ID	Category	Summary	Description	ASR_TicketType	Status
22712	General	Usage of value INF not according ASR requirements	Problem description: According AUTOSAR_TPS_ECUConfiguration the value 'inf' derived from standard module definition STMD must be used as follows: • [ecuc_sws_6045] If the min value equals -inf or the max value equals inf in the StMD the min/max values in the VSMD shall be replaced with the actually supported min/max values for this implementation. Expected behavior: INF shall not be used, but instead the actual MIN/MAX values shall be available in PDFs Current behavior: See problem description field.	BUG	OPEN ISSUE
26927	General	[Port] AUTOSAR_PORT_Component_UserManual.pdf is lacking information about Exclusives areas for CRITICAL SECTION PROTECTION	Problem description: Lack of information about Exclusives areas in AUTOSAR_PORT_Component_UserManual.pdf. As a result user is facing difficulty during integration. Source Package: AUTOSAR_RH850_P1x_MCAL_E4.03 Expected Behavior: The user manual should contain information about INIT_CONFIG_PROTECTION, REFRESH_PORT_INTERNAL_PROTECTION and SET_TO_DIO_ALT_PROTECTION. Actual behaviour: UM only describes SET_PIN_MODE_PROTECTION in chapter 4.4, but SET_PIN_DIR_PROTECTION, INIT_CONFIG_PROTECTION, REFRESH_PORT_INTERNAL_PROTECTION and SET_TO_DIO_ALT_PROTECTION, SET_PIN_DEFAULT_MODE_PROTECTION, SET_PIN_DEFAULT_DIR_PROTECTION are not mentioned.	BUG	OPEN ISSUE
26988	General	CAN and LIN modules not following Autosar requirement BSW00347	Problem description: As per AUTOSAR requirement BSW00347, the driver modules shall be named as per <MSN>_<VendorId>_<VendorSpecificName>_<ServiceName>. For Example : 'Can_Init()' will become 'Can_59_Renesas_Init()'. It shall be followed for File Names, Public APIs, Published Parameters, Memory allocation Keywords and Public data types. But this is not followed in CAN and LIN modules which support multiple instance as per autosar base definition file. Expected behaviour: The driver modules shall be named as per <MSN>_<VendorId>_<VendorSpecificName>_<ServiceName>. Actual behaviour:♂ The driver modules(CAN and LIN) are not named as per <MSN>_<VendorId>_<VendorSpecificName>_<ServiceName>. The following MCAL modules have the tag 'UPPER-MULTIPLICITY-INFINITE' is set to 'true' in Autosar Base Definition file AUTOSAR_MOD_ECUConfigurationParameters.arxml and hence support multiple instance. 1. CAN 2. Ethernet 3. FLS 4. Flexray 5. ICU 6. LIN 7. PWM 8. WDG But for Ethernet, FLS, ICU, and PWM modules, the requirement BSW00347 is moved to NA requirements in the Traceability section.	BUG	OPEN ISSUE

27639	General	PRAGMA define inconsistent to device header file package	<u>Problem Description:</u> PRAGMA define differs between io_macros_v2.h from device header file packages and compiler.h in MCAL package. In Compiler.h: #define PRAGMA(x) _Pragma(x) In io_macros_v2.h: #define PRAGMA(x) _Pragma(#x) <u>Current Behaviour:</u> In customer application this might cause a compilation warning due to a macro redefinition if both header files are used. <u>Expected Behaviour:</u> Consistent define used in both header files.	BUG	OPEN ISSUE
27721	General	Command line option -F not working	Problem description: The -F/FILEVERSION option of generation tool is not working. Instead of listing the version of tool code files, the tool is throwing error 'ERR000011:ECU Configuration Description File is not provided as input to the Generation Tool'. Expected behavior: On the usage of -F/FILEVERSION option, generation tool must list the version of tool code files. Actual behaviour: See Problem description.	BUG	OPEN ISSUE
27747	General	Makefiles use invalid include paths for GHS builder	Problem Description: The GHS makefiles for sample applications use invalid include paths parameters. This behaviour has currently no effect to GHS builder but this might change. It lengthens the command lines without any use. Actual behaviour: GHS builder is called with invalid options like -I\4.0.3 -I\common Expected behaviour: Only valid include path parameters shall be used.	BUG	OPEN ISSUE
27766	General	Functional codes are executing Even after DEM is reported.	Problem Description: Even after DEM error is reported, functional codes are getting executed, which may result in unexpected behavior of driver. Similar issue is found in SPI while doing functional testing for E1x V4.00.04 release, And an issue is reported in mantis # bug:26731. Decided to create new ticket to start investigation for similar issues in all other module (see note:181849). Expected behavior: Functional codes shall not execute after reporting DEM Error. Actual behavior: Functional codes are executed even after reporting DEM Error.	BUG	OPEN ISSUE

27974	General	Makefiles specify irrelevant folders for header search	Problem description: The -I parameter is used to specify folders where the GHS builder shall search for header files. But also source folders are given. Actual behaviour: Many irrelevant folders are given as parameter to GHS builder. When project becomes large the maximum command line length (8k) is exceeded. Expected behaviour: Only relevant folders shall be given with -I parameter.	BUG	OPEN ISSUE
28478	General	CAN-ENTER-EXCLUSIVE-AREA-REF tag is missing in the BSWMDT	canEnterExclusiveArea is required inside the entity in BSWMDT, if the referenced exclusive area is used in the entity's code. The entity can be BswCalledEntity, BswSchedulableEntity or BswInterruptEntity. Actual behaviour: <BSW-INTERNAL-BEHAVIOR UUID="ECUS:951843a9-6848-4a8d-869a-7b29a87158fa"> <SHORT-NAME>BswInternalBehavior_0</SHORT-NAME> <EXCLUSIVE-AREA UUID="ECUS:1b965de4-5e4a-49d8-a4cb-e7782386f347"> <SHORT-NAME>VARIABLE_PROTECTION</SHORT-NAME> </EXCLUSIVE-AREA> </EXCLUSIVE-AREAS> <ENTITYS> <BSW-INTERRUPT-ENTITY UUID="ECUS:8d9d01cd-59a4-4f9d-a0c5-a46966e1b2ad"> <SHORT-NAME>BswInterruptEntity_1</SHORT-NAME> <IMPLEMENTED-ENTRY-REF DEST="BSW-MODULE-ENTRY">/ArPackage_0/MCU_FEINT_ISR</IMPLEMENTED-ENTRY-REF> <INTERRUPT-CATEGORY>CAT-1</INTERRUPT-CATEGORY> <INTERRUPT-SOURCE>INTLVI</INTERRUPT-SOURCE> </BSW-INTERRUPT-ENTITY> Expected behaviour: <BSW-INTERNAL-BEHAVIOR UUID="ECUS:951843a9-6848-4a8d-869a-7b29a87158fa"> <SHORT-NAME>BswInternalBehavior_0</SHORT-NAME> <EXCLUSIVE-AREA UUID="ECUS:1b965de4-5e4a-49d8-a4cb-e7782386f347"> <SHORT-NAME>VARIABLE_PROTECTION</SHORT-NAME> </EXCLUSIVE-AREA>	BUG	OPEN ISSUE
28534	General	Wrong upper multiplicity definition for Configuration container.	Problem description: In PDF of some MCAL modules the upper multiplicity is defined as <UPPER-MULTIPLICITY-INFINITE>XX</UPPER-MULTIPLICITY-INFINITE> The above definitions are not correct according to ASR ecuc_sws_2011. Actual behavior: Multiple configuration is not possible due to the above problem. Expected behavior: The correct definitions must be as follows: <UPPER-MULTIPLICITY>XX</UPPER-MULTIPLICITY>	BUG	OPEN ISSUE

26812	ADC	Unexpected DET ADC_E_IDLE is been raised from Adc_DisableHardware Trigger	Problem Description: Unexpected DET ADC_E_IDLE is being raised when Adc_DisableHardwareTrigger is invoked for an already enabled group (using the api Adc_EnableHardwareTrigger)whose status is ADC_STREAM_COMPLETED. Expected Behaviour : As per requirement ADC304, the DET ADC_E_IDLE should not be reported when Adc_DisableHardwareTrigger is called for a group that has already been enabled using Adc_EnableHardwareTrigger. Actual Behaviour : The DET ADC_E_IDLE is reported when Adc_DisableHardwareTrigger is called for a group that has already been enabled using Adc_EnableHardwareTrigger.	BUG	OPEN ISSUE
27492	ADC	HW triggered One-shot conversion in Circular Streaming is not working as expected	Problem Description: As per AUTOSAR specification, one HW trigger should trigger only one ADC channel group conversion stream. The conversion must finish once it receives the HW trigger that is equal to number of streams configured for the group. But in the current design single trigger executes the whole stream of conversions. Expected Behavior: Only one ADC channel Group conversion stream should happen per H/W trigger. Actual Behavior: Streaming conversion is getting completed with single H/W trigger	BUG	OPEN ISSUE
27505	ADC	'ucGroupSettings' element of Adc_GstGroupConfig[] is not generated properly	Problem Description: LSB of 'ucGroupSettings' element decides whether a group is one-shot or continuous. '0' means continuous group and '1' means one-shot group. But code generator is not generating this properly. For one-shot mode and circular streaming group it generates '1' and for continuous mode and linear streaming group it generates '0' Expected Behavior: LSB of 'ucGroupSettings' must be '0' for continuous group and '1' for one-shot group. Actual Behavior: Code generator is generating the below values, For one-shot mode and circular streaming group it is generates '1' and for continuous mode and linear streaming group it generates '0'	BUG	OPEN ISSUE
27592	ADC	DMA enabled ADC conversion gives wrong conversion result if DTFRRQn register signals a pending transfer request	Problem description: Conversion of an ADC group with 'AdcResultAccessMode' => 'ADC_ISR_ACCESS' (Group without DMA) leads to the flagging of DTFRRQn.DRQ bit. Conversion of an ADC group with 'AdcResultAccessMode' => 'ADC_DMA_ACCESS' (DMA enabled group) when DTFRRQn.DRQ bit already set leads to the return of converted value from previous cycle. Therefore DTFRRQn.DRQ bit must be cleared each time when DMA enabled ADC groups conversion is started. Expected Behavior: DMA enabled ADC Groups should return the converted result of the current input voltage. Actual Behavior: DMA enabled ADC Groups are not returning the converted result of the current input voltage because of already set DTFRRQn.DRQ bit.	BUG	OPEN ISSUE

27603	ADC	Conversion of a DMA enabled one-shot ADC group is happening only for the first HW trigger and not for the next triggers	Problem description: One-shot ADC groups with DMA as result access mode is getting converted only for the first trigger and not for the consecutive triggers. Note that if the result access mode is selected as interrupt then it is working as expected. Expected Behavior: One-shot ADC groups with DMA as result access mode should convert the group channels for every HW trigger until it is stopped explicitly by Adc_DisableHardwareTrigger() Actual Behavior: One-shot ADC groups with DMA as result access mode is getting converted only for the first trigger and not for the consecutive triggers after enabling the group via Adc_EnableHardwareTrigger()	BUG	OPEN ISSUE
27613	ADC	HW triggered ADC group with DMA circular streaming access should convert one sample per one trigger	Problem description: Current behaviour of the driver is, HW triggered ADC group with DMA and circular streaming access mode is converts more than one sample per H/W trigger. Sample conversion continues until the group is stopped by calling Adc_DisableHardwareTrigger(). As per the AUTOSAR specification only one sample conversion must be initiated for one HW trigger. Expected Behavior: As per the AUTOSAR specification only one sample conversion must be initiated for one HW trigger. Actual Behavior: Stream conversion continues with a single HW trigger.	BUG	OPEN ISSUE
27624	ADC	Adc_GetStreamLastPointer() API does not return the valid sample count when the status of the group is ADC_STREAM_COMPLETED	Problem description: Adc_GetStreamLastPointer() API does not return correct number of valid samples when the status of the circular streaming group is ADC_STREAM_COMPLETED. This API must return the value equal to the configured 'AdcStreamingNumSamples' parameter of that group. Expected Behavior: Adc_GetStreamLastPointer() API must return the maximum sample value when the status of the circular streaming group is ADC_STREAM_COMPLETED. Actual Behavior: Adc_GetStreamLastPointer() API is returning random value when the status of the group is ADC_STREAM_COMPLETED.	BUG	OPEN ISSUE

28094	ADC	Value assigned to register (ADCDnTHGSR) is not correct in API Adc_HwInit().	Problem Description:-
			In Private API Adc_HwInit() value assigned to register (ADCDnTHGSR) usADCXnTHGSR is not correct, assignment to this register is as mentioned below.
			
			 LpAdcRegisters->usADCXnTHGSR = (uint16)(LpHwUnitConfig->ucGroupSelectMask);
			
			Generated value in structure element "ucGroupSelectMask" is not correct, Thereby selection of T&H group B will not work properly.
			
			In tool code value for structure element "ucGroupSelectMask" is as mentioned below. Here final value in "ucGroupSelectMask" is the value in local variable $grp_mask, Understanding is that for register ADCDnTHGSR bit positions are in even number, as mentioned in section 30.3.23 of Device manual R01UH0436EJ0070 Rev.0.70, But in Generation Tool code its considered that ADCDnTHGSR register bit positions are continues, As mentioned in Actual Behavior.
			
			Please check and do needful.
			
			Actual Behavior:-
			@Line No 749 of BswPbIm.pm file, SVN Revision $185322
			Looping below mentioned code for each channel.
			# Fill ucGroupSelectMask
			$grp_mask = 0;
			$BswPbIm::Dbrom_PbImage{Adc_GstHWUnitConfig}{$index}
			{ucGroupSelectMask} = $grp_mask;
			if ($chn_trck eq "ADC_TH_GROUPB")
			{
			$grp_mask = $grp_mask | (1 << $ch_id);
			$BswPbIm::Dbrom_PbImage{Adc_GstHWUnitConfig}{$index}
			{ucGroupSelectMask} = $grp_mask;
			}
				BUG	OPEN ISSUE

28109	ADC	Limit check implementation is not proper for Polling mode	Problem description: In the case of Polling, Adc_ReadGroup API is called multiple times in a Loop. From Adc_PollingReadGroup function(called form Adc_ReadGroup API), Adc_ErrIsr is called. In Adc_ErrIsr the following errors are being cleared by writing 0x0E to ADCDnECR register. 1. Upper Limit/Lower Limit Error. 2. Overwrite Error. 3. Parity Error Clear. In next polling(calling Adc_ReadGroup), even if the value is beyond Limit/Lower limit, ADC group notification is called and the conversion status is changed to ADC_STREAM_COMPLETED. Expected Behavior: ADC driver should trigger the next conversion even if it is configured for one-shot conversion. Actual Behaviour: Further conversion is not triggered for one-shot groups. ADC group notification is called and the conversion status is changed to ADC_STREAM_COMPLETED.	BUG	OPEN ISSUE

28111	ADC	Conversion is not happening for TH groups when parameter 'AdcSuspendMode' is 'ADC_ASYNCHRONOUS_SUSPEND' and the Trigger is SW	Problem Description: Conversion is happening for ADC groups which contains Track and Hold enabled channels when the configuration parameter 'AdcSuspendMode' in container 'AdcHwUnit' is configured as 'ADC_ASYNCHRONOUS_SUSPEND' and the parameter 'AdcGroupTriggSrc' is 'ADC_TRGG_SRC_SW'. The track and hold functionality is working fine for HW triggered groups even if the parameter 'AdcSuspendMode' is 'ADC_ASYNCHRONOUS_SUSPEND'. Also, it is working properly for both SW and HW triggered groups if the parameter 'AdcSuspendMode' is 'ADC_SYNCHRONOUS_SUSPEND' But if the parameter 'AdcSuspendMode' is configured as 'ADC_SYNCHRONOUS_SUSPEND' the generator tool produces the following warning. "The parameter 'AdcSuspendMode' should be configured as <ADC_ASYNCHRONOUS_SUSPEND> when the channels are enabled for Track and hold feature." Expected Behavior: conversion should be happened for both SW and HW triggered groups if parameter 'AdcSuspendMode' is 'ADC_ASYNCHRONOUS_SUSPEND' Actual Behaviour: Conversion is not happening if parameter 'AdcSuspendMode' is 'ADC_ASYNCHRONOUS_SUSPEND' for SW triggered groups.	BUG	OPEN ISSUE
28118	ADC	Critical section protection for global structure array "Adc_GpGroupRamData" is not implemented.	Problem Description : Critical section protection for global structure array "Adc_GpGroupRamData" is not implemented, even though the values of the structure "Adc_GpGroupRamData" is modified in most of the APIs. In most of the private API, address of global structure array "Adc_GpGroupRamData" is assigned to a local pointer and then the elements values are changed / modified / read. Even though the values are changed using local pointer, the value in global structure array "Adc_GpGroupRamData" also gets changed. Example : The value of structure element "ddGroupStatus" of "Adc_GpGroupRamData" global structure is being modified in Adc_GroupCompleteMode() private API called from Adc_Isr(). /* Set group status as conversion completed */ LpGroupData->ddGroupStatus = ADC_STREAM_COMPLETED; Consider a situation in which API Adc_StopGroupConversion() is called from a high priority task (say Timer Isr) than that of Adc_Isr(), And if Adc_StopGroupConversion() is called (for same group) just after start exciting Adc_Isr(), but not reached above mentioned code. In this case the status Group Status remain ADC_STREAM_COMPLETED, even after calling Adc_StopGroupConversion(). To avoid such issues, critical sections need to be implemented properly. Required MO suggestion on same. Actual Behavior : Critical section protection not implemented.	BUG	OPEN ISSUE

28120	ADC	Autosar requirement ADC413 is not taken care.	Problem Description : As per AUTOSAR requirement ADC413 all API functions, except Adc_Init, Adc_DeInit and Adc_GetVersionInfo are re-entrant. But in current implementation it is not taken care. If the functions are called for different channel groups, in current implementation these re-entrant API will not work properly. Example:- Consider that SW triggered ADC channel group 0 is already in queue. Considered that we call Adc_ReadGroup() for ADC group 1 and API Adc_DisableHardwareTrigger() is invoked for ADC Channel group 1 form a interrupt (Like Timer ISR) when only DET check in Adc_ReadGroup() API is completed. Then execution of Adc_ReadGroup() is pushed to stack and start execution of Adc_DisableHardwareTrigger() API. When Adc_DisableHardwareTrigger() API complete execution, it pops the ADC channel group 0 from queue, trigger its conversion, and when Adc_ReadGroup() start execution, it will give unexpected behavior. similar issues exist in most of the API's, Requesting MO suggestions on same. Expected Behavior : Requirement should be taken care properly and API's except above mentioned should be re-entrant. Actual Behavior : Requirement is not taken care.	BUG	OPEN ISSUE

28121	ADC	Memory section mapping used for global variable "Adc_GaaHwUnitIndex[]" and "Adc_GaaResultGroupRamData[]" is not correct.	Probable Description:-
			1.The global variable "Adc_GaaResultGroupRamData[]" is mapped to memory section "CONFIG_DATA_UNSPECIFIED_SEC_STARTED" (ADC_START_SEC_CONFIG_DATA_UNSPECIFIED), But this global variable is not initialized in Adc_PBcfg.c (Generated file).
			
			Declaration of this variable is as mentioned below.
			
			extern VAR(Adc_ValueGroupType, ADC_NOINIT_DATA) Adc_GaaResultGroupRamData[];
			of Adc_Ram.c file.
			
			2. The global variable "Adc_GaaHwUnitIndex[]" is mapped to memory section "VAR_NOINIT_UNSPECIFIED_SEC_STARTED" (ADC_START_SEC_CONFIG_VAR_NOINIT_UNSPECIFIED), But this global variable is of type const and initialized in Adc_PBcfg.c (Generated file).
			
			Declaration of this variable is as mentioned below.
			
			extern CONST(uint8, ADC_CONST) Adc_GaaHwUnitIndex[];
			of Adc_Ram.c file.
			
			Suggested Solution:
			
			1.Adc_GaaResultGroupRamData[] global variable needs to mapped to Uninitialized variable section.
			
			2.Adc_GaaHwUnitIndex[] global variable needs to mapped to initialized constant variable section.
			
			Expected Behavior:- NA
			
			Actual Behavior:- NA
				BUG	OPEN ISSUE

28125	ADC	Register ADCDnTHSTPCR (ucADCXnTHSTPCR) needs to use instead of ADCDnTHSMPSTCR (ucADCXnTHSMPSTCR) to stop T&H.	Problem Description : Register ADCDnTHSTPCR (ucADCXnTHSTPCR) needs to use instead of ADCDnTHSMPSTCR (ucADCXnTHSMPSTCR) to stop TRACK & HOLD in following mentioned Line of code. As per device manual R01UH0436EJ0070 Rev.0.70 section 30.3.15 ADCDnTHSMPSTCR register is used for starting T&H. As per device manual R01UH0436EJ0070 Rev.0.70 section 30.3.16 ADCDnTHSTPCR register is used for stop T&H. 1. LpAdcRegisters->ucADCXnTHSMPSTCR = ADC_ZERO; of Adc_Private_ADCD_ADCB.c file in Private API Adc_HwDeInit(). 2. LpAdcRegisters->ucADCXnTHSMPSTCR = ADC_BYTE_ZERO; of Adc_Private_ADCD_ADCB.c file in Private API Adc_HwStopGroupConversion(). 3. Also Adc_Init() needs to update to stop T&H by setting this register. Actual Behavior : Register use ADCDnTHSMPSTCR (ucADCXnTHSMPSTCR) to stop TRACK & HOLD. Expected Behavior : Register ADCDnTHSTPCR (ucADCXnTHSTPCR) needs to be use to stop TRACK & HOLD.	BUG	OPEN ISSUE
28134	ADC	Implementation of MRS requirement "AR_PN0076_FR_0201" is not proper.	Problem Description: If we call Adc_EnableChannel() API before calling Adc_DisableChannel() API, illegal memory access will occur. Because when we call Adc_EnableChannel() API internally private API Adc_IntDisableEnableChannel() will call as mentioned below. Adc_IntDisableEnableChannel(Group, ChannelId, ADC_TRUE) and in private API Adc_IntDisableEnableChannel(), if LblApiType(3rd argument) == ADC_TRUE, then, decrement the number of channels to disabled,as mentioned in below code LpGroupData->ucNoofChDisabled--; Initial value of "ucNoofChDisabled" is zero, so after decrement it become 255, which is not correct result in illegal memory access or unexpected behavior in API Adc_ReadGroup(), Adc_GetStreamLastPointer(), Adc_ConfigureGroupForConversion() and Adc_IsrConfigureGroupForConversion(). Expected Behavior: N/A Actual Behavior: N/A Suggested solution: Add a DET check if particular channel is trying to enable before it is disabled.	BUG	OPEN ISSUE
28135	ADC	Version check for Dem.h file is not present.	Problem description: As per autosar requirement ADC124 The ADC module shall perform Inter Module Checks to avoid integration of incompatible files. But version check for Dem.h file is not present in the above mentioned file. FILE : Adc_Private_ADCD_ADCB.c Expected Behavior : Version check should be done. Actual Behavior : No version check is performed.	BUG	OPEN ISSUE

28143	ADC	General requirement "AR_PN0034_FR_0025" is not considered.	Problem description : The DMA related registers are not initialized in Adc_init(). The same issue is there with following registers also: 1. ADCDnTHBCR (ucADCXnTHBCR), 2. ADCDnSGCRx (ucADCXnSGCRx), 3. ADCDnSGVCSPx (ucADCXnSGVCSPx) 4. ADCDnSGVCEPx (ucADCXnSGVCEPx) 5. ADCDnSGMCYCRx (ucADCXnSGMCYCRx) 6. ADCDnULLMSRx (ucADCXnULLMSRx) 7. ADCDnADTIPRy (ulADCXnADTIPRy) 8. ADOPDIGn (ulADOPDIGn) 9. ADCDnTHACR 10. ADCDnTHCR 11. ADCXnULLMTBR 0 to 2 are not initialized in Adc_Init(). But as per General MRS "AR_PN0034_FR_0025" : The <MSN>_Init API shall ensure that the related peripheral is running correctly, even if the peripheral was previously configured by another Application that changed the registers' default values. Thereby this General requirement "AR_PN0034_FR_0025" is not considered for above mentioned registers. Expected Behavior: The general MRS requirement "AR_PN0034_FR_0025" should be taken care for all above mentioned registers and DMA related registers and all should be initialized in Adc_init().	BUG	OPEN ISSUE
28151	ADC	AUTOSAR requirement ADC077 is not taken care while implementing Adc_Init().	Problem Description: As per this requirement, [ADC077] The function Adc_Init shall disable the notifications and hardware trigger capability (if statically configured as active). Configured HW triggers are not disabled in Adc_Init(). If we consider General MRS requirement "AR_PN0034_FR_0025", it needs to be corrected accordingly. As per this requirement:- "The <MSN>_Init API shall ensure that the related peripheral is running correctly, even if the peripheral was previously configured by another Application that changed the registers' default values. " Expected Behavior: Configured HW triggers to be disabled in Adc_Init() Actual Behavior: Configured HW triggers are not disabled in Adc_Init().	BUG	OPEN ISSUE

28158	ADC	The Cautions mentioned in Device manual are not implemented.	Problem Description: For example As per Caution[1]: section 30.3.9 To prevent malfunctions, make ADCDnADCR1 settings after making or confirming the following settings. (1) HLDTE of T&H group A and B are 0. (2) ADSTARTE of all scan groups are 0 and TRGMD of all scan groups are 0H. (3) SGACT of all scan groups are 0 (before scan groups are started).] Before setting value to ADCDnADCR1 (ucADCXnADCR1) Register in Adc_Private_ADCD_ADCB.c file in Adc_HwInit() API, we are not setting the bits or values of these register bits mentioned in Caution are unknown to the driver. So based on General MRS requirement "AR_PN0034_FR_0025" this is to be implemented. Same type of issues are applicable for following registers also. Please check all applicable Cautions. 1. ADCDnADCR2 (ucADCXnADCR2) 2. ADCDnSFTCR (ucADCXnSFTCR) 3. ADCDnTDCR (ucADCXnTDCR) 4. ADCDnODCR (ucADCXnODCR) 5. ADCDnTHACR (ucADCXnTHACR) 6. ADCDnTHER (ucADCXnTHER) 7. ADCDnTHGSR(usADCXnTHGSR) 8. ADCDnSGVCSPx (ucADCXnSGVCSPx) 9. ADOPDIGn (ulADOPDIGn) Expected Behavior: none	BUG	OPEN ISSUE
28165	ADC	DEM error is not reported for Overwrite Error in error ISR Adc_ErrIsr()	Problem description: In the API Adc_ErrIsr(), ucADCXnOWER register is only used to calculate Physical channel ID. Understanding is that DEM error of Overwrite Error also needs to be reported. Also add similar DEM error report to other errors like Parity Error, Upper / lower limit error and ID Error. Expected Behavior: DEM error needs to be reported. Actual Behavior: DEm error is not reported.	FEATURE	OPEN ISSUE

28179	ADC	Device manual CAUTION is not considered for implementation in Adc_HwInit() and Adc_HwDeInit()	Problem Description: CAUTION mentioned in section 7.10.2.13 of R01UH0436EJ0070 Rev.0.70, is not considered for implementation . As per device manual Caution "DTFR.REQSEL can be changed while DTFR.REQEN is 0." But in current implementation both bits REQSEL and REQEN of DTFRn Register are updating simultaneously as mentioned in below code snippet. In Adc_HwInit(): LpDmaRegisters->ulDTFRn = LpSGmDmaConfig->ulDmaDtfrRegValue; ulDmaDtfrRegValue is generated value contain value of both bits REQSEL and REQEN of DTFRn Register. In Adc_HwDeInit(): LpDmaRegisters->ulDTFRn = ADC_DOUBLE_WORD_ZERO; Understanding is that before changing the value of DTFR.REQSEL, needs to clear bit DTFR.REQEN. Expected Behavior: Please update the design as below. In Adc_HwInit() 1. Reset: DTFRRQn register (0x00) 2. Clear bits LpDmaRegisters->ulDCENn = ADC_DMA_DTE_DISABLE; 3. LpDmaRegisters->ulDTFRn = LpSGmDmaConfig->ulDmaDtfrRegValue; For Adc_HwDeInit() Make similiar changes Actual Behavior: In Adc_HwInit():	BUG	OPEN ISSUE
28184	ADC	Clearing of ID Error is not correct.	Problem Description : Clearing of Error flags are not correct, In current implementation ADCDnIER.IDE error flag for ID Error is not clearing. LpAdcRegisters->ucADCXnECR = ADC_CHKCLR_ERROR_FLAG; of Adc_Private_ADCD_ADCB.c file in Adc_ErrIsr() Private API is used to clear all the error flag. As per device manual R01UH0436EJ0070 Rev.0.70 Section 30.3.29, register ADCDnECR (Error Clear Register) last 4 digits should be set to clear all error flag. In Adc_PBTypes_ADCD_ADCB.h file ADC_CHKCLR_ERROR_FLAG value of this macro is 0x0E. So last ADCDnIER.IDE bit is not clear. Suggested Solution : Adc_PBTypes_ADCD_ADCB.h file ADC_CHKCLR_ERROR_FLAG value of this macro should be 0x0F to clear all the flag. Actual Behavior : ID error is not getting clear when all the error flags are made clear. Expected Behavior : ID error should also getting clear when all the error flags made clear.	BUG	OPEN ISSUE

28187	ADC	Implementation of MRS requirement AR_PN0076_FR_0087 and AR_PN0076_FR_0089 is not proper.	problem Description : 1.MRS requirement "AR_PN0076_FR_0087" is needs to be take care in Tool code, But in current implementation its not taken care. As per Device manual R01UH0436EJ0070 Rev.0.70 section 30.3.28, number of ADCDnULLMTBRx register is 3 (x = 0, 1, 2). In current implementation tool code will not give any error even if we configure channel having limit check more than 2, when value of parameter "AdcPriorityImplementation" = ADC_PRIORITY_NONE. 2.MRS requirement "AR_PN0076_FR_0089" is needs to be take care in Tool code, But in current implementation its not taken care. As per Device manual R01UH0436EJ0070 Rev.0.70 section 30.3.28, number of ADCDnULLMTBRx register is 3 (x = 0, 1, 2). In current implementation tool code will not give any error even if we configure channel having limit check more than 2, when value of parameter "AdcPriorityImplementation" = ADC_PRIORITY_HW_SW or ADC_PRIORITY_HW. Actual Behavior : For both case generation tool code is not generating validation error, if limit check enabled for more than 3 channel Group with different Limit check setting. Understating is that, it will not work properly when these groups are queued. Expected Behavior : For both case proper validation needs to add in Tool code. Required MO suggestion on same.	BUG	OPEN ISSUE
28188	ADC	Register "ADCDnECR" (ucADCXnECR) is not handled properly in Adc_DeInit and Adc_Init() API's.	Problem description: In private API Adc_Init() register "ADCDnECR" (ucADCXnECR) is not implemented. and also in private API Adc_DeInit() register "ADCDnECR" (ucADCXnECR) is upated with zero as mentioned below : LpAdcRegisters->ucADCXnECR = ADC_ZERO; Understanding is that to clear error flags ADCDnOWER.OWE, ADCDnPER.PE and ADCDnIER.IDE, we need to set (write 1) to respective bits of the "ADCDnECR" (ucADCXnECR) register in Adc_DeInit() and Adc_Init() API. We can also consider general MRS requirement "AR_PN0034_FR_0025", As per this requirement, The Adc_Init API shall ensure that the related peripheral is running correctly, even if the peripheral was previously configured by another Application that changed the registers' default values. and autosar requirement ADC110 : The function Adc_DeInit shall return all ADC HW Units to a state comparable to their power on reset state. Values of registers which are not writeable are excluded. It’s the responsibility of the hardware design that this state does not lead to undefined activities in the µC. Expected Behavior : Register ADCDnECR" (ucADCXnECR) should be updated correctly in Adc_Init and Adc_DeInit APIs as per requirements. Actual Behavior : Register ADCDnECR" (ucADCXnECR) is not handled as per the requirements.	BUG	OPEN ISSUE

28214	ADC	AUTOSAR Requirement ADC091 and ADC277 is not taken care.	1. as per ADC091 requirement : "The ADC module's configuration shall be such that an ADC Channel group contains at least one ADC Channel" In current implementation AUTOSAR Requirement ADC091 is not taken care, in generation tool. As per current implementation, tool code is not giving any proper error even if no channel is configured under a ADC Group and tool code will crash by giving error "ERR123001", which is not correct. 2. As per ADC277 Requirement, "The ADC module's configuration shall be such that all channels contained in one ADC Channel group shall belong to the same ADC HW Unit." Its found that this SWS requirement is not mapped properly in TSDD, Traceability and TSTP, that's needs to fix accordingly. Most of the requirement are not tracked properly in Traceability sheet. Expected Behavior : point 1. Tool code should give a error message stating no channel is configured for particular ADC group. point 2. Update TSDD, TSTP, Traceability Actual Behavior : point 1. Tool code crash if no channel is configured for a ADC group. point 2. Requirements are not tracked properly in Traceability sheet.	BUG	OPEN ISSUE
26442	Can	Transmission History List issues	1) CanIf (CanIf_TxConfirmation) is being called while looping by the Can_TxConfirmationProcessing function. Any processing isn't being done by hardware, so I'm thinking the time-out isn't being confirmed. So don't we have to check the time out handling here? 2) It's written on "CLEARING OF ALL TRANSMIT MESSAGE BUFFERS" and a comment in the Can_StartMode function, but how is a transmission history buffer initialized? RSCAN0THLACCm and RSCAN0TXQPCTRm resister weren't being read, so I didn't understand how to be initialized.	BUG	OPEN ISSUE
26903	Can	PBcfg file Generation operation terminated due to Illegal division by zero	Problem description: Generation terminated due to Illegal division by zero at /PerlApp/BswConfigValidate.pm line 561. can_X1X.exe Can.arxml Sample_Application_F1x.trxml R403_can_F1x_BSWMDT.arxml EcuM.arxml Mcu.arxml Os.arxml Dem.arxml INF000001: Tool Version: 1.2.3 INF000002: Command line arguments: Can_X1x.exe Can.arxml Sample_Application_F1x.trxml R403_can_F1x_BSWMDT.arxml EcuM.arxml Mcu.arxml Os.arxml Dem.arxml Illegal division by zero at /PerlApp/BswConfigValidate.pm line 561. Expected behavior: PBcfg file Generation operation should not be terminated. If any error occurred Generator tool should throw out error with ambiguous error message. Actual behavior: PBcfg file Generation operation is terminated due to Illegal division.If any error occurred Generator tool is throwing out error with unambiguous error message.	BUG	OPEN ISSUE

27020	Can	Walking 0 pattern is not implemented in Can_RamTst_WalkPath_Algorithm() API	Problem description: As per Renesas requirement AR_PN0069_FR_0023,the RAM is checked by using data patterns (checker pattern, walking-0 and walking-1 pattern). But in the current implementation walking-0 pattern is not implemented. This shall be implemented as an enhancement, but it is not a bug since Walking "0"'s pattern and Walking "1"'s pattern is similar. Expected behaviour: As per requirement walking-0 pattern shall be implemented. Actual behaviour: In the current code, walking-0 pattern is not implemented.	BUG	OPEN ISSUE
27183	Can	Can_SelfTestChannel API executes in modes other than STOPPED	Problem description: If the controller is not in STOPPED state, the function 'Can_SelfTestChannel' shall be aborted and returns E_NOT_OK. But as per the current implementation, this check is not provided and the code try to set the operation mode as Halt mode. Expected behavior: The function shall be aborted and returns E_NOT_OK, if the controller is not in the STOPPED state. Actual behavior: Please see the problem description.	BUG	OPEN ISSUE
27647	Can	Transmission occurs even if the return of Can_write() API is CAN_BUSY	Problem description: If cancellation is enabled, cancellation has to be initiated for the lower priority ID/Identical ID (if identical Id cancellation is also enabled) request when the write request came with the higher priority ID / identical ID for the same HTH. The TX request for the new L-PDU shall be repeated by the CanIf module, inside the notification function CanIf_CancelTxConfirmation - requirement [CAN288]. If cancellation is disabled, the new Can_Write() request for the same HTH shall not be accepted and returned with CAN_BUSY. The first write request shall be transmitted which was in pending state. The same information is covered with the requirements [CAN213], [CAN214], [CAN215] and [CAN434] Expected Behaviour: 1. The transmission shall not be there for the Can_Write() request when its return value is CAN_BUSY. 2. The cancellation of the pending transmission has to happen properly, when transmission cancellation is enabled. Actual Behaviour: In different scenarios the transmission of the frame happens even after returning the reply as CAN_BUSY for the Can_Write() API call. EX: 1. When the cancellation is OFF (In polling mode), however the return value of the Can_Write() request for the same HTH is CAN_BUSY, the transmission of the frame is observed on the CANAlyzer and also Tx confirmation is received. 2. When Identical ID cancellation is ON, however the return for the Can_Write() request for the same HTH with identical ID is CAN_BUSY as expected and cancel confirmation is received, the transmission is happening without re-requested as stated in requirement [CAN288]. Also complete frame data is '0' and Tx confirmation is received as well. 3. When cancellation is ON, however the return for the Can_Write() request for the same HTH with higher priority ID is CAN_BUSY as expected and cancel confirmation is received, the transmission is happening without re-requested as stated in requirement [CAN288] and also Tx confirmation is received .	BUG	OPEN ISSUE

27649	Can	Hth Cancellation is not notifed correctly to the upper layer (CanIf) in case of multiple Hth to cancel	Problem description:
			There is no while loop, but unnecessarily LucArrPosition incremented and wrong comment provided.
			uint8_least LucArrPosition;
			--code----
			if (LblTxCancelFlag == CAN_TRUE)
			{
			/* Set the BasicCAN HTH count to maximum to exit the while loop */
			LucArrPosition = LpPBController->ucNoOfBasicCanHth;
			/* Set the TX Cancellation Status flag of the HTH */
			Can_RSCAN_GaaTxCancelStsFlgs[(LucArrPosition >> CAN_THREE)] =
			(Can_RSCAN_GaaTxCancelStsFlgs[(LucArrPosition >> CAN_THREE)]) |
			((uint8)(CAN_ONE << (LucCount % CAN_EIGHT)));
			/* Increment the array position to point to next
			* BasicCAN HTH of the controller */
			LucArrPosition++;
			}
			else
			{
			/* No action required */
			}
			
			Expected behavior:
			None
			
			Actual behavior:
			None
				BUG	OPEN
					ISSUE

27650	Can	Multiple DET error reporting from one API not compliant with AUTOSAR requirement	Description: According to CAN091 from AUTOSAR 4.0.3 CAN SWS - a function that reports a development error shall return immediately after. In many CAN driver APIs multiple errors are checked and reported, before the function returns. Examples: * Can_Init() may report CAN_E_TRANSITION, CAN_E_PARAM_POINTER before it returns * Can_Can_InitController() may report e.g. CAN_E_UNINIT and CAN_E_PARAM_POINTER before it returns) Expected behaviour: Every CAN driver API should return immediately with not action after a development error is detected is reported Actual behaviour: In some case in CAN APIs multiple development errors can be reported, like for Can_Init(), Can_InitController()	BUG	OPEN ISSUE
27654	Can	DEM version Check is missing	Problem Description: As per the requirement [CAN111](BSW004), the AUTOSAR major and minor release version needs to be checked for all the external modules. But the version check for the DEM module is missing. Expected Behaviour: The compilation error shall be reported when the DEM module with different AUTOSAR release version is integrated. Actual behaviour: The compilation is happening successfully even when the DEM module with different AUTOSAR release version is integrated.	BUG	OPEN ISSUE
27742	Can	The MCAL CAN driver shall clear the corresponding WUF flag in the ISR (AR_PN0069_FR_0025)	Problem Description: The MCAL CAN driver will not clear the corresponding WUF flag in the ISR if the precompile configuration parameter CanWakeUpFactorClearIsr is set to TRUE. Default value should be FALSE. PDF is reviewed and found that the parameter CanWakeUpFactorClearIsr is not implemented. Expected Behavior: AR_PN0069_FR_0025 shall be implemented. Actual Behavior: AR_PN0069_FR_0025 is not implemented properly	BUG	OPEN ISSUE

27874	Can	Autosare requirment CAN065_Conf is not taken care	Problem Description: As per Autosare SWS CAN065_Conf, CanIdType shall support ID's of type STANDARD, MIXED and EXTENDED. In the current implementation only STANDARD and EXTENDED types are taken care. Expected Behavior: MIXED ID type shall also be supported by the PDF. Actual Behavior: NA	BUG	OPEN ISSUE
27880	Can	Additional API to cancel Tx is not available for CanIf / Upper layer.	Problem Description: The internal function “Can_TxCancel” is available as private API. As per the requirement description, the API shall be available as public API such that CanIf AUTOSAR module can have corresponding call to this API in order to use it. Expected Behavior: The internal function “Can_TxCancel” shall be available as additional API (public) Actual Behavior: NA	BUG	OPEN ISSUE
27882	Can	CAN_E_DATALOST development error is not reported	Problem description: According to AUTOSAR 4.0.3 CAN SWS CAN395: "If the development error detection for the Can module is enabled, the Can module shall raise the error CAN_E_DATALOST in case of “overwrite” or “overrun” event detection." This DET error CAN_E_DATALOST is declared in Can.h but it is not used in the code. Expected behaviour: Development error CAN_E_DATALOST should be reported if overwrite or overrun events are detected in the reception buffers. Actual behaviour: CAN_E_DATALOST development error reporting is nowhere encountered in CAN driver code	BUG	OPEN ISSUE
27934	Can	Unexpected Behavior of in Can wake up	Problem description: CAN wake up shows unexpected behavior. Wake up ISR is getting triggered, even if wake up is not initiated. Expected Bahavior: Wake up shall be triggered only on occurance of a wake up event Actual Behavior: Undefined behavior of CAN wake up	BUG	OPEN ISSUE
27945	Can	Can_Write request returns CAN_OK when HTH is busy	Problem Description: Can_Write request returns CAN_OK when HTH is busy to process another transmit request. Can transmission shall only be initiated after getting tx confirmation on the last transmitted message. Expected Behavior: Can_Write request shall return CAN_BUSY when HTH is busy to process another transmit request Actual Behavior: NA	BUG	OPEN ISSUE
27965	Can	Can module is malfunctioning on HW Tx Cancellation.	Problem description: Can module is malfunctioning when HW Tx Cancellation support is enabled and a Transmit abort request was not successful (the CAN frame was transmitted). It this particular case, once the Tx Cancellation is initiated inside the Can module source code the global transmit cancel flag “Can_GblTxCancelIntFlg” is set to CAN_TRUE. Later this flag “Can_GblTxCancelIntFlg” is cleared inside the source code invoked when the "INTCnWUP" interrupt is serviced. But in the current use case, since the Transmit abort request was not successful, the "INTCnWUP" interruupt is not activated. Instead, the "INTCnTRX" interrupt is activated (frame successfully transmitted from message buffer m), but the source code invoked when "INTCnTRX" interrupt is serviced (ex. CAN_CONTROLLER0_TX_ISR) does not clear the “Can_GblTxCancelIntFlg” flag. The flag remains set and this causes subsequent calls to "Can_Write()" to fail and return "CAN_BUSY" result. This renders the Can module incapable to transmit CAN frames any more (until being re-initialized). Actual behavior: N/A Expected behavior: N/A	BUG	OPEN ISSUE
28070	Can	[X1x][CAN] Can_RamTst_WalkPath_Algorithm is not called from Can_RAMTest API	Problem Description: While updating ECODE for merging the changes done as part of F1H E4.00.01 release to trunk, in Can_RAMTest API call of 'Can_RamTst_WalkPath_Algorithm' was replaced by 'Can_RamTest_Checker_Algorithm'. So now instead of calling Can_RamTst_WalkPath_Algorithm, Can_RamTest_Checker_Algorithm is called second time. Work Around: Repeated call of 'Can_RamTest_Checker_Algorithm' has to be replaced with 'Can_RamTst_WalkPath_Algorithm'. Expected behavior: Can_RamTst_WalkPath_Algorithm should be called from Can_RAMTest API. Actual behavior: Can_RamTst_WalkPath_Algorithm is not called from Can_RAMTest API.	BUG	OPEN ISSUE

28503	Can	CAN_E_DATALOST development error is not reported	Problem description: According to AUTOSAR 4.0.3 CAN SWS CAN395: "If the development error detection for the Can module is enabled, the Can module shall raise the error CAN_E_DATALOST in case of “overwrite” or “overrun” event detection." This DET error CAN_E_DATALOST is declared in Can.h but it is not used in the code. Expected behaviour: Development error CAN_E_DATALOST should be reported if overwrite or overrun events are detected in the reception buffers. Actual behaviour: CAN_E_DATALOST development error reporting is nowhere encountered in CAN driver code	BUG	OPEN ISSUE
28505	Can	Multiple polling period for Tx and Rx (valid for R3.2and R4.x) is not supported	Problem Description: This is a task to verify the possibility to configure and handle multiple transmission or reception in polling mode based on SWS R3.2 ID: CAN356,CAN436 (Rx) and CAN358, CAN345 (Tx). The solution to this issue is to update the CAN code generator to count how many instances of the parameter CanMainFunctionWritePeriod and CanMainFunctionReadPeriod are configured and based on this to generate the opportune number of macro #defines Can_MainFunction_Write_0 #define Can_MainFunction_Read_0 for the 1st instance, #define Can_MainFunction_Write_1 #define Can_MainFunction_Read_1 for the 2nd instance The generated code should look like this: /*Polling Period 0 for Write*/ #define Can_MainFunction_Write_0() Can_MainFunction_Write() /*Polling Period 1 for Write*/ #define Can_MainFunction_Write_1() Can_MainFunction_Write() ...etc. Expected behaviour: N/A Actual Behaviour:	BUG	OPEN ISSUE
28541	Can	Restrict the baudrate to only a limited range is wrong.	Restrict the baudrate to only a limited range is wrong. There are other buadrate like 50K or 25K that can be realized. Actually the generator impelementation is to occur error, if the value of the parameter CanControllerBaudRate is other than 33, 83,100,125,250,500 or 1000Kbps for the particular controller. Actual Behaviour: Error occurs, if the value of the parameter CanControllerBaudRate is other than 33,83,100,125,250,500 or 1000Kbps for the particular controller. Expected Behaviour:Error should not occur, if the value of the parameter CanControllerBaudRate is other buadrate like 50K or 25K.	BUG	OPEN ISSUE

28601	Can	Correct typo in error ERR080034.	Problem Description: Correct typo in error ERR080034 in BswConfigValidate.pm Expected behavior: The description of the ERR shall be "The calculated 'Time Quanta' should be in the range of <8- 25>" Actual Behavior: The description of the error is currently The calculated 'DBT (Data Bit Time)' should be in the range of <8- 25>.	BUG	OPEN ISSUE

24131	DIO	Initialization Check is not performed for Dio_MaskedWritePort() API In the DIO driver.	Problem Description:
			Dio_MaskedWritePort() API is not checked whether DIO Is initialized or not.
			
			Expected behavior:
			FUNC(void, DIO_PUBLIC_CODE) Dio_MaskedWritePort
			(Dio_PortType PortId,
			Dio_PortLevelType Level,
			Dio_PortLevelType Mask)
			{
			------variable declaraiton ----------
			/* Check whether DIO_DEV_ERROR_DETECT is enabled */
			#if (DIO_DEV_ERROR_DETECT == STD_ON)
			/*START of DIO_AR_VERSION */
			#if (DIO_AR_VERSION == DIO_AR_HIGHER_VERSION)
			if (Dio_GblDriverStatus == DIO_UNINITIALIZED)
			{
			/* Report Error to DET */
			(void)Det_ReportError(DIO_MODULE_ID, DIO_INSTANCE_ID,
			DIO_MASKED_WRITE_PORT_SID, DIO_E_UNINIT);
			LenDetErrFlag = E_OK;
			}
			else
			{
			/* No action required */
			}
			#endif
				BUG	OPEN ISSUE

26249	DIO	The channel group not validated correctly in Dio_ReadChannelGroup/Dio_WriteChannel Group APIs	Problem Description: Functional argument pointer 'ChannelGroupIdPtr' is checked only against NULL_PTR in Dio_ReadChannelGroup/Dio_WriteChannelGroup APIs and it is not validated properly to report DIO_E_PARAM_INVALID_GROUP. Currently, if wrong or out-of-bound address is passed (Ex: If Dio_GstChannelGroupData[] size is 4 i.e. 0-3 and if &Dio_GstChannelGroupData[4] is passed by mistake), the NULL_PTR condition check is unable to catch this and the API proceeds for further processing instead of reporting DET DIO_E_PARAM_INVALID_GROUP. While doing further processing the behaviour might be un-predictable. Expected Behavior: The channel group needs to be validated to report DET and it shall not do any further processing detecting DET. Actual Behavior: The channel group is not validated to report DET and it shall do further processing even during development error condition.	BUG	OPEN ISSUE

26251	DIO	Global variable (Dio_GusNoOfChannelGroups) and config structure member (usNoofChannelGroups) names are misleading	Problem Description: The global variable 'Dio_GusNoOfChannelGroups' which is initialised with the value of the config structure member 'usNoofChannelGroups' is used as offset for accessing the channel group structure when multi config set comes into picture. But the name suggests that it holds the value of "number of channel groups configured" which misleading. E.g.: If 8 channel groups are configured with two config sets (each), the handles will be generated as, #define DioConf_DioChannelGroup_DioChannelGroup1 (&Dio_GstChannelGroupData[0]) #define DioConf_DioChannelGroup_DioChannelGroup2 (&Dio_GstChannelGroupData[1]) #define DioConf_DioChannelGroup_DioChannelGroup3 (&Dio_GstChannelGroupData[2]) #define DioConf_DioChannelGroup_DioChannelGroup4 (&Dio_GstChannelGroupData[3]) #define DioConf_DioChannelGroup_DioChannelGroup5 (&Dio_GstChannelGroupData[4]) #define DioConf_DioChannelGroup_DioChannelGroup6 (&Dio_GstChannelGroupData[5]) #define DioConf_DioChannelGroup_DioChannelGroup7 (&Dio_GstChannelGroupData[6]) #define DioConf_DioChannelGroup_DioChannelGroup8 (&Dio_GstChannelGroupData[7]) It shall generate 8 handles pointing 8 channel group structures and the channel structures 9-16 which is applicable for second config set shall be accessed with the help of same handles with offset value in 'Dio_GusNoOfChannelGroups'. i.e. When config set 0 is initialised its value will be 0. When config set 1 is initialised its value will be 8 for the above case. Expected Behavior: The name of this global variable and respective config structure member has to be corrected with meaningful name which is near to it usage. Ex: 'Dio_GusChannelGroupsOffset' and 'usChannelGroupsOffset' respectively. Actual Behavior: Variable names are misleading	BUG	OPEN ISSUE
26572	DIO	Optimize the execution time of RSR register setting sequence	Problem description: Check the pin direction and prepare the value and write to psr register will take less time to execute when the pin direction check is failed(beginning check itself it will come out). As per current method Even the pin direction check is failed, Prepare the value to set to the register operation is performed. Expected behavior: 1.Check the pins direction.(Check the PMSR register.) 2.Prepare the value to set to the register. 3.Write the PSR register. Source : Please refer to the attached file.(Proposed amendments) Dio_WriteChannel 168-191 Dio_WriteChannelGroup 407-415 Dio_MaskedWritePort 584-591 Actual behavior: 1.Prepare the value to set to the register. 2.Check the pins direction.(Check the PMSR register.) 3.Write the PSR register. Function : Dio_WriteChannel, Dio_WriteChannelGroup, Dio_MaskedWritePort Line : Dio_WriteChannel 931-956 Dio_WriteChannelGroup 1573-1583 Dio_MaskedWritePort 1714-1721	BUG	OPEN ISSUE

27729	DIO	Unreachable code present in Dio.c	In API Dio_ReadChannelGroup and API Dio_WriteChannelGroup, In the below code if the first condition got true, code will not go inside else part. If the first condition fails the second condition would also be false. So the code inside the second if block is unreachable that is dead code. The issue is found during UT.
			
			if (ChannelGroupIdPtr == NULL_PTR)
			{
			/* TRACE [R3, DIO140][R4, DIO140] */
			/* TRACE [R4, DIO178] */
			/* Report Error to DET */
			(void)Det_ReportError(DIO_MODULE_ID, DIO_INSTANCE_ID,
			DIO_WRITE_CHANNEL_GROUP_SID, DIO_E_PARAM_POINTER);
			LenDetErrFlag = E_OK;
			}
			else
			{
			/* Get the pointer to corresponding index in the
			array Dio_GstChannelGroupData */
			/* MISRA Violation: START Msg(4:0492)-6 */
			LpChannelGroupPtr = &ChannelGroupIdPtr[Dio_GusNoOfChannelGroups];
			/* END Msg(4:0492)-6 */
			if (NULL_PTR == LpChannelGroupPtr)
			{
			/* Report Error to DET */
			(void)Det_ReportError(DIO_MODULE_ID, DIO_INSTANCE_ID,
			DIO_WRITE_CHANNEL_GROUP_SID, DIO_E_PARAM_INVALID_GROUP);
			LenDetErrFlag = E_OK;
			}
			else
				BUG	OPEN ISSUE

26594	FLS	[F1L][FLS] Mismatch in memory mapping of Fls_MainFunction	Problem description:♂ There is a mismatch in the memory mapping of Fls_MainFunction. In Fls.h, Fls_MainFunction is mapped to FLS_START_SEC_PUBLIC_CODE:(See the code below) 254 : #define FLS_START_SEC_PUBLIC_CODE 255 : #include "MemMap.h" 280 : extern FUNC(void, FLS_PUBLIC_CODE) Fls_MainFunction(void); In Fls.c, Fls_MainFunction is mapped to FLS_START_SEC_SCHEDULER_CODE(See the code below) 1689 : define FLS_START_SEC_SCHEDULER_CODE 1690 : #include "MemMap.h" 1691 : 1692 : FUNC(void, FLS_PUBLIC_CODE)Fls_MainFunction(void) And in MemMap.h, both FLS_START_SEC_PUBLIC_CODE and FLS_START_SEC_SCHEDULER_CODE are defined as ".FLS_PUBLIC_CODE_RAM". Expected behaviour:♂ Memory mapping in both declaration and definition of function shall be unique. Actual behaviour: There is difference in the memory mapping of declaration and definition of function Fls_MainFunction.	BUG	OPEN ISSUE

26925	FLS	Length calculation for misaligned access fails	Description: Length calculation in Fls_Internal leads to invalid values. Actual behaviour: Resulting length is around 4 billion which causes other function to be in endless loop. Expected behaviour: Correct length calculation	BUG	OPEN ISSUE
27581	FLS	Fls_GVar structure is not initialised properly according to C89/C90 (ISO/IEC 9899:1990)	Problem description: The initialization of Fls_GVar in Fls_Ram.c is not according to C90 standard. A structure that contains pointers, variables and further structures is simply initialized with "0". This is possible in C99 (ISO/IEC 9899:1999, chapter 6.7.8.21), but MCAL shall be implemented according to C89/C90 (ISO/IEC 9899:1990). In Coding Guidelines EAAR-GL-0084.pdf, EAAR_PN0084_NR_0065 an example is given: /* Usage and initialization in a C file: */ MyModule_Vector_tst Vector1_st = { 0, 0, 0 }; Expected behaviour: Each element in the structure shall be initialised properly. Actual behaviour: Structure is initialised as VAR(Fls_GVarProperties, FLS_INIT_DATA) Fls_GVar = {FLS_ZERO};	BUG	OPEN ISSUE
27819	FLS	Fls_Resume cannot interrupt the FLS ISR(JOB END).(AR_PN0072_FR_0046)	Problem Description: As per current implementation in Fls_Resume() API, If FLS_DEV_ERROR_DETECT = STD_ON and FLS_TIMEOUT_MONITORING = STD_ON then after waiting for FLS_ISR_TIMEOUT_VALUE, It starts FLS Resume process without checking whether FLS ISR is serviced (Check whether Fls_GVar.Fls_MutexFlag == FLS_ZERO). In current implementation, It will also not report any DET if timeout occurred after waiting for FLS ISR is serviced. This issue exist in file Fls.c V1.3.6. As per requirement AR_PN0072_FR_0046:- <Fls_Suspend cannot interrupt the FLS ISR(JOB END). It means when FLS ISR(JOB END) has already entered critical section (protected by semaphore/mutex) the upcoming Fls_Suspend must wait until ISR(JOB END) exits critical section. until ISR(JOB END) exits critical section.> Understanding is that above mentioned point in MRS is not correct. As per our understanding correct one is as mentioned below. <Fls_Resume cannot interrupt the FLS ISR(JOB END). It means when FLS ISR(JOB END) has already entered critical section (protected by semaphore/mutex) the upcoming Fls_Resume must wait until ISR(JOB END) exits critical section. until ISR(JOB END) exits critical section.> Expected Behavior: MRS requirement AR_PN0072_FR_0046 needs to implement properly. Actual Behavior: MRS requirement AR_PN0072_FR_0046 is not implemented properly.	BUG	OPEN ISSUE

27829	FLS	Fls_Suspend cannot interrupt the FLS ISR(JOB END).(AR_PN0072_FR_0045)	Problem Description: As per current implementation in Fls_Suspend() API, If FLS_DEV_ERROR_DETECT = STD_ON and FLS_TIMEOUT_MONITORING = STD_ON then after waiting for FLS_ISR_TIMEOUT_VALUE, It starts FLS Suspend process without checking whether FLS ISR is serviced (Check whether Fls_GVar.Fls_MutexFlag == FLS_ZERO). In current implementation, It will also not report any DET if timeout occurred after waiting for FLS ISR is serviced. This issue exist in file Fls.c V1.3.6. As per requirement AR_PN0072_FR_0045:- Fls_Suspend cannot interrupt the FLS ISR(JOB END). It means when FLS ISR(JOB END) has already entered critical section (protected by semaphore/mutex) the upcoming Fls_Suspend must wait until ISR(JOB END) exits critical section. Expected Behavior: MRS requirement AR_PN0072_FR_0045 needs to implement properly. Actual Behavior: MRS requirement AR_PN0072_FR_0045 is not implemented properly.	BUG	OPEN ISSUE
27839	FLS	Fls_Cancel cannot interrupt the FLS ISR(JOB END).(AR_PN0072_FR_0011)	Problem Description: As per current implementation in Fls_Cancel() API if FLS_DEV_ERROR_DETECT = STD_ON and FLS_TIMEOUT_MONITORING = STD_ON then after waiting for FLS_ISR_TIMEOUT_VALUE, It starts FLS Cancel process without checking whether FLS ISR is serviced (Check whether Fls_GVar.Fls_MutexFlag == FLS_ZERO). In current implementation, It will also not report any DET if timeout occurred after waiting for FLS ISR is serviced. This issue exist in file Fls.c V1.3.6. As per requirement AR_PN0072_FR_0011:- Fls_Cancel cannot interrupt the FLS ISR(JOB END). It means when FLS ISR(JOB END) has already entered critical section (protected by semaphore/mutex) the upcoming Fls_Cancel must wait until ISR(JOB END) exits critical section. Expected Behavior: MRS requirement AR_PN0072_FR_0011 needs to implement properly. Actual Behavior: NA	BUG	OPEN ISSUE
27867	FLS	Pre compile switch required to partition FCL and FDL in source code (AR_PN0072_FR_0027)	Problem description: As per AR_PN0072_FR_0027, The source code of FLS module must be partitioned in FCL part and FDL part by using pre-compile switches. The Flash library files, in this case FCL/FDL files, shall be included in build process as per FLS module configuration for respective use-cases. Redundant library files shall be excluded from build process Expected behavior: The requirement AR_PN0072_FR_0027 shall be implemented Actual Behavior: NA	BUG	OPEN ISSUE
27890	FLS	Flash library status mapping	Problem description: Internal status of underlying libraries, ie. FCL and FDL shall be mapped to FLS driver status accordingly and shall lead to proper job processing result of FLS driver. In case of critical internal errors, notification must be given and user can decide to take necessary remedy. Expected behavior: MRS requirement AR_PN0072_FR_0042 needs to implement. Actual behavior: MRS requirement AR_PN0072_FR_0042 is not implemented properly.	BUG	OPEN ISSUE

28459	FLS	Incomplete linker directive file for FLS sample application	<B>Problem Description:</B> The linker directive file of FLS sample application does not contain the entries for copying the initial variable values from ROM to RAM during startup. This is typically done by e.g. <pre>.romdata ROM(.data)</pre> <B>Expected Behaviour:</B> Variables are initialized by GHS startup as required by C standard. <B>Actual Behaviour:</B> Variables are uninitialized, typically at 0 after power on, at any undefined value after reset. Application might show strange behaviour.	BUG	OPEN ISSUE

28514	FLS	Negative test cases required to verify the boundary check of FLS_CF_OFFSET_VALUE	In Fls.c,in section
			
			#if (FLS_FLASH_ACCESS == FLS_CODEFLASH_ACCESS)
			  /* Virtual address is mapped to physical address */
			  TargetAddress = TargetAddress - FLS_CF_OFFSET_VALUE;
			
			There shall be a check for TargetAddress against FLS_CF_OFFSET_VALUE before doing subtraction.
			
			Further, it requires negative test cases to verify the boundary check.
				BUG	OPEN
					ISSUE

28515	FLS	Fls_GulTimeout to be removed from functions Fls_CFProcessReadCommand and Fls_CFProcessCompareCommand	In Fls_Internal.c, Fls_GulTimeout is actually not used in functions Fls_CFProcessReadCommand and Fls_CFProcessCompareCommand. So it should be removed from above mentioned two subfunctions.	BUG	OPEN ISSUE
28516	FLS	Setting of variable job notification to True is actually not depending on data flash status or code flash status	In Fls_Internal.c, in API Fls_EndJobProcess(), 'LblJobNotification' variable is set to true irrespective of the condition check "if (R_FDL_OK == Fls_GstVar.GucDFStatus)". Similar is the case with the check "if (R_FCL_OK == Fls_GstVar.GucCFStatus)". So the redundant code can be merged in these cases.	BUG	OPEN ISSUE
28517	FLS	DEM error report should be added depending on the Erase/Write operations.	In Fls_Irq.c, Dem_ReportErrorStatus(FLS_E_ERASE_FAILED, DEM_EVENT_STATUS_FAILED) and Dem_ReportErrorStatus(FLS_E_WRITE_FAILED, DEM_EVENT_STATUS_FAILED) should be added depending on the Erase/Write operations.	BUG	OPEN ISSUE
28518	FLS	Tool shall throw error message when 'FlsUseInterrupt = ON' and the call back functions are not mapped	If interrupt is supported (FlsUseInterrupt = ON) and the call back functions are not mapped (NULL), the program will hang. The tool code shall be updated to throw error message for above mentioned user configuration.	BUG	OPEN ISSUE
28520	FLS	Default value of LenReturnValue shall be E_NOT_OK	In Fls.c, Default value of LenReturnValue shall be E_NOT_OK. In this way, FLS job request will be rejected in the first place when driver state is busy. This logic is not dependent on DET settings.	BUG	OPEN ISSUE
28521	FLS	The logic for updating Fls_GVar.Fls_GenState and Fls_GVar.Fls_GenJobResult shall be in line with the logic in Fls_EndJobProcess	In Fls_Irq.c, at end of job processing, the logic for updating Fls_GVar.Fls_GenState, Fls_GVar.Fls_GenJobResult and triggering JobEnd or JobError notifications shall be in line with the logic in Fls_EndJobProcess function in Fls_Internal.c. In the current implementation, Fls_Erase and Fls_Write support interrupt based job processing. And Fls_EndJobProcess is not required at end of interrupt based job processing.	BUG	OPEN ISSUE

28522	FLS	JobEnd and JobErrorNotification should depend on both (FLS_JOB_NOTIF_CONFIG == STD_ON) && (FLS_INTERRUPT_MODE == STD_ON)	In Fls_Irq.c, Fls_GpConfigPtr->pJobEndNotificationPointer() and Fls_GpConfigPtr->pJobErrorNotificationPointer() should be depending on both (FLS_JOB_NOTIF_CONFIG == STD_ON) && (FLS_INTERRUPT_MODE == STD_ON).
			
			Actual Behaviour :
			
			if (R_FDL_OK == Fls_GstVar.GucDFStatus)
			{
			/* Set the job Result to OK */
			Fls_GVar.Fls_GenJobResult = MEMIF_JOB_OK;
			/* If job ended with success and call the job end call back
			* function.
			*/
			Fls_GpConfigPtr->pJobEndNotificationPointer();
			}
			else
			{
			/* Set the job Result to Failed */
			Fls_GVar.Fls_GenJobResult = MEMIF_JOB_FAILED;
			/* If job ended with error and call the job error call back
			* function.
			*/
			Fls_GpConfigPtr->pJobErrorNotificationPointer();
			}
			
			Expected Behaviour :
			
			if (R_FDL_OK == Fls_GstVar.GucDFStatus)
				BUG	OPEN ISSUE

28523	FLS	Improvement in PDF of P1x	The following points should be taken care in the Parameter Definition File of P1x. a. The upper bound/maximum value of FlsFclRamAddress shall be 4276092927 (0xFEDF_FFFF) for P1x. b. The parameters used in each of the following containers should be re-ordered to give a better overview of parameters. 1. FlsDataFlash 2. FlsCodeFlash 3. FlsPublishedInformation c. Parameter "FlsDFTotalSize" in 'FlsDataFlash' container should be renamed as "FlsDataFlashSize". d. The statement in the description of following parameters shall be updated as mentioned. 1. FlsMaxWriteNormalMode : add into description - This parameter is not used for implementation. 2. FlsMaxEraseNormalMode : add into description - This parameter is not used for implementation. 3. FlsTotalSize : improve description - This parameter specifies the total amount of flash memory in bytes that is accessible by FLS driver. 4. FlsNumberOfSectors : add into description - This parameter setting shall be in line with FlsSectorStartaddress. 5. FlsFclRamAddress : add into description - This parameter is not used for implementation. 6. FlsDFTotalSize : improve description - This parameter indicates the physical total size of Data Flash memory.	BUG	OPEN ISSUE
28545	FLS	Safe exit from while-loop for R_FDL_Handler in Fls_Init shall be realized as per general requirement	In Fls.c, Safe exit from while-loop for R_FDL_Handler in Fls_Init shall be realized as per general requirement. Timeout monitoring can be considered here.	BUG	OPEN ISSUE
28546	FLS	DEM error should be reported for transient failures	FLS Initialization (Fls_Init) can fail during driving cycle of ECU, due to eg. operation voltage change, clock frequency change, etc. (transient failures). Such kind of fault must be notified and afterwards the upper layer can, for example, retry with init procedure until succeeds, or the system can be switched to a safety state if required. DEM error should be reported here.	BUG	OPEN ISSUE
28547	FLS	R_FDL_Handler() call in Fls_MainFunction and Fls_Init should be protected with critical section	In Fls.c, R_FDL_Handler() call in Fls_MainFunction and Fls_Init should be protected with critical section.	BUG	OPEN ISSUE

28587	FLS	Unexpected Interrupt pending bit is set in Fls_Write and Fls_Erase APIs	Problem description: In Fls_Write and Fls_Erase APIs, interrupt processing is enabled as follows: #if (FLS_INTERRUPT_MODE == STD_ON) /* Enable interrupt processing */ RH850_SV_MODE_IMR_AND(16, (Fls_GpConfigPtr->pFlEndImrAddress), (Fls_GpConfigPtr->usFlEndImrMask)); #endif At that point SW nearly always had the interrupt pending bit set, so a first unwanted interrupt occurs quite fast there. Pending interrupts are not properly handled in the code so that this will cause some confusion because of a pending interrupt from the previous operation. There is a chance of setting interrupt pending bit from the previous Read operation, which have inbuilt blank check. Expected behaviour: Before enabling interrupts, interrupt pending bit shall be cleared. Actual behaviour: Interrupt pending bit is set when interrupt processing is enabled in Fls_Write and Fls_Erase APIs	FEATURE	OPEN ISSUE
28507	FlsTst	As per Autosar requirement, FlsTst.h should include Std_Types.h directly.	According to Autosar requirement specification, the include of Std_Types.h should be done in FlsTst.h But in FlsTst, Std_Types.h is included through FlsTst_PBTypes.h and FlsTst_Types.h in the current implementation. The files can be found in the following svn path - /trunk/external/X1X/common_platform/modules/flstst/include Expected behaviour : FlsTst.h should include Std_Types.h directly. Actual behaviour : Std_Types.h is included through FlsTst_PBTypes.h and FlsTst_Types.h	BUG	OPEN ISSUE
28511	FlsTst	FlsTst_GenLastFgndResult and FlsTst_GenOverallBgndResult shall be declared as FLSTST_INIT_DATA	FlsTst_GenLastFgndResult and FlsTst_GenOverallBgndResult shall be declared as FLSTST_INIT_DATA, so as to match with memory section FLSTST_START_SEC_VAR_UNSPECIFIED. But in the current implementation FlsTst_GenLastFgndResult and FlsTst_GenOverallBgndResult are declared as FLSTST_NOINIT_DATA. This can be found in the path - \trunk\external\X1X\common_platform\modules\flstst\src\FlsTst_Ram.c Expected Behaviour : /* Variable to store the fgnd test result */ VAR(FlsTst_TestResultFgndType, FLSTST_INIT_DATA)FlsTst_GenLastFgndResult = FLSTST_NOT_TESTED; /* TRACE [R4, FlsTst154] */ /* Variable to store the overall Bgnd test result */ VAR(FlsTst_TestResultType, FLSTST_INIT_DATA)FlsTst_GenOverallBgndResult = FLSTST_RESULT_NOT_TESTED; Actual Behaviour : /* Variable to store the fgnd test result */ VAR(FlsTst_TestResultFgndType, FLSTST_NOINIT_DATA)FlsTst_GenLastFgndResult = FLSTST_NOT_TESTED; /* TRACE [R4, FlsTst154] */ /* Variable to store the overall Bgnd test result */ VAR(FlsTst_TestResultType, FLSTST_NOINIT_DATA)FlsTst_GenOverallBgndResult = FLSTST_RESULT_NOT_TESTED;	BUG	OPEN ISSUE

27528	Fr	[P1x][Fr] While doing QAC Static Analysis, Some of the Misra violations are not justified.	Problem Description: While running QAC Static Analysis for the Fr_59.c file and Fr_59_Internal.c, QAC Misra rules violations are occuring.Some of the violations are justified and some violations are not justified.(For example: The messages such as 4:1843, 4:1863, 4:2985 etc. are not justified.). Common code change is not in the scope of V4.00.04 P1x release. Expected Behaviour: NA Actual Behaviour: NA	BUG	OPEN ISSUE
27727	Fr	[P1x][V4.00.04][FR] The test cases related to Dem_ReportErrorStatus(FrDemCtrlTestResultRef, DEM_EVENT_STATUS_FAILED) - FR_ETC_06	Problem Description: The test cases related to Dem_ReportErrorStatus(FrDemCtrlTestResultRef, DEM_EVENT_STATUS_FAILED) - FR_ETC_065, Dem_ReportErrorStatus (FrIfDemFTSlotStatusRef, DEM_EVENT_STATUS_FAILED) - FR_ETC_066 and FR_ETC_067 are failing. FR_ETC_065: In the ESTS, in the Tested functionality, Dem_ReportErrorStatus(FR_E_ACCESS, DEM_EVENT_STATUS_FAILED) is tested.But in the expected test result, Dem_ReportErrorStatus(FrDemCtrlTestResultRef, DEM_EVENT_STATUS_FAILED) is checked for the APIs Fr_ReceiveRxLPdu, Fr_CheckTxLPduStatus, Fr_TransmitTxLPdu, Fr_CancelTxLPdu. The test cases FR_ETC_065 is not tested because the Dem_ReportErrorStatus (FrDemCtrlTestResultRef, DEM_EVENT_STATUS_FAILED) is not implemented in the following APIs Fr_ReceiveRxLPdu, Fr_CheckTxLPduStatus, Fr_TransmitTxLPdu, Fr_CancelTxLPdu. FR_ETC_066 and FR_ETC_067: In the ESTS, in the Tested functionality, Dem_ReportErrorStatus(FR_E_ACCESS, DEM_EVENT_STATUS_FAILED) is tested.But in the expected test result,Dem_ReportErrorStatus (FrIfDemFTSlotStatusRef, DEM_EVENT_STATUS_FAILED) is checked for the APIs Fr_TransmitTxLPdu, Fr_ReceiveRxLPdu. The test cases FR_ETC_066 and FR_ETC_067 are not tested because the Dem_ReportErrorStatus (FrIfDemFTSlotStatusRef, DEM_EVENT_STATUS_FAILED) is not implemented in the following APIs Fr_TransmitTxLPdu, Fr_ReceiveRxLPdu. Expected Behaviour: FR_ETC_065: As per the Autosar R4.03 FlexRay SWS requirement, if any hardware error occurs while running the APIs Fr_ReceiveRxLPdu[FR232], Fr_CheckTxLPduStatus[FR243] , Fr_TransmitTxLPdu[FR223], Fr_CancelTxLPdu[FR613], then it should call Dem_ReportErrorStatus (FrDemCtrlTestResultRef, DEM_EVENT_STATUS_FAILED) and return E_NOT_OK. FR_ETC_066 and FR_ETC_067: As per the Autosar R4.03 FlexRay SWS requirement, In the API Fr_ReceiveRxLPdu, [FR605]If the optional configuration parameter FrIfDemFTSlotStatusRef exists and a single slot status error bit (vSS!SyntaxError, vSS!ContentError, vSS!Bviolation) is set, then the slot status information shall be reported to DEM as Dem_ReportErrorStatus (FrIfDemFTSlotStatusRef, DEM_EVENT_STATUS_FAILED).	BUG	OPEN ISSUE
27728	Fr	[P1x][V4.00.04][FR] the functional testcases related to Transmit Queue and Receive Queue functionality are failing.	Problem Description: In ESTS the functional testcases related to Transmit Queue and Receive Queue functionality (FR_ETC_098, FR_ETC_099, FR_ETC_100, FR_ETC_101, FR_ETC_102, FR_ETC_103) are failing. Expected Behaviour: After transmitting the Data by invoking Fr_TransmitQueue_Table() it is returning E_OK as per the Expected Test result and when Fr_ReceiveQueue_Table() is invoked, it should E_OK and also the transmitted data should be received by the controller. Actual Behaviour: After transmitting the Data by invoking Fr_TransmitQueue_Table() it is returning E_OK as per the Expected Test result and when Fr_ReceiveQueue_Table() is invoked, it is returning E_OK as per the expected Test Result but the data is not received.	BUG	OPEN ISSUE
28147	Fr	[P1x][Fr][R4.0] Fr_User_Request_Output_Transfer and Fr_User_Request_Input_Transfer API's are not working.	Problem Description: Fr_User_Request_Output_Transfer and Fr_User_Request_Input_Transfer API's are returning E_OK but no transmit/receive functionality is happening. Expected Behaviour: Transmit/receive functionality should happen in this Fr_User_Request_Output_Transfer and Fr_User_Request_Input_Transfer API's . Actual Behaviour: Transmit/receive functionality is not happen in this Fr_User_Request_Output_Transfer and Fr_User_Request_Input_Transfer API's .	BUG	OPEN ISSUE

28326	Fr	[P1x][FR]Some Ecode lines are more than 80 characters and trailing spaces are present.	Problem Description: 1.More than 80 characters in following lines : Fr_59_Internal.c : @L1332 , L1352, 1405,1424,1519,1591,1620,1632,1774,1905,2101 Fr_59.c : @L1909,2561 Fr_59_Debug.h : @L83 Fr_59_GeneralTypes.h:@L277,286 Fr_59_Internal.h : @L71, @L97 Fr_59_Version.h: @L86 Fr_59.h : @L436, L70 2. Trim trailing space not done in following files: Fr_59.h, Fr_59_Ram.h, Fr_59_PBTypes.h, Fr_59_Internal.h, Fr_59.c, Fr_59_Ram.c, Fr_59_Internal.c Actual Behavior: NA Expected Behavior: NA	BUG	OPEN ISSUE
28474	Fr	[P1x][Fr][R4.0] Null pointer checking is not performing.	Problem Description: Null pointer checking is not performing in the following API's 1.Fr_59_ReceiveRxLPdu line: 3053 (*Fr_LPduStatusPtr = FR_59_NOT_RECEIVED;) line: 3058 (*Fr_LSduLengthPtr = FR_59_ZERO;) 2.Fr_59_CheckTxLPduStatus line: 8485 (*Fr_TxLPduStatusPtr = FR_59_NOT_TRANSMITTED;) Actual behaviour: When dereference a NULL pointer thereby raising a NullPointerException. It will cause the controller to reset. Expected behaviour: Null pointer checking should be performed in the API Fr_59_ReceiveRxLPdu, Fr_59_CheckTxLPduStatus	BUG	OPEN ISSUE

28397	GPT	Wake up disabled channels are giving DET after GPT_MODE_SLEEP to GPT_MODE_NORMAL mode transition.	Description ---------- While Running ATF test case "GPT_FTC_142" its observed that unexpected DET with ApiId = 0X2 => Service Id of Gpt_DeInit API. ErrorId = 0XB => DET code to report Timer is already running. is occurring when calling Gpt_DeInit(), here all channels are expected be in "stopped" state. We tested as mentioned below, in ATF configuration "ATF_cfg01". Test Scenario ---------- 1. Call Gpt_Init(&GPT_CONFIG_01) to initialize the driver with [GPT_CONFIG_01]. 2. Call Gpt_EnableNotification() for channel 0. 3. Call Gpt_EnableNotification() for channel 1. 4. Call Gpt_EnableNotification() for channel 3. 5. Call Gpt_EnableWakeup() for channel 0. 6. Call Gpt_DisableWakeup() for channel 1. 5. Call Gpt_DisableWakeup() for channel 3. 6. Call Gpt_StartTimer() for channel 0. 7. Call Gpt_StartTimer() for channel 1. 8. Call Gpt_StartTimer() for channel 3.	BUG	OPEN ISSUE
28405	GPT	Timer is starting automatically when call Gpt_EnableWakeup() in GPT_MODE_SLEEP mode.	While testing ATF test case "App_Gpt_Sample" its observed that notification is occurring when call Gpt_EnableWakeup() in GPT_MODE_SLEEP mode before starting the timer. We tested as mentioned below, in ATF configuration "ATF_cfg05". 1. Call Gpt_Init(&GPT_CONFIG_01) to initialize the driver with [GPT_CONFIG_01]. 2. Call Gpt_EnableNotification() for channel 0. 3. Start timer for channel 0. 4. Wait for 200 (ms). 5. call Gpt_GetTimeElapsed() for channel 0. 6. Check whether Time Elapsed > 0 , and it's obtained as expected. 7. Wait for 10 (ms). 8. call Gpt_GetTimeRemaining() for channel 0. 9. Check whether Time Remaining > 0 , and it's obtained as expected. 10. Wait for 5 Seconds. 11. Check whether notification obtained is > 0 , and it's obtained as expected. 12. call Gpt_DisableWakeup() for channel 0. 13. Set GPT mode to 'GPT_MODE_SLEEP' by calling "Gpt_SetMode(GPT_MODE_SLEEP)". 14. Wait for 1 Seconds. 15. Clear The Notification Count. 16. Wait for 2 Seconds. 17. Check whether notification obtained is = 0 , and it's obtained as expected. 18. Set GPT mode to 'GPT_MODE_NORMAL' by calling "Gpt_SetMode(GPT_MODE_NORMAL)". 19. Wait for 2 Seconds. 20. Check whether notification obtained is = 0 , and it's obtained as expected. 21. Set GPT mode to 'GPT_MODE_SLEEP' by calling "Gpt_SetMode(GPT_MODE_SLEEP)". 22. call Gpt_EnableWakeup() for channel 0. 23. Set GPT mode to 'GPT_MODE_NORMAL' by calling "Gpt_SetMode(GPT_MODE_NORMAL)".	BUG	OPEN ISSUE

25856	ICU	Obsolete code in Icu_HW_Init() function.	Problem description:
			At line 754 in Icu_LLDriver.c the following loop is present:
			for (LucCnt = ICU_MAX_TIMER_CHANNELS_CONFIGURED; LucCnt < ICU_MAX_CHANNEL;
			LucCnt++)
			{
			
			This "for" loop is used for external interrupts only and not for timer channels initialization.
			
			Inside the loop you have checks for timer channels, which will always fail, since the loop is only for channels configured with external interrupts.
			
			It seems that everything above the "switch" at line 823 in this loop is obsolete code.
			
			Expected behavior:
			N/A
			
			Actual behavior:
			N/A
				BUG	OPEN ISSUE

27622	ICU	Interrupts(IMR) are enabled All Channel After calling Icu_SetMode() Api form ICU_MODE_SLEEP to ICU_MODE_NORMAL.	Problem Description: As per Autosar4.0.3 ICU requirement says that ICU194 ICU_MODE_NORMAL: Normal operation, all used interrupts are enabled according to the notification requests. ICU_MODE_SLEEP: Reducedpower mode. In sleep mode only those notifications are available which are configured as wakeup capable. Current implementation is In Icu_SetMode(ICU_MODE_NORMAL) Api called after Icu_SetMode(ICU_MODE_SLEEP) Api. All interrupts are enabled with out considering Current notification status. Ver4.01.06 -- release Icu_LLDriver.c 1765: /* Enable Interrupt */ 1766: RH850_SV_MODE_IMR_AND(16, (LpImrIntpCntrlReg), (LpChannelConfig->usImrMaskValue)); Expected Behaviour: All used interrupts are enabled according to the notification requests Actual Behaviour: All interrupts are enabled with out considering notification status.	BUG	OPEN ISSUE
28585	ICU	Component User Manual - Unimplemented APIs and Not supported features	Problem Description: 1. The functionalities which are not supported by the hardware are present in the component user manual.Icu_CheckWakeup, Icu_DisableWakeup and Icu_EnableWakeup should be removed from user manual. 2. If a feature is not supported please mention "Not supported" in Table 5-1 Expected Behaviour: NA Actual Behaviour: Wakeup functionalities(Icu_CheckWakeup, Icu_DisableWakeup and Icu_EnableWakeup) which are not implemented are mentioned in the user manual, which misleads the user.	BUG	OPEN ISSUE

27490	MCU	McuResetReason could not be accessed By EcuMResetReason from McuPublishedInformation0/McuResetReasonConf0 container	Problem Description: The contents of McuPublishedInformation0/McuResetReasonConf0/McuResetReason could not be accessed from EcuMResetReason since the implementation methodology in McuPublishedInformation dont seem to be according to what AUTOSAR has prescribed Note: Why is the P1x McuResetReason implementation different from F1x (The McuResetReason implemented in different way for F1x and P1x devices) Expected Behavior: None Actual Behavior: None	BUG	OPEN ISSUE
28160	MCU	GetVersionInfo() API of each module shall also return “instanceID” as one of the parameter in Std_VersionInfoType	Problem Description: This ticket is created to track the pre-release review done on P1x MCU work products as part of V4.00.04 release. Requirement: AR_PN0034_FR_0017 Finding: As per the requirement GetVersionInfo() API of each module shall also return “instanceID” as one of the parameter in Std_VersionInfoType pointed by the output parameter versioninfo. In the current implementation only moduleid, and vendorid are returned. Instance id is not returned Expected Behavior: GetVersionInfo() API of each module shall return moduleid, vendorid and instanceID Actual Behavior: GetVersionInfo() API of each module shall return moduleid and vendorid	BUG	OPEN ISSUE
28170	MCU	Dummy read to register must be performed after writing to register.	Problem Description: This ticket is created to track the pre-release review done on P1x MCU work products as part of V4.00.04 release. Requirement: AR_PN0034_FR_0068 Finding: Synchronizing peripherals register write operation by dummy read. As per this requirement, Dummy read to register must be performed after writing to register. The requirement is not taken care in Mcu source code. Expected Behavior: NA Actual Behavior: NA	BUG	OPEN ISSUE

28202	MCU	APIs Mcu_GetResetReason() and Mcu_GetResetRawValue() shall return the same result in case they are called multiple times	Problem Description: This ticket is created to track the pre-release review done on P1x MCU work products as part of V4.00.04 release. Requirement: EAAR_PN0079_FR_0086 The APIs Mcu_GetResetReason() and Mcu_GetResetRawValue() shall return the same result in case they are called multiple times after a reset or a power on event. Finding: Add test case to test the requirement. Expected behaviour: NA Actual behaviour: NA	BUG	OPEN ISSUE
28382	MCU	McuResetReason added in Mcu schema cannot be referenced by EcuM	In Mcu Schema several reset reason was added in McuPublishedInformation container. According with ECUM128_Conf Mcu Reset Reason should be in McuResetReasonConf container so this can be referenced by EcuM module .	BUG	OPEN ISSUE
28421	MCU	In definition file Mandatory parameter's Lower and Upper multiplicity values are not taken care properly	Problem Description: In definition .arxml file Mandatory parameter's Lower and Upper multiplicity value should be one. parameters are McuLoopCount and McuPbusWaitCount. example: In Mcu_PBTypes.h MCU_PBUSWAITCOUNT is define with MCU_PBUSWAITCOUNT_VALUE and is used in Mcu.c file If McuPbusWaitCount value is not set, In Mcu_Cfg.h for MCU_PBUSWAITCOUNT_VALUE will not be generated any value: In PBcfg.h file /* Pbus Count Value for the MCU_PBUSWAITCOUNT */ #define MCU_PBUSWAITCOUNT_VALUE Expected Behavior: <!-- PARAMETER DEFINITION: McuPbusWaitCount --> <ECUC-INTEGER-PARAM-DEF UUID="ECUS:dbeafba1-bfaf-4ca0-8572-235f3c9b9b35"> <SHORT-NAME>McuPbusWaitCount</SHORT-NAME> <DESC> <L-2 L="EN">The parameter represents the PBus access wait time.The loop can be minimum 1 to maximum 65535</L-2> </DESC> <LOWER-MULTIPLICITY>1</LOWER-MULTIPLICITY> <UPPER-MULTIPLICITY>1</UPPER-MULTIPLICITY> Actual Behavior: <!-- PARAMETER DEFINITION: McuPbusWaitCount --> <ECUC-INTEGER-PARAM-DEF UUID="ECUS:dbeafba1-bfaf-4ca0-8572-235f3c9b9b35"> <SHORT-NAME>McuPbusWaitCount</SHORT-NAME> <DESC>	BUG	OPEN ISSUE
28444	MCU	Reset reson handling depending on POF.POF bit	What is the difference between "McuResetReasonConfPowerOnReset" and "McuResetReasonConfPowerOnFlagReset" ? The implementation seems to be wrong as POF.POF bit seems to be set in both cases mentioned, which means only McuResetReasonConfPowerOnFlagReset is reported and McuResetReasonConfPowerOnFlagReset is never reported.	BUG	OPEN ISSUE

28473	MCU	App_MCU_Device_Sample.h Which is not as per AR_PN0034_FR_0039.	Problem Description: App_MCU_Device_Sample.h Which is not as per AR_PN0034_FR_0039. Expected Behavior: NA Actual Behavior: NA	BUG	OPEN ISSUE
25132	Port	When changing port pin to a DIO mode, handling of PSRn(Pn) is different by API.	Problem Description: The port pin can be changed to a DIO mode at API Port_SetPinMode, Port_SetToDioMode and Port_SetPinDefaultMode.Among these the Port_SetToDioMode doesn't set to PSR register.Is this what Development Team intended? DIO output level change should be performed in DIO Driver and the user can also decide at the timing of change in the DIO output level.I'm thinking this specification is simplest and is without mistakes. Could you tell me why it's such specification? Expected Behavior : It is necessary to unify these specifications Actual Behavior : the Port_SetToDioMode doesn't set to PSR register.	BUG	OPEN ISSUE

26487	Port	When Port_SetPinDirection() change direction from o/p to o/p (refeshing) for JTAG pins, the o/p level will set to default state.	PortPinDirectionChangeable = True
			PortPinModeChangeable = True
			PortPinLevelValue = PORT_PIN_LEVEL_LOW
			PortPinInitialMode = DIO_SUPP_PFC_PMCSR
			PortPinDirection = PORT_PIN_OUT
			
			Test case:
			Port_Init(PortConfigSet0);
			while(1)
			{
			/* This API will initilize all the registers to the initial values */
			Port_Init(PortConfigSet0);
			/* Set Port Pin level of for JP0_6 to High. */
			JPSR0 = 0xFFFF0040;
			/* Refresh the pin. */
			Port_SetPinDirection (Port_PortGroupJtag00_PortPin60, PORT_PIN_OUT);
			}
			
			Expected result:
			JP0_6 remains High.
			
			Actual result:
			JP0_6 sets to Low.
				BUG	OPEN ISSUE

26852	Port	Local variables may remain not initialized in Port_SetToDioOrAltMode API.	Problem description: If we have a configuration with the following generated code /* Availability of numeric port groups */ #define PORT_NUM_PORT_GROUPS_AVAILABLE STD_OFF /* Availability of alphabetic port groups */ #define PORT_ALPHA_PORT_GROUPS_AVAILABLE STD_OFF /* Availability of jtag port groups */ #define PORT_JTAG_PORT_GROUPS_AVAILABLE STD_OFF then in Port_SetToDioOrAltMode() API the local variables LpFuncCtrlReg and LulBaseAddress will be used without being initialized. Expected result: N/A Actual result: N/A	BUG	OPEN ISSUE
27039	Port	Port Group 4 Pin 6 MUX appears to be mis-labeled in the Parameter defitnition file.	Problem description: Port Group 4 Pin 6 MUX appears to be mis-labeled in the Parameter definition file. It is labeled as TAUD202_ALT6_OUT, while according to the HW user manual "2.4.1.7 Port 4 (P4)" it is related to TAUD2O3. Expected behavior: Port Group 4 Pin 6 MUX must be labeled as TAUD2O3_ALT6_OUT. Actual behavior: Port Group 4 Pin 6 MUX is labeled as TAUD2O2_ALT6_OUT.	BUG	OPEN ISSUE
27079	Port	Port Generator throwing unwanted error	Problem description: If PortIpControl is enabled for e.g. CSI pins as recommended in description of PortIpControl, then error 124018 is raised. Pin names in description of PortIpControl do not match to available options in PortPinInitialMode. This issue is valid for 701011 device, but not for 701035. Expected behaviour: No error should occur. Actual behaviour: ERR124018 Error occurs that is in contradiction to description.	BUG	OPEN ISSUE
27676	Port	Fail to initialize the PFCAE registers correctly	Problem Description: The register initialization sequence in Port_InitConfig() api is not as mentioned in r01uh0436ej0070_rh850p1x.pdf(v0.70) at page 122 Section 2.3.4.6. Expected Behavior : PFCAE register along with PFC and PFCE should be initialized after initializing PINV register. Actual Behavior : The function control registers(PFC,PFCE,PFCAE) are initialized before PINV register initialization	BUG	OPEN ISSUE

28391	Port	PINVn register is not setting properly in API Port_SetPinDirection()	Problem description: Value updating to PINVn — Port Output Level Inversion Register write protection is not implemented as in device User Manual. Expected behavior: Needs to follow the write protection sequence mentioned in device User Manual. Actual behavior: Register write protection is not implemented properly.	BUG	OPEN ISSUE

28447	Port	PMSR register access is not correct	Problem description:
			PMSR register is accessing without checking whether PMSR register is present for that particular Port group.
			
			Expected behavior: Before accessing PMSR register check whether PMSR register is exist is required.
			/*Check for PMSR register availability */
			if (PORT_REG_NOTAVAILABLE != LpSetPinModeGroupStruct->ucPMSRRegIndex)
			{
			...
			}
			
			Actual behavior:
			This check is not present result in illegal memory access.
				BUG	OPEN ISSUE

28539	Port	Pre compiler Macro is surrounded code at wrong place in Port_FilterConfig()	Problem description: Pre compiler Macro "PORT_DNFA_REG_CONFIG" is surrounded code at wrong place in Port_FilterConfig(). #if ((PORT_DNFA_REG_CONFIG == STD_ON) || (PORT_FCLA_REG_CONFIG == STD_ON)) #define PORT_START_SEC_PRIVATE_CODE #include "MemMap.h" STATIC FUNC(void, PORT_PRIVATE_CODE) Port_FilterConfig(void) { /* Pointer to digital filter DNFA register data structure */ #if (PORT_DNFA_REG_CONFIG == STD_ON) P2CONST(volatile Port_DNFARegs, AUTOMATIC, PORT_CONFIG_DATA) LpDNFAReg; /* Pointer to Edge control EDC register data structure */ #if (PORT_EDGE_DETECT_CONTROL == STD_ON) P2CONST(volatile Port_EDCRegs, AUTOMATIC, PORT_CONFIG_DATA) LpEDCReg; #endif /* End of PORT_EDGE_DETECT_CONTROL == STD_ON */ -----code----- #endif /* End of PORT_DNFA_REG_CONFIG == STD_ON */ -----code---- Port_FilterConfig() API is enabled by PORT_DNFA_REG_CONFIG is STD_ON or PORT_FCLA_REG_CONFIG is STD_ON, In side variable declaration is done for PORT_DNFA_REG_CONFIG is STD_ON, When PORT_DNFA_REG_CONFIG is STD_OFF and PORT_FCLA_REG_CONFIG is STD_ON it will corrupted. Expected behavior:	BUG	OPEN ISSUE

25724	PWM	Det PWM_E_PARAM_CHANNEL is not reporting for Pwm_SetTriggerDelay().	Problem description: For the current PWM driver implementation we face the problem that it is not reporting Det PWM_E_PARAM_CHANNEL for Pwm_SetTriggerDelay() when configured for PWM TAU channel. Expected behavior: Det PWM_E_PARAM_CHANNEL should report for Pwm_SetTriggerDelay() when configured for PWM TAU channel. Actual behavior: DET is not occuring.	BUG	OPEN ISSUE
26874	PWM	Pwm_SelectChannelClk is starting the PWM diag channels configured for sync start	Problem Description: If you configure 2 channels 1 on TAU and one PWM diag in sync start mode. If after Pwm_SynchronousInit() API, Pwm_SelectChannelClk() is called, the pwm diag channel will start even before calling Pwm_SynchronousStart(). Expected behavior: Channels marked as synchronous shall start only after Pwm_SynchronousStart() API is called. Current behavior: Check the problem description	BUG	OPEN ISSUE
28292	PWM	[P1x][PWM] PWM notification is not handled properly	Problem Description: 1. PWM notification null pointer checking is not performing on Pwm_HW_Callback ISR 2. Notification will send for wrong PWM channels/channels which are not configured. Actual behaviour: With in Pwm_HW_Callback ISR, channel id is incrementing with in a for loop. Notification checking and sending is doing outside this for loop. After the execution of that for loop channel id will be, exact channel id + 1 + number of slave channels. So the notification will send for wrong channels or try to send notification for the channels which are not configured. Example: we have configured 7 channels out of which 3 are slave channels. and the interrupt is coming for 4th master channel. So at the end of the 'for' loop, channel id will be 8. This will cause out of array access and if the value of that memory location is one, it will try to send notification, which is not configured. This will cause the controller to reset. Expected behaviour: 1.PWM notification null pointer checking should be performed in Pwm_HW_Callback ISR 2.PWM notification checking should be with proper channel id.	BUG	OPEN ISSUE
25663	RamTst	[P1x][RAMTST][R4.0] The test result is not RAMTST_RESULT_UNDEFINED, if a March Test on this block is running.	RamTst_GetTestResultPerBlock() does not return RAMTST_RESULT_UNDEFINED, when March test on the specific block is running.	BUG	OPEN ISSUE
25664	RamTst	[P1x][RAMTST][R4.0] RamTst_FillPattern not getting updated in the RAM location when RamTstTestPolicy is RAMTEST_DESTRUCTIVE.	When the configuration parameter RamTstTestPolicy for a block is set to RAMTEST_DESTRUCTIVE, the test algorithm does not fill the tested cells after the test with the bit pattern defined for this block by parameter RamTst_FillPattern except for the test algorithm RAMTST_ABRAHAM_TEST_APP.	BUG	OPEN ISSUE

26482	RamTst	Dem event parameter name not generated correctly	Problem Description: If the short name of DemEventParameter in file Dem_RamTst.arxml is not appended with any number then the Dem event parameter name is not getting generated correctly. Expected Behaviour: DEM event parameters should generated correctly as follows #define RAMTST_E_RAM_FAILURE \ DemConf_DemEventParameter_DemEventParameter Actual Behaviour: DEM event parameters are generated as follows #define RAMTST_E_RAM_FAILURE \ DemConf_DemEventParameter_ This results in compilation issues as "the identifier "DemConf_DemEventParameter_" is undefined"	BUG	OPEN ISSUE
28604	RamTst	Issues in EUM.	Problem Description: This ticket is to report the defects found in EUM. 1.In Section 8 "Software Generation Tool" and "Driver Generation Tool" are used in parallel, which is misleading. [Driver Generation Tool] should be used here. 2.In Revision History SI. No. 4 "As part of P1x V4.00.04 activity following changes are made:" should be removed. 3.In Section 4.5 'X' is not marked for user mode for RamTst APIs, even with known limitations listed in Table 4-1. This is not in line with other MCAL modules.User mode is supported by RamTst_RunFullTest, RamTst_RunPartialTest, RamTst_MainFunction, but with precondition that the critical section should be disabled. Expected behavior N/A Actual behavior N/A	BUG	OPEN ISSUE
28605	RamTst	RamTst_Ram.c and RamTst_Ram.h files are missing.	Problem Description : Unlike other MCAL modules there are no dedicated files, ie. RamTst_Ram.c and RamTst_Ram.h, to address global variables. In other MCAL modules <MSN>_Ram.c and <MSN_Ram.h> used to address global variables. Actual Behavior : No dedicated file is exist to address global variables. Expected Behavior : To maintain consistent file structures across all the MCAL modules these files should be added to address global variables.	BUG	OPEN ISSUE

28607	RamTst	Autosar requirement RamTst033 is not implemented properly.	Problem Description : As per AUTOSAR requirement RamTst033 : "If the DET is enabled and the execution status of the RAM Test is not RAMTST_EXECUTION_RUNNING or RAMTST_EXECUTION_SUSPENDED, the function RamTst_Stop shall report the error value RAMTST_E_STATUS_FAILURE to the DET, and then immediately return." Actual Behavior : In the current implementation execution status is checked against STOPPED as below: else if (RAMTST_EXECUTION_STOPPED == RamTst_ExecutionStatus) in RamTst_Stop API. Expected Behavior : N/A	BUG	OPEN ISSUE
28608	RamTst	Autosar requirement RamTst003 is not implemented properly.	Problem Description : As per this requirement RamTst.h shall include Std_Types.h directly. In current implementation Std_Types.h is included via RamTst_Types.h. Actual Behavior : Std_Types.h is included RamTst_Types.h and RamTst_Types.h is included RamTst.h which is not correct as per Autosar requirement RamTst003. Expected Behavior : RamTst.h shall include Std_Types.h directly.	BUG	OPEN ISSUE
25109	SPI	SpiJobEndNotification functions are not generated correctly, when two or more Jobs have the same JobEndNotification function.	Problem description: SpiJobEndNotification functions are not generated correctly, when two or more Jobs have the same JobEndNotification function. Some JobEndNotifications functions are NULL after the generation, in spite they are not configured as NULL. There is no information or warning in the generator's manual, that this configuration would not be allowed. Expected behavior: If the following SpiJobEndNotifications are in one configuration: TswSpi_AsyncJobEndNotif TswSpi_AsyncJobEndNotif NULL NULL TswSpi_PrioCheckJob1EndNotif TswSpi_PrioCheckJob2EndNotif TswSpi_PrioCheckJob3EndNotif TswSpi_PrioCheckJob4EndNotif TswSpi_PrioCheckJob5EndNotif NULL NULL the same should be expected to be generated in Spi_BPcfg.c Actual behavior: Instead in Spi_BPcfg.c we have the following: NULL_PTR TswSpi_AsyncJobEndNotif	BUG	OPEN ISSUE

26389	SPI	Short name, File name and Path generated for error id ERR083058 is incorrect	Problem Description: ERR083058 message is generating as follows The reference path </AUTOSAR/EcucDefs/Dem0/DemConfigSet0/DemEventParamete> provided for the parameter 'SPI_E_HARDWARE_ERROR' in the container 'SpiDemEventParameterRefs', having shortname <HASH(0x31829ac){ShortName}> is incorrect. File Name: HASH(0x31829ac) {FileName} Path: HASH(0x31829ac){ShortName} Expected Behaviour: Correct Short name, File name and Path should be generated for error id ERR083058. Actual Behaviour: Short name, File name and Path generated for error id ERR083058 are wrong.	BUG	OPEN ISSUE
26420	SPI	Execution stuck in Spi_HWTransmitSyncJob function	Problem Description: When a sequence is transmitted synchronously, the execution hangs in Spi_HWTransmitSyncJob. Expected Behaviour: Sequence should be transmitted without hang. Actual Behaviour: Calling Spi_SyncTransmit API results hanging in Spi_HWTransmitSyncJob API .	BUG	OPEN ISSUE
26764	SPI	The SPI driver does not change its status in case of data consistency error occurs during sync transmission.	Problem description: In case of data consistency error flag (CSIHnDCE) is set during SPI sync transmission, the ongoing sequence is aborted. The problem is that after the ongoing sequence was aborted, the global variable Spi_GusHwStatus is not changed by the Spi_SyncTransmit API. This blocks all the next SPI communication. Expected result: In case of consistency error detection during SPI Sync transmission, the ongoing sequence must be cancelled. Actual result: After consistency error detection during SPI Sync transmission, whole further communication is blocked. A workaround is not to enable the CSIHnCTL1.CSIHnDCS bit, until this issue is not fixed.	BUG	OPEN ISSUE
27688	SPI	Illegal Memory access in Spi_Driver.c in API Spi_TransmitISR()	Problem description: If the if condition: if (SPI_FIFO_BUFFER_FULL != Spi_GucHWFifoBufferSts[SPI_FIFO_RX_INDEX])) fails, the pointer LpPBChannelConfig will not be initialised (since it is written in the if condition at Line 5618) This would lead to an illegal memory access (at Line:5707) where LpPBChannelConfig is used. Expected Behavior: The variable LpPBChannelConfig shall be initialized before used Actual behavior: If the if condition ((if (SPI_FIFO_BUFFER_FULL != Spi_GucHWFifoBufferSts[SPI_FIFO_RX_INDEX]))) fails, LpPBChannelConfig is not initialized.	BUG	OPEN ISSUE

27707	SPI	Illegal Memory access in Spi_Driver.c	Problem Description: In Spi_Driver.c, API Spi_TransmitISR(), the local variable LpJobConfig is initialized in the part of code as below.
			
			if (SPI_FIFO_BUFFER_UNINIT == Spi_GucHWFifoBufferSts[SPI_FIFO_RX_INDEX])
			{
			.....;
			.....;
			LpJobConfig = Spi_GpFirstJob + LddJobIndex;
			}
			
			If the condition SPI_FIFO_BUFFER_UNINIT is not equal to Spi_GucHWFifoBufferSts[SPI_FIFO_RX_INDEX]), the variable LpJobConfig will not be initialized. This could lead to illegal memory access since the variable is also used elsewhere. Eg: In the do while loop below the if loop mentioned in the description,
			
			#if (SPI_DMA_MODE_ENABLE == STD_ON)
			/* MISRA Violation: START Msg(4:2962)-18 */
			if ((SPI_INTERRUPT_MODE == Spi_GddAsyncMode) &&
			(SPI_INVALID_DMAUNIT == LpJobConfig->ucRxDmaDeviceIndex))
			/* END Msg(4:2962)-18 */
			#endif
			
			Expected Behavior: None
			
			Actual Behavior: Illegal Memory access could happen if the if condition mentioned in the problem description fails.
				BUG	OPEN ISSUE

27978	SPI	Spi_SyncTransmit API is not working properly.	Problem Description: when calling Spi_SyncTransmit() an exception is occurring from private API Spi_HWTransmitSyncJob(). On analysis we found that this is because of improper handling of a while loop exit criteria, resulting in illegal memory access. Expected behavior: Spi_SyncTransmit() execute with out any exception. Actual behavior: An exception is occurring while execution of Spi_SyncTransmit() API.	BUG	OPEN ISSUE
28233	SPI	Improper pre-compiler switch for the Spi_Mainfunction_Handling function definition	Problem Description: Spi_Mainfunction_Handling function should be invoked only when polling mechanism is selected by Spi_SetAsyncMode API. This mode can be set only when the SPI_LEVEL_DELIVERED is two. but the pre compiler switch for the function definition is as follows #if (((SPI_LEVEL_DELIVERED == SPI_ONE) || (SPI_LEVEL_DELIVERED == SPI_TWO)) \ && (SPI_HWUNIT_ASYNCHRONOUS == STD_ON)) #define SPI_START_SEC_PUBLIC_CODE #include "MemMap.h" FUNC(void, SPI_PUBLIC_CODE) Spi_MainFunction_Handling (void) { ... Expected Behaviour: Spi_Mainfunction_Handling function shall be available in Level 2 only. Actual Behaviour: Spi_Mainfunction_Handling function is available in Level 1 and 2 also.	BUG	OPEN ISSUE
28249	SPI	SpiFifoTimeOut parameter is mandatory	Problem description: SpiFifoTimeOut parameter is made mandatory but it is only valid for CSIH Expected behaviour: SpiFifoTimeOut parameter should be optional (multiplicity 0..1) and additionally there should be a generator error check in case it is not configured for CSIH HW units in FIFO mode.	BUG	OPEN ISSUE

28251	SPI	The information provided about user mode and supervisor mode is not correct in the user manual	Problem Description: In the user manual Table 4-5, it indicates that Spi_MainFunction_Handling() requires Supervisor mode access when Interrupt mode is active (SI. No. 14), though Spi_MainFunction_Handling() is not necessary in interrupt mode. Also, Spi_AsyncTransmit(), SI. No. 4 in Table 4-5, is missing any mark in the Interrupt Mode/user mode column.. Expected Behaviour: Spi_MainFunction_Handling shall be removed in interrupt mode and Spi_AsyncTransmit() shall be corrected for applicable modes. Actual Behaviour: Spi_MainFunction_Handling is marked for both interrupt and Polling modes. and Spi_AsyncTransmit() is missing any mark in the Interrupt Mode/user mode column.	BUG	OPEN ISSUE
28364	SPI	Error ERR083120 is generated when SpiEnableCs is configured as false	Problem Description: When the Parameter SpiEnableCs is configured as false SpiPortPinSelect should not be configured. But when SpiPortPinSelect is not configured tool is generating error ERR083120- the parameter 'SpiPortPinSelect' value in the container 'SpiJob<x>', should be configured as CSL<n> since 'CSIH<x>' is configured. Expected Behaviour: Tool should not generate error. Actual Behaviour: ERR083120 is generated.	BUG	OPEN ISSUE
28456	SPI	Variables are uninitialized when the certain condition does not meet.	Problem description: Some of the Variables are uninitialized when the following conditions are not met. when the SPI_DIRECT_ACCESS_MODE is STD_OFF Expected behavior: Before using the variables, Variables should be initialized. Actual behavior: Before using the variables, Variables are not initialized when SPI_DIRECT_ACCESS_MODE is STD_OFF	BUG	OPEN ISSUE
28676	SPI	Calling of Spi_MainFunction_Handling possible in interrupt mode	Description: If interrupt mode is selected (Spi_SetAsyncMode(SPI_INTERRUPT_MODE)) a call to Spi_MainFunction_Handling() is possible. Functions Spi_TransmitISR and Spi_ReceiveISR are called there without further checks. This can cause corrupted data transmission. Actual Behavior: No error but corrupted data. Expected Behavior: In interrupt mode a call to Spi_MainFunction_Handling shall be rejected, e.g. by DET.	BUG	OPEN ISSUE

28199	WDG	Wdg_SetMode function reports DEM error if WDGIF_OFF_MODE is selected	Problem description:
			As per AUTOSAR specification [WDG160], Wdg_SetMode function supports WDGIF_OFF_MODE.
			And the user sets WdgDisableAllowed parameter 'true' in Configuration tool.
			However, in code Wdg_59_DriverA_SetMode function reports a Dem Error in case WDGIF_OFF_MODE is selected.
			
			Wdg_59_DriverA_SetMode function in Wdg_59_DriverA.c:
			if (Mode == WDGIF_OFF_MODE)
			{
			/* Report Error to DEM */
			Dem_ReportErrorStatus(WDG_59_DRIVERA_E_DISABLE_REJECTED,
			DEM_EVENT_STATUS_FAILED);
			
			WDG driver does not allow Wdg_SetMode function to translate the state into OFF by "4.4 WDG State Diagram" in WDG Driver Component Embedded User's Manual Rev.0.01 Nov 2013.
			
			Customer need to know the background for this.
			
			Expected behaviour:
			Wdg_SetMode function shall report DEM error only if required mode is 'WDGIF_OFF_MODE' and 'WdgDisableAllowed' is false.
			
			Actual behaviour:
			Wdg_SetMode function is reporting DEM error only if required mode is 'WDGIF_OFF_MODE' and 'WdgDisableAllowed' is true.
				BUG	OPEN
					ISSUE