_GetDuplexMode —
~[SWS_EthSwt_00052]
The function EthSwt_GetDuplexMode() shall read the current duplex mode of the indexed switch port. If the indexed Ethernet port reference an Ethernet transceiver, the duplex mode shall be obtained by calling the function EthTrcv_ GetDuplexMode() of the Ethernet Transceiver Driver. If the indexed Ethernet Switch port does not reference an Ethernet transceiver, the duplex mode shall be obtained from the MAC interface of the Switch port.
-[SWS_EthSwt_00155] as # 77349 introduces a general req to this.

— EthSwt_StartSwitchPortAutoNegotiation —
~[SWS_EthSwt_00032]
The function EthSwt_StartSwitchPortAutoNegotiation shall restart the automatic ne-

gotiation of the used transmission parameters of the referenced Ethernet transceiver driver by calling the function EthTrcv_StartAutoNegotiation().
-[SWS_EthSwt_00158] as # 77349 introduces a general req to this.
–Last change on issue 77664 comment 26–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.111 Specification Item SWS_EthSwt_00165

Trace References:

SRS_ETH_00086 BSW_00395

Content:

Name:	EthSwt_ConfigTypeEthSwt_ConfigType		
Type:	Structure		
Element:	void	implementation specificEth Swt_Config Type.implementation specific	–
Description:	Implementation specific structure of the post build configuration.		

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76248: [EthSwt]Remove references to non existing requirements (SRS, SWS)

  Problem description:

  Some review findings brought up references to non existing requirements. Please check and remove the references and if needed reformulate the requirements.

  Agreed solution:

  SWS_EthSwt_00123 –> SRS_BSW_00406
  SWS_EthSwt_00165 -> SRS_BSW_00395
  SWS_EthSwt_00227 -> SRS_ETH_00087
  SWS_EthSwt_91017, SWS_EthSwt_91020, –> SRS_ETH_00123
  SWS_EthSwt_00006 –> SRS_BSW_00101
  SWS_EthSwt_00010 left blank as SRS doe not yet provide a requirement here but maybe with RFC # 75170 AUTOSAR will introduce an SRS-item after all.
  SWS_EthSwt_00376, SWS_EthSwt_00374, SWS_EthSwt_00375 –> SRS-Missing! same as for SWS_EThSwt_00010
  SWS_EthSwt_00018, SRS_EthSwt_00025 –> SRS_ETH_00118

SWS_EthSwt_00031 –> SRS_ETH_00087
SWS_EthSwt_00037 –> SRS_ETH_00119
SWS_EthSwt_00044, SWS_EthSwt_00051 –> SRS_ETH_00118
SWS_EthSwt_00060 –> SRS_ETH_00087
SWS_EthSwt_00111 –> erase SRS_ETH_00086
SWS_EthSwt00079 –>SRS_ETH_00119
SWS_EthSwt_00206, SWS_EthSwt_00211, SWS_EthSwt_00216, SWS_EthSwt_00221 –>SRS_Eth_00120
SWS_EthSwt_00172 –>SRS_ETH_00121, SRS_ETH_00114
SWS_EthSwt_00086, SWS_EthSwt_00091, SWS_EthSwt_00182, SWS_EthSwt_00125 –> SRS_ETH_00087, SRS_ETH_00122
SWS_EthSwt_00187 –> SRS_ETH_00087
SWS_EthSwt_00058 –> SRS_BSW_00171
SWS_EthSwt_91012, SWS_EthSwt_91013 –> SRS_ETH_00126
SWS_EthSwt_91014, SWS_EthSwt_91015, SWS_EthSwt_91016, SWS_EthSwt_91018, SWS_EthSwt_91019, SWS_EthSwt_91021, SWS_EthSwt_91022, SWS_EthSwt_91022, SWS_EthSwt_91023, SWS_EthSwt_91024, SWS_EthSwt_91025, SWS_EthSwt_91026, SWS_EthSwt_91027 –> SRS_Eth_00123
SWS_EthSwt_00117, SWS_EthSwt_00203 –> SRS_ETH_00119, SRS_ETH_00087
SWS_EthSwt_00193 –> SRS_ETH_00122, SRS_ETH_00087
SWS_EthSwt_00114 –> SRS_BSW_00433
SWS_EthSwt_00098 –> SRS_Eth_00122, SRS_ETH_00118, SRS_ETH_00119, SRS_ETH_00120, SRS_ETH_00087, SRS_ETH_00125, SRS_BSW_00375
–Last change on issue 76248 comment 13–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.112 Specification Item SWS_EthSwt_00166

Trace References:

SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchPortIdx is not valid,EthSwt_SetSwitchPortMode shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

  For Error Detection following general rules apply:
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
  [SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with

input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.113 Specification Item SWS_EthSwt_00167

Trace References:

SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchPortIdx is not valid,EthSwt_GetSwitchPortMode shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

[SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

In the above requirement -
1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.

[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.114 Specification Item SWS_EthSwt_00168

Trace References:

SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchPortIdx is not valid,EthSwt_StartSwitchPortAutoNegotiation shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -

1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevEr-

rorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.115 Specification Item SWS_EthSwt_00169

Trace References:

SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchPortIdx is not valid, EthSwt_GetLinkState shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.116 Specification Item SWS_EthSwt_00170

Trace References:

SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchPortIdx is not valid, EthSwt_GetBaudRate shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.

Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.117 Specification Item SWS_EthSwt_00171

Trace References:

SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchPortIdx is not valid,EthSwt_GetDuplexMode shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in

chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.118 Specification Item SWS_EthSwt_00172

Trace References:

SRS_ETH_00086 00121, SRS_ETH_00114

Content:

Service name:	EthSwt_EnableVlanEthSwt_EnableVlan	
Syntax:	Std_ReturnType EthSwt_EnableVlan(uint8 SwitchIdx, uint8 SwitchPortIdx, uint16 VlanId, boolean Enable)	
Service ID[hex]:	0x12	
Sync/Async:	Synchronous /Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	SwitchIdxEthSwt_EnableVlan.SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	SwitchPortIdxEthSwt_EnableVlan.SwitchPortIdx	Index of the port at the addressed switch
	VlanIdEthSwt_EnableVlan.VlanId	VLAN-ID to a preconfigured configuration on the given ingress port
	EnableEthSwt_EnableVlan.Enable	1 = VLAN-configuration enabled 0 = VLAN-configuration disabled (frames with given VLAN-ID will be dropped)
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: success E_NOT_OK: buffer level could not be obtained
Description:	Enables or disables a pre-configured VLAN at a certain port of a switch.	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76248: [EthSwt]Remove references to non existing requirements (SRS, SWS)

  Problem description:

  Some review findings brought up references to non existing requirements. Please check and remove the references and if needed reformulate the requirements.

  Agreed solution:

SWS_EthSwt_00123 –> SRS_BSW_00406
SWS_EthSwt_00165 -> SRS_BSW_00395
SWS_EthSwt_00227 -> SRS_ETH_00087
SWS_EthSwt_91017, SWS_EthSwt_91020, –> SRS_ETH_00123
SWS_EthSwt_00006 –> SRS_BSW_00101
SWS_EthSwt_00010 left blank as SRS doe not yet provide a requirement here but maybe with RFC # 75170 AUTOSAR will introduce an SRS-item after all.
SWS_EthSwt_00376, SWS_EthSwt_00374, SWS_EthSwt_00375 –> SRS-Missing! same as for SWS_EThSwt_00010
SWS_EthSwt_00018, SRS_EthSwt_00025 –> SRS_ETH_00118
SWS_EthSwt_00031 –> SRS_ETH_00087
SWS_EthSwt_00037 –> SRS_ETH_00119
SWS_EthSwt_00044, SWS_EthSwt_00051 –> SRS_ETH_00118
SWS_EthSwt_00060 –> SRS_ETH_00087
SWS_EthSwt_00111 –> erase SRS_ETH_00086
SWS_EthSwt00079 –>SRS_ETH_00119
SWS_EthSwt_00206, SWS_EthSwt_00211, SWS_EthSwt_00216, SWS_EthSwt_00221 –>SRS_Eth_00120
SWS_EthSwt_00172 –>SRS_ETH_00121, SRS_ETH_00114
SWS_EthSwt_00086, SWS_EthSwt_00091, SWS_EthSwt_00182, SWS_EthSwt_00125 –> SRS_ETH_00087, SRS_ETH_00122
SWS_EthSwt_00187 –> SRS_ETH_00087
SWS_EthSwt_00058 –> SRS_BSW_00171
SWS_EthSwt_91012, SWS_EthSwt_91013 –> SRS_ETH_00126
SWS_EthSwt_91014, SWS_EthSwt_91015, SWS_EthSwt_91016, SWS_EthSwt_91018, SWS_EthSwt_91019, SWS_EthSwt_91021, SWS_EthSwt_91022, SWS_EthSwt_91022, SWS_EthSwt_91023, SWS_EthSwt_91024, SWS_EthSwt_91025, SWS_EthSwt_91026, SWS_EthSwt_91027 –> SRS_Eth_00123
SWS_EthSwt_00117, SWS_EthSwt_00203 –> SRS_ETH_00119, SRS_ETH_00087
SWS_EthSwt_00193 –> SRS_ETH_00122, SRS_ETH_00087
SWS_EthSwt_00114 –> SRS_BSW_00433
SWS_EthSwt_00098 –> SRS_Eth_00122, SRS_ETH_00118, SRS_ETH_00119, SRS_ETH_00120, SRS_ETH_00087, SRS_ETH_00125, SRS_BSW_00375
–Last change on issue 76248 comment 13–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.119 Specification Item SWS_EthSwt_00174

Trace References:

SRS_BSW_00406

Content:

If development error detection is enabled: the function EthSwt_EnableVlan shall check that the service EthSwt_SwitchInit was previously called. If the check fails, the function shall raise the development error ETHSWT_E_NOT_INITIALIZED and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.120 Specification Item SWS_EthSwt_00175

Trace References:

SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchIdx is not valid,EthSwt_EnableVlan shall raise the development error ETHSWT_E_INV_SWITCH_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

  For Error Detection following general rules apply:
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with in-

put parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.121 Specification Item SWS_EthSwt_00176

Trace References:

SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchPortIdx is not valid,EthSwt_EnableVlan shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

  For Error Detection following general rules apply:
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with

input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.122 Specification Item SWS_EthSwt_00182

Trace References:

SRS_ETH_0008600087, SRS_ETH_00087 00122

Content:

Service name:	EthSwt_SetMacLearningModeEthSwt_SetMacLearningMode
Syntax:	Std_ReturnType EthSwt_SetMacLearningMode(uint8 SwitchIdx, uint8 SwitchPortIdx, EthSwt_MacLearningType MacLearningMode)

Service ID[hex]:	0x15	
Sync/Async:	Synchronous /Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	SwitchIdxEthSwt_SetMacLearning Mode.SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	SwitchPortIdxEthSwt_SetMacLearning Mode.SwitchPortIdx	Index of the port at the addressed switch
	MacLearningModeEthSwt_SetMac LearningMode.MacLearningMode	Defines whether MAC addresses shall be learned and if they shall be learned in software or hardware.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: success E_NOT_OK: configuration could be persistently reset
Description:	Sets the MAC learning mode in one of the tree modes: 1.) HW learning enabled, 2.) Hardware learning disabled, 3.) Software learning enabled. Note: This feature is hardware dependent, i.e. the switch hardware needs to support the different learning modes.	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76248: [EthSwt]Remove references to non existing requirements (SRS, SWS)

  Problem description:

  Some review findings brought up references to non existing requirements. Please check and remove the references and if needed reformulate the requirements.

  Agreed solution:

  SWS_EthSwt_00123 –> SRS_BSW_00406
  SWS_EthSwt_00165 -> SRS_BSW_00395
  SWS_EthSwt_00227 -> SRS_ETH_00087
  SWS_EthSwt_91017, SWS_EthSwt_91020, –> SRS_ETH_00123
  SWS_EthSwt_00006 –> SRS_BSW_00101
  SWS_EthSwt_00010 left blank as SRS doe not yet provide a requirement here but maybe with RFC # 75170 AUTOSAR will introduce an SRS-item after all.
  SWS_EthSwt_00376, SWS_EthSwt_00374, SWS_EthSwt_00375 –> SRS-Missing! same as for SWS_EThSwt_00010
  SWS_EthSwt_00018, SRS_EthSwt_00025 –> SRS_ETH_00118
  SWS_EthSwt_00031 –> SRS_ETH_00087
  SWS_EthSwt_00037 –> SRS_ETH_00119
  SWS_EthSwt_00044, SWS_EthSwt_00051 –> SRS_ETH_00118
  SWS_EthSwt_00060 –> SRS_ETH_00087
  SWS_EthSwt_00111 –> erase SRS_ETH_00086
  SWS_EthSwt00079 –>SRS_ETH_00119

SWS_EthSwt_00206, SWS_EthSwt_00211, SWS_EthSwt_00216, SWS_EthSwt_00221 –>SRS_Eth_00120
SWS_EthSwt_00172 –>SRS_ETH_00121, SRS_ETH_00114
SWS_EthSwt_00086, SWS_EthSwt_00091, SWS_EthSwt_00182, SWS_EthSwt_00125 –> SRS_ETH_00087, SRS_ETH_00122
SWS_EthSwt_00187 –> SRS_ETH_00087
SWS_EthSwt_00058 –> SRS_BSW_00171
SWS_EthSwt_91012, SWS_EthSwt_91013 –> SRS_ETH_00126
SWS_EthSwt_91014, SWS_EthSwt_91015, SWS_EthSwt_91016, SWS_EthSwt_91018, SWS_EthSwt_91019, SWS_EthSwt_91021, SWS_EthSwt_91022, SWS_EthSwt_91022, SWS_EthSwt_91023, SWS_EthSwt_91024, SWS_EthSwt_91025, SWS_EthSwt_91026, SWS_EthSwt_91027 –> SRS_Eth_00123
SWS_EthSwt_00117, SWS_EthSwt_00203 –> SRS_ETH_00119, SRS_ETH_00087
SWS_EthSwt_00193 –> SRS_ETH_00122, SRS_ETH_00087
SWS_EthSwt_00114 –> SRS_BSW_00433
SWS_EthSwt_00098 –> SRS_Eth_00122, SRS_ETH_00118, SRS_ETH_00119, SRS_ETH_00120, SRS_ETH_00087, SRS_ETH_00125, SRS_BSW_00375
–Last change on issue 76248 comment 13–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.123 Specification Item SWS_EthSwt_00184

Trace References:

SRS_BSW_00406

Content:

If development error detection is enabled: the function EthSwt_SetMacLearningMode shall check that the service EthSwt_SwitchInit was previously called. If the check fails, the function shall raise the development error ETHSWT_E_NOT_INITIALIZED and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

[SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

In the above requirement -
1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.

[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.124 Specification Item SWS_EthSwt_00185

Trace References:

SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchIdx is not valid,EthSwt_SetMacLearningMode shall raise the development error ETHSWT_E_INV_SWITCH_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -

1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevEr-

rorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.125 Specification Item SWS_EthSwt_00187

Trace References:

SRS_ETH_00086, SRS_ETH_00087

Content:

Service name:	EthSwt_GetMacLearningModeEthSwt_GetMacLearningMode	
Syntax:	Std_ReturnType EthSwt_GetMacLearningMode(uint8 SwitchIdx, uint8 SwitchPortIdx, EthSwt_MacLearningType* MacLearningMode)	
Service ID[hex]:	0x16	
Sync/Async:	Synchronous /Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	SwitchIdxEthSwt_GetMacLearning Mode.SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	SwitchPortIdxEthSwt_GetMacLearning Mode.SwitchPortIdx	Index of the port at the addressed switch
Parameters (inout):	None	
Parameters (out):	MacLearningModeEthSwt_GetMac LearningMode.MacLearningMode	Defines whether MAC addresses shall be learned and if they shall be learned in software or hardware.
Return value:	Std_ReturnType	E_OK: success E_NOT_OK: configuration could be persistently reset
Description:	Returns the MAC learning mode, i.e. 1.) HW learning enabled, 2.) Hardware learning disabled, 3.) Software learning enabled. Note: This feature is hardware dependent, i.e. the switch hardware needs to support the different learning modes	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76248: [EthSwt]Remove references to non existing requirements (SRS, SWS)

  Problem description:

  Some review findings brought up references to non existing requirements. Please check and remove the references and if needed reformulate the requirements.

  Agreed solution:

  SWS_EthSwt_00123 –> SRS_BSW_00406
  SWS_EthSwt_00165 -> SRS_BSW_00395
  SWS_EthSwt_00227 -> SRS_ETH_00087
  SWS_EthSwt_91017, SWS_EthSwt_91020, –> SRS_ETH_00123
  SWS_EthSwt_00006 –> SRS_BSW_00101
  SWS_EthSwt_00010 left blank as SRS doe not yet provide a requirement here but maybe with RFC # 75170 AUTOSAR will introduce an SRS-item after all.
  SWS_EthSwt_00376, SWS_EthSwt_00374, SWS_EthSwt_00375 –> SRS-Missing! same as for SWS_EThSwt_00010
  SWS_EthSwt_00018, SRS_EthSwt_00025 –> SRS_ETH_00118
  SWS_EthSwt_00031 –> SRS_ETH_00087
  SWS_EthSwt_00037 –> SRS_ETH_00119
  SWS_EthSwt_00044, SWS_EthSwt_00051 –> SRS_ETH_00118
  SWS_EthSwt_00060 –> SRS_ETH_00087
  SWS_EthSwt_00111 –> erase SRS_ETH_00086
  SWS_EthSwt00079 –>SRS_ETH_00119
  SWS_EthSwt_00206, SWS_EthSwt_00211, SWS_EthSwt_00216, SWS_EthSwt_00221 –>SRS_Eth_00120
  SWS_EthSwt_00172 –>SRS_ETH_00121, SRS_ETH_00114
  SWS_EthSwt_00086, SWS_EthSwt_00091, SWS_EthSwt_00182, SWS_EthSwt_00125 –> SRS_ETH_00087, SRS_ETH_00122
  SWS_EthSwt_00187 –> SRS_ETH_00087
  SWS_EthSwt_00058 –> SRS_BSW_00171
  SWS_EthSwt_91012, SWS_EthSwt_91013 –> SRS_ETH_00126
  SWS_EthSwt_91014, SWS_EthSwt_91015, SWS_EthSwt_91016, SWS_EthSwt_91018, SWS_EthSwt_91019, SWS_EthSwt_91021, SWS_EthSwt_91022, SWS_EthSwt_91022, SWS_EthSwt_91023, SWS_EthSwt_91024, SWS_EthSwt_91025, SWS_EthSwt_91026, SWS_EthSwt_91027 –> SRS_Eth_00123
  SWS_EthSwt_00117, SWS_EthSwt_00203 –> SRS_ETH_00119, SRS_ETH_00087
  SWS_EthSwt_00193 –> SRS_ETH_00122, SRS_ETH_00087
  SWS_EthSwt_00114 –> SRS_BSW_00433
  SWS_EthSwt_00098 –> SRS_Eth_00122, SRS_ETH_00118, SRS_ETH_00119,

SRS_ETH_00120, SRS_ETH_00087, SRS_ETH_00125, SRS_BSW_00375
–Last change on issue 76248 comment 13–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.126 Specification Item SWS_EthSwt_00189

Trace References:

SRS_BSW_00406

Content:

If development error detection is disabled: the function EthSwt_GetMacLearningMode shall check that the service EthSwt_SwitchInit was previously called. If the check fails, the function shall raise the development error ETHSWT_E_NOT_INITIALIZED and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.127 Specification Item SWS_EthSwt_00190

Trace References:

SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchIdx is not valid,EthSwt_GetMacLearningMode shall raise the development error ETHSWT_E_INV_SWITCH_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001

rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.128 Specification Item SWS_EthSwt_00193

Trace References:

SRS_ETH_0008600122, SRS_ETH_00087

Content:

Service name:	<User>_PersistentConfigurationResult<User>_PersistentConfigurationResult	
Syntax:	void <User>_PersistentConfigurationResult(NvM_RequestResultType JobResult)	
Service ID[hex]:	0x1b	
Sync/Async:	Synchronous /Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	JobResult<User>_Persistent ConfigurationResult.JobResult	Covers the job result of the previous processed single block job.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	None	
Description:	Shall be called by the EthSwt_NvmSingleBlockCallback	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76248: [EthSwt]Remove references to non existing requirements (SRS, SWS)

  Problem description:

  Some review findings brought up references to non existing requirements. Please check and remove the references and if needed reformulate the requirements.

  Agreed solution:

  SWS_EthSwt_00123 –> SRS_BSW_00406
  SWS_EthSwt_00165 -> SRS_BSW_00395
  SWS_EthSwt_00227 -> SRS_ETH_00087
  SWS_EthSwt_91017, SWS_EthSwt_91020, –> SRS_ETH_00123
  SWS_EthSwt_00006 –> SRS_BSW_00101
  SWS_EthSwt_00010 left blank as SRS doe not yet provide a requirement here but maybe with RFC # 75170 AUTOSAR will introduce an SRS-item after all.
  SWS_EthSwt_00376, SWS_EthSwt_00374, SWS_EthSwt_00375 –> SRS-Missing! same as for SWS_EThSwt_00010
  SWS_EthSwt_00018, SRS_EthSwt_00025 –> SRS_ETH_00118
  SWS_EthSwt_00031 –> SRS_ETH_00087
  SWS_EthSwt_00037 –> SRS_ETH_00119
  SWS_EthSwt_00044, SWS_EthSwt_00051 –> SRS_ETH_00118

SWS_EthSwt_00060 –> SRS_ETH_00087
SWS_EthSwt_00111 –> erase SRS_ETH_00086
SWS_EthSwt00079 –>SRS_ETH_00119
SWS_EthSwt_00206, SWS_EthSwt_00211, SWS_EthSwt_00216, SWS_EthSwt_00221 –>SRS_Eth_00120
SWS_EthSwt_00172 –>SRS_ETH_00121, SRS_ETH_00114
SWS_EthSwt_00086, SWS_EthSwt_00091, SWS_EthSwt_00182, SWS_EthSwt_00125 –> SRS_ETH_00087, SRS_ETH_00122
SWS_EthSwt_00187 –> SRS_ETH_00087
SWS_EthSwt_00058 –> SRS_BSW_00171
SWS_EthSwt_91012, SWS_EthSwt_91013 –> SRS_ETH_00126
SWS_EthSwt_91014, SWS_EthSwt_91015, SWS_EthSwt_91016, SWS_EthSwt_91018, SWS_EthSwt_91019, SWS_EthSwt_91021, SWS_EthSwt_91022, SWS_EthSwt_91022, SWS_EthSwt_91023, SWS_EthSwt_91024, SWS_EthSwt_91025, SWS_EthSwt_91026, SWS_EthSwt_91027 –> SRS_Eth_00123
SWS_EthSwt_00117, SWS_EthSwt_00203 –> SRS_ETH_00119, SRS_ETH_00087
SWS_EthSwt_00193 –> SRS_ETH_00122, SRS_ETH_00087
SWS_EthSwt_00114 –> SRS_BSW_00433
SWS_EthSwt_00098 –> SRS_Eth_00122, SRS_ETH_00118, SRS_ETH_00119, SRS_ETH_00120, SRS_ETH_00087, SRS_ETH_00125, SRS_BSW_00375
–Last change on issue 76248 comment 13–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.129 Specification Item SWS_EthSwt_00200

Trace References:

SRS_BSW_00406

Content:

If development error detection is enabled: the function EthSwt_GetRxStats shall check that the service EthSwt_SwitchInit was previously called. If the check fails, the function shall raise the development error ETHSWT_E_NOT_INITIALIZED and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

  For Error Detection following general rules apply:
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
  [SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.

[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.130 Specification Item SWS_EthSwt_00201

Trace References:

SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchIdx is not valid,EthSwt_GetRxStats shall raise the development error ETHSWT_E_INV_SWITCH_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return

E_NOT_OK if the driver is not yet initialized. ()

In the above requirement -
1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being

valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.131 Specification Item SWS_EthSwt_00203

Trace References:

SRS_ETH_0008600119, SRS_ETH_00087

Content:

Service name:	<User>_EthSwtLinkUp<UserCallout>_<EthSwtLinkUpCallout>	
Syntax:	void <User>_EthSwtLinkUpCallout>(uint8* uint8 SwitchIdxPtr, uint8* uint8 PortIdx Ptr)	
Service ID[hex]:	0x1a	
Sync/Async:	Synchronous /Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	None	
	Parameters (inout):	
	Parameters (out):	SwitchIdxPtr<User>_EthSwtLinkUp Callout>.SwitchIdx Ptr
PortIdxPtr<User>_EthSwtLinkUp Callout>.PortIdx Ptr		Pointer to the port index Index of the port at the addressed switch
Parameters (inout):	None	
Parameters (out):	None	

Return value:	None
Description:	Shall be Is called, if a link up occurred. In case the hardware is not able to signal a link up via interrupt, this function needs to poll the link status in its main function. which is configured goes up

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76248: [EthSwt]Remove references to non existing requirements (SRS, SWS)

  Problem description:

  Some review findings brought up references to non existing requirements. Please check and remove the references and if needed reformulate the requirements.

  Agreed solution:

  SWS_EthSwt_00123 –> SRS_BSW_00406
  SWS_EthSwt_00165 -> SRS_BSW_00395
  SWS_EthSwt_00227 -> SRS_ETH_00087
  SWS_EthSwt_91017, SWS_EthSwt_91020, –> SRS_ETH_00123
  SWS_EthSwt_00006 –> SRS_BSW_00101
  SWS_EthSwt_00010 left blank as SRS doe not yet provide a requirement here but maybe with RFC # 75170 AUTOSAR will introduce an SRS-item after all.
  SWS_EthSwt_00376, SWS_EthSwt_00374, SWS_EthSwt_00375 –> SRS-Missing! same as for SWS_EThSwt_00010
  SWS_EthSwt_00018, SRS_EthSwt_00025 –> SRS_ETH_00118
  SWS_EthSwt_00031 –> SRS_ETH_00087
  SWS_EthSwt_00037 –> SRS_ETH_00119
  SWS_EthSwt_00044, SWS_EthSwt_00051 –> SRS_ETH_00118
  SWS_EthSwt_00060 –> SRS_ETH_00087
  SWS_EthSwt_00111 –> erase SRS_ETH_00086
  SWS_EthSwt00079 –>SRS_ETH_00119
  SWS_EthSwt_00206, SWS_EthSwt_00211, SWS_EthSwt_00216, SWS_EthSwt_00221 –>SRS_Eth_00120
  SWS_EthSwt_00172 –>SRS_ETH_00121, SRS_ETH_00114
  SWS_EthSwt_00086, SWS_EthSwt_00091, SWS_EthSwt_00182, SWS_EthSwt_00125 –> SRS_ETH_00087, SRS_ETH_00122
  SWS_EthSwt_00187 –> SRS_ETH_00087
  SWS_EthSwt_00058 –> SRS_BSW_00171
  SWS_EthSwt_91012, SWS_EthSwt_91013 –> SRS_ETH_00126
  SWS_EthSwt_91014, SWS_EthSwt_91015, SWS_EthSwt_91016, SWS_EthSwt_91018, SWS_EthSwt_91019, SWS_EthSwt_91021, SWS_EthSwt_91022, SWS_EthSwt_91022, SWS_EthSwt_91023, SWS_EthSwt_91024, SWS_EthSwt_91025, SWS_EthSwt_91026,

SWS_EthSwt_91027 –> SRS_Eth_00123
SWS_EthSwt_00117, SWS_EthSwt_00203 –> SRS_ETH_00119, SRS_ETH_00087
SWS_EthSwt_00193 –> SRS_ETH_00122, SRS_ETH_00087
SWS_EthSwt_00114 –> SRS_BSW_00433
SWS_EthSwt_00098 –> SRS_Eth_00122, SRS_ETH_00118, SRS_ETH_00119, SRS_ETH_00120, SRS_ETH_00087, SRS_ETH_00125, SRS_BSW_00375
–Last change on issue 76248 comment 13–

BW-C-Level:

Application	Specification	Bus
1	1	1

- RfC #76531: [EthSwt] Syntax for callback functions <user>_LinkUp and <user>_LinkDown might be wrong

  Problem description:

  The callback function for <User>_LinkUp and <User>_LinkDown to inform the user about a Link state change might be wrong
  Currently the parameter of these both functions are defined as Pointer to uint8
  This makes no sense for me to use here pointers and not a uint 8 variable.

  Agreed solution:

  Chapter 8.6.3 Configurable Interfaces
  add following intro:
  In this chapter all interfaces are listed where the target function could be configured. The names of these kind of interfaces are not fixed because they are configurable.

  Add Subchapter with interface
  move SWS_EthSwt_00117 here and change to:
  Service name: <EthSwtLinkDownCallout>
  syntax: void <EthSwtLinkDownCallout>(uint8 SwitchIdx, uint8 PortIdx)
  Service ID: 0x19
  Sync/Async:Synchronous
  Reentrancy: Non Reentrant
  Parameters (in): SwitchIdx: Index of the switch within the context of the Ethernet Switch Driver
  PortIdx: Index of the port at the addressed switch
  Parameters (in/out): none
  Parameters (out): none
  Return value: none
  Description: Is called, if a link which is configured goes down.

~[SWS_EthSwt_00118]The function <EthSwtLinkDownCallout> shall be called if a link, which is configured, goes down. The function provides the Switch index and the Port index, such that the port which went down can be identified.

-[SWS_EthSwt_00119]

add second subchapter:

move SWS_EthSwt_00203 here and change to:
Service name: <EthSwtLinkUpCallout>
syntax: void <EthSwtLinkUpCallout>(uint8 SwitchIdx, uint8 PortIdx)
Service ID: 0x1a
Sync/Async:Synchronous
Reentrancy: Non Reentrant
Parameters (in): SwitchIdx: Index of the switch within the context of the Ethernet Switch Driver
PortIdx: Index of the port at the addressed switch
Parameters (in/out): none
Parameters (out): none
Return value: none
Description: Is called, if a link which is configured goes up

~[SWS_EthSwt_00204]The function <EthSwtLinkUpCallout> shall be called if a link which is configured goes up. The function provides the Switch index and the Port index, such that the port which went up can be identified.

-[SWS_EthSwt_00205] This is no clear as it is a configurable interface now

~ECUC_EthSwt_00048 set EthSwtLinkDownUser to obsolete
+ add optional function name parameter with same attributes named EthSwtLinkDownCallout
change description to Defines the function name for <EthSwtLinkDownCallout>.

~ECUC_EthSwt_00068 set EthSwtLinkUpUser to obsolete
+add optional function name parameter with same attributes named EthSwtLinkUpCallout
change description to Defines the function name for <EthSwtLinkUpCallout>.

set ECUC_EthSwt_00047 EthSwtPortEnableLinkDownCallback to obsolete.

Header File name is defined by ECUC_EthSwt_00064 EthSwtPublicCddHead-

erFile.
–Last change on issue 76531 comment 54–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.132 Specification Item SWS_EthSwt_00204

Trace References:

SRS_ETH_00119, SRS_ETH_00087

Content:

The function <User>_EthSwtLinkUpCallout> shall be called if a linkcomes , which is configured, goes up. The function returns provides the Switch index and the Port index, such that the port which went down up can be identified.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76531: [EthSwt] Syntax for callback functions <user>_LinkUp and <user>_LinkDown might be wrong

  Problem description:

  The callback function for <User>_LinkUp and <User>_LinkDown to inform the user about a Link state change might be wrong
  Currently the parameter of these both functions are defined as Pointer to uint8
  This makes no sense for me to use here pointers and not a uint 8 variable.

  Agreed solution:

  Chapter 8.6.3 Configurable Interfaces
  add following intro:
  In this chapter all interfaces are listed where the target function could be configured.
  The names of these kind of interfaces are not fixed because they are configurable.

  Add Subchapter with interface
  move SWS_EthSwt_00117 here and change to:
  Service name: <EthSwtLinkDownCallout>
  syntax: void <EthSwtLinkDownCallout>(uint8 SwitchIdx, uint8 PortIdx)
  Service ID: 0x19
  Sync/Async:Synchronous
  Reentrancy: Non Reentrant

Parameters (in): SwitchIdx: Index of the switch within the context of the Ethernet Switch Driver
PortIdx: Index of the port at the addressed switch
Parameters (in/out): none
Parameters (out): none
Return value: none
Description: Is called, if a link which is configured goes down.

~[SWS_EthSwt_00118]The function <EthSwtLinkDownCallout> shall be called if a link, which is configured, goes down. The function provides the Switch index and the Port index, such that the port which went down can be identified.

-[SWS_EthSwt_00119]

add second subchapter:

move SWS_EthSwt_00203 here and change to:
Service name: <EthSwtLinkUpCallout>
syntax: void <EthSwtLinkUpCallout>(uint8 SwitchIdx, uint8 PortIdx)
Service ID: 0x1a
Sync/Async:Synchronous
Reentrancy: Non Reentrant
Parameters (in): SwitchIdx: Index of the switch within the context of the Ethernet Switch Driver
PortIdx: Index of the port at the addressed switch
Parameters (in/out): none
Parameters (out): none
Return value: none
Description: Is called, if a link which is configured goes up

~[SWS_EthSwt_00204]The function <EthSwtLinkUpCallout> shall be called if a link which is configured goes up. The function provides the Switch index and the Port index, such that the port which went up can be identified.

-[SWS_EthSwt_00205] This is no clear as it is a configurable interface now

~ECUC_EthSwt_00048 set EthSwtLinkDownUser to obsolete
+ add optional function name parameter with same attributes named EthSwtLinkDownCallout
change description to Defines the function name for <EthSwtLinkDownCallout>.

~ECUC_EthSwt_00068 set EthSwtLinkUpUser to obsolete
+add optional function name parameter with same attributes named EthSwtLinkUpCallout
change description to Defines the function name for <EthSwtLinkUpCallout>.

set ECUC_EthSwt_00047 EthSwtPortEnableLinkDownCallback to obsolete.

Header File name is defined by ECUC_EthSwt_00064 EthSwtPublicCddHeaderFile.
–Last change on issue 76531 comment 54–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.133 Specification Item SWS_EthSwt_00205

Trace References:

SRS_BSW_00171

Content:

The function <User>_LinkUp shall be pre compile time configurable by the <user> with content of EthSwtLinkUpUser.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76531: [EthSwt] Syntax for callback functions <user>_LinkUp and <user>_LinkDown might be wrong

  Problem description:

  The callback function for <User>_LinkUp and <User>_LinkDown to inform the user about a Link state change might be wrong
  Currently the parameter of these both functions are defined as Pointer to uint8
  This makes no sense for me to use here pointers and not a uint 8 variable.

  Agreed solution:

  Chapter 8.6.3 Configurable Interfaces
  add following intro:
  In this chapter all interfaces are listed where the target function could be configured.
  The names of these kind of interfaces are not fixed because they are configurable.

Add Subchapter with interface
move SWS_EthSwt_00117 here and change to:
Service name: <EthSwtLinkDownCallout>
syntax: void <EthSwtLinkDownCallout>(uint8 SwitchIdx, uint8 PortIdx)
Service ID: 0x19
Sync/Async:Synchronous
Reentrancy: Non Reentrant
Parameters (in): SwitchIdx: Index of the switch within the context of the Ethernet Switch Driver
PortIdx: Index of the port at the addressed switch
Parameters (in/out): none
Parameters (out): none
Return value: none
Description: Is called, if a link which is configured goes down.

~[SWS_EthSwt_00118]The function <EthSwtLinkDownCallout> shall be called if a link, which is configured, goes down. The function provides the Switch index and the Port index, such that the port which went down can be identified.

-[SWS_EthSwt_00119]

add second subchapter:

move SWS_EthSwt_00203 here and change to:
Service name: <EthSwtLinkUpCallout>
syntax: void <EthSwtLinkUpCallout>(uint8 SwitchIdx, uint8 PortIdx)
Service ID: 0x1a
Sync/Async:Synchronous
Reentrancy: Non Reentrant
Parameters (in): SwitchIdx: Index of the switch within the context of the Ethernet Switch Driver
PortIdx: Index of the port at the addressed switch
Parameters (in/out): none
Parameters (out): none
Return value: none
Description: Is called, if a link which is configured goes up

~[SWS_EthSwt_00204]The function <EthSwtLinkUpCallout> shall be called if a link which is configured goes up. The function provides the Switch index and the Port index, such that the port which went up can be identified.

-[SWS_EthSwt_00205] This is no clear as it is a configurable interface now

~ECUC_EthSwt_00048 set EthSwtLinkDownUser to obsolete
+ add optional function name parameter with same attributes named EthSwtLinkDownCallout
change description to Defines the function name for <EthSwtLinkDownCallout>.

~ECUC_EthSwt_00068 set EthSwtLinkUpUser to obsolete
+add optional function name parameter with same attributes named EthSwtLinkUpCallout
change description to Defines the function name for <EthSwtLinkUpCallout>.

set ECUC_EthSwt_00047 EthSwtPortEnableLinkDownCallback to obsolete.

Header File name is defined by ECUC_EthSwt_00064 EthSwtPublicCddHeaderFile.
–Last change on issue 76531 comment 54–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.134 Specification Item SWS_EthSwt_00206

Trace References:

SRS_ETH_00086 Eth_00120

Content:

Service name:	EthSwt_GetSwitchRegEthSwt_GetSwitchReg	
Syntax:	Std_ReturnType EthSwt_GetSwitchReg(uint8 SwitchIdx, uint32 page, uint32 register, uint32* registerContent)	
Service ID[hex]:	0x0e	
Sync/Async:	Synchronous /Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	SwitchIdxEthSwt_GetSwitchReg.SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	pageEthSwt_GetSwitchReg.page	Address of a register page
	registerEthSwt_GetSwitchReg.register	Address of a register
Parameters (inout):	None	

Parameters (out):	registerContentEthSwt_GetSwitch Reg.registerContent	Content of the addresses register
Return value:	Std_ReturnType	E_OK: success E_NOT_OK: drop counter could not be obtained
Description:	Generic API for reading the content of a switch register	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76248: [EthSwt]Remove references to non existing requirements (SRS, SWS)

  Problem description:

  Some review findings brought up references to non existing requirements. Please check and remove the references and if needed reformulate the requirements.

  Agreed solution:

  SWS_EthSwt_00123 –> SRS_BSW_00406
  SWS_EthSwt_00165 -> SRS_BSW_00395
  SWS_EthSwt_00227 -> SRS_ETH_00087
  SWS_EthSwt_91017, SWS_EthSwt_91020, –> SRS_ETH_00123
  SWS_EthSwt_00006 –> SRS_BSW_00101
  SWS_EthSwt_00010 left blank as SRS doe not yet provide a requirement here but maybe with RFC # 75170 AUTOSAR will introduce an SRS-item after all.
  SWS_EthSwt_00376, SWS_EthSwt_00374, SWS_EthSwt_00375 –> SRS-Missing! same as for SWS_EThSwt_00010
  SWS_EthSwt_00018, SRS_EthSwt_00025 –> SRS_ETH_00118
  SWS_EthSwt_00031 –> SRS_ETH_00087
  SWS_EthSwt_00037 –> SRS_ETH_00119
  SWS_EthSwt_00044, SWS_EthSwt_00051 –> SRS_ETH_00118
  SWS_EthSwt_00060 –> SRS_ETH_00087
  SWS_EthSwt_00111 –> erase SRS_ETH_00086
  SWS_EthSwt00079 –>SRS_ETH_00119
  SWS_EthSwt_00206, SWS_EthSwt_00211, SWS_EthSwt_00216, SWS_EthSwt_00221 –>SRS_Eth_00120
  SWS_EthSwt_00172 –>SRS_ETH_00121, SRS_ETH_00114
  SWS_EthSwt_00086, SWS_EthSwt_00091, SWS_EthSwt_00182, SWS_EthSwt_00125 –> SRS_ETH_00087, SRS_ETH_00122
  SWS_EthSwt_00187 –> SRS_ETH_00087
  SWS_EthSwt_00058 –> SRS_BSW_00171
  SWS_EthSwt_91012, SWS_EthSwt_91013 –> SRS_ETH_00126
  SWS_EthSwt_91014, SWS_EthSwt_91015, SWS_EthSwt_91016, SWS_EthSwt_91018, SWS_EthSwt_91019, SWS_EthSwt_91021, SWS_EthSwt_91022, SWS_EthSwt_91022, SWS_EthSwt_91023,

SWS_EthSwt_91024, SWS_EthSwt_91025, SWS_EthSwt_91026, SWS_EthSwt_91027 –> SRS_Eth_00123
SWS_EthSwt_00117, SWS_EthSwt_00203 –> SRS_ETH_00119, SRS_ETH_00087
SWS_EthSwt_00193 –> SRS_ETH_00122, SRS_ETH_00087
SWS_EthSwt_00114 –> SRS_BSW_00433
SWS_EthSwt_00098 –> SRS_Eth_00122, SRS_ETH_00118, SRS_ETH_00119, SRS_ETH_00120, SRS_ETH_00087, SRS_ETH_00125, SRS_BSW_00375
–Last change on issue 76248 comment 13–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.135 Specification Item SWS_EthSwt_00208

Trace References:

SRS_BSW_00406

Content:

If development error detection is enabled: the function EthSwt_GetSwitchRegs shall check that the service EthSwt_SwitchInit was previously called. If the check fails, the function shall raise the development error ETHSWT_E_NOT_INITIALIZED and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to

If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.136 Specification Item SWS_EthSwt_00209

Trace References:

SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchIdx is not valid,EthSwt_GetSwitchReg shall raise the development error ETHSWT_E_INV_SWITCH_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.

Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.137 Specification Item SWS_EthSwt_00211

Trace References:

SRS_ETH_00086 Eth_00120

Content:

Service name:	EthSwt_SetSwitchRegEthSwt_SetSwitchReg	
Syntax:	Std_ReturnType EthSwt_SetSwitchReg(uint8 SwitchIdx, uint32 page, uint32 register, uint32 registerContent)	
Service ID[hex]:	0x0f	
Sync/Async:	Synchronous /Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	SwitchIdxEthSwt_SetSwitchReg.SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	pageEthSwt_SetSwitchReg.page	Address of a register page
	registerEthSwt_SetSwitchReg.register	Address of a register
	registerContentEthSwt_SetSwitchReg.registerContent	Content of the addresses register
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: success E_NOT_OK: drop counter could not be obtained
Description:	Generic API for writing the content of a switch register	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76248: [EthSwt]Remove references to non existing requirements (SRS, SWS)

  Problem description:

  Some review findings brought up references to non existing requirements. Please check and remove the references and if needed reformulate the requirements.

Agreed solution:

SWS_EthSwt_00123 –> SRS_BSW_00406
SWS_EthSwt_00165 -> SRS_BSW_00395
SWS_EthSwt_00227 -> SRS_ETH_00087
SWS_EthSwt_91017, SWS_EthSwt_91020, –> SRS_ETH_00123
SWS_EthSwt_00006 –> SRS_BSW_00101
SWS_EthSwt_00010 left blank as SRS doe not yet provide a requirement here but maybe with RFC # 75170 AUTOSAR will introduce an SRS-item after all.
SWS_EthSwt_00376, SWS_EthSwt_00374, SWS_EthSwt_00375 –> SRS-Missing! same as for SWS_EThSwt_00010
SWS_EthSwt_00018, SRS_EthSwt_00025 –> SRS_ETH_00118
SWS_EthSwt_00031 –> SRS_ETH_00087
SWS_EthSwt_00037 –> SRS_ETH_00119
SWS_EthSwt_00044, SWS_EthSwt_00051 –> SRS_ETH_00118
SWS_EthSwt_00060 –> SRS_ETH_00087
SWS_EthSwt_00111 –> erase SRS_ETH_00086
SWS_EthSwt00079 –>SRS_ETH_00119
SWS_EthSwt_00206, SWS_EthSwt_00211, SWS_EthSwt_00216, SWS_EthSwt_00221 –>SRS_Eth_00120
SWS_EthSwt_00172 –>SRS_ETH_00121, SRS_ETH_00114
SWS_EthSwt_00086, SWS_EthSwt_00091, SWS_EthSwt_00182, SWS_EthSwt_00125 –> SRS_ETH_00087, SRS_ETH_00122
SWS_EthSwt_00187 –> SRS_ETH_00087
SWS_EthSwt_00058 –> SRS_BSW_00171
SWS_EthSwt_91012, SWS_EthSwt_91013 –> SRS_ETH_00126
SWS_EthSwt_91014, SWS_EthSwt_91015, SWS_EthSwt_91016, SWS_EthSwt_91018, SWS_EthSwt_91019, SWS_EthSwt_91021, SWS_EthSwt_91022, SWS_EthSwt_91022, SWS_EthSwt_91023, SWS_EthSwt_91024, SWS_EthSwt_91025, SWS_EthSwt_91026, SWS_EthSwt_91027 –> SRS_Eth_00123
SWS_EthSwt_00117, SWS_EthSwt_00203 –> SRS_ETH_00119, SRS_ETH_00087
SWS_EthSwt_00193 –> SRS_ETH_00122, SRS_ETH_00087
SWS_EthSwt_00114 –> SRS_BSW_00433
SWS_EthSwt_00098 –> SRS_Eth_00122, SRS_ETH_00118, SRS_ETH_00119, SRS_ETH_00120, SRS_ETH_00087, SRS_ETH_00125, SRS_BSW_00375
–Last change on issue 76248 comment 13–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.138 Specification Item SWS_EthSwt_00213

Trace References:

SRS_BSW_00406

Content:

If development error detection is enabled: the function EthSwt_SetSwitchRegs shall check that the service EthSwt_SwitchInit was previously called. If the check fails, the function shall raise the development error ETHSWT_E_NOT_INITIALIZED and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.139 Specification Item SWS_EthSwt_00214

Trace References:

SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchIdx is not valid,EthSwt_SetSwitchReg shall raise the development error ETHSWT_E_INV_SWITCH_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

  For Error Detection following general rules apply:
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with in-

put parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	**Specification**	**Bus**
1	4	1

1.140 Specification Item SWS_EthSwt_00216

Trace References:

SRS_ETH_00086 Eth_00120

Content:

Service name:	EthSwt_ReadTrcvRegisterEthSwt_ReadTrcvRegister

Syntax:	Std_ReturnType EthSwt_ReadTrcvRegister(uint8 SwitchIdx, uint8 SwitchPortIdx, uint8 RegIdx, uint16* RegValPtr)	
Service ID[hex]:	0x10	
Sync/Async:	Synchronous /Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	SwitchIdxEthSwt_ReadTrcvRegister.SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	SwitchPortIdxEthSwt_ReadTrcvRegister.SwitchPortIdx	Index of the port at the addressed switch
	RegIdxEthSwt_ReadTrcvRegister.RegIdx	Index of the register
Parameters (inout):	None	
Parameters (out):	RegValPtrEthSwt_ReadTrcvRegister.RegValPtr	Pointer to the register content
Return value:	Std_ReturnType	E_OK: success E_NOT_OK: drop counter could not be obtained
Description:	Generic API for reading the content of a transceiver register	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76248: [EthSwt]Remove references to non existing requirements (SRS, SWS)

  Problem description:

  Some review findings brought up references to non existing requirements. Please check and remove the references and if needed reformulate the requirements.

  Agreed solution:

  SWS_EthSwt_00123 –> SRS_BSW_00406
  SWS_EthSwt_00165 -> SRS_BSW_00395
  SWS_EthSwt_00227 -> SRS_ETH_00087
  SWS_EthSwt_91017, SWS_EthSwt_91020, –> SRS_ETH_00123
  SWS_EthSwt_00006 –> SRS_BSW_00101
  SWS_EthSwt_00010 left blank as SRS doe not yet provide a requirement here but maybe with RFC # 75170 AUTOSAR will introduce an SRS-item after all.
  SWS_EthSwt_00376, SWS_EthSwt_00374, SWS_EthSwt_00375 –> SRS-Missing! same as for SWS_EThSwt_00010
  SWS_EthSwt_00018, SRS_EthSwt_00025 –> SRS_ETH_00118
  SWS_EthSwt_00031 –> SRS_ETH_00087
  SWS_EthSwt_00037 –> SRS_ETH_00119
  SWS_EthSwt_00044, SWS_EthSwt_00051 –> SRS_ETH_00118
  SWS_EthSwt_00060 –> SRS_ETH_00087

SWS_EthSwt_00111 –> erase SRS_ETH_00086
SWS_EthSwt00079 –>SRS_ETH_00119
SWS_EthSwt_00206, SWS_EthSwt_00211, SWS_EthSwt_00216, SWS_EthSwt_00221 –>SRS_Eth_00120
SWS_EthSwt_00172 –>SRS_ETH_00121, SRS_ETH_00114
SWS_EthSwt_00086, SWS_EthSwt_00091, SWS_EthSwt_00182, SWS_EthSwt_00125 –> SRS_ETH_00087, SRS_ETH_00122
SWS_EthSwt_00187 –> SRS_ETH_00087
SWS_EthSwt_00058 –> SRS_BSW_00171
SWS_EthSwt_91012, SWS_EthSwt_91013 –> SRS_ETH_00126
SWS_EthSwt_91014, SWS_EthSwt_91015, SWS_EthSwt_91016, SWS_EthSwt_91018, SWS_EthSwt_91019, SWS_EthSwt_91021, SWS_EthSwt_91022, SWS_EthSwt_91022, SWS_EthSwt_91023, SWS_EthSwt_91024, SWS_EthSwt_91025, SWS_EthSwt_91026, SWS_EthSwt_91027 –> SRS_Eth_00123
SWS_EthSwt_00117, SWS_EthSwt_00203 –> SRS_ETH_00119, SRS_ETH_00087
SWS_EthSwt_00193 –> SRS_ETH_00122, SRS_ETH_00087
SWS_EthSwt_00114 –> SRS_BSW_00433
SWS_EthSwt_00098 –> SRS_Eth_00122, SRS_ETH_00118, SRS_ETH_00119, SRS_ETH_00120, SRS_ETH_00087, SRS_ETH_00125, SRS_BSW_00375
–Last change on issue 76248 comment 13–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.141 Specification Item SWS_EthSwt_00218

Trace References:

SRS_BSW_00406

Content:

If development error detection is enabled: the function EthSwt_ReadTrcvRegister shall check that the service EthSwt_SwitchInit was previously called. If the check fails, the function shall raise the development error ETHSWT_E_NOT_INITIALIZED and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

Problem description:

[SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

In the above requirement -
1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with

input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.142 Specification Item SWS_EthSwt_00219

Trace References:

SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchIdx is not valid,EthSwt_ReadTrcvRegister shall raise the development error ETHSWT_E_INV_SWITCH_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

In the above requirement -
1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETH-

SWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	**Specification**	**Bus**
1	4	1

1.143 Specification Item SWS_EthSwt_00221

Trace References:

SRS_ETH_00086 Eth_00120

Content:

Service name:	EthSwt_WriteTrcvRegisterEthSwt_WriteTrcvRegister	
Syntax:	Std_ReturnType EthSwt_WriteTrcvRegister(uint8 SwitchIdx, uint8 SwitchPortIdx, uint8 RegIdx, uint16 RegVal)	
Service ID[hex]:	0x11	
Sync/Async:	Synchronous /Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	SwitchIdxEthSwt_WriteTrcv Register.SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	SwitchPortIdxEthSwt_WriteTrcv Register.SwitchPortIdx	Index of the port at the addressed switch
	RegIdxEthSwt_WriteTrcvRegister.Reg Idx	Index of the register
	RegValEthSwt_WriteTrcvRegister.Reg Val	Content for the indexed register
Parameters (inout):	None	
Parameters (out):	None	

Return value:	Std_ReturnType	E_OK: success E_NOT_OK: drop counter could not be obtained
Description:	Generic API for writing the content of a transceiver register	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76248: [EthSwt]Remove references to non existing requirements (SRS, SWS)

  Problem description:

  Some review findings brought up references to non existing requirements. Please check and remove the references and if needed reformulate the requirements.

  Agreed solution:

  SWS_EthSwt_00123 –> SRS_BSW_00406
  SWS_EthSwt_00165 -> SRS_BSW_00395
  SWS_EthSwt_00227 -> SRS_ETH_00087
  SWS_EthSwt_91017, SWS_EthSwt_91020, –> SRS_ETH_00123
  SWS_EthSwt_00006 –> SRS_BSW_00101
  SWS_EthSwt_00010 left blank as SRS doe not yet provide a requirement here but maybe with RFC # 75170 AUTOSAR will introduce an SRS-item after all.
  SWS_EthSwt_00376, SWS_EthSwt_00374, SWS_EthSwt_00375 –> SRS-Missing! same as for SWS_EThSwt_00010
  SWS_EthSwt_00018, SRS_EthSwt_00025 –> SRS_ETH_00118
  SWS_EthSwt_00031 –> SRS_ETH_00087
  SWS_EthSwt_00037 –> SRS_ETH_00119
  SWS_EthSwt_00044, SWS_EthSwt_00051 –> SRS_ETH_00118
  SWS_EthSwt_00060 –> SRS_ETH_00087
  SWS_EthSwt_00111 –> erase SRS_ETH_00086
  SWS_EthSwt00079 –>SRS_ETH_00119
  SWS_EthSwt_00206, SWS_EthSwt_00211, SWS_EthSwt_00216, SWS_EthSwt_00221 –>SRS_Eth_00120
  SWS_EthSwt_00172 –>SRS_ETH_00121, SRS_ETH_00114
  SWS_EthSwt_00086, SWS_EthSwt_00091, SWS_EthSwt_00182, SWS_EthSwt_00125 –> SRS_ETH_00087, SRS_ETH_00122
  SWS_EthSwt_00187 –> SRS_ETH_00087
  SWS_EthSwt_00058 –> SRS_BSW_00171
  SWS_EthSwt_91012, SWS_EthSwt_91013 –> SRS_ETH_00126
  SWS_EthSwt_91014, SWS_EthSwt_91015, SWS_EthSwt_91016, SWS_EthSwt_91018, SWS_EthSwt_91019, SWS_EthSwt_91021, SWS_EthSwt_91022, SWS_EthSwt_91022, SWS_EthSwt_91023, SWS_EthSwt_91024, SWS_EthSwt_91025, SWS_EthSwt_91026, SWS_EthSwt_91027 –> SRS_Eth_00123

SWS_EthSwt_00117, SWS_EthSwt_00203 –> SRS_ETH_00119, SRS_ETH_00087
SWS_EthSwt_00193 –> SRS_ETH_00122, SRS_ETH_00087
SWS_EthSwt_00114 –> SRS_BSW_00433
SWS_EthSwt_00098 –> SRS_Eth_00122, SRS_ETH_00118, SRS_ETH_00119, SRS_ETH_00120, SRS_ETH_00087, SRS_ETH_00125, SRS_BSW_00375
–Last change on issue 76248 comment 13–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.144 Specification Item SWS_EthSwt_00223

Trace References:

SRS_BSW_00406

Content:

If development error detection is enabled: the function EthSwt_WriteTrcvRegister shall check that the service EthSwt_SwitchInit was previously called. If the check fails, the function shall raise the development error ETHSWT_E_NOT_INITIALIZED and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the

development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.145 Specification Item SWS_EthSwt_00224

Trace References:

SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchIdx is not valid, EthSwt_WriteTrcvRegister shall raise the development error ETHSWT_E_INV_SWITCH_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.146 Specification Item SWS_EthSwt_00227

Trace References:

SRS_ETH_00086 00087

Content:

Name:	EthSwt_MacLearningTypeEthSwt_MacLearningType	
Type:	Enumeration	
Range:	ETHSWT_MACLEARNING_HWDISABLEDEth Swt_MacLearning Type.ETHSWT_MACLEARNING_HWDISABLED	If hardware learning disabled, the switch must not learn new MAC addresses
	ETHSWT_MACLEARNING_HWENABLEDEth Swt_MacLearning Type.ETHSWT_MACLEARNING_HWENABLED	If hardware learning enabled, the switch learns new MAC addresses
	ETHSWT_MACLEARNING_SWENABLEDEth Swt_MacLearning Type.ETHSWT_MACLEARNING_SWENABLED	If software learning enabled, the hardware learning is disabled and the switch forwards packets with an unknown source address to a host CPU
Description:	The interpretation of this value	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76248: [EthSwt]Remove references to non existing requirements (SRS, SWS)

  Problem description:

  Some review findings brought up references to non existing requirements. Please check and remove the references and if needed reformulate the requirements.

  Agreed solution:

  SWS_EthSwt_00123 –> SRS_BSW_00406
  SWS_EthSwt_00165 -> SRS_BSW_00395
  SWS_EthSwt_00227 -> SRS_ETH_00087
  SWS_EthSwt_91017, SWS_EthSwt_91020, –> SRS_ETH_00123
  SWS_EthSwt_00006 –> SRS_BSW_00101
  SWS_EthSwt_00010 left blank as SRS doe not yet provide a requirement here but maybe with RFC # 75170 AUTOSAR will introduce an SRS-item after all.
  SWS_EthSwt_00376, SWS_EthSwt_00374, SWS_EthSwt_00375 –> SRS-Missing!

same as for SWS_EThSwt_00010
SWS_EthSwt_00018, SRS_EthSwt_00025 –> SRS_ETH_00118
SWS_EthSwt_00031 –> SRS_ETH_00087
SWS_EthSwt_00037 –> SRS_ETH_00119
SWS_EthSwt_00044, SWS_EthSwt_00051 –> SRS_ETH_00118
SWS_EthSwt_00060 –> SRS_ETH_00087
SWS_EthSwt_00111 –> erase SRS_ETH_00086
SWS_EthSwt00079 –>SRS_ETH_00119
SWS_EthSwt_00206, SWS_EthSwt_00211, SWS_EthSwt_00216, SWS_EthSwt_00221 –>SRS_Eth_00120
SWS_EthSwt_00172 –>SRS_ETH_00121, SRS_ETH_00114
SWS_EthSwt_00086, SWS_EthSwt_00091, SWS_EthSwt_00182, SWS_EthSwt_00125 –> SRS_ETH_00087, SRS_ETH_00122
SWS_EthSwt_00187 –> SRS_ETH_00087
SWS_EthSwt_00058 –> SRS_BSW_00171
SWS_EthSwt_91012, SWS_EthSwt_91013 –> SRS_ETH_00126
SWS_EthSwt_91014, SWS_EthSwt_91015, SWS_EthSwt_91016, SWS_EthSwt_91018, SWS_EthSwt_91019, SWS_EthSwt_91021, SWS_EthSwt_91022, SWS_EthSwt_91022, SWS_EthSwt_91023, SWS_EthSwt_91024, SWS_EthSwt_91025, SWS_EthSwt_91026, SWS_EthSwt_91027 –> SRS_Eth_00123
SWS_EthSwt_00117, SWS_EthSwt_00203 –> SRS_ETH_00119, SRS_ETH_00087
SWS_EthSwt_00193 –> SRS_ETH_00122, SRS_ETH_00087
SWS_EthSwt_00114 –> SRS_BSW_00433
SWS_EthSwt_00098 –> SRS_Eth_00122, SRS_ETH_00118, SRS_ETH_00119, SRS_ETH_00120, SRS_ETH_00087, SRS_ETH_00125, SRS_BSW_00375
–Last change on issue 76248 comment 13–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.147 Specification Item SWS_EthSwt_00237

Trace References:

SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchIdx is not valid, EthSwt_GetArlTable API shall raise the development error ETHSWT_E_INV_SWITCH_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

  For Error Detection following general rules apply:
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with

input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.148 Specification Item SWS_EthSwt_00238

Trace References:

SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchPortIdx is not valid, EthSwt_ GetCounterValues shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

  For Error Detection following general rules apply:
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
  [SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.

[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.149 Specification Item SWS_EthSwt_00239

Trace References:

SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter CounterPtr is a NULL pointer, EthSwt_ GetCounterValues shall raise the development error ETHSWT_E_INV_POINTER and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return

E_NOT_OK if the driver is not yet initialized. ()

In the above requirement -
1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being

valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.150 Specification Item SWS_EthSwt_00244

Trace References:

SRS_ETH_00125

Content:

If EthSwt_EnableTimeStamp is called for a SwitchIdx, the switch driver shall enable the time-stamping for all his ports except the ports where EthSwtPortRole is set to ETHSWT_UP_LINK_PORT and if EthSwtPortTimeStampSupport is set to TRUEfor this port.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77050: [EthSwt] Uplink ports shall not be excluded from timestamping

  Problem description:

  [SWS_EthSwt_00244] currently limits timestamping to
  "all ... ports except the ports where EthSwtPortRole is set to ETHSWT_UP_LINK_PORT".

  This limitation is problematic for cascaded switches since
  no timestamps are taken when SYNC messages exit the first
  switch and enter the second one.

It may be possible to omit timestamping on the link
between the cascaded switches if the timestamp counters in both switches are synchronized.
It is then possible to use the ingress timestamp on the first switch in combination with any egress timestamp on the second switch.

However, without a synchronization of the timestamp counters in both switches it is necessary to consider ingress and egress (on the uplink port) timestamps on the first switch and ingress (on the uplink port) and egress timestamps on the second switch.

Since it is unclear if all switch devices support timestamp counter synchronization and since this feature may result in additional hardware requirements (e.g., common clock source, reset lines etc.) it may be a show-stopper to omit timestamping at the uplink ports.

I recommend to either just remove this restriction or to make it configurable in case that the hardware supports synchronized counters.

Agreed solution:

~[SWS_EthSwt_00244][If EthSwt_EnableTimeStamp is called for a SwitchIdx, the switch driver shall enable the time-stamping for all his ports where EthSwtPortTimeStampSupport is set to TRUE.]

~[SWS_EthSwt_00378] If EthSwt_PortEnableTimeStamp is called for a PortIdx, the switch driver shall enable the time-stamping for this port if EthSwtPortTimeStampSupport is set to TRUE for this port.

~SWS_EthSwt_91011 EthSwt_EnableTimeStamping
Description: Activates egress time stamping on a dedicated message object on all ports of a Switch where EthSwtPortTimeStampSupport is set to TRUE. The selective activation of dedicated message objects for time stamping reduces the number of notification calls only to the required calls. Some HW does store once the egress time stamp marker and some HW needs it always before transmission. There will be no disabled functionality, due to the fact, that the message type is always "time stamped" by network design.

~SWS_EthSwt_91028 EthSwt_PortEnableTimeStamp
Description: Activates egress time stamping on a dedicated message object on a dedicated port of a Switch if EthSwtPortTimeStampSupport is set to TRUE for this port. The selective activation of dedicated message objects for time stamping reduces the number of notification calls only to the required calls. Some HW does

store once the egress time stamp marker and some HW needs it always before transmission. There will be no disabled functionality, due to the fact, that the message type is always "time stamped" by network design.
–Last change on issue 77050 comment 9–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.151 Specification Item SWS_EthSwt_00247

Trace References:

SRS_BSW_00406

Content:

If default error detection is enabled: the function EthSwt_MgmtInit() shall check that the service EthSwt_Init() was previously called.

If the check fails, the function EthSwt_MgmtInit() shall raise the development error ETHSWT_E_NOT_INITIALIZED.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.

Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.152 Specification Item SWS_EthSwt_00250

Trace References:

SRS_BSW_00406

Content:

If default error detection is enabled: the function EthSwt_EthRxProcessFrame() shall check that the service EthSwt_Init() was previously called.

If the check fails, the function EthSwt_EthRxProcessFrame() shall raise the development error ETHSWT_E_NOT_INITIALIZED and return E_NOT_OK..

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.153 Specification Item SWS_EthSwt_00251

Trace References:

SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369

Content:

If default error detection is enabled: the function EthSwt_EthRxProcessFrame() shall check the parameter CtrlIdx for being valid. If the check fails, the function EthSwt_EthRxProcessFrame() shall raise the development error ETHSWT_E_INV_CTRL_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:

SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.154 Specification Item SWS_EthSwt_00252

Trace References:

SRS_BSW_323, SRS_BSW_369

Content:

If default error detection is enabled: the function EthSwt_EthRxProcessFrame() shall check the parameter BufIdx for being valid.

If the check fails, the function EthSwt_EthRxProcessFrame()raise the development error ETHSWT_E_INV_PARAM and return E_NOT_OK

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.155 Specification Item SWS_EthSwt_00254

Trace References:

SRS_BSW_00406

Content:

If default error detection is enabled: the function EthSwt_EthRxFinishedIndication() shall check that the service EthSwt_Init() was previously called.

If the check fails, the function EthSwt_EthRxFinishedIndication() shall raise the development error ETHSWT_E_NOT_INITIALIZED and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

  For Error Detection following general rules apply:
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the

check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.156 Specification Item SWS_EthSwt_00255

Trace References:

SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369

Content:

If default error detection is enabled: the function EthSwt_EthRxFinishedIndication() shall check the parameter CtrlIdx for being valid.

If the check fails, the function EthSwt_EthRxFinishedIndication() shall raise the development error ETHSWT_E_INV_CTRL_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

  For Error Detection following general rules apply:
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails,

the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.157 Specification Item SWS_EthSwt_00256

Trace References:

SRS_BSW_323, SRS_BSW_369

Content:

If default error detection is enabled: the function EthSwt_EthRxFinishedIndication() shall check the parameter BufIdx for being valid.

If the check fails, the function EthSwt_EthRxFinishedIndication() shall raise the development error ETHSWT_E_INV_PARAM and return E_NOT_OK

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

  For Error Detection following general rules apply:
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being

valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.158 Specification Item SWS_EthSwt_00258

Trace References:

SRS_BSW_00406

Content:

If default error detection is enabled: the function EthSwt_EthTxPrepareFrame() shall check that the service EthSwt_Init() was previously called.

If the check fails, the function EthSwt_EthTxPrepareFrame() shall raise the development error ETHSWT_E_NOT_INITIALIZED and return E_NOT_OK

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

  For Error Detection following general rules apply:
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
  [SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with

input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.159 Specification Item SWS_EthSwt_00259

Trace References:

SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369

Content:

If default error detection is enabled: the function EthSwt_EthTxPrepareFrame() shall check the parameter CtrlIdx for being valid.

If the check fails, the function EthSwt_EthTxPrepareFrame() shall raise the development error ETHSWT_E_INV_CTRL_IDX. and return E_NOT_OK

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

Problem description:

[SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

In the above requirement -
1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with

input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.160 Specification Item SWS_EthSwt_00260

Trace References:

SRS_BSW_323, SRS_BSW_369

Content:

If default error detection is enabled: the function EthSwt_EthTxPrepareFrame() shall check the parameter BufIdx for being valid.

If the check fails, the function EthSwt_EthTxPrepareFrame() shall raise the development error ETHSWT_E_INV_PARAM and return E_NOT_OK

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

[SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

In the above requirement -
1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.

[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.161 Specification Item SWS_EthSwt_00262

Trace References:

SRS_BSW_00406

Content:

If default error detection is enabled: the function EthSwt_EthTxAdaptBufferLength() shall check that the service EthSwt_Init() was previously called.

If the check fails, the function EthSwt_EthTxAdaptBufferLength() shall raise the development error ETHSWT_E_NOT_INITIALIZED and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

In the above requirement -
1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETH-

SWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.162 Specification Item SWS_EthSwt_00263

Trace References:

SRS_BSW_323, SRS_BSW_369

Content:

If default error detection is enabled: the function EthSwt_EthTxAdaptBufferLength() shall check the parameter LengthPtr for being valid.

If the check fails, the function EthSwt_EthTxAdaptBufferLength() shall raise the development error ETHSWT_E_INV_POINTER and return E_NOT_OK

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most

probably should be development error detection.

Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain

the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	**Specification**	**Bus**
1	4	1

1.163 Specification Item SWS_EthSwt_00265

Trace References:

SRS_BSW_00406

Content:

If default error detection is enabled: the function EthSwt_SetMgmtInfo() shall check that the service EthSwt_Init() was previously called.

If the check fails, the function EthSwt_SetMgmtInfo() shall raise the development error ETHSWT_E_NOT_INITIALIZED and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not de-

fault errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.164 Specification Item SWS_EthSwt_00266

Trace References:

SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369

Content:

f default error detection is enabled: the function EthSwt_SetMgmtInfo() shall check the parameter CtrlIdx for being valid.

If the check fails, the function EthSwt_EthTxPrepareFrame() shall raise the development error ETHSWT_E_INV_CTRL_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check

the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.165 Specification Item SWS_EthSwt_00267

Trace References:

SRS_BSW_323, SRS_BSW_369

Content:

If default error detection is enabled: the function EthSwt_SetMgmtInfo() shall check the parameter BufIdx for being valid.

If the check fails, the function EthSwt_SetMgmtInfo() shall raise the development error ETHSWT_E_INV_PARAM and return E_NOT_OK

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.

Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.166 Specification Item SWS_EthSwt_00269

Trace References:

SRS_BSW_00406

Content:

If default error detection is enabled: the function EthSwt_EthTxProcessFrame() shall check that the service EthSwt_Init() was previously called.

If the check fails, the function EthSwt_EthTxProcessFrame() shall raise the development error ETHSWT_E_NOT_INITIALIZED and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.167 Specification Item SWS_EthSwt_00270

Trace References:

SRS_BSW_323, SRS_BSW_369

Content:

If default error detection is enabled: the function EthSwt_EthTxProcessFrame() shall check the parameter CtrlIdx for being valid.

If the check fails, the function EthSwt_EthTxProcessFrame() shall raise the development error ETHSWT_E_INV_CTRL_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:

SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.168 Specification Item SWS_EthSwt_00271

Trace References:

SRS_BSW_323, SRS_BSW_369

Content:

If default error detection is enabled: the function EthSwt_EthTxProcessFrame() shall check the parameter BufIdx for being valid.

If the check fails, the function EthSwt_EthTxProcessFrame() shall raise the development error ETHIF_E_INV_PARAM and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.169 Specification Item SWS_EthSwt_00272

Trace References:

SRS_BSW_323, SRS_BSW_369

Content:

If default error detection is enabled: the function EthSwt_EthTxProcessFrame() shall check the parameter DataPtr and LengthPtr for being valid.

If the check fails, the function EthSwt_EthTxProcessFrame() shall raise the development error ETHSWT_E_INV_POINTER.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

  For Error Detection following general rules apply:
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the

check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.170 Specification Item SWS_EthSwt_00274

Trace References:

SRS_BSW_00406

Content:

If default error detection is enabled: the function EthSwt_EthTxFinishedIndication() shall check that the service EthSwt_Init() was previously called.

If the check fails, the function EthSwt_EthTxFinishedIndication() shall raise the development error ETHSWT_E_NOT_INITIALIZED and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

  For Error Detection following general rules apply:
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails,

the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.171 Specification Item SWS_EthSwt_00275

Trace References:

SRS_BSW_323, SRS_BSW_369

Content:

If default error detection is enabled: the function EthSwt_EthTxFinishedIndication() shall check the parameter CtrlIdx for being valid.

If the check fails, the function EthSwt_EthTxFinishedIndication() shall raise the development error ETHSWT_E_INV_CTRL_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

  For Error Detection following general rules apply:
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being

valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.172 Specification Item SWS_EthSwt_00276

Trace References:

SRS_BSW_323, SRS_BSW_369

Content:

If default error detection is enabled: the function EthSwt_EthTxFinishedIndication() shall check the parameter BufIdx for being valid.

If the check fails, the function EthSwt_EthTxFinishedIndication() shall raise the development error ETHIF_E_INV_PARAM and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

  For Error Detection following general rules apply:
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
  [SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with

input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.173 Specification Item SWS_EthSwt_00278

Trace References:

SRS_BSW_00406

Content:

If default error detection is enabled: the function EthSwt_EnableTimeStamping() shall check that the service EthSwt_Init() was previously called.

If the check fails, the function EthSwt_EnableTimeStamping() shall raise the development error ETHIF_E_NOT_INITIALIZED and return E_NOT_OK..

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

Problem description:

[SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

In the above requirement -
1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with

input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.174 Specification Item SWS_EthSwt_00279

Trace References:

SRS_BSW_323, SRS_BSW_369

Content:

If default error detection is enabled: the function EthSwt_EnableTimeStamping() shall check the parameter SwitchIdx for being valid.

If the check fails, the function EthSwt_EnableTimeStamping() shall raise the development error ETHIF_E_INV_SWITCH_IDX and return E_NOT_OK..

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

[SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

In the above requirement -
1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.

[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	**Specification**	**Bus**
1	4	1

1.175 Specification Item SWS_EthSwt_00280

Trace References:

SRS_BSW_323, SRS_BSW_369

Content:

If default error detection is enabled: the function EthSwt_EnableTimeStamping() shall check the parameter BufIdx for being valid.

If the check fails, the function EthSwt_EnableTimeStamping() shall raise the development error ETHIF_E_INV_PARAM and return E_NOT_OK..

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

In the above requirement -
1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETH-

SWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.176 Specification Item SWS_EthSwt_00283

Trace References:

SRS_BSW_323, SRS_BSW_369

Content:

If default error detection is enabled: the function EthSwt_EthTxPrepareFrame() shall check the parameter DataPtr and LengthPtr for being valid.

If the check fails, the function EthSwt_EthTxPrepareFrame() shall raise the default error ETHSWT_E_INV_POINTER.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most

probably should be development error detection.

Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain

the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.177 Specification Item SWS_EthSwt_00284

Trace References:

SRS_BSW_323, SRS_BSW_369

Content:

If default error detection is enabled: the function EthSwt_EthRxProcessFrame() shall check the parameter DataPtr, LengthPtr and IsMgmtFrameOnlyPtr for being valid.

If the check fails, the function EthSwt_EthRxProcessFrame() shall raise the default error ETHSWT_E_INV_POINTER and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not de-

fault errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.178 Specification Item SWS_EthSwt_00285

Trace References:

SRS_BSW_00406

Content:

If development error detection is enabled: the function EthSwt_VerifyConfig shall check that the service EthSwt_SwitchInit was previously called. If the check fails, the function shall raise the development error ETHSWT_E_NOT_INITIALIZED and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the

development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.179 Specification Item SWS_EthSwt_00286

Trace References:

SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchIdx is not valid, EthSwt_VerifyConfig shall raise the development error ETHSWT_E_INV_SWITCH_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.180 Specification Item SWS_EthSwt_00289

Trace References:

SRS_BSW_00406

Content:

If development error detection is enabled: the function EthSwt_SetForwardingMode shall check that the service EthSwt_SwitchInit was previously called. If the check fails, the function shall raise the development error ETHSWT_E_NOT_INITIALIZED and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:

SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.181 Specification Item SWS_EthSwt_00290

Trace References:

SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchIdx is not valid, EthSwt_SetForwardingMode shall raise the development error ETHSWT_E_INV_SWITCH_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

  For Error Detection following general rules apply:

[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.182 Specification Item SWS_EthSwt_00293

Trace References:

SRS_Eth_00123

Content:

The function EthSwt_GetPortSignalQualityshall return the value of the signal quality of the indexed Ethernet transceiver that is connected to the indexed port. The indexed port is connected to the indexed EthSwt. The value is provided () shall obtain the signal quality by calling the function EthTrcv_GetPhySignalQualityof the () of the referenced Ethernet Transceiver Driver. If the current signal quality is not available, the signal quality shall be set to 0xFFFFFFFF.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77628: [EthSwt] Behaviour of certain APIs for ports without EthSwtPortTrcvRef (part I)

  Problem description:

  The following APIs have partly a description regarding the handling of ports without EthSwtPortTrcvRef:
  EthSwt_GetPortSignalQuality
  EthSwt_GetPortIdentifier
  EthSwt_SetPortTestMode
  EthTrcv_SetPhyTestMode
  EthSwt_SetPortTxMode
  EthSwt_GetPortCableDiagnosticsResult

  The description should be adjusted and harmonized with related requirements.
  –Last change on issue 77628 comment 2–

  Agreed solution:

  === EthSwt ===
  ~[SWS_EthSwt_91014]EthSwt_GetPortSignalQuality
  Description:
  The function retrieves the signal quality of the link of the indexed Ethernet switch port.
  ~[SWS_EthSwt_00293]
  The function EthSwt_GetPortSignalQuality() shall obtain the signal quality by calling the function EthTrcv_GetPhySignalQuality() of the referenced Ethernet Transceiver Driver. If the current signal quality is not available, the signal quality shall be set to 0xFF.
  -[SWS_EthSwt_00298]as # 77349 introduces a general req

  ~[SWS_EthSwt_91015]EthSwt_GetPortIdentifier
  Return value

E_NOT_OK: organizationally unique identifier of the Ethernet transceiver could not be obtained (i.e. OUI is not available)
Description:
This function retrieves the OUI (24 bit) of the indexed Ethernet switch port.

~[SWS_EthSwt_00299]
The function EthSwt_GetPortIdentifier() shall return the value of the organizationally unique identifier (OUI 24 bit) of the indexed Ethernet switch port that is connected to the indexed Ethernet switch. It shall set the 8 most significant bits of the OUI to 0xFFxxxxxx. If the Ethernet switch port references an Ethernet transceiver, the function shall obtain the OUI by calling the function EthTrcv_GetPhyIdentifier() and set the 8 most significant bits of the OUI to 0x00xxxxxx.

SWS_EthSwt_xxxxx] If neither the Ethernet switch port nor the Ethernet Transceiver Driver can provide an OUI the function EthSwt_GetPortIdentifier() shall return E_NOT_OK.

-[SWS_EthSwt_00304]as # 77349 introduces a general req

~[SWS_EthSwt_91029] EthSwt_SetPortTestMode
Description:
Activates a given test mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00328]
The function EthSwt_SetPortTestMode shall forward the call with the given test mode by calling the function EthTrcv_SetPhyTestMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00333]as # 77349 introduces a general req

~[SWS_EthSwt_91023] EthSwt_SetPortLoopbackMode
Description:
Activates a given test loop-back mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00334]
The function EthSwt_SetPortLoopbackMode() shall forward the call with the given loop-back mode by calling the function EthTrcv_SetPhyLoopbackMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00339] as # 77349 introduces a general req

~[SWS_EthSwt_91024] EthSwt_SetPortTxMode
Description:
Activates a given transmission mode of the indexed Ethernet switch port.

~[SWS_EthSwt_00340]
The function EthSwt_SetPortTxMode() shall forward the call with the given transmission mode by calling the function EthTrcv_SetPhyTxMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00345] as # 77349 introduces a general req

~[SWS_EthSwt_91025]EthSwt_GetPortCableDiagnosticsResult
Description:
Retrieves the cable diagnostics result of the indexed Ethernet switch port respectively the referenced Ethernet Transceiver Driver.

~[SWS_EthSwt_00346]
The function EthSwt_GetPortCableDiagnosticsResult() shall obtain the cable diagnostics result by calling the function EthTrcv_GetCableDiagnosticsResult() of the referenced Ethernet Transceiver Driver. If no Ethernet transceiver is referenced by the Ethernet switch port and development error detection is not enabled, the cable diagnostic result shall be set to ETHTRCV_CABLEDIAG_OK.
-[SWS_EthSwt_00351] as # 77349 introduces a general req

=== EthTrcv ===
add certain parameter to enable/disable API functions:
+ SWS item ECUC_EthTrcv_xxxx1
Name EthTrcvGetPhySignalQualityApi
Description Enables / Disables EthTrcv_GetPhySignalQuality API

+ SWS item ECUC_EthTrcv_xxxx2
Name EthTrcvGetPhyIdentifierApi
Description Enables / Disables EthTrcv_GetPhyIdentifier API

+ SWS item ECUC_EthTrcv_xxxx3
Name EthTrcvSetPhyTestModeApi
Description Enables / Disables EthTrcv_SetPhyTestMode API

+ SWS item ECUC_EthTrcv_xxxx4
Name EthTrcvSetPhyTxModeApi
Description Enables / Disables EthTrcv_SetPhyTxMode API

+ SWS item ECUC_EthTrcv_xxxx5
Name EthTrcvGetCableDiagnosticsResultApi
Description Enables / Disables EthTrcv_GetCableDiagnosticsResult API

add the following specification to the configuration parameter above:
Multiplicity 1

Type EcucBooleanParamDef
Default value –
Post-Build Variant Value false
Value Configuration
Class Pre-compile time X
All Variants Link time –
Post-build time –
Scope / Dependency scope: local
–Last change on issue 77628 comment 31–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.183 Specification Item SWS_EthSwt_00294

Trace References:

SRS_BSW_00406

Content:

If development error detection is enabled: the function EthSwt_GetPortSignalQuality shall check that the service EthSwt_SwitchInit was previously called. If the check fails, the function shall raise the development error ETHSWT_E_NOT_INITIALIZED and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to

If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.184 Specification Item SWS_EthSwt_00295

Trace References:

SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchIdx is not valid, EthSwt_GetPortSignalQuality shall raise the development error ETHSWT_E_INV_SWITCH_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.

Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.185 Specification Item SWS_EthSwt_00296

Trace References:

SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchPortIdx is not valid, EthSwt_GetPortSignalQuality shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in

chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.186 Specification Item SWS_EthSwt_00298

Trace References:

SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369, SRS_ETH_00118

Content:

The function EthSwt_GetPortSignalQuality shall check if the corresponding EthTrcv API EthTrcv_GetPhySignalQuality() of the indexed transceiver driver for the given SwitchPort Idx is available. If this is not the case, the function shall return E_NOT_OK and if development error tracing is activated by EthSwtDevErrorDetect the ETHSWT_E_INV_API shall be raised.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77628: [EthSwt] Behaviour of certain APIs for ports without EthSwtPortTrcvRef (part I)

  Problem description:

  The following APIs have partly a description regarding the handling of ports without EthSwtPortTrcvRef:
  EthSwt_GetPortSignalQuality
  EthSwt_GetPortIdentifier
  EthSwt_SetPortTestMode
  EthTrcv_SetPhyTestMode
  EthSwt_SetPortTxMode
  EthSwt_GetPortCableDiagnosticsResult

  The description should be adjusted and harmonized with related requirements.
  –Last change on issue 77628 comment 2–

  Agreed solution:

  === EthSwt ===
  ~[SWS_EthSwt_91014]EthSwt_GetPortSignalQuality
  Description:
  The function retrieves the signal quality of the link of the indexed Ethernet switch port.
  ~[SWS_EthSwt_00293]
  The function EthSwt_GetPortSignalQuality() shall obtain the signal quality by calling

the function EthTrcv_GetPhySignalQuality() of the referenced Ethernet Transceiver Driver. If the current signal quality is not available, the signal quality shall be set to 0xFF.
-[SWS_EthSwt_00298]as # 77349 introduces a general req

~[SWS_EthSwt_91015]EthSwt_GetPortIdentifier
Return value
E_NOT_OK: organizationally unique identifier of the Ethernet transceiver could not be obtained (i.e. OUI is not available)
Description:
This function retrieves the OUI (24 bit) of the indexed Ethernet switch port.

~[SWS_EthSwt_00299]
The function EthSwt_GetPortIdentifier() shall return the value of the organizationally unique identifier (OUI 24 bit) of the indexed Ethernet switch port that is connected to the indexed Ethernet switch. It shall set the 8 most significant bits of the OUI to 0xFFxxxxxx. If the Ethernet switch port references an Ethernet transceiver, the function shall obtain the OUI by calling the function EthTrcv_GetPhyIdentifier() and set the 8 most significant bits of the OUI to 0x00xxxxxx.

SWS_EthSwt_xxxxx] If neither the Ethernet switch port nor the Ethernet Transceiver Driver can provide an OUI the function EthSwt_GetPortIdentifier() shall return E_NOT_OK.

-[SWS_EthSwt_00304]as # 77349 introduces a general req

~[SWS_EthSwt_91029] EthSwt_SetPortTestMode
Description:
Activates a given test mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00328]
The function EthSwt_SetPortTestMode shall forward the call with the given test mode by calling the function EthTrcv_SetPhyTestMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00333]as # 77349 introduces a general req

~[SWS_EthSwt_91023] EthSwt_SetPortLoopbackMode
Description:
Activates a given test loop-back mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00334]
The function EthSwt_SetPortLoopbackMode() shall forward the call with the given

loop-back mode by calling the function EthTrcv_SetPhyLoopbackMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00339] as # 77349 introduces a general req

~[SWS_EthSwt_91024] EthSwt_SetPortTxMode
Description:
Activates a given transmission mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00340]
The function EthSwt_SetPortTxMode() shall forward the call with the given transmission mode by calling the function EthTrcv_SetPhyTxMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00345] as # 77349 introduces a general req

~[SWS_EthSwt_91025]EthSwt_GetPortCableDiagnosticsResult
Description:
Retrieves the cable diagnostics result of the indexed Ethernet switch port respectively the referenced Ethernet Transceiver Driver.

~[SWS_EthSwt_00346]
The function EthSwt_GetPortCableDiagnosticsResult() shall obtain the cable diagnostics result by calling the function EthTrcv_GetCableDiagnosticsResult() of the referenced Ethernet Transceiver Driver. If no Ethernet transceiver is referenced by the Ethernet switch port and development error detection is not enabled, the cable diagnostic result shall be set to ETHTRCV_CABLEDIAG_OK.
-[SWS_EthSwt_00351] as # 77349 introduces a general req

=== EthTrcv ===
add certain parameter to enable/disable API functions:
+ SWS item ECUC_EthTrcv_xxxx1
Name EthTrcvGetPhySignalQualityApi
Description Enables / Disables EthTrcv_GetPhySignalQuality API

+ SWS item ECUC_EthTrcv_xxxx2
Name EthTrcvGetPhyIdentifierApi
Description Enables / Disables EthTrcv_GetPhyIdentifier API

+ SWS item ECUC_EthTrcv_xxxx3
Name EthTrcvSetPhyTestModeApi
Description Enables / Disables EthTrcv_SetPhyTestMode API

+ SWS item ECUC_EthTrcv_xxxx4
Name EthTrcvSetPhyTxModeApi

Description Enables / Disables EthTrcv_SetPhyTxMode API

+ SWS item ECUC_EthTrcv_xxxx5
Name EthTrcvGetCableDiagnosticsResultApi
Description Enables / Disables EthTrcv_GetCableDiagnosticsResult API

add the following specification to the configuration parameter above:
Multiplicity 1
Type EcucBooleanParamDef
Default value –
Post-Build Variant Value false
Value Configuration
Class Pre-compile time X
All Variants Link time –
Post-build time –
Scope / Dependency scope: local
–Last change on issue 77628 comment 31–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.187 Specification Item SWS_EthSwt_00299

Trace References:

SRS_Eth_00123

Content:

The function EthSwt_GetPortIdentifier() shall return the value of the organizationally unique identifier (OUI 24 bit) of the indexed Ethernet transceiver switch port that is connected to the indexed port. The indexed port is connected to the indexed EthSwt. The value is provided Ethernet switch. It shall set the 8 most significant bits of the OUI to 0x FFxxxxxx. If the Ethernet switch port references an Ethernet transceiver, the function shall obtain the OUI by calling the function EthTrcv_GetPhyIdentifierof the Ethernet Transceiver Driver() and set the 8 most significant bits of the OUI to 0x00xxxxxx.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77628: [EthSwt] Behaviour of certain APIs for ports without EthSwtPortTrcvRef (part I)

Problem description:

The following APIs have partly a description regarding the handling of ports without EthSwtPortTrcvRef:
EthSwt_GetPortSignalQuality
EthSwt_GetPortIdentifier
EthSwt_SetPortTestMode
EthTrcv_SetPhyTestMode
EthSwt_SetPortTxMode
EthSwt_GetPortCableDiagnosticsResult

The description should be adjusted and harmonized with related requirements.
–Last change on issue 77628 comment 2–

Agreed solution:

=== EthSwt ===
~[SWS_EthSwt_91014]EthSwt_GetPortSignalQuality
Description:
The function retrieves the signal quality of the link of the indexed Ethernet switch port.
~[SWS_EthSwt_00293]
The function EthSwt_GetPortSignalQuality() shall obtain the signal quality by calling the function EthTrcv_GetPhySignalQuality() of the referenced Ethernet Transceiver Driver. If the current signal quality is not available, the signal quality shall be set to 0xFF.
-[SWS_EthSwt_00298]as # 77349 introduces a general req

~[SWS_EthSwt_91015]EthSwt_GetPortIdentifier
Return value
E_NOT_OK: organizationally unique identifier of the Ethernet transceiver could not be obtained (i.e. OUI is not available)
Description:
This function retrieves the OUI (24 bit) of the indexed Ethernet switch port.

~[SWS_EthSwt_00299]
The function EthSwt_GetPortIdentifier() shall return the value of the organizationally unique identifier (OUI 24 bit) of the indexed Ethernet switch port that is connected to the indexed Ethernet switch. It shall set the 8 most significant bits of the OUI to 0xFFxxxxxx. If the Ethernet switch port references an Ethernet transceiver, the function shall obtain the OUI by calling the function EthTrcv_GetPhyIdentifier() and set the 8 most significant bits of the OUI to 0x00xxxxxx.

SWS_EthSwt_xxxxx] If neither the Ethernet switch port nor the Ethernet Transceiver Driver can provide an OUI the function EthSwt_GetPortIdentifier() shall return E_NOT_OK.

-[SWS_EthSwt_00304]as # 77349 introduces a general req

~[SWS_EthSwt_91029] EthSwt_SetPortTestMode
Description:
Activates a given test mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00328]
The function EthSwt_SetPortTestMode shall forward the call with the given test mode by calling the function EthTrcv_SetPhyTestMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00333]as # 77349 introduces a general req

~[SWS_EthSwt_91023] EthSwt_SetPortLoopbackMode
Description:
Activates a given test loop-back mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00334]
The function EthSwt_SetPortLoopbackMode() shall forward the call with the given loop-back mode by calling the function EthTrcv_SetPhyLoopbackMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00339] as # 77349 introduces a general req

~[SWS_EthSwt_91024] EthSwt_SetPortTxMode
Description:
Activates a given transmission mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00340]
The function EthSwt_SetPortTxMode() shall forward the call with the given transmission mode by calling the function EthTrcv_SetPhyTxMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00345] as # 77349 introduces a general req

~[SWS_EthSwt_91025]EthSwt_GetPortCableDiagnosticsResult
Description:
Retrieves the cable diagnostics result of the indexed Ethernet switch port respectively the referenced Ethernet Transceiver Driver.

~[SWS_EthSwt_00346]
The function EthSwt_GetPortCableDiagnosticsResult() shall obtain the cable diag-

nostics result by calling the function EthTrcv_GetCableDiagnosticsResult() of the referenced Ethernet Transceiver Driver. If no Ethernet transceiver is referenced by the Ethernet switch port and development error detection is not enabled, the cable diagnostic result shall be set to ETHTRCV_CABLEDIAG_OK.
-[SWS_EthSwt_00351] as # 77349 introduces a general req

=== EthTrcv ===
add certain parameter to enable/disable API functions:
+ SWS item ECUC_EthTrcv_xxxx1
Name EthTrcvGetPhySignalQualityApi
Description Enables / Disables EthTrcv_GetPhySignalQuality API

+ SWS item ECUC_EthTrcv_xxxx2
Name EthTrcvGetPhyIdentifierApi
Description Enables / Disables EthTrcv_GetPhyIdentifier API

+ SWS item ECUC_EthTrcv_xxxx3
Name EthTrcvSetPhyTestModeApi
Description Enables / Disables EthTrcv_SetPhyTestMode API

+ SWS item ECUC_EthTrcv_xxxx4
Name EthTrcvSetPhyTxModeApi
Description Enables / Disables EthTrcv_SetPhyTxMode API

+ SWS item ECUC_EthTrcv_xxxx5
Name EthTrcvGetCableDiagnosticsResultApi
Description Enables / Disables EthTrcv_GetCableDiagnosticsResult API

add the following specification to the configuration parameter above:
Multiplicity 1
Type EcucBooleanParamDef
Default value –
Post-Build Variant Value false
Value Configuration
Class Pre-compile time X
All Variants Link time –
Post-build time –
Scope / Dependency scope: local
–Last change on issue 77628 comment 31–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.188 Specification Item SWS_EthSwt_00300

Trace References:

SRS_BSW_00406

Content:

If development error detection is enabled: the function EthSwt_GetPortIdentifier shall check that the service EthSwt_SwitchInit was previously called. If the check fails, the function shall raise the development error ETHSWT_E_NOT_INITIALIZED and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.189 Specification Item SWS_EthSwt_00301

Trace References:

SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchIdx is not valid,EthSwt_GetPortIdentifier shall raise the development error ETHSWT_E_INV_SWITCH_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

  For Error Detection following general rules apply:
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with in-

put parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.190 Specification Item SWS_EthSwt_00302

Trace References:

SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchPortIdx is not valid,EthSwt_GetPortIdentifier shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

  For Error Detection following general rules apply:
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with

input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.191 Specification Item SWS_EthSwt_00304

Trace References:

SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369, SRS_ETH_00118

Content:

The function EthSwt_GetPortIdentifier shall check if the corresponding EthTrcv API EthTrcv_GetPhyIdentifier() of the indexed transceiver driver for the given SwitchPortIdx is available. If this is not the case, the function shall return E_NOT_OK and if development error tracing is activated by EthSwtDevErrorDetect the ETHSWT_E_INV_API shall be raised.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77628: [EthSwt] Behaviour of certain APIs for ports without EthSwtPortTrcvRef (part I)

  Problem description:

  The following APIs have partly a description regarding the handling of ports without EthSwtPortTrcvRef:
  EthSwt_GetPortSignalQuality
  EthSwt_GetPortIdentifier
  EthSwt_SetPortTestMode
  EthTrcv_SetPhyTestMode
  EthSwt_SetPortTxMode
  EthSwt_GetPortCableDiagnosticsResult

  The description should be adjusted and harmonized with related requirements.
  –Last change on issue 77628 comment 2–

  Agreed solution:

  === EthSwt ===
  ~[SWS_EthSwt_91014]EthSwt_GetPortSignalQuality
  Description:
  The function retrieves the signal quality of the link of the indexed Ethernet switch port.
  ~[SWS_EthSwt_00293]
  The function EthSwt_GetPortSignalQuality() shall obtain the signal quality by calling the function EthTrcv_GetPhySignalQuality() of the referenced Ethernet Transceiver Driver. If the current signal quality is not available, the signal quality shall be set to 0xFF.
  -[SWS_EthSwt_00298]as # 77349 introduces a general req

  ~[SWS_EthSwt_91015]EthSwt_GetPortIdentifier
  Return value
  E_NOT_OK: organizationally unique identifier of the Ethernet transceiver could not be obtained (i.e. OUI is not available)
  Description:
  This function retrieves the OUI (24 bit) of the indexed Ethernet switch port.

  ~[SWS_EthSwt_00299]
  The function EthSwt_GetPortIdentifier() shall return the value of the organizationally unique identifier (OUI 24 bit) of the indexed Ethernet switch port that is connected to the indexed Ethernet switch. It shall set the 8 most significant bits of the OUI

to 0xFFxxxxxx. If the Ethernet switch port references an Ethernet transceiver, the function shall obtain the OUI by calling the function EthTrcv_GetPhyIdentifier() and set the 8 most significant bits of the OUI to 0x00xxxxxx.

SWS_EthSwt_xxxxx] If neither the Ethernet switch port nor the Ethernet Transceiver Driver can provide an OUI the function EthSwt_GetPortIdentifier() shall return E_NOT_OK.

-[SWS_EthSwt_00304]as # 77349 introduces a general req

~[SWS_EthSwt_91029] EthSwt_SetPortTestMode
Description:
Activates a given test mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00328]
The function EthSwt_SetPortTestMode shall forward the call with the given test mode by calling the function EthTrcv_SetPhyTestMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00333]as # 77349 introduces a general req

~[SWS_EthSwt_91023] EthSwt_SetPortLoopbackMode
Description:
Activates a given test loop-back mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00334]
The function EthSwt_SetPortLoopbackMode() shall forward the call with the given loop-back mode by calling the function EthTrcv_SetPhyLoopbackMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00339] as # 77349 introduces a general req

~[SWS_EthSwt_91024] EthSwt_SetPortTxMode
Description:
Activates a given transmission mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00340]
The function EthSwt_SetPortTxMode() shall forward the call with the given transmission mode by calling the function EthTrcv_SetPhyTxMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00345] as # 77349 introduces a general req

~[SWS_EthSwt_91025]EthSwt_GetPortCableDiagnosticsResult
Description:
Retrieves the cable diagnostics result of the indexed Ethernet switch port respec-

tively the referenced Ethernet Transceiver Driver.

~[SWS_EthSwt_00346]
The function EthSwt_GetPortCableDiagnosticsResult() shall obtain the cable diagnostics result by calling the function EthTrcv_GetCableDiagnosticsResult() of the referenced Ethernet Transceiver Driver. If no Ethernet transceiver is referenced by the Ethernet switch port and development error detection is not enabled, the cable diagnostic result shall be set to ETHTRCV_CABLEDIAG_OK.
-[SWS_EthSwt_00351] as # 77349 introduces a general req

=== EthTrcv ===
add certain parameter to enable/disable API functions:
+ SWS item ECUC_EthTrcv_xxxx1
Name EthTrcvGetPhySignalQualityApi
Description Enables / Disables EthTrcv_GetPhySignalQuality API

+ SWS item ECUC_EthTrcv_xxxx2
Name EthTrcvGetPhyIdentifierApi
Description Enables / Disables EthTrcv_GetPhyIdentifier API

+ SWS item ECUC_EthTrcv_xxxx3
Name EthTrcvSetPhyTestModeApi
Description Enables / Disables EthTrcv_SetPhyTestMode API

+ SWS item ECUC_EthTrcv_xxxx4
Name EthTrcvSetPhyTxModeApi
Description Enables / Disables EthTrcv_SetPhyTxMode API

+ SWS item ECUC_EthTrcv_xxxx5
Name EthTrcvGetCableDiagnosticsResultApi
Description Enables / Disables EthTrcv_GetCableDiagnosticsResult API

add the following specification to the configuration parameter above:
Multiplicity 1
Type EcucBooleanParamDef
Default value –
Post-Build Variant Value false
Value Configuration
Class Pre-compile time X
All Variants Link time –
Post-build time –
Scope / Dependency scope: local
–Last change on issue 77628 comment 31–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.192 Specification Item SWS_EthSwt_00306

Trace References:

SRS_BSW_00406

Content:

If development error detection is enabled: the function EthSwt_GetSwitchIdentifier shall check that the service EthSwt_SwitchInit was previously called. If the check fails, the function shall raise the development error ETHSWT_E_NOT_INITIALIZED and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:

SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.193 Specification Item SWS_EthSwt_00307

Trace References:

SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchIdx is not valid,EthSwt_GetSwitchIdentifier shall raise the development error ETHSWT_E_INV_SWITCH_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

  For Error Detection following general rules apply:

[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.194 Specification Item SWS_EthSwt_00310

Trace References:

SRS_BSW_00406

Content:

If development error detection is enabled: the function EthSwt_WritePortMirrorConfiguration shall check that the service EthSwt_SwitchInit was previously called. If the check fails, the function shall raise the development error ETHSWT_E_NOT_INITIALIZED and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

  For Error Detection following general rules apply:
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with in-

put parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.195 Specification Item SWS_EthSwt_00311

Trace References:

SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchIdx is not valid, EthSwt_WritePortMirrorConfiguration shall raise the development error ETHSWT_E_INV_SWITCH_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

  For Error Detection following general rules apply:
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with

input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.196 Specification Item SWS_EthSwt_00314

Trace References:

SRS_BSW_00406

Content:

If development error detection is enabled: the function EthSwt_ReadPortMirrorConfiguration shall check that the service EthSwt_SwitchInit was previously called. If the check fails, the function shall raise the development error ETHSWT_E_NOT_INITIALIZED and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

  For Error Detection following general rules apply:
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
  [SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with

input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.197 Specification Item SWS_EthSwt_00315

Trace References:

SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchIdx is not valid, EthSwt_ReadPortMirrorConfiguration shall raise the development error ETHSWT_E_INV_SWITCH_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

[SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

In the above requirement -
1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.

[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.198 Specification Item SWS_EthSwt_00316

Trace References:

SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchPortIdx is not valid, EthSwt_ReadPortMirrorConfiguration shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -

1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevEr-

rorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.199 Specification Item SWS_EthSwt_00319

Trace References:

SRS_BSW_00406

Content:

If development error detection is enabled: the function EthSwt_GetPortMirrorState shall check that the service EthSwt_SwitchInit was previously called. If the check fails, the function shall raise the development error ETHSWT_E_NOT_INITIALIZED and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not de-

fault errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.200 Specification Item SWS_EthSwt_00320

Trace References:

SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchIdx is not valid, EthSwt_GetPortMirrorState shall raise the development error ETHSWT_E_INV_SWITCH_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the

development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.201 Specification Item SWS_EthSwt_00321

Trace References:

SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchPortIdx is not valid, EthSwt_GetPortMirrorState shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.202 Specification Item SWS_EthSwt_00324

Trace References:

SRS_BSW_00406

Content:

If development error detection is enabled: the function EthSwt_SetPortMirrorState shall check that the service EthSwt_SwitchInit was previously called. If the check fails, the function shall raise the development error ETHSWT_E_NOT_INITIALIZED and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:

SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.203 Specification Item SWS_EthSwt_00325

Trace References:

SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchIdx is not valid, EthSwt_SetPortMirrorState shall raise the development error ETHSWT_E_INV_SWITCH_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

  For Error Detection following general rules apply:

[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.204 Specification Item SWS_EthSwt_00326

Trace References:

SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchPortIdx is not valid, EthSwt_SetPortMirrorState shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

  For Error Detection following general rules apply:
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails,

the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.205 Specification Item SWS_EthSwt_00328

Trace References:

SRS_Eth_00123

Content:

The function EthSwt_SetPortTestMode shall activate the indexed Ethernet transceiver that is connected to the indexed port to the forward the call with the given test mode . The indexed port is connected to the indexed EthSwt. The value is provided by calling the function EthTrcv_SetPhyTestModeof the () of the referenced Ethernet Transceiver Driver.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77628: [EthSwt] Behaviour of certain APIs for ports without EthSwtPortTrcvRef (part I)

  Problem description:

  The following APIs have partly a description regarding the handling of ports without EthSwtPortTrcvRef:
  EthSwt_GetPortSignalQuality
  EthSwt_GetPortIdentifier
  EthSwt_SetPortTestMode
  EthTrcv_SetPhyTestMode
  EthSwt_SetPortTxMode
  EthSwt_GetPortCableDiagnosticsResult

  The description should be adjusted and harmonized with related requirements.
  –Last change on issue 77628 comment 2–

  Agreed solution:

  === EthSwt ===
  ~[SWS_EthSwt_91014]EthSwt_GetPortSignalQuality
  Description:
  The function retrieves the signal quality of the link of the indexed Ethernet switch port.
  ~[SWS_EthSwt_00293]
  The function EthSwt_GetPortSignalQuality() shall obtain the signal quality by calling the function EthTrcv_GetPhySignalQuality() of the referenced Ethernet Transceiver Driver. If the current signal quality is not available, the signal quality shall be set to 0xFF.
  -[SWS_EthSwt_00298]as # 77349 introduces a general req

  ~[SWS_EthSwt_91015]EthSwt_GetPortIdentifier
  Return value
  E_NOT_OK: organizationally unique identifier of the Ethernet transceiver could not be obtained (i.e. OUI is not available)
  Description:
  This function retrieves the OUI (24 bit) of the indexed Ethernet switch port.

  ~[SWS_EthSwt_00299]

The function EthSwt_GetPortIdentifier() shall return the value of the organizationally unique identifier (OUI 24 bit) of the indexed Ethernet switch port that is connected to the indexed Ethernet switch. It shall set the 8 most significant bits of the OUI to 0xFFxxxxxx. If the Ethernet switch port references an Ethernet transceiver, the function shall obtain the OUI by calling the function EthTrcv_GetPhyIdentifier() and set the 8 most significant bits of the OUI to 0x00xxxxxx.

SWS_EthSwt_xxxxx] If neither the Ethernet switch port nor the Ethernet Transceiver Driver can provide an OUI the function EthSwt_GetPortIdentifier() shall return E_NOT_OK.

-[SWS_EthSwt_00304]as # 77349 introduces a general req

~[SWS_EthSwt_91029] EthSwt_SetPortTestMode
Description:
Activates a given test mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00328]
The function EthSwt_SetPortTestMode shall forward the call with the given test mode by calling the function EthTrcv_SetPhyTestMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00333]as # 77349 introduces a general req

~[SWS_EthSwt_91023] EthSwt_SetPortLoopbackMode
Description:
Activates a given test loop-back mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00334]
The function EthSwt_SetPortLoopbackMode() shall forward the call with the given loop-back mode by calling the function EthTrcv_SetPhyLoopbackMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00339] as # 77349 introduces a general req

~[SWS_EthSwt_91024] EthSwt_SetPortTxMode
Description:
Activates a given transmission mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00340]
The function EthSwt_SetPortTxMode() shall forward the call with the given transmission mode by calling the function EthTrcv_SetPhyTxMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00345] as # 77349 introduces a general req

~[SWS_EthSwt_91025]EthSwt_GetPortCableDiagnosticsResult
Description:
Retrieves the cable diagnostics result of the indexed Ethernet switch port respectively the referenced Ethernet Transceiver Driver.

~[SWS_EthSwt_00346]
The function EthSwt_GetPortCableDiagnosticsResult() shall obtain the cable diagnostics result by calling the function EthTrcv_GetCableDiagnosticsResult() of the referenced Ethernet Transceiver Driver. If no Ethernet transceiver is referenced by the Ethernet switch port and development error detection is not enabled, the cable diagnostic result shall be set to ETHTRCV_CABLEDIAG_OK.
-[SWS_EthSwt_00351] as # 77349 introduces a general req

=== EthTrcv ===
add certain parameter to enable/disable API functions:
+ SWS item ECUC_EthTrcv_xxxx1
Name EthTrcvGetPhySignalQualityApi
Description Enables / Disables EthTrcv_GetPhySignalQuality API

+ SWS item ECUC_EthTrcv_xxxx2
Name EthTrcvGetPhyIdentifierApi
Description Enables / Disables EthTrcv_GetPhyIdentifier API

+ SWS item ECUC_EthTrcv_xxxx3
Name EthTrcvSetPhyTestModeApi
Description Enables / Disables EthTrcv_SetPhyTestMode API

+ SWS item ECUC_EthTrcv_xxxx4
Name EthTrcvSetPhyTxModeApi
Description Enables / Disables EthTrcv_SetPhyTxMode API

+ SWS item ECUC_EthTrcv_xxxx5
Name EthTrcvGetCableDiagnosticsResultApi
Description Enables / Disables EthTrcv_GetCableDiagnosticsResult API

add the following specification to the configuration parameter above:
Multiplicity 1
Type EcucBooleanParamDef
Default value –
Post-Build Variant Value false
Value Configuration
Class Pre-compile time X
All Variants Link time –

Post-build time –
Scope / Dependency scope: local
–Last change on issue 77628 comment 31–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.206 Specification Item SWS_EthSwt_00329

Trace References:

SRS_BSW_00406

Content:

If development error detection is enabled: the function EthSwt_SetPortTestMode shall check that the service EthSwt_SwitchInit was previously called. If the check fails, the function shall raise the development error ETHSWT_E_NOT_INITIALIZED and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.207 Specification Item SWS_EthSwt_00330

Trace References:

SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchIdx is not valid,EthSwt_SetPortTestMode shall raise the development error ETHSWT_E_INV_SWITCH_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001

rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.208 Specification Item SWS_EthSwt_00331

Trace References:

SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchPortIdx is not valid,EthSwt_SetPortTestMode shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

  For Error Detection following general rules apply:

[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.209 Specification Item SWS_EthSwt_00333

Trace References:

SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369, SRS_ETH_00118

Content:

The function EthSwt_SetPortTestMode shall check if the corresponding EthTrcv API EthTrcv_GetPhyIdentifier() of the indexed transceiver driver for the given SwitchPortIdx is available. If this is not the case, the function shall return E_NOT_OK and if development error tracing is activated by EthSwtDevErrorDetect the ETHSWT_E_INV_API shall be raised.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77628: [EthSwt] Behaviour of certain APIs for ports without EthSwtPortTrcvRef (part I)

  Problem description:

  The following APIs have partly a description regarding the handling of ports without EthSwtPortTrcvRef:
  EthSwt_GetPortSignalQuality
  EthSwt_GetPortIdentifier
  EthSwt_SetPortTestMode
  EthTrcv_SetPhyTestMode
  EthSwt_SetPortTxMode
  EthSwt_GetPortCableDiagnosticsResult

  The description should be adjusted and harmonized with related requirements.
  –Last change on issue 77628 comment 2–

  Agreed solution:

  === EthSwt ===
  ~[SWS_EthSwt_91014]EthSwt_GetPortSignalQuality
  Description:
  The function retrieves the signal quality of the link of the indexed Ethernet switch port.
  ~[SWS_EthSwt_00293]
  The function EthSwt_GetPortSignalQuality() shall obtain the signal quality by calling the function EthTrcv_GetPhySignalQuality() of the referenced Ethernet Transceiver Driver. If the current signal quality is not available, the signal quality shall be set to 0xFF.
  -[SWS_EthSwt_00298]as # 77349 introduces a general req

  ~[SWS_EthSwt_91015]EthSwt_GetPortIdentifier
  Return value
  E_NOT_OK: organizationally unique identifier of the Ethernet transceiver could not

be obtained (i.e. OUI is not available)
Description:
This function retrieves the OUI (24 bit) of the indexed Ethernet switch port.

~[SWS_EthSwt_00299]
The function EthSwt_GetPortIdentifier() shall return the value of the organizationally unique identifier (OUI 24 bit) of the indexed Ethernet switch port that is connected to the indexed Ethernet switch. It shall set the 8 most significant bits of the OUI to 0xFFxxxxxx. If the Ethernet switch port references an Ethernet transceiver, the function shall obtain the OUI by calling the function EthTrcv_GetPhyIdentifier() and set the 8 most significant bits of the OUI to 0x00xxxxxx.

SWS_EthSwt_xxxxx] If neither the Ethernet switch port nor the Ethernet Transceiver Driver can provide an OUI the function EthSwt_GetPortIdentifier() shall return E_NOT_OK.

-[SWS_EthSwt_00304]as # 77349 introduces a general req

~[SWS_EthSwt_91029] EthSwt_SetPortTestMode
Description:
Activates a given test mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00328]
The function EthSwt_SetPortTestMode shall forward the call with the given test mode by calling the function EthTrcv_SetPhyTestMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00333]as # 77349 introduces a general req

~[SWS_EthSwt_91023] EthSwt_SetPortLoopbackMode
Description:
Activates a given test loop-back mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00334]
The function EthSwt_SetPortLoopbackMode() shall forward the call with the given loop-back mode by calling the function EthTrcv_SetPhyLoopbackMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00339] as # 77349 introduces a general req

~[SWS_EthSwt_91024] EthSwt_SetPortTxMode
Description:
Activates a given transmission mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00340]

The function EthSwt_SetPortTxMode() shall forward the call with the given transmission mode by calling the function EthTrcv_SetPhyTxMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00345] as # 77349 introduces a general req

~[SWS_EthSwt_91025]EthSwt_GetPortCableDiagnosticsResult
Description:
Retrieves the cable diagnostics result of the indexed Ethernet switch port respectively the referenced Ethernet Transceiver Driver.

~[SWS_EthSwt_00346]
The function EthSwt_GetPortCableDiagnosticsResult() shall obtain the cable diagnostics result by calling the function EthTrcv_GetCableDiagnosticsResult() of the referenced Ethernet Transceiver Driver. If no Ethernet transceiver is referenced by the Ethernet switch port and development error detection is not enabled, the cable diagnostic result shall be set to ETHTRCV_CABLEDIAG_OK.
-[SWS_EthSwt_00351] as # 77349 introduces a general req

=== EthTrcv ===
add certain parameter to enable/disable API functions:
+ SWS item ECUC_EthTrcv_xxxx1
Name EthTrcvGetPhySignalQualityApi
Description Enables / Disables EthTrcv_GetPhySignalQuality API

+ SWS item ECUC_EthTrcv_xxxx2
Name EthTrcvGetPhyIdentifierApi
Description Enables / Disables EthTrcv_GetPhyIdentifier API

+ SWS item ECUC_EthTrcv_xxxx3
Name EthTrcvSetPhyTestModeApi
Description Enables / Disables EthTrcv_SetPhyTestMode API

+ SWS item ECUC_EthTrcv_xxxx4
Name EthTrcvSetPhyTxModeApi
Description Enables / Disables EthTrcv_SetPhyTxMode API

+ SWS item ECUC_EthTrcv_xxxx5
Name EthTrcvGetCableDiagnosticsResultApi
Description Enables / Disables EthTrcv_GetCableDiagnosticsResult API

add the following specification to the configuration parameter above:
Multiplicity 1
Type EcucBooleanParamDef

Default value –
Post-Build Variant Value false
Value Configuration
Class Pre-compile time X
All Variants Link time –
Post-build time –
Scope / Dependency scope: local
–Last change on issue 77628 comment 31–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.210 Specification Item SWS_EthSwt_00334

Trace References:

SRS_Eth_00123

Content:

The function EthSwt_SetPortLoopbackModeshall activate the indexed Ethernet transceiver that is connected to the indexed port to the given test mode . The indexed port is connected to the indexed EthSwt. The value is provided () shall forward the call with the given loop-back mode by calling the function EthTrcv_SetPhyTestModeof the LoopbackMode() of the referenced Ethernet Transceiver Driver.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76891: [EthSwt] In requirement SWS_EthSwt_00334 EthTrcv_SetPhyTestMode should be replaced with EthTrcv_SetPhyLoopbackMode

  Problem description:

  In chapter 8.3.47 EthSwt_SetPortLoopbackMode, in the requirement SWS_EthSwt_00334 text EthTrcv_SetPhyTestMode should be replaced with EthTrcv_SetPhyLoopbackMode

  Agreed solution:

  In chapter 8.3.47 EthSwt_SetPortLoopbackMode, in the requirement text SWS_EthSwt_00334 EthTrcv_SetPhyTestMode should be replaced with EthTrcv_SetPhyLoopbackMode

  BW-C-Level:

Application	Specification	Bus
1	4	1

- RfC #77628: [EthSwt] Behaviour of certain APIs for ports without EthSwtPortTrcvRef (part I)

  Problem description:

  The following APIs have partly a description regarding the handling of ports without EthSwtPortTrcvRef:
  EthSwt_GetPortSignalQuality
  EthSwt_GetPortIdentifier
  EthSwt_SetPortTestMode
  EthTrcv_SetPhyTestMode
  EthSwt_SetPortTxMode
  EthSwt_GetPortCableDiagnosticsResult

  The description should be adjusted and harmonized with related requirements.
  –Last change on issue 77628 comment 2–

  Agreed solution:

  === EthSwt ===
  ~[SWS_EthSwt_91014]EthSwt_GetPortSignalQuality
  Description:
  The function retrieves the signal quality of the link of the indexed Ethernet switch port.
  ~[SWS_EthSwt_00293]
  The function EthSwt_GetPortSignalQuality() shall obtain the signal quality by calling the function EthTrcv_GetPhySignalQuality() of the referenced Ethernet Transceiver Driver. If the current signal quality is not available, the signal quality shall be set to 0xFF.
  -[SWS_EthSwt_00298]as # 77349 introduces a general req

  ~[SWS_EthSwt_91015]EthSwt_GetPortIdentifier
  Return value
  E_NOT_OK: organizationally unique identifier of the Ethernet transceiver could not be obtained (i.e. OUI is not available)
  Description:
  This function retrieves the OUI (24 bit) of the indexed Ethernet switch port.

  ~[SWS_EthSwt_00299]
  The function EthSwt_GetPortIdentifier() shall return the value of the organizationally unique identifier (OUI 24 bit) of the indexed Ethernet switch port that is connected

to the indexed Ethernet switch. It shall set the 8 most significant bits of the OUI to 0xFFxxxxxx. If the Ethernet switch port references an Ethernet transceiver, the function shall obtain the OUI by calling the function EthTrcv_GetPhyIdentifier() and set the 8 most significant bits of the OUI to 0x00xxxxxx.

SWS_EthSwt_xxxxx] If neither the Ethernet switch port nor the Ethernet Transceiver Driver can provide an OUI the function EthSwt_GetPortIdentifier() shall return E_NOT_OK.

-[SWS_EthSwt_00304]as # 77349 introduces a general req

~[SWS_EthSwt_91029] EthSwt_SetPortTestMode
Description:
Activates a given test mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00328]
The function EthSwt_SetPortTestMode shall forward the call with the given test mode by calling the function EthTrcv_SetPhyTestMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00333]as # 77349 introduces a general req

~[SWS_EthSwt_91023] EthSwt_SetPortLoopbackMode
Description:
Activates a given test loop-back mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00334]
The function EthSwt_SetPortLoopbackMode() shall forward the call with the given loop-back mode by calling the function EthTrcv_SetPhyLoopbackMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00339] as # 77349 introduces a general req

~[SWS_EthSwt_91024] EthSwt_SetPortTxMode
Description:
Activates a given transmission mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00340]
The function EthSwt_SetPortTxMode() shall forward the call with the given transmission mode by calling the function EthTrcv_SetPhyTxMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00345] as # 77349 introduces a general req

~[SWS_EthSwt_91025]EthSwt_GetPortCableDiagnosticsResult
Description:

Retrieves the cable diagnostics result of the indexed Ethernet switch port respectively the referenced Ethernet Transceiver Driver.

~[SWS_EthSwt_00346]
The function EthSwt_GetPortCableDiagnosticsResult() shall obtain the cable diagnostics result by calling the function EthTrcv_GetCableDiagnosticsResult() of the referenced Ethernet Transceiver Driver. If no Ethernet transceiver is referenced by the Ethernet switch port and development error detection is not enabled, the cable diagnostic result shall be set to ETHTRCV_CABLEDIAG_OK.
-[SWS_EthSwt_00351] as # 77349 introduces a general req

=== EthTrcv ===
add certain parameter to enable/disable API functions:
+ SWS item ECUC_EthTrcv_xxxx1
Name EthTrcvGetPhySignalQualityApi
Description Enables / Disables EthTrcv_GetPhySignalQuality API

+ SWS item ECUC_EthTrcv_xxxx2
Name EthTrcvGetPhyIdentifierApi
Description Enables / Disables EthTrcv_GetPhyIdentifier API

+ SWS item ECUC_EthTrcv_xxxx3
Name EthTrcvSetPhyTestModeApi
Description Enables / Disables EthTrcv_SetPhyTestMode API

+ SWS item ECUC_EthTrcv_xxxx4
Name EthTrcvSetPhyTxModeApi
Description Enables / Disables EthTrcv_SetPhyTxMode API

+ SWS item ECUC_EthTrcv_xxxx5
Name EthTrcvGetCableDiagnosticsResultApi
Description Enables / Disables EthTrcv_GetCableDiagnosticsResult API

add the following specification to the configuration parameter above:
Multiplicity 1
Type EcucBooleanParamDef
Default value –
Post-Build Variant Value false
Value Configuration
Class Pre-compile time X
All Variants Link time –
Post-build time –

Scope / Dependency scope: local
–Last change on issue 77628 comment 31–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.211 Specification Item SWS_EthSwt_00335

Trace References:

SRS_BSW_00406

Content:

If development error detection is enabled: the function EthSwt_SetPortLoopbackMode shall check that the service EthSwt_SwitchInit was previously called. If the check fails, the function shall raise the development error ETHSWT_E_NOT_INITIALIZED and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.212 Specification Item SWS_EthSwt_00336

Trace References:

SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchIdx is not valid, EthSwt_SetPortLoopbackMode shall raise the development error ETHSWT_E_INV_SWITCH_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001

rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.213 Specification Item SWS_EthSwt_00337

Trace References:

SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchPortIdx is not valid, EthSwt_SetPortLoopbackMode shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

  For Error Detection following general rules apply:

[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.214 Specification Item SWS_EthSwt_00339

Trace References:

SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369, SRS_ETH_00118

Content:

The function EthSwt_SetPortLoopbackMode shall check if the corresponding EthTrcv API EthTrcv_SetPhyLoopbackMode() of the indexed transceiver driver for the given SwitchPort Idx is available. If this is not the case, the function shall return E_NOT_OK and if development error tracing is activated by EthSwtDevErrorDetect the ETHSWT_E_INV_API shall be raised.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77628: [EthSwt] Behaviour of certain APIs for ports without EthSwtPortTrcvRef (part I)

  Problem description:

  The following APIs have partly a description regarding the handling of ports without EthSwtPortTrcvRef:
  EthSwt_GetPortSignalQuality
  EthSwt_GetPortIdentifier
  EthSwt_SetPortTestMode
  EthTrcv_SetPhyTestMode
  EthSwt_SetPortTxMode
  EthSwt_GetPortCableDiagnosticsResult

  The description should be adjusted and harmonized with related requirements.
  –Last change on issue 77628 comment 2–

  Agreed solution:

  === EthSwt ===
  ~[SWS_EthSwt_91014]EthSwt_GetPortSignalQuality
  Description:
  The function retrieves the signal quality of the link of the indexed Ethernet switch port.
  ~[SWS_EthSwt_00293]
  The function EthSwt_GetPortSignalQuality() shall obtain the signal quality by calling the function EthTrcv_GetPhySignalQuality() of the referenced Ethernet Transceiver Driver. If the current signal quality is not available, the signal quality shall be set to 0xFF.
  -[SWS_EthSwt_00298]as # 77349 introduces a general req

  ~[SWS_EthSwt_91015]EthSwt_GetPortIdentifier
  Return value
  E_NOT_OK: organizationally unique identifier of the Ethernet transceiver could not

be obtained (i.e. OUI is not available)
Description:
This function retrieves the OUI (24 bit) of the indexed Ethernet switch port.

~[SWS_EthSwt_00299]
The function EthSwt_GetPortIdentifier() shall return the value of the organizationally unique identifier (OUI 24 bit) of the indexed Ethernet switch port that is connected to the indexed Ethernet switch. It shall set the 8 most significant bits of the OUI to 0xFFxxxxxx. If the Ethernet switch port references an Ethernet transceiver, the function shall obtain the OUI by calling the function EthTrcv_GetPhyIdentifier() and set the 8 most significant bits of the OUI to 0x00xxxxxx.

SWS_EthSwt_xxxxx] If neither the Ethernet switch port nor the Ethernet Transceiver Driver can provide an OUI the function EthSwt_GetPortIdentifier() shall return E_NOT_OK.

-[SWS_EthSwt_00304]as # 77349 introduces a general req

~[SWS_EthSwt_91029] EthSwt_SetPortTestMode
Description:
Activates a given test mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00328]
The function EthSwt_SetPortTestMode shall forward the call with the given test mode by calling the function EthTrcv_SetPhyTestMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00333]as # 77349 introduces a general req

~[SWS_EthSwt_91023] EthSwt_SetPortLoopbackMode
Description:
Activates a given test loop-back mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00334]
The function EthSwt_SetPortLoopbackMode() shall forward the call with the given loop-back mode by calling the function EthTrcv_SetPhyLoopbackMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00339] as # 77349 introduces a general req

~[SWS_EthSwt_91024] EthSwt_SetPortTxMode
Description:
Activates a given transmission mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00340]

The function EthSwt_SetPortTxMode() shall forward the call with the given transmission mode by calling the function EthTrcv_SetPhyTxMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00345] as # 77349 introduces a general req

~[SWS_EthSwt_91025]EthSwt_GetPortCableDiagnosticsResult
Description:
Retrieves the cable diagnostics result of the indexed Ethernet switch port respectively the referenced Ethernet Transceiver Driver.

~[SWS_EthSwt_00346]
The function EthSwt_GetPortCableDiagnosticsResult() shall obtain the cable diagnostics result by calling the function EthTrcv_GetCableDiagnosticsResult() of the referenced Ethernet Transceiver Driver. If no Ethernet transceiver is referenced by the Ethernet switch port and development error detection is not enabled, the cable diagnostic result shall be set to ETHTRCV_CABLEDIAG_OK.
-[SWS_EthSwt_00351] as # 77349 introduces a general req

=== EthTrcv ===
add certain parameter to enable/disable API functions:
+ SWS item ECUC_EthTrcv_xxxx1
Name EthTrcvGetPhySignalQualityApi
Description Enables / Disables EthTrcv_GetPhySignalQuality API

+ SWS item ECUC_EthTrcv_xxxx2
Name EthTrcvGetPhyIdentifierApi
Description Enables / Disables EthTrcv_GetPhyIdentifier API

+ SWS item ECUC_EthTrcv_xxxx3
Name EthTrcvSetPhyTestModeApi
Description Enables / Disables EthTrcv_SetPhyTestMode API

+ SWS item ECUC_EthTrcv_xxxx4
Name EthTrcvSetPhyTxModeApi
Description Enables / Disables EthTrcv_SetPhyTxMode API

+ SWS item ECUC_EthTrcv_xxxx5
Name EthTrcvGetCableDiagnosticsResultApi
Description Enables / Disables EthTrcv_GetCableDiagnosticsResult API

add the following specification to the configuration parameter above:
Multiplicity 1
Type EcucBooleanParamDef

Default value –
Post-Build Variant Value false
Value Configuration
Class Pre-compile time X
All Variants Link time –
Post-build time –
Scope / Dependency scope: local
–Last change on issue 77628 comment 31–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.215 Specification Item SWS_EthSwt_00340

Trace References:

SRS_Eth_00123

Content:

The function EthSwt_SetPortTxModeshall activate the indexed Ethernet transceiver that is connected to the indexed port to the given test mode . The indexed port is connected to the indexed EthSwt. The value is provided () shall forward the call with the given transmission mode by calling the function EthTrcv_SetPhyTxModeof the () of the referenced Ethernet Transceiver Driver.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77628: [EthSwt] Behaviour of certain APIs for ports without EthSwtPortTrcvRef (part I)

  Problem description:

  The following APIs have partly a description regarding the handling of ports without EthSwtPortTrcvRef:
  EthSwt_GetPortSignalQuality
  EthSwt_GetPortIdentifier
  EthSwt_SetPortTestMode
  EthTrcv_SetPhyTestMode
  EthSwt_SetPortTxMode
  EthSwt_GetPortCableDiagnosticsResult

The description should be adjusted and harmonized with related requirements.
–Last change on issue 77628 comment 2–

Agreed solution:

=== EthSwt ===
~[SWS_EthSwt_91014]EthSwt_GetPortSignalQuality
Description:
The function retrieves the signal quality of the link of the indexed Ethernet switch port.
~[SWS_EthSwt_00293]
The function EthSwt_GetPortSignalQuality() shall obtain the signal quality by calling the function EthTrcv_GetPhySignalQuality() of the referenced Ethernet Transceiver Driver. If the current signal quality is not available, the signal quality shall be set to 0xFF.
-[SWS_EthSwt_00298]as # 77349 introduces a general req

~[SWS_EthSwt_91015]EthSwt_GetPortIdentifier
Return value
E_NOT_OK: organizationally unique identifier of the Ethernet transceiver could not be obtained (i.e. OUI is not available)
Description:
This function retrieves the OUI (24 bit) of the indexed Ethernet switch port.

~[SWS_EthSwt_00299]
The function EthSwt_GetPortIdentifier() shall return the value of the organizationally unique identifier (OUI 24 bit) of the indexed Ethernet switch port that is connected to the indexed Ethernet switch. It shall set the 8 most significant bits of the OUI to 0xFFxxxxxx. If the Ethernet switch port references an Ethernet transceiver, the function shall obtain the OUI by calling the function EthTrcv_GetPhyIdentifier() and set the 8 most significant bits of the OUI to 0x00xxxxxx.

SWS_EthSwt_xxxxx] If neither the Ethernet switch port nor the Ethernet Transceiver Driver can provide an OUI the function EthSwt_GetPortIdentifier() shall return E_NOT_OK.

-[SWS_EthSwt_00304]as # 77349 introduces a general req

~[SWS_EthSwt_91029] EthSwt_SetPortTestMode
Description:
Activates a given test mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00328]

The function EthSwt_SetPortTestMode shall forward the call with the given test mode by calling the function EthTrcv_SetPhyTestMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00333]as # 77349 introduces a general req

~[SWS_EthSwt_91023] EthSwt_SetPortLoopbackMode
Description:
Activates a given test loop-back mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00334]
The function EthSwt_SetPortLoopbackMode() shall forward the call with the given loop-back mode by calling the function EthTrcv_SetPhyLoopbackMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00339] as # 77349 introduces a general req

~[SWS_EthSwt_91024] EthSwt_SetPortTxMode
Description:
Activates a given transmission mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00340]
The function EthSwt_SetPortTxMode() shall forward the call with the given transmission mode by calling the function EthTrcv_SetPhyTxMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00345] as # 77349 introduces a general req

~[SWS_EthSwt_91025]EthSwt_GetPortCableDiagnosticsResult
Description:
Retrieves the cable diagnostics result of the indexed Ethernet switch port respectively the referenced Ethernet Transceiver Driver.

~[SWS_EthSwt_00346]
The function EthSwt_GetPortCableDiagnosticsResult() shall obtain the cable diagnostics result by calling the function EthTrcv_GetCableDiagnosticsResult() of the referenced Ethernet Transceiver Driver. If no Ethernet transceiver is referenced by the Ethernet switch port and development error detection is not enabled, the cable diagnostic result shall be set to ETHTRCV_CABLEDIAG_OK.
-[SWS_EthSwt_00351] as # 77349 introduces a general req

=== EthTrcv ===
add certain parameter to enable/disable API functions:
+ SWS item ECUC_EthTrcv_xxxx1
Name EthTrcvGetPhySignalQualityApi
Description Enables / Disables EthTrcv_GetPhySignalQuality API

+ SWS item ECUC_EthTrcv_xxxx2
Name EthTrcvGetPhyIdentifierApi
Description Enables / Disables EthTrcv_GetPhyIdentifier API

+ SWS item ECUC_EthTrcv_xxxx3
Name EthTrcvSetPhyTestModeApi
Description Enables / Disables EthTrcv_SetPhyTestMode API

+ SWS item ECUC_EthTrcv_xxxx4
Name EthTrcvSetPhyTxModeApi
Description Enables / Disables EthTrcv_SetPhyTxMode API

+ SWS item ECUC_EthTrcv_xxxx5
Name EthTrcvGetCableDiagnosticsResultApi
Description Enables / Disables EthTrcv_GetCableDiagnosticsResult API

add the following specification to the configuration parameter above:
Multiplicity 1
Type EcucBooleanParamDef
Default value –
Post-Build Variant Value false
Value Configuration
Class Pre-compile time X
All Variants Link time –
Post-build time –
Scope / Dependency scope: local
–Last change on issue 77628 comment 31–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.216 Specification Item SWS_EthSwt_00341

Trace References:

SRS_BSW_00406

Content:

If development error detection is enabled: the function EthSwt_SetPortTxMode shall check that the service EthSwt_SwitchInit was previously called. If the check fails, the function shall raise the development error ETHSWT_E_NOT_INITIALIZED and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

  For Error Detection following general rules apply:
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being

valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.217 Specification Item SWS_EthSwt_00342

Trace References:

SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchIdx is not valid, Eth Swt_SetPortTxMode shall raise the development error ETHSWT_E_INV_SWITCH_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

  For Error Detection following general rules apply:
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
  [SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.

[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.218 Specification Item SWS_EthSwt_00343

Trace References:

SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchPortIdx is not valid, EthSwt_SetPortTxMode shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return

E_NOT_OK if the driver is not yet initialized. ()

In the above requirement -
1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being

valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.219 Specification Item SWS_EthSwt_00345

Trace References:

SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369, SRS_ETH_00118

Content:

The function EthSwt_SetPortTxMode shall check if the corresponding EthTrcv API EthTrcv_SetPhyTxMode() of the indexed transceiver driver for the given SwitchPortIdx is available. If this is not the case, the function shall return E_NOT_OK and if development error tracing is activated by EthSwtDevErrorDetect the ETHSWT_E_INV_API shall be raised.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most

probably should be development error detection.

Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain

the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

- RfC #77628: [EthSwt] Behaviour of certain APIs for ports without EthSwtPortTrcvRef (part I)

  Problem description:

  The following APIs have partly a description regarding the handling of ports without EthSwtPortTrcvRef:
  EthSwt_GetPortSignalQuality
  EthSwt_GetPortIdentifier
  EthSwt_SetPortTestMode
  EthTrcv_SetPhyTestMode
  EthSwt_SetPortTxMode
  EthSwt_GetPortCableDiagnosticsResult

  The description should be adjusted and harmonized with related requirements.
  –Last change on issue 77628 comment 2–

  Agreed solution:

  === EthSwt ===
  ~[SWS_EthSwt_91014]EthSwt_GetPortSignalQuality
  Description:
  The function retrieves the signal quality of the link of the indexed Ethernet switch port.
  ~[SWS_EthSwt_00293]
  The function EthSwt_GetPortSignalQuality() shall obtain the signal quality by calling the function EthTrcv_GetPhySignalQuality() of the referenced Ethernet Transceiver Driver. If the current signal quality is not available, the signal quality shall be set to 0xFF.
  -[SWS_EthSwt_00298]as # 77349 introduces a general req

~[SWS_EthSwt_91015]EthSwt_GetPortIdentifier
Return value
E_NOT_OK: organizationally unique identifier of the Ethernet transceiver could not be obtained (i.e. OUI is not available)
Description:
This function retrieves the OUI (24 bit) of the indexed Ethernet switch port.

~[SWS_EthSwt_00299]
The function EthSwt_GetPortIdentifier() shall return the value of the organizationally unique identifier (OUI 24 bit) of the indexed Ethernet switch port that is connected to the indexed Ethernet switch. It shall set the 8 most significant bits of the OUI to 0xFFxxxxxx. If the Ethernet switch port references an Ethernet transceiver, the function shall obtain the OUI by calling the function EthTrcv_GetPhyIdentifier() and set the 8 most significant bits of the OUI to 0x00xxxxxx.

SWS_EthSwt_xxxxx] If neither the Ethernet switch port nor the Ethernet Transceiver Driver can provide an OUI the function EthSwt_GetPortIdentifier() shall return E_NOT_OK.

-[SWS_EthSwt_00304]as # 77349 introduces a general req

~[SWS_EthSwt_91029] EthSwt_SetPortTestMode
Description:
Activates a given test mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00328]
The function EthSwt_SetPortTestMode shall forward the call with the given test mode by calling the function EthTrcv_SetPhyTestMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00333]as # 77349 introduces a general req

~[SWS_EthSwt_91023] EthSwt_SetPortLoopbackMode
Description:
Activates a given test loop-back mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00334]
The function EthSwt_SetPortLoopbackMode() shall forward the call with the given loop-back mode by calling the function EthTrcv_SetPhyLoopbackMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00339] as # 77349 introduces a general req

~[SWS_EthSwt_91024] EthSwt_SetPortTxMode
Description:
Activates a given transmission mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00340]
The function EthSwt_SetPortTxMode() shall forward the call with the given transmission mode by calling the function EthTrcv_SetPhyTxMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00345] as # 77349 introduces a general req

~[SWS_EthSwt_91025]EthSwt_GetPortCableDiagnosticsResult
Description:
Retrieves the cable diagnostics result of the indexed Ethernet switch port respectively the referenced Ethernet Transceiver Driver.

~[SWS_EthSwt_00346]
The function EthSwt_GetPortCableDiagnosticsResult() shall obtain the cable diagnostics result by calling the function EthTrcv_GetCableDiagnosticsResult() of the referenced Ethernet Transceiver Driver. If no Ethernet transceiver is referenced by the Ethernet switch port and development error detection is not enabled, the cable diagnostic result shall be set to ETHTRCV_CABLEDIAG_OK.
-[SWS_EthSwt_00351] as # 77349 introduces a general req

=== EthTrcv ===
add certain parameter to enable/disable API functions:
+ SWS item ECUC_EthTrcv_xxxx1
Name EthTrcvGetPhySignalQualityApi
Description Enables / Disables EthTrcv_GetPhySignalQuality API

+ SWS item ECUC_EthTrcv_xxxx2
Name EthTrcvGetPhyIdentifierApi
Description Enables / Disables EthTrcv_GetPhyIdentifier API

+ SWS item ECUC_EthTrcv_xxxx3
Name EthTrcvSetPhyTestModeApi
Description Enables / Disables EthTrcv_SetPhyTestMode API

+ SWS item ECUC_EthTrcv_xxxx4
Name EthTrcvSetPhyTxModeApi
Description Enables / Disables EthTrcv_SetPhyTxMode API

+ SWS item ECUC_EthTrcv_xxxx5
Name EthTrcvGetCableDiagnosticsResultApi

Description Enables / Disables EthTrcv_GetCableDiagnosticsResult API

add the following specification to the configuration parameter above:
Multiplicity 1
Type EcucBooleanParamDef
Default value –
Post-Build Variant Value false
Value Configuration
Class Pre-compile time X
All Variants Link time –
Post-build time –
Scope / Dependency scope: local
–Last change on issue 77628 comment 31–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.220 Specification Item SWS_EthSwt_00346

Trace References:

SRS_Eth_00123

Content:

The function EthSwt_GetPortCableDiagnosticsResultshall retrieve the cable diagnostic result of the indexed Ethernet transceiver that is connected to the indexed port. The indexed port is connected to the indexed EthSwt. The value is provided () shall obtain the cable diagnostics result by calling the function EthTrcv_GetCableDiagnosticsResultof the () of the referenced Ethernet Transceiver Driver. If no Ethernet transceiver is referenced by the Ethernet switch port and development error detection is not enabled, the cable diagnostic result shall be set to ETHTRCV_CABLEDIAG_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

In the above requirement -
1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.

[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

- RfC #77628: [EthSwt] Behaviour of certain APIs for ports without EthSwtPortTrcvRef (part I)

  Problem description:

  The following APIs have partly a description regarding the handling of ports without EthSwtPortTrcvRef:
  EthSwt_GetPortSignalQuality
  EthSwt_GetPortIdentifier
  EthSwt_SetPortTestMode
  EthTrcv_SetPhyTestMode
  EthSwt_SetPortTxMode
  EthSwt_GetPortCableDiagnosticsResult

  The description should be adjusted and harmonized with related requirements.
  –Last change on issue 77628 comment 2–

  Agreed solution:

  === EthSwt ===
  ~[SWS_EthSwt_91014]EthSwt_GetPortSignalQuality
  Description:
  The function retrieves the signal quality of the link of the indexed Ethernet switch port.
  ~[SWS_EthSwt_00293]
  The function EthSwt_GetPortSignalQuality() shall obtain the signal quality by calling the function EthTrcv_GetPhySignalQuality() of the referenced Ethernet Transceiver Driver. If the current signal quality is not available, the signal quality shall be set to

0xFF.
-[SWS_EthSwt_00298]as # 77349 introduces a general req

~[SWS_EthSwt_91015]EthSwt_GetPortIdentifier
Return value
E_NOT_OK: organizationally unique identifier of the Ethernet transceiver could not be obtained (i.e. OUI is not available)
Description:
This function retrieves the OUI (24 bit) of the indexed Ethernet switch port.

~[SWS_EthSwt_00299]
The function EthSwt_GetPortIdentifier() shall return the value of the organizationally unique identifier (OUI 24 bit) of the indexed Ethernet switch port that is connected to the indexed Ethernet switch. It shall set the 8 most significant bits of the OUI to 0xFFxxxxxx. If the Ethernet switch port references an Ethernet transceiver, the function shall obtain the OUI by calling the function EthTrcv_GetPhyIdentifier() and set the 8 most significant bits of the OUI to 0x00xxxxxx.

SWS_EthSwt_xxxxx] If neither the Ethernet switch port nor the Ethernet Transceiver Driver can provide an OUI the function EthSwt_GetPortIdentifier() shall return E_NOT_OK.

-[SWS_EthSwt_00304]as # 77349 introduces a general req

~[SWS_EthSwt_91029] EthSwt_SetPortTestMode
Description:
Activates a given test mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00328]
The function EthSwt_SetPortTestMode shall forward the call with the given test mode by calling the function EthTrcv_SetPhyTestMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00333]as # 77349 introduces a general req

~[SWS_EthSwt_91023] EthSwt_SetPortLoopbackMode
Description:
Activates a given test loop-back mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00334]
The function EthSwt_SetPortLoopbackMode() shall forward the call with the given loop-back mode by calling the function EthTrcv_SetPhyLoopbackMode() of the referenced Ethernet Transceiver Driver.

-[SWS_EthSwt_00339] as # 77349 introduces a general req

~[SWS_EthSwt_91024] EthSwt_SetPortTxMode
Description:
Activates a given transmission mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00340]
The function EthSwt_SetPortTxMode() shall forward the call with the given transmission mode by calling the function EthTrcv_SetPhyTxMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00345] as # 77349 introduces a general req

~[SWS_EthSwt_91025]EthSwt_GetPortCableDiagnosticsResult
Description:
Retrieves the cable diagnostics result of the indexed Ethernet switch port respectively the referenced Ethernet Transceiver Driver.

~[SWS_EthSwt_00346]
The function EthSwt_GetPortCableDiagnosticsResult() shall obtain the cable diagnostics result by calling the function EthTrcv_GetCableDiagnosticsResult() of the referenced Ethernet Transceiver Driver. If no Ethernet transceiver is referenced by the Ethernet switch port and development error detection is not enabled, the cable diagnostic result shall be set to ETHTRCV_CABLEDIAG_OK.
-[SWS_EthSwt_00351] as # 77349 introduces a general req

=== EthTrcv ===
add certain parameter to enable/disable API functions:
+ SWS item ECUC_EthTrcv_xxxx1
Name EthTrcvGetPhySignalQualityApi
Description Enables / Disables EthTrcv_GetPhySignalQuality API

+ SWS item ECUC_EthTrcv_xxxx2
Name EthTrcvGetPhyIdentifierApi
Description Enables / Disables EthTrcv_GetPhyIdentifier API

+ SWS item ECUC_EthTrcv_xxxx3
Name EthTrcvSetPhyTestModeApi
Description Enables / Disables EthTrcv_SetPhyTestMode API

+ SWS item ECUC_EthTrcv_xxxx4
Name EthTrcvSetPhyTxModeApi
Description Enables / Disables EthTrcv_SetPhyTxMode API

+ SWS item ECUC_EthTrcv_xxxx5
Name EthTrcvGetCableDiagnosticsResultApi
Description Enables / Disables EthTrcv_GetCableDiagnosticsResult API

add the following specification to the configuration parameter above:
Multiplicity 1
Type EcucBooleanParamDef
Default value –
Post-Build Variant Value false
Value Configuration
Class Pre-compile time X
All Variants Link time –
Post-build time –
Scope / Dependency scope: local
–Last change on issue 77628 comment 31–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.221 Specification Item SWS_EthSwt_00347

Trace References:

SRS_BSW_00406

Content:

If development error detection is enabled: the function EthSwt_GetPortCableDiagnosticsResult shall check that the service EthSwt_SwitchInit was previously called. If the check fails, the function shall raise the development error ETHSWT_E_NOT_INITIALIZED and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -

1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevEr-

rorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.222 Specification Item SWS_EthSwt_00348

Trace References:

SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchIdx is not valid, EthSwt_GetPortCableDiagnosticsResult shall raise the development error ETHSWT_E_INV_SWITCH_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.223 Specification Item SWS_EthSwt_00349

Trace References:

SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchPortIdx is not valid, EthSwt_GetPortCableDiagnosticsResult shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.

Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.224 Specification Item SWS_EthSwt_00351

Trace References:

SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369, SRS_ETH_00118

Content:

The function EthSwt_GetPortCableDiagnosticsResult shall check if the corresponding Eth Trcv API EthTrcv_SetPhyTxMode() of the indexed transceiver driver for the given Switch PortIdx is available. If this is not the case, the function shall return E_NOT_OK and if development error tracing is activated by EthSwtDevErrorDetect the ETHSWT_E_INV_API shall be raised.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

- RfC #77628: [EthSwt] Behaviour of certain APIs for ports without EthSwtPortTrcvRef (part I)

  Problem description:

  The following APIs have partly a description regarding the handling of ports without EthSwtPortTrcvRef:
  EthSwt_GetPortSignalQuality
  EthSwt_GetPortIdentifier
  EthSwt_SetPortTestMode
  EthTrcv_SetPhyTestMode
  EthSwt_SetPortTxMode
  EthSwt_GetPortCableDiagnosticsResult

  The description should be adjusted and harmonized with related requirements.
  –Last change on issue 77628 comment 2–

  Agreed solution:

  === EthSwt ===
  ~[SWS_EthSwt_91014]EthSwt_GetPortSignalQuality
  Description:
  The function retrieves the signal quality of the link of the indexed Ethernet switch port.
  ~[SWS_EthSwt_00293]
  The function EthSwt_GetPortSignalQuality() shall obtain the signal quality by calling the function EthTrcv_GetPhySignalQuality() of the referenced Ethernet Transceiver Driver. If the current signal quality is not available, the signal quality shall be set to 0xFF.
  -[SWS_EthSwt_00298]as # 77349 introduces a general req

  ~[SWS_EthSwt_91015]EthSwt_GetPortIdentifier
  Return value
  E_NOT_OK: organizationally unique identifier of the Ethernet transceiver could not be obtained (i.e. OUI is not available)
  Description:
  This function retrieves the OUI (24 bit) of the indexed Ethernet switch port.

  ~[SWS_EthSwt_00299]
  The function EthSwt_GetPortIdentifier() shall return the value of the organizationally

unique identifier (OUI 24 bit) of the indexed Ethernet switch port that is connected to the indexed Ethernet switch. It shall set the 8 most significant bits of the OUI to 0xFFxxxxxx. If the Ethernet switch port references an Ethernet transceiver, the function shall obtain the OUI by calling the function EthTrcv_GetPhyIdentifier() and set the 8 most significant bits of the OUI to 0x00xxxxxx.

SWS_EthSwt_xxxxx] If neither the Ethernet switch port nor the Ethernet Transceiver Driver can provide an OUI the function EthSwt_GetPortIdentifier() shall return E_NOT_OK.

-[SWS_EthSwt_00304]as # 77349 introduces a general req

~[SWS_EthSwt_91029] EthSwt_SetPortTestMode
Description:
Activates a given test mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00328]
The function EthSwt_SetPortTestMode shall forward the call with the given test mode by calling the function EthTrcv_SetPhyTestMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00333]as # 77349 introduces a general req

~[SWS_EthSwt_91023] EthSwt_SetPortLoopbackMode
Description:
Activates a given test loop-back mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00334]
The function EthSwt_SetPortLoopbackMode() shall forward the call with the given loop-back mode by calling the function EthTrcv_SetPhyLoopbackMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00339] as # 77349 introduces a general req

~[SWS_EthSwt_91024] EthSwt_SetPortTxMode
Description:
Activates a given transmission mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00340]
The function EthSwt_SetPortTxMode() shall forward the call with the given transmission mode by calling the function EthTrcv_SetPhyTxMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00345] as # 77349 introduces a general req

~[SWS_EthSwt_91025]EthSwt_GetPortCableDiagnosticsResult

Description:
Retrieves the cable diagnostics result of the indexed Ethernet switch port respectively the referenced Ethernet Transceiver Driver.

~[SWS_EthSwt_00346]
The function EthSwt_GetPortCableDiagnosticsResult() shall obtain the cable diagnostics result by calling the function EthTrcv_GetCableDiagnosticsResult() of the referenced Ethernet Transceiver Driver. If no Ethernet transceiver is referenced by the Ethernet switch port and development error detection is not enabled, the cable diagnostic result shall be set to ETHTRCV_CABLEDIAG_OK.
-[SWS_EthSwt_00351] as # 77349 introduces a general req

=== EthTrcv ===
add certain parameter to enable/disable API functions:
+ SWS item ECUC_EthTrcv_xxxx1
Name EthTrcvGetPhySignalQualityApi
Description Enables / Disables EthTrcv_GetPhySignalQuality API

+ SWS item ECUC_EthTrcv_xxxx2
Name EthTrcvGetPhyIdentifierApi
Description Enables / Disables EthTrcv_GetPhyIdentifier API

+ SWS item ECUC_EthTrcv_xxxx3
Name EthTrcvSetPhyTestModeApi
Description Enables / Disables EthTrcv_SetPhyTestMode API

+ SWS item ECUC_EthTrcv_xxxx4
Name EthTrcvSetPhyTxModeApi
Description Enables / Disables EthTrcv_SetPhyTxMode API

+ SWS item ECUC_EthTrcv_xxxx5
Name EthTrcvGetCableDiagnosticsResultApi
Description Enables / Disables EthTrcv_GetCableDiagnosticsResult API

add the following specification to the configuration parameter above:
Multiplicity 1
Type EcucBooleanParamDef
Default value –
Post-Build Variant Value false
Value Configuration
Class Pre-compile time X
All Variants Link time –
Post-build time –

Scope / Dependency scope: local
–Last change on issue 77628 comment 31–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.225 Specification Item SWS_EthSwt_00352

Trace References:

SRS_Eth_00123

Content:

The function EthSwt_GetCfgHexDump shall return a memory block of the Ethernet switch configuration. The first 4 byte (byte0 ... byte3) shall contain the absolute start address in big endian format followed by the memory block content (Ethernet switch register). Unused memory bytes shall be padded with 0x00. The length of the block is given as parameter "CfgBlockLengthPtr". The memory block shall be copied to the given location "resultHexDumpBlockPtr". The length of the copied memory block (including the 4 byte address) shall be rewritten to "CfgBlockLengthPtr"

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77018: [EthSwt] Clarification about the use case regarding API "EthSwt_GetCfgHexDump"

  Problem description:

  We discussed the use case regarding the API "EthSwt_GetCfgHexDump". Daimler want to read out the register memory of an Ethernet switch as a memory dump. The memory dump is transmitted via an diagnostic service over the network consecutively to an diagnostic tester. The memory dump is used to verify and compare to the expected / required values. It is not clear if the API is sufficient for this use case.

  Agreed solution:

  Capter 7.1.2.13 exchange in listing of APIS EthSwt_GetCfgHexDump with EthSwt_GetCfgDataRaw and EthSwt_GetCfgHexDumpLength with EthSwt_GetCfgDataInfo.

  Chapter 8
  -SWS_EthSwt_91026,
  -SWS_EthSwt_00352

-SWS_EthSwt_00353
-SWS_EthSwt_00355
-ECUC_EthSwt_00093
-SWS_EthSwt_91027
-SWS_EthSwt_00356
-SWS_EthSwt_00357
-SWS_EthSwt_00358
-SWS_EthSwt_00359
Set APIs EthSwt_GetCfgHexDump and EthSwt_GetCfgHexDumpLength and config parameters EthSwtGetCfgHexDumpApi and EthSwt_GetCfgHexDumpLengthApi to deprecated or delete.

Chapter 10
Add parameter to enable disable APIs with name EthSwtGetCfgRaw to EthSwtGeneral-Container
Description Disable /Enable support of reading raw data from switch memory.
Multiplicity: 1
Type: EcucBooleanParamDef
Default: false
Post build variant false
Config class Pre-compile time for all variants.
scope: local

Introduce following new APIs to function: definitions
EthSwt_GetCfgDataRaw(
uint8 SwitchIdx,
uint32 Offset,
uint16 Length,
uint8 *BufferPtr
)
[attributes
- asynchronous
- non-reentrant
parameters IN: SwitchIdx:Index of the Ethernet switch within the context of the Ethernet Switch Driver
Offset: Offset of the Ethernet switch memory from where the reading starts
Length: Length of data in bytes that shall be copied.
parameters OUT: BufferPtr: Pointer to the location where the data shall be copied.
Std_ReturnType: E_OK: the data read was triggered successfully. E_NOT_OK: the data read was not triggered successfully. (i.e. indexed Ethernet switch is not available)
Description:

Retrieves the data in memory of the indexed Ethernet switch in variable length.]

+[SWS_EthSwt_xxxxx1] The function EthSwt_GetCfgDataRaw() shall only be available if parameter EthSwtGetCfgRaw is set to TRUE.(SRS_BSW_00171)
+[SWS_EthSwt_xxxxx2] When calling the function EthSwt_GetCfgDataRaw(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.()

EthSwt_GetCfgDataInfo(
uint8 SwitchIdx,
uint32 *DataSizePtr,
uint32 *DataAdressPtr
)
[attributes
- synchronous
- reentrant
parameters IN: SwitchIdx:Index of the Ethernet switch within the context of the Ethernet Switch Driver
parameters OUT: DataSizePtr: Pointer to the location where the total size of the configuration data shall be copied.
DataAddressPtr: Pointer to the location where the start address of the configuration registers shall be copied.
Std_ReturnType: E_OK: the data was obtained successfully. E_NOT_OK: the data was not obtained successfully. (i.e. indexed Ethernet switch is not available)
Description:
Retrieves the total size of data and the memory start address of the indexed Ethernet Switch.]

+[SWS_EthSwt_xxxxx3] The function EthSwt_GetCfgDataInfo() shall only be available if parameter EthSwtGetCfgRaw is set to TRUE.(SRS_BSW_00171)
+[SWS_EthSwt_xxxxx4] When calling the function EthSwt_GetCfgDataInfo(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.()

add following API to chapter "8.6.3 Configurable interfaces"
<GetCfgDataRawDone>(
uint8 SwitchIdx

)
[attributes:
synchronous
reentrant
parameters IN: SwitchIdx: Index of the Ethernet switch where the Configuration is read.
Return value : void
Description: The call of the function EthSwt_GetCfgDataRaw() triggers a asynchrony read of a certain memory section of the Ethernet switch driver. If the read is done, the configured callout function <GetCfgDataRawDone> shall be called]

Add a parameter to EthSwtGeneral-Container
GetCfgDataRawDone
Decription: Defines the function name for <GetCfgDataRawDone>
Multplicity: 0 ..1
Type EcucFuntionNameDef
Variant and Variant Multiplicity false
Config class Pre-compile for all variants
scope: local
dependency: The function GetCfgDataRawDone shall only be configured if parameter EthSwtGetCfgRaw is set to TRUE.

Header file name parameter is defined in ECUC_EthSwt_00064
–Last change on issue 77018 comment 32–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.226 Specification Item SWS_EthSwt_00353

Trace References:

SRS_BSW_00406

Content:

If development error detection is enabled: the function EthSwt_GetCfgHexDump shall check that the service EthSwt_SwitchInit was previously called. If the check fails, the function shall raise the development error ETHSWT_E_NOT_INITIALIZED and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77018: [EthSwt] Clarification about the use case regarding API "EthSwt_GetCfgHexDump"

  Problem description:

  We discussed the use case regarding the API "EthSwt_GetCfgHexDump". Daimler want to read out the register memory of an Ethernet switch as a memory dump. The memory dump is transmitted via an diagnostic service over the network consecutively to an diagnostic tester. The memory dump is used to verify and compare to the expected / required values. It is not clear if the API is sufficient for this use case.

  Agreed solution:

  Capter 7.1.2.13 exchange in listing of APIS EthSwt_GetCfgHexDump with EthSwt_GetCfgDataRaw and EthSwt_GetCfgHexDumpLength with EthSwt_GetCfgDataInfo.

  Chapter 8
  -SWS_EthSwt_91026,
  -SWS_EthSwt_00352
  -SWS_EthSwt_00353
  -SWS_EthSwt_00355
  -ECUC_EthSwt_00093
  -SWS_EthSwt_91027
  -SWS_EthSwt_00356
  -SWS_EthSwt_00357
  -SWS_EthSwt_00358
  -SWS_EthSwt_00359
  Set APIs EthSwt_GetCfgHexDump and EthSwt_GetCfgHexDumpLength and config parameters EthSwtGetCfgHexDumpApi and EthSwt_GetCfgHexDumpLengthApi to deprecated or delete.

  Chapter 10
  Add parameter to enable disable APIs with name EthSwtGetCfgRaw to EthSwtGeneral-Container
  Description Disable /Enable support of reading raw data from switch memory.
  Multiplicity: 1
  Type: EcucBooleanParamDef
  Default: false
  Post build variant false
  Config class Pre-compile time for all variants.
  scope: local

Introduce following new APIs to function: definitions
EthSwt_GetCfgDataRaw(
uint8 SwitchIdx,
uint32 Offset,
uint16 Length,
uint8 *BufferPtr
)
[attributes
- asynchronous
- non-reentrant
parameters IN: SwitchIdx:Index of the Ethernet switch within the context of the Ethernet Switch Driver
Offset: Offset of the Ethernet switch memory from where the reading starts
Length: Length of data in bytes that shall be copied.
parameters OUT: BufferPtr: Pointer to the location where the data shall be copied.
Std_ReturnType: E_OK: the data read was triggered successfully. E_NOT_OK: the data read was not triggered successfully. (i.e. indexed Ethernet switch is not available)
Description:
Retrieves the data in memory of the indexed Ethernet switch in variable length.]

+[SWS_EthSwt_xxxxx1] The function EthSwt_GetCfgDataRaw() shall only be available if parameter EthSwtGetCfgRaw is set to TRUE.(SRS_BSW_00171)
+[SWS_EthSwt_xxxxx2] When calling the function EthSwt_GetCfgDataRaw(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.()

EthSwt_GetCfgDataInfo(
uint8 SwitchIdx,
uint32 *DataSizePtr,
uint32 *DataAdressPtr
)
[attributes
- synchronous
- reentrant
parameters IN: SwitchIdx:Index of the Ethernet switch within the context of the Ethernet Switch Driver
parameters OUT: DataSizePtr: Pointer to the location where the total size of the configuration data shall be copied.

DataAddressPtr: Pointer to the location where the start address of the configuration registers shall be copied.
Std_ReturnType: E_OK: the data was obtained successfully. E_NOT_OK: the data was not obtained successfully. (i.e. indexed Ethernet switch is not available)
Description:
Retrieves the total size of data and the memory start address of the indexed Ethernet Switch.]

+[SWS_EthSwt_xxxxx3] The function EthSwt_GetCfgDataInfo() shall only be available if parameter EthSwtGetCfgRaw is set to TRUE.(SRS_BSW_00171)
+[SWS_EthSwt_xxxxx4] When calling the function EthSwt_GetCfgDataInfo(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.()

add following API to chapter "8.6.3 Configurable interfaces"
<GetCfgDataRawDone>(
uint8 SwitchIdx
)
[attributes:
synchronous
reentrant
parameters IN: SwitchIdx: Index of the Ethernet switch where the Configuration is read.
Return value : void
Description: The call of the function EthSwt_GetCfgDataRaw() triggers a asynchrony read of a certain memory section of the Ethernet switch driver. If the read is done, the configured callout function <GetCfgDataRawDone> shall be called]

Add a parameter to EthSwtGeneral-Container
GetCfgDataRawDone
Decription: Defines the function name for <GetCfgDataRawDone>
Multplicity: 0 ..1
Type EcucFuntionNameDef
Variant and Variant Multiplicity false
Config class Pre-compile for all variants
scope: local
dependency: The function GetCfgDataRawDone shall only be configured if parameter EthSwtGetCfgRaw is set to TRUE.

Header file name parameter is defined in ECUC_EthSwt_00064
–Last change on issue 77018 comment 32–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.227 Specification Item SWS_EthSwt_00354

Trace References:

SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchIdx is not valid, EthSwt_GetCfgHexDump shall raise the development error ETHSWT_E_INV_SWITCH_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77018: [EthSwt] Clarification about the use case regarding API "EthSwt_GetCfgHexDump"

  Problem description:

  We discussed the use case regarding the API "EthSwt_GetCfgHexDump". Daimler want to read out the register memory of an Ethernet switch as a memory dump. The memory dump is transmitted via an diagnostic service over the network consecutively to an diagnostic tester. The memory dump is used to verify and compare to the expected / required values. It is not clear if the API is sufficient for this use case.

  Agreed solution:

  Capter 7.1.2.13 exchange in listing of APIS EthSwt_GetCfgHexDump with EthSwt_GetCfgDataRaw and EthSwt_GetCfgHexDumpLength with EthSwt_GetCfgDataInfo.

  Chapter 8
  -SWS_EthSwt_91026,
  -SWS_EthSwt_00352
  -SWS_EthSwt_00353
  -SWS_EthSwt_00355
  -ECUC_EthSwt_00093
  -SWS_EthSwt_91027

-SWS_EthSwt_00356
-SWS_EthSwt_00357
-SWS_EthSwt_00358
-SWS_EthSwt_00359
Set APIs EthSwt_GetCfgHexDump and EthSwt_GetCfgHexDumpLength and config parameters EthSwtGetCfgHexDumpApi and EthSwt_GetCfgHexDumpLengthApi to deprecated or delete.

Chapter 10
Add parameter to enable disable APIs with name EthSwtGetCfgRaw to EthSwtGeneral-Container
Description Disable /Enable support of reading raw data from switch memory.
Multiplicity: 1
Type: EcucBooleanParamDef
Default: false
Post build variant false
Config class Pre-compile time for all variants.
scope: local

Introduce following new APIs to function: definitions
EthSwt_GetCfgDataRaw(
uint8 SwitchIdx,
uint32 Offset,
uint16 Length,
uint8 *BufferPtr
)
[attributes
- asynchronous
- non-reentrant
parameters IN: SwitchIdx:Index of the Ethernet switch within the context of the Ethernet Switch Driver
Offset: Offset of the Ethernet switch memory from where the reading starts
Length: Length of data in bytes that shall be copied.
parameters OUT: BufferPtr: Pointer to the location where the data shall be copied.
Std_ReturnType: E_OK: the data read was triggered successfully. E_NOT_OK: the data read was not triggered successfully. (i.e. indexed Ethernet switch is not available)
Description:
Retrieves the data in memory of the indexed Ethernet switch in variable length.]

+[SWS_EthSwt_xxxxx1] The function EthSwt_GetCfgDataRaw() shall only be available if parameter EthSwtGetCfgRaw is set to TRUE.(SRS_BSW_00171)

+[SWS_EthSwt_xxxxx2] When calling the function EthSwt_GetCfgDataRaw(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.()

EthSwt_GetCfgDataInfo(
uint8 SwitchIdx,
uint32 *DataSizePtr,
uint32 *DataAdressPtr
)
[attributes
- synchronous
- reentrant
parameters IN: SwitchIdx:Index of the Ethernet switch within the context of the Ethernet Switch Driver
parameters OUT: DataSizePtr: Pointer to the location where the total size of the configuration data shall be copied.
DataAddressPtr: Pointer to the location where the start address of the configuration registers shall be copied.
Std_ReturnType: E_OK: the data was obtained successfully. E_NOT_OK: the data was not obtained successfully. (i.e. indexed Ethernet switch is not available)
Description:
Retrieves the total size of data and the memory start address of the indexed Ethernet Switch.]

+[SWS_EthSwt_xxxxx3] The function EthSwt_GetCfgDataInfo() shall only be available if parameter EthSwtGetCfgRaw is set to TRUE.(SRS_BSW_00171)
+[SWS_EthSwt_xxxxx4] When calling the function EthSwt_GetCfgDataInfo(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.()

add following API to chapter "8.6.3 Configurable interfaces"
<GetCfgDataRawDone>(
uint8 SwitchIdx
)
[attributes:
synchronous
reentrant

parameters IN: SwitchIdx: Index of the Ethernet switch where the Configuration is read.
Return value : void
Description: The call of the function EthSwt_GetCfgDataRaw() triggers a asynchrony read of a certain memory section of the Ethernet switch driver. If the read is done, the configured callout function <GetCfgDataRawDone> shall be called]

Add a parameter to EthSwtGeneral-Container
GetCfgDataRawDone
Decription: Defines the function name for <GetCfgDataRawDone>
Multplicity: 0 ..1
Type EcucFuntionNameDef
Variant and Variant Multiplicity false
Config class Pre-compile for all variants
scope: local
dependency: The function GetCfgDataRawDone shall only be configured if parameter EthSwtGetCfgRaw is set to TRUE.

Header file name parameter is defined in ECUC_EthSwt_00064
–Last change on issue 77018 comment 32–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.228 Specification Item SWS_EthSwt_00355

Trace References:

SRS_BSW_00171

Content:

The function EthSwt_GetCfgHexDump shall be pre compile time configurable On/Off by the configuration parameter: EthSwt_GetCfgHexDumpApi.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77018: [EthSwt] Clarification about the use case regarding API "EthSwt_GetCfgHexDump"

  Problem description:

We discussed the use case regarding the API "EthSwt_GetCfgHexDump". Daimler want to read out the register memory of an Ethernet switch as a memory dump. The memory dump is transmitted via an diagnostic service over the network consecutively to an diagnostic tester. The memory dump is used to verify and compare to the expected / required values. It is not clear if the API is sufficient for this use case.

Agreed solution:

Capter 7.1.2.13 exchange in listing of APIS EthSwt_GetCfgHexDump with EthSwt_GetCfgDataRaw and EthSwt_GetCfgHexDumpLength with EthSwt_GetCfgDataInfo.

Chapter 8
-SWS_EthSwt_91026,
-SWS_EthSwt_00352
-SWS_EthSwt_00353
-SWS_EthSwt_00355
-ECUC_EthSwt_00093
-SWS_EthSwt_91027
-SWS_EthSwt_00356
-SWS_EthSwt_00357
-SWS_EthSwt_00358
-SWS_EthSwt_00359
Set APIs EthSwt_GetCfgHexDump and EthSwt_GetCfgHexDumpLength and config parameters EthSwtGetCfgHexDumpApi and EthSwt_GetCfgHexDumpLengthApi to deprecated or delete.

Chapter 10
Add parameter to enable disable APIs with name EthSwtGetCfgRaw to EthSwtGeneral-Container
Description Disable /Enable support of reading raw data from switch memory.
Multiplicity: 1
Type: EcucBooleanParamDef
Default: false
Post build variant false
Config class Pre-compile time for all variants.
scope: local

Introduce following new APIs to function: definitions
EthSwt_GetCfgDataRaw(
uint8 SwitchIdx,
uint32 Offset,
uint16 Length,

uint8 *BufferPtr
)
[attributes
- asynchronous
- non-reentrant
parameters IN: SwitchIdx:Index of the Ethernet switch within the context of the Ethernet Switch Driver
Offset: Offset of the Ethernet switch memory from where the reading starts
Length: Length of data in bytes that shall be copied.
parameters OUT: BufferPtr: Pointer to the location where the data shall be copied.
Std_ReturnType: E_OK: the data read was triggered successfully. E_NOT_OK: the data read was not triggered successfully. (i.e. indexed Ethernet switch is not available)
Description:
Retrieves the data in memory of the indexed Ethernet switch in variable length.]

+[SWS_EthSwt_xxxxx1] The function EthSwt_GetCfgDataRaw() shall only be available if parameter EthSwtGetCfgRaw is set to TRUE.(SRS_BSW_00171)
+[SWS_EthSwt_xxxxx2] When calling the function EthSwt_GetCfgDataRaw(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.()

EthSwt_GetCfgDataInfo(
uint8 SwitchIdx,
uint32 *DataSizePtr,
uint32 *DataAdressPtr
)
[attributes
- synchronous
- reentrant
parameters IN: SwitchIdx:Index of the Ethernet switch within the context of the Ethernet Switch Driver
parameters OUT: DataSizePtr: Pointer to the location where the total size of the configuration data shall be copied.
DataAddressPtr: Pointer to the location where the start address of the configuration registers shall be copied.
Std_ReturnType: E_OK: the data was obtained successfully. E_NOT_OK: the data was not obtained successfully. (i.e. indexed Ethernet switch is not available)
Description:
Retrieves the total size of data and the memory start address of the indexed

Ethernet Switch.]

+[SWS_EthSwt_xxxxx3] The function EthSwt_GetCfgDataInfo() shall only be available if parameter EthSwtGetCfgRaw is set to TRUE.(SRS_BSW_00171)
+[SWS_EthSwt_xxxxx4] When calling the function EthSwt_GetCfgDataInfo(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.()

add following API to chapter "8.6.3 Configurable interfaces"
<GetCfgDataRawDone>(
uint8 SwitchIdx
)
[attributes:
synchronous
reentrant
parameters IN: SwitchIdx: Index of the Ethernet switch where the Configuration is read.
Return value : void
Description: The call of the function EthSwt_GetCfgDataRaw() triggers a asynchrony read of a certain memory section of the Ethernet switch driver. If the read is done, the configured callout function <GetCfgDataRawDone> shall be called]

Add a parameter to EthSwtGeneral-Container
GetCfgDataRawDone
Decription: Defines the function name for <GetCfgDataRawDone>
Multplicity: 0 ..1
Type EcucFuntionNameDef
Variant and Variant Multiplicity false
Config class Pre-compile for all variants
scope: local
dependency: The function GetCfgDataRawDone shall only be configured if parameter EthSwtGetCfgRaw is set to TRUE.

Header file name parameter is defined in ECUC_EthSwt_00064
–Last change on issue 77018 comment 32–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.229 Specification Item SWS_EthSwt_00356

Trace References:

SRS_Eth_00123

Content:

The function EthSwt_GetCfgHexDumpLength shall return the total amount of the memory size of the Ethernet switch configuration.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77018: [EthSwt] Clarification about the use case regarding API "EthSwt_GetCfgHexDump"

  Problem description:

  We discussed the use case regarding the API "EthSwt_GetCfgHexDump". Daimler want to read out the register memory of an Ethernet switch as a memory dump. The memory dump is transmitted via an diagnostic service over the network consecutively to an diagnostic tester. The memory dump is used to verify and compare to the expected / required values. It is not clear if the API is sufficient for this use case.

  Agreed solution:

  Capter 7.1.2.13 exchange in listing of APIS EthSwt_GetCfgHexDump with EthSwt_GetCfgDataRaw and EthSwt_GetCfgHexDumpLength with EthSwt_GetCfgDataInfo.

  Chapter 8
  -SWS_EthSwt_91026,
  -SWS_EthSwt_00352
  -SWS_EthSwt_00353
  -SWS_EthSwt_00355
  -ECUC_EthSwt_00093
  -SWS_EthSwt_91027
  -SWS_EthSwt_00356
  -SWS_EthSwt_00357
  -SWS_EthSwt_00358
  -SWS_EthSwt_00359
  Set APIs EthSwt_GetCfgHexDump and EthSwt_GetCfgHexDumpLength and config parameters EthSwtGetCfgHexDumpApi and EthSwt_GetCfgHexDumpLengthApi to deprecated or delete.

  Chapter 10

Add parameter to enable disable APIs with name EthSwtGetCfgRaw to EthSwtGeneral-Container
Description Disable /Enable support of reading raw data from switch memory.
Multiplicity: 1
Type: EcucBooleanParamDef
Default: false
Post build variant false
Config class Pre-compile time for all variants.
scope: local

Introduce following new APIs to function: definitions
EthSwt_GetCfgDataRaw(
uint8 SwitchIdx,
uint32 Offset,
uint16 Length,
uint8 *BufferPtr
)
[attributes
- asynchronous
- non-reentrant
parameters IN: SwitchIdx:Index of the Ethernet switch within the context of the Ethernet Switch Driver
Offset: Offset of the Ethernet switch memory from where the reading starts
Length: Length of data in bytes that shall be copied.
parameters OUT: BufferPtr: Pointer to the location where the data shall be copied.
Std_ReturnType: E_OK: the data read was triggered successfully. E_NOT_OK: the data read was not triggered successfully. (i.e. indexed Ethernet switch is not available)
Description:
Retrieves the data in memory of the indexed Ethernet switch in variable length.]

+[SWS_EthSwt_xxxxx1] The function EthSwt_GetCfgDataRaw() shall only be available if parameter EthSwtGetCfgRaw is set to TRUE.(SRS_BSW_00171)
+[SWS_EthSwt_xxxxx2] When calling the function EthSwt_GetCfgDataRaw(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.()

EthSwt_GetCfgDataInfo(
uint8 SwitchIdx,

uint32 *DataSizePtr,
uint32 *DataAdressPtr
)
[attributes
- synchronous
- reentrant
parameters IN: SwitchIdx:Index of the Ethernet switch within the context of the Ethernet Switch Driver
parameters OUT: DataSizePtr: Pointer to the location where the total size of the configuration data shall be copied.
DataAddressPtr: Pointer to the location where the start address of the configuration registers shall be copied.
Std_ReturnType: E_OK: the data was obtained successfully. E_NOT_OK: the data was not obtained successfully. (i.e. indexed Ethernet switch is not available)
Description:
Retrieves the total size of data and the memory start address of the indexed Ethernet Switch.]

+[SWS_EthSwt_xxxxx3] The function EthSwt_GetCfgDataInfo() shall only be available if parameter EthSwtGetCfgRaw is set to TRUE.(SRS_BSW_00171)
+[SWS_EthSwt_xxxxx4] When calling the function EthSwt_GetCfgDataInfo(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.()

add following API to chapter "8.6.3 Configurable interfaces"
<GetCfgDataRawDone>(
uint8 SwitchIdx
)
[attributes:
synchronous
reentrant
parameters IN: SwitchIdx: Index of the Ethernet switch where the Configuration is read.
Return value : void
Description: The call of the function EthSwt_GetCfgDataRaw() triggers a asynchrony read of a certain memory section of the Ethernet switch driver. If the read is done, the configured callout function <GetCfgDataRawDone> shall be called]

Add a parameter to EthSwtGeneral-Container

GetCfgDataRawDone
Decription: Defines the function name for <GetCfgDataRawDone>
Multplicity: 0 ..1
Type EcucFuntionNameDef
Variant and Variant Multiplicity false
Config class Pre-compile for all variants
scope: local
dependency: The function GetCfgDataRawDone shall only be configured if parameter EthSwtGetCfgRaw is set to TRUE.

Header file name parameter is defined in ECUC_EthSwt_00064
–Last change on issue 77018 comment 32–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.230 Specification Item SWS_EthSwt_00357

Trace References:

SRS_BSW_00406

Content:

If development error detection is enabled: the function EthSwt_GetCfgHexDumpLength shall check that the service EthSwt_SwitchInit was previously called. If the check fails, the function shall raise the development error ETHSWT_E_NOT_INITIALIZED and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77018: [EthSwt] Clarification about the use case regarding API "EthSwt_GetCfgHexDump"

  Problem description:

  We discussed the use case regarding the API "EthSwt_GetCfgHexDump". Daimler want to read out the register memory of an Ethernet switch as a memory dump. The memory dump is transmitted via an diagnostic service over the network consecutively to an diagnostic tester. The memory dump is used to verify and compare to the expected / required values. It is not clear if the API is sufficient for this use case.

  Agreed solution:

Capter 7.1.2.13 exchange in listing of APIS EthSwt_GetCfgHexDump with EthSwt_GetCfgDataRaw and EthSwt_GetCfgHexDumpLength with EthSwt_GetCfgDataInfo.

Chapter 8
-SWS_EthSwt_91026,
-SWS_EthSwt_00352
-SWS_EthSwt_00353
-SWS_EthSwt_00355
-ECUC_EthSwt_00093
-SWS_EthSwt_91027
-SWS_EthSwt_00356
-SWS_EthSwt_00357
-SWS_EthSwt_00358
-SWS_EthSwt_00359
Set APIs EthSwt_GetCfgHexDump and EthSwt_GetCfgHexDumpLength and config parameters EthSwtGetCfgHexDumpApi and EthSwt_GetCfgHexDumpLengthApi to deprecated or delete.

Chapter 10
Add parameter to enable disable APIs with name EthSwtGetCfgRaw to EthSwtGeneral-Container
Description Disable /Enable support of reading raw data from switch memory.
Multiplicity: 1
Type: EcucBooleanParamDef
Default: false
Post build variant false
Config class Pre-compile time for all variants.
scope: local

Introduce following new APIs to function: definitions
EthSwt_GetCfgDataRaw(
uint8 SwitchIdx,
uint32 Offset,
uint16 Length,
uint8 *BufferPtr
)
[attributes
- asynchronous
- non-reentrant
parameters IN: SwitchIdx:Index of the Ethernet switch within the context of the Ethernet Switch Driver

Offset: Offset of the Ethernet switch memory from where the reading starts
Length: Length of data in bytes that shall be copied.
parameters OUT: BufferPtr: Pointer to the location where the data shall be copied.
Std_ReturnType: E_OK: the data read was triggered successfully. E_NOT_OK: the data read was not triggered successfully. (i.e. indexed Ethernet switch is not available)
Description:
Retrieves the data in memory of the indexed Ethernet switch in variable length.]

+[SWS_EthSwt_xxxxx1] The function EthSwt_GetCfgDataRaw() shall only be available if parameter EthSwtGetCfgRaw is set to TRUE.(SRS_BSW_00171)
+[SWS_EthSwt_xxxxx2] When calling the function EthSwt_GetCfgDataRaw(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.()

EthSwt_GetCfgDataInfo(
uint8 SwitchIdx,
uint32 *DataSizePtr,
uint32 *DataAdressPtr
)
[attributes
- synchronous
- reentrant
parameters IN: SwitchIdx:Index of the Ethernet switch within the context of the Ethernet Switch Driver
parameters OUT: DataSizePtr: Pointer to the location where the total size of the configuration data shall be copied.
DataAddressPtr: Pointer to the location where the start address of the configuration registers shall be copied.
Std_ReturnType: E_OK: the data was obtained successfully. E_NOT_OK: the data was not obtained successfully. (i.e. indexed Ethernet switch is not available)
Description:
Retrieves the total size of data and the memory start address of the indexed Ethernet Switch.]

+[SWS_EthSwt_xxxxx3] The function EthSwt_GetCfgDataInfo() shall only be available if parameter EthSwtGetCfgRaw is set to TRUE.(SRS_BSW_00171)
+[SWS_EthSwt_xxxxx4] When calling the function EthSwt_GetCfgDataInfo(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return

E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.()

add following API to chapter "8.6.3 Configurable interfaces"
<GetCfgDataRawDone>(
uint8 SwitchIdx
)
[attributes:
synchronous
reentrant
parameters IN: SwitchIdx: Index of the Ethernet switch where the Configuration is read.
Return value : void
Description: The call of the function EthSwt_GetCfgDataRaw() triggers a asynchrony read of a certain memory section of the Ethernet switch driver. If the read is done, the configured callout function <GetCfgDataRawDone> shall be called]

Add a parameter to EthSwtGeneral-Container
GetCfgDataRawDone
Decription: Defines the function name for <GetCfgDataRawDone>
Multplicity: 0 ..1
Type EcucFuntionNameDef
Variant and Variant Multiplicity false
Config class Pre-compile for all variants
scope: local
dependency: The function GetCfgDataRawDone shall only be configured if parameter EthSwtGetCfgRaw is set to TRUE.

Header file name parameter is defined in ECUC_EthSwt_00064
–Last change on issue 77018 comment 32–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.231 Specification Item SWS_EthSwt_00358

Trace References:

SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchIdx is not valid, EthSwt_GetCfgHexDumpLength shall raise the development error ETHSWT_E_INV_SWITCH_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77018: [EthSwt] Clarification about the use case regarding API "EthSwt_GetCfgHexDump"

  Problem description:

  We discussed the use case regarding the API "EthSwt_GetCfgHexDump". Daimler want to read out the register memory of an Ethernet switch as a memory dump. The memory dump is transmitted via an diagnostic service over the network consecutively to an diagnostic tester. The memory dump is used to verify and compare to the expected / required values. It is not clear if the API is sufficient for this use case.

  Agreed solution:

  Capter 7.1.2.13 exchange in listing of APIS EthSwt_GetCfgHexDump with EthSwt_GetCfgDataRaw and EthSwt_GetCfgHexDumpLength with EthSwt_GetCfgDataInfo.

  Chapter 8
  -SWS_EthSwt_91026,
  -SWS_EthSwt_00352
  -SWS_EthSwt_00353
  -SWS_EthSwt_00355
  -ECUC_EthSwt_00093
  -SWS_EthSwt_91027
  -SWS_EthSwt_00356
  -SWS_EthSwt_00357
  -SWS_EthSwt_00358
  -SWS_EthSwt_00359
  Set APIs EthSwt_GetCfgHexDump and EthSwt_GetCfgHexDumpLength and config parameters EthSwtGetCfgHexDumpApi and EthSwt_GetCfgHexDumpLengthApi to deprecated or delete.

  Chapter 10
  Add parameter to enable disable APIs with name EthSwtGetCfgRaw to EthSwtGeneral-Container
  Description Disable /Enable support of reading raw data from switch memory.
  Multiplicity: 1

Type: EcucBooleanParamDef
Default: false
Post build variant false
Config class Pre-compile time for all variants.
scope: local

Introduce following new APIs to function: definitions
EthSwt_GetCfgDataRaw(
uint8 SwitchIdx,
uint32 Offset,
uint16 Length,
uint8 *BufferPtr
)
[attributes
- asynchronous
- non-reentrant
parameters IN: SwitchIdx:Index of the Ethernet switch within the context of the Ethernet Switch Driver
Offset: Offset of the Ethernet switch memory from where the reading starts
Length: Length of data in bytes that shall be copied.
parameters OUT: BufferPtr: Pointer to the location where the data shall be copied.
Std_ReturnType: E_OK: the data read was triggered successfully. E_NOT_OK: the data read was not triggered successfully. (i.e. indexed Ethernet switch is not available)
Description:
Retrieves the data in memory of the indexed Ethernet switch in variable length.]

+[SWS_EthSwt_xxxxx1] The function EthSwt_GetCfgDataRaw() shall only be available if parameter EthSwtGetCfgRaw is set to TRUE.(SRS_BSW_00171)
+[SWS_EthSwt_xxxxx2] When calling the function EthSwt_GetCfgDataRaw(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.()

EthSwt_GetCfgDataInfo(
uint8 SwitchIdx,
uint32 *DataSizePtr,
uint32 *DataAdressPtr
)
[attributes

- synchronous
- reentrant

parameters IN: SwitchIdx:Index of the Ethernet switch within the context of the Ethernet Switch Driver
parameters OUT: DataSizePtr: Pointer to the location where the total size of the configuration data shall be copied.
DataAddressPtr: Pointer to the location where the start address of the configuration registers shall be copied.
Std_ReturnType: E_OK: the data was obtained successfully. E_NOT_OK: the data was not obtained successfully. (i.e. indexed Ethernet switch is not available)
Description:
Retrieves the total size of data and the memory start address of the indexed Ethernet Switch.]

+[SWS_EthSwt_xxxxx3] The function EthSwt_GetCfgDataInfo() shall only be available if parameter EthSwtGetCfgRaw is set to TRUE.(SRS_BSW_00171)
+[SWS_EthSwt_xxxxx4] When calling the function EthSwt_GetCfgDataInfo(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.()

add following API to chapter "8.6.3 Configurable interfaces"
<GetCfgDataRawDone>(
uint8 SwitchIdx
)
[attributes:
synchronous
reentrant
parameters IN: SwitchIdx: Index of the Ethernet switch where the Configuration is read.
Return value : void
Description: The call of the function EthSwt_GetCfgDataRaw() triggers a asynchrony read of a certain memory section of the Ethernet switch driver. If the read is done, the configured callout function <GetCfgDataRawDone> shall be called]

Add a parameter to EthSwtGeneral-Container
GetCfgDataRawDone
Decription: Defines the function name for <GetCfgDataRawDone>
Multplicity: 0 ..1
Type EcucFuntionNameDef

Variant and Variant Multiplicity false
Config class Pre-compile for all variants
scope: local
dependency: The function GetCfgDataRawDone shall only be configured if parameter EthSwtGetCfgRaw is set to TRUE.

Header file name parameter is defined in ECUC_EthSwt_00064
–Last change on issue 77018 comment 32–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.232 Specification Item SWS_EthSwt_00359

Trace References:

SRS_BSW_00171

Content:

The function EthSwt_GetCfgHexDumpLength shall be pre compile time configurable On/Off by the configuration parameter: EthSwt_GetCfgHexDumpLengthApi.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77018: [EthSwt] Clarification about the use case regarding API "EthSwt_GetCfgHexDump"

  Problem description:

  We discussed the use case regarding the API "EthSwt_GetCfgHexDump". Daimler want to read out the register memory of an Ethernet switch as a memory dump. The memory dump is transmitted via an diagnostic service over the network consecutively to an diagnostic tester. The memory dump is used to verify and compare to the expected / required values. It is not clear if the API is sufficient for this use case.

  Agreed solution:

  Capter 7.1.2.13 exchange in listing of APIS EthSwt_GetCfgHexDump with EthSwt_GetCfgDataRaw and EthSwt_GetCfgHexDumpLength with EthSwt_GetCfgDataInfo.

  Chapter 8
  -SWS_EthSwt_91026,

-SWS_EthSwt_00352
-SWS_EthSwt_00353
-SWS_EthSwt_00355
-ECUC_EthSwt_00093
-SWS_EthSwt_91027
-SWS_EthSwt_00356
-SWS_EthSwt_00357
-SWS_EthSwt_00358
-SWS_EthSwt_00359
Set APIs EthSwt_GetCfgHexDump and EthSwt_GetCfgHexDumpLength and config parameters EthSwtGetCfgHexDumpApi and EthSwt_GetCfgHexDumpLengthApi to deprecated or delete.

Chapter 10
Add parameter to enable disable APIs with name EthSwtGetCfgRaw to EthSwtGeneral-Container
Description Disable /Enable support of reading raw data from switch memory.
Multiplicity: 1
Type: EcucBooleanParamDef
Default: false
Post build variant false
Config class Pre-compile time for all variants.
scope: local

Introduce following new APIs to function: definitions
EthSwt_GetCfgDataRaw(
uint8 SwitchIdx,
uint32 Offset,
uint16 Length,
uint8 *BufferPtr
)
[attributes
- asynchronous
- non-reentrant
parameters IN: SwitchIdx:Index of the Ethernet switch within the context of the Ethernet Switch Driver
Offset: Offset of the Ethernet switch memory from where the reading starts
Length: Length of data in bytes that shall be copied.
parameters OUT: BufferPtr: Pointer to the location where the data shall be copied.
Std_ReturnType: E_OK: the data read was triggered successfully. E_NOT_OK: the data read was not triggered successfully. (i.e. indexed Ethernet switch is not available)

Description:
Retrieves the data in memory of the indexed Ethernet switch in variable length.]

+[SWS_EthSwt_xxxxx1] The function EthSwt_GetCfgDataRaw() shall only be available if parameter EthSwtGetCfgRaw is set to TRUE.(SRS_BSW_00171)
+[SWS_EthSwt_xxxxx2] When calling the function EthSwt_GetCfgDataRaw(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.()

EthSwt_GetCfgDataInfo(
uint8 SwitchIdx,
uint32 *DataSizePtr,
uint32 *DataAdressPtr
)
[attributes
- synchronous
- reentrant
parameters IN: SwitchIdx:Index of the Ethernet switch within the context of the Ethernet Switch Driver
parameters OUT: DataSizePtr: Pointer to the location where the total size of the configuration data shall be copied.
DataAddressPtr: Pointer to the location where the start address of the configuration registers shall be copied.
Std_ReturnType: E_OK: the data was obtained successfully. E_NOT_OK: the data was not obtained successfully. (i.e. indexed Ethernet switch is not available)
Description:
Retrieves the total size of data and the memory start address of the indexed Ethernet Switch.]

+[SWS_EthSwt_xxxxx3] The function EthSwt_GetCfgDataInfo() shall only be available if parameter EthSwtGetCfgRaw is set to TRUE.(SRS_BSW_00171)
+[SWS_EthSwt_xxxxx4] When calling the function EthSwt_GetCfgDataInfo(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.()

add following API to chapter "8.6.3 Configurable interfaces"
<GetCfgDataRawDone>(

uint8 SwitchIdx
)
[attributes:
synchronous
reentrant
parameters IN: SwitchIdx: Index of the Ethernet switch where the Configuration is read.
Return value : void
Description: The call of the function EthSwt_GetCfgDataRaw() triggers a asynchrony read of a certain memory section of the Ethernet switch driver. If the read is done, the configured callout function <GetCfgDataRawDone> shall be called]

Add a parameter to EthSwtGeneral-Container
GetCfgDataRawDone
Decription: Defines the function name for <GetCfgDataRawDone>
Multplicity: 0 ..1
Type EcucFuntionNameDef
Variant and Variant Multiplicity false
Config class Pre-compile for all variants
scope: local
dependency: The function GetCfgDataRawDone shall only be configured if parameter EthSwtGetCfgRaw is set to TRUE.

Header file name parameter is defined in ECUC_EthSwt_00064
–Last change on issue 77018 comment 32–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.233 Specification Item SWS_EthSwt_00360

Trace References:

SRS_BSW_00406

Content:

If development error detection is enabled: the function EthSwt_GetTxStats shall check that the service EthSwt_SwitchInit was previously called. If the check fails, the function shall raise the development error ETHSWT_E_NOT_INITIALIZED and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

  For Error Detection following general rules apply:
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
  [SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with

input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.234 Specification Item SWS_EthSwt_00361

Trace References:

SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchIdx is not valid,EthSwt_GetTxStats shall raise the development error ETHSWT_E_INV_SWITCH_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

[SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

In the above requirement -
1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.

[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.235 Specification Item SWS_EthSwt_00363

Trace References:

SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchPortIdx is not valid, EthSwt_ GetRxStats shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -

1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevEr-

rorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.236 Specification Item SWS_EthSwt_00364

Trace References:

SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter RxStats is a NULL pointer, EthSwt_ GetRxStats shall raise the development error ETHSWT_E_INV_POINTER and return E_NOT_OK

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.237 Specification Item SWS_EthSwt_00365

Trace References:

SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchPortIdx is not valid, EthSwt_ GetTxStats shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.

Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.238 Specification Item SWS_EthSwt_00366

Trace References:

SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter TxStats is a NULL pointer, EthSwt_ GetTxStats shall raise the development error ETHSWT_E_INV_POINTER and return E_NOT_OK

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in

chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.239 Specification Item SWS_EthSwt_00367

Trace References:

SRS_BSW_00406

Content:

If development error detection is enabled: the function EthSwt_GetTxErrorCounterValues shall check that the service EthSwt_SwitchInit was previously called. If the check fails, the function shall raise the development error ETHSWT_E_NOT_INITIALIZED and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER

add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.240 Specification Item SWS_EthSwt_00368

Trace References:

SRS_BSW_00413, SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchIdx is not valid,EthSwt_GetTxErrorCounterValues shall raise the development error ETHSWT_E_INV_SWITCH_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

  For Error Detection following general rules apply:

[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.241 Specification Item SWS_EthSwt_00369

Trace References:

SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter SwitchPortIdx is not valid, EthSwt_ GetTxErrorCounterValues shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

  For Error Detection following general rules apply:
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails,

the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.242 Specification Item SWS_EthSwt_00371

Trace References:

SRS_BSW_323, SRS_BSW_369

Content:

If development error detection is enabled and the parameter TxErrorCounterValues is a NULL pointer, EthSwt_ GetTxErrorCounterValues shall raise the development error ETHSWT_E_INV_POINTER and return E_NOT_OK

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

  For Error Detection following general rules apply:
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
  [SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with

input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.243 Specification Item SWS_EthSwt_00378

Trace References:

SRS_ETH_00125

Content:

If EthSwt_PortEnableTimeStamp is called for a PortIdx, the switch driver shall enable the time-stamping for all this port if EthSwtPortRole is not set to ETHSWT_UP_LINK_PORT and this port if EthSwtPortTimeStampSupport is set to TRUE for this port.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77050: [EthSwt] Uplink ports shall not be excluded from timestamping

  Problem description:

[SWS_EthSwt_00244] currently limits timestamping to
"all ... ports except the ports where EthSwtPortRole is set to ETHSWT_UP_LINK_PORT".

This limitation is problematic for cascaded switches since
no timestamps are taken when SYNC messages exit the first
switch and enter the second one.

It may be possible to omit timestamping on the link
between the cascaded switches if the timestamp counters in both switches are synchronized.
It is then possible to use the ingress timestamp on the first switch in combination with any egress timestamp on the second switch.

However, without a synchronization of the timestamp counters in both switches it is necessary to consider ingress and egress (on the uplink port) timestamps on the first switch and ingress (on the uplink port) and egress timestamps on the second switch.

Since it is unclear if all switch devices support timestamp counter synchronization and since this feature may result in additional hardware requirements (e.g., common clock source, reset lines etc.) it may be a show-stopper to omit timestamping at the uplink ports.

I recommend to either just remove this restriction or to make it configurable in case that the hardware supports synchronized counters.

Agreed solution:

~[SWS_EthSwt_00244][If EthSwt_EnableTimeStamp is called for a SwitchIdx, the switch driver shall enable the time-stamping for all his ports where EthSwtPortTimeStampSupport is set to TRUE.]

~[SWS_EthSwt_00378] If EthSwt_PortEnableTimeStamp is called for a PortIdx, the switch driver shall enable the time-stamping for this port if EthSwtPortTimeStampSupport is set to TRUE for this port.

~SWS_EthSwt_91011 EthSwt_EnableTimeStamping
Description: Activates egress time stamping on a dedicated message object on all ports of a Switch where EthSwtPortTimeStampSupport is set to TRUE. The selective activation of dedicated message objects for time stamping reduces the number of notification calls only to the required calls. Some HW does store once the egress time stamp marker and some HW needs it always before transmission. There will be no disabled functionality, due to the fact, that the message type is

always "time stamped" by network design.

~SWS_EthSwt_91028 EthSwt_PortEnableTimeStamp
Description: Activates egress time stamping on a dedicated message object on a dedicated port of a Switch if EthSwtPortTimeStampSupport is set to TRUE for this port. The selective activation of dedicated message objects for time stamping reduces the number of notification calls only to the required calls. Some HW does store once the egress time stamp marker and some HW needs it always before transmission. There will be no disabled functionality, due to the fact, that the message type is always "time stamped" by network design.
–Last change on issue 77050 comment 9–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.244 Specification Item SWS_EthSwt_00380

Trace References:

none

Content:

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76890: [EthSwt] SWS_EthSwt_91022 requirement ID is duplicated

  Problem description:

  In the chapters 8.3.45 EthSwt_SetPortMirrorState and 8.3.46 EthSwt_SetPortTestMode SWS_EthSwt_91022 requirement ID is reused.

  A different requirement ID should be used in chapter 8.3.46 EthSwt_SetPortTestMode.

  Agreed solution:

  - give new ReqID to EthSwt_SetPortTestMode
  - correct the ID format of requirement SWS_EthSwt_00380 (from "[SWS_EthSwt_00380]]" to "[SWS_EthSwt_00380]")
  –Last change on issue 76890 comment 6–

  BW-C-Level:

Application	Specification	Bus
1	1	1

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

  For Error Detection following general rules apply:
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.

[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.245 Specification Item SWS_EthSwt_00381

Trace References:

SRS_BSW_323, SRS_BSW_369

Content:

If default error detection is enabled: the function EthSwt_PortEnableTimeStamp() shall check the parameter SwitchIdx for being valid.

If the check fails, the function EthSwt_PortEnableTimeStamp() shall raise the development error ETHIF_E_INV_SWITCH_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

  For Error Detection following general rules apply:
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
  [SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.

[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.246 Specification Item SWS_EthSwt_00382

Trace References:

SRS_BSW_323, SRS_BSW_369

Content:

If default error detection is enabled: the function EthSwt_PortEnableTimeStamp() shall check the parameter BufIdx for being valid.

If the check fails, the function EthSwt_PortEnableTimeStamp() shall raise the development error ETHIF_E_INV_PARAM and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

[SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

In the above requirement -
1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.

[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.247 Specification Item SWS_EthSwt_00383

Trace References:

none

Content:

If development error detection is enabled and the parameter SwitchPortIdx is not valid, EthSwt_PortEnableTimeStamp() shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76248: [EthSwt]Remove references to non existing requirements (SRS, SWS)

  Problem description:

  Some review findings brought up references to non existing requirements. Please check and remove the references and if needed reformulate the requirements.

  Agreed solution:

SWS_EthSwt_00123 –> SRS_BSW_00406
SWS_EthSwt_00165 -> SRS_BSW_00395
SWS_EthSwt_00227 -> SRS_ETH_00087
SWS_EthSwt_91017, SWS_EthSwt_91020, –> SRS_ETH_00123
SWS_EthSwt_00006 –> SRS_BSW_00101
SWS_EthSwt_00010 left blank as SRS doe not yet provide a requirement here but maybe with RFC # 75170 AUTOSAR will introduce an SRS-item after all.
SWS_EthSwt_00376, SWS_EthSwt_00374, SWS_EthSwt_00375 –> SRS-Missing! same as for SWS_EThSwt_00010
SWS_EthSwt_00018, SRS_EthSwt_00025 –> SRS_ETH_00118
SWS_EthSwt_00031 –> SRS_ETH_00087
SWS_EthSwt_00037 –> SRS_ETH_00119
SWS_EthSwt_00044, SWS_EthSwt_00051 –> SRS_ETH_00118
SWS_EthSwt_00060 –> SRS_ETH_00087
SWS_EthSwt_00111 –> erase SRS_ETH_00086
SWS_EthSwt00079 –>SRS_ETH_00119
SWS_EthSwt_00206, SWS_EthSwt_00211, SWS_EthSwt_00216, SWS_EthSwt_00221 –>SRS_Eth_00120
SWS_EthSwt_00172 –>SRS_ETH_00121, SRS_ETH_00114
SWS_EthSwt_00086, SWS_EthSwt_00091, SWS_EthSwt_00182, SWS_EthSwt_00125 –> SRS_ETH_00087, SRS_ETH_00122
SWS_EthSwt_00187 –> SRS_ETH_00087
SWS_EthSwt_00058 –> SRS_BSW_00171
SWS_EthSwt_91012, SWS_EthSwt_91013 –> SRS_ETH_00126
SWS_EthSwt_91014, SWS_EthSwt_91015, SWS_EthSwt_91016, SWS_EthSwt_91018, SWS_EthSwt_91019, SWS_EthSwt_91021, SWS_EthSwt_91022, SWS_EthSwt_91022, SWS_EthSwt_91023, SWS_EthSwt_91024, SWS_EthSwt_91025, SWS_EthSwt_91026, SWS_EthSwt_91027 –> SRS_Eth_00123
SWS_EthSwt_00117, SWS_EthSwt_00203 –> SRS_ETH_00119, SRS_ETH_00087
SWS_EthSwt_00193 –> SRS_ETH_00122, SRS_ETH_00087
SWS_EthSwt_00114 –> SRS_BSW_00433
SWS_EthSwt_00098 –> SRS_Eth_00122, SRS_ETH_00118, SRS_ETH_00119, SRS_ETH_00120, SRS_ETH_00087, SRS_ETH_00125, SRS_BSW_00375
–Last change on issue 76248 comment 13–

BW-C-Level:

Application	Specification	Bus
1	1	1

- RfC #77349: Error name is "Development error" and not "Default error".

Problem description:

[SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

In the above requirement -
1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.

[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.248 Specification Item SWS_EthSwt_00386

Trace References:

SRS_BSW_00350

Content:

If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -

1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevEr-

rorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.249 Specification Item SWS_EthSwt_00387

Trace References:

SRS_BSW_00350

Content:

If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

Please check other similar functions too.

Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.250 Specification Item SWS_EthSwt_00389

Trace References:

SRS_BSW_00350

Content:

If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

Agreed solution:

Add a requirement for every Error-kind to Chapter 8
Delete all individual Error Detection requirements under the Function definitions in chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.

Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.251 Specification Item SWS_EthSwt_00390

Trace References:

SRS_BSW_00350

Content:

If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in

chapter 8:
SWS_EthSwt_00001
rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.252 Specification Item SWS_EthSwt_00391

Trace References:

SRS_BSW_00350

Content:

If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

add new requirements to chapter "7.2.1 Development Errors"

For Error Detection following general rules apply:
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.253 Specification Item SWS_EthSwt_00392

Trace References:

SRS_BSW_00350

Content:

If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

  For Error Detection following general rules apply:

[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.254 Specification Item SWS_EthSwt_00393

Trace References:

SRS_BSW_00350

Content:

If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

  For Error Detection following general rules apply:
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with in-

put parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.255 Specification Item SWS_EthSwt_00394

Trace References:

none

Content:

If neither the Ethernet switch port nor the Ethernet Transceiver Driver can provide an OUI the function EthSwt_GetPortIdentifier() shall return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

  For Error Detection following general rules apply:
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCHPORT_IDX.
  [SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with

input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.256 Specification Item SWS_EthSwt_00395

Trace References:

SRS_BSW_00385

Content:

Error Name:	ETHSWT_E_SYNCPORT2PHY
Short Description:	Ethernet switch port and the referenced Ethernet transceiver are in contradicting modes.
Long Description:	While getting the Ethernet switch port mode, the Ethernet switch driver detected an inconsistent state between Ethernet switch port and the referenced Ethernet transceiver Mode.
Recommended DTC:	N/A

Detection Criteria:	Fail	When getting the Ethernet switch port mode together with the Ethernet transceiver mode and the mode of the two referenced modules was found inconsistent the module shall report the extended production error with event status DEM_EVENT_STATUS_PREFAILED to DEM.
	Pass	When getting the Ethernet switch port mode together with the Ethernet transceiver mode and the mode of the two referenced modules was found consistent the module shall report the extended production error with event status DEM_EVENT_STATUS_PREPASSED to DEM.
Secondary Parameters:	N/A	
Time Required:	N/A	
Monitor Frequency	N/A	
MIL illumination:	N/A	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77250: [EthIf][EthSwt][EthTrcv] EthSwt_SetSwitchPortMode() broken for ports without Ethernet transceiver

  Problem description:

  SWS_EthSwt_00019 states that EthSwt_SetSwitchPortMode() shall (unconditionally) invoke EthTrcv_SetTransceiverMode() of the Ethernet Transceiver Driver.

  This is bogus for the following two reasons:
  1) a particular switch port does not necessarily have an associated Ethernet transceiver. - In that case EthSwtPortTrcvRef won't be present. This is probably the case for MAC2MAC links if EthSwtPortRole is present an either set to ETHSWT_HOST_PORT or to ETHSWT_UP_LINK_PORT.
  2) Even if the particular switch port has an associated Ethernet transceiver, SWS_EthTrcv_00043 states that the EthTrcv shall invoke EthIf_TrcvModeIndication() as a response to EthTrcv_SetTransceiverMode().

  IMHO 1) should be solved by extending SWS_EthSwt_00019 distinguishing between ports which are associated with a transceiver (i.e., EthSwtPortTrcvRef present) - in that case EthTrcv_SetTransceiverMode() should be called - and ports which are not associated with a transceiver (i.e., EthSwtPortTrcvRef is not present) -

in that case something else has to happen and this needs to be specified (probably the switch port shall be internally just disabled - maybe this needs to happen in the first case as well?)

2) leads to the following call chain: EthIf_SwitchPortGroupRequestMode() -> EthSwt_SetSwitchPortMode() -> EthTrcv_SetTransceiverMode() -> EthIf_TrcvModeIndication(). Thus the indication actually bypasses the EthSwt which IMHO is extremely ugly. Additionally for switch ports without associated transceiver, the EthIf will never get a EthIf_TrcvModeIndication() ...

Agreed solution:

=== System Template ===
+ Add upstream mapping of EthSwtPortTrcvRef to CouplingPort.physicalLayerType

=== EthSwt ===

~ch. 7.2.4 Production Errors
move SWS_EthSwt_00113 to 7.2.5 Extended production Errors
~SWS_EthSwt_00113 change detection criteria to
Fail:When access to the Ethernet Switch fails the module shall report the extended production error with event status DEM_EVENT_STATUS_PREFAILED to DEM.
Pass:When access to the Ethernet Switch succeeds the module shall report the extended production error with event status DEM_EVENT_STATUS_PREPASSED to DEM.

+ SWS_EthSwt_xxxxx new ext. production Error
Error Name: ETHSWT_E_SYNCPORT2PHY
Short Description: Ethernet switch port and the referenced Ethernet transceiver are in contradicting modes.
Long description: While getting the Ethernet switch port mode, the Ethernet switch driver detected an inconsistent state between Ethernet switch port and the referenced Ethernet transceiver Mode.
Detection Criteria: Fail: When getting the Ethernet switch port mode together with the Ethernet transceiver mode and the mode of the two referenced modules was found
inconsistent the module shall report the extended production error with event status DEM_EVENT_STATUS_PREFAILED to DEM.
Pass:When getting the Ethernet switch port mode together with the Ethernet transceiver mode and the mode of the two referenced modules was found
consistent the module shall report the extended production error with event status DEM_EVENT_STATUS_PREPASSED to DEM.
all other attributer: N/A

~ ch. 8.3.2 EthSwt_SwitchInit
~[SWS_EthSwt_00016] change "production error" to "extended production error"

~ch. 8.3.3 EthSwt_SetSwitchPortMode
~[SWS_EthSwt_00019]The function EthSwt_SetSwitchPortMode() shall put the indexed port of the switch into the specified mode. If EthSwtPort references an EthTrcv then the function EthTrcv_SetTransceiverMode() of the Ethernet Transceiver Driver shall additionally be called with the corresponding transceiver mode.

+[SWS_EthSwt_xxxxx] When calling the function EthSwt_SetSwitchPortMode() with mode ETHTRCV_MODE_DOWN, the EthSwt shall disable the Ethernet switch port directly for reduction of power consumption, if it is possible.

+[SWS_EthSwt_xxxxx] When calling the function EthSwt_SetSwitchPortMode(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.

+[SWS_EthSwt_xxxxxx] If EthSwtPort does not references an EthTrcv, EthSwt shall indicate a mode of the port by the API EthIf_SwitchPortModeIndication latest during the next EthSwt_MainFunction.(SRS_ETH_00118)

~ch. 10.1.3 EthSwtDemEventParameterRefs
+ SWS Item ECUC_EthSwt_xxxxx1 : Name ETHSWT_E_SYNCPORT2PHY
Parent Container EthSwtDemEventParameterRefs
Description Reference to the DemEventParameter which shall be issued when the error "Ethernet switch port and the referenced Ethernet transceiver are in contradicting modes" has occurred.
Multiplicity 0..1
Type Symbolic name reference to [DemEventParameter]
Post-Build Variant Multiplicity true
Post-Build Variant Value true
Multiplicity Configuration
Class Pre-compile time X VARIANT-PRE-COMPILE
Link time X VARIANT-LINK-TIME
Post-build time X VARIANT-POST-BUILD
Value Configuration Class
Pre-compile time X VARIANT-PRE-COMPILE
Link time X VARIANT-LINK-TIME
Post-build time X VARIANT-POST-BUILD

Scope / Dependency
scope: local

~ch. 10.1.5 EthSwtPort
~ECUC_EthSwt_00041 EthSwtPortTrcvRef change following attributes:
Multiplicity Configuration Class | Link time | VARIANT-LINK-TIME, VARIANT-POST-BUILD
| Post build | –
add to dependency: If EthSwtPortPhysicalLayerType is defined, then EthSwtPortTrcvRef holds the reference to the corresponding EthTrcv.

~ECUC_EthSwt_00054 add dependency to EthSwtPortPhysicalLayerType: If a EthSwtPort has an EthSwtPortPhysicalLayerType then EthSwtPort shall reference an EthTrcv.

=== EthIf ===
Add to chapter 8.4 (callback notifications):
+ EthIf_SwitchPortModeIndication
Syntax: void EthIf_SwitchPortModeIndication(uint8 SwitchIdx, uint8 SwitchPortIdx, EthSwt_ModeType PortMode)
Sevice ID: pick a free one
Sync/Async: Synchronous
Reentrancy: Non Reentrant
Parmaters in: SwitchIdx: Index of the switch within the context of the Ethernet Switch Driver
SwitchPortIdx: Index of the port at the addressed switch
PortMode: notified Ethernet Switch port mode.
Description: The EthIf shall determine the expected notifications based on the EthSwtPort configuration. In case the EthSwtPort references an EthTrcv the EthIf expects a notification from the EthTrcv via API EthIf_TrcvModeIndication(). Otherwise the EthIf expects a notification from the EthSwt via API EthIf_SwitchPortModeIndication().

~SWS_EthIf_00232
change from
Called asynchronously when mode has been read out. Triggered by previous
EthTrcv_SetTransceiverMode call. Can directly be called within the trigger
functions.
To
Called asynchronously when a mode change has been read out. If the function is triggered by previous call of
EthTrcv_SetTransceiverMode it can directly be called within the trigger function.
–Last change on issue 77250 comment 50–

BW-C-Level:

Application	Specification	Bus
1	4	1

- RfC #77349: Error name is "Development error" and not "Default error".

  Problem description:

  [SWS_CAN_00492] If default error detection for the Can module is enabled: The function Can_SetBaudrate shall raise the error CAN_E_UNINIT and return E_NOT_OK if the driver is not yet initialized. ()

  In the above requirement -
  1. There is no parameter which can turn default error detection ON/OFF. This most probably should be development error detection.

  Ref 7.11.1 mentions CanDrv errors has only development errors and not default errors. Again the ECUC_Can_00064 also specifies about development errors.

  Please check other similar functions too.

  Also the other part please delete - "and return E_NOT_OK" because DET call is not expected to return.

  Agreed solution:

  Add a requirement for every Error-kind to Chapter 8
  Delete all individual Error Detection requirements under the Function definitions in chapter 8:
  SWS_EthSwt_00001
  rename ETHSWT_E_INV_POINTER to ETHSWT_E_PARAM_POINTER
  add: Invalid configuration set selection ETHSWT_E_INIT_FAILED 0x09.

  add new requirements to chapter "7.2.1 Development Errors"

  For Error Detection following general rules apply:
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions except EthSwt_Init shall check that the service EthSwt_Init was previously called. If the check fails, the function shall raise the development error ETHSWT_E_UNINIT.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_SWITCH_IDX.
  [SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETH-

SWT_E_INV_SWITCHPORT_IDX.
[SWS_EthSwt_xxxxx]If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_CTRL_IDX.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_INV_PARAM.
[SWS_EthSwt_xxxxx] If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT_E_PARAM_POINTER.
[SWS_EthSwt_xxxxx] If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the functions shall raise the development error ETHSWT_E_INV_API.

Change SWS_EthSwt_00009 to
If development error detection is enabled, the function EthSwt_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt_Init shall raise the development error ETHSWT_E_INIT_FAILED.
Note: Please note that in case of variant pre-compile NULL_PTR is allowed.
–Last change on issue 77349 comment 30–

BW-C-Level:

Application	**Specification**	**Bus**
1	4	1

1.257 Specification Item SWS_EthSwt_00396

Trace References:

none

Content:

When calling the function EthSwt_SetSwitchPortMode() with mode ETHTRCV_MODE_DOWN, the EthSwt shall disable the Ethernet switch port directly for reduction of power consumption, if it is possible.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77250: [EthIf][EthSwt][EthTrcv] EthSwt_SetSwitchPortMode() broken for ports without Ethernet transceiver

  Problem description:

  SWS_EthSwt_00019 states that EthSwt_SetSwitchPortMode() shall (unconditionally) invoke EthTrcv_SetTransceiverMode() of the Ethernet Transceiver Driver.

  This is bogus for the following two reasons:
  1) a particular switch port does not necessarily have an associated Ethernet transceiver. - In that case EthSwtPortTrcvRef won't be present. This is probably the case for MAC2MAC links if EthSwtPortRole is present an either set to ETHSWT_HOST_PORT or to ETHSWT_UP_LINK_PORT.
  2) Even if the particular switch port has an associated Ethernet transceiver, SWS_EthTrcv_00043 states that the EthTrcv shall invoke EthIf_TrcvModeIndication() as a response to EthTrcv_SetTransceiverMode().

  IMHO 1) should be solved by extending SWS_EthSwt_00019 distinguishing between ports which are associated with a transceiver (i.e., EthSwtPortTrcvRef present) - in that case EthTrcv_SetTransceiverMode() should be called - and ports which are not associated with a transceiver (i.e., EthSwtPortTrcvRef is not present) - in that case something else has to happen and this needs to be specified (probably the switch port shall be internally just disabled - maybe this needs to happen in the first case as well?)

  2) leads to the following call chain: EthIf_SwitchPortGroupRequestMode() -> EthSwt_SetSwitchPortMode() -> EthTrcv_SetTransceiverMode() -> EthIf_TrcvModeIndication(). Thus the indication actually bypasses the EthSwt which IMHO is extremely ugly. Additionally for switch ports without associated transceiver, the EthIf will never get a EthIf_TrcvModeIndication() ...

  Agreed solution:

  === System Template ===
  + Add upstream mapping of EthSwtPortTrcvRef to CouplingPort.physicalLayerType

  === EthSwt ===

  ~ch. 7.2.4 Production Errors
  move SWS_EthSwt_00113 to 7.2.5 Extended production Errors
  ~SWS_EthSwt_00113 change detection criteria to
  Fail:When access to the Ethernet Switch fails the module shall report the extended production error with event status DEM_EVENT_STATUS_PREFAILED to DEM.
  Pass:When access to the Ethernet Switch succeeds the module shall report the extended production error with event status DEM_EVENT_STATUS_PREPASSED

to DEM.

+ SWS_EthSwt_xxxxx new ext. production Error
Error Name: ETHSWT_E_SYNCPORT2PHY
Short Description: Ethernet switch port and the referenced Ethernet transceiver are in contradicting modes.
Long description: While getting the Ethernet switch port mode, the Ethernet switch driver detected an inconsistent state between Ethernet switch port and the referenced Ethernet transceiver Mode.
Detection Criteria: Fail: When getting the Ethernet switch port mode together with the Ethernet transceiver mode and the mode of the two referenced modules was found
inconsistent the module shall report the extended production error with event status DEM_EVENT_STATUS_PREFAILED to DEM.
Pass:When getting the Ethernet switch port mode together with the Ethernet transceiver mode and the mode of the two referenced modules was found
consistent the module shall report the extended production error with event status DEM_EVENT_STATUS_PREPASSED to DEM.
all other attributer: N/A

~ ch. 8.3.2 EthSwt_SwitchInit
~[SWS_EthSwt_00016] change "production error" to "extended production error"

~ch. 8.3.3 EthSwt_SetSwitchPortMode
~[SWS_EthSwt_00019]The function EthSwt_SetSwitchPortMode() shall put the indexed port of the switch into the specified mode. If EthSwtPort references an EthTrcv then the function EthTrcv_SetTransceiverMode() of the Ethernet Transceiver Driver shall additionally be called with the corresponding transceiver mode.

+[SWS_EthSwt_xxxxx] When calling the function EthSwt_SetSwitchPortMode() with mode ETHTRCV_MODE_DOWN, the EthSwt shall disable the Ethernet switch port directly for reduction of power consumption, if it is possible.

+[SWS_EthSwt_xxxxx] When calling the function EthSwt_SetSwitchPortMode(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.

+[SWS_EthSwt_xxxxxx] If EthSwtPort does not references an EthTrcv, EthSwt shall indicate a mode of the port by the API EthIf_SwitchPortModeIndication latest during the next EthSwt_MainFunction.(SRS_ETH_00118)

~ch. 10.1.3 EthSwtDemEventParameterRefs
+ SWS Item ECUC_EthSwt_xxxxx1 : Name ETHSWT_E_SYNCPORT2PHY
Parent Container EthSwtDemEventParameterRefs
Description Reference to the DemEventParameter which shall be issued when the error "Ethernet switch port and the referenced Ethernet transceiver are in contradicting modes" has occurred.
Multiplicity 0..1
Type Symbolic name reference to [DemEventParameter]
Post-Build Variant Multiplicity true
Post-Build Variant Value true
Multiplicity Configuration
Class Pre-compile time X VARIANT-PRE-COMPILE
Link time X VARIANT-LINK-TIME
Post-build time X VARIANT-POST-BUILD
Value Configuration Class
Pre-compile time X VARIANT-PRE-COMPILE
Link time X VARIANT-LINK-TIME
Post-build time X VARIANT-POST-BUILD
Scope / Dependency
scope: local

~ch. 10.1.5 EthSwtPort
~ECUC_EthSwt_00041 EthSwtPortTrcvRef change following attributes:
Multiplicity Configuration Class | Link time | VARIANT-LINK-TIME, VARIANT-POST-BUILD
| Post build | –
add to dependency: If EthSwtPortPhysicalLayerType is defined, then EthSwtPortTrcvRef holds the reference to the corresponding EthTrcv.

~ECUC_EthSwt_00054 add dependency to EthSwtPortPhysicalLayerType: If a EthSwtPort has an EthSwtPortPhysicalLayerType then EthSwtPort shall reference an EthTrcv.

=== EthIf ===
Add to chapter 8.4 (callback notifications):
+ EthIf_SwitchPortModeIndication
Syntax: void EthIf_SwitchPortModeIndication(uint8 SwitchIdx, uint8 SwitchPortIdx, EthSwt_ModeType PortMode)
Sevice ID: pick a free one
Sync/Async: Synchronous
Reentrancy: Non Reentrant

Parmaters in: SwitchIdx: Index of the switch within the context of the Ethernet Switch Driver
SwitchPortIdx: Index of the port at the addressed switch
PortMode: notified Ethernet Switch port mode.
Description: The EthIf shall determine the expected notifications based on the EthSwtPort configuration. In case the EthSwtPort references an EthTrcv the EthIf expects a notification from the EthTrcv via API EthIf_TrcvModeIndication(). Otherwise the EthIf expects a notification from the EthSwt via API EthIf_SwitchPortModeIndication().

~SWS_EthIf_00232
change from
Called asynchronously when mode has been read out. Triggered by previous
EthTrcv_SetTransceiverMode call. Can directly be called within the trigger
functions.
To
Called asynchronously when a mode change has been read out. If the function is triggered by previous call of
EthTrcv_SetTransceiverMode it can directly be called within the trigger function.
–Last change on issue 77250 comment 50–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.258 Specification Item SWS_EthSwt_00397

Trace References:

none

Content:

When calling the function EthSwt_SetSwitchPortMode(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77250: [EthIf][EthSwt][EthTrcv] EthSwt_SetSwitchPortMode() broken for ports without Ethernet transceiver

Problem description:

SWS_EthSwt_00019 states that EthSwt_SetSwitchPortMode() shall (unconditionally) invoke EthTrcv_SetTransceiverMode() of the Ethernet Transceiver Driver.

This is bogus for the following two reasons:
1) a particular switch port does not necessarily have an associated Ethernet transceiver. - In that case EthSwtPortTrcvRef won't be present. This is probably the case for MAC2MAC links if EthSwtPortRole is present an either set to ETHSWT_HOST_PORT or to ETHSWT_UP_LINK_PORT.
2) Even if the particular switch port has an associated Ethernet transceiver, SWS_EthTrcv_00043 states that the EthTrcv shall invoke EthIf_TrcvModeIndication() as a response to EthTrcv_SetTransceiverMode().

IMHO 1) should be solved by extending SWS_EthSwt_00019 distinguishing between ports which are associated with a transceiver (i.e., EthSwtPortTrcvRef present) - in that case EthTrcv_SetTransceiverMode() should be called - and ports which are not associated with a transceiver (i.e., EthSwtPortTrcvRef is not present) - in that case something else has to happen and this needs to be specified (probably the switch port shall be internally just disabled - maybe this needs to happen in the first case as well?)

2) leads to the following call chain: EthIf_SwitchPortGroupRequestMode() -> EthSwt_SetSwitchPortMode() -> EthTrcv_SetTransceiverMode() -> EthIf_TrcvModeIndication(). Thus the indication actually bypasses the EthSwt which IMHO is extremely ugly. Additionally for switch ports without associated transceiver, the EthIf will never get a EthIf_TrcvModeIndication() ...

Agreed solution:

=== System Template ===
+ Add upstream mapping of EthSwtPortTrcvRef to CouplingPort.physicalLayerType

=== EthSwt ===

~ch. 7.2.4 Production Errors
move SWS_EthSwt_00113 to 7.2.5 Extended production Errors
~SWS_EthSwt_00113 change detection criteria to
Fail:When access to the Ethernet Switch fails the module shall report the extended production error with event status DEM_EVENT_STATUS_PREFAILED to DEM.
Pass:When access to the Ethernet Switch succeeds the module shall report the extended production error with event status DEM_EVENT_STATUS_PREPASSED to DEM.

+ SWS_EthSwt_xxxxx new ext. production Error
Error Name: ETHSWT_E_SYNCPORT2PHY
Short Description: Ethernet switch port and the referenced Ethernet transceiver are in contradicting modes.
Long description: While getting the Ethernet switch port mode, the Ethernet switch driver detected an inconsistent state between Ethernet switch port and the referenced Ethernet transceiver Mode.
Detection Criteria: Fail: When getting the Ethernet switch port mode together with the Ethernet transceiver mode and the mode of the two referenced modules was found
inconsistent the module shall report the extended production error with event status DEM_EVENT_STATUS_PREFAILED to DEM.
Pass:When getting the Ethernet switch port mode together with the Ethernet transceiver mode and the mode of the two referenced modules was found
consistent the module shall report the extended production error with event status DEM_EVENT_STATUS_PREPASSED to DEM.
all other attributer: N/A

~ ch. 8.3.2 EthSwt_SwitchInit
~[SWS_EthSwt_00016] change "production error" to "extended production error"

~ch. 8.3.3 EthSwt_SetSwitchPortMode
~[SWS_EthSwt_00019]The function EthSwt_SetSwitchPortMode() shall put the indexed port of the switch into the specified mode. If EthSwtPort references an EthTrcv then the function EthTrcv_SetTransceiverMode() of the Ethernet Transceiver Driver shall additionally be called with the corresponding transceiver mode.

+[SWS_EthSwt_xxxxx] When calling the function EthSwt_SetSwitchPortMode() with mode ETHTRCV_MODE_DOWN, the EthSwt shall disable the Ethernet switch port directly for reduction of power consumption, if it is possible.

+[SWS_EthSwt_xxxxx] When calling the function EthSwt_SetSwitchPortMode(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.

+[SWS_EthSwt_xxxxxx] If EthSwtPort does not references an EthTrcv, EthSwt shall indicate a mode of the port by the API EthIf_SwitchPortModeIndication latest during the next EthSwt_MainFunction.(SRS_ETH_00118)

~ch. 10.1.3 EthSwtDemEventParameterRefs
+ SWS Item ECUC_EthSwt_xxxxx1 : Name ETHSWT_E_SYNCPORT2PHY
Parent Container EthSwtDemEventParameterRefs
Description Reference to the DemEventParameter which shall be issued when the error "Ethernet switch port and the referenced Ethernet transceiver are in contradicting modes" has occurred.
Multiplicity 0..1
Type Symbolic name reference to [DemEventParameter]
Post-Build Variant Multiplicity true
Post-Build Variant Value true
Multiplicity Configuration
Class Pre-compile time X VARIANT-PRE-COMPILE
Link time X VARIANT-LINK-TIME
Post-build time X VARIANT-POST-BUILD
Value Configuration Class
Pre-compile time X VARIANT-PRE-COMPILE
Link time X VARIANT-LINK-TIME
Post-build time X VARIANT-POST-BUILD
Scope / Dependency
scope: local

~ch. 10.1.5 EthSwtPort
~ECUC_EthSwt_00041 EthSwtPortTrcvRef change following attributes:
Multiplicity Configuration Class | Link time | VARIANT-LINK-TIME, VARIANT-POST-BUILD
| Post build | –
add to dependency: If EthSwtPortPhysicalLayerType is defined, then EthSwtPortTrcvRef holds the reference to the corresponding EthTrcv.

~ECUC_EthSwt_00054 add dependency to EthSwtPortPhysicalLayerType: If a EthSwtPort has an EthSwtPortPhysicalLayerType then EthSwtPort shall reference an EthTrcv.

=== EthIf ===
Add to chapter 8.4 (callback notifications):
+ EthIf_SwitchPortModeIndication
Syntax: void EthIf_SwitchPortModeIndication(uint8 SwitchIdx, uint8 SwitchPortIdx, EthSwt_ModeType PortMode)
Sevice ID: pick a free one
Sync/Async: Synchronous
Reentrancy: Non Reentrant
Parmaters in: SwitchIdx: Index of the switch within the context of the Ethernet Switch Driver

SwitchPortIdx: Index of the port at the addressed switch
PortMode: notified Ethernet Switch port mode.
Description: The EthIf shall determine the expected notifications based on the EthSwtPort configuration. In case the EthSwtPort references an EthTrcv the EthIf expects a notification from the EthTrcv via API EthIf_TrcvModeIndication(). Otherwise the EthIf expects a notification from the EthSwt via API EthIf_SwitchPortModeIndication().

~SWS_EthIf_00232
change from
Called asynchronously when mode has been read out. Triggered by previous
EthTrcv_SetTransceiverMode call. Can directly be called within the trigger
functions.
To
Called asynchronously when a mode change has been read out. If the function is triggered by previous call of
EthTrcv_SetTransceiverMode it can directly be called within the trigger function.
–Last change on issue 77250 comment 50–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.259 Specification Item SWS_EthSwt_00398

Trace References:

none

Content:

If EthSwtPort does not references an EthTrcv, EthSwt shall indicate a mode of the port by the API EthIf_SwitchPortModeIndication latest during the next EthSwt_MainFunction.(SRS_ETH_00118)

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77250: [EthIf][EthSwt][EthTrcv] EthSwt_SetSwitchPortMode() broken for ports without Ethernet transceiver

  Problem description:

  SWS_EthSwt_00019 states that EthSwt_SetSwitchPortMode() shall (unconditionally) invoke EthTrcv_SetTransceiverMode() of the Ethernet Transceiver Driver.

This is bogus for the following two reasons:
1) a particular switch port does not necessarily have an associated Ethernet transceiver. - In that case EthSwtPortTrcvRef won't be present. This is probably the case for MAC2MAC links if EthSwtPortRole is present an either set to ETHSWT_HOST_PORT or to ETHSWT_UP_LINK_PORT.
2) Even if the particular switch port has an associated Ethernet transceiver, SWS_EthTrcv_00043 states that the EthTrcv shall invoke EthIf_TrcvModeIndication() as a response to EthTrcv_SetTransceiverMode().

IMHO 1) should be solved by extending SWS_EthSwt_00019 distinguishing between ports which are associated with a transceiver (i.e., EthSwtPortTrcvRef present) - in that case EthTrcv_SetTransceiverMode() should be called - and ports which are not associated with a transceiver (i.e., EthSwtPortTrcvRef is not present) - in that case something else has to happen and this needs to be specified (probably the switch port shall be internally just disabled - maybe this needs to happen in the first case as well?)

2) leads to the following call chain: EthIf_SwitchPortGroupRequestMode() -> EthSwt_SetSwitchPortMode() -> EthTrcv_SetTransceiverMode() -> EthIf_TrcvModeIndication(). Thus the indication actually bypasses the EthSwt which IMHO is extremely ugly. Additionally for switch ports without associated transceiver, the EthIf will never get a EthIf_TrcvModeIndication() ...

Agreed solution:

=== System Template ===
+ Add upstream mapping of EthSwtPortTrcvRef to CouplingPort.physicalLayerType

=== EthSwt ===

~ch. 7.2.4 Production Errors
move SWS_EthSwt_00113 to 7.2.5 Extended production Errors
~SWS_EthSwt_00113 change detection criteria to
Fail:When access to the Ethernet Switch fails the module shall report the extended production error with event status DEM_EVENT_STATUS_PREFAILED to DEM.
Pass:When access to the Ethernet Switch succeeds the module shall report the extended production error with event status DEM_EVENT_STATUS_PREPASSED to DEM.

+ SWS_EthSwt_xxxxx new ext. production Error
Error Name: ETHSWT_E_SYNCPORT2PHY
Short Description: Ethernet switch port and the referenced Ethernet transceiver are in contradicting modes.

Long description: While getting the Ethernet switch port mode, the Ethernet switch driver detected an inconsistent state between Ethernet switch port and the referenced Ethernet transceiver Mode.
Detection Criteria: Fail: When getting the Ethernet switch port mode together with the Ethernet transceiver mode and the mode of the two referenced modules was found
inconsistent the module shall report the extended production error with event status DEM_EVENT_STATUS_PREFAILED to DEM.
Pass:When getting the Ethernet switch port mode together with the Ethernet transceiver mode and the mode of the two referenced modules was found
consistent the module shall report the extended production error with event status DEM_EVENT_STATUS_PREPASSED to DEM.
all other attributer: N/A

~ ch. 8.3.2 EthSwt_SwitchInit
~[SWS_EthSwt_00016] change "production error" to "extended production error"

~ch. 8.3.3 EthSwt_SetSwitchPortMode
~[SWS_EthSwt_00019]The function EthSwt_SetSwitchPortMode() shall put the indexed port of the switch into the specified mode. If EthSwtPort references an EthTrcv then the function EthTrcv_SetTransceiverMode() of the Ethernet Transceiver Driver shall additionally be called with the corresponding transceiver mode.

+[SWS_EthSwt_xxxxx] When calling the function EthSwt_SetSwitchPortMode() with mode ETHTRCV_MODE_DOWN, the EthSwt shall disable the Ethernet switch port directly for reduction of power consumption, if it is possible.

+[SWS_EthSwt_xxxxx] When calling the function EthSwt_SetSwitchPortMode(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.

+[SWS_EthSwt_xxxxxx] If EthSwtPort does not references an EthTrcv, EthSwt shall indicate a mode of the port by the API EthIf_SwitchPortModeIndication latest during the next EthSwt_MainFunction.(SRS_ETH_00118)

~ch. 10.1.3 EthSwtDemEventParameterRefs
+ SWS Item ECUC_EthSwt_xxxxx1 : Name ETHSWT_E_SYNCPORT2PHY
Parent Container EthSwtDemEventParameterRefs
Description Reference to the DemEventParameter which shall be issued when

the error "Ethernet switch port and the referenced Ethernet transceiver are in contradicting modes" has occurred.
Multiplicity 0..1
Type Symbolic name reference to [DemEventParameter]
Post-Build Variant Multiplicity true
Post-Build Variant Value true
Multiplicity Configuration
Class Pre-compile time X VARIANT-PRE-COMPILE
Link time X VARIANT-LINK-TIME
Post-build time X VARIANT-POST-BUILD
Value Configuration Class
Pre-compile time X VARIANT-PRE-COMPILE
Link time X VARIANT-LINK-TIME
Post-build time X VARIANT-POST-BUILD
Scope / Dependency
scope: local

~ch. 10.1.5 EthSwtPort
~ECUC_EthSwt_00041 EthSwtPortTrcvRef change following attributes:
Multiplicity Configuration Class | Link time | VARIANT-LINK-TIME, VARIANT-POST-BUILD
| Post build | –
add to dependency: If EthSwtPortPhysicalLayerType is defined, then EthSwtPortTrcvRef holds the reference to the corresponding EthTrcv.

~ECUC_EthSwt_00054 add dependency to EthSwtPortPhysicalLayerType: If a EthSwtPort has an EthSwtPortPhysicalLayerType then EthSwtPort shall reference an EthTrcv.

=== EthIf ===
Add to chapter 8.4 (callback notifications):
+ EthIf_SwitchPortModeIndication
Syntax: void EthIf_SwitchPortModeIndication(uint8 SwitchIdx, uint8 SwitchPortIdx, EthSwt_ModeType PortMode)
Sevice ID: pick a free one
Sync/Async: Synchronous
Reentrancy: Non Reentrant
Parmaters in: SwitchIdx: Index of the switch within the context of the Ethernet Switch Driver
SwitchPortIdx: Index of the port at the addressed switch
PortMode: notified Ethernet Switch port mode.
Description: The EthIf shall determine the expected notifications based on the EthSwtPort configuration. In case the EthSwtPort references an EthTrcv the EthIf

expects a notification from the EthTrcv via API EthIf_TrcvModeIndication(). Otherwise the EthIf expects a notification from the EthSwt via API EthIf_SwitchPortModeIndication().

~SWS_EthIf_00232
change from
Called asynchronously when mode has been read out. Triggered by previous EthTrcv_SetTransceiverMode call. Can directly be called within the trigger functions.
To
Called asynchronously when a mode change has been read out. If the function is triggered by previous call of
EthTrcv_SetTransceiverMode it can directly be called within the trigger function.
–Last change on issue 77250 comment 50–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.260 Specification Item SWS_EthSwt_00399

Trace References:

none

Content:

If the obtained modes of the EthSwtPort and the EthTrcv are not aligned, the function EthSwt_GetSwitchPortMode shall raise the extended production error ETHSWT_E_SYNCPORT2PHY and return E_NOT_OK.

If EthTrcv_GetTransceiverMode returns E_NOT_OK, the EthSwt_GetSwitchPortMode shall also return E_NOT_OK without raising an error.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77497: [EthIf][EthSwt][EthTrcv] EthSwt_GetSwitchPortMode() broken for ports without Ethernet transceiver

  Problem description:

  RfC # 77250 introduce the correction for the API EthSwt_SetSwitchPortMode(). Thus, also the EthSwt_GetSwitchPortMode)() has to be revised.

  Agreed solution:

change [SWS_EthSwt_00026] to
The function EthSwt_GetSwitchPortMode shall read the mode of the indexed port of the switch. If EthSwtPort references an EthTrcv then the function shall additionally call the corresponding function EthTrcv_GetTransceiverMode of the Ethernet Transceiver Driver.

+[SWS_EthSwt_xxxxx] If the obtained modes of the EthSwtPort and the EthTrcv are not aligned, the function EthSwt_GetSwitchPortMode shall raise the extended production error ETHSWT_E_SYNCPORT2PHY and return E_NOT_OK.
If EthTrcv_GetTransceiverMode returns E_NOT_OK, the EthSwt_GetSwitchPortMode shall also return E_NOT_OK without raising an error.

+[SWS_EthSwt_xxxxx] If the function EthSwt_GetSwitchPortMode() is called, the function shall check the access to the Ethernet Switch Driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the production error ETHSWT_E_ACCESS and return E_OK.
–Last change on issue 77497 comment 12–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.261 Specification Item SWS_EthSwt_00400

Trace References:

none

Content:

If the function EthSwt_GetSwitchPortMode() is called, the function shall check the access to the Ethernet Switch Driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the production error ETHSWT_E_ACCESS and return E_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77497: [EthIf][EthSwt][EthTrcv] EthSwt_GetSwitchPortMode() broken for ports without Ethernet transceiver

  Problem description:

RfC # 77250 introduce the correction for the API EthSwt_SetSwitchPortMode(). Thus, also the EthSwt_GetSwitchPortMode)() has to be revised.

Agreed solution:

change [SWS_EthSwt_00026] to
The function EthSwt_GetSwitchPortMode shall read the mode of the indexed port of the switch. If EthSwtPort references an EthTrcv then the function shall additionally call the corresponding function EthTrcv_GetTransceiverMode of the Ethernet Transceiver Driver.

+[SWS_EthSwt_xxxxx] If the obtained modes of the EthSwtPort and the EthTrcv are not aligned, the function EthSwt_GetSwitchPortMode shall raise the extended production error ETHSWT_E_SYNCPORT2PHY and return E_NOT_OK.
If EthTrcv_GetTransceiverMode returns E_NOT_OK, the EthSwt_GetSwitchPortMode shall also return E_NOT_OK without raising an error.

+[SWS_EthSwt_xxxxx] If the function EthSwt_GetSwitchPortMode() is called, the function shall check the access to the Ethernet Switch Driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the production error ETHSWT_E_ACCESS and return E_OK.
–Last change on issue 77497 comment 12–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.262 Specification Item SWS_EthSwt_00404

Trace References:

none

Content:

When calling the function EthSwt_GetCfgDataRaw(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77018: [EthSwt] Clarification about the use case regarding API "EthSwt_GetCfgHexDump"

  Problem description:

  We discussed the use case regarding the API "EthSwt_GetCfgHexDump". Daimler want to read out the register memory of an Ethernet switch as a memory dump. The memory dump is transmitted via an diagnostic service over the network consecutively to an diagnostic tester. The memory dump is used to verify and compare to the expected / required values. It is not clear if the API is sufficient for this use case.

  Agreed solution:

  Capter 7.1.2.13 exchange in listing of APIS EthSwt_GetCfgHexDump with EthSwt_GetCfgDataRaw and EthSwt_GetCfgHexDumpLength with EthSwt_GetCfgDataInfo.

  Chapter 8
  -SWS_EthSwt_91026,
  -SWS_EthSwt_00352
  -SWS_EthSwt_00353
  -SWS_EthSwt_00355
  -ECUC_EthSwt_00093
  -SWS_EthSwt_91027
  -SWS_EthSwt_00356
  -SWS_EthSwt_00357
  -SWS_EthSwt_00358
  -SWS_EthSwt_00359
  Set APIs EthSwt_GetCfgHexDump and EthSwt_GetCfgHexDumpLength and config parameters EthSwtGetCfgHexDumpApi and EthSwt_GetCfgHexDumpLengthApi to deprecated or delete.

  Chapter 10
  Add parameter to enable disable APIs with name EthSwtGetCfgRaw to EthSwtGeneral-Container
  Description Disable /Enable support of reading raw data from switch memory.
  Multiplicity: 1
  Type: EcucBooleanParamDef
  Default: false
  Post build variant false
  Config class Pre-compile time for all variants.
  scope: local

  Introduce following new APIs to function: definitions

EthSwt_GetCfgDataRaw(
uint8 SwitchIdx,
uint32 Offset,
uint16 Length,
uint8 *BufferPtr
)
[attributes
- asynchronous
- non-reentrant
parameters IN: SwitchIdx:Index of the Ethernet switch within the context of the Ethernet Switch Driver
Offset: Offset of the Ethernet switch memory from where the reading starts
Length: Length of data in bytes that shall be copied.
parameters OUT: BufferPtr: Pointer to the location where the data shall be copied.
Std_ReturnType: E_OK: the data read was triggered successfully. E_NOT_OK: the data read was not triggered successfully. (i.e. indexed Ethernet switch is not available)
Description:
Retrieves the data in memory of the indexed Ethernet switch in variable length.]

+[SWS_EthSwt_xxxxx1] The function EthSwt_GetCfgDataRaw() shall only be available if parameter EthSwtGetCfgRaw is set to TRUE.(SRS_BSW_00171)
+[SWS_EthSwt_xxxxx2] When calling the function EthSwt_GetCfgDataRaw(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.()

EthSwt_GetCfgDataInfo(
uint8 SwitchIdx,
uint32 *DataSizePtr,
uint32 *DataAdressPtr
)
[attributes
- synchronous
- reentrant
parameters IN: SwitchIdx:Index of the Ethernet switch within the context of the Ethernet Switch Driver
parameters OUT: DataSizePtr: Pointer to the location where the total size of the configuration data shall be copied.
DataAddressPtr: Pointer to the location where the start address of the configuration registers shall be copied.

Std_ReturnType: E_OK: the data was obtained successfully. E_NOT_OK: the data was not obtained successfully. (i.e. indexed Ethernet switch is not available)
Description:
Retrieves the total size of data and the memory start address of the indexed Ethernet Switch.]

+[SWS_EthSwt_xxxxx3] The function EthSwt_GetCfgDataInfo() shall only be available if parameter EthSwtGetCfgRaw is set to TRUE.(SRS_BSW_00171)
+[SWS_EthSwt_xxxxx4] When calling the function EthSwt_GetCfgDataInfo(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.()

add following API to chapter "8.6.3 Configurable interfaces"
<GetCfgDataRawDone>(
uint8 SwitchIdx
)
[attributes:
synchronous
reentrant
parameters IN: SwitchIdx: Index of the Ethernet switch where the Configuration is read.
Return value : void
Description: The call of the function EthSwt_GetCfgDataRaw() triggers a asynchrony read of a certain memory section of the Ethernet switch driver. If the read is done, the configured callout function <GetCfgDataRawDone> shall be called]

Add a parameter to EthSwtGeneral-Container
GetCfgDataRawDone
Decription: Defines the function name for <GetCfgDataRawDone>
Multplicity: 0 ..1
Type EcucFuntionNameDef
Variant and Variant Multiplicity false
Config class Pre-compile for all variants
scope: local
dependency: The function GetCfgDataRawDone shall only be configured if parameter EthSwtGetCfgRaw is set to TRUE.

Header file name parameter is defined in ECUC_EthSwt_00064
–Last change on issue 77018 comment 32–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.263 Specification Item SWS_EthSwt_00405

Trace References:

SRS_BSW_00171

Content:

The function EthSwt_GetCfgDataInfo() shall only be available if parameter EthSwtGetCfgRaw is set to TRUE.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77018: [EthSwt] Clarification about the use case regarding API "EthSwt_GetCfgHexDump"

  Problem description:

  We discussed the use case regarding the API "EthSwt_GetCfgHexDump". Daimler want to read out the register memory of an Ethernet switch as a memory dump. The memory dump is transmitted via an diagnostic service over the network consecutively to an diagnostic tester. The memory dump is used to verify and compare to the expected / required values. It is not clear if the API is sufficient for this use case.

  Agreed solution:

  Capter 7.1.2.13 exchange in listing of APIS EthSwt_GetCfgHexDump with EthSwt_GetCfgDataRaw and EthSwt_GetCfgHexDumpLength with EthSwt_GetCfgDataInfo.

  Chapter 8
  -SWS_EthSwt_91026,
  -SWS_EthSwt_00352
  -SWS_EthSwt_00353
  -SWS_EthSwt_00355
  -ECUC_EthSwt_00093
  -SWS_EthSwt_91027
  -SWS_EthSwt_00356
  -SWS_EthSwt_00357
  -SWS_EthSwt_00358

-SWS_EthSwt_00359
Set APIs EthSwt_GetCfgHexDump and EthSwt_GetCfgHexDumpLength and config parameters EthSwtGetCfgHexDumpApi and EthSwt_GetCfgHexDumpLengthApi to deprecated or delete.

Chapter 10
Add parameter to enable disable APIs with name EthSwtGetCfgRaw to EthSwtGeneral-Container
Description Disable /Enable support of reading raw data from switch memory.
Multiplicity: 1
Type: EcucBooleanParamDef
Default: false
Post build variant false
Config class Pre-compile time for all variants.
scope: local

Introduce following new APIs to function: definitions
EthSwt_GetCfgDataRaw(
uint8 SwitchIdx,
uint32 Offset,
uint16 Length,
uint8 *BufferPtr
)
[attributes
- asynchronous
- non-reentrant
parameters IN: SwitchIdx:Index of the Ethernet switch within the context of the Ethernet Switch Driver
Offset: Offset of the Ethernet switch memory from where the reading starts
Length: Length of data in bytes that shall be copied.
parameters OUT: BufferPtr: Pointer to the location where the data shall be copied.
Std_ReturnType: E_OK: the data read was triggered successfully. E_NOT_OK: the data read was not triggered successfully. (i.e. indexed Ethernet switch is not available)
Description:
Retrieves the data in memory of the indexed Ethernet switch in variable length.]

+[SWS_EthSwt_xxxxx1] The function EthSwt_GetCfgDataRaw() shall only be available if parameter EthSwtGetCfgRaw is set to TRUE.(SRS_BSW_00171)
+[SWS_EthSwt_xxxxx2] When calling the function EthSwt_GetCfgDataRaw(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return

E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.()

EthSwt_GetCfgDataInfo(
uint8 SwitchIdx,
uint32 *DataSizePtr,
uint32 *DataAdressPtr
)
[attributes
- synchronous
- reentrant
parameters IN: SwitchIdx:Index of the Ethernet switch within the context of the Ethernet Switch Driver
parameters OUT: DataSizePtr: Pointer to the location where the total size of the configuration data shall be copied.
DataAddressPtr: Pointer to the location where the start address of the configuration registers shall be copied.
Std_ReturnType: E_OK: the data was obtained successfully. E_NOT_OK: the data was not obtained successfully. (i.e. indexed Ethernet switch is not available)
Description:
Retrieves the total size of data and the memory start address of the indexed Ethernet Switch.]

+[SWS_EthSwt_xxxxx3] The function EthSwt_GetCfgDataInfo() shall only be available if parameter EthSwtGetCfgRaw is set to TRUE.(SRS_BSW_00171)
+[SWS_EthSwt_xxxxx4] When calling the function EthSwt_GetCfgDataInfo(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.()

add following API to chapter "8.6.3 Configurable interfaces"
<GetCfgDataRawDone>(
uint8 SwitchIdx
)
[attributes:
synchronous
reentrant
parameters IN: SwitchIdx: Index of the Ethernet switch where the Configuration is read.
Return value : void

Description: The call of the function EthSwt_GetCfgDataRaw() triggers a asynchrony read of a certain memory section of the Ethernet switch driver. If the read is done, the configured callout function <GetCfgDataRawDone> shall be called]

Add a parameter to EthSwtGeneral-Container
GetCfgDataRawDone
Decription: Defines the function name for <GetCfgDataRawDone>
Multplicity: 0 ..1
Type EcucFuntionNameDef
Variant and Variant Multiplicity false
Config class Pre-compile for all variants
scope: local
dependency: The function GetCfgDataRawDone shall only be configured if parameter EthSwtGetCfgRaw is set to TRUE.

Header file name parameter is defined in ECUC_EthSwt_00064
–Last change on issue 77018 comment 32–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.264 Specification Item SWS_EthSwt_00406

Trace References:

none

Content:

When calling the function EthSwt_GetCfgDataInfo(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77018: [EthSwt] Clarification about the use case regarding API "EthSwt_GetCfgHexDump"

  Problem description:

We discussed the use case regarding the API "EthSwt_GetCfgHexDump". Daimler want to read out the register memory of an Ethernet switch as a memory dump. The memory dump is transmitted via an diagnostic service over the network consecutively to an diagnostic tester. The memory dump is used to verify and compare to the expected / required values. It is not clear if the API is sufficient for this use case.

Agreed solution:

Capter 7.1.2.13 exchange in listing of APIS EthSwt_GetCfgHexDump with EthSwt_GetCfgDataRaw and EthSwt_GetCfgHexDumpLength with EthSwt_GetCfgDataInfo.

Chapter 8
-SWS_EthSwt_91026,
-SWS_EthSwt_00352
-SWS_EthSwt_00353
-SWS_EthSwt_00355
-ECUC_EthSwt_00093
-SWS_EthSwt_91027
-SWS_EthSwt_00356
-SWS_EthSwt_00357
-SWS_EthSwt_00358
-SWS_EthSwt_00359
Set APIs EthSwt_GetCfgHexDump and EthSwt_GetCfgHexDumpLength and config parameters EthSwtGetCfgHexDumpApi and EthSwt_GetCfgHexDumpLengthApi to deprecated or delete.

Chapter 10
Add parameter to enable disable APIs with name EthSwtGetCfgRaw to EthSwtGeneral-Container
Description Disable /Enable support of reading raw data from switch memory.
Multiplicity: 1
Type: EcucBooleanParamDef
Default: false
Post build variant false
Config class Pre-compile time for all variants.
scope: local

Introduce following new APIs to function: definitions
EthSwt_GetCfgDataRaw(
uint8 SwitchIdx,
uint32 Offset,
uint16 Length,

uint8 *BufferPtr
)
[attributes
- asynchronous
- non-reentrant
parameters IN: SwitchIdx:Index of the Ethernet switch within the context of the Ethernet Switch Driver
Offset: Offset of the Ethernet switch memory from where the reading starts
Length: Length of data in bytes that shall be copied.
parameters OUT: BufferPtr: Pointer to the location where the data shall be copied.
Std_ReturnType: E_OK: the data read was triggered successfully. E_NOT_OK: the data read was not triggered successfully. (i.e. indexed Ethernet switch is not available)
Description:
Retrieves the data in memory of the indexed Ethernet switch in variable length.]

+[SWS_EthSwt_xxxxx1] The function EthSwt_GetCfgDataRaw() shall only be available if parameter EthSwtGetCfgRaw is set to TRUE.(SRS_BSW_00171)
+[SWS_EthSwt_xxxxx2] When calling the function EthSwt_GetCfgDataRaw(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.()

EthSwt_GetCfgDataInfo(
uint8 SwitchIdx,
uint32 *DataSizePtr,
uint32 *DataAdressPtr
)
[attributes
- synchronous
- reentrant
parameters IN: SwitchIdx:Index of the Ethernet switch within the context of the Ethernet Switch Driver
parameters OUT: DataSizePtr: Pointer to the location where the total size of the configuration data shall be copied.
DataAddressPtr: Pointer to the location where the start address of the configuration registers shall be copied.
Std_ReturnType: E_OK: the data was obtained successfully. E_NOT_OK: the data was not obtained successfully. (i.e. indexed Ethernet switch is not available)
Description:
Retrieves the total size of data and the memory start address of the indexed

Ethernet Switch.]

+[SWS_EthSwt_xxxxx3] The function EthSwt_GetCfgDataInfo() shall only be available if parameter EthSwtGetCfgRaw is set to TRUE.(SRS_BSW_00171)
+[SWS_EthSwt_xxxxx4] When calling the function EthSwt_GetCfgDataInfo(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.()

add following API to chapter "8.6.3 Configurable interfaces"
<GetCfgDataRawDone>(
uint8 SwitchIdx
)
[attributes:
synchronous
reentrant
parameters IN: SwitchIdx: Index of the Ethernet switch where the Configuration is read.
Return value : void
Description: The call of the function EthSwt_GetCfgDataRaw() triggers a asynchrony read of a certain memory section of the Ethernet switch driver. If the read is done, the configured callout function <GetCfgDataRawDone> shall be called]

Add a parameter to EthSwtGeneral-Container
GetCfgDataRawDone
Decription: Defines the function name for <GetCfgDataRawDone>
Multplicity: 0 ..1
Type EcucFuntionNameDef
Variant and Variant Multiplicity false
Config class Pre-compile for all variants
scope: local
dependency: The function GetCfgDataRawDone shall only be configured if parameter EthSwtGetCfgRaw is set to TRUE.

Header file name parameter is defined in ECUC_EthSwt_00064
–Last change on issue 77018 comment 32–

BW-C-Level:

Application	**Specification**	**Bus**
1	4	1

1.265 Specification Item SWS_EthSwt_91011

Trace References:

SRS_Eth_00125

Content:

Service name:	EthSwt_EnableTimeStamping (obsolete)EthSwt_EnableTimeStamping	
Syntax:	Std_ReturnType EthSwt_EnableTimeStamping(uint8 SwitchIdx, Eth_BufIdxType BufIdx, EthSwt_MgmtInfoType* MgmtInfoPtr)	
Service ID[hex]:	0x30	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	SwitchIdxEthSwt_EnableTimeStamping.SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	BufIdxEthSwt_EnableTimeStamping.BufIdx	Index of the message buffer, where Application expects egress time stamping
	MgmtInfoPtrEthSwt_EnableTimeStamping.MgmtInfoPtr	Management information
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: Time stamping on egress successfully enabled E_NOT_OK: Enabling of time stamping on egress has been failed
Description:	Activates egress time stamping on a dedicated message object on all ports of a Switch but on uplink ports between cascaded switcheswhere EthSwtPortTimeStampSupport is set to TRUE. The selective activation of dedicated message objects for time stamping reduces the number of notification calls only to the required calls. Some HW does store once the egress time stamp marker and some HW needs it always before transmission. There will be no disabled functionality, due to the fact, that the message type is always "time stamped" by network design. Tags: atp.Status=obsolete	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77050: [EthSwt] Uplink ports shall not be excluded from timestamping

  Problem description:

  [SWS_EthSwt_00244] currently limits timestamping to
  "all ... ports except the ports where EthSwtPortRole is set to ETH-SWT_UP_LINK_PORT".

  This limitation is problematic for cascaded switches since

no timestamps are taken when SYNC messages exit the first
switch and enter the second one.

It may be possible to omit timestamping on the link
between the cascaded switches if the timestamp counters in both switches are synchronized.
It is then possible to use the ingress timestamp on the first switch in combination with any egress timestamp on the second switch.

However, without a synchronization of the timestamp counters in both switches it is necessary to consider ingress and egress (on the uplink port) timestamps on the first switch and ingress (on the uplink port) and egress timestamps on the second switch.

Since it is unclear if all switch devices support timestamp counter synchronization and since this feature may result in additional hardware requirements (e.g., common clock source, reset lines etc.) it may be a show-stopper to omit timestamping at the uplink ports.

I recommend to either just remove this restriction or to make it configurable in case that the hardware supports synchronized counters.

Agreed solution:

~[SWS_EthSwt_00244][If EthSwt_EnableTimeStamp is called for a SwitchIdx, the switch driver shall enable the time-stamping for all his ports where EthSwtPortTimeStampSupport is set to TRUE.]

~[SWS_EthSwt_00378] If EthSwt_PortEnableTimeStamp is called for a PortIdx, the switch driver shall enable the time-stamping for this port if EthSwtPortTimeStampSupport is set to TRUE for this port.

~SWS_EthSwt_91011 EthSwt_EnableTimeStamping
Description: Activates egress time stamping on a dedicated message object on all ports of a Switch where EthSwtPortTimeStampSupport is set to TRUE. The selective activation of dedicated message objects for time stamping reduces the number of notification calls only to the required calls. Some HW does store once the egress time stamp marker and some HW needs it always before transmission. There will be no disabled functionality, due to the fact, that the message type is always "time stamped" by network design.

~SWS_EthSwt_91028 EthSwt_PortEnableTimeStamp
Description: Activates egress time stamping on a dedicated message object on a dedicated port of a Switch if EthSwtPortTimeStampSupport is set to TRUE for

this port. The selective activation of dedicated message objects for time stamping reduces the number of notification calls only to the required calls. Some HW does store once the egress time stamp marker and some HW needs it always before transmission. There will be no disabled functionality, due to the fact, that the message type is always "time stamped" by network design.
–Last change on issue 77050 comment 9–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.266 Specification Item SWS_EthSwt_91012

Trace References:

SRS_Eth_00086 ETH_00126

Content:

Service name:	EthSwt_VerifyConfigEthSwt_VerifyConfig	
Syntax:	Std_ReturnType EthSwt_VerifyConfig(uint8 SwitchIdx, boolean* Result)	
Service ID[hex]:	0x31	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	SwitchIdxEthSwt_VerifyConfig.SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
Parameters (inout):	None	
Parameters (out):	ResultEthSwt_VerifyConfig.Result	Result of verification, TRUE: configureation verified ok, FALSE: configuraton values found corrupted
Return value:	Std_ReturnType	E_OK: Configuration verificaton succeeded, E_NOT_OK: Configuration verification not succeeded.
Description:	Verifies the Switch Configuration depending on the HW-Architecture, HW-capability and the intended accuracy of this verification.	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76248: [EthSwt]Remove references to non existing requirements (SRS, SWS)

  Problem description:

Some review findings brought up references to non existing requirements. Please check and remove the references and if needed reformulate the requirements.

Agreed solution:

SWS_EthSwt_00123 –> SRS_BSW_00406
SWS_EthSwt_00165 -> SRS_BSW_00395
SWS_EthSwt_00227 -> SRS_ETH_00087
SWS_EthSwt_91017, SWS_EthSwt_91020, –> SRS_ETH_00123
SWS_EthSwt_00006 –> SRS_BSW_00101
SWS_EthSwt_00010 left blank as SRS doe not yet provide a requirement here but maybe with RFC # 75170 AUTOSAR will introduce an SRS-item after all.
SWS_EthSwt_00376, SWS_EthSwt_00374, SWS_EthSwt_00375 –> SRS-Missing! same as for SWS_EThSwt_00010
SWS_EthSwt_00018, SRS_EthSwt_00025 –> SRS_ETH_00118
SWS_EthSwt_00031 –> SRS_ETH_00087
SWS_EthSwt_00037 –> SRS_ETH_00119
SWS_EthSwt_00044, SWS_EthSwt_00051 –> SRS_ETH_00118
SWS_EthSwt_00060 –> SRS_ETH_00087
SWS_EthSwt_00111 –> erase SRS_ETH_00086
SWS_EthSwt00079 –>SRS_ETH_00119
SWS_EthSwt_00206, SWS_EthSwt_00211, SWS_EthSwt_00216, SWS_EthSwt_00221 –>SRS_Eth_00120
SWS_EthSwt_00172 –>SRS_ETH_00121, SRS_ETH_00114
SWS_EthSwt_00086, SWS_EthSwt_00091, SWS_EthSwt_00182, SWS_EthSwt_00125 –> SRS_ETH_00087, SRS_ETH_00122
SWS_EthSwt_00187 –> SRS_ETH_00087
SWS_EthSwt_00058 –> SRS_BSW_00171
SWS_EthSwt_91012, SWS_EthSwt_91013 –> SRS_ETH_00126
SWS_EthSwt_91014, SWS_EthSwt_91015, SWS_EthSwt_91016, SWS_EthSwt_91018, SWS_EthSwt_91019, SWS_EthSwt_91021, SWS_EthSwt_91022, SWS_EthSwt_91022, SWS_EthSwt_91023, SWS_EthSwt_91024, SWS_EthSwt_91025, SWS_EthSwt_91026, SWS_EthSwt_91027 –> SRS_Eth_00123
SWS_EthSwt_00117, SWS_EthSwt_00203 –> SRS_ETH_00119, SRS_ETH_00087
SWS_EthSwt_00193 –> SRS_ETH_00122, SRS_ETH_00087
SWS_EthSwt_00114 –> SRS_BSW_00433
SWS_EthSwt_00098 –> SRS_Eth_00122, SRS_ETH_00118, SRS_ETH_00119, SRS_ETH_00120, SRS_ETH_00087, SRS_ETH_00125, SRS_BSW_00375
–Last change on issue 76248 comment 13–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.267 Specification Item SWS_EthSwt_91013

Trace References:

SRS_ETH_00126

Content:

Service name:	EthSwt_SetForwardingModeEthSwt_SetForwardingMode	
Syntax:	Std_ReturnType EthSwt_SetForwardingMode(uint8 SwitchIdx, boolean mode)	
Service ID[hex]:	0x32	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	SwitchIdxEthSwt_SetForwarding Mode.SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	modeEthSwt_SetForwardingMode.mode	True Forewarding enabled, False Forwarding disabled
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: stopping of frame forwarding succeeded, E_NOT_OK: stopping of frame forwarding not succeeded.
Description:	Configures switch to start or stop forwarding for all ports. This API call may be used during switch configuration verification.	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76248: [EthSwt]Remove references to non existing requirements (SRS, SWS)

  Problem description:

  Some review findings brought up references to non existing requirements. Please check and remove the references and if needed reformulate the requirements.

  Agreed solution:

  SWS_EthSwt_00123 –> SRS_BSW_00406
  SWS_EthSwt_00165 -> SRS_BSW_00395
  SWS_EthSwt_00227 -> SRS_ETH_00087
  SWS_EthSwt_91017, SWS_EthSwt_91020, –> SRS_ETH_00123
  SWS_EthSwt_00006 –> SRS_BSW_00101

SWS_EthSwt_00010 left blank as SRS doe not yet provide a requirement here but maybe with RFC # 75170 AUTOSAR will introduce an SRS-item after all.
SWS_EthSwt_00376, SWS_EthSwt_00374, SWS_EthSwt_00375 –> SRS-Missing! same as for SWS_EThSwt_00010
SWS_EthSwt_00018, SRS_EthSwt_00025 –> SRS_ETH_00118
SWS_EthSwt_00031 –> SRS_ETH_00087
SWS_EthSwt_00037 –> SRS_ETH_00119
SWS_EthSwt_00044, SWS_EthSwt_00051 –> SRS_ETH_00118
SWS_EthSwt_00060 –> SRS_ETH_00087
SWS_EthSwt_00111 –> erase SRS_ETH_00086
SWS_EthSwt00079 –>SRS_ETH_00119
SWS_EthSwt_00206, SWS_EthSwt_00211, SWS_EthSwt_00216, SWS_EthSwt_00221 –>SRS_Eth_00120
SWS_EthSwt_00172 –>SRS_ETH_00121, SRS_ETH_00114
SWS_EthSwt_00086, SWS_EthSwt_00091, SWS_EthSwt_00182, SWS_EthSwt_00125 –> SRS_ETH_00087, SRS_ETH_00122
SWS_EthSwt_00187 –> SRS_ETH_00087
SWS_EthSwt_00058 –> SRS_BSW_00171
SWS_EthSwt_91012, SWS_EthSwt_91013 –> SRS_ETH_00126
SWS_EthSwt_91014, SWS_EthSwt_91015, SWS_EthSwt_91016, SWS_EthSwt_91018, SWS_EthSwt_91019, SWS_EthSwt_91021, SWS_EthSwt_91022, SWS_EthSwt_91022, SWS_EthSwt_91023, SWS_EthSwt_91024, SWS_EthSwt_91025, SWS_EthSwt_91026, SWS_EthSwt_91027 –> SRS_Eth_00123
SWS_EthSwt_00117, SWS_EthSwt_00203 –> SRS_ETH_00119, SRS_ETH_00087
SWS_EthSwt_00193 –> SRS_ETH_00122, SRS_ETH_00087
SWS_EthSwt_00114 –> SRS_BSW_00433
SWS_EthSwt_00098 –> SRS_Eth_00122, SRS_ETH_00118, SRS_ETH_00119, SRS_ETH_00120, SRS_ETH_00087, SRS_ETH_00125, SRS_BSW_00375
–Last change on issue 76248 comment 13–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.268 Specification Item SWS_EthSwt_91014

Trace References:

SRS_Eth_00123

Content:

Service name:	EthSwt_GetPortSignalQualityEthSwt_GetPortSignalQuality	
Syntax:	Std_ReturnType EthSwt_GetPortSignalQuality(uint8 SwitchIdx, uint8 PortIdx, uint8* uint32* SignalQualityPtr)	
Service ID[hex]:	0x33	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	SwitchIdxEthSwt_GetPortSignal Quality.SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	PortIdxEthSwt_GetPortSignal Quality.PortIdx	Index of the port at the addressed switch
Parameters (inout):	None	
Parameters (out):	SignalQualityPtrEthSwt_GetPortSignal Quality.SignalQualityPtr	Pointer to the memory where the signal quality in percent shall be stored.
Return value:	Std_ReturnType	E_OK: signal quality could be read. E_NOT_OK: signal quality could not be read (i.e. no Ethernet transceiver is available for this Ethernet switch port)
Description:	Obtains the current The function retrieves the signal quality of the link of the indexed Ethernet switch port. This function shall obtain the signal quality of Ethernet transceiver referenced by the EthSwtPort by calling EthTrcv_GetPhySignalQuality(). If no transceiver is referenced the signal quality shall be set to 0xFFFFFFFFFF.	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76248: [EthSwt]Remove references to non existing requirements (SRS, SWS)

  Problem description:

  Some review findings brought up references to non existing requirements. Please check and remove the references and if needed reformulate the requirements.

  Agreed solution:

  SWS_EthSwt_00123 –> SRS_BSW_00406
  SWS_EthSwt_00165 -> SRS_BSW_00395
  SWS_EthSwt_00227 -> SRS_ETH_00087
  SWS_EthSwt_91017, SWS_EthSwt_91020, –> SRS_ETH_00123
  SWS_EthSwt_00006 –> SRS_BSW_00101
  SWS_EthSwt_00010 left blank as SRS doe not yet provide a requirement here but maybe with RFC # 75170 AUTOSAR will introduce an SRS-item after all.
  SWS_EthSwt_00376, SWS_EthSwt_00374, SWS_EthSwt_00375 –> SRS-Missing! same as for SWS_EThSwt_00010
  SWS_EthSwt_00018, SRS_EthSwt_00025 –> SRS_ETH_00118

SWS_EthSwt_00031 –> SRS_ETH_00087
SWS_EthSwt_00037 –> SRS_ETH_00119
SWS_EthSwt_00044, SWS_EthSwt_00051 –> SRS_ETH_00118
SWS_EthSwt_00060 –> SRS_ETH_00087
SWS_EthSwt_00111 –> erase SRS_ETH_00086
SWS_EthSwt00079 –>SRS_ETH_00119
SWS_EthSwt_00206, SWS_EthSwt_00211, SWS_EthSwt_00216, SWS_EthSwt_00221 –>SRS_Eth_00120
SWS_EthSwt_00172 –>SRS_ETH_00121, SRS_ETH_00114
SWS_EthSwt_00086, SWS_EthSwt_00091, SWS_EthSwt_00182, SWS_EthSwt_00125 –> SRS_ETH_00087, SRS_ETH_00122
SWS_EthSwt_00187 –> SRS_ETH_00087
SWS_EthSwt_00058 –> SRS_BSW_00171
SWS_EthSwt_91012, SWS_EthSwt_91013 –> SRS_ETH_00126
SWS_EthSwt_91014, SWS_EthSwt_91015, SWS_EthSwt_91016, SWS_EthSwt_91018, SWS_EthSwt_91019, SWS_EthSwt_91021, SWS_EthSwt_91022, SWS_EthSwt_91022, SWS_EthSwt_91023, SWS_EthSwt_91024, SWS_EthSwt_91025, SWS_EthSwt_91026, SWS_EthSwt_91027 –> SRS_Eth_00123
SWS_EthSwt_00117, SWS_EthSwt_00203 –> SRS_ETH_00119, SRS_ETH_00087
SWS_EthSwt_00193 –> SRS_ETH_00122, SRS_ETH_00087
SWS_EthSwt_00114 –> SRS_BSW_00433
SWS_EthSwt_00098 –> SRS_Eth_00122, SRS_ETH_00118, SRS_ETH_00119, SRS_ETH_00120, SRS_ETH_00087, SRS_ETH_00125, SRS_BSW_00375
–Last change on issue 76248 comment 13–

BW-C-Level:

Application	Specification	Bus
1	1	1

- RfC #77628: [EthSwt] Behaviour of certain APIs for ports without EthSwtPortTrcvRef (part I)

  Problem description:

  The following APIs have partly a description regarding the handling of ports without EthSwtPortTrcvRef:
  EthSwt_GetPortSignalQuality
  EthSwt_GetPortIdentifier
  EthSwt_SetPortTestMode
  EthTrcv_SetPhyTestMode
  EthSwt_SetPortTxMode
  EthSwt_GetPortCableDiagnosticsResult

The description should be adjusted and harmonized with related requirements.
–Last change on issue 77628 comment 2–

Agreed solution:

=== EthSwt ===
~[SWS_EthSwt_91014]EthSwt_GetPortSignalQuality
Description:
The function retrieves the signal quality of the link of the indexed Ethernet switch port.
~[SWS_EthSwt_00293]
The function EthSwt_GetPortSignalQuality() shall obtain the signal quality by calling the function EthTrcv_GetPhySignalQuality() of the referenced Ethernet Transceiver Driver. If the current signal quality is not available, the signal quality shall be set to 0xFF.
-[SWS_EthSwt_00298]as # 77349 introduces a general req

~[SWS_EthSwt_91015]EthSwt_GetPortIdentifier
Return value
E_NOT_OK: organizationally unique identifier of the Ethernet transceiver could not be obtained (i.e. OUI is not available)
Description:
This function retrieves the OUI (24 bit) of the indexed Ethernet switch port.

~[SWS_EthSwt_00299]
The function EthSwt_GetPortIdentifier() shall return the value of the organizationally unique identifier (OUI 24 bit) of the indexed Ethernet switch port that is connected to the indexed Ethernet switch. It shall set the 8 most significant bits of the OUI to 0xFFxxxxxx. If the Ethernet switch port references an Ethernet transceiver, the function shall obtain the OUI by calling the function EthTrcv_GetPhyIdentifier() and set the 8 most significant bits of the OUI to 0x00xxxxxx.

SWS_EthSwt_xxxxx] If neither the Ethernet switch port nor the Ethernet Transceiver Driver can provide an OUI the function EthSwt_GetPortIdentifier() shall return E_NOT_OK.

-[SWS_EthSwt_00304]as # 77349 introduces a general req

~[SWS_EthSwt_91029] EthSwt_SetPortTestMode
Description:
Activates a given test mode of the indexed Ethernet switch port.

~[SWS_EthSwt_00328]
The function EthSwt_SetPortTestMode shall forward the call with the given test mode by calling the function EthTrcv_SetPhyTestMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00333]as # 77349 introduces a general req

~[SWS_EthSwt_91023] EthSwt_SetPortLoopbackMode
Description:
Activates a given test loop-back mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00334]
The function EthSwt_SetPortLoopbackMode() shall forward the call with the given loop-back mode by calling the function EthTrcv_SetPhyLoopbackMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00339] as # 77349 introduces a general req

~[SWS_EthSwt_91024] EthSwt_SetPortTxMode
Description:
Activates a given transmission mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00340]
The function EthSwt_SetPortTxMode() shall forward the call with the given transmission mode by calling the function EthTrcv_SetPhyTxMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00345] as # 77349 introduces a general req

~[SWS_EthSwt_91025]EthSwt_GetPortCableDiagnosticsResult
Description:
Retrieves the cable diagnostics result of the indexed Ethernet switch port respectively the referenced Ethernet Transceiver Driver.

~[SWS_EthSwt_00346]
The function EthSwt_GetPortCableDiagnosticsResult() shall obtain the cable diagnostics result by calling the function EthTrcv_GetCableDiagnosticsResult() of the referenced Ethernet Transceiver Driver. If no Ethernet transceiver is referenced by the Ethernet switch port and development error detection is not enabled, the cable diagnostic result shall be set to ETHTRCV_CABLEDIAG_OK.
-[SWS_EthSwt_00351] as # 77349 introduces a general req

=== EthTrcv ===
add certain parameter to enable/disable API functions:
+ SWS item ECUC_EthTrcv_xxxx1
Name EthTrcvGetPhySignalQualityApi

Description Enables / Disables EthTrcv_GetPhySignalQuality API

+ SWS item ECUC_EthTrcv_xxxx2
Name EthTrcvGetPhyIdentifierApi
Description Enables / Disables EthTrcv_GetPhyIdentifier API

+ SWS item ECUC_EthTrcv_xxxx3
Name EthTrcvSetPhyTestModeApi
Description Enables / Disables EthTrcv_SetPhyTestMode API

+ SWS item ECUC_EthTrcv_xxxx4
Name EthTrcvSetPhyTxModeApi
Description Enables / Disables EthTrcv_SetPhyTxMode API

+ SWS item ECUC_EthTrcv_xxxx5
Name EthTrcvGetCableDiagnosticsResultApi
Description Enables / Disables EthTrcv_GetCableDiagnosticsResult API

add the following specification to the configuration parameter above:
Multiplicity 1
Type EcucBooleanParamDef
Default value –
Post-Build Variant Value false
Value Configuration
Class Pre-compile time X
All Variants Link time –
Post-build time –
Scope / Dependency scope: local
–Last change on issue 77628 comment 31–

BW-C-Level:

Application	Specification	Bus
1	4	1

- RfC #77676: [EthIf][EthSwt][EthTrcv] Complete handling to determine signal quality

  Problem description:

  AR4.3.0 introduce APIs EthSwt_GetPortSignalQuality and EthTrcv_GetPhySignalQuality. This API's return the result of the signal quality meassured by a dedicated Ethernet transceiver. But there is a lack of how the meassurement for the signal quality shall be triggert respectively handled.

  Agreed solution:

=== EthTrcv ===
~[SWS_EthTrcv_91001] EthTrcv_GetPhySignalQuality
~ uint8* SignalQualityPtr change to uint32* SignalQualityPtr
~ change the description of parameter SignalQualityPtr to "Pointer to the memory where the signal quality shall be stored."

=== EthSwt ===
~[SWS_EthSwt_91014] EthSwt_GetPortSignalQuality
~ uint8* SignalQualityPtr change to uint32* SignalQualityPtr
~ change the description of parameter SignalQualityPtr to "Pointer to the memory where the signal quality shall be stored."
~ change the description: "...If no transceiver is referenced the signal quality shall be set to 0xFFFFFFFF."
~[SWS_EthSwt_00293] change "0xFF" to "0xFFFFFFFF"

=== EthIf ===
~ch.8.2
+[EthIf_xxxxx1] EthIf_SignalQualityResultType
Type Structure
Element uint32 HighestSignalQuality the highest signal quality of a link since last clear
Element uint32 LowestSignalQuality the lowest link signal quality of a link since last clear
Element uint32 ActualSignalQuality the actual signal quality

~ch. 8.3
~ [SWS_EthIf_91019] set EthIf_GetPhySignalQuality to "deprecated"

+[EthIf_xxxxx2] EthIf_GetTrcvSignalQuality
Syntax: Std_ReturnType EthIf_GetTrcvSignalQuality(
uint8 TrcvIdx,
EthIf_SignalQualityResultType* ResultPtr)
Sync/Async: Synchronous
Reentrancy: Reentrant for different TrcvIdx. Non reentrant for the same TrcvIdx.
Parameters (in): TrcvIdx Index of the transceiver within the context of the Ethernet Interface
Parameters (inout): None
Parameters (out): ResultPtr Pointer to the memory where the signal quality in percent shall be stored.
Return value: Std_ReturnType
E_OK: The signal quality retrieved successfully
E_NOT_OK: The signal quality not retrieved successfully

Description: Retrieves the signal quality of the link of the given Ethernet transceiver

+[EthIf_xxxxx3] EthIf_GetSwitchPortSignalQuality
Syntax: Std_ReturnType EthIf_GetSwitchPortSignalQuality(
uint8 SwitchIdx,
uint8 SwitchPortIdx,
EthIf_SignalQualityResultType* ResultPtr)
Sync/Async: Synchronous
Reentrancy: Reentrant for different Ethernet switch indexes and Ethernet Switch port indexes. Non reentrant for the same SwitchPortIdx.
Parameters (in): SwitchIdx Index of the Ethernet switch within the context of the Ethernet Interface
SwitchPortIdx Index of the Ethernet switch port within the context of the Ethernet Interface
Parameters (inout): None
Parameters (out): ResultPtr Pointer to the memory where the signal quality in percent shall be stored.
Return value: Std_ReturnType
E_OK: The signal quality retrieved successfully
E_NOT_OK: The signal quality not retrieved successfully
Description: Retrieves the signal quality of the link of the given Ethernet switch port

+[EthIf_xxxxx4] EthIf_ClearTrcvSignalQuality
Syntax: Std_ReturnType EthIf_ClearTrcvSignalQuality(
uint8 TrcvIdx)
Sync/Async: Synchronous
Reentrancy: Reentrant for different TrcvIdx. Non reentrant for the same TrcvIdx.
Parameters (in): TrcvIdx Index of the transceiver within the context of the Ethernet Interface
Parameters (inout): None
Parameters (out): None
Return value: Std_ReturnType
E_OK: The signal quality cleared successfully
E_NOT_OK: The signal quality cleared not successfully
Description: Clear the stored signal quality of the link of the given Ethernet transceiver

+[EthIf_xxxxx5] EthIf_ClearSwitchPortSignalQuality
Syntax: Std_ReturnType EthIf_ClearSwitchPortSignalQuality(
uint8 SwitchIdx,
uint8 SwitchPortIdx)
Sync/Async: Synchronous
Reentrancy: Reentrant for different Ethernet switch indexes and Ethernet Switch

port indexes. Non reentrant for the same SwitchPortIdx.
Parameters (in): SwitchIdx Index of the Ethernet switch within the context of the Ethernet Interface
SwitchPortIdx Index of the Ethernet switch port within the context of the Ethernet Interface
Parameters (inout): None
Parameters (out): None
Return value: Std_ReturnType
E_OK: The signal quality cleared successfully
E_NOT_OK: The signal quality cleared not successfully
Description: Clear the stored signal quality of the link of the given Ethernet switch port

~ch. 8.5 Scheduled functions
~ SWS_EthIf_91104 EthIf_MainFunctionState
~Description: The function is polling different communication hardware (Ethernet transceiver, Ethernet switch ports) related information, e.g. link state, signal quality.

+[EthIf_xxxx12] The EthIf_MainFunctionState shall poll Ethernet communication hardware related information with the period of EthIfMainFunctionStatePeriod.

+[EthIf_xxxxx6] For each Ethernet switch port where a link state of ETHSWT_LINK_STATE_ACTIVE is yielded and references an Ethernet Transceiver the function shall poll the signal quality by calling EthSwt_GetPortSignalQuality().

+[EthIf_xxxxx7] For each Ethernet transceiver where a link state of ETHTRCV_LINK_STATE_ACTIVE is yielded the function shall poll the signal quality by calling EthTrcv_GetPhySignalQuality().

+[EthIf_xxxxx9] The obtained signal quality value shall be stored as type of EthIf_SignalQualityResultType. The value shall always be stored as ActualSignalQuality. If the obtained signal quality is higher than the stored highest signal quality (HighestSignalQuality), then HighestSignalQuality shall be updated with the obtained signal quality. If the obtained signal quality is lower than the lowest signal quality (LowestSignalQuality), then LowestSignalQuality shall be updated with the obtained signal quality.

~ch. 10 Configuration specification
add the following parameters to section 10.1.2 EthIfGeneral
+ [ECUC_EthIf_xxxxx1] EthIfEnableSignalQualityApi
+ Description: enable/disable the APIs read and clear the signal quality
+ Multiplicity : 1

+ Type : EcucBooleanParamDef
+ Scope / Dependency : scope: local

+ [ECUC_EthIf_xxxxx2] EthIfSignalQualityCheckPeriod
+ Description : Specifies the period in units of seconds in which the signal quality it polled in the context of EthIf_MainfunctionState. The value shall be an integral multiple of EthIfMainFunctionStatePeriod.
+ Multiplicity : 0..1
+ Type : EcucFloatParamDef
+ Scope / Dependency : scope: local
dependency: If this parameter is defined, the EthIf_MainFunctionState shall be generated and parameter EthIfEnableSignalQualityApi shall be set to TRUE.
–Last change on issue 77676 comment 38–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.269 Specification Item SWS_EthSwt_91015

Trace References:

SRS_Eth_00123

Content:

Service name:	EthSwt_GetPortIdentifierEthSwt_GetPortIdentifier	
Syntax:	Std_ReturnType EthSwt_GetPortIdentifier(uint8 SwitchIdx, uint8 PortIdx, uint32* OrgUniqueIdPtr, uint8* ModelNrPtr, uint8* RevisionNrPtr)	
Service ID[hex]:	0x34	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	SwitchIdxEthSwt_GetPort Identifier.SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	PortIdxEthSwt_GetPortIdentifier.PortIdx	Index of the port at the addressed switch
Parameters (inout):	None	

Parameters (out):	OrgUniqueIdPtrEthSwt_GetPortIdentifier.OrgUniqueIdPtr	Pointer to the memory where the Organizationally Unique Identifier (OUI) shall be stored.
	ModelNrPtrEthSwt_GetPortIdentifier.ModelNrPtr	Pointer to the memory where the Manufacturer's Model Number shall be stored.
	RevisionNrPtrEthSwt_GetPortIdentifier.RevisionNrPtr	Pointer to the memory where the Revision Number shall be stored.
Return value:	Std_ReturnType	E_OK: organizationally unique identifier of the Ethernet transceiver could be read. E_NOT_OK: organizationally unique identifier of the Ethernet transceiver could not be read obtained (i.e. no Ethernet transceiver is available for this Ethernet switch port)OUI is not available).
Description:	This function shall provide the OUI of Ethernet transceiver referenced by the EthSwt Port. The function shall call EthTrcv_GetPhyIdentifier. The OUI has a size of retrieves the OUI (24 bit. If a Ethernet transceiver can provide the OUI the 8 most significant bits of the OUI shall be set to 0x00xxxxxx. If no tranceiver is referenced by the Eth SwtPort the 8 most significant bits of the OUI shall be set to 0xFFxxxxxx) of the indexed Ethernet switch port.	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76248: [EthSwt]Remove references to non existing requirements (SRS, SWS)

  Problem description:

  Some review findings brought up references to non existing requirements. Please check and remove the references and if needed reformulate the requirements.

  Agreed solution:

  SWS_EthSwt_00123 –> SRS_BSW_00406
  SWS_EthSwt_00165 -> SRS_BSW_00395
  SWS_EthSwt_00227 -> SRS_ETH_00087
  SWS_EthSwt_91017, SWS_EthSwt_91020, –> SRS_ETH_00123
  SWS_EthSwt_00006 –> SRS_BSW_00101
  SWS_EthSwt_00010 left blank as SRS doe not yet provide a requirement here but maybe with RFC # 75170 AUTOSAR will introduce an SRS-item after all.
  SWS_EthSwt_00376, SWS_EthSwt_00374, SWS_EthSwt_00375 –> SRS-Missing! same as for SWS_EThSwt_00010
  SWS_EthSwt_00018, SRS_EthSwt_00025 –> SRS_ETH_00118
  SWS_EthSwt_00031 –> SRS_ETH_00087
  SWS_EthSwt_00037 –> SRS_ETH_00119
  SWS_EthSwt_00044, SWS_EthSwt_00051 –> SRS_ETH_00118
  SWS_EthSwt_00060 –> SRS_ETH_00087
  SWS_EthSwt_00111 –> erase SRS_ETH_00086

SWS_EthSwt00079 –>SRS_ETH_00119
SWS_EthSwt_00206, SWS_EthSwt_00211, SWS_EthSwt_00216, SWS_EthSwt_00221 –>SRS_Eth_00120
SWS_EthSwt_00172 –>SRS_ETH_00121, SRS_ETH_00114
SWS_EthSwt_00086, SWS_EthSwt_00091, SWS_EthSwt_00182, SWS_EthSwt_00125 –> SRS_ETH_00087, SRS_ETH_00122
SWS_EthSwt_00187 –> SRS_ETH_00087
SWS_EthSwt_00058 –> SRS_BSW_00171
SWS_EthSwt_91012, SWS_EthSwt_91013 –> SRS_ETH_00126
SWS_EthSwt_91014, SWS_EthSwt_91015, SWS_EthSwt_91016, SWS_EthSwt_91018, SWS_EthSwt_91019, SWS_EthSwt_91021, SWS_EthSwt_91022, SWS_EthSwt_91022, SWS_EthSwt_91023, SWS_EthSwt_91024, SWS_EthSwt_91025, SWS_EthSwt_91026, SWS_EthSwt_91027 –> SRS_Eth_00123
SWS_EthSwt_00117, SWS_EthSwt_00203 –> SRS_ETH_00119, SRS_ETH_00087
SWS_EthSwt_00193 –> SRS_ETH_00122, SRS_ETH_00087
SWS_EthSwt_00114 –> SRS_BSW_00433
SWS_EthSwt_00098 –> SRS_Eth_00122, SRS_ETH_00118, SRS_ETH_00119, SRS_ETH_00120, SRS_ETH_00087, SRS_ETH_00125, SRS_BSW_00375
–Last change on issue 76248 comment 13–

BW-C-Level:

Application	Specification	Bus
1	1	1

- RfC #77628: [EthSwt] Behaviour of certain APIs for ports without EthSwtPortTrcvRef (part I)

  Problem description:

  The following APIs have partly a description regarding the handling of ports without EthSwtPortTrcvRef:
  EthSwt_GetPortSignalQuality
  EthSwt_GetPortIdentifier
  EthSwt_SetPortTestMode
  EthTrcv_SetPhyTestMode
  EthSwt_SetPortTxMode
  EthSwt_GetPortCableDiagnosticsResult

  The description should be adjusted and harmonized with related requirements.
  –Last change on issue 77628 comment 2–

  Agreed solution:

=== EthSwt ===
~[SWS_EthSwt_91014]EthSwt_GetPortSignalQuality
Description:
The function retrieves the signal quality of the link of the indexed Ethernet switch port.
~[SWS_EthSwt_00293]
The function EthSwt_GetPortSignalQuality() shall obtain the signal quality by calling the function EthTrcv_GetPhySignalQuality() of the referenced Ethernet Transceiver Driver. If the current signal quality is not available, the signal quality shall be set to 0xFF.
-[SWS_EthSwt_00298]as # 77349 introduces a general req

~[SWS_EthSwt_91015]EthSwt_GetPortIdentifier
Return value
E_NOT_OK: organizationally unique identifier of the Ethernet transceiver could not be obtained (i.e. OUI is not available)
Description:
This function retrieves the OUI (24 bit) of the indexed Ethernet switch port.

~[SWS_EthSwt_00299]
The function EthSwt_GetPortIdentifier() shall return the value of the organizationally unique identifier (OUI 24 bit) of the indexed Ethernet switch port that is connected to the indexed Ethernet switch. It shall set the 8 most significant bits of the OUI to 0xFFxxxxxx. If the Ethernet switch port references an Ethernet transceiver, the function shall obtain the OUI by calling the function EthTrcv_GetPhyIdentifier() and set the 8 most significant bits of the OUI to 0x00xxxxxx.

SWS_EthSwt_xxxxx] If neither the Ethernet switch port nor the Ethernet Transceiver Driver can provide an OUI the function EthSwt_GetPortIdentifier() shall return E_NOT_OK.

-[SWS_EthSwt_00304]as # 77349 introduces a general req

~[SWS_EthSwt_91029] EthSwt_SetPortTestMode
Description:
Activates a given test mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00328]
The function EthSwt_SetPortTestMode shall forward the call with the given test mode by calling the function EthTrcv_SetPhyTestMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00333]as # 77349 introduces a general req

~[SWS_EthSwt_91023] EthSwt_SetPortLoopbackMode
Description:
Activates a given test loop-back mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00334]
The function EthSwt_SetPortLoopbackMode() shall forward the call with the given loop-back mode by calling the function EthTrcv_SetPhyLoopbackMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00339] as # 77349 introduces a general req

~[SWS_EthSwt_91024] EthSwt_SetPortTxMode
Description:
Activates a given transmission mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00340]
The function EthSwt_SetPortTxMode() shall forward the call with the given transmission mode by calling the function EthTrcv_SetPhyTxMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00345] as # 77349 introduces a general req

~[SWS_EthSwt_91025]EthSwt_GetPortCableDiagnosticsResult
Description:
Retrieves the cable diagnostics result of the indexed Ethernet switch port respectively the referenced Ethernet Transceiver Driver.

~[SWS_EthSwt_00346]
The function EthSwt_GetPortCableDiagnosticsResult() shall obtain the cable diagnostics result by calling the function EthTrcv_GetCableDiagnosticsResult() of the referenced Ethernet Transceiver Driver. If no Ethernet transceiver is referenced by the Ethernet switch port and development error detection is not enabled, the cable diagnostic result shall be set to ETHTRCV_CABLEDIAG_OK.
-[SWS_EthSwt_00351] as # 77349 introduces a general req

=== EthTrcv ===
add certain parameter to enable/disable API functions:
+ SWS item ECUC_EthTrcv_xxxx1
Name EthTrcvGetPhySignalQualityApi
Description Enables / Disables EthTrcv_GetPhySignalQuality API

+ SWS item ECUC_EthTrcv_xxxx2
Name EthTrcvGetPhyIdentifierApi
Description Enables / Disables EthTrcv_GetPhyIdentifier API

+ SWS item ECUC_EthTrcv_xxxx3
Name EthTrcvSetPhyTestModeApi
Description Enables / Disables EthTrcv_SetPhyTestMode API

+ SWS item ECUC_EthTrcv_xxxx4
Name EthTrcvSetPhyTxModeApi
Description Enables / Disables EthTrcv_SetPhyTxMode API

+ SWS item ECUC_EthTrcv_xxxx5
Name EthTrcvGetCableDiagnosticsResultApi
Description Enables / Disables EthTrcv_GetCableDiagnosticsResult API

add the following specification to the configuration parameter above:
Multiplicity 1
Type EcucBooleanParamDef
Default value –
Post-Build Variant Value false
Value Configuration
Class Pre-compile time X
All Variants Link time –
Post-build time –
Scope / Dependency scope: local
–Last change on issue 77628 comment 31–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.270 Specification Item SWS_EthSwt_91016

Trace References:

SRS_Eth_00123

Content:

Service name:	EthSwt_GetSwitchIdentifierEthSwt_GetSwitchIdentifier
Syntax:	Std_ReturnType EthSwt_GetSwitchIdentifier(uint8 SwitchIdx, uint32* OrgUniqueIdPtr)
Service ID[hex]:	0x35
Sync/Async:	Synchronous

Reentrancy:	Non Reentrant	
Parameters (in):	SwitchIdxEthSwt_GetSwitchIdentifier.SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
Parameters (inout):	None	
Parameters (out):	OrgUniqueIdPtrEthSwt_GetSwitchIdentifier.OrgUniqueIdPtr	Pointer to the memory where the Organizationally Unique Identifier shall be stored.
Return value:	Std_ReturnType	E_OK: organizationally unique identifier of the Ethernet switch could be read. E_NOT_OK: organizationally unique identifier of the Ethernet switch could not be read (i.e. no OUI is available for this Ethernet switch)
Description:	Obtain the Organizationally Unique Identifier that is given by the IEEE of the indexed Ethernet switch. This function shall provide the OUI of Ethernet switch. The OUI has a size of 24 bit. If a ethernet switch can provide the OUI the 8 most significant bits of the OUI shall be set to 0x00xxxxxx. If a Ethernet switch can not provide the OUI the 8 most significant bits of the OUI shall be set to 0xFFxxxxxx.	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76248: [EthSwt]Remove references to non existing requirements (SRS, SWS)

  Problem description:

  Some review findings brought up references to non existing requirements. Please check and remove the references and if needed reformulate the requirements.

  Agreed solution:

  SWS_EthSwt_00123 –> SRS_BSW_00406
  SWS_EthSwt_00165 -> SRS_BSW_00395
  SWS_EthSwt_00227 -> SRS_ETH_00087
  SWS_EthSwt_91017, SWS_EthSwt_91020, –> SRS_ETH_00123
  SWS_EthSwt_00006 –> SRS_BSW_00101
  SWS_EthSwt_00010 left blank as SRS doe not yet provide a requirement here but maybe with RFC # 75170 AUTOSAR will introduce an SRS-item after all.
  SWS_EthSwt_00376, SWS_EthSwt_00374, SWS_EthSwt_00375 –> SRS-Missing! same as for SWS_EThSwt_00010
  SWS_EthSwt_00018, SRS_EthSwt_00025 –> SRS_ETH_00118
  SWS_EthSwt_00031 –> SRS_ETH_00087
  SWS_EthSwt_00037 –> SRS_ETH_00119
  SWS_EthSwt_00044, SWS_EthSwt_00051 –> SRS_ETH_00118
  SWS_EthSwt_00060 –> SRS_ETH_00087
  SWS_EthSwt_00111 –> erase SRS_ETH_00086
  SWS_EthSwt00079 –>SRS_ETH_00119
  SWS_EthSwt_00206, SWS_EthSwt_00211, SWS_EthSwt_00216, SWS_EthSwt_00221 –>SRS_Eth_00120

SWS_EthSwt_00172 –>SRS_ETH_00121, SRS_ETH_00114
SWS_EthSwt_00086, SWS_EthSwt_00091, SWS_EthSwt_00182,
SWS_EthSwt_00125 –> SRS_ETH_00087, SRS_ETH_00122
SWS_EthSwt_00187 –> SRS_ETH_00087
SWS_EthSwt_00058 –> SRS_BSW_00171
SWS_EthSwt_91012, SWS_EthSwt_91013 –> SRS_ETH_00126
SWS_EthSwt_91014, SWS_EthSwt_91015, SWS_EthSwt_91016,
SWS_EthSwt_91018, SWS_EthSwt_91019, SWS_EthSwt_91021,
SWS_EthSwt_91022, SWS_EthSwt_91022, SWS_EthSwt_91023,
SWS_EthSwt_91024, SWS_EthSwt_91025, SWS_EthSwt_91026,
SWS_EthSwt_91027 –> SRS_Eth_00123
SWS_EthSwt_00117, SWS_EthSwt_00203 –> SRS_ETH_00119,
SRS_ETH_00087
SWS_EthSwt_00193 –> SRS_ETH_00122, SRS_ETH_00087
SWS_EthSwt_00114 –> SRS_BSW_00433
SWS_EthSwt_00098 –> SRS_Eth_00122, SRS_ETH_00118, SRS_ETH_00119,
SRS_ETH_00120, SRS_ETH_00087, SRS_ETH_00125, SRS_BSW_00375
–Last change on issue 76248 comment 13–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.271 Specification Item SWS_EthSwt_91017

Trace References:

SRS_ETH_00123

Content:

Name:	EthSwt_PortMirrorCfgTypeEthSwt_PortMirrorCfgType
Type:	Structure

Element:	uint8	SetSelectionEthSwt_PortMirrorCfgType.SetSelection	specifies the type selection 0x00== free configuration, 0x01 - 0x0FF == set number
	uint8	TrafficDirectionEthSwt_PortMirrorCfgType.TrafficDirection	specifies whether the direction is Ingress (0) or Egress (1)
	uint8[6]	srcMacAddrFilterEthSwt_PortMirrorCfgType.srcMacAddrFilter	Specifies the source MAC address [0..255,0..255,0..255,0..255,0..255,0..255] that should be mirrored
	uint8[6]	dstMacAddrFilterEthSwt_PortMirrorCfgType.dstMacAddrFilter	Specifies the source MAC address [0..255,0..255,0..255,0..255,0..255,0..255] that should be mirrored
	uint16	VlanIdFilterEthSwt_PortMirrorCfgType.VlanIdFilter	Specifies the VLAN address 0..65535 that should be mirrored
	uint8	MirroringPacketDividerEthSwt_PortMirrorCfgType.MirroringPacketDivider	Divider if only a subset of received frames should be mirrored. E.g. MirroringPacketDivider = 2 means every second frames is mirrored
	uint8	MirroringModeEthSwt_PortMirrorCfgType.MirroringMode	specifies the mode how the mirrored traffic should be tagged : 0x00 == No VLAN retagging; 0x01 == VLAN retagging; 0x03 == VLAN Double tagging
	uint16	MirroringTimeoutEthSwt_PortMirrorCfgType.MirroringTimeout	specifies a time constant in seconds after the mirroring configuration should be resumed (e.g. ensuring that mirroring is not endlessly active).
Description:	In case of port mirroring it shall be possible to set up port mirroring configurations. It shall be possible to store different filter packages in the Ethernet switch. The parameter SetSelection set the index for the filter package (0x01 .. 0xFF). Filter packages are accessible by the index. In case SetSelection is set to 0x00, the package use the a free configuration.		

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76248: [EthSwt]Remove references to non existing requirements (SRS, SWS)

  Problem description:

  Some review findings brought up references to non existing requirements. Please check and remove the references and if needed reformulate the requirements.

  Agreed solution:

  SWS_EthSwt_00123 –> SRS_BSW_00406
  SWS_EthSwt_00165 -> SRS_BSW_00395
  SWS_EthSwt_00227 -> SRS_ETH_00087

SWS_EthSwt_91017, SWS_EthSwt_91020, –> SRS_ETH_00123
SWS_EthSwt_00006 –> SRS_BSW_00101
SWS_EthSwt_00010 left blank as SRS doe not yet provide a requirement here but maybe with RFC # 75170 AUTOSAR will introduce an SRS-item after all.
SWS_EthSwt_00376, SWS_EthSwt_00374, SWS_EthSwt_00375 –> SRS-Missing! same as for SWS_EThSwt_00010
SWS_EthSwt_00018, SRS_EthSwt_00025 –> SRS_ETH_00118
SWS_EthSwt_00031 –> SRS_ETH_00087
SWS_EthSwt_00037 –> SRS_ETH_00119
SWS_EthSwt_00044, SWS_EthSwt_00051 –> SRS_ETH_00118
SWS_EthSwt_00060 –> SRS_ETH_00087
SWS_EthSwt_00111 –> erase SRS_ETH_00086
SWS_EthSwt00079 –>SRS_ETH_00119
SWS_EthSwt_00206, SWS_EthSwt_00211, SWS_EthSwt_00216, SWS_EthSwt_00221 –>SRS_Eth_00120
SWS_EthSwt_00172 –>SRS_ETH_00121, SRS_ETH_00114
SWS_EthSwt_00086, SWS_EthSwt_00091, SWS_EthSwt_00182, SWS_EthSwt_00125 –> SRS_ETH_00087, SRS_ETH_00122
SWS_EthSwt_00187 –> SRS_ETH_00087
SWS_EthSwt_00058 –> SRS_BSW_00171
SWS_EthSwt_91012, SWS_EthSwt_91013 –> SRS_ETH_00126
SWS_EthSwt_91014, SWS_EthSwt_91015, SWS_EthSwt_91016, SWS_EthSwt_91018, SWS_EthSwt_91019, SWS_EthSwt_91021, SWS_EthSwt_91022, SWS_EthSwt_91022, SWS_EthSwt_91023, SWS_EthSwt_91024, SWS_EthSwt_91025, SWS_EthSwt_91026, SWS_EthSwt_91027 –> SRS_Eth_00123
SWS_EthSwt_00117, SWS_EthSwt_00203 –> SRS_ETH_00119, SRS_ETH_00087
SWS_EthSwt_00193 –> SRS_ETH_00122, SRS_ETH_00087
SWS_EthSwt_00114 –> SRS_BSW_00433
SWS_EthSwt_00098 –> SRS_Eth_00122, SRS_ETH_00118, SRS_ETH_00119, SRS_ETH_00120, SRS_ETH_00087, SRS_ETH_00125, SRS_BSW_00375
–Last change on issue 76248 comment 13–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.272 Specification Item SWS_EthSwt_91018

Trace References:

SRS_Eth_00123

Content:

Service name:	EthSwt_WritePortMirrorConfigurationEthSwt_WritePortMirrorConfiguration	
Syntax:	Std_ReturnType EthSwt_WritePortMirrorConfiguration(uint8 SwitchIdx, uint8 PortIdx, EthSwt_PortMirrorCfgType* const PortMirrorConfigurationPtr)	
Service ID[hex]:	0x36	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	SwitchIdxEthSwt_WritePortMirror Configuration.SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	PortIdxEthSwt_WritePortMirror Configuration.PortIdx	Index of the port at the addressed switch
	PortMirrorConfigurationPtrEthSwt_Write PortMirrorConfiguration.PortMirror ConfigurationPtr	Pointer to the memory where the port configuration is stored.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: the port mirror configuration for the indexed Ethernet switch port was written. E_NOT_OK: the port mirror configuration for the indexed Ethernet switch port was not written. (i.e. indexed ethernet switch is not available)
Description:	The given mirror configuration shall be written to the given Ethernet switch port.	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76248: [EthSwt]Remove references to non existing requirements (SRS, SWS)

  Problem description:

  Some review findings brought up references to non existing requirements. Please check and remove the references and if needed reformulate the requirements.

  Agreed solution:

  SWS_EthSwt_00123 –> SRS_BSW_00406
  SWS_EthSwt_00165 -> SRS_BSW_00395
  SWS_EthSwt_00227 -> SRS_ETH_00087
  SWS_EthSwt_91017, SWS_EthSwt_91020, –> SRS_ETH_00123
  SWS_EthSwt_00006 –> SRS_BSW_00101
  SWS_EthSwt_00010 left blank as SRS doe not yet provide a requirement here but maybe with RFC # 75170 AUTOSAR will introduce an SRS-item after all.
  SWS_EthSwt_00376, SWS_EthSwt_00374, SWS_EthSwt_00375 –> SRS-Missing!

same as for SWS_EThSwt_00010
SWS_EthSwt_00018, SRS_EthSwt_00025 –> SRS_ETH_00118
SWS_EthSwt_00031 –> SRS_ETH_00087
SWS_EthSwt_00037 –> SRS_ETH_00119
SWS_EthSwt_00044, SWS_EthSwt_00051 –> SRS_ETH_00118
SWS_EthSwt_00060 –> SRS_ETH_00087
SWS_EthSwt_00111 –> erase SRS_ETH_00086
SWS_EthSwt00079 –>SRS_ETH_00119
SWS_EthSwt_00206, SWS_EthSwt_00211, SWS_EthSwt_00216, SWS_EthSwt_00221 –>SRS_Eth_00120
SWS_EthSwt_00172 –>SRS_ETH_00121, SRS_ETH_00114
SWS_EthSwt_00086, SWS_EthSwt_00091, SWS_EthSwt_00182, SWS_EthSwt_00125 –> SRS_ETH_00087, SRS_ETH_00122
SWS_EthSwt_00187 –> SRS_ETH_00087
SWS_EthSwt_00058 –> SRS_BSW_00171
SWS_EthSwt_91012, SWS_EthSwt_91013 –> SRS_ETH_00126
SWS_EthSwt_91014, SWS_EthSwt_91015, SWS_EthSwt_91016, SWS_EthSwt_91018, SWS_EthSwt_91019, SWS_EthSwt_91021, SWS_EthSwt_91022, SWS_EthSwt_91022, SWS_EthSwt_91023, SWS_EthSwt_91024, SWS_EthSwt_91025, SWS_EthSwt_91026, SWS_EthSwt_91027 –> SRS_Eth_00123
SWS_EthSwt_00117, SWS_EthSwt_00203 –> SRS_ETH_00119, SRS_ETH_00087
SWS_EthSwt_00193 –> SRS_ETH_00122, SRS_ETH_00087
SWS_EthSwt_00114 –> SRS_BSW_00433
SWS_EthSwt_00098 –> SRS_Eth_00122, SRS_ETH_00118, SRS_ETH_00119, SRS_ETH_00120, SRS_ETH_00087, SRS_ETH_00125, SRS_BSW_00375
–Last change on issue 76248 comment 13–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.273 Specification Item SWS_EthSwt_91019

Trace References:

SRS_Eth_00123

Content:

Service name:	EthSwt_ReadPortMirrorConfigurationEthSwt_ReadPortMirrorConfiguration

Syntax:	Std_ReturnType EthSwt_ReadPortMirrorConfiguration(uint8 SwitchIdx, uint8 PortIdx, EthSwt_PortMirrorCfgType* PortMirrorConfigurationPtr)	
Service ID[hex]:	0x37	
Sync/Async:	Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	SwitchIdxEthSwt_ReadPortMirror Configuration.SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	PortIdxEthSwt_ReadPortMirror Configuration.PortIdx	Index of the port at the addressed switch
Parameters (inout):	None	
Parameters (out):	PortMirrorConfigurationPtrEthSwt_Read PortMirrorConfiguration.PortMirror ConfigurationPtr	Pointer to the memory where the port configuration shall be stored.
Return value:	Std_ReturnType	E_OK: the port mirror configuration for the indexed Ethernet switch port was red successfully. E_NOT_OK: the port mirror configuration for the indexed Ethernet switch was not red successfully. (i.e. indexed Ethernet switch is not available)
Description:	Obtain the mirror configuration of the given Ethernet switch port.	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76248: [EthSwt]Remove references to non existing requirements (SRS, SWS)

  Problem description:

  Some review findings brought up references to non existing requirements. Please check and remove the references and if needed reformulate the requirements.

  Agreed solution:

  SWS_EthSwt_00123 –> SRS_BSW_00406
  SWS_EthSwt_00165 -> SRS_BSW_00395
  SWS_EthSwt_00227 -> SRS_ETH_00087
  SWS_EthSwt_91017, SWS_EthSwt_91020, –> SRS_ETH_00123
  SWS_EthSwt_00006 –> SRS_BSW_00101
  SWS_EthSwt_00010 left blank as SRS doe not yet provide a requirement here but maybe with RFC # 75170 AUTOSAR will introduce an SRS-item after all.
  SWS_EthSwt_00376, SWS_EthSwt_00374, SWS_EthSwt_00375 –> SRS-Missing! same as for SWS_EThSwt_00010
  SWS_EthSwt_00018, SRS_EthSwt_00025 –> SRS_ETH_00118
  SWS_EthSwt_00031 –> SRS_ETH_00087
  SWS_EthSwt_00037 –> SRS_ETH_00119

SWS_EthSwt_00044, SWS_EthSwt_00051 –> SRS_ETH_00118
SWS_EthSwt_00060 –> SRS_ETH_00087
SWS_EthSwt_00111 –> erase SRS_ETH_00086
SWS_EthSwt00079 –>SRS_ETH_00119
SWS_EthSwt_00206, SWS_EthSwt_00211, SWS_EthSwt_00216, SWS_EthSwt_00221 –>SRS_Eth_00120
SWS_EthSwt_00172 –>SRS_ETH_00121, SRS_ETH_00114
SWS_EthSwt_00086, SWS_EthSwt_00091, SWS_EthSwt_00182, SWS_EthSwt_00125 –> SRS_ETH_00087, SRS_ETH_00122
SWS_EthSwt_00187 –> SRS_ETH_00087
SWS_EthSwt_00058 –> SRS_BSW_00171
SWS_EthSwt_91012, SWS_EthSwt_91013 –> SRS_ETH_00126
SWS_EthSwt_91014, SWS_EthSwt_91015, SWS_EthSwt_91016, SWS_EthSwt_91018, SWS_EthSwt_91019, SWS_EthSwt_91021, SWS_EthSwt_91022, SWS_EthSwt_91022, SWS_EthSwt_91023, SWS_EthSwt_91024, SWS_EthSwt_91025, SWS_EthSwt_91026, SWS_EthSwt_91027 –> SRS_Eth_00123
SWS_EthSwt_00117, SWS_EthSwt_00203 –> SRS_ETH_00119, SRS_ETH_00087
SWS_EthSwt_00193 –> SRS_ETH_00122, SRS_ETH_00087
SWS_EthSwt_00114 –> SRS_BSW_00433
SWS_EthSwt_00098 –> SRS_Eth_00122, SRS_ETH_00118, SRS_ETH_00119, SRS_ETH_00120, SRS_ETH_00087, SRS_ETH_00125, SRS_BSW_00375
–Last change on issue 76248 comment 13–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.274 Specification Item SWS_EthSwt_91020

Trace References:

SRS_ETH_00123

Content:

Name:	EthSwt_PortMirrorStateTypeEthSwt_PortMirrorStateType
Type:	Enumeration

Range:	PORT_MIRROR_DISABLEDEth Swt_PortMirrorState Type.PORT_MIRROR_DISABLED	0x00	port mirroring disabled
	PORT_MIRROR_ENABLEDEth Swt_PortMirrorState Type.PORT_MIRROR_ENABLED	0x01	port mirroring enabled
Description:	It shall be possible to set and get the state (enable / disable) of the ehternet switch port.		

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76248: [EthSwt]Remove references to non existing requirements (SRS, SWS)

  Problem description:

  Some review findings brought up references to non existing requirements. Please check and remove the references and if needed reformulate the requirements.

  Agreed solution:

  SWS_EthSwt_00123 –> SRS_BSW_00406
  SWS_EthSwt_00165 -> SRS_BSW_00395
  SWS_EthSwt_00227 -> SRS_ETH_00087
  SWS_EthSwt_91017, SWS_EthSwt_91020, –> SRS_ETH_00123
  SWS_EthSwt_00006 –> SRS_BSW_00101
  SWS_EthSwt_00010 left blank as SRS doe not yet provide a requirement here but maybe with RFC # 75170 AUTOSAR will introduce an SRS-item after all.
  SWS_EthSwt_00376, SWS_EthSwt_00374, SWS_EthSwt_00375 –> SRS-Missing! same as for SWS_EThSwt_00010
  SWS_EthSwt_00018, SRS_EthSwt_00025 –> SRS_ETH_00118
  SWS_EthSwt_00031 –> SRS_ETH_00087
  SWS_EthSwt_00037 –> SRS_ETH_00119
  SWS_EthSwt_00044, SWS_EthSwt_00051 –> SRS_ETH_00118
  SWS_EthSwt_00060 –> SRS_ETH_00087
  SWS_EthSwt_00111 –> erase SRS_ETH_00086
  SWS_EthSwt00079 –>SRS_ETH_00119
  SWS_EthSwt_00206, SWS_EthSwt_00211, SWS_EthSwt_00216, SWS_EthSwt_00221 –>SRS_Eth_00120
  SWS_EthSwt_00172 –>SRS_ETH_00121, SRS_ETH_00114
  SWS_EthSwt_00086, SWS_EthSwt_00091, SWS_EthSwt_00182, SWS_EthSwt_00125 –> SRS_ETH_00087, SRS_ETH_00122
  SWS_EthSwt_00187 –> SRS_ETH_00087
  SWS_EthSwt_00058 –> SRS_BSW_00171
  SWS_EthSwt_91012, SWS_EthSwt_91013 –> SRS_ETH_00126
  SWS_EthSwt_91014, SWS_EthSwt_91015, SWS_EthSwt_91016, SWS_EthSwt_91018, SWS_EthSwt_91019, SWS_EthSwt_91021,

SWS_EthSwt_91022, SWS_EthSwt_91022, SWS_EthSwt_91023, SWS_EthSwt_91024, SWS_EthSwt_91025, SWS_EthSwt_91026, SWS_EthSwt_91027 –> SRS_Eth_00123
SWS_EthSwt_00117, SWS_EthSwt_00203 –> SRS_ETH_00119, SRS_ETH_00087
SWS_EthSwt_00193 –> SRS_ETH_00122, SRS_ETH_00087
SWS_EthSwt_00114 –> SRS_BSW_00433
SWS_EthSwt_00098 –> SRS_Eth_00122, SRS_ETH_00118, SRS_ETH_00119, SRS_ETH_00120, SRS_ETH_00087, SRS_ETH_00125, SRS_BSW_00375
–Last change on issue 76248 comment 13–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.275 Specification Item SWS_EthSwt_91021

Trace References:

SRS_Eth_00123

Content:

Service name:	EthSwt_GetPortMirrorStateEthSwt_GetPortMirrorState	
Syntax:	Std_ReturnType EthSwt_GetPortMirrorState(uint8 SwitchIdx, uint8 PortIdx, EthSwt_PortMirrorStateType* PortMirrorStatePtr)	
Service ID[hex]:	0x38	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	SwitchIdxEthSwt_GetPortMirrorState.SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	PortIdxEthSwt_GetPortMirrorState.PortIdx	Index of the port at the addressed switch
Parameters (inout):	None	
Parameters (out):	PortMirrorStatePtrEthSwt_GetPortMirrorState.PortMirrorStatePtr	Pointer to the memory where the port state shall be stored. 0x00 == port mirroring disabled, 0x01 == port mirroring enabled
Return value:	Std_ReturnType	E_OK: the port mirroring state for the indexed Ethernet switch port returned successfully. E_NOT_OK: the port mirror configuration for the indexed Ethernet switch returned not successfully. (i.e. indexed ethernet switch port is not available)

Description:	Obtain the current status of the port mirroring for the indexed Ethernet switch port

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76248: [EthSwt]Remove references to non existing requirements (SRS, SWS)

  Problem description:

  Some review findings brought up references to non existing requirements. Please check and remove the references and if needed reformulate the requirements.

  Agreed solution:

  SWS_EthSwt_00123 –> SRS_BSW_00406
  SWS_EthSwt_00165 -> SRS_BSW_00395
  SWS_EthSwt_00227 -> SRS_ETH_00087
  SWS_EthSwt_91017, SWS_EthSwt_91020, –> SRS_ETH_00123
  SWS_EthSwt_00006 –> SRS_BSW_00101
  SWS_EthSwt_00010 left blank as SRS doe not yet provide a requirement here but maybe with RFC # 75170 AUTOSAR will introduce an SRS-item after all.
  SWS_EthSwt_00376, SWS_EthSwt_00374, SWS_EthSwt_00375 –> SRS-Missing! same as for SWS_EThSwt_00010
  SWS_EthSwt_00018, SRS_EthSwt_00025 –> SRS_ETH_00118
  SWS_EthSwt_00031 –> SRS_ETH_00087
  SWS_EthSwt_00037 –> SRS_ETH_00119
  SWS_EthSwt_00044, SWS_EthSwt_00051 –> SRS_ETH_00118
  SWS_EthSwt_00060 –> SRS_ETH_00087
  SWS_EthSwt_00111 –> erase SRS_ETH_00086
  SWS_EthSwt00079 –>SRS_ETH_00119
  SWS_EthSwt_00206, SWS_EthSwt_00211, SWS_EthSwt_00216, SWS_EthSwt_00221 –>SRS_Eth_00120
  SWS_EthSwt_00172 –>SRS_ETH_00121, SRS_ETH_00114
  SWS_EthSwt_00086, SWS_EthSwt_00091, SWS_EthSwt_00182, SWS_EthSwt_00125 –> SRS_ETH_00087, SRS_ETH_00122
  SWS_EthSwt_00187 –> SRS_ETH_00087
  SWS_EthSwt_00058 –> SRS_BSW_00171
  SWS_EthSwt_91012, SWS_EthSwt_91013 –> SRS_ETH_00126
  SWS_EthSwt_91014, SWS_EthSwt_91015, SWS_EthSwt_91016, SWS_EthSwt_91018, SWS_EthSwt_91019, SWS_EthSwt_91021, SWS_EthSwt_91022, SWS_EthSwt_91022, SWS_EthSwt_91023, SWS_EthSwt_91024, SWS_EthSwt_91025, SWS_EthSwt_91026, SWS_EthSwt_91027 –> SRS_Eth_00123
  SWS_EthSwt_00117, SWS_EthSwt_00203 –> SRS_ETH_00119,

SRS_ETH_00087
SWS_EthSwt_00193 –> SRS_ETH_00122, SRS_ETH_00087
SWS_EthSwt_00114 –> SRS_BSW_00433
SWS_EthSwt_00098 –> SRS_Eth_00122, SRS_ETH_00118, SRS_ETH_00119, SRS_ETH_00120, SRS_ETH_00087, SRS_ETH_00125, SRS_BSW_00375
–Last change on issue 76248 comment 13–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.276 Specification Item SWS_EthSwt_91022

Trace References:

SRS_Eth_00123

Content:

Service name:	EthSwt_SetPortTestModeMirrorStateEthSwt_SetPortTestMode MirrorState	
Syntax:	Std_ReturnType EthSwt_SetPortTestModeMirrorState(uint8 SwitchIdx, uint8 PortIdx, EthTrcv_PhyTestModeType Mode const EthSwt_PortMirrorStateType* PortMirror StatePtr)	
Service ID[hex]:	0x3a 0x39	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	SwitchIdxEthSwt_SetPortTestModeMirrorState.SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	PortIdxEthSwt_SetPortTestModeMirrorState.PortIdx	Index of the port at the addressed switch
	ModePortMirrorStatePtrEthSwt_SetPortTestMode.Mode MirrorState.PortMirrorStatePtr	Test mode to be activated Pointer to the memory where the requested port state is stored. 0x00 == port mirroring disabled, 0x01 == port mirroring enabled
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: the port test mode mirroring state for the indexed Ethernet switch port was set successfully. E_NOT_OK: the port test mode mirror configuration for the indexed Ethernet switch was not set successfully. (i.e. indexed Ethernet switch port is not available)

Description:	Activates a given test mode of Set the current state of port mirroring for the indexed Ethernet switch port . The test mode shall be forwarded to the EthTrcv (EthTrcv_SetPhyTestMode) that is referenced by the EthSwtPort. If no tranceiver is referenced the return value shall be E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76248: [EthSwt]Remove references to non existing requirements (SRS, SWS)

  Problem description:

  Some review findings brought up references to non existing requirements. Please check and remove the references and if needed reformulate the requirements.

  Agreed solution:

  SWS_EthSwt_00123 –> SRS_BSW_00406
  SWS_EthSwt_00165 -> SRS_BSW_00395
  SWS_EthSwt_00227 -> SRS_ETH_00087
  SWS_EthSwt_91017, SWS_EthSwt_91020, –> SRS_ETH_00123
  SWS_EthSwt_00006 –> SRS_BSW_00101
  SWS_EthSwt_00010 left blank as SRS doe not yet provide a requirement here but maybe with RFC # 75170 AUTOSAR will introduce an SRS-item after all.
  SWS_EthSwt_00376, SWS_EthSwt_00374, SWS_EthSwt_00375 –> SRS-Missing! same as for SWS_EThSwt_00010
  SWS_EthSwt_00018, SRS_EthSwt_00025 –> SRS_ETH_00118
  SWS_EthSwt_00031 –> SRS_ETH_00087
  SWS_EthSwt_00037 –> SRS_ETH_00119
  SWS_EthSwt_00044, SWS_EthSwt_00051 –> SRS_ETH_00118
  SWS_EthSwt_00060 –> SRS_ETH_00087
  SWS_EthSwt_00111 –> erase SRS_ETH_00086
  SWS_EthSwt00079 –>SRS_ETH_00119
  SWS_EthSwt_00206, SWS_EthSwt_00211, SWS_EthSwt_00216, SWS_EthSwt_00221 –>SRS_Eth_00120
  SWS_EthSwt_00172 –>SRS_ETH_00121, SRS_ETH_00114
  SWS_EthSwt_00086, SWS_EthSwt_00091, SWS_EthSwt_00182, SWS_EthSwt_00125 –> SRS_ETH_00087, SRS_ETH_00122
  SWS_EthSwt_00187 –> SRS_ETH_00087
  SWS_EthSwt_00058 –> SRS_BSW_00171
  SWS_EthSwt_91012, SWS_EthSwt_91013 –> SRS_ETH_00126
  SWS_EthSwt_91014, SWS_EthSwt_91015, SWS_EthSwt_91016, SWS_EthSwt_91018, SWS_EthSwt_91019, SWS_EthSwt_91021, SWS_EthSwt_91022, SWS_EthSwt_91022, SWS_EthSwt_91023, SWS_EthSwt_91024, SWS_EthSwt_91025, SWS_EthSwt_91026,

SWS_EthSwt_91027 –> SRS_Eth_00123
SWS_EthSwt_00117, SWS_EthSwt_00203 –> SRS_ETH_00119, SRS_ETH_00087
SWS_EthSwt_00193 –> SRS_ETH_00122, SRS_ETH_00087
SWS_EthSwt_00114 –> SRS_BSW_00433
SWS_EthSwt_00098 –> SRS_Eth_00122, SRS_ETH_00118, SRS_ETH_00119, SRS_ETH_00120, SRS_ETH_00087, SRS_ETH_00125, SRS_BSW_00375
–Last change on issue 76248 comment 13–

BW-C-Level:

Application	Specification	Bus
1	1	1

- RfC #76890: [EthSwt] SWS_EthSwt_91022 requirement ID is duplicated

  Problem description:

  In the chapters 8.3.45 EthSwt_SetPortMirrorState and 8.3.46 EthSwt_SetPortTestMode SWS_EthSwt_91022 requirement ID is reused.

  A different requirement ID should be used in chapter 8.3.46 EthSwt_SetPortTestMode.

  Agreed solution:

  - give new ReqID to EthSwt_SetPortTestMode
  - correct the ID format of requirement SWS_EthSwt_00380 (from "[SWS_EthSwt_00380]]" to "[SWS_EthSwt_00380]")
  –Last change on issue 76890 comment 6–

  BW-C-Level:

Application	Specification	Bus
1	1	1

- RfC #77628: [EthSwt] Behaviour of certain APIs for ports without EthSwtPortTrcvRef (part I)

  Problem description:

  The following APIs have partly a description regarding the handling of ports without EthSwtPortTrcvRef:
  EthSwt_GetPortSignalQuality
  EthSwt_GetPortIdentifier
  EthSwt_SetPortTestMode
  EthTrcv_SetPhyTestMode
  EthSwt_SetPortTxMode

EthSwt_GetPortCableDiagnosticsResult

The description should be adjusted and harmonized with related requirements.
–Last change on issue 77628 comment 2–

Agreed solution:

=== EthSwt ===
~[SWS_EthSwt_91014]EthSwt_GetPortSignalQuality
Description:
The function retrieves the signal quality of the link of the indexed Ethernet switch port.
~[SWS_EthSwt_00293]
The function EthSwt_GetPortSignalQuality() shall obtain the signal quality by calling the function EthTrcv_GetPhySignalQuality() of the referenced Ethernet Transceiver Driver. If the current signal quality is not available, the signal quality shall be set to 0xFF.
-[SWS_EthSwt_00298]as # 77349 introduces a general req

~[SWS_EthSwt_91015]EthSwt_GetPortIdentifier
Return value
E_NOT_OK: organizationally unique identifier of the Ethernet transceiver could not be obtained (i.e. OUI is not available)
Description:
This function retrieves the OUI (24 bit) of the indexed Ethernet switch port.

~[SWS_EthSwt_00299]
The function EthSwt_GetPortIdentifier() shall return the value of the organizationally unique identifier (OUI 24 bit) of the indexed Ethernet switch port that is connected to the indexed Ethernet switch. It shall set the 8 most significant bits of the OUI to 0xFFxxxxxx. If the Ethernet switch port references an Ethernet transceiver, the function shall obtain the OUI by calling the function EthTrcv_GetPhyIdentifier() and set the 8 most significant bits of the OUI to 0x00xxxxxx.

SWS_EthSwt_xxxxx] If neither the Ethernet switch port nor the Ethernet Transceiver Driver can provide an OUI the function EthSwt_GetPortIdentifier() shall return E_NOT_OK.

-[SWS_EthSwt_00304]as # 77349 introduces a general req

~[SWS_EthSwt_91029] EthSwt_SetPortTestMode
Description:

Activates a given test mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00328]
The function EthSwt_SetPortTestMode shall forward the call with the given test mode by calling the function EthTrcv_SetPhyTestMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00333]as # 77349 introduces a general req

~[SWS_EthSwt_91023] EthSwt_SetPortLoopbackMode
Description:
Activates a given test loop-back mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00334]
The function EthSwt_SetPortLoopbackMode() shall forward the call with the given loop-back mode by calling the function EthTrcv_SetPhyLoopbackMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00339] as # 77349 introduces a general req

~[SWS_EthSwt_91024] EthSwt_SetPortTxMode
Description:
Activates a given transmission mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00340]
The function EthSwt_SetPortTxMode() shall forward the call with the given transmission mode by calling the function EthTrcv_SetPhyTxMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00345] as # 77349 introduces a general req

~[SWS_EthSwt_91025]EthSwt_GetPortCableDiagnosticsResult
Description:
Retrieves the cable diagnostics result of the indexed Ethernet switch port respectively the referenced Ethernet Transceiver Driver.

~[SWS_EthSwt_00346]
The function EthSwt_GetPortCableDiagnosticsResult() shall obtain the cable diagnostics result by calling the function EthTrcv_GetCableDiagnosticsResult() of the referenced Ethernet Transceiver Driver. If no Ethernet transceiver is referenced by the Ethernet switch port and development error detection is not enabled, the cable diagnostic result shall be set to ETHTRCV_CABLEDIAG_OK.
-[SWS_EthSwt_00351] as # 77349 introduces a general req

=== EthTrcv ===
add certain parameter to enable/disable API functions:
+ SWS item ECUC_EthTrcv_xxxx1

Name EthTrcvGetPhySignalQualityApi
Description Enables / Disables EthTrcv_GetPhySignalQuality API

+ SWS item ECUC_EthTrcv_xxxx2
Name EthTrcvGetPhyIdentifierApi
Description Enables / Disables EthTrcv_GetPhyIdentifier API

+ SWS item ECUC_EthTrcv_xxxx3
Name EthTrcvSetPhyTestModeApi
Description Enables / Disables EthTrcv_SetPhyTestMode API

+ SWS item ECUC_EthTrcv_xxxx4
Name EthTrcvSetPhyTxModeApi
Description Enables / Disables EthTrcv_SetPhyTxMode API

+ SWS item ECUC_EthTrcv_xxxx5
Name EthTrcvGetCableDiagnosticsResultApi
Description Enables / Disables EthTrcv_GetCableDiagnosticsResult API

add the following specification to the configuration parameter above:
Multiplicity 1
Type EcucBooleanParamDef
Default value –
Post-Build Variant Value false
Value Configuration
Class Pre-compile time X
All Variants Link time –
Post-build time –
Scope / Dependency scope: local
–Last change on issue 77628 comment 31–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.277 Specification Item SWS_EthSwt_91023

Trace References:

SRS_Eth_00123

Content:

Service name:	EthSwt_SetPortLoopbackModeEthSwt_SetPortLoopbackMode	
Syntax:	Std_ReturnType EthSwt_SetPortLoopbackMode(uint8 SwitchIdx, uint8 PortIdx, EthTrcv_PhyLoopbackModeType Mode)	
Service ID[hex]:	0x3b	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	SwitchIdxEthSwt_SetPortLoopback Mode.SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	PortIdxEthSwt_SetPortLoopback Mode.PortIdx	Index of the port at the addressed switch
	ModeEthSwt_SetPortLoopback Mode.Mode	Loop-back mode to be activated
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: the port mirroring loop-back back mode for the indexed Ethernet switch port was activated successfully. E_NOT_OK: the port mirroring loop-back back mode for the indexed Ethernet switch port was not activated successfully. (i.e. indexed Ethernet switch port is not available)
Description:	Activates a given test loop-back mode of the indexed Ethernet switch port. The loop-back mode shall be forwarded to the EthTrcv (EthTrcv_SetPhyLoopbackMode) that is referenced by the EthSwtPort. If no transceiver is referenced the return value shall be E_NOT_OK.	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76248: [EthSwt]Remove references to non existing requirements (SRS, SWS)

  Problem description:

  Some review findings brought up references to non existing requirements. Please check and remove the references and if needed reformulate the requirements.

  Agreed solution:

  SWS_EthSwt_00123 –> SRS_BSW_00406
  SWS_EthSwt_00165 -> SRS_BSW_00395
  SWS_EthSwt_00227 -> SRS_ETH_00087
  SWS_EthSwt_91017, SWS_EthSwt_91020, –> SRS_ETH_00123
  SWS_EthSwt_00006 –> SRS_BSW_00101
  SWS_EthSwt_00010 left blank as SRS doe not yet provide a requirement here but maybe with RFC # 75170 AUTOSAR will introduce an SRS-item after all.
  SWS_EthSwt_00376, SWS_EthSwt_00374, SWS_EthSwt_00375 –> SRS-Missing!

same as for SWS_EThSwt_00010
SWS_EthSwt_00018, SRS_EthSwt_00025 –> SRS_ETH_00118
SWS_EthSwt_00031 –> SRS_ETH_00087
SWS_EthSwt_00037 –> SRS_ETH_00119
SWS_EthSwt_00044, SWS_EthSwt_00051 –> SRS_ETH_00118
SWS_EthSwt_00060 –> SRS_ETH_00087
SWS_EthSwt_00111 –> erase SRS_ETH_00086
SWS_EthSwt00079 –>SRS_ETH_00119
SWS_EthSwt_00206, SWS_EthSwt_00211, SWS_EthSwt_00216, SWS_EthSwt_00221 –>SRS_Eth_00120
SWS_EthSwt_00172 –>SRS_ETH_00121, SRS_ETH_00114
SWS_EthSwt_00086, SWS_EthSwt_00091, SWS_EthSwt_00182, SWS_EthSwt_00125 –> SRS_ETH_00087, SRS_ETH_00122
SWS_EthSwt_00187 –> SRS_ETH_00087
SWS_EthSwt_00058 –> SRS_BSW_00171
SWS_EthSwt_91012, SWS_EthSwt_91013 –> SRS_ETH_00126
SWS_EthSwt_91014, SWS_EthSwt_91015, SWS_EthSwt_91016, SWS_EthSwt_91018, SWS_EthSwt_91019, SWS_EthSwt_91021, SWS_EthSwt_91022, SWS_EthSwt_91022, SWS_EthSwt_91023, SWS_EthSwt_91024, SWS_EthSwt_91025, SWS_EthSwt_91026, SWS_EthSwt_91027 –> SRS_Eth_00123
SWS_EthSwt_00117, SWS_EthSwt_00203 –> SRS_ETH_00119, SRS_ETH_00087
SWS_EthSwt_00193 –> SRS_ETH_00122, SRS_ETH_00087
SWS_EthSwt_00114 –> SRS_BSW_00433
SWS_EthSwt_00098 –> SRS_Eth_00122, SRS_ETH_00118, SRS_ETH_00119, SRS_ETH_00120, SRS_ETH_00087, SRS_ETH_00125, SRS_BSW_00375
–Last change on issue 76248 comment 13–

BW-C-Level:

Application	Specification	Bus
1	1	1

- RfC #77628: [EthSwt] Behaviour of certain APIs for ports without EthSwtPortTrcvRef (part I)

  Problem description:

  The following APIs have partly a description regarding the handling of ports without EthSwtPortTrcvRef:
  EthSwt_GetPortSignalQuality
  EthSwt_GetPortIdentifier
  EthSwt_SetPortTestMode
  EthTrcv_SetPhyTestMode

EthSwt_SetPortTxMode
EthSwt_GetPortCableDiagnosticsResult

The description should be adjusted and harmonized with related requirements.
–Last change on issue 77628 comment 2–

Agreed solution:

=== EthSwt ===
~[SWS_EthSwt_91014]EthSwt_GetPortSignalQuality
Description:
The function retrieves the signal quality of the link of the indexed Ethernet switch port.
~[SWS_EthSwt_00293]
The function EthSwt_GetPortSignalQuality() shall obtain the signal quality by calling the function EthTrcv_GetPhySignalQuality() of the referenced Ethernet Transceiver Driver. If the current signal quality is not available, the signal quality shall be set to 0xFF.
-[SWS_EthSwt_00298]as # 77349 introduces a general req

~[SWS_EthSwt_91015]EthSwt_GetPortIdentifier
Return value
E_NOT_OK: organizationally unique identifier of the Ethernet transceiver could not be obtained (i.e. OUI is not available)
Description:
This function retrieves the OUI (24 bit) of the indexed Ethernet switch port.

~[SWS_EthSwt_00299]
The function EthSwt_GetPortIdentifier() shall return the value of the organizationally unique identifier (OUI 24 bit) of the indexed Ethernet switch port that is connected to the indexed Ethernet switch. It shall set the 8 most significant bits of the OUI to 0xFFxxxxxx. If the Ethernet switch port references an Ethernet transceiver, the function shall obtain the OUI by calling the function EthTrcv_GetPhyIdentifier() and set the 8 most significant bits of the OUI to 0x00xxxxxx.

SWS_EthSwt_xxxxx] If neither the Ethernet switch port nor the Ethernet Transceiver Driver can provide an OUI the function EthSwt_GetPortIdentifier() shall return E_NOT_OK.

-[SWS_EthSwt_00304]as # 77349 introduces a general req

~[SWS_EthSwt_91029] EthSwt_SetPortTestMode

Description:
Activates a given test mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00328]
The function EthSwt_SetPortTestMode shall forward the call with the given test mode by calling the function EthTrcv_SetPhyTestMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00333]as # 77349 introduces a general req

~[SWS_EthSwt_91023] EthSwt_SetPortLoopbackMode
Description:
Activates a given test loop-back mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00334]
The function EthSwt_SetPortLoopbackMode() shall forward the call with the given loop-back mode by calling the function EthTrcv_SetPhyLoopbackMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00339] as # 77349 introduces a general req

~[SWS_EthSwt_91024] EthSwt_SetPortTxMode
Description:
Activates a given transmission mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00340]
The function EthSwt_SetPortTxMode() shall forward the call with the given transmission mode by calling the function EthTrcv_SetPhyTxMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00345] as # 77349 introduces a general req

~[SWS_EthSwt_91025]EthSwt_GetPortCableDiagnosticsResult
Description:
Retrieves the cable diagnostics result of the indexed Ethernet switch port respectively the referenced Ethernet Transceiver Driver.

~[SWS_EthSwt_00346]
The function EthSwt_GetPortCableDiagnosticsResult() shall obtain the cable diagnostics result by calling the function EthTrcv_GetCableDiagnosticsResult() of the referenced Ethernet Transceiver Driver. If no Ethernet transceiver is referenced by the Ethernet switch port and development error detection is not enabled, the cable diagnostic result shall be set to ETHTRCV_CABLEDIAG_OK.
-[SWS_EthSwt_00351] as # 77349 introduces a general req

=== EthTrcv ===
add certain parameter to enable/disable API functions:

+ SWS item ECUC_EthTrcv_xxxx1
Name EthTrcvGetPhySignalQualityApi
Description Enables / Disables EthTrcv_GetPhySignalQuality API

+ SWS item ECUC_EthTrcv_xxxx2
Name EthTrcvGetPhyIdentifierApi
Description Enables / Disables EthTrcv_GetPhyIdentifier API

+ SWS item ECUC_EthTrcv_xxxx3
Name EthTrcvSetPhyTestModeApi
Description Enables / Disables EthTrcv_SetPhyTestMode API

+ SWS item ECUC_EthTrcv_xxxx4
Name EthTrcvSetPhyTxModeApi
Description Enables / Disables EthTrcv_SetPhyTxMode API

+ SWS item ECUC_EthTrcv_xxxx5
Name EthTrcvGetCableDiagnosticsResultApi
Description Enables / Disables EthTrcv_GetCableDiagnosticsResult API

add the following specification to the configuration parameter above:
Multiplicity 1
Type EcucBooleanParamDef
Default value –
Post-Build Variant Value false
Value Configuration
Class Pre-compile time X
All Variants Link time –
Post-build time –
Scope / Dependency scope: local
–Last change on issue 77628 comment 31–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.278 Specification Item SWS_EthSwt_91024

Trace References:

SRS_Eth_00123

Content:

Service name:	EthSwt_SetPortTxModeEthSwt_SetPortTxMode	
Syntax:	Std_ReturnType EthSwt_SetPortTxMode(uint8 SwitchIdx, uint8 PortIdx, EthTrcv_PhyTxModeType Mode)	
Service ID[hex]:	0x3c	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	SwitchIdxEthSwt_SetPortTx Mode.SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	PortIdxEthSwt_SetPortTxMode.PortIdx	Index of the port at the addressed switch
	ModeEthSwt_SetPortTxMode.Mode	Transmission mode to be activated
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: the port Tx mode for the indexed Ethernet switch port was activated successfully. E_NOT_OK: the port Tx mode for the indexed Ethernet switch port was not activated successfully. (i.e. indexed Ethernet switch port is not available)
Description:	Activates a given transmission mode of the indexed Ethernet switch port. The transmission mode shall be forwarded to the EthTrcv (EthTrcv_SetPhyTxMode) that is referenced by the EthSwtPort. If no transceiver is referenced the return value shall be E_NOT_OK.	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76248: [EthSwt]Remove references to non existing requirements (SRS, SWS)

  Problem description:

  Some review findings brought up references to non existing requirements. Please check and remove the references and if needed reformulate the requirements.

  Agreed solution:

  SWS_EthSwt_00123 –> SRS_BSW_00406
  SWS_EthSwt_00165 -> SRS_BSW_00395
  SWS_EthSwt_00227 -> SRS_ETH_00087
  SWS_EthSwt_91017, SWS_EthSwt_91020, –> SRS_ETH_00123
  SWS_EthSwt_00006 –> SRS_BSW_00101
  SWS_EthSwt_00010 left blank as SRS doe not yet provide a requirement here but maybe with RFC # 75170 AUTOSAR will introduce an SRS-item after all.
  SWS_EthSwt_00376, SWS_EthSwt_00374, SWS_EthSwt_00375 –> SRS-Missing! same as for SWS_EThSwt_00010

SWS_EthSwt_00018, SRS_EthSwt_00025 –> SRS_ETH_00118
SWS_EthSwt_00031 –> SRS_ETH_00087
SWS_EthSwt_00037 –> SRS_ETH_00119
SWS_EthSwt_00044, SWS_EthSwt_00051 –> SRS_ETH_00118
SWS_EthSwt_00060 –> SRS_ETH_00087
SWS_EthSwt_00111 –> erase SRS_ETH_00086
SWS_EthSwt00079 –>SRS_ETH_00119
SWS_EthSwt_00206, SWS_EthSwt_00211, SWS_EthSwt_00216, SWS_EthSwt_00221 –>SRS_Eth_00120
SWS_EthSwt_00172 –>SRS_ETH_00121, SRS_ETH_00114
SWS_EthSwt_00086, SWS_EthSwt_00091, SWS_EthSwt_00182, SWS_EthSwt_00125 –> SRS_ETH_00087, SRS_ETH_00122
SWS_EthSwt_00187 –> SRS_ETH_00087
SWS_EthSwt_00058 –> SRS_BSW_00171
SWS_EthSwt_91012, SWS_EthSwt_91013 –> SRS_ETH_00126
SWS_EthSwt_91014, SWS_EthSwt_91015, SWS_EthSwt_91016, SWS_EthSwt_91018, SWS_EthSwt_91019, SWS_EthSwt_91021, SWS_EthSwt_91022, SWS_EthSwt_91022, SWS_EthSwt_91023, SWS_EthSwt_91024, SWS_EthSwt_91025, SWS_EthSwt_91026, SWS_EthSwt_91027 –> SRS_Eth_00123
SWS_EthSwt_00117, SWS_EthSwt_00203 –> SRS_ETH_00119, SRS_ETH_00087
SWS_EthSwt_00193 –> SRS_ETH_00122, SRS_ETH_00087
SWS_EthSwt_00114 –> SRS_BSW_00433
SWS_EthSwt_00098 –> SRS_Eth_00122, SRS_ETH_00118, SRS_ETH_00119, SRS_ETH_00120, SRS_ETH_00087, SRS_ETH_00125, SRS_BSW_00375
–Last change on issue 76248 comment 13–

BW-C-Level:

Application	Specification	Bus
1	1	1

- RfC #77628: [EthSwt] Behaviour of certain APIs for ports without EthSwtPortTrcvRef (part I)

  Problem description:

  The following APIs have partly a description regarding the handling of ports without EthSwtPortTrcvRef:
  EthSwt_GetPortSignalQuality
  EthSwt_GetPortIdentifier
  EthSwt_SetPortTestMode
  EthTrcv_SetPhyTestMode
  EthSwt_SetPortTxMode

EthSwt_GetPortCableDiagnosticsResult

The description should be adjusted and harmonized with related requirements.
–Last change on issue 77628 comment 2–

Agreed solution:

=== EthSwt ===
~[SWS_EthSwt_91014]EthSwt_GetPortSignalQuality
Description:
The function retrieves the signal quality of the link of the indexed Ethernet switch port.
~[SWS_EthSwt_00293]
The function EthSwt_GetPortSignalQuality() shall obtain the signal quality by calling the function EthTrcv_GetPhySignalQuality() of the referenced Ethernet Transceiver Driver. If the current signal quality is not available, the signal quality shall be set to 0xFF.
-[SWS_EthSwt_00298]as # 77349 introduces a general req

~[SWS_EthSwt_91015]EthSwt_GetPortIdentifier
Return value
E_NOT_OK: organizationally unique identifier of the Ethernet transceiver could not be obtained (i.e. OUI is not available)
Description:
This function retrieves the OUI (24 bit) of the indexed Ethernet switch port.

~[SWS_EthSwt_00299]
The function EthSwt_GetPortIdentifier() shall return the value of the organizationally unique identifier (OUI 24 bit) of the indexed Ethernet switch port that is connected to the indexed Ethernet switch. It shall set the 8 most significant bits of the OUI to 0xFFxxxxxx. If the Ethernet switch port references an Ethernet transceiver, the function shall obtain the OUI by calling the function EthTrcv_GetPhyIdentifier() and set the 8 most significant bits of the OUI to 0x00xxxxxx.

SWS_EthSwt_xxxxx] If neither the Ethernet switch port nor the Ethernet Transceiver Driver can provide an OUI the function EthSwt_GetPortIdentifier() shall return E_NOT_OK.

-[SWS_EthSwt_00304]as # 77349 introduces a general req

~[SWS_EthSwt_91029] EthSwt_SetPortTestMode
Description:

Activates a given test mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00328]
The function EthSwt_SetPortTestMode shall forward the call with the given test mode by calling the function EthTrcv_SetPhyTestMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00333]as # 77349 introduces a general req

~[SWS_EthSwt_91023] EthSwt_SetPortLoopbackMode
Description:
Activates a given test loop-back mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00334]
The function EthSwt_SetPortLoopbackMode() shall forward the call with the given loop-back mode by calling the function EthTrcv_SetPhyLoopbackMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00339] as # 77349 introduces a general req

~[SWS_EthSwt_91024] EthSwt_SetPortTxMode
Description:
Activates a given transmission mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00340]
The function EthSwt_SetPortTxMode() shall forward the call with the given transmission mode by calling the function EthTrcv_SetPhyTxMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00345] as # 77349 introduces a general req

~[SWS_EthSwt_91025]EthSwt_GetPortCableDiagnosticsResult
Description:
Retrieves the cable diagnostics result of the indexed Ethernet switch port respectively the referenced Ethernet Transceiver Driver.

~[SWS_EthSwt_00346]
The function EthSwt_GetPortCableDiagnosticsResult() shall obtain the cable diagnostics result by calling the function EthTrcv_GetCableDiagnosticsResult() of the referenced Ethernet Transceiver Driver. If no Ethernet transceiver is referenced by the Ethernet switch port and development error detection is not enabled, the cable diagnostic result shall be set to ETHTRCV_CABLEDIAG_OK.
-[SWS_EthSwt_00351] as # 77349 introduces a general req

=== EthTrcv ===
add certain parameter to enable/disable API functions:
+ SWS item ECUC_EthTrcv_xxxx1

Name EthTrcvGetPhySignalQualityApi
Description Enables / Disables EthTrcv_GetPhySignalQuality API

+ SWS item ECUC_EthTrcv_xxxx2
Name EthTrcvGetPhyIdentifierApi
Description Enables / Disables EthTrcv_GetPhyIdentifier API

+ SWS item ECUC_EthTrcv_xxxx3
Name EthTrcvSetPhyTestModeApi
Description Enables / Disables EthTrcv_SetPhyTestMode API

+ SWS item ECUC_EthTrcv_xxxx4
Name EthTrcvSetPhyTxModeApi
Description Enables / Disables EthTrcv_SetPhyTxMode API

+ SWS item ECUC_EthTrcv_xxxx5
Name EthTrcvGetCableDiagnosticsResultApi
Description Enables / Disables EthTrcv_GetCableDiagnosticsResult API

add the following specification to the configuration parameter above:
Multiplicity 1
Type EcucBooleanParamDef
Default value –
Post-Build Variant Value false
Value Configuration
Class Pre-compile time X
All Variants Link time –
Post-build time –
Scope / Dependency scope: local
–Last change on issue 77628 comment 31–

BW-C-Level:

Application	**Specification**	**Bus**
1	4	1

1.279 Specification Item SWS_EthSwt_91025

Trace References:

SRS_Eth_00123

Content:

Service name:	EthSwt_GetPortCableDiagnosticsResultEthSwt_GetPortCableDiagnosticsResult	
Syntax:	Std_ReturnType EthSwt_GetPortCableDiagnosticsResult(uint8 SwitchIdx, uint8 PortIdx, EthTrcv_CableDiagResultType* ResultPtr)	
Service ID[hex]:	0x3f	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	SwitchIdxEthSwt_GetPortCable DiagnosticsResult.SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	PortIdxEthSwt_GetPortCable DiagnosticsResult.PortIdx	Index of the port at the addressed switch
Parameters (inout):	None	
Parameters (out):	ResultPtrEthSwt_GetPortCable DiagnosticsResult.ResultPtr	Pointer to the location where the cable diagnostics result shall be stored
Return value:	Std_ReturnType	E_OK:the port cable diagnostic result for the indexed Ethernet switch port was obtained successfully. E_NOT_OK: the port cable diagnostic result for the indexed Ethernet switch port was not obtained successfully. (i.e. indexed Ethernet switch port is not available)
Description:	Retrieves the cable diagnostics result of the indexed Ethernet switch port . The transmission mode shall be forwarded to the EthTrcv (EthTrcv_GetCableDiagnostics Result) that is referenced by the EthSwtPort. If no Ethernet transceiver is referenced by the Ethernet switch port the cable diagnostic shall be set to ETHTRCV_CABLEDIAG_OKrespectively the referenced Ethernet Transceiver Driver.	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76248: [EthSwt]Remove references to non existing requirements (SRS, SWS)

  Problem description:

  Some review findings brought up references to non existing requirements. Please check and remove the references and if needed reformulate the requirements.

  Agreed solution:

  SWS_EthSwt_00123 –> SRS_BSW_00406
  SWS_EthSwt_00165 -> SRS_BSW_00395
  SWS_EthSwt_00227 -> SRS_ETH_00087
  SWS_EthSwt_91017, SWS_EthSwt_91020, –> SRS_ETH_00123
  SWS_EthSwt_00006 –> SRS_BSW_00101
  SWS_EthSwt_00010 left blank as SRS doe not yet provide a requirement here but maybe with RFC # 75170 AUTOSAR will introduce an SRS-item after all.
  SWS_EthSwt_00376, SWS_EthSwt_00374, SWS_EthSwt_00375 –> SRS-Missing! same as for SWS_EThSwt_00010

SWS_EthSwt_00018, SRS_EthSwt_00025 –> SRS_ETH_00118
SWS_EthSwt_00031 –> SRS_ETH_00087
SWS_EthSwt_00037 –> SRS_ETH_00119
SWS_EthSwt_00044, SWS_EthSwt_00051 –> SRS_ETH_00118
SWS_EthSwt_00060 –> SRS_ETH_00087
SWS_EthSwt_00111 –> erase SRS_ETH_00086
SWS_EthSwt00079 –>SRS_ETH_00119
SWS_EthSwt_00206, SWS_EthSwt_00211, SWS_EthSwt_00216, SWS_EthSwt_00221 –>SRS_Eth_00120
SWS_EthSwt_00172 –>SRS_ETH_00121, SRS_ETH_00114
SWS_EthSwt_00086, SWS_EthSwt_00091, SWS_EthSwt_00182, SWS_EthSwt_00125 –> SRS_ETH_00087, SRS_ETH_00122
SWS_EthSwt_00187 –> SRS_ETH_00087
SWS_EthSwt_00058 –> SRS_BSW_00171
SWS_EthSwt_91012, SWS_EthSwt_91013 –> SRS_ETH_00126
SWS_EthSwt_91014, SWS_EthSwt_91015, SWS_EthSwt_91016, SWS_EthSwt_91018, SWS_EthSwt_91019, SWS_EthSwt_91021, SWS_EthSwt_91022, SWS_EthSwt_91022, SWS_EthSwt_91023, SWS_EthSwt_91024, SWS_EthSwt_91025, SWS_EthSwt_91026, SWS_EthSwt_91027 –> SRS_Eth_00123
SWS_EthSwt_00117, SWS_EthSwt_00203 –> SRS_ETH_00119, SRS_ETH_00087
SWS_EthSwt_00193 –> SRS_ETH_00122, SRS_ETH_00087
SWS_EthSwt_00114 –> SRS_BSW_00433
SWS_EthSwt_00098 –> SRS_Eth_00122, SRS_ETH_00118, SRS_ETH_00119, SRS_ETH_00120, SRS_ETH_00087, SRS_ETH_00125, SRS_BSW_00375
–Last change on issue 76248 comment 13–

BW-C-Level:

Application	Specification	Bus
1	1	1

- RfC #77628: [EthSwt] Behaviour of certain APIs for ports without EthSwtPortTrcvRef (part I)

  Problem description:

  The following APIs have partly a description regarding the handling of ports without EthSwtPortTrcvRef:
  EthSwt_GetPortSignalQuality
  EthSwt_GetPortIdentifier
  EthSwt_SetPortTestMode
  EthTrcv_SetPhyTestMode
  EthSwt_SetPortTxMode

EthSwt_GetPortCableDiagnosticsResult

The description should be adjusted and harmonized with related requirements.
–Last change on issue 77628 comment 2–

Agreed solution:

=== EthSwt ===
~[SWS_EthSwt_91014]EthSwt_GetPortSignalQuality
Description:
The function retrieves the signal quality of the link of the indexed Ethernet switch port.
~[SWS_EthSwt_00293]
The function EthSwt_GetPortSignalQuality() shall obtain the signal quality by calling the function EthTrcv_GetPhySignalQuality() of the referenced Ethernet Transceiver Driver. If the current signal quality is not available, the signal quality shall be set to 0xFF.
-[SWS_EthSwt_00298]as # 77349 introduces a general req

~[SWS_EthSwt_91015]EthSwt_GetPortIdentifier
Return value
E_NOT_OK: organizationally unique identifier of the Ethernet transceiver could not be obtained (i.e. OUI is not available)
Description:
This function retrieves the OUI (24 bit) of the indexed Ethernet switch port.

~[SWS_EthSwt_00299]
The function EthSwt_GetPortIdentifier() shall return the value of the organizationally unique identifier (OUI 24 bit) of the indexed Ethernet switch port that is connected to the indexed Ethernet switch. It shall set the 8 most significant bits of the OUI to 0xFFxxxxxx. If the Ethernet switch port references an Ethernet transceiver, the function shall obtain the OUI by calling the function EthTrcv_GetPhyIdentifier() and set the 8 most significant bits of the OUI to 0x00xxxxxx.

SWS_EthSwt_xxxxx] If neither the Ethernet switch port nor the Ethernet Transceiver Driver can provide an OUI the function EthSwt_GetPortIdentifier() shall return E_NOT_OK.

-[SWS_EthSwt_00304]as # 77349 introduces a general req

~[SWS_EthSwt_91029] EthSwt_SetPortTestMode
Description:

Activates a given test mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00328]
The function EthSwt_SetPortTestMode shall forward the call with the given test mode by calling the function EthTrcv_SetPhyTestMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00333]as # 77349 introduces a general req

~[SWS_EthSwt_91023] EthSwt_SetPortLoopbackMode
Description:
Activates a given test loop-back mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00334]
The function EthSwt_SetPortLoopbackMode() shall forward the call with the given loop-back mode by calling the function EthTrcv_SetPhyLoopbackMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00339] as # 77349 introduces a general req

~[SWS_EthSwt_91024] EthSwt_SetPortTxMode
Description:
Activates a given transmission mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00340]
The function EthSwt_SetPortTxMode() shall forward the call with the given transmission mode by calling the function EthTrcv_SetPhyTxMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00345] as # 77349 introduces a general req

~[SWS_EthSwt_91025]EthSwt_GetPortCableDiagnosticsResult
Description:
Retrieves the cable diagnostics result of the indexed Ethernet switch port respectively the referenced Ethernet Transceiver Driver.

~[SWS_EthSwt_00346]
The function EthSwt_GetPortCableDiagnosticsResult() shall obtain the cable diagnostics result by calling the function EthTrcv_GetCableDiagnosticsResult() of the referenced Ethernet Transceiver Driver. If no Ethernet transceiver is referenced by the Ethernet switch port and development error detection is not enabled, the cable diagnostic result shall be set to ETHTRCV_CABLEDIAG_OK.
-[SWS_EthSwt_00351] as # 77349 introduces a general req

=== EthTrcv ===
add certain parameter to enable/disable API functions:
+ SWS item ECUC_EthTrcv_xxxx1

Name EthTrcvGetPhySignalQualityApi
Description Enables / Disables EthTrcv_GetPhySignalQuality API

+ SWS item ECUC_EthTrcv_xxxx2
Name EthTrcvGetPhyIdentifierApi
Description Enables / Disables EthTrcv_GetPhyIdentifier API

+ SWS item ECUC_EthTrcv_xxxx3
Name EthTrcvSetPhyTestModeApi
Description Enables / Disables EthTrcv_SetPhyTestMode API

+ SWS item ECUC_EthTrcv_xxxx4
Name EthTrcvSetPhyTxModeApi
Description Enables / Disables EthTrcv_SetPhyTxMode API

+ SWS item ECUC_EthTrcv_xxxx5
Name EthTrcvGetCableDiagnosticsResultApi
Description Enables / Disables EthTrcv_GetCableDiagnosticsResult API

add the following specification to the configuration parameter above:
Multiplicity 1
Type EcucBooleanParamDef
Default value –
Post-Build Variant Value false
Value Configuration
Class Pre-compile time X
All Variants Link time –
Post-build time –
Scope / Dependency scope: local
–Last change on issue 77628 comment 31–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.280 Specification Item SWS_EthSwt_91026

Trace References:

SRS_Eth_00123

Content:

Service name:	EthSwt_GetCfgHexDump (obsolete)EthSwt_GetCfgHexDump	
Syntax:	Std_ReturnType EthSwt_GetCfgHexDump(uint8 SwitchIdx, uint32* uint32 CfgBlockNumber, uint32* CfgBlockLengthPtr, uint8* ResultHexDumpBlockPtr)	
Service ID[hex]:	0x3d	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	SwitchIdxEthSwt_GetCfgHexDump.SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	CfgBlockNumberEthSwt_GetCfgHexDump.CfgBlockNumber	Number of the memory block (0...(23̂2)-1). The number is a zero based consecutive range with max (2**32)-1. The CfgBlockNumber is the quotient of the total hex dump length (EthSwt_GetCfgHexDumpLength())devided by the length of one block (*(CfgBlockLengthPtr))
Parameters (inout):	CfgBlockLengthPtrEthSwt_GetCfgHexDump.CfgBlockLengthPtr	Length of the memory block in bytes that shall be copied (in). Return the number of bytes that where copied (out).
Parameters (out):	ResultHexDumpBlockPtrEthSwt_GetCfgHexDump.ResultHexDumpBlockPtr	Pointer to the location where where the memory block shall be copied. The first 4 bytes represent the absolute start address of the memory block.
Return value:	Std_ReturnType	E_OK: the switch hex dump of the configuration was obtained successfully. E_NOT_OK: the switch hex dump of the configuration was not obtained successfully. (i.e. indexed Ethernet switch is not available)
Description:	Retrieves the switch configuration of the indexed Ethernet switch for a certain block with a certain length. Tags: atp.Status=obsolete	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76248: [EthSwt]Remove references to non existing requirements (SRS, SWS)

  Problem description:

  Some review findings brought up references to non existing requirements. Please check and remove the references and if needed reformulate the requirements.

  Agreed solution:

  SWS_EthSwt_00123 –> SRS_BSW_00406
  SWS_EthSwt_00165 -> SRS_BSW_00395
  SWS_EthSwt_00227 -> SRS_ETH_00087

SWS_EthSwt_91017, SWS_EthSwt_91020, –> SRS_ETH_00123
SWS_EthSwt_00006 –> SRS_BSW_00101
SWS_EthSwt_00010 left blank as SRS doe not yet provide a requirement here but maybe with RFC # 75170 AUTOSAR will introduce an SRS-item after all.
SWS_EthSwt_00376, SWS_EthSwt_00374, SWS_EthSwt_00375 –> SRS-Missing! same as for SWS_EThSwt_00010
SWS_EthSwt_00018, SRS_EthSwt_00025 –> SRS_ETH_00118
SWS_EthSwt_00031 –> SRS_ETH_00087
SWS_EthSwt_00037 –> SRS_ETH_00119
SWS_EthSwt_00044, SWS_EthSwt_00051 –> SRS_ETH_00118
SWS_EthSwt_00060 –> SRS_ETH_00087
SWS_EthSwt_00111 –> erase SRS_ETH_00086
SWS_EthSwt00079 –>SRS_ETH_00119
SWS_EthSwt_00206, SWS_EthSwt_00211, SWS_EthSwt_00216, SWS_EthSwt_00221 –>SRS_Eth_00120
SWS_EthSwt_00172 –>SRS_ETH_00121, SRS_ETH_00114
SWS_EthSwt_00086, SWS_EthSwt_00091, SWS_EthSwt_00182, SWS_EthSwt_00125 –> SRS_ETH_00087, SRS_ETH_00122
SWS_EthSwt_00187 –> SRS_ETH_00087
SWS_EthSwt_00058 –> SRS_BSW_00171
SWS_EthSwt_91012, SWS_EthSwt_91013 –> SRS_ETH_00126
SWS_EthSwt_91014, SWS_EthSwt_91015, SWS_EthSwt_91016, SWS_EthSwt_91018, SWS_EthSwt_91019, SWS_EthSwt_91021, SWS_EthSwt_91022, SWS_EthSwt_91022, SWS_EthSwt_91023, SWS_EthSwt_91024, SWS_EthSwt_91025, SWS_EthSwt_91026, SWS_EthSwt_91027 –> SRS_Eth_00123
SWS_EthSwt_00117, SWS_EthSwt_00203 –> SRS_ETH_00119, SRS_ETH_00087
SWS_EthSwt_00193 –> SRS_ETH_00122, SRS_ETH_00087
SWS_EthSwt_00114 –> SRS_BSW_00433
SWS_EthSwt_00098 –> SRS_Eth_00122, SRS_ETH_00118, SRS_ETH_00119, SRS_ETH_00120, SRS_ETH_00087, SRS_ETH_00125, SRS_BSW_00375
–Last change on issue 76248 comment 13–

BW-C-Level:

Application	Specification	Bus
1	1	1

- RfC #76892: [EthSwt] CfgBlockNumber parameter of EthSwt_GetCfgHexDump can be passed as uint32

Problem description:

In chapter 8.3.50 EthSwt_GetCfgHexDump
CfgBlockNumber parameter of EthSwt_GetCfgHexDump should not be passed as a pointer. It is an input parameter that can be passed as an uint32.

Agreed solution:

EthSwt_GetCfgHexDump:
change uint32* CfgBlockNumber to uint32 CfgBlockNumber
–Last change on issue 76892 comment 2–

BW-C-Level:

Application	Specification	Bus
1	4	1

- RfC #77018: [EthSwt] Clarification about the use case regarding API "EthSwt_GetCfgHexDump"

  Problem description:

  We discussed the use case regarding the API "EthSwt_GetCfgHexDump". Daimler want to read out the register memory of an Ethernet switch as a memory dump. The memory dump is transmitted via an diagnostic service over the network consecutively to an diagnostic tester. The memory dump is used to verify and compare to the expected / required values. It is not clear if the API is sufficient for this use case.

  Agreed solution:

  Capter 7.1.2.13 exchange in listing of APIS EthSwt_GetCfgHexDump with EthSwt_GetCfgDataRaw and EthSwt_GetCfgHexDumpLength with EthSwt_GetCfgDataInfo.

  Chapter 8
  -SWS_EthSwt_91026,
  -SWS_EthSwt_00352
  -SWS_EthSwt_00353
  -SWS_EthSwt_00355
  -ECUC_EthSwt_00093
  -SWS_EthSwt_91027
  -SWS_EthSwt_00356
  -SWS_EthSwt_00357
  -SWS_EthSwt_00358
  -SWS_EthSwt_00359
  Set APIs EthSwt_GetCfgHexDump and EthSwt_GetCfgHexDumpLength and config parameters EthSwtGetCfgHexDumpApi and EthSwt_GetCfgHexDumpLengthApi to deprecated or delete.

Chapter 10
Add parameter to enable disable APIs with name EthSwtGetCfgRaw to EthSwtGeneral-Container
Description Disable /Enable support of reading raw data from switch memory.
Multiplicity: 1
Type: EcucBooleanParamDef
Default: false
Post build variant false
Config class Pre-compile time for all variants.
scope: local

Introduce following new APIs to function: definitions
EthSwt_GetCfgDataRaw(
uint8 SwitchIdx,
uint32 Offset,
uint16 Length,
uint8 *BufferPtr
)
[attributes
- asynchronous
- non-reentrant
parameters IN: SwitchIdx:Index of the Ethernet switch within the context of the Ethernet Switch Driver
Offset: Offset of the Ethernet switch memory from where the reading starts
Length: Length of data in bytes that shall be copied.
parameters OUT: BufferPtr: Pointer to the location where the data shall be copied.
Std_ReturnType: E_OK: the data read was triggered successfully. E_NOT_OK: the data read was not triggered successfully. (i.e. indexed Ethernet switch is not available)
Description:
Retrieves the data in memory of the indexed Ethernet switch in variable length.]

+[SWS_EthSwt_xxxxx1] The function EthSwt_GetCfgDataRaw() shall only be available if parameter EthSwtGetCfgRaw is set to TRUE.(SRS_BSW_00171)
+[SWS_EthSwt_xxxxx2] When calling the function EthSwt_GetCfgDataRaw(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.()

EthSwt_GetCfgDataInfo(

uint8 SwitchIdx,
uint32 *DataSizePtr,
uint32 *DataAdressPtr
)
[attributes
- synchronous
- reentrant
parameters IN: SwitchIdx:Index of the Ethernet switch within the context of the Ethernet Switch Driver
parameters OUT: DataSizePtr: Pointer to the location where the total size of the configuration data shall be copied.
DataAddressPtr: Pointer to the location where the start address of the configuration registers shall be copied.
Std_ReturnType: E_OK: the data was obtained successfully. E_NOT_OK: the data was not obtained successfully. (i.e. indexed Ethernet switch is not available)
Description:
Retrieves the total size of data and the memory start address of the indexed Ethernet Switch.]

+[SWS_EthSwt_xxxxx3] The function EthSwt_GetCfgDataInfo() shall only be available if parameter EthSwtGetCfgRaw is set to TRUE.(SRS_BSW_00171)
+[SWS_EthSwt_xxxxx4] When calling the function EthSwt_GetCfgDataInfo(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.()

add following API to chapter "8.6.3 Configurable interfaces"
<GetCfgDataRawDone>(
uint8 SwitchIdx
)
[attributes:
synchronous
reentrant
parameters IN: SwitchIdx: Index of the Ethernet switch where the Configuration is read.
Return value : void
Description: The call of the function EthSwt_GetCfgDataRaw() triggers a asynchrony read of a certain memory section of the Ethernet switch driver. If the read is done, the configured callout function <GetCfgDataRawDone> shall be called]

Add a parameter to EthSwtGeneral-Container
GetCfgDataRawDone
Decription: Defines the function name for <GetCfgDataRawDone>
Multplicity: 0 ..1
Type EcucFuntionNameDef
Variant and Variant Multiplicity false
Config class Pre-compile for all variants
scope: local
dependency: The function GetCfgDataRawDone shall only be configured if parameter EthSwtGetCfgRaw is set to TRUE.

Header file name parameter is defined in ECUC_EthSwt_00064
–Last change on issue 77018 comment 32–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.281 Specification Item SWS_EthSwt_91027

Trace References:

SRS_Eth_00123

Content:

Service name:	EthSwt_GetCfgHexDumpLength (obsolete)EthSwt_GetCfgHexDumpLength	
Syntax:	Std_ReturnType EthSwt_GetCfgHexDumpLength(uint8 SwitchIdx, uint32* ResultTotalLengthPtr)	
Service ID[hex]:	0x3e	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	SwitchIdxEthSwt_GetCfgHexDumpLength.SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
Parameters (inout):	None	
Parameters (out):	ResultTotalLengthPtrEthSwt_GetCfgHexDumpLength.ResultTotalLengthPtr	Total Length of the memory where the configuration of all registers are stored
Return value:	Std_ReturnType	E_OK: the switch hex dump length was obtained successfully. E_NOT_OK: the switch hex dump length was not obtained successfully. (i.e. indexed Ethernet switch is not available)

Description:	Retrieves the total length of the Ethernet switch configuration of the indexed Ethernet switch Tags: atp.Status=obsolete

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76248: [EthSwt]Remove references to non existing requirements (SRS, SWS)

  Problem description:

  Some review findings brought up references to non existing requirements. Please check and remove the references and if needed reformulate the requirements.

  Agreed solution:

  SWS_EthSwt_00123 –> SRS_BSW_00406
  SWS_EthSwt_00165 -> SRS_BSW_00395
  SWS_EthSwt_00227 -> SRS_ETH_00087
  SWS_EthSwt_91017, SWS_EthSwt_91020, –> SRS_ETH_00123
  SWS_EthSwt_00006 –> SRS_BSW_00101
  SWS_EthSwt_00010 left blank as SRS doe not yet provide a requirement here but maybe with RFC # 75170 AUTOSAR will introduce an SRS-item after all.
  SWS_EthSwt_00376, SWS_EthSwt_00374, SWS_EthSwt_00375 –> SRS-Missing! same as for SWS_EThSwt_00010
  SWS_EthSwt_00018, SRS_EthSwt_00025 –> SRS_ETH_00118
  SWS_EthSwt_00031 –> SRS_ETH_00087
  SWS_EthSwt_00037 –> SRS_ETH_00119
  SWS_EthSwt_00044, SWS_EthSwt_00051 –> SRS_ETH_00118
  SWS_EthSwt_00060 –> SRS_ETH_00087
  SWS_EthSwt_00111 –> erase SRS_ETH_00086
  SWS_EthSwt00079 –>SRS_ETH_00119
  SWS_EthSwt_00206, SWS_EthSwt_00211, SWS_EthSwt_00216, SWS_EthSwt_00221 –>SRS_Eth_00120
  SWS_EthSwt_00172 –>SRS_ETH_00121, SRS_ETH_00114
  SWS_EthSwt_00086, SWS_EthSwt_00091, SWS_EthSwt_00182, SWS_EthSwt_00125 –> SRS_ETH_00087, SRS_ETH_00122
  SWS_EthSwt_00187 –> SRS_ETH_00087
  SWS_EthSwt_00058 –> SRS_BSW_00171
  SWS_EthSwt_91012, SWS_EthSwt_91013 –> SRS_ETH_00126
  SWS_EthSwt_91014, SWS_EthSwt_91015, SWS_EthSwt_91016, SWS_EthSwt_91018, SWS_EthSwt_91019, SWS_EthSwt_91021, SWS_EthSwt_91022, SWS_EthSwt_91022, SWS_EthSwt_91023, SWS_EthSwt_91024, SWS_EthSwt_91025, SWS_EthSwt_91026,

SWS_EthSwt_91027 –> SRS_Eth_00123
SWS_EthSwt_00117, SWS_EthSwt_00203 –> SRS_ETH_00119, SRS_ETH_00087
SWS_EthSwt_00193 –> SRS_ETH_00122, SRS_ETH_00087
SWS_EthSwt_00114 –> SRS_BSW_00433
SWS_EthSwt_00098 –> SRS_Eth_00122, SRS_ETH_00118, SRS_ETH_00119, SRS_ETH_00120, SRS_ETH_00087, SRS_ETH_00125, SRS_BSW_00375
–Last change on issue 76248 comment 13–

BW-C-Level:

Application	Specification	Bus
1	1	1

- RfC #77018: [EthSwt] Clarification about the use case regarding API "EthSwt_GetCfgHexDump"

  Problem description:

  We discussed the use case regarding the API "EthSwt_GetCfgHexDump". Daimler want to read out the register memory of an Ethernet switch as a memory dump. The memory dump is transmitted via an diagnostic service over the network consecutively to an diagnostic tester. The memory dump is used to verify and compare to the expected / required values. It is not clear if the API is sufficient for this use case.

  Agreed solution:

  Capter 7.1.2.13 exchange in listing of APIS EthSwt_GetCfgHexDump with EthSwt_GetCfgDataRaw and EthSwt_GetCfgHexDumpLength with EthSwt_GetCfgDataInfo.

  Chapter 8
  -SWS_EthSwt_91026,
  -SWS_EthSwt_00352
  -SWS_EthSwt_00353
  -SWS_EthSwt_00355
  -ECUC_EthSwt_00093
  -SWS_EthSwt_91027
  -SWS_EthSwt_00356
  -SWS_EthSwt_00357
  -SWS_EthSwt_00358
  -SWS_EthSwt_00359
  Set APIs EthSwt_GetCfgHexDump and EthSwt_GetCfgHexDumpLength and config parameters EthSwtGetCfgHexDumpApi and EthSwt_GetCfgHexDumpLengthApi to deprecated or delete.

Chapter 10
Add parameter to enable disable APIs with name EthSwtGetCfgRaw to EthSwtGeneral-Container
Description Disable /Enable support of reading raw data from switch memory.
Multiplicity: 1
Type: EcucBooleanParamDef
Default: false
Post build variant false
Config class Pre-compile time for all variants.
scope: local

Introduce following new APIs to function: definitions
EthSwt_GetCfgDataRaw(
uint8 SwitchIdx,
uint32 Offset,
uint16 Length,
uint8 *BufferPtr
)
[attributes
- asynchronous
- non-reentrant
parameters IN: SwitchIdx:Index of the Ethernet switch within the context of the Ethernet Switch Driver
Offset: Offset of the Ethernet switch memory from where the reading starts
Length: Length of data in bytes that shall be copied.
parameters OUT: BufferPtr: Pointer to the location where the data shall be copied.
Std_ReturnType: E_OK: the data read was triggered successfully. E_NOT_OK: the data read was not triggered successfully. (i.e. indexed Ethernet switch is not available)
Description:
Retrieves the data in memory of the indexed Ethernet switch in variable length.]

+[SWS_EthSwt_xxxxx1] The function EthSwt_GetCfgDataRaw() shall only be available if parameter EthSwtGetCfgRaw is set to TRUE.(SRS_BSW_00171)
+[SWS_EthSwt_xxxxx2] When calling the function EthSwt_GetCfgDataRaw(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.()

EthSwt_GetCfgDataInfo(

uint8 SwitchIdx,
uint32 *DataSizePtr,
uint32 *DataAdressPtr
)
[attributes
- synchronous
- reentrant
parameters IN: SwitchIdx:Index of the Ethernet switch within the context of the Ethernet Switch Driver
parameters OUT: DataSizePtr: Pointer to the location where the total size of the configuration data shall be copied.
DataAddressPtr: Pointer to the location where the start address of the configuration registers shall be copied.
Std_ReturnType: E_OK: the data was obtained successfully. E_NOT_OK: the data was not obtained successfully. (i.e. indexed Ethernet switch is not available)
Description:
Retrieves the total size of data and the memory start address of the indexed Ethernet Switch.]

+[SWS_EthSwt_xxxxx3] The function EthSwt_GetCfgDataInfo() shall only be available if parameter EthSwtGetCfgRaw is set to TRUE.(SRS_BSW_00171)
+[SWS_EthSwt_xxxxx4] When calling the function EthSwt_GetCfgDataInfo(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.()

add following API to chapter "8.6.3 Configurable interfaces"
<GetCfgDataRawDone>(
uint8 SwitchIdx
)
[attributes:
synchronous
reentrant
parameters IN: SwitchIdx: Index of the Ethernet switch where the Configuration is read.
Return value : void
Description: The call of the function EthSwt_GetCfgDataRaw() triggers a asynchrony read of a certain memory section of the Ethernet switch driver. If the read is done, the configured callout function <GetCfgDataRawDone> shall be called]

Add a parameter to EthSwtGeneral-Container
GetCfgDataRawDone
Decription: Defines the function name for <GetCfgDataRawDone>
Multplicity: 0 ..1
Type EcucFuntionNameDef
Variant and Variant Multiplicity false
Config class Pre-compile for all variants
scope: local
dependency: The function GetCfgDataRawDone shall only be configured if parameter EthSwtGetCfgRaw is set to TRUE.

Header file name parameter is defined in ECUC_EthSwt_00064
–Last change on issue 77018 comment 32–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.282 Specification Item SWS_EthSwt_91028

Trace References:

SRS_Eth_00125

Content:

Service name:	EthSwt_PortEnableTimeStampEthSwt_PortEnableTimeStamp	
Syntax:	Std_ReturnType EthSwt_PortEnableTimeStamp(uint8 SwitchCtrlIdx, uint8 PortIdx, Eth_BufIdxType BufIdx, EthSwt_MgmtInfoType* MgmtInfoPtr)	
Service ID[hex]:	0x40	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	SwitchCtrlIdxEthSwt_PortEnableTimeStamp.SwitchCtrlIdx	Index of the switch within the context of the Ethernet Switch Driver Ethernet Controller index
	PortIdxEthSwt_PortEnableTimeStamp.PortIdx	Index of the port at the addressed switch
	BufIdxEthSwt_PortEnableTimeStamp.BufIdx	Index of the message buffer, where Application expects egress time stamping Ethernet Rx Buffer index
	MgmtInfoPtrEthSwt_PortEnableTimeStamp.MgmtInfoPtr	Management information including SwitchIdx and SwitchPortIdx

Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: Time stamping on egress successfully enabled E_NOT_OK: Enabling of time stamping on egress has been failed
Description:	Activates egress time stamping on a dedicated message object on a dedicated port of a Switch if EthSwtPortTimeStampSupport is set to TRUE for this port. The selective activation of dedicated message objects for time stamping reduces the number of notification calls only to the required calls. Some HW does store once the egress time stamp marker and some HW needs it always before transmission. There will be no disabled functionality, due to the fact, that the message type is always "time stamped" by network design.	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77044: [EthSwt] Duplicated information in the parameters of EthSwt_EnableTimeStamping and EthSwt_PortEnableTimeStamp

  Problem description:

  EthSwt_MgmtInfoType is defined in req SWS_EthSwt_91002 as a structure containing the switch index and port index of the switch.

  The API EthSwt_EnableTimeStamping (SWS_EthSwt_91011) gets the switchIdx as input parameter, but also a pointer to management information (containing a switch index and a port index).
  The API EthSwt_PortEnableTimeStamp (SWS_EthSwt_91028) gets the switchIdx and PortIdx as input parameters and a pointer to management information.

  What is the purpose of providing both the management information and the switch/port index?

  Agreed solution:

  Modify:

  ReturnType EthSwt_PortEnableTimeStamp
  (
  uint8 SwitchIdx,
  uint8 PortIdx,
  Eth_BufIdxType BufIdx,
  EthSwt_MgmtInfoType* MgmtInfoPtr
)

  to

ReturnType EthSwt_PortEnableTimeStamp
(
uint8 CtrlIdx,
Eth_BufIdxType BufIdx,
EthSwt_MgmtInfoType* MgmtInfoPtr
)
Parameters (in): CtrlIdx: Ethernet Controller index
BufIdx: Ethernet Rx Buffer index
MgmtInfoPtr: Management information including SwitchIdx and SwitchPortIdx

set to obsolete:

Std_ReturnType EthSwt_EnableTimeStamping
(
uint8 SwitchIdx,
Eth_BufIdxType BufIdx,
EthSwt_MgmtInfoType* MgmtInfoPtr
)
–Last change on issue 77044 comment 12–

BW-C-Level:

Application	Specification	Bus
1	4	1

- RfC #77050: [EthSwt] Uplink ports shall not be excluded from timestamping

Problem description:

[SWS_EthSwt_00244] currently limits timestamping to
"all ... ports except the ports where EthSwtPortRole is set to ETHSWT_UP_LINK_PORT".

This limitation is problematic for cascaded switches since
no timestamps are taken when SYNC messages exit the first
switch and enter the second one.

It may be possible to omit timestamping on the link
between the cascaded switches if the timestamp counters in both switches are synchronized.
It is then possible to use the ingress timestamp on the first switch in combination with any egress timestamp on the second switch.

However, without a synchronization of the timestamp counters in both switches it is necessary to consider ingress and egress (on the uplink port) timestamps on the first switch and ingress (on the uplink port) and egress timestamps on the second switch.

Since it is unclear if all switch devices support timestamp counter synchronization and since this feature may result in additional hardware requirements (e.g., common clock source, reset lines etc.) it may be a show-stopper to omit timestamping at the uplink ports.

I recommend to either just remove this restriction or to make it configurable in case that the hardware supports synchronized counters.

Agreed solution:

~[SWS_EthSwt_00244][If EthSwt_EnableTimeStamp is called for a SwitchIdx, the switch driver shall enable the time-stamping for all his ports where EthSwtPortTimeStampSupport is set to TRUE.]

~[SWS_EthSwt_00378] If EthSwt_PortEnableTimeStamp is called for a PortIdx, the switch driver shall enable the time-stamping for this port if EthSwtPortTimeStampSupport is set to TRUE for this port.

~SWS_EthSwt_91011 EthSwt_EnableTimeStamping
Description: Activates egress time stamping on a dedicated message object on all ports of a Switch where EthSwtPortTimeStampSupport is set to TRUE. The selective activation of dedicated message objects for time stamping reduces the number of notification calls only to the required calls. Some HW does store once the egress time stamp marker and some HW needs it always before transmission. There will be no disabled functionality, due to the fact, that the message type is always "time stamped" by network design.

~SWS_EthSwt_91028 EthSwt_PortEnableTimeStamp
Description: Activates egress time stamping on a dedicated message object on a dedicated port of a Switch if EthSwtPortTimeStampSupport is set to TRUE for this port. The selective activation of dedicated message objects for time stamping reduces the number of notification calls only to the required calls. Some HW does store once the egress time stamp marker and some HW needs it always before transmission. There will be no disabled functionality, due to the fact, that the message type is always "time stamped" by network design.
–Last change on issue 77050 comment 9–

BW-C-Level:

Application	**Specification**	**Bus**
1	4	1

1.283 Specification Item SWS_EthSwt_91029

Trace References:

SRS_Eth_00123

Content:

Service name:	EthSwt_SetPortTestModeEthSwt_SetPortTestMode	
Syntax:	Std_ReturnType EthSwt_SetPortTestMode(uint8 SwitchIdx, uint8 PortIdx, EthTrcv_PhyTestModeType Mode)	
Service ID[hex]:	0x3a	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	SwitchIdxEthSwt_SetPortTestMode.SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	PortIdxEthSwt_SetPortTestMode.PortIdx	Index of the port at the addressed switch
	ModeEthSwt_SetPortTestMode.Mode	Test mode to be activated
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: the port test mode for the indexed Ethernet switch port was set successfully. E_NOT_OK: the port test mode for the indexed Ethernet switch was not set successfully. (i.e. indexed Ethernet switch port is not available)
Description:	Activates a given test mode of the indexed Ethernet switch port.	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76879: [EthTrcv] Add EthSwt API's to optional interfaces

  Problem description:

  Optional interfaces are missing, in case the Ethernet transceiver is connected to a Ethernet switch

Agreed solution:

add the following API's to optional interfaces:
EthSwt_ReadTrcvRegister
EthSwt_WriteTrcvRegister

move the following API's from mandatory interfaces to optional interfaces:
Eth_ReadMii
Eth_WriteMii
–Last change on issue 76879 comment 1–

BW-C-Level:

Application	Specification	Bus
1	1	1

- RfC #77628: [EthSwt] Behaviour of certain APIs for ports without EthSwtPortTrcvRef (part I)

  Problem description:

  The following APIs have partly a description regarding the handling of ports without EthSwtPortTrcvRef:
  EthSwt_GetPortSignalQuality
  EthSwt_GetPortIdentifier
  EthSwt_SetPortTestMode
  EthTrcv_SetPhyTestMode
  EthSwt_SetPortTxMode
  EthSwt_GetPortCableDiagnosticsResult

  The description should be adjusted and harmonized with related requirements.
  –Last change on issue 77628 comment 2–

  Agreed solution:

  === EthSwt ===
  ~[SWS_EthSwt_91014]EthSwt_GetPortSignalQuality
  Description:
  The function retrieves the signal quality of the link of the indexed Ethernet switch port.
  ~[SWS_EthSwt_00293]
  The function EthSwt_GetPortSignalQuality() shall obtain the signal quality by calling the function EthTrcv_GetPhySignalQuality() of the referenced Ethernet Transceiver Driver. If the current signal quality is not available, the signal quality shall be set to 0xFF.
  -[SWS_EthSwt_00298]as # 77349 introduces a general req

~[SWS_EthSwt_91015]EthSwt_GetPortIdentifier
Return value
E_NOT_OK: organizationally unique identifier of the Ethernet transceiver could not be obtained (i.e. OUI is not available)
Description:
This function retrieves the OUI (24 bit) of the indexed Ethernet switch port.

~[SWS_EthSwt_00299]
The function EthSwt_GetPortIdentifier() shall return the value of the organizationally unique identifier (OUI 24 bit) of the indexed Ethernet switch port that is connected to the indexed Ethernet switch. It shall set the 8 most significant bits of the OUI to 0xFFxxxxxx. If the Ethernet switch port references an Ethernet transceiver, the function shall obtain the OUI by calling the function EthTrcv_GetPhyIdentifier() and set the 8 most significant bits of the OUI to 0x00xxxxxx.

SWS_EthSwt_xxxxx] If neither the Ethernet switch port nor the Ethernet Transceiver Driver can provide an OUI the function EthSwt_GetPortIdentifier() shall return E_NOT_OK.

-[SWS_EthSwt_00304]as # 77349 introduces a general req

~[SWS_EthSwt_91029] EthSwt_SetPortTestMode
Description:
Activates a given test mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00328]
The function EthSwt_SetPortTestMode shall forward the call with the given test mode by calling the function EthTrcv_SetPhyTestMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00333]as # 77349 introduces a general req

~[SWS_EthSwt_91023] EthSwt_SetPortLoopbackMode
Description:
Activates a given test loop-back mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00334]
The function EthSwt_SetPortLoopbackMode() shall forward the call with the given loop-back mode by calling the function EthTrcv_SetPhyLoopbackMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00339] as # 77349 introduces a general req

~[SWS_EthSwt_91024] EthSwt_SetPortTxMode
Description:
Activates a given transmission mode of the indexed Ethernet switch port.
~[SWS_EthSwt_00340]
The function EthSwt_SetPortTxMode() shall forward the call with the given transmission mode by calling the function EthTrcv_SetPhyTxMode() of the referenced Ethernet Transceiver Driver.
-[SWS_EthSwt_00345] as # 77349 introduces a general req

~[SWS_EthSwt_91025]EthSwt_GetPortCableDiagnosticsResult
Description:
Retrieves the cable diagnostics result of the indexed Ethernet switch port respectively the referenced Ethernet Transceiver Driver.

~[SWS_EthSwt_00346]
The function EthSwt_GetPortCableDiagnosticsResult() shall obtain the cable diagnostics result by calling the function EthTrcv_GetCableDiagnosticsResult() of the referenced Ethernet Transceiver Driver. If no Ethernet transceiver is referenced by the Ethernet switch port and development error detection is not enabled, the cable diagnostic result shall be set to ETHTRCV_CABLEDIAG_OK.
-[SWS_EthSwt_00351] as # 77349 introduces a general req

=== EthTrcv ===
add certain parameter to enable/disable API functions:
+ SWS item ECUC_EthTrcv_xxxx1
Name EthTrcvGetPhySignalQualityApi
Description Enables / Disables EthTrcv_GetPhySignalQuality API

+ SWS item ECUC_EthTrcv_xxxx2
Name EthTrcvGetPhyIdentifierApi
Description Enables / Disables EthTrcv_GetPhyIdentifier API

+ SWS item ECUC_EthTrcv_xxxx3
Name EthTrcvSetPhyTestModeApi
Description Enables / Disables EthTrcv_SetPhyTestMode API

+ SWS item ECUC_EthTrcv_xxxx4
Name EthTrcvSetPhyTxModeApi
Description Enables / Disables EthTrcv_SetPhyTxMode API

+ SWS item ECUC_EthTrcv_xxxx5
Name EthTrcvGetCableDiagnosticsResultApi

Description Enables / Disables EthTrcv_GetCableDiagnosticsResult API

add the following specification to the configuration parameter above:
Multiplicity 1
Type EcucBooleanParamDef
Default value –
Post-Build Variant Value false
Value Configuration
Class Pre-compile time X
All Variants Link time –
Post-build time –
Scope / Dependency scope: local
–Last change on issue 77628 comment 31–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.284 Specification Item SWS_EthSwt_91030

Trace References:

SRS_Eth_00123

Content:

Service name:	EthSwt_GetCfgDataRawEthSwt_GetCfgDataRaw	
Syntax:	Std_ReturnType EthSwt_GetCfgDataRaw(uint8 SwitchIdx, uint32 Offset, uint16 Length, uint8* BufferPtr)	
Service ID[hex]:	0x41	
Sync/Async:	Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	SwitchIdxEthSwt_GetCfgData Raw.SwitchIdx	Index of the Ethernet switch within the context of the Ethernet Switch Driver
	OffsetEthSwt_GetCfgDataRaw.Offset	Offset of the Ethernet switch memory from where the reading starts
	LengthEthSwt_GetCfgDataRaw.Length	Length of data in bytes that shall be copied

Parameters (inout):	None	
Parameters (out):	BufferPtrEthSwt_GetCfgDataRaw.BufferPtr	Pointer to the location where the data shall be copied
Return value:	Std_ReturnType	E_OK: the data read was triggered successfully E_NOT_OK: the data read was not triggered successfully (i.e. indexed Ethernet switch is not available)
Description:	Retrieves the data in memory of the indexed Ethernet switch in variable length	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77018: [EthSwt] Clarification about the use case regarding API "EthSwt_GetCfgHexDump"

  Problem description:

  We discussed the use case regarding the API "EthSwt_GetCfgHexDump". Daimler want to read out the register memory of an Ethernet switch as a memory dump. The memory dump is transmitted via an diagnostic service over the network consecutively to an diagnostic tester. The memory dump is used to verify and compare to the expected / required values. It is not clear if the API is sufficient for this use case.

  Agreed solution:

  Capter 7.1.2.13 exchange in listing of APIS EthSwt_GetCfgHexDump with EthSwt_GetCfgDataRaw and EthSwt_GetCfgHexDumpLength with EthSwt_GetCfgDataInfo.

  Chapter 8
  -SWS_EthSwt_91026,
  -SWS_EthSwt_00352
  -SWS_EthSwt_00353
  -SWS_EthSwt_00355
  -ECUC_EthSwt_00093
  -SWS_EthSwt_91027
  -SWS_EthSwt_00356
  -SWS_EthSwt_00357
  -SWS_EthSwt_00358
  -SWS_EthSwt_00359
  Set APIs EthSwt_GetCfgHexDump and EthSwt_GetCfgHexDumpLength and config parameters EthSwtGetCfgHexDumpApi and EthSwt_GetCfgHexDumpLengthApi to deprecated or delete.

Chapter 10
Add parameter to enable disable APIs with name EthSwtGetCfgRaw to EthSwtGeneral-Container
Description Disable /Enable support of reading raw data from switch memory.
Multiplicity: 1
Type: EcucBooleanParamDef
Default: false
Post build variant false
Config class Pre-compile time for all variants.
scope: local

Introduce following new APIs to function: definitions
EthSwt_GetCfgDataRaw(
uint8 SwitchIdx,
uint32 Offset,
uint16 Length,
uint8 *BufferPtr
)
[attributes
- asynchronous
- non-reentrant
parameters IN: SwitchIdx:Index of the Ethernet switch within the context of the Ethernet Switch Driver
Offset: Offset of the Ethernet switch memory from where the reading starts
Length: Length of data in bytes that shall be copied.
parameters OUT: BufferPtr: Pointer to the location where the data shall be copied.
Std_ReturnType: E_OK: the data read was triggered successfully. E_NOT_OK: the data read was not triggered successfully. (i.e. indexed Ethernet switch is not available)
Description:
Retrieves the data in memory of the indexed Ethernet switch in variable length.]

+[SWS_EthSwt_xxxxx1] The function EthSwt_GetCfgDataRaw() shall only be available if parameter EthSwtGetCfgRaw is set to TRUE.(SRS_BSW_00171)
+[SWS_EthSwt_xxxxx2] When calling the function EthSwt_GetCfgDataRaw(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.()

EthSwt_GetCfgDataInfo(

uint8 SwitchIdx,
uint32 *DataSizePtr,
uint32 *DataAdressPtr
)
[attributes
- synchronous
- reentrant
parameters IN: SwitchIdx:Index of the Ethernet switch within the context of the Ethernet Switch Driver
parameters OUT: DataSizePtr: Pointer to the location where the total size of the configuration data shall be copied.
DataAddressPtr: Pointer to the location where the start address of the configuration registers shall be copied.
Std_ReturnType: E_OK: the data was obtained successfully. E_NOT_OK: the data was not obtained successfully. (i.e. indexed Ethernet switch is not available)
Description:
Retrieves the total size of data and the memory start address of the indexed Ethernet Switch.]

+[SWS_EthSwt_xxxxx3] The function EthSwt_GetCfgDataInfo() shall only be available if parameter EthSwtGetCfgRaw is set to TRUE.(SRS_BSW_00171)
+[SWS_EthSwt_xxxxx4] When calling the function EthSwt_GetCfgDataInfo(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.()

add following API to chapter "8.6.3 Configurable interfaces"
<GetCfgDataRawDone>(
uint8 SwitchIdx
)
[attributes:
synchronous
reentrant
parameters IN: SwitchIdx: Index of the Ethernet switch where the Configuration is read.
Return value : void
Description: The call of the function EthSwt_GetCfgDataRaw() triggers a asynchrony read of a certain memory section of the Ethernet switch driver. If the read is done, the configured callout function <GetCfgDataRawDone> shall be called]

Add a parameter to EthSwtGeneral-Container
GetCfgDataRawDone
Decription: Defines the function name for <GetCfgDataRawDone>
Multplicity: 0 ..1
Type EcucFuntionNameDef
Variant and Variant Multiplicity false
Config class Pre-compile for all variants
scope: local
dependency: The function GetCfgDataRawDone shall only be configured if parameter EthSwtGetCfgRaw is set to TRUE.

Header file name parameter is defined in ECUC_EthSwt_00064
–Last change on issue 77018 comment 32–

BW-C-Level:

Application	**Specification**	**Bus**
1	4	1

1.285 Specification Item SWS_EthSwt_91031

Trace References:

SRS_Eth_00123

Content:

Service name:	EthSwt_GetCfgDataInfoEthSwt_GetCfgDataInfo	
Syntax:	Std_ReturnType EthSwt_GetCfgDataInfo(uint8 SwitchIdx, uint32* DataSizePtr, uint32* DataAdressPtr)	
Service ID[hex]:	0x42	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	SwitchIdxEthSwt_GetCfgDataInfo.SwitchIdx	Index of the Ethernet switch within the context of the Ethernet Switch Driver
Parameters (inout):	None	

Parameters (out):	DataSizePtrEthSwt_GetCfgDataInfo.DataSizePtr	Pointer to the location where the total size of the configuration data shall be copied
	DataAdressPtrEthSwt_GetCfgDataInfo.DataAdressPtr	Pointer to the location where the start address of the configuration registers shall be copied
Return value:	Std_ReturnType	E_OK: the data was obtained successfully E_NOT_OK: the data was not obtained successfully. (i.e. indexed Ethernet switch is not available)
Description:	Retrieves the total size of data and the memory start address of the indexed Ethernet Switch.	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77018: [EthSwt] Clarification about the use case regarding API "EthSwt_GetCfgHexDump"

  Problem description:

  We discussed the use case regarding the API "EthSwt_GetCfgHexDump". Daimler want to read out the register memory of an Ethernet switch as a memory dump. The memory dump is transmitted via an diagnostic service over the network consecutively to an diagnostic tester. The memory dump is used to verify and compare to the expected / required values. It is not clear if the API is sufficient for this use case.

  Agreed solution:

  Capter 7.1.2.13 exchange in listing of APIS EthSwt_GetCfgHexDump with EthSwt_GetCfgDataRaw and EthSwt_GetCfgHexDumpLength with EthSwt_GetCfgDataInfo.

  Chapter 8
  -SWS_EthSwt_91026,
  -SWS_EthSwt_00352
  -SWS_EthSwt_00353
  -SWS_EthSwt_00355
  -ECUC_EthSwt_00093
  -SWS_EthSwt_91027
  -SWS_EthSwt_00356
  -SWS_EthSwt_00357
  -SWS_EthSwt_00358
  -SWS_EthSwt_00359
  Set APIs EthSwt_GetCfgHexDump and EthSwt_GetCfgHexDumpLength and config parameters EthSwtGetCfgHexDumpApi and EthSwt_GetCfgHexDumpLengthApi to

deprecated or delete.

Chapter 10
Add parameter to enable disable APIs with name EthSwtGetCfgRaw to EthSwtGeneral-Container
Description Disable /Enable support of reading raw data from switch memory.
Multiplicity: 1
Type: EcucBooleanParamDef
Default: false
Post build variant false
Config class Pre-compile time for all variants.
scope: local

Introduce following new APIs to function: definitions
EthSwt_GetCfgDataRaw(
uint8 SwitchIdx,
uint32 Offset,
uint16 Length,
uint8 *BufferPtr
)
[attributes
- asynchronous
- non-reentrant
parameters IN: SwitchIdx:Index of the Ethernet switch within the context of the Ethernet Switch Driver
Offset: Offset of the Ethernet switch memory from where the reading starts
Length: Length of data in bytes that shall be copied.
parameters OUT: BufferPtr: Pointer to the location where the data shall be copied.
Std_ReturnType: E_OK: the data read was triggered successfully. E_NOT_OK: the data read was not triggered successfully. (i.e. indexed Ethernet switch is not available)
Description:
Retrieves the data in memory of the indexed Ethernet switch in variable length.]

+[SWS_EthSwt_xxxxx1] The function EthSwt_GetCfgDataRaw() shall only be available if parameter EthSwtGetCfgRaw is set to TRUE.(SRS_BSW_00171)
+[SWS_EthSwt_xxxxx2] When calling the function EthSwt_GetCfgDataRaw(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.()

EthSwt_GetCfgDataInfo(
uint8 SwitchIdx,
uint32 *DataSizePtr,
uint32 *DataAdressPtr
)
[attributes
- synchronous
- reentrant
parameters IN: SwitchIdx:Index of the Ethernet switch within the context of the Ethernet Switch Driver
parameters OUT: DataSizePtr: Pointer to the location where the total size of the configuration data shall be copied.
DataAddressPtr: Pointer to the location where the start address of the configuration registers shall be copied.
Std_ReturnType: E_OK: the data was obtained successfully. E_NOT_OK: the data was not obtained successfully. (i.e. indexed Ethernet switch is not available)
Description:
Retrieves the total size of data and the memory start address of the indexed Ethernet Switch.]

+[SWS_EthSwt_xxxxx3] The function EthSwt_GetCfgDataInfo() shall only be available if parameter EthSwtGetCfgRaw is set to TRUE.(SRS_BSW_00171)
+[SWS_EthSwt_xxxxx4] When calling the function EthSwt_GetCfgDataInfo(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.()

add following API to chapter "8.6.3 Configurable interfaces"
<GetCfgDataRawDone>(
uint8 SwitchIdx
)
[attributes:
synchronous
reentrant
parameters IN: SwitchIdx: Index of the Ethernet switch where the Configuration is read.
Return value : void
Description: The call of the function EthSwt_GetCfgDataRaw() triggers a asynchrony read of a certain memory section of the Ethernet switch driver. If the read is done, the configured callout function <GetCfgDataRawDone> shall be called]

Add a parameter to EthSwtGeneral-Container
GetCfgDataRawDone
Decription: Defines the function name for <GetCfgDataRawDone>
Multplicity: 0 ..1
Type EcucFuntionNameDef
Variant and Variant Multiplicity false
Config class Pre-compile for all variants
scope: local
dependency: The function GetCfgDataRawDone shall only be configured if parameter EthSwtGetCfgRaw is set to TRUE.

Header file name parameter is defined in ECUC_EthSwt_00064
–Last change on issue 77018 comment 32–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.286 Specification Item SWS_EthSwt_91032

Trace References:

SRS_Eth_00123

Content:

Service name:	<GetCfgDataRawDone><GetCfgDataRawDone>	
Syntax:	void <GetCfgDataRawDone>(uint8 SwitchIdx)	
Service ID[hex]:	0x43	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	SwitchIdx<GetCfgDataRawDone>.SwitchIdx	Index of the Ethernet switch where the Configuration is read.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	None	

Description:	The call of the function EthSwt_GetCfgDataRaw() triggers a asynchrony read of a certain memory section of the Ethernet switch driver. If the read is done, the configured callout function <GetCfgDataRawDone> shall be called]

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77018: [EthSwt] Clarification about the use case regarding API "EthSwt_GetCfgHexDump"

  Problem description:

  We discussed the use case regarding the API "EthSwt_GetCfgHexDump". Daimler want to read out the register memory of an Ethernet switch as a memory dump. The memory dump is transmitted via an diagnostic service over the network consecutively to an diagnostic tester. The memory dump is used to verify and compare to the expected / required values. It is not clear if the API is sufficient for this use case.

  Agreed solution:

  Capter 7.1.2.13 exchange in listing of APIS EthSwt_GetCfgHexDump with EthSwt_GetCfgDataRaw and EthSwt_GetCfgHexDumpLength with EthSwt_GetCfgDataInfo.

  Chapter 8
  -SWS_EthSwt_91026,
  -SWS_EthSwt_00352
  -SWS_EthSwt_00353
  -SWS_EthSwt_00355
  -ECUC_EthSwt_00093
  -SWS_EthSwt_91027
  -SWS_EthSwt_00356
  -SWS_EthSwt_00357
  -SWS_EthSwt_00358
  -SWS_EthSwt_00359
  Set APIs EthSwt_GetCfgHexDump and EthSwt_GetCfgHexDumpLength and config parameters EthSwtGetCfgHexDumpApi and EthSwt_GetCfgHexDumpLengthApi to deprecated or delete.

  Chapter 10
  Add parameter to enable disable APIs with name EthSwtGetCfgRaw to EthSwtGeneral-Container
  Description Disable /Enable support of reading raw data from switch memory.
  Multiplicity: 1
  Type: EcucBooleanParamDef

Default: false
Post build variant false
Config class Pre-compile time for all variants.
scope: local

Introduce following new APIs to function: definitions
EthSwt_GetCfgDataRaw(
uint8 SwitchIdx,
uint32 Offset,
uint16 Length,
uint8 *BufferPtr
)
[attributes
- asynchronous
- non-reentrant
parameters IN: SwitchIdx:Index of the Ethernet switch within the context of the Ethernet Switch Driver
Offset: Offset of the Ethernet switch memory from where the reading starts
Length: Length of data in bytes that shall be copied.
parameters OUT: BufferPtr: Pointer to the location where the data shall be copied.
Std_ReturnType: E_OK: the data read was triggered successfully. E_NOT_OK: the data read was not triggered successfully. (i.e. indexed Ethernet switch is not available)
Description:
Retrieves the data in memory of the indexed Ethernet switch in variable length.]

+[SWS_EthSwt_xxxxx1] The function EthSwt_GetCfgDataRaw() shall only be available if parameter EthSwtGetCfgRaw is set to TRUE.(SRS_BSW_00171)
+[SWS_EthSwt_xxxxx2] When calling the function EthSwt_GetCfgDataRaw(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.()

EthSwt_GetCfgDataInfo(
uint8 SwitchIdx,
uint32 *DataSizePtr,
uint32 *DataAdressPtr
)
[attributes
- synchronous

- reentrant
parameters IN: SwitchIdx:Index of the Ethernet switch within the context of the Ethernet Switch Driver
parameters OUT: DataSizePtr: Pointer to the location where the total size of the configuration data shall be copied.
DataAddressPtr: Pointer to the location where the start address of the configuration registers shall be copied.
Std_ReturnType: E_OK: the data was obtained successfully. E_NOT_OK: the data was not obtained successfully. (i.e. indexed Ethernet switch is not available)
Description:
Retrieves the total size of data and the memory start address of the indexed Ethernet Switch.]

+[SWS_EthSwt_xxxxx3] The function EthSwt_GetCfgDataInfo() shall only be available if parameter EthSwtGetCfgRaw is set to TRUE.(SRS_BSW_00171)
+[SWS_EthSwt_xxxxx4] When calling the function EthSwt_GetCfgDataInfo(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.()

add following API to chapter "8.6.3 Configurable interfaces"
<GetCfgDataRawDone>(
uint8 SwitchIdx
)
[attributes:
synchronous
reentrant
parameters IN: SwitchIdx: Index of the Ethernet switch where the Configuration is read.
Return value : void
Description: The call of the function EthSwt_GetCfgDataRaw() triggers a asynchrony read of a certain memory section of the Ethernet switch driver. If the read is done, the configured callout function <GetCfgDataRawDone> shall be called]

Add a parameter to EthSwtGeneral-Container
GetCfgDataRawDone
Decription: Defines the function name for <GetCfgDataRawDone>
Multplicity: 0 ..1
Type EcucFuntionNameDef
Variant and Variant Multiplicity false

Config class Pre-compile for all variants
scope: local
dependency: The function GetCfgDataRawDone shall only be configured if parameter EthSwtGetCfgRaw is set to TRUE.

Header file name parameter is defined in ECUC_EthSwt_00064
–Last change on issue 77018 comment 32–

BW-C-Level:

Application	Specification	Bus
1	4	1