/IPB/20_Software/SCDD Library/LVDC DSP_C

SCDD_Mpu

Software Component Detailed Design

Version: 0.8
Printed by: I-Ritesh.K
Printed on: Friday, August 2, 2024

ID	Software Component Detailed Design
SCDD_ Mpu_1	1 **Software Component Design Description**
SCDD_ Mpu_2	1.1 **Introduction**
SCDD_ Mpu_3	This document describes the needed requirements for a SWC or BSWM.
SCDD_ Mpu_4	This is module is the Software Component Detail Description. It contains each SW component of each SW architecture. It is always structured in: External Interface Internal Design Requirements

ID	Software Component Detailed Design
SCDD_Mpu_5	2 **Attributes**
SCDD_Mpu_6	Agreed attributes for SWE.3 (ENG.6)

ID	Software Component Detailed Design
SCDD_ Mpu_10	3 **Views**
SCDD_ Mpu_11	**Review View**: This view is used for SCDD edit and review.

SCDD_Mpu_13

4 Mpu

SCDD_Mpu_14

4.1 External Interfaces

SCDD_Mpu_15

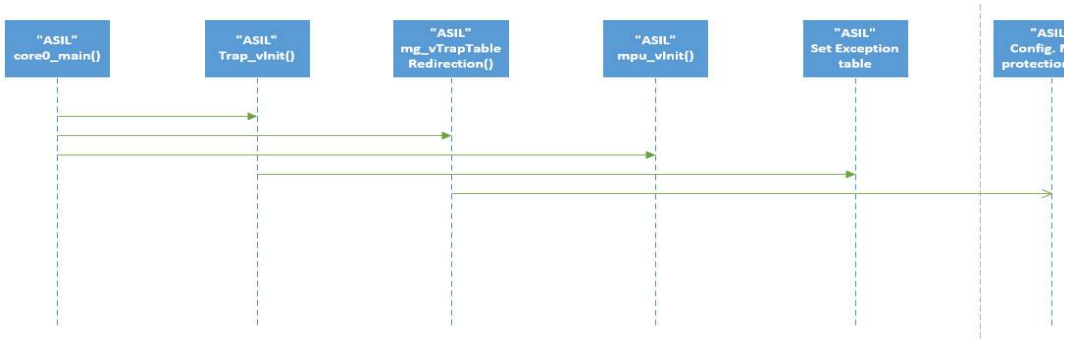
The function interface of this component are as following:

Function	Signal Name	Data Type	Direction
MPU_vInit	void	void	N/A
MPU_vTrapInit	void	void	N/A
MPU_vTrapTableRedirection	N/A	N/A	N/A
MPU_vTrapTableRestore	N/A	N/A	N/A

SCDD_Mpu_16

4.2 Internal design

SCDD_Mpu_97



SCDD_Mpu_184

Detailed Memory Regions and descriptor design can be found here : https://desoeap16.delta.corp/svn/External_IPB_auto_pag/trunk/20_Design/23_Software/2311_Safety/IPB_LVDC_FFI_Access_Protection.xlsx

Memory Space	Physical address	MBIST	Memory Space Areas	Address Range	CMPU Core0	CMPU Access
FLASH Space	PFLASH0	N/A	—	0xA0000000 - 0xA0080000	CODE_MPU_PFLASH0_QM_NoCache	P0 : R/W, P1 : R/W
	PFLASH0	N/A	—	0x80000000 - 0x80080000	CODE_MPU_PFLASH0_QM_Cache	
	DFLASH	N/A	—	0xAF000000 - 0xAF018000	DATA_MPU_DFLASH_QM	
	UCBs	N/A	—	0xAF100000 - 0xAF102000	DATA_MPU_UCBs_QM	
LOCAL RAM Space	DSPR0 0x70000000 - 0x7000BFFF	Yes	DSPR_Core0_STACK (0x70008E00 - 0x70009AFF)	__USTACK0_END __USTACK0	DATA_MPU_CORE0_DSPR0_STACK	P0 : R/W P1 : R/W
				__ISTACK0_END __ISTACK0		
			DSPR_Core0_RAM	__DSPR_QM_VAR_START __DSPR_QM_VAR_END	DATA_MPU_CORE0_DSPR0_QM	P0 : R/W P1 : R/W
				__DSPR_ASIL_VAR_START __DSPR_ASIL_VAR_END	DATA_MPU_CORE0_DSPR0_ASILB	P0 : R/W P1 : R
				__DSPR_DEFAULT_VAR_START __DSPR_DEFAULT_VAR_END	DATA_MPU_CORE0_DSPR0_DEFAULT_QM	No_Acce
PERIPHERAL RAM Space	PERIPHERAL SPACE	Only for CAN RAM	N/A	0xF0000000 - 0xF8820000	DATA_MPU_SEG15_REGs_QM	P0 : R/W P1 : R/W

SCDD_Mpu_18

4.3 Requirements

SCDD_Mpu_19

4.3.1 ASIL

SCDD_Mpu_23

The MPU component is ASIL-B level.

SCDD_Mpu_20

4.3.2 Data Define

ID	Software Component Detailed Design
SCDD_Mpu_24	The C code of this component can be found in following link: https://desoeap16.delta.corp/svn/IPB_PPE_auto_porsche/trunk/20_Design/23_Software/2304_Implementation/10_APPL/40_DcDcController/4010_HSFB_LVDC_B1_MBD/30_Bsw/Mpu
SCDD_Mpu_21	4.3.3 **MPU Functions**
SCDD_Mpu_42	4.3.3.1 **ASIL Functions**
SCDD_Mpu_26	4.3.3.1.1 **void Trap_vInit(void)**
SCDD_Mpu_28	Initialize Exception (Trap) vector table with difffrent trap Handlers
SCDD_Mpu_27	4.3.3.1.2 **void mg_vTrapTableRedirection(void)**
SCDD_Mpu_29	Save default Trap table base address in RAM and redirect CPU_BTV to Customized Trap vector table
SCDD_Mpu_119	The Entry point of the Customized trap table should be TriCore_trap_table() Function
SCDD_Mpu_30	4.3.3.1.3 **void mg_vTrapTableRestore(void)**
SCDD_Mpu_34	Restore default Trap table base address
SCDD_Mpu_117	4.3.3.1.4 **void TriCore_trap_table(void)**
SCDD_Mpu_118	Entry point for all MCU traps handlers
SCDD_Mpu_120	When Trap fires MCU will jump to this function and the according to the Trap source on of the below handler will be excuted
SCDD_Mpu_185	Trap table : void _trap_0(void) - TIN 0 : VIRTUAL_ADDRESS_FILL - TIN 1 : VIRTUAL_ADDRESS_PROTECTION void _trap_1(void) - TIN 1 : PRIVILEGE_INSTRUCTION - TIN 2 : MEMORY_PROTECTION_READ - TIN 3 : MEMORY_PROTECTION_WRITE - TIN 4 : MEMORY_PROTECTION_EXECUTION - TIN 5 : PROTECTION_PERIPHERAL_ACCESS - TIN 6 : MEMORY_PROTECTION_NULL_ADDRESS - TIN 7 : GLOBAL_REGISTER_WRITE_PROTECTION

ID	Software Component Detailed Design
SCDD_Mpu_185	void _trap_2(void) - TIN 1 : ILLEGAL_OPCODE - TIN 2 : UNIMPLEMENTED_OPCODE - TIN 3 : INVALID_OPERAND_SPECIFICATION - TIN 4 : DATA_ADDRESS_ALIGNMENT - TIN 5 : INVALID_LOCAL_MEMORY_ADDRESS void _trap_3(void) - TIN 1 : FREE_CONTEXT_LIST_DEPLETION - TIN 2 : CALL_DEPTH_OVERFLOW - TIN 3 : CALL_DEPTH_UNDERFLOW - TIN 4 : FREE_CONTEXT_LIST_UNDERFLOW - TIN 5 : CALL_STACK_UNDERFLOW - TIN 6 : CONTEXT_TYPE - TIN 7 : NESTING_ERROR void _trap_4(void) - TIN 1 : PROGRAM_FETCH_SYNCHRONOUS_ERROR - TIN 2 : DATA_ACCESS_SYNCHRONOUS_ERROR - TIN 3 : DATA_ACCESS_SYNCHRONOUS_ERROR void _trap_5(void) - TIN 1 : ARITHEMETIC_OVERFLOW - TIN 2 : STICKY_ARITHEMETIC_OVERFLOW void _trap_6(void) - TIN 0 - 255 void _trap_7(void) - TIN 0 : UNEXPECTED_NMI
SCDD_Mpu_113	4.3.3.1.5 **void IOHWSF_vDefaultErrorHandler(IOHWSF_E_ErrorTypeeError)**
SCDD_Mpu_114	This Function should impement the reaction for all MCU traps and other errors.
SCDD_Mpu_115	This function shall act as secondary safety path in case of SBC reset failure on detecting PFM violations, errors in SBC communication or monitoring via QSPI and MPU failures.
SCDD_Mpu_116	It shall perform System reset by calling mg_vMcu_PerformReset().
SCDD_Mpu_122	4.3.3.1.6 **void mpu_vInit(void)**
SCDD_Mpu_123	Configure and initialize memory protection unit

ID	Software Component Detailed Design
SCDD_Mpu_124	The MPU should be configured to have 2 Protecton sets : - ASIL Protection Set - QM Protection Set
SCDD_Mpu_126	MPU should be configured to allow access to ASIL memory **ONLY** from trusted (ASIL) tasks
SCDD_Mpu_127	MPU should be configured to allow access to QM memory from ALL tasks
SCDD_Mpu_128	this function should enable core MPU after its configuration is done
SCDD_Mpu_129	4.3.3.1.7 **static uint8 mpu_u8SetProtRange(uint8 u8ProtectionSet)**
SCDD_Mpu_143	**u8ProtectionSet (Direction in) :** this variable holdes Prtotection set ID with the function will use to configure
SCDD_Mpu_144	**Return :** Configuration status code
SCDD_Mpu_130	Configure address ranges for memory protection sets
SCDD_Mpu_131	PROTECTION_SET_0 should be used as ASILB protection set with full Access to all memory areas
SCDD_Mpu_133	PROTECTION_SET_1 should be used as QM protection set with limited Access only to non-ASIL memory areas
SCDD_Mpu_134	4.3.3.1.8 **static uint8 mpu_u8SetProtAccess(uint8 u8ProtectionSet)**
SCDD_Mpu_147	**u8ProtectionSet (Direction in) :** this variable holdes Prtotection set ID with the function will use to configure
SCDD_Mpu_148	**Return :** Configuration status code
SCDD_Mpu_135	Configure memory protection sets accesses
SCDD_Mpu_136	Read , Write and Excute access has to be granted for PROTECTION_SET_0 for full memory range

ID	Software Component Detailed Design
SCDD_Mpu_137	Read access has to be granted for PROTECTION_SET_1 for full memory range
SCDD_Mpu_139	Write access has to be granted for PROTECTION_SET_1 only for non ASIL memory ranges
SCDD_Mpu_140	Execute access has to be granted for PROTECTION_SET_1 for full Flash range
SCDD_Mpu_141	4.3.3.1.9 **static uint8 mpu_u8Enable_MemProt(void)**
SCDD_Mpu_149	**Return :** Enabling MPU status
SCDD_Mpu_150	This Function enable Core0 memory protection unit
SCDD_Mpu_151	This function must be called after MPU configuration is done
SCDD_Mpu_152	4.3.3.1.10 **static void mpu_vDefine_DataProtRange(uint32 u32lowerBoundAddress, uint32 u32UpperBoundAddress, uint8 u8RangeId)**
SCDD_Mpu_154	**u32lowerBoundAddress(Direction in) :** Start address of descriptor data range
SCDD_Mpu_156	**u32UpperBoundAddress(Direction in) :** End address of descriptor data range
SCDD_Mpu_157	**u8RangeId(Direction in) :** Descriptor ID
SCDD_Mpu_153	This function Write lower and upper addresses in DPRx registers according to range index
SCDD_Mpu_158	-It Set the lower and upper bound of CPU Data Protection Range x (0-15) -Maximum 16 ranges allowed
SCDD_Mpu_159	4.3.3.1.11 **static void mpu_vDefine_CodeProtRange(uint32 u32lowerBoundAddress, uint32 u32UpperBoundAddress, uint8 u8RangeId)**

ID	Software Component Detailed Design
SCDD_Mpu_160	**u32lowerBoundAddress(Direction in) :** Start address of descriptor data range
SCDD_Mpu_161	**u32UpperBoundAddress(Direction in) :** End address of descriptor data range
SCDD_Mpu_162	**u8RangeId(Direction in) :** Descriptor ID
SCDD_Mpu_164	This function Write lower and upper addresses in CPRx registers according to range indexty
SCDD_Mpu_163	-It Set the lower and upper bound of CPU Code Protection Range x (0-7) -Maximum 8 ranges allowed
SCDD_Mpu_165	4.3.3.1.12 **static void mpu_vEnable_DataRd(uint8 u8ProtectionSet, uint8 u8RangeId)**
SCDD_Mpu_170	**u8ProtectionSet (Direction in) :** this variable holdes Prtotection set ID with the function will use to configure
SCDD_Mpu_169	**u8RangeId(Direction in) :** Descriptor ID
SCDD_Mpu_168	Using range index, enable read access for respective protection set by writing into DPREx registers
SCDD_Mpu_175	4.3.3.1.13 **static void mpu_vEnable_DataWr(uint8 u8ProtectionSet, uint8 u8RangeId)**
SCDD_Mpu_176	**u8ProtectionSet (Direction in) :** this variable holdes Prtotection set ID with the function will use to configure
SCDD_Mpu_177	**u8RangeId(Direction in) :** Descriptor ID
SCDD_Mpu_178	Using range index, enable write access for respective protection set by writing into DPWEx registers
SCDD_Mpu_179	4.3.3.1.14 **static void mpu_vEnable_CodeExe(uint8 u8ProtectionSet, uint8 u8RangeId)**
SCDD_Mpu_180	**u8ProtectionSet (Direction in) :** this variable holdes Prtotection set ID with the function will use to configure

ID	Software Component Detailed Design
SCDD_Mpu_181	**u8RangeId(Direction in) :** Descriptor ID
SCDD_Mpu_182	Using range index, enable code execute access for respective protection set by writing into CPXEx registers
SCDD_Mpu_186	4.3.3.1.15 **inline void mpu_vSet_ActiveProtSet(uint8 protectionSet)**
SCDD_Mpu_187	**uint8 protectionSet (Direction in) :** This variable holds the Protection set ID to be set for the current task.
SCDD_Mpu_188	This function will activate the given Protection Set for the current callee task.