

ADAS Graduation Project

Generated on Mon Dec 12 2022 20:39:45 for ADAS Graduation Project by Doxygen 1.9.5

Mon Dec 12 2022 20:39:45

1 ADAS Graudation Project	1
1.1 Description	1
1.1.1 Adaptive Cruise Control	1
1.1.2 Adaptive Light Control	1
1.1.3 Driver Health Care	1
1.2 Hardware specifications	1
2 Module Index	3
2.1 Modules	3
3 Data Structure Index	5
3.1 Data Structures	5
4 File Index	7
4.1 File List	7
5 Module Documentation	9
5.1 Bit Manipulation Math Macros	9
5.1.1 Detailed Description	9
5.1.2 Macro Definition Documentation	9
5.1.2.1 SET_BIT	9
5.1.2.2 CLR_BIT	10
5.1.2.3 TOGGLE_BIT	10
5.1.2.4 GET_BIT	10
5.2 Bit Group Manipulation Math Macros	10
5.2.1 Detailed Description	11
5.2.2 Macro Definition Documentation	11
5.2.2.1 SET_BITS	11
5.2.2.2 CLR_BITS	11
5.2.2.3 TOGGLE_BITS	11
5.2.2.4 GET_BITS	12
5.3 Compiler standard macros	12
5.3.1 Detailed Description	12
5.3.2 Macro Definition Documentation	12
5.3.2.1 VAR	12
5.3.2.2 FUNC	13
5.3.2.3 P2VAR	13
5.3.2.4 P2CONST	13
5.3.2.5 CONSTP2VAR	13
5.3.2.6 CONSTP2CONST	14
5.3.2.7 P2FUNC	14
5.3.2.8 CONST	14
5.3.2.9 STATIC	14
5.4 Utilities macros	15

5.4.1 Detailed Description	15
5.4.2 Macro Definition Documentation	15
5.4.2.1 MY_MS_DELAY	15
5.5 Standard types	15
5.5.1 Detailed Description	15
5.5.2 Typedef Documentation	15
5.5.2.1 bool_t	16
5.5.2.2 u8_t	16
5.5.2.3 u16_t	16
5.5.2.4 u32_t	16
5.5.2.5 s8_t	17
5.5.2.6 s16_t	17
5.5.2.7 s32_t	17
5.5.2.8 fl32_t	17
5.5.2.9 fl64_t	18
5.6 Standard values	18
5.6.1 Detailed Description	18
5.6.2 Macro Definition Documentation	18
5.6.2.1 TRUE	18
5.6.2.2 FALSE	19
5.6.2.3 NULL	19
5.6.2.4 INITIAL_ZERO	19
5.6.2.5 FLAG_SET	19
5.6.2.6 FLAG_CLEARED	20
5.6.2.7 RUN	20
5.6.2.8 STOP	20
5.6.2.9 PRESSED	20
5.6.2.10 RELEASED	21
5.7 Interrupt line IDs	21
5.7.1 Detailed Description	22
5.7.2 Macro Definition Documentation	22
5.7.2.1 EXTI_LINE0	22
5.7.2.2 EXTI_LINE1	22
5.7.2.3 EXTI_LINE2	22
5.7.2.4 EXTI_LINE3	22
5.7.2.5 EXTI_LINE4	23
5.7.2.6 EXTI_LINE5	23
5.7.2.7 EXTI_LINE6	23
5.7.2.8 EXTI_LINE7	23
5.7.2.9 EXTI_LINE8	24
5.7.2.10 EXTI_LINE9	24
5.7.2.11 EXTI_LINE10	24

5.7.2.12 EXTI_LINE11	24
5.7.2.13 EXTI_LINE12	25
5.7.2.14 EXTI_LINE13	25
5.7.2.15 EXTI_LINE14	25
5.7.2.16 EXTI_LINE15	25
5.8 Interrupt trigger status	26
5.8.1 Detailed Description	26
5.8.2 Macro Definition Documentation	26
5.8.2.1 EXTI_FallingEdge	26
5.8.2.2 EXTI_RisingEdge	26
5.8.2.3 EXTI_OnChange	26
5.9 EXTI Memory Addresses	27
5.9.1 Detailed Description	27
5.9.2 Macro Definition Documentation	27
5.9.2.1 EXTI_BASE_ADDRESS	27
5.10 EXTI Registers	27
5.10.1 Detailed Description	27
5.10.2 Macro Definition Documentation	27
5.10.2.1 MEXTI	27
5.11 EXTI Lines Status	28
5.11.1 Detailed Description	28
5.11.2 Macro Definition Documentation	28
5.11.2.1 ENABLE	28
5.11.2.2 DISABLE	28
5.12 EXTI Line Settings	28
5.12.1 Detailed Description	28
5.12.2 Macro Definition Documentation	28
5.12.2.1 EXTI_MAX_EXTI_NUM	29
5.13 SYSCFG Memory Addresses	29
5.13.1 Detailed Description	29
5.13.2 Macro Definition Documentation	29
5.13.2.1 SYSCFG_BASE_ADDR	29
5.14 SYSCFG Memory Registers	29
5.14.1 Detailed Description	29
5.14.2 Macro Definition Documentation	29
5.14.2.1 MSYSCFG	30
5.15 GPIO Modes	30
5.15.1 Detailed Description	30
5.15.2 Macro Definition Documentation	30
5.15.2.1 GPIOx_MODE_INPUT	30
5.15.2.2 GPIOx_MODE_OUTPUT	30
5.15.2.3 GPIOx_MODE_AF	31

5.15.2.4 GPIOx_MODE_ANALOG	31
5.16 GPIO Ports	31
5.16.1 Detailed Description	31
5.16.2 Macro Definition Documentation	31
5.16.2.1 GPIO_PORTA	31
5.16.2.2 GPIO_PORTB	32
5.16.2.3 GPIO_PORTC	32
5.17 GPIO Output Types	32
5.17.1 Detailed Description	32
5.17.2 Macro Definition Documentation	32
5.17.2.1 GPIOx_OPENDRAIN	32
5.17.2.2 GPIOx_PUSH_PULL	33
5.18 GPIO PIN Speed	33
5.18.1 Detailed Description	33
5.18.2 Macro Definition Documentation	33
5.18.2.1 GPIOx_LowSpeed	33
5.18.2.2 GPIOx_MediumSpeed	33
5.18.2.3 GPIOx_HighSpeed	34
5.18.2.4 GPIOx_VeryHighSpeed	34
5.19 GPIO Pull Types	34
5.19.1 Detailed Description	34
5.19.2 Macro Definition Documentation	34
5.19.2.1 GPIOx_NoPull	34
5.19.2.2 GPIOx_PullUp	35
5.19.2.3 GPIOx_PullDown	35
5.20 GPIO Output Values	35
5.20.1 Detailed Description	35
5.20.2 Macro Definition Documentation	35
5.20.2.1 GPIOx_HIGH	35
5.20.2.2 GPIOx_LOW	36
5.21 GPIO Output PINs	36
5.21.1 Detailed Description	37
5.21.2 Macro Definition Documentation	37
5.21.2.1 GPIOx_PIN0	37
5.21.2.2 GPIOx_PIN1	37
5.21.2.3 GPIOx_PIN2	37
5.21.2.4 GPIOx_PIN3	37
5.21.2.5 GPIOx_PIN4	38
5.21.2.6 GPIOx_PIN5	38
5.21.2.7 GPIOx_PIN6	38
5.21.2.8 GPIOx_PIN7	38
5.21.2.9 GPIOx_PIN8	39

5.21.2.10 GPIOx_PIN9	39
5.21.2.11 GPIOx_PIN10	39
5.21.2.12 GPIOx_PIN11	39
5.21.2.13 GPIOx_PIN12	40
5.21.2.14 GPIOx_PIN13	40
5.21.2.15 GPIOx_PIN14	40
5.21.2.16 GPIOx_PIN15	40
5.22 GPIO Alternate Functions	41
5.22.1 Detailed Description	41
5.22.2 Macro Definition Documentation	41
5.22.2.1 GPIOx_AF0	41
5.22.2.2 GPIOx_AF1	42
5.22.2.3 GPIOx_AF2	42
5.22.2.4 GPIOx_AF3	42
5.22.2.5 GPIOx_AF4	42
5.22.2.6 GPIOx_AF5	43
5.22.2.7 GPIOx_AF6	43
5.22.2.8 GPIOx_AF7	43
5.22.2.9 GPIOx_AF8	43
5.22.2.10 GPIOx_AF9	44
5.22.2.11 GPIOx_AF10	44
5.22.2.12 GPIOx_AF11	44
5.22.2.13 GPIOx_AF12	44
5.22.2.14 GPIOx_AF13	45
5.22.2.15 GPIOx_AF14	45
5.22.2.16 GPIOx_AF15	45
5.23 GPIO Addresses	45
5.23.1 Detailed Description	45
5.23.2 Macro Definition Documentation	45
5.23.2.1 GPIOA_BASE_ADDRESS	46
5.23.2.2 GPIOB_BASE_ADDRESS	46
5.23.2.3 GPIOC_BASE_ADDRESS	46
5.23.2.4 CLK_SOURCE	46
5.24 GPIO Registers	46
5.24.1 Detailed Description	47
5.24.2 Macro Definition Documentation	47
5.24.2.1 GPIOA	47
5.24.2.2 GPIOB	47
5.24.2.3 GPIOC	47
5.25 Interrupt priority grouping	47
5.25.1 Detailed Description	48
5.25.2 Macro Definition Documentation	48

5.25.2.1 _16GROUP_NoSub_Priorities	48
5.25.2.2 _8GROUP_2Sub_Priorities	49
5.25.2.3 _4GROUP_4Sub_Priorities	49
5.25.2.4 _2GROUP_8Sub_Priorities	49
5.25.2.5 NoGROUP_16Sub_Priorities	50
5.25.2.6 GROUP_4BITS	50
5.25.2.7 GROUP_3BITS	50
5.25.2.8 GROUP_2BITS	51
5.25.2.9 GROUP_1BITS	51
5.25.2.10 GROUP_0BITS	51
5.26 Group priorities	51
5.26.1 Detailed Description	52
5.26.2 Macro Definition Documentation	52
5.26.2.1 NO_GROUP_PRIORITY	53
5.26.2.2 GROUP_PRIORITY_0	53
5.26.2.3 GROUP_PRIORITY_1	53
5.26.2.4 GROUP_PRIORITY_2	53
5.26.2.5 GROUP_PRIORITY_3	54
5.26.2.6 GROUP_PRIORITY_4	54
5.26.2.7 GROUP_PRIORITY_5	54
5.26.2.8 GROUP_PRIORITY_6	54
5.26.2.9 GROUP_PRIORITY_7	55
5.26.2.10 GROUP_PRIORITY_8	55
5.26.2.11 GROUP_PRIORITY_9	55
5.26.2.12 GROUP_PRIORITY_10	55
5.26.2.13 GROUP_PRIORITY_11	56
5.26.2.14 GROUP_PRIORITY_12	56
5.26.2.15 GROUP_PRIORITY_13	56
5.26.2.16 GROUP_PRIORITY_14	56
5.26.2.17 GROUP_PRIORITY_15	57
5.27 Sub-Group priorities	57
5.27.1 Detailed Description	58
5.27.2 Macro Definition Documentation	58
5.27.2.1 NO_SUB_PRIORITY	58
5.27.2.2 SUB_PRIORITY_0	58
5.27.2.3 SUB_PRIORITY_1	58
5.27.2.4 SUB_PRIORITY_3	59
5.27.2.5 SUB_PRIORITY_2	59
5.27.2.6 SUB_PRIORITY_4	59
5.27.2.7 SUB_PRIORITY_5	59
5.27.2.8 SUB_PRIORITY_6	60
5.27.2.9 SUB_PRIORITY_7	60

5.27.2.10 SUB_PRIORITY_8	60
5.27.2.11 SUB_PRIORITY_9	60
5.27.2.12 SUB_PRIORITY_10	61
5.27.2.13 SUB_PRIORITY_11	61
5.27.2.14 SUB_PRIORITY_12	61
5.27.2.15 SUB_PRIORITY_13	61
5.27.2.16 SUB_PRIORITY_14	62
5.27.2.17 SUB_PRIORITY_15	62
5.28 NVIC Vector Table	62
5.28.1 Detailed Description	64
5.28.2 Macro Definition Documentation	64
5.28.2.1 WWDG	65
5.28.2.2 EXTI16	65
5.28.2.3 EXTI21	65
5.28.2.4 EXTI22	65
5.28.2.5 FLASH	66
5.28.2.6 RCC	66
5.28.2.7 EXTI0	66
5.28.2.8 EXTI1	66
5.28.2.9 EXTI2	67
5.28.2.10 EXTI3	67
5.28.2.11 EXTI4	67
5.28.2.12 DMA1_STREAM0	67
5.28.2.13 DMA1_STREAM1	68
5.28.2.14 DMA1_STREAM2	68
5.28.2.15 DMA1_STREAM3	68
5.28.2.16 DMA1_STREAM4	68
5.28.2.17 DMA1_STREAM5	69
5.28.2.18 DMA1_STREAM6	69
5.28.2.19 ADC	69
5.28.2.20 EXTI9	69
5.28.2.21 TIM1_BRK_TIM9	70
5.28.2.22 TIM1_UP_TIM10	70
5.28.2.23 TIM1_TRG_COM_TIM11	70
5.28.2.24 TIM1_CC	70
5.28.2.25 TIM2	71
5.28.2.26 TIM3	71
5.28.2.27 TIM4	71
5.28.2.28 I2C1_EV	71
5.28.2.29 I2C1_ER	72
5.28.2.30 I2C2_EV	72
5.28.2.31 I2C2_ER	72

5.28.2.32 SPI1	72
5.28.2.33 SPI2	73
5.28.2.34 USART1	73
5.28.2.35 USART2	73
5.28.2.36 EXTI15_10	73
5.28.2.37 EXTI17	74
5.28.2.38 EXTI18	74
5.28.2.39 DMA1_STREAM7	74
5.28.2.40 SDIO	74
5.28.2.41 TIM5	75
5.28.2.42 SPI3	75
5.28.2.43 DMA2_STREAM0	75
5.28.2.44 DMA2_STREAM1	75
5.28.2.45 DMA2_STREAM2	76
5.28.2.46 DMA2_STREAM5	76
5.28.2.47 DMA2_STREAM3	76
5.28.2.48 DMA2_STREAM4	76
5.28.2.49 OTG_FS	77
5.28.2.50 DMA2_STREAM6	77
5.28.2.51 DMA2_STREAM7	77
5.28.2.52 USART6	77
5.28.2.53 I2C3_EV	78
5.28.2.54 I2C3_ER	78
5.28.2.55 FPU	78
5.28.2.56 SPI4	78
5.29 NVIC Addresses	78
5.29.1 Detailed Description	79
5.29.2 Macro Definition Documentation	79
5.29.2.1 NVIC_BASE_ADDRESS	79
5.29.2.2 SCB_BASE_ADDRESS	79
5.30 NVIC Registers	79
5.30.1 Detailed Description	80
5.30.2 Macro Definition Documentation	80
5.30.2.1 MSCB	80
5.30.2.2 MNVIC	80
5.31 NVIC Settable Priorities	80
5.31.1 Detailed Description	80
5.31.2 Macro Definition Documentation	81
5.31.2.1 PEND_SV	81
5.31.2.2 SYSTICK	81
5.31.2.3 SV_CALL	81
5.31.2.4 MEMORY_MANAGE	81

5.31.2.5 BUS_FAULT	82
5.31.2.6 USAGE_FAULT	82
5.32 ID options	82
5.32.1 Detailed Description	82
5.32.2 Macro Definition Documentation	82
5.32.2.1 RCC_AHB1	82
5.32.2.2 RCC_AHB2	83
5.32.2.3 RCC_APB1	83
5.32.2.4 RCC_APB2	83
5.32.2.5 RCC_AHB1LPENR	83
5.33 ID options	84
5.33.1 Detailed Description	85
5.33.2 Macro Definition Documentation	85
5.33.2.1 AHB1ENR_DMA2EN	85
5.33.2.2 AHB1ENR_DMA1EN	85
5.33.2.3 AHB1ENR_CRCEN	86
5.33.2.4 AHB1ENR_GPIOHEN	86
5.33.2.5 AHB1ENR_GPIOEEN	86
5.33.2.6 AHB1ENR_GPIODEN	86
5.33.2.7 AHB1ENR_GPIOCEN	87
5.33.2.8 AHB1ENR_GPIOBEN	87
5.33.2.9 AHB1ENR_GPIOAEN	87
5.33.2.10 AHB2ENR_OTGFSEN	87
5.33.2.11 APB1ENR_PWREN	88
5.33.2.12 APB1ENR_I2C3EN	88
5.33.2.13 APB1ENR_I2C2EN	88
5.33.2.14 APB1ENR_I2C1EN	88
5.33.2.15 APB1ENR_USART2EN	89
5.33.2.16 APB1ENR_SPI3EN	89
5.33.2.17 APB1ENR_SPI2EN	89
5.33.2.18 APB1ENR_WWDGEN	89
5.33.2.19 APB1ENR_TIM5EN	90
5.33.2.20 APB1ENR_TIM4EN	90
5.33.2.21 APB1ENR_TIM3EN	90
5.33.2.22 APB1ENR_TIM2EN	90
5.33.2.23 APB2ENR_TIM11EN	91
5.33.2.24 APB2ENR_TIM10EN	91
5.33.2.25 APB2ENR_TIM9EN	91
5.33.2.26 APB2ENR_SYSCFGEN	91
5.33.2.27 APB2ENR_SPI4EN	92
5.33.2.28 APB2ENR_SPI1EN	92
5.33.2.29 APB2ENR_SDIOEN	92

5.33.2.30 APB2ENR_ADC1EN	92
5.33.2.31 APB2ENR_USART6EN	93
5.33.2.32 APB2ENR_USART1EN	93
5.33.2.33 APB2ENR_TIM1EN	93
5.33.2.34 AHB1LPENR_FLITFLPEN	93
5.34 Systick Interval Mode Configuration	94
5.34.1 Detailed Description	94
5.34.2 Macro Definition Documentation	94
5.34.2.1 SINGLE_INTERVAL_MODE	94
5.34.2.2 PERIODIC_INTERVAL_MODE	94
5.35 Delay Units	94
5.35.1 Detailed Description	95
5.35.2 Macro Definition Documentation	95
5.35.2.1 MILLI_SEC	95
5.35.2.2 MICRO_SEC	95
5.35.2.3 SEC	95
5.36 Systick Addresses	95
5.36.1 Detailed Description	95
5.36.2 Macro Definition Documentation	95
5.36.2.1 SysTick_BASE_ADDRESS	96
5.37 Systick Registers	96
5.37.1 Detailed Description	96
5.37.2 Macro Definition Documentation	96
5.37.2.1 SysTick	96
5.38 Systick Register Bits Positions	96
5.38.1 Detailed Description	96
5.38.2 Macro Definition Documentation	96
5.38.2.1 COUNTFLAG	97
5.38.2.2 CLKSOURCE	97
5.38.2.3 TICKINT	97
5.38.2.4 COUNTER_ENABLE	97
5.39 Systick Clock Sources	98
5.39.1 Detailed Description	98
5.39.2 Macro Definition Documentation	98
5.39.2.1 AHB_DividedBy8	98
5.39.2.2 AHB	98
5.40 Systick Exception (Interrupt) Status	98
5.40.1 Detailed Description	98
5.40.2 Macro Definition Documentation	98
5.40.2.1 Dont AssertRequest	99
5.40.2.2 AssertRequest	99

6 Data Structure Documentation	101
6.1 DCM_MotorConfiguration Struct Reference	101
6.1.1 Detailed Description	101
6.1.2 Field Documentation	101
6.1.2.1 Side1Port	102
6.1.2.2 Side1Pin	102
6.1.2.3 Side2Port	102
6.1.2.4 Side2Pin	102
6.2 EXTI_ConfigType Struct Reference	103
6.2.1 Detailed Description	103
6.2.2 Field Documentation	103
6.2.2.1 LineNum	103
6.2.2.2 PortNum	103
6.2.2.3 TriggerStatus	104
6.3 EXTI_Type Struct Reference	104
6.3.1 Detailed Description	104
6.3.2 Field Documentation	104
6.3.2.1 IMR	105
6.3.2.2 EMR	105
6.3.2.3 RTSR	105
6.3.2.4 FTSR	105
6.3.2.5 SWIER	105
6.3.2.6 PR	106
6.4 GPIOx_MemoryMapType Struct Reference	106
6.4.1 Detailed Description	106
6.4.2 Field Documentation	107
6.4.2.1 MODERx	107
6.4.2.2 OTYPERx	107
6.4.2.3 OSPEEDRx	107
6.4.2.4 PUPDRx	107
6.4.2.5 IDRx	107
6.4.2.6 ODRx	108
6.4.2.7 BSRRx	108
6.4.2.8 LCKRx	108
6.4.2.9 AFRLx	108
6.4.2.10 AFRHx	108
6.5 MGPIOp_ConfigType Struct Reference	109
6.5.1 Detailed Description	109
6.5.2 Field Documentation	109
6.5.2.1 Port	109
6.5.2.2 Pin	110
6.5.2.3 Mode	110

6.5.2.4 OutputType	110
6.5.2.5 OutputSpeed	110
6.5.2.6 InputType	111
6.5.2.7 AF_Type	111
6.6 MSYSCFG_MemMap_t Struct Reference	111
6.6.1 Detailed Description	111
6.6.2 Field Documentation	112
6.6.2.1 MEMRMP	112
6.6.2.2 PMC	112
6.6.2.3 EXTICR	112
6.6.2.4 CMPCR	112
6.7 NVIC_MemoryMapType Struct Reference	113
6.7.1 Detailed Description	113
6.7.2 Field Documentation	113
6.7.2.1 ISERx	113
6.7.2.2 RESERVED0x	114
6.7.2.3 ICERx	114
6.7.2.4 RSERVED1x	114
6.7.2.5 ISPRx	115
6.7.2.6 RESERVED2x	115
6.7.2.7 ICPRx	115
6.7.2.8 RESERVED3x	116
6.7.2.9 IABRx	116
6.7.2.10 RESERVED4x	116
6.7.2.11 IPRx	116
6.8 RCC_MemoryMapType Struct Reference	117
6.8.1 Detailed Description	118
6.8.2 Field Documentation	118
6.8.2.1 CR	118
6.8.2.2 PLLCFGR	119
6.8.2.3 CFGR	119
6.8.2.4 CIR	119
6.8.2.5 AHB1RSTR	119
6.8.2.6 AHB2RSTR	119
6.8.2.7 Reserved1	120
6.8.2.8 Reserved2	120
6.8.2.9 APB1RSTR	120
6.8.2.10 APB2RSTR	120
6.8.2.11 Reserved3	120
6.8.2.12 Reserved4	121
6.8.2.13 AHB1ENR	121
6.8.2.14 AHB2ENR	121

6.8.2.15 Reserved5	121
6.8.2.16 Reserved6	121
6.8.2.17 APB1ENR	122
6.8.2.18 APB2ENR	122
6.8.2.19 Reserved7	122
6.8.2.20 Reserved8	122
6.8.2.21 AHB1LPENR	122
6.8.2.22 AHB2LPENR	123
6.8.2.23 Reserved9	123
6.8.2.24 Reserved10	123
6.8.2.25 APB1LPENR	123
6.8.2.26 APB2LPENR	123
6.8.2.27 Reserved11	124
6.8.2.28 Reserved12	124
6.8.2.29 BDCR	124
6.8.2.30 CSR	124
6.8.2.31 Reserved13	124
6.8.2.32 Reserved14	125
6.8.2.33 SSCGR	125
6.8.2.34 PLLI2SCFGR	125
6.8.2.35 DCKCFGGR	125
6.9 SCB_MemoryMapType Struct Reference	125
6.9.1 Detailed Description	126
6.9.2 Field Documentation	126
6.9.2.1 CPUID	127
6.9.2.2 ICSR	127
6.9.2.3 VTOR	127
6.9.2.4 AIRCR	127
6.9.2.5 SCR	128
6.9.2.6 CCR	128
6.9.2.7 SHPR1	128
6.9.2.8 SHPR2	129
6.9.2.9 SHPR3	129
6.9.2.10 SHCSR	129
6.9.2.11 CFSR	130
6.9.2.12 HFSR	130
6.9.2.13 RESERVED	130
6.9.2.14 MMFAR	131
6.9.2.15 BFAR	131
6.10 SysTick_Type Struct Reference	131
6.10.1 Detailed Description	131
6.10.2 Field Documentation	132

6.10.2.1 CTRL	132
6.10.2.2 LOAD	132
6.10.2.3 VAL	132
6.10.2.4 CALIB	132
7 File Documentation	133
7.1 D:/GProject/ADAS_Project/README.md File Reference	133
7.2 COTS/HAL/DCMOTOR/DCM_config.h File Reference	133
7.2.1 Detailed Description	133
7.3 DCM_config.h	133
7.4 COTS/HAL/DCMOTOR/DCM_interface.h File Reference	134
7.4.1 Detailed Description	134
7.4.2 Function Documentation	134
7.4.2.1 HDCM_vInitMotor()	134
7.4.2.2 HDCM_vMoveForward()	135
7.4.2.3 HDCM_vMoveBackward()	135
7.4.2.4 HDCM_vStopMotor()	135
7.5 DCM_interface.h	136
7.6 COTS/HAL/DCMOTOR/DCM_private.h File Reference	136
7.6.1 Detailed Description	136
7.7 DCM_private.h	136
7.8 COTS/HAL/DCMOTOR/DCM_program.c File Reference	137
7.8.1 Detailed Description	137
7.8.2 Function Documentation	137
7.8.2.1 HDCM_vInitMotor()	138
7.8.2.2 HDCM_vMoveForward()	138
7.8.2.3 HDCM_vMoveBackward()	138
7.8.2.4 HDCM_vStopMotor()	139
7.9 DCM_program.c	139
7.10 COTS/LIB/LSTD_BITMATH.h File Reference	140
7.10.1 Detailed Description	140
7.11 LSTD_BITMATH.h	140
7.12 COTS/LIB/LSTD_COMPILER.h File Reference	141
7.12.1 Detailed Description	141
7.13 LSTD_COMPILER.h	141
7.14 COTS/LIB/LSTD MCU UTILITIES.h File Reference	142
7.14.1 Detailed Description	142
7.15 LSTD MCU UTILITIES.h	142
7.16 COTS/LIB/LSTD TYPES.h File Reference	142
7.16.1 Detailed Description	143
7.17 LSTD TYPES.h	143
7.18 COTS/LIB/LSTD VALUES.h File Reference	143

7.18.1 Detailed Description	144
7.19 LSTD_VALUES.h	144
7.20 COTS/MCAL/EXTI/EXTI_config.h File Reference	145
7.20.1 Macro Definition Documentation	145
7.20.1.1 EXTI_LINE0_EN	145
7.20.1.2 EXTI_LINE0_TRIGGER	146
7.20.1.3 EXTI_LINE1_EN	146
7.20.1.4 EXTI_LINE1_TRIGGER	146
7.20.1.5 EXTI_LINE2_EN	146
7.20.1.6 EXTI_LINE2_TRIGGER	146
7.20.1.7 EXTI_LINE3_EN	147
7.20.1.8 EXTI_LINE3_TRIGGER	147
7.20.1.9 EXTI_LINE4_EN	147
7.20.1.10 EXTI_LINE4_TRIGGER	147
7.20.1.11 EXTI_LINE5_EN	147
7.20.1.12 EXTI_LINE5_TRIGGER	147
7.20.1.13 EXTI_LINE6_EN	148
7.20.1.14 EXTI_LINE6_TRIGGER	148
7.20.1.15 EXTI_LINE7_EN	148
7.20.1.16 EXTI_LINE7_TRIGGER	148
7.20.1.17 EXTI_LINE8_EN	148
7.20.1.18 EXTI_LINE8_TRIGGER	148
7.20.1.19 EXTI_LINE9_EN	149
7.20.1.20 EXTI_LINE9_TRIGGER	149
7.20.1.21 EXTI_LINE10_EN	149
7.20.1.22 EXTI_LINE10_TRIGGER	149
7.20.1.23 EXTI_LINE11_EN	149
7.20.1.24 EXTI_LINE11_TRIGGER	149
7.20.1.25 EXTI_LINE12_EN	150
7.20.1.26 EXTI_LINE12_TRIGGER	150
7.20.1.27 EXTI_LINE13_EN	150
7.20.1.28 EXTI_LINE13_TRIGGER	150
7.20.1.29 EXTI_LINE14_EN	150
7.20.1.30 EXTI_LINE14_TRIGGER	150
7.20.1.31 EXTI_LINE15_EN	151
7.20.1.32 EXTI_LINE15_TRIGGER	151
7.21 EXTI_config.h	151
7.22 COTS/MCAL/EXTI/EXTI_interface.h File Reference	154
7.22.1 Detailed Description	155
7.22.2 Function Documentation	155
7.22.2.1 MEXTI_vInit()	156
7.22.2.2 MEXTI_vInit_WithStruct()	158

7.22.2.3 MEXTI_vEnableLine()	159
7.22.2.4 MEXTI_vDisableLine()	159
7.22.2.5 MEXTI_vSWITrigger()	159
7.22.2.6 MEXTI_vSetTrigger()	159
7.22.2.7 MEXTI_vSetCallback()	160
7.22.2.8 MSYSCFG_vSetEXTIPort()	160
7.23 EXTI_interface.h	160
7.24 COTS/MCAL/EXTI/EXTI_private.h File Reference	161
7.24.1 Detailed Description	162
7.24.2 Macro Definition Documentation	162
7.24.2.1 EXTI IRQs	162
7.25 EXTI_private.h	162
7.26 COTS/MCAL/EXTI/EXTI_program.c File Reference	163
7.26.1 Detailed Description	163
7.26.2 Function Documentation	164
7.26.2.1 MSYSCFG_vSetEXTIPort()	164
7.26.2.2 MEXTI_vInit()	164
7.26.2.3 MEXTI_vInit_WithStruct()	167
7.26.2.4 MEXTI_vEnableLine()	167
7.26.2.5 MEXTI_vDisableLine()	167
7.26.2.6 MEXTI_vSWITrigger()	168
7.26.2.7 MEXTI_vSetTrigger()	168
7.26.2.8 MEXTI_vSetCallback()	169
7.26.2.9 EXTI0_IRQHandler()	169
7.26.2.10 EXTI1_IRQHandler()	169
7.26.2.11 EXTI2_IRQHandler()	170
7.26.2.12 EXTI3_IRQHandler()	170
7.26.2.13 EXTI4_IRQHandler()	170
7.26.2.14 EXTI9_5_IRQHandler()	171
7.26.2.15 EXTI15_10_IRQHandler()	171
7.27 EXTI_program.c	171
7.28 COTS/MCAL/EXTI/SYSCFG_private.h File Reference	177
7.28.1 Detailed Description	177
7.29 SYSCFG_private.h	178
7.30 COTS/MCAL/GPIO/GPIO_config.h File Reference	178
7.30.1 Detailed Description	178
7.30.2 Macro Definition Documentation	179
7.30.2.1 GPIOA_PIN_POS	179
7.30.2.2 GPIOB_PIN_POS	179
7.30.2.3 LCKK_BIT_POS	179
7.31 GPIO_config.h	179
7.32 COTS/MCAL/GPIO/GPIO_interface.h File Reference	180

7.32.1 Detailed Description	183
7.32.2 Function Documentation	183
7.32.2.1 MGPIox_vLockedPins()	183
7.32.2.2 MGPIox_vSetPinMode()	184
7.32.2.3 MGPIox_vSetPinOutputType()	184
7.32.2.4 MGPIox_vSetPinOutputSpeed()	184
7.32.2.5 MGPIox_vSetPinInputPullType()	185
7.32.2.6 MGPIox_u8GetPinValue()	185
7.32.2.7 MGPIox_vSetPinValue()	185
7.32.2.8 MGPIox_vSetResetAtomic()	186
7.32.2.9 MGPIox_vSetAlternateFunctionON()	186
7.32.2.10 MGPIox_vSetPortConfigLock()	186
7.32.2.11 MGPIox_vInit()	188
7.32.2.12 MGPIox_vTogglePinValue()	188
7.33 GPIO_interface.h	188
7.34 COTS/MCAL/GPIO(GPIO_private.h File Reference	190
7.34.1 Detailed Description	191
7.35 GPIO_private.h	191
7.36 COTS/MCAL/GPIO(GPIO_program.c File Reference	192
7.36.1 Detailed Description	192
7.36.2 Function Documentation	192
7.36.2.1 MGPIox_vLockedPins()	193
7.36.2.2 MGPIox_vSetPinMode()	193
7.36.2.3 MGPIox_vSetPinOutputType()	194
7.36.2.4 MGPIox_vSetPinOutputSpeed()	195
7.36.2.5 MGPIox_vSetPinInputPullType()	195
7.36.2.6 MGPIox_u8GetPinValue()	196
7.36.2.7 MGPIox_vSetPinValue()	196
7.36.2.8 MGPIox_vSetResetAtomic()	197
7.36.2.9 MGPIox_vSetAlternateFunctionON()	198
7.36.2.10 MGPIox_vInit()	199
7.36.2.11 MGPIox_vTogglePinValue()	199
7.37 GPIO_program.c	200
7.38 COTS/MCAL/NVIC/NVIC_config.h File Reference	205
7.39 NVIC_config.h	205
7.40 COTS/MCAL/NVIC/NVIC_interface.h File Reference	205
7.40.1 Detailed Description	210
7.40.2 Function Documentation	210
7.40.2.1 MNVIC_vEnablePeriphral()	210
7.40.2.2 MNVIC_vDisablePeriphral()	211
7.40.2.3 MNVIC_vSetPendingFlag()	211
7.40.2.4 MNVIC_vClearPendingFlag()	212

7.40.2.5 MNVIC_u8GetActive()	212
7.40.2.6 MNVIC_vSetPriorityConfig()	212
7.40.2.7 MNVIC_vSetPriority()	213
7.40.2.8 NVIC_GetPriority()	213
7.41 NVIC_interface.h	213
7.42 COTS/MCAL/NVIC/NVIC_private.h File Reference	215
7.42.1 Detailed Description	216
7.42.2 Macro Definition Documentation	216
7.42.2.1 MNVIC_STIR	216
7.42.2.2 VECTKEY_PASSWORD	216
7.43 NVIC_private.h	217
7.44 COTS/MCAL/NVIC/NVIC_program.c File Reference	217
7.44.1 Detailed Description	218
7.44.2 Macro Definition Documentation	218
7.44.2.1 REGISTER_SIZE	218
7.44.3 Function Documentation	218
7.44.3.1 MNVIC_vEnablePeriphral()	219
7.44.3.2 MNVIC_vDisablePeriphral()	219
7.44.3.3 MNVIC_vSetPendingFlag()	219
7.44.3.4 MNVIC_vClearPendingFlag()	219
7.44.3.5 MNVIC_u8GetActive()	220
7.44.3.6 MNVIC_vSetPriorityConfig()	220
7.44.3.7 MNVIC_vSetPriority()	220
7.44.3.8 NVIC_GetPriority()	221
7.45 NVIC_program.c	221
7.46 COTS/MCAL/RCC/MRCC_config.h File Reference	224
7.46.1 Detailed Description	225
7.46.2 Macro Definition Documentation	225
7.46.2.1 PLLI2S	225
7.46.2.2 PLL	225
7.46.2.3 CSS	226
7.46.2.4 HSEBYP	226
7.46.2.5 HSE_EN	226
7.46.2.6 HSI_EN	226
7.46.2.7 PLLQ	226
7.46.2.8 PLLSRC	226
7.46.2.9 PLLP	227
7.46.2.10 PLLN	227
7.46.2.11 PLLM	227
7.46.2.12 MCO2	227
7.46.2.13 MCO2PRE	227
7.46.2.14 MCO1PRE	227

7.46.2.15 I2SSRC	228
7.46.2.16 MCO1	228
7.46.2.17 RTCPRE	228
7.46.2.18 PPREG	228
7.46.2.19 PPRE1	228
7.46.2.20 HPRE	228
7.46.2.21 SWS	229
7.46.2.22 SW	229
7.46.2.23 DMA2CLK	229
7.46.2.24 DMA1CLK	229
7.46.2.25 CRCCCLK	229
7.46.2.26 GPIOHCLK	229
7.46.2.27 GPIOECLK	230
7.46.2.28 GPIODCLK	230
7.46.2.29 GPIOCCLK	230
7.46.2.30 GPIOBCLK	230
7.46.2.31 GPIOACLK	230
7.46.2.32 OTGFS	230
7.46.2.33 PWREN	231
7.46.2.34 I2C3EN	231
7.46.2.35 I2C2EN	231
7.46.2.36 I2C1EN	231
7.46.2.37 USART2EN	231
7.46.2.38 SPI3EN	231
7.46.2.39 SPI2EN	232
7.46.2.40 WWDGEN	232
7.46.2.41 TIM5EN	232
7.46.2.42 TIM4EN	232
7.46.2.43 TIM3EN	232
7.46.2.44 TIM2EN	232
7.46.2.45 TIM11EN	233
7.46.2.46 TIM10EN	233
7.46.2.47 TIM9EN	233
7.46.2.48 SYSCFGEN	233
7.46.2.49 SPI4EN	233
7.46.2.50 SPI1EN	233
7.46.2.51 SDIOEN	234
7.46.2.52 ADC1EN	234
7.46.2.53 USART6EN	234
7.46.2.54 USART1EN	234
7.46.2.55 TIM1EN	234
7.47 MRCC_config.h	235

7.48 COTS/MCAL/RCC/MRCC_interface.h File Reference	242
7.48.1 Detailed Description	244
7.48.2 Function Documentation	245
7.48.2.1 MRCC_vInit()	245
7.48.2.2 MRCC_vEnablePeriphralCLK()	249
7.48.2.3 MRCC_vDisablePeriphralCLK()	250
7.49 MRCC_interface.h	250
7.50 COTS/MCAL/RCC/MRCC_private.h File Reference	251
7.50.1 Detailed Description	256
7.50.2 Macro Definition Documentation	256
7.50.2.1 RCC_BASE_ADDRESS	256
7.50.2.2 RCC	256
7.50.2.3 RCC_CR_PLLI2SRDY	256
7.50.2.4 RCC_CR_PLLI2SON	257
7.50.2.5 RCC_CR_PLLRDY	257
7.50.2.6 RCC_CR_PLLON	257
7.50.2.7 RCC_CR_CSSON	257
7.50.2.8 RCC_CR_HSEBYP	257
7.50.2.9 RCC_CR_HSERDY	257
7.50.2.10 RCC_CR_HSEON	258
7.50.2.11 RCC_CR_HSIRDY	258
7.50.2.12 RCC_CR_HSION	258
7.50.2.13 RCC_PLLCFGR_PLLQ_b0	258
7.50.2.14 RCC_PLLCFGR_PLLQ_b1	258
7.50.2.15 RCC_PLLCFGR_PLLQ_b2	258
7.50.2.16 RCC_PLLCFGR_PLLQ_b3	259
7.50.2.17 RCC_PLLCFGR_PLLSRC	259
7.50.2.18 RCC_PLLCFGR_PLLP_b0	259
7.50.2.19 RCC_PLLCFGR_PLLP_b1	259
7.50.2.20 RCC_PLLCFGR_PLLN_b0	259
7.50.2.21 RCC_PLLCFGR_PLLN_b1	259
7.50.2.22 RCC_PLLCFGR_PLLN_b2	260
7.50.2.23 RCC_PLLCFGR_PLLN_b3	260
7.50.2.24 RCC_PLLCFGR_PLLN_b4	260
7.50.2.25 RCC_PLLCFGR_PLLN_b5	260
7.50.2.26 RCC_PLLCFGR_PLLN_b6	260
7.50.2.27 RCC_PLLCFGR_PLLN_b7	260
7.50.2.28 RCC_PLLCFGR_PLLN_b8	261
7.50.2.29 RCC_PLLCFGR_PLLM_b0	261
7.50.2.30 RCC_PLLCFGR_PLLM_b1	261
7.50.2.31 RCC_PLLCFGR_PLLM_b2	261
7.50.2.32 RCC_PLLCFGR_PLLM_b3	261

7.50.2.33 RCC_PLLCFGR_PLLM_b4	261
7.50.2.34 RCC_PLLCFGR_PLLM_b5	262
7.50.2.35 RCC_CFGR_MOC2_b0	262
7.50.2.36 RCC_CFGR_MOC2_b1	262
7.50.2.37 RCC_CFGR_MOC2PRE_b0	262
7.50.2.38 RCC_CFGR_MOC2PRE_b1	262
7.50.2.39 RCC_CFGR_MOC2PRE_b2	262
7.50.2.40 RCC_CFGR_MOC1PRE_b0	263
7.50.2.41 RCC_CFGR_MOC1PRE_b1	263
7.50.2.42 RCC_CFGR_MOC1PRE_b2	263
7.50.2.43 RCC_CFGR_I2SSRC	263
7.50.2.44 RCC_CFGR_MOC1_b0	263
7.50.2.45 RCC_CFGR_MOC1_b1	263
7.50.2.46 RCC_CFGR_RTCPRE_b0	264
7.50.2.47 RCC_CFGR_RTCPRE_b1	264
7.50.2.48 RCC_CFGR_RTCPRE_b2	264
7.50.2.49 RCC_CFGR_RTCPRE_b3	264
7.50.2.50 RCC_CFGR_RTCPRE_b4	264
7.50.2.51 RCC_CFGR_PPREGROUP1_b0	264
7.50.2.52 RCC_CFGR_PPREGROUP1_b1	265
7.50.2.53 RCC_CFGR_PPREGROUP1_b2	265
7.50.2.54 RCC_CFGR_PPREGROUP1_b3	265
7.50.2.55 RCC_CFGR_PPREGROUP1_b4	265
7.50.2.56 RCC_CFGR_PPREGROUP1_b5	265
7.50.2.57 RCC_CFGR_HPRE_b0	265
7.50.2.58 RCC_CFGR_HPRE_b1	266
7.50.2.59 RCC_CFGR_HPRE_b2	266
7.50.2.60 RCC_CFGR_HPRE_b3	266
7.50.2.61 RCC_CFGR_SWS_b0	266
7.50.2.62 RCC_CFGR_SWS_b1	266
7.50.2.63 RCC_CFGR_SW_b0	266
7.50.2.64 RCC_CFGR_SW_b1	267
7.50.2.65 RCC_AHB1ENR_DMA2EN	267
7.50.2.66 RCC_AHB1ENR_DMA1EN	267
7.50.2.67 RCC_AHB1ENR_CRCEN	267
7.50.2.68 RCC_AHB1ENR_GPIOHEN	267
7.50.2.69 RCC_AHB1ENR_GPIOEEN	267
7.50.2.70 RCC_AHB1ENR_GPIODEN	268
7.50.2.71 RCC_AHB1ENR_GPIOCEN	268
7.50.2.72 RCC_AHB1ENR_GPIOBEN	268
7.50.2.73 RCC_AHB1ENR_GPIOAEN	268
7.50.2.74 RCC_AHB2ENR_OTGFSEN	268

7.50.2.75 RCC_APB1ENR_PWREN	268
7.50.2.76 RCC_APB1ENR_I2C3EN	269
7.50.2.77 RCC_APB1ENR_I2C2EN	269
7.50.2.78 RCC_APB1ENR_I2C1EN	269
7.50.2.79 RCC_APB1ENR_USART2EN	269
7.50.2.80 RCC_APB1ENR_SPI3EN	269
7.50.2.81 RCC_APB1ENR_SPI2EN	269
7.50.2.82 RCC_APB1ENR_WWDGEN	270
7.50.2.83 RCC_APB1ENR_TIM5EN	270
7.50.2.84 RCC_APB1ENR_TIM4EN	270
7.50.2.85 RCC_APB1ENR_TIM3EN	270
7.50.2.86 RCC_APB1ENR_TIM2EN	270
7.50.2.87 RCC_APB2ENR_TIM11EN	270
7.50.2.88 RCC_APB2ENR_TIM10EN	271
7.50.2.89 RCC_APB2ENR_TIM9EN	271
7.50.2.90 RCC_APB2ENR_SYSCFGEN	271
7.50.2.91 RCC_APB2ENR_SPI4EN	271
7.50.2.92 RCC_APB2ENR_SPI1EN	271
7.50.2.93 RCC_APB2ENR_SDIOEN	271
7.50.2.94 RCC_APB2ENR_ADC1EN	272
7.50.2.95 RCC_APB2ENR_USART6EN	272
7.50.2.96 RCC_APB2ENR_USART1EN	272
7.50.2.97 RCC_APB2ENR_TIM1EN	272
7.50.2.98 RCC_AHB1LPENR_FLITFLPEN	272
7.50.2.99 ENABLE	272
7.50.2.100 DISABLE	273
7.50.2.101 BYBASED	273
7.50.2.102 NOTBYBASED	273
7.50.2.103 SYSCLK	273
7.50.2.104 PLLI2SCLK	273
7.50.2.105 HSE	273
7.50.2.106 PLLCLK [1/2]	274
7.50.2.107 NoDivision	274
7.50.2.108 DivisionBy2	274
7.50.2.109 DivisionBy3	274
7.50.2.110 DivisionBy4	274
7.50.2.111 DivisionBy5	274
7.50.2.112 I2S_CKIN	275
7.50.2.113 HSI	275
7.50.2.114 LSE	275
7.50.2.115 PLLCLK [2/2]	275
7.50.2.116 SYSCLKby2	275

7.50.2.117 SYSCLKby4	275
7.50.2.118 SYSCLKby8	276
7.50.2.119 SYSCLKby16	276
7.50.2.120 SYSCLKby64	276
7.50.2.121 SYSCLKby128	276
7.50.2.122 SYSCLKby256	276
7.50.2.123 SYSCLKby512	276
7.50.2.124 NoCLK0	277
7.50.2.125 NoCLK1	277
7.50.2.126 HSEby2	277
7.50.2.127 HSEby3	277
7.50.2.128 HSEby4	277
7.50.2.129 HSEby5	277
7.50.2.130 HSEby6	278
7.50.2.131 HSEby7	278
7.50.2.132 HSEby8	278
7.50.2.133 HSEby9	278
7.50.2.134 HSEby10	278
7.50.2.135 HSEby11	278
7.50.2.136 HSEby12	279
7.50.2.137 HSEby13	279
7.50.2.138 HSEby14	279
7.50.2.139 HSEby15	279
7.50.2.140 HSEby16	279
7.50.2.141 HSEby17	279
7.50.2.142 HSEby18	280
7.50.2.143 HSEby19	280
7.50.2.144 HSEby20	280
7.50.2.145 HSEby21	280
7.50.2.146 HSEby22	280
7.50.2.147 HSEby23	280
7.50.2.148 HSEby24	281
7.50.2.149 HSEby25	281
7.50.2.150 HSEby26	281
7.50.2.151 HSEby27	281
7.50.2.152 HSEby28	281
7.50.2.153 HSEby29	281
7.50.2.154 HSEby30	282
7.50.2.155 HSEby31	282
7.50.2.156 AHBby2	282
7.50.2.157 AHBby4	282
7.50.2.158 AHBby8	282

7.50.2.159 AHBby16	282
7.50.2.160 Equal_0	283
7.50.2.161 Equal_1	283
7.50.2.162 Equal_2	283
7.50.2.163 Equal_3	283
7.50.2.164 Equal_4	283
7.50.2.165 Equal_5	283
7.50.2.166 Equal_6	284
7.50.2.167 Equal_7	284
7.50.2.168 Equal_8	284
7.50.2.169 Equal_9	284
7.50.2.170 Equal_10	284
7.50.2.171 Equal_11	284
7.50.2.172 Equal_12	285
7.50.2.173 Equal_13	285
7.50.2.174 Equal_14	285
7.50.2.175 Equal_15	285
7.50.2.176 Equal_16	285
7.50.2.177 Equal_17	285
7.50.2.178 Equal_18	286
7.50.2.179 Equal_19	286
7.50.2.180 Equal_20	286
7.50.2.181 Equal_21	286
7.50.2.182 Equal_22	286
7.50.2.183 Equal_23	286
7.50.2.184 Equal_24	287
7.50.2.185 Equal_25	287
7.50.2.186 Equal_26	287
7.50.2.187 Equal_27	287
7.50.2.188 Equal_28	287
7.50.2.189 Equal_29	287
7.50.2.190 Equal_30	288
7.50.2.191 Equal_31	288
7.50.2.192 Equal_32	288
7.50.2.193 Equal_33	288
7.50.2.194 Equal_34	288
7.50.2.195 Equal_35	288
7.50.2.196 Equal_36	289
7.50.2.197 Equal_37	289
7.50.2.198 Equal_38	289
7.50.2.199 Equal_39	289
7.50.2.200 Equal_40	289

7.50.2.201 Equal_41	289
7.50.2.202 Equal_42	290
7.50.2.203 Equal_43	290
7.50.2.204 Equal_44	290
7.50.2.205 Equal_45	290
7.50.2.206 Equal_46	290
7.50.2.207 Equal_47	290
7.50.2.208 Equal_48	291
7.50.2.209 Equal_49	291
7.50.2.210 Equal_50	291
7.50.2.211 Equal_51	291
7.50.2.212 Equal_52	291
7.50.2.213 Equal_53	291
7.50.2.214 Equal_54	292
7.50.2.215 Equal_55	292
7.50.2.216 Equal_56	292
7.50.2.217 Equal_57	292
7.50.2.218 Equal_58	292
7.50.2.219 Equal_59	292
7.50.2.220 Equal_60	293
7.50.2.221 Equal_61	293
7.50.2.222 Equal_62	293
7.50.2.223 Equal_63	293
7.51 MRCC_private.h	293
7.52 COTS/MCAL/RCC/MRCC_program.c File Reference	297
7.52.1 Detailed Description	298
7.52.2 Function Documentation	298
7.52.2.1 MRCC_vInit()	298
7.52.2.2 MRCC_vEnablePeriphralCLK()	303
7.52.2.3 MRCC_vDisablePeriphralCLK()	304
7.53 MRCC_program.c	304
7.54 COTS/MCAL/SysTick/SysTick_config.h File Reference	310
7.54.1 Macro Definition Documentation	310
7.54.1.1 Exception_Request	311
7.54.1.2 MAX_TICKS	311
7.55 SysTick_config.h	311
7.56 COTS/MCAL/SysTick/SysTick_interface.h File Reference	311
7.56.1 Detailed Description	313
7.56.2 Macro Definition Documentation	313
7.56.2.1 BUSY_TICK_TIME	313
7.56.2.2 SINGLE_INTERVAL_TICK_TIME	313
7.56.2.3 PERIODIC_INTERVAL_TICK_TIME	313

7.56.3 Function Documentation	314
7.56.3.1 MSysTick_vInit()	314
7.56.3.2 MSysTick_vSetBusyWait()	314
7.56.3.3 MSysTick_vDelay()	314
7.56.3.4 MSysTick_vDelayMicroSec()	315
7.56.3.5 MSysTick_vDelayMilliSec()	315
7.56.3.6 MSysTick_vDelaySec()	315
7.56.3.7 MSysTick_vSetSingleInterval()	316
7.56.3.8 MSysTick_vSetPeriodicInterval()	316
7.56.3.9 MSysTick_vStopInterval()	316
7.56.3.10 MSysTick_u32GetElapsedTime()	317
7.56.3.11 MSysTick_u32GetRemainingTime()	317
7.56.3.12 MSysTick_vEnable()	318
7.56.3.13 MSysTick_vDisable()	318
7.56.3.14 MSysTick_vEnableException()	318
7.57 SysTick_interface.h	319
7.58 COTS/MCAL/SysTick/SysTick_private.h File Reference	319
7.58.1 Detailed Description	320
7.59 SysTick_private.h	320
7.60 COTS/MCAL/SysTick/SysTick_program.c File Reference	321
7.60.1 Detailed Description	321
7.60.2 Function Documentation	322
7.60.2.1 MSysTick_vEnableException()	322
7.60.2.2 MSysTick_vInit()	322
7.60.2.3 MSysTick_vSetBusyWait()	323
7.60.2.4 MSysTick_vDelay()	323
7.60.2.5 MSysTick_vDelayMicroSec()	324
7.60.2.6 MSysTick_vDelayMilliSec()	324
7.60.2.7 MSysTick_vDelaySec()	325
7.60.2.8 MSysTick_vSetSingleInterval()	326
7.60.2.9 MSysTick_vSetPeriodicInterval()	326
7.60.2.10 MSysTick_vStopInterval()	327
7.60.2.11 MSysTick_u32GetElapsedTime()	327
7.60.2.12 MSysTick_u32GetRemainingTime()	328
7.60.2.13 MSysTick_vEnable()	328
7.60.2.14 MSysTick_vDisable()	328
7.60.2.15 SysTick_Handler()	329
7.61 SysTick_program.c	329
Index	335

Chapter 1

ADAS Graduation Project

1.1 Description

The graduation project is ADAS and consists of 3 systems:

1. Adaptive Cruise Control
2. Adaptive Light Control
3. Driver Health Care

1.1.1 Adaptive Cruise Control

Adaptive cruise control (ACC) is an enhancement of conventional cruise control. ACC automatically adjusts the speed of your car to match the speed of the car in front of you. If the car ahead slows down, ACC can automatically match it.

1.1.2 Adaptive Light Control

Adaptive light control is a system that changes the brightness of the car's headlights automatically based on the environment's light as well as the cars that are coming in the opposite way. The lighter the environment, the darker the headlights.

1.1.3 Driver Health Care

Driver health care is a system that is responsible for the detection of the driver's sleepiness and give the suitable alters to warn the driver as well as the other cars in real-time using Image Processing and Machine Learning techniques.

1.2 Hardware specifications

This project is developed using:

- Microcontroller: STM32F401CCU6

Chapter 2

Module Index

2.1 Modules

Here is a list of all modules:

Bit Manipulation Math Macros	9
Bit Group Manipulation Math Macros	10
Compiler standard macros	12
Utilities macros	15
Standard types	15
Standard values	18
Interrupt line IDs	21
Interrupt trigger status	26
EXTI Memory Addresses	27
EXTI Registers	27
EXTI Lines Status	28
EXTI Line Settings	28
SYSCFG Memory Addresses	29
SYSCFG Memory Registers	29
GPIO Modes	30
GPIO Ports	31
GPIO Output Types	32
GPIO PIN Speed	33
GPIO Pull Types	34
GPIO Output Values	35
GPIO Output PINs	36
GPIO Alternate Functions	41
GPIO Addresses	45
GPIO Registers	46
Interrupt priority grouping	47
Group priorities	51
Sub-Group priorities	57
NVIC Vector Table	62
NVIC Addresses	78
NVIC Registers	79
NVIC Settable Priorities	80
ID options	82
ID options	84
Systick Interval Mode Configuration	94
Delay Units	94

Systick Addresses	95
Systick Registers	96
Systick Register Bits Positions	96
Systick Clock Sources	98
Systick Exception (Interrupt) Status	98

Chapter 3

Data Structure Index

3.1 Data Structures

Here are the data structures with brief descriptions:

DCM_MotorConfiguration	DC Motor configuration structure for motor initialization DC Motor configuration for motor initialization for a certain port and pin. Since DC Motors have 2 sides to be controller, we will initialize each motor with 2 pins. Taking into consideration that the pins could be on different ports	101
EXTI_ConfigType	Interrupt configuration structure to initialize the interrupt with	103
EXTI_Type	EXTI Configuration structure for interrupt initialization process	104
GPIOx_MemoryMapType	GPIO declaration structure for its registers	106
MGPIOx_ConfigType	MDIO Configuration structure for a specific PIN initialization	109
MSYSCFG_MemMap_t	System configuration structure to initialize the SYSCFG module with	111
NVIC_MemoryMapType	NVIC configuration structure for NVIC memory map	113
RCC_MemoryMapType	RCC declaration structure for its registers	117
SCB_MemoryMapType	System control block memory map structure	125
SysTick_Type	Systick memory map structure declaration for the Systick's registers	131

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

COTS/HAL/DCMOTOR/ DCM_config.h	This file contains the configuration information for the DC Motor module	133
COTS/HAL/DCMOTOR/ DCM_interface.h	This file contains the interfacing information for the DC Motor module	134
COTS/HAL/DCMOTOR/ DCM_private.h	This file contains the private information for the DC Motor module	136
COTS/HAL/DCMOTOR/ DCM_program.c	This file contains the source code of the interfacing information for the DC Motor module	137
COTS/LIB/LSTD_BITMATH.h	This file contains the bit math manipulation macro-functions	140
COTS/LIB/LSTD_COMPILER.h	This file contains the compiler standard macros	141
COTS/LIB/LSTD MCU UTILITIES.h	This file contains the MCU utility macro-functions	142
COTS/LIB/LSTD TYPES.h	This file contains the standard types	142
COTS/LIB/LSTD VALUES.h	This file contains the standard values	143
COTS/MCAL/EXTI/ EXTI_config.h	This file contains the interfacing information for the EXTI module	145
COTS/MCAL/EXTI/ EXTI_interface.h	This file contains the interfacing information for the EXTI module	154
COTS/MCAL/EXTI/ EXTI_private.h	This file contains the private information related to the EXTI module	161
COTS/MCAL/EXTI/ EXTI_program.c	This file contains the source code of the interfacing for the EXTI module	163
COTS/MCAL/EXTI/ SYSCFG_private.h	This file contains the private information regarding the SYSCFG	177
COTS/MCAL/GPIO/ GPIO_config.h	This file contains the GPIO configurations	178
COTS/MCAL/GPIO/ GPIO_interface.h	This file contains the interfacing information for the GPIO module	180
COTS/MCAL/GPIO/ GPIO_private.h	This file contains the registers information and addresses for the GPIO module	190
COTS/MCAL/GPIO/ GPIO_program.c	This file contains the source code of the interfacing for the GPIO modules	192

COTS/MCAL/NVIC/ NVIC_config.h	205
COTS/MCAL/NVIC/ NVIC_interface.h	
This file contains the interface information and addresses for the NVIC module	205
COTS/MCAL/NVIC/ NVIC_private.h	
This file contains the registers information and addresses for the NVIC module	215
COTS/MCAL/NVIC/ NVIC_program.c	
This file contains the source code of the interfacing for the NVIC modules	217
COTS/MCAL/RCC/ MRCC_config.h	
This file contains the RCC configurations	224
COTS/MCAL/RCC/ MRCC_interface.h	
This file contains the interfacing information for the RCC module	242
COTS/MCAL/RCC/ MRCC_private.h	
This file contains the registers information and addresses for the RCC module	251
COTS/MCAL/RCC/ MRCC_program.c	
This file contains the source code of the interfacing for the RCC module	297
COTS/MCAL/SysTick/ SysTick_config.h	310
COTS/MCAL/SysTick/ SysTick_interface.h	
This file contains the interfacing information for the Systick module	311
COTS/MCAL/SysTick/ SysTick_private.h	
This file contains the registers information and addresses for the Systick module	319
COTS/MCAL/SysTick/ SysTick_program.c	
This file contains the source code of the interfacing for the Systick modules	321

Chapter 5

Module Documentation

5.1 Bit Manipulation Math Macros

Macros

- #define [SET_BIT](#)(Reg, bitnum) (Reg) |= (1 << (bitnum))
- #define [CLR_BIT](#)(Reg, bitnum) (Reg) &= ~(1 << (bitnum))
- #define [TOGGLE_BIT](#)(Reg, bitnum) (Reg) ^= (1 << (bitnum))
- #define [GET_BIT](#)(Reg, bitnum) (((Reg)>>(bitnum)) & 1)

5.1.1 Detailed Description

5.1.2 Macro Definition Documentation

5.1.2.1 SET_BIT

```
#define SET_BIT(
    Reg,
    bitnum ) (Reg) |= (1 << (bitnum))

#include <COTS/LIB/LSTD\_BITMATH.h>
```

Sets a certain bit's value

Definition at line [23](#) of file [LSTD_BITMATH.h](#).

5.1.2.2 CLR_BIT

```
#define CLR_BIT(
    Reg,
    bitnum ) (Reg) &= ~(1 << (bitnum))

#include <COTS/LIB/LSTD_BITMATH.h>
```

Clears a certain bit's value to

Definition at line 30 of file [LSTD_BITMATH.h](#).

5.1.2.3 TOGGLE_BIT

```
#define TOGGLE_BIT(
    Reg,
    bitnum ) (Reg) ^= (1 << (bitnum))

#include <COTS/LIB/LSTD_BITMATH.h>
```

Toggle a bit to 0 if it's 1, 1 otherwise

Definition at line 37 of file [LSTD_BITMATH.h](#).

5.1.2.4 GET_BIT

```
#define GET_BIT(
    Reg,
    bitnum ) (((Reg)>>(bitnum)) & 1)

#include <COTS/LIB/LSTD_BITMATH.h>
```

Returns the value of the bit whether it's 1 or 0

Definition at line 44 of file [LSTD_BITMATH.h](#).

5.2 Bit Group Manipulation Math Macros

Macros

- #define [SET_BITs](#)(Reg, bits, bitnum, factor) (Reg) |= ((bits) << (bitnum * factor))
- #define [CLR_BITs](#)(Reg, bits, bitnum, factor) (Reg) &= ~((bits) << (bitnum * factor))
- #define [TOGGLE_BITs](#)(Reg, bits, bitnum, factor) (Reg) ^= ((bits) << (bitnum * factor))
- #define [GET_BITs](#)(Reg, bits, bitnum, factor) (((Reg) >> (bitnum * factor)) & (bits))

5.2.1 Detailed Description

5.2.2 Macro Definition Documentation

5.2.2.1 SET_BITs

```
#define SET_BITs(
    Reg,
    bits,
    bitnum,
    factor ) (Reg) |= ((bits) << (bitnum * factor))

#include <COTS/LIB/LSTD_BITMATH.h>
```

Sets the values of a group of bits

Definition at line 59 of file [LSTD_BITMATH.h](#).

5.2.2.2 CLR_BITs

```
#define CLR_BITs(
    Reg,
    bits,
    bitnum,
    factor ) (Reg) &= ~((bits) << (bitnum * factor))

#include <COTS/LIB/LSTD_BITMATH.h>
```

Clears the value of a group of bits

Definition at line 66 of file [LSTD_BITMATH.h](#).

5.2.2.3 TOGGLE_BITs

```
#define TOGGLE_BITs(
    Reg,
    bits,
    bitnum,
    factor ) (Reg) ^= ((bits) << (bitnum * factor))

#include <COTS/LIB/LSTD_BITMATH.h>
```

Toggles the value of a group of bits

Definition at line 73 of file [LSTD_BITMATH.h](#).

5.2.2.4 GET_BITS

```
#define GET_BITS(
    Reg,
    bits,
    bitnum,
    factor ) (((Reg) >> (bitnum * factor)) & (bits))

#include <COTS/LIB/LSTD_BITMATH.h>
```

Returns the value of a group of bits

Definition at line 80 of file [LSTD_BITMATH.h](#).

5.3 Compiler standard macros

Macros

- #define VAR(vartype) vartype
- #define FUNC(rettype) rettype
- #define P2VAR(ptrtype) ptrtype *
- #define P2CONST(ptrtype) const ptrtype *
- #define CONSTP2VAR(ptrtype) ptrtype * const
- #define CONSTP2CONST(ptrtype) const ptrtype * const
- #define P2FUNC(rettype, fctname) rettype (*fctname)
- #define CONST(consttype) const consttype
- #define STATIC static

5.3.1 Detailed Description

5.3.2 Macro Definition Documentation

5.3.2.1 VAR

```
#define VAR(
    vartype ) vartype

#include <COTS/LIB/LSTD_COMPILER.h>
```

Declare a variable with the specified type

Definition at line 23 of file [LSTD_COMPILER.h](#).

5.3.2.2 FUNC

```
#define FUNC(
    rettype ) rettype

#include <COTS/LIB/LSTD_COMPILER.h>
```

Declare a function's return type

Definition at line 30 of file [LSTD_COMPILER.h](#).

5.3.2.3 P2VAR

```
#define P2VAR(
    ptrtype ) ptrtype *

#include <COTS/LIB/LSTD_COMPILER.h>
```

Declare a pointer-to-variable with the specified type

Definition at line 37 of file [LSTD_COMPILER.h](#).

5.3.2.4 P2CONST

```
#define P2CONST(
    ptrtype ) const ptrtype *

#include <COTS/LIB/LSTD_COMPILER.h>
```

Declare a constant pointer-to-variable with the specified type

Definition at line 44 of file [LSTD_COMPILER.h](#).

5.3.2.5 CONSTP2VAR

```
#define CONSTP2VAR(
    ptrtype ) ptrtype * const

#include <COTS/LIB/LSTD_COMPILER.h>
```

Declare a pointer-to-variable constant with the specified type

Definition at line 51 of file [LSTD_COMPILER.h](#).

5.3.2.6 CONSTP2CONST

```
#define CONSTP2CONST(  
    ptrtype ) const ptrtype * const  
  
#include <COTS/LIB/LSTD_COMPILER.h>
```

Declare a constant pointer-to-variable constant with the specified type

Definition at line 58 of file [LSTD_COMPILER.h](#).

5.3.2.7 P2FUNC

```
#define P2FUNC(  
    rettype,  
    fctname ) rettype (*fctname)  
  
#include <COTS/LIB/LSTD_COMPILER.h>
```

Declare a pointer-to-function with the specified type

Definition at line 65 of file [LSTD_COMPILER.h](#).

5.3.2.8 CONST

```
#define CONST(  
    consttype ) const consttype  
  
#include <COTS/LIB/LSTD_COMPILER.h>
```

Declare a standard constant variable with the specified type

Definition at line 73 of file [LSTD_COMPILER.h](#).

5.3.2.9 STATIC

```
#define STATIC static  
  
#include <COTS/LIB/LSTD_COMPILER.h>
```

Declare a standard static variable

Definition at line 82 of file [LSTD_COMPILER.h](#).

5.4 Utilities macros

Macros

- #define MY_MS_DELAY(T) do{ u32 Timer = (T * 500); while (Timer--) { asm ("nop"); } } while(0);

5.4.1 Detailed Description

5.4.2 Macro Definition Documentation

5.4.2.1 MY_MS_DELAY

```
#define MY_MS_DELAY(  
    T ) do{ u32 Timer = (T * 500); while (Timer--) { asm ("nop"); } } while(0);  
  
#include <COTS/LIB/LSTD MCU UTILITIES.h>
```

Declare a delay with a T microseconds with a NOP delay

Definition at line 23 of file [LSTD MCU UTILITIES.h](#).

5.5 Standard types

Typedefs

- typedef unsigned char [bool_t](#)
- typedef unsigned char [u8_t](#)
- typedef unsigned short int [u16_t](#)
- typedef unsigned long int [u32_t](#)
- typedef signed char [s8_t](#)
- typedef signed short int [s16_t](#)
- typedef signed long int [s32_t](#)
- typedef float [f32_t](#)
- typedef double [f64_t](#)

5.5.1 Detailed Description

5.5.2 Typedef Documentation

5.5.2.1 bool_t

```
bool_t  
  
#include <COTS/LIB/LSTD_TYPES.h>
```

Type definition for boolean

Definition at line 23 of file [LSTD_TYPES.h](#).

5.5.2.2 u8_t

```
u8_t  
  
#include <COTS/LIB/LSTD_TYPES.h>
```

Type definition for 8-bit unsigned INT

Definition at line 30 of file [LSTD_TYPES.h](#).

5.5.2.3 u16_t

```
u16_t  
  
#include <COTS/LIB/LSTD_TYPES.h>
```

Type definition for 16-bit unsigned INT

Definition at line 37 of file [LSTD_TYPES.h](#).

5.5.2.4 u32_t

```
u32_t  
  
#include <COTS/LIB/LSTD_TYPES.h>
```

Type definition for 32-bit unsigned INT

Definition at line 44 of file [LSTD_TYPES.h](#).

5.5.2.5 s8_t

s8_t

```
#include <COTS/LIB/LSTD_TYPES.h>
```

Type definition for 8-bit signed INT

Definition at line 51 of file [LSTD_TYPES.h](#).

5.5.2.6 s16_t

s16_t

```
#include <COTS/LIB/LSTD_TYPES.h>
```

Type definition for 16-bit signed INT

Definition at line 58 of file [LSTD_TYPES.h](#).

5.5.2.7 s32_t

s32_t

```
#include <COTS/LIB/LSTD_TYPES.h>
```

Type definition for 32-bit signed INT

Definition at line 65 of file [LSTD_TYPES.h](#).

5.5.2.8 fl32_t

fl32_t

```
#include <COTS/LIB/LSTD_TYPES.h>
```

Type definition for 32-bit float

Definition at line 72 of file [LSTD_TYPES.h](#).

5.5.2.9 fl64_t

```
fl64_t  
#include <COTS/LIB/LSTD_TYPES.h>
```

Type definition for 64-bit float

Definition at line 79 of file [LSTD_TYPES.h](#).

5.6 Standard values

Macros

- #define **TRUE** (1)
- #define **FALSE** (0)
- #define **NULL** ((P2VAR(void))0)
- #define **INITIAL_ZERO** (0)
- #define **FLAG_SET** (1)
- #define **FLAG_CLEARED** (0)
- #define **RUN** (1)
- #define **STOP** (0)
- #define **PRESSED** (1)
- #define **RELEASED** (0)

5.6.1 Detailed Description

5.6.2 Macro Definition Documentation

5.6.2.1 TRUE

```
#define TRUE (1)  
  
#include <COTS/LIB/LSTD_VALUES.h>
```

Type definition for TRUE

Definition at line 24 of file [LSTD_VALUES.h](#).

5.6.2.2 FALSE

```
#define FALSE (0)

#include <COTS/LIB/LSTD_VALUES.h>
```

Type definition for FALSE

Definition at line 33 of file [LSTD_VALUES.h](#).

5.6.2.3 NULL

```
#define NULL ((P2VAR(void))0)

#include <COTS/LIB/LSTD_VALUES.h>
```

Type definition for NULL

Definition at line 43 of file [LSTD_VALUES.h](#).

5.6.2.4 INITIAL_ZERO

```
#define INITIAL_ZERO (0)

#include <COTS/LIB/LSTD_VALUES.h>
```

Type definition for INITIAL_ZERO

Definition at line 57 of file [LSTD_VALUES.h](#).

5.6.2.5 FLAG_SET

```
#define FLAG_SET (1)

#include <COTS/LIB/LSTD_VALUES.h>
```

Type definition for FLAG_SET

Definition at line 66 of file [LSTD_VALUES.h](#).

5.6.2.6 FLAG_CLEARED

```
#define FLAG_CLEARED (0)

#include <COTS/LIB/LSTD_VALUES.h>
```

Type definition for FLAG_CLEARED

Definition at line 75 of file [LSTD_VALUES.h](#).

5.6.2.7 RUN

```
#define RUN (1)

#include <COTS/LIB/LSTD_VALUES.h>
```

Type definition for RUN

Definition at line 84 of file [LSTD_VALUES.h](#).

5.6.2.8 STOP

```
#define STOP (0)

#include <COTS/LIB/LSTD_VALUES.h>
```

Type definition for STOP

Definition at line 94 of file [LSTD_VALUES.h](#).

5.6.2.9 PRESSED

```
#define PRESSED (1)

#include <COTS/LIB/LSTD_VALUES.h>
```

Type definition for PRESSED

Definition at line 103 of file [LSTD_VALUES.h](#).

5.6.2.10 RELEASED

```
#define RELEASED (0)

#include <COTS/LIB/LSTD_VALUES.h>
```

Type definition for RELEASED

Definition at line 113 of file [LSTD_VALUES.h](#).

5.7 Interrupt line IDs

Macros

- #define EXTI_LINE0 (0)
Line 0.
- #define EXTI_LINE1 (1)
Line 1.
- #define EXTI_LINE2 (2)
Line 2.
- #define EXTI_LINE3 (3)
Line 3.
- #define EXTI_LINE4 (4)
Line 4.
- #define EXTI_LINE5 (5)
Line 5.
- #define EXTI_LINE6 (6)
Line 6.
- #define EXTI_LINE7 (7)
Line 7.
- #define EXTI_LINE8 (8)
Line 8.
- #define EXTI_LINE9 (9)
Line 9.
- #define EXTI_LINE10 (10)
Line 10.
- #define EXTI_LINE11 (11)
Line 11.
- #define EXTI_LINE12 (12)
Line 12.
- #define EXTI_LINE13 (13)
Line 13.
- #define EXTI_LINE14 (14)
Line 14.
- #define EXTI_LINE15 (15)
Line 15.

5.7.1 Detailed Description

5.7.2 Macro Definition Documentation

5.7.2.1 EXTI_LINE0

```
#define EXTI_LINE0 (0)  
  
#include <COTS/MCAL/EXTI/EXTI_interface.h>
```

Line 0.

Definition at line 104 of file [EXTI_interface.h](#).

5.7.2.2 EXTI_LINE1

```
#define EXTI_LINE1 (1)  
  
#include <COTS/MCAL/EXTI/EXTI_interface.h>
```

Line 1.

Definition at line 110 of file [EXTI_interface.h](#).

5.7.2.3 EXTI_LINE2

```
#define EXTI_LINE2 (2)  
  
#include <COTS/MCAL/EXTI/EXTI_interface.h>
```

Line 2.

Definition at line 116 of file [EXTI_interface.h](#).

5.7.2.4 EXTI_LINE3

```
#define EXTI_LINE3 (3)  
  
#include <COTS/MCAL/EXTI/EXTI_interface.h>
```

Line 3.

Definition at line 122 of file [EXTI_interface.h](#).

5.7.2.5 EXTI_LINE4

```
#define EXTI_LINE4 (4)

#include <COTS/MCAL/EXTI/EXTI_interface.h>
```

Line 4.

Definition at line 128 of file [EXTI_interface.h](#).

5.7.2.6 EXTI_LINE5

```
#define EXTI_LINE5 (5)

#include <COTS/MCAL/EXTI/EXTI_interface.h>
```

Line 5.

Definition at line 134 of file [EXTI_interface.h](#).

5.7.2.7 EXTI_LINE6

```
#define EXTI_LINE6 (6)

#include <COTS/MCAL/EXTI/EXTI_interface.h>
```

Line 6.

Definition at line 140 of file [EXTI_interface.h](#).

5.7.2.8 EXTI_LINE7

```
#define EXTI_LINE7 (7)

#include <COTS/MCAL/EXTI/EXTI_interface.h>
```

Line 7.

Definition at line 146 of file [EXTI_interface.h](#).

5.7.2.9 EXTI_LINE8

```
#define EXTI_LINE8 (8)

#include <COTS/MCAL/EXTI/EXTI_interface.h>
```

Line 8.

Definition at line 152 of file [EXTI_interface.h](#).

5.7.2.10 EXTI_LINE9

```
#define EXTI_LINE9 (9)

#include <COTS/MCAL/EXTI/EXTI_interface.h>
```

Line 9.

Definition at line 158 of file [EXTI_interface.h](#).

5.7.2.11 EXTI_LINE10

```
#define EXTI_LINE10 (10)

#include <COTS/MCAL/EXTI/EXTI_interface.h>
```

Line 10.

Definition at line 164 of file [EXTI_interface.h](#).

5.7.2.12 EXTI_LINE11

```
#define EXTI_LINE11 (11)

#include <COTS/MCAL/EXTI/EXTI_interface.h>
```

Line 11.

Definition at line 170 of file [EXTI_interface.h](#).

5.7.2.13 EXTI_LINE12

```
#define EXTI_LINE12 (12)

#include <COTS/MCAL/EXTI/EXTI_interface.h>
```

Line 12.

Definition at line 176 of file [EXTI_interface.h](#).

5.7.2.14 EXTI_LINE13

```
#define EXTI_LINE13 (13)

#include <COTS/MCAL/EXTI/EXTI_interface.h>
```

Line 13.

Definition at line 182 of file [EXTI_interface.h](#).

5.7.2.15 EXTI_LINE14

```
#define EXTI_LINE14 (14)

#include <COTS/MCAL/EXTI/EXTI_interface.h>
```

Line 14.

Definition at line 188 of file [EXTI_interface.h](#).

5.7.2.16 EXTI_LINE15

```
#define EXTI_LINE15 (15)

#include <COTS/MCAL/EXTI/EXTI_interface.h>
```

Line 15.

Definition at line 194 of file [EXTI_interface.h](#).

5.8 Interrupt trigger status

Macros

- `#define EXTI_FallingEdge (1)`
trigger interrupt on falling edge
- `#define EXTI_RisingEdge (2)`
trigger interrupt on rising edge
- `#define EXTI_OnChange (3)`
trigger interrupt on level change

5.8.1 Detailed Description

5.8.2 Macro Definition Documentation

5.8.2.1 EXTI_FallingEdge

```
#define EXTI_FallingEdge (1)

#include <COTS/MCAL/EXTI/EXTI_interface.h>

trigger interrupt on falling edge
```

Definition at line 212 of file [EXTI_interface.h](#).

5.8.2.2 EXTI_RisingEdge

```
#define EXTI_RisingEdge (2)

#include <COTS/MCAL/EXTI/EXTI_interface.h>

trigger interrupt on rising edge
```

Definition at line 218 of file [EXTI_interface.h](#).

5.8.2.3 EXTI_OnChange

```
#define EXTI_OnChange (3)

#include <COTS/MCAL/EXTI/EXTI_interface.h>

trigger interrupt on level change
```

Definition at line 224 of file [EXTI_interface.h](#).

5.9 EXTI Memory Addresses

Macros

- #define `EXTI_BASE_ADDRESS` (0x40013C00)

5.9.1 Detailed Description

5.9.2 Macro Definition Documentation

5.9.2.1 EXTI_BASE_ADDRESS

```
#define EXTI_BASE_ADDRESS (0x40013C00)

#include <COTS/MCAL/EXTI/EXTI_private.h>

EXTI base address
```

Definition at line 57 of file `EXTI_private.h`.

5.10 EXTI Registers

Macros

- #define `MEXTI` ((volatile P2VAR(EXTI_Type))(EXTI_BASE_ADDRESS))

5.10.1 Detailed Description

5.10.2 Macro Definition Documentation

5.10.2.1 MEXTI

```
#define MEXTI ((volatile P2VAR(EXTI_Type))(EXTI_BASE_ADDRESS))

#include <COTS/MCAL/EXTI/EXTI_private.h>
```

EXTI register

Definition at line 71 of file `EXTI_private.h`.

5.11 EXTI Lines Status

Macros

- #define ENABLE (1)
- #define DISABLE (2)

5.11.1 Detailed Description

5.11.2 Macro Definition Documentation

5.11.2.1 ENABLE

```
#define ENABLE (1)

#include <COTS/MCAL/EXTI/EXTI_private.h>
```

Enable a certain line ID from the config

Definition at line 85 of file [EXTI_private.h](#).

5.11.2.2 DISABLE

```
#define DISABLE (2)

#include <COTS/MCAL/EXTI/EXTI_private.h>
```

Disable a certain line ID from the config

Definition at line 91 of file [EXTI_private.h](#).

5.12 EXTI Line Settings

Macros

- #define EXTI_MAX_EXTI_NUM (22)

5.12.1 Detailed Description

5.12.2 Macro Definition Documentation

5.12.2.1 EXTI_MAX_EXTI_NUM

```
#define EXTI_MAX_EXTI_NUM (22)

#include <COTS/MCAL/EXTI/EXTI_private.h>
```

Maximum number of available interrupt lines

Definition at line 105 of file [EXTI_private.h](#).

5.13 SYSCFG Memory Addresses

Macros

- `#define SYSCFG_BASE_ADDR (0x40013800)`

5.13.1 Detailed Description

5.13.2 Macro Definition Documentation

5.13.2.1 SYSCFG_BASE_ADDR

```
#define SYSCFG_BASE_ADDR (0x40013800)

#include <COTS/MCAL/EXTI/SYSCFG_private.h>

SYSCFG base address

Definition at line 51 of file SYSCFG\_private.h.
```

5.14 SYSCFG Memory Registers

Macros

- `#define MSYSCFG ((volatile MSYSCFG_MemMap_t *) (SYSCFG_BASE_ADDR))`

5.14.1 Detailed Description

5.14.2 Macro Definition Documentation

5.14.2.1 MSYSCFG

```
#define MSYSCFG ((volatile MSYSCFG_MemMap_t *) (SYSCFG_BASE_ADDR))
```

```
#include <COTS/MCAL/EXTI/SYSCFG_private.h>
```

SYSCFG register

Definition at line 64 of file [SYSCFG_private.h](#).

5.15 GPIO Modes

Macros

- `#define GPIOx_MODE_INPUT (0b00)`
Control input mode.
- `#define GPIOx_MODE_OUTPUT (0b01)`
Control output mode.
- `#define GPIOx_MODE_AF (0b10)`
Control alternate function mode.
- `#define GPIOx_MODE_ANALOG (0b11)`
Control analog mode.

5.15.1 Detailed Description

5.15.2 Macro Definition Documentation

5.15.2.1 GPIOx_MODE_INPUT

```
#define GPIOx_MODE_INPUT (0b00)

#include <COTS/MCAL/GPIO/GPIO_interface.h>
```

Control input mode.

Definition at line 158 of file [GPIO_interface.h](#).

5.15.2.2 GPIOx_MODE_OUTPUT

```
#define GPIOx_MODE_OUTPUT (0b01)

#include <COTS/MCAL/GPIO/GPIO_interface.h>
```

Control output mode.

Definition at line 165 of file [GPIO_interface.h](#).

5.15.2.3 GPIOx_MODE_AF

```
#define GPIOx_MODE_AF (0b10)

#include <COTS/MCAL/GPIO/GPIO_interface.h>
```

Control alternate function mode.

Definition at line 172 of file [GPIO_interface.h](#).

5.15.2.4 GPIOx_MODE_ANALOG

```
#define GPIOx_MODE_ANALOG (0b11)

#include <COTS/MCAL/GPIO/GPIO_interface.h>
```

Control analog mode.

Definition at line 179 of file [GPIO_interface.h](#).

5.16 GPIO Ports

Macros

- #define [GPIO_PORTA](#) (0)
GPIO Port A.
- #define [GPIO_PORTB](#) (1)
GPIO Port B.
- #define [GPIO_PORTC](#) (2)
GPIO Port C.

5.16.1 Detailed Description

5.16.2 Macro Definition Documentation

5.16.2.1 GPIO_PORTA

```
#define GPIO_PORTA (0)

#include <COTS/MCAL/GPIO/GPIO_interface.h>

GPIO Port A.
```

Definition at line 194 of file [GPIO_interface.h](#).

5.16.2.2 GPIO_PORTB

```
#define GPIO_PORTB (1)

#include <COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO Port B.

Definition at line 201 of file [GPIO_interface.h](#).

5.16.2.3 GPIO_PORTC

```
#define GPIO_PORTC (2)

#include <COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO Port C.

Definition at line 208 of file [GPIO_interface.h](#).

5.17 GPIO Output Types

Macros

- `#define GPIOx_OPENDRAIN (1)`
GPIO open-drain.
- `#define GPIOx_PUSH_PULL (2)`
GPIO push-pull.

5.17.1 Detailed Description

5.17.2 Macro Definition Documentation

5.17.2.1 GPIOx_OPENDRAIN

```
#define GPIOx_OPENDRAIN (1)

#include <COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO open-drain.

Definition at line 223 of file [GPIO_interface.h](#).

5.17.2.2 GPIOx_PUSH_PULL

```
#define GPIOx_PUSH_PULL (2)

#include <COTS/MCAL/GPIO/GPIO_interface.h>

GPIO push-pull.
```

Definition at line 230 of file [GPIO_interface.h](#).

5.18 GPIO PIN Speed

Macros

- #define [GPIOx_LowSpeed](#) (0b00)
GPIO low speed.
- #define [GPIOx_MediumSpeed](#) (0b01)
GPIO medium speed.
- #define [GPIOx_HighSpeed](#) (0b10)
GPIO high speed.
- #define [GPIOx_VeryHighSpeed](#) (0b11)
GPIO very high speed.

5.18.1 Detailed Description

5.18.2 Macro Definition Documentation

5.18.2.1 GPIOx_LowSpeed

```
#define GPIOx_LowSpeed (0b00)

#include <COTS/MCAL/GPIO/GPIO_interface.h>

GPIO low speed.
```

Definition at line 245 of file [GPIO_interface.h](#).

5.18.2.2 GPIOx_MediumSpeed

```
#define GPIOx_MediumSpeed (0b01)

#include <COTS/MCAL/GPIO/GPIO_interface.h>

GPIO medium speed.

Definition at line 252 of file GPIO\_interface.h.
```

5.18.2.3 GPIOx_HighSpeed

```
#define GPIOx_HighSpeed (0b10)

#include <COTS/MCAL/GPIO/GPIO_interface.h>

GPIO high speed.
```

Definition at line 259 of file [GPIO_interface.h](#).

5.18.2.4 GPIOx_VeryHighSpeed

```
#define GPIOx_VeryHighSpeed (0b11)

#include <COTS/MCAL/GPIO/GPIO_interface.h>

GPIO very high speed.
```

Definition at line 266 of file [GPIO_interface.h](#).

5.19 GPIO Pull Types

Macros

- #define [GPIOx_NoPull](#) (0b00)
GPIO No PULL.
- #define [GPIOx_PullUp](#) (0b01)
GPIO Pull UP.
- #define [GPIOx_PullDown](#) (0b10)
GPIO Pull Down.

5.19.1 Detailed Description

5.19.2 Macro Definition Documentation

5.19.2.1 GPIOx_NoPull

```
#define GPIOx_NoPull (0b00)

#include <COTS/MCAL/GPIO/GPIO_interface.h>

GPIO No PULL.
```

Definition at line 281 of file [GPIO_interface.h](#).

5.19.2.2 GPIOx_PullUp

```
#define GPIOx_PullUp (0b01)

#include <COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO Pull UP.

Definition at line 288 of file [GPIO_interface.h](#).

5.19.2.3 GPIOx_PullDown

```
#define GPIOx_PullDown (0b10)

#include <COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO Pull Down.

Definition at line 295 of file [GPIO_interface.h](#).

5.20 GPIO Output Values

Macros

- #define [GPIOx_HIGH](#) (1)
GPIO output high.
- #define [GPIOx_LOW](#) (2)
GPIO output low.

5.20.1 Detailed Description

5.20.2 Macro Definition Documentation

5.20.2.1 GPIOx_HIGH

```
#define GPIOx_HIGH (1)

#include <COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO output high.

Definition at line 310 of file [GPIO_interface.h](#).

5.20.2.2 GPIOx_LOW

```
#define GPIOx_LOW (2)

#include <COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO output low.

Definition at line 317 of file [GPIO_interface.h](#).

5.21 GPIO Output PINs

Macros

- #define [GPIOx_PIN0](#) (0)
GPIO PIN 0.
- #define [GPIOx_PIN1](#) (1)
GPIO PIN 1.
- #define [GPIOx_PIN2](#) (2)
GPIO PIN 2.
- #define [GPIOx_PIN3](#) (3)
GPIO PIN 3.
- #define [GPIOx_PIN4](#) (4)
GPIO PIN 4.
- #define [GPIOx_PIN5](#) (5)
GPIO PIN 5.
- #define [GPIOx_PIN6](#) (6)
GPIO PIN 6.
- #define [GPIOx_PIN7](#) (7)
GPIO PIN 7.
- #define [GPIOx_PIN8](#) (8)
GPIO PIN 8.
- #define [GPIOx_PIN9](#) (9)
brief GPIO PIN 9
- #define [GPIOx_PIN10](#) (10)
GPIO PIN 10.
- #define [GPIOx_PIN11](#) (11)
GPIO PIN 11.
- #define [GPIOx_PIN12](#) (12)
GPIO PIN 12.
- #define [GPIOx_PIN13](#) (13)
GPIO PIN 13.
- #define [GPIOx_PIN14](#) (14)
GPIO PIN 14.
- #define [GPIOx_PIN15](#) (15)
GPIO PIN 15.

5.21.1 Detailed Description

5.21.2 Macro Definition Documentation

5.21.2.1 GPIOx_PIN0

```
#define GPIOx_PIN0 (0)  
#include <COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO PIN 0.

Definition at line 332 of file [GPIO_interface.h](#).

5.21.2.2 GPIOx_PIN1

```
#define GPIOx_PIN1 (1)  
#include <COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO PIN 1.

Definition at line 339 of file [GPIO_interface.h](#).

5.21.2.3 GPIOx_PIN2

```
#define GPIOx_PIN2 (2)  
#include <COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO PIN 2.

Definition at line 346 of file [GPIO_interface.h](#).

5.21.2.4 GPIOx_PIN3

```
#define GPIOx_PIN3 (3)  
#include <COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO PIN 3.

Definition at line 353 of file [GPIO_interface.h](#).

5.21.2.5 GPIOx_PIN4

```
#define GPIOx_PIN4 (4)

#include <COTS/MCAL/GPIO(GPIO_interface.h>
```

GPIO PIN 4.

Definition at line 360 of file [GPIO_interface.h](#).

5.21.2.6 GPIOx_PIN5

```
#define GPIOx_PIN5 (5)

#include <COTS/MCAL/GPIO(GPIO_interface.h>
```

GPIO PIN 5.

Definition at line 367 of file [GPIO_interface.h](#).

5.21.2.7 GPIOx_PIN6

```
#define GPIOx_PIN6 (6)

#include <COTS/MCAL/GPIO(GPIO_interface.h>
```

GPIO PIN 6.

Definition at line 374 of file [GPIO_interface.h](#).

5.21.2.8 GPIOx_PIN7

```
#define GPIOx_PIN7 (7)

#include <COTS/MCAL/GPIO(GPIO_interface.h>
```

GPIO PIN 7.

Definition at line 381 of file [GPIO_interface.h](#).

5.21.2.9 GPIOx_PIN8

```
#define GPIOx_PIN8 (8)

#include <COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO PIN 8.

Definition at line 388 of file [GPIO_interface.h](#).

5.21.2.10 GPIOx_PIN9

```
#define GPIOx_PIN9 (9)

#include <COTS/MCAL/GPIO/GPIO_interface.h>
```

brief GPIO PIN 9

Definition at line 395 of file [GPIO_interface.h](#).

5.21.2.11 GPIOx_PIN10

```
#define GPIOx_PIN10 (10)

#include <COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO PIN 10.

Definition at line 402 of file [GPIO_interface.h](#).

5.21.2.12 GPIOx_PIN11

```
#define GPIOx_PIN11 (11)

#include <COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO PIN 11.

Definition at line 409 of file [GPIO_interface.h](#).

5.21.2.13 GPIOx_PIN12

```
#define GPIOx_PIN12 (12)

#include <COTS/MCAL/GPIO(GPIO_interface.h>
```

GPIO PIN 12.

Definition at line 416 of file [GPIO_interface.h](#).

5.21.2.14 GPIOx_PIN13

```
#define GPIOx_PIN13 (13)

#include <COTS/MCAL/GPIO(GPIO_interface.h>
```

GPIO PIN 13.

Definition at line 423 of file [GPIO_interface.h](#).

5.21.2.15 GPIOx_PIN14

```
#define GPIOx_PIN14 (14)

#include <COTS/MCAL/GPIO(GPIO_interface.h>
```

GPIO PIN 14.

Definition at line 430 of file [GPIO_interface.h](#).

5.21.2.16 GPIOx_PIN15

```
#define GPIOx_PIN15 (15)

#include <COTS/MCAL/GPIO(GPIO_interface.h>
```

GPIO PIN 15.

Definition at line 437 of file [GPIO_interface.h](#).

5.22 GPIO Alternate Functions

Macros

- #define **GPIOx_AF0** (0)
GPIO Alternate function 0.
- #define **GPIOx_AF1** (1)
GPIO Alternate function 1.
- #define **GPIOx_AF2** (2)
GPIO Alternate function 2.
- #define **GPIOx_AF3** (3)
GPIO Alternate function 3.
- #define **GPIOx_AF4** (4)
GPIO Alternate function 4.
- #define **GPIOx_AF5** (5)
GPIO Alternate function 5.
- #define **GPIOx_AF6** (6)
GPIO Alternate function 6.
- #define **GPIOx_AF7** (7)
GPIO Alternate function 7.
- #define **GPIOx_AF8** (8)
GPIO Alternate function 8.
- #define **GPIOx_AF9** (9)
GPIO Alternate function 9.
- #define **GPIOx_AF10** (10)
GPIO Alternate function 10.
- #define **GPIOx_AF11** (11)
GPIO Alternate function 11.
- #define **GPIOx_AF12** (12)
GPIO Alternate function 12.
- #define **GPIOx_AF13** (13)
GPIO Alternate function 13.
- #define **GPIOx_AF14** (14)
GPIO Alternate function 14.
- #define **GPIOx_AF15** (15)
GPIO Alternate function 15.

5.22.1 Detailed Description

5.22.2 Macro Definition Documentation

5.22.2.1 **GPIOx_AF0**

```
#define GPIOx_AF0 (0)

#include <COTS/MCAL/GPIO/GPIO_interface.h>

GPIO Alternate function 0.

Definition at line 452 of file GPIO\_interface.h.
```

5.22.2.2 GPIOx_AF1

```
#define GPIOx_AF1 (1)

#include <COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO Alternate function 1.

Definition at line [459](#) of file [GPIO_interface.h](#).

5.22.2.3 GPIOx_AF2

```
#define GPIOx_AF2 (2)

#include <COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO Alternate function 2.

Definition at line [466](#) of file [GPIO_interface.h](#).

5.22.2.4 GPIOx_AF3

```
#define GPIOx_AF3 (3)

#include <COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO Alternate function 3.

Definition at line [473](#) of file [GPIO_interface.h](#).

5.22.2.5 GPIOx_AF4

```
#define GPIOx_AF4 (4)

#include <COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO Alternate function 4.

Definition at line [480](#) of file [GPIO_interface.h](#).

5.22.2.6 GPIOx_AF5

```
#define GPIOx_AF5 (5)

#include <COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO Alternate function 5.

Definition at line 487 of file [GPIO_interface.h](#).

5.22.2.7 GPIOx_AF6

```
#define GPIOx_AF6 (6)

#include <COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO Alternate function 6.

Definition at line 494 of file [GPIO_interface.h](#).

5.22.2.8 GPIOx_AF7

```
#define GPIOx_AF7 (7)

#include <COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO Alternate function 7.

Definition at line 501 of file [GPIO_interface.h](#).

5.22.2.9 GPIOx_AF8

```
#define GPIOx_AF8 (8)

#include <COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO Alternate function 8.

Definition at line 508 of file [GPIO_interface.h](#).

5.22.2.10 GPIOx_AF9

```
#define GPIOx_AF9 (9)

#include <COTS/MCAL/GPIO(GPIO_interface.h>
```

GPIO Alternate function 9.

Definition at line 515 of file [GPIO_interface.h](#).

5.22.2.11 GPIOx_AF10

```
#define GPIOx_AF10 (10)

#include <COTS/MCAL/GPIO(GPIO_interface.h>
```

GPIO Alternate function 10.

Definition at line 522 of file [GPIO_interface.h](#).

5.22.2.12 GPIOx_AF11

```
#define GPIOx_AF11 (11)

#include <COTS/MCAL/GPIO(GPIO_interface.h>
```

GPIO Alternate function 11.

Definition at line 529 of file [GPIO_interface.h](#).

5.22.2.13 GPIOx_AF12

```
#define GPIOx_AF12 (12)

#include <COTS/MCAL/GPIO(GPIO_interface.h>
```

GPIO Alternate function 12.

Definition at line 536 of file [GPIO_interface.h](#).

5.22.2.14 GPIOx_AF13

```
#define GPIOx_AF13 (13)

#include <COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO Alternate function 13.

Definition at line [543](#) of file [GPIO_interface.h](#).

5.22.2.15 GPIOx_AF14

```
#define GPIOx_AF14 (14)

#include <COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO Alternate function 14.

Definition at line [550](#) of file [GPIO_interface.h](#).

5.22.2.16 GPIOx_AF15

```
#define GPIOx_AF15 (15)

#include <COTS/MCAL/GPIO/GPIO_interface.h>
```

GPIO Alternate function 15.

Definition at line [557](#) of file [GPIO_interface.h](#).

5.23 GPIO Addresses

Macros

- #define [GPIOA_BASE_ADDRESS](#) (0x40020000)
- #define [GPIOB_BASE_ADDRESS](#) (0x40020400)
- #define [GPIOC_BASE_ADDRESS](#) (0x40020800)
- #define [CLK_SOURCE AHB_DividedBy8](#)

5.23.1 Detailed Description

5.23.2 Macro Definition Documentation

5.23.2.1 GPIOA_BASE_ADDRESS

```
#define GPIOA_BASE_ADDRESS (0x40020000)

#include <COTS/MCAL/GPIO/GPIO_private.h>
```

Port A Base address

5.23.2.2 GPIOB_BASE_ADDRESS

```
#define GPIOB_BASE_ADDRESS (0x40020400)

#include <COTS/MCAL/GPIO/GPIO_private.h>
```

Port B Base address

Definition at line 90 of file [GPIO_private.h](#).

5.23.2.3 GPIOC_BASE_ADDRESS

```
#define GPIOC_BASE_ADDRESS (0x40020800)

#include <COTS/MCAL/GPIO/GPIO_private.h>
```

Port C Base address

Definition at line 97 of file [GPIO_private.h](#).

5.23.2.4 CLK_SOURCE

```
#define CLK_SOURCE AHB_DividedBy8

#include <COTS/MCAL/SysTick/SysTick_config.h>
```

Definition at line 28 of file [SysTick_config.h](#).

5.24 GPIO Registers

Macros

- #define **GPIOA** ((volatile P2VAR(GPIOx_MemoryMapType))(GPIOA_BASE_ADDRESS))
- #define **GPIOB** ((volatile P2VAR(GPIOx_MemoryMapType))(GPIOB_BASE_ADDRESS))
- #define **GPIOC** ((volatile P2VAR(GPIOx_MemoryMapType))(GPIOC_BASE_ADDRESS))

5.24.1 Detailed Description

5.24.2 Macro Definition Documentation

5.24.2.1 GPIOA

```
#define GPIOA ((volatile P2VAR(GPIOx_MemoryMapType))(GPIOA_BASE_ADDRESS))
```

```
#include <COTS/MCAL/GPIO/GPIO_private.h>
```

GPIO register for port A

Definition at line 112 of file [GPIO_private.h](#).

5.24.2.2 GPIOB

```
#define GPIOB ((volatile P2VAR(GPIOx_MemoryMapType))(GPIOB_BASE_ADDRESS))
```

```
#include <COTS/MCAL/GPIO/GPIO_private.h>
```

GPIO register for port B

Definition at line 119 of file [GPIO_private.h](#).

5.24.2.3 GPIOC

```
#define GPIOC ((volatile P2VAR(GPIOx_MemoryMapType))(GPIOC_BASE_ADDRESS))
```

```
#include <COTS/MCAL/GPIO/GPIO_private.h>
```

GPIO register for port C

Definition at line 126 of file [GPIO_private.h](#).

5.25 Interrupt priority grouping

Interrupt priority grouping values.

Macros

- `#define _16GROUP_NoSub_Priorities (0b011)`
16 groups and 0 sub-groups
- `#define _8GROUP_2Sub_Priorities (0b100)`
8 groups and 2 sub-groups
- `#define _4GROUP_4Sub_Priorities (0b101)`
4 groups and 4 sub-groups
- `#define _2GROUP_8Sub_Priorities (0b110)`
2 groups and 8 sub-groups
- `#define NoGROUP_16Sub_Priorities (0b111)`
0 groups and 16 sub-groups
- `#define GROUP_4BITS (0b011)`
16 groups and 0 sub-groups
- `#define GROUP_3BITS (0b100)`
8 groups and 2 sub-groups
- `#define GROUP_2BITS (0b101)`
4 groups and 4 sub-groups
- `#define GROUP_1BITS (0b110)`
2 groups and 8 sub-groups
- `#define GROUP_0BITS (0b111)`
0 groups and 16 sub-groups

5.25.1 Detailed Description

Interrupt priority grouping values.

Interrupt priority grouping values for PRIGROUP in AIRCR register

See also

[SCB_MemoryMapType::AIRCR](#)

5.25.2 Macro Definition Documentation

5.25.2.1 _16GROUP_NoSub_Priorities

```
#define _16GROUP_NoSub_Priorities (0b011)

#include <COTS/MCAL/NVIC/NVIC_interface.h>

16 groups and 0 sub-groups
```

This tells the system there is going to be 16 groups and 0 sub-groups

See also

[GROUP_4BITS](#)

Definition at line [97](#) of file [NVIC_interface.h](#).

5.25.2.2 _8GROUP_2Sub_Priorities

```
#define _8GROUP_2Sub_Priorities (0b100)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

8 groups and 2 sub-groups

This tells the system there is going to be 8 groups and 2 sub-groups

See also

[GROUP_3BITS](#)

Definition at line 105 of file [NVIC_interface.h](#).

5.25.2.3 _4GROUP_4Sub_Priorities

```
#define _4GROUP_4Sub_Priorities (0b101)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

4 groups and 4 sub-groups

This tells the system there is going to be 4 groups and 4 sub-groups

See also

[GROUP_2BITS](#)

Definition at line 113 of file [NVIC_interface.h](#).

5.25.2.4 _2GROUP_8Sub_Priorities

```
#define _2GROUP_8Sub_Priorities (0b110)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

2 groups and 8 sub-groups

This tells the system there is going to be 2 groups and 8 sub-groups

See also

[GROUP_1BITS](#)

Definition at line 121 of file [NVIC_interface.h](#).

5.25.2.5 NoGROUP_16Sub_Priorities

```
#define NoGROUP_16Sub_Priorities (0b111)

#include <COTS/MCAL/NVIC/NVIC_interface.h>

0 groups and 16 sub-groups
```

This tells the system there is going to be 0 groups and 16 sub-groups

See also

[GROUP_0BITS](#)

Definition at line [129](#) of file [NVIC_interface.h](#).

5.25.2.6 GROUP_4BITS

```
#define GROUP_4BITS (0b011)

#include <COTS/MCAL/NVIC/NVIC_interface.h>

16 groups and 0 sub-groups
```

This tells the system there is going to be 16 groups and 0 sub-groups

Definition at line [147](#) of file [NVIC_interface.h](#).

5.25.2.7 GROUP_3BITS

```
#define GROUP_3BITS (0b100)

#include <COTS/MCAL/NVIC/NVIC_interface.h>

8 groups and 2 sub-groups
```

This tells the system there is going to be 8 groups and 2 sub-groups

Definition at line [154](#) of file [NVIC_interface.h](#).

5.25.2.8 GROUP_2BITS

```
#define GROUP_2BITS (0b101)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

4 groups and 4 sub-groups

This tells the system there is going to be 4 groups and 4 sub-groups

Definition at line 161 of file [NVIC_interface.h](#).

5.25.2.9 GROUP_1BITS

```
#define GROUP_1BITS (0b110)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

2 groups and 8 sub-groups

This tells the system there is going to be 2 groups and 8 sub-groups

Definition at line 168 of file [NVIC_interface.h](#).

5.25.2.10 GROUP_0BITS

```
#define GROUP_0BITS (0b111)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

0 groups and 16 sub-groups

This tells the system there is going to be 0 groups and 16 sub-groups

Definition at line 175 of file [NVIC_interface.h](#).

5.26 Group priorities

The group priority values.

Macros

- `#define NO_GROUP_PRIORITY (0)`
No group priority.
- `#define GROUP_PRIORITY_0 (0)`
Priority: 0.
- `#define GROUP_PRIORITY_1 (1)`
Priority: 1.
- `#define GROUP_PRIORITY_2 (2)`
Priority: 2.
- `#define GROUP_PRIORITY_3 (3)`
Priority: 3.
- `#define GROUP_PRIORITY_4 (4)`
Priority: 4.
- `#define GROUP_PRIORITY_5 (5)`
Priority: 5.
- `#define GROUP_PRIORITY_6 (6)`
Priority: 6.
- `#define GROUP_PRIORITY_7 (7)`
Priority: 7.
- `#define GROUP_PRIORITY_8 (8)`
Priority: 8.
- `#define GROUP_PRIORITY_9 (9)`
Priority: 9.
- `#define GROUP_PRIORITY_10 (10)`
Priority: 10.
- `#define GROUP_PRIORITY_11 (11)`
Priority: 11.
- `#define GROUP_PRIORITY_12 (12)`
Priority: 12.
- `#define GROUP_PRIORITY_13 (13)`
Priority: 13.
- `#define GROUP_PRIORITY_14 (14)`
Priority: 14.
- `#define GROUP_PRIORITY_15 (15)`
Priority: 15.

5.26.1 Detailed Description

The group priority values.

See also

[MNVIC_vSetPriority](#)
[Sub-Group priorities](#)

5.26.2 Macro Definition Documentation

5.26.2.1 NO_GROUP_PRIORITY

```
#define NO_GROUP_PRIORITY (0)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

No group priority.

Definition at line 189 of file [NVIC_interface.h](#).

5.26.2.2 GROUP_PRIORITY_0

```
#define GROUP_PRIORITY_0 (0)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

Priority: 0.

Definition at line 194 of file [NVIC_interface.h](#).

5.26.2.3 GROUP_PRIORITY_1

```
#define GROUP_PRIORITY_1 (1)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

Priority: 1.

Definition at line 199 of file [NVIC_interface.h](#).

5.26.2.4 GROUP_PRIORITY_2

```
#define GROUP_PRIORITY_2 (2)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

Priority: 2.

Definition at line 204 of file [NVIC_interface.h](#).

5.26.2.5 GROUP_PRIORITY_3

```
#define GROUP_PRIORITY_3 (3)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

Priority: 3.

Definition at line 209 of file [NVIC_interface.h](#).

5.26.2.6 GROUP_PRIORITY_4

```
#define GROUP_PRIORITY_4 (4)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

Priority: 4.

Definition at line 214 of file [NVIC_interface.h](#).

5.26.2.7 GROUP_PRIORITY_5

```
#define GROUP_PRIORITY_5 (5)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

Priority: 5.

Definition at line 219 of file [NVIC_interface.h](#).

5.26.2.8 GROUP_PRIORITY_6

```
#define GROUP_PRIORITY_6 (6)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

Priority: 6.

Definition at line 224 of file [NVIC_interface.h](#).

5.26.2.9 GROUP_PRIORITY_7

```
#define GROUP_PRIORITY_7 (7)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

Priority: 7.

Definition at line 229 of file [NVIC_interface.h](#).

5.26.2.10 GROUP_PRIORITY_8

```
#define GROUP_PRIORITY_8 (8)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

Priority: 8.

Definition at line 234 of file [NVIC_interface.h](#).

5.26.2.11 GROUP_PRIORITY_9

```
#define GROUP_PRIORITY_9 (9)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

Priority: 9.

Definition at line 239 of file [NVIC_interface.h](#).

5.26.2.12 GROUP_PRIORITY_10

```
#define GROUP_PRIORITY_10 (10)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

Priority: 10.

Definition at line 244 of file [NVIC_interface.h](#).

5.26.2.13 GROUP_PRIORITY_11

```
#define GROUP_PRIORITY_11 (11)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

Priority: 11.

Definition at line [249](#) of file [NVIC_interface.h](#).

5.26.2.14 GROUP_PRIORITY_12

```
#define GROUP_PRIORITY_12 (12)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

Priority: 12.

Definition at line [254](#) of file [NVIC_interface.h](#).

5.26.2.15 GROUP_PRIORITY_13

```
#define GROUP_PRIORITY_13 (13)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

Priority: 13.

Definition at line [259](#) of file [NVIC_interface.h](#).

5.26.2.16 GROUP_PRIORITY_14

```
#define GROUP_PRIORITY_14 (14)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

Priority: 14.

Definition at line [264](#) of file [NVIC_interface.h](#).

5.26.2.17 GROUP_PRIORITY_15

```
#define GROUP_PRIORITY_15 (15)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

Priority: 15.

Definition at line 269 of file [NVIC_interface.h](#).

5.27 Sub-Group priorities

The sub-group priority values.

Macros

- #define NO_SUB_PRIORITY (0)
No sub-group priority.
- #define SUB_PRIORITY_0 (0)
Priority: 0.
- #define SUB_PRIORITY_1 (1)
Priority: 1.
- #define SUB_PRIORITY_3 (3)
Priority: 3.
- #define SUB_PRIORITY_2 (2)
- #define SUB_PRIORITY_4 (4)
Priority: 4.
- #define SUB_PRIORITY_5 (5)
Priority: 5.
- #define SUB_PRIORITY_6 (6)
Priority: 6.
- #define SUB_PRIORITY_7 (7)
Priority: 7.
- #define SUB_PRIORITY_8 (8)
Priority: 8.
- #define SUB_PRIORITY_9 (9)
Priority: 9.
- #define SUB_PRIORITY_10 (10)
Priority: 10.
- #define SUB_PRIORITY_11 (11)
Priority: 11.
- #define SUB_PRIORITY_12 (12)
Priority: 12.
- #define SUB_PRIORITY_13 (13)
Priority: 13.
- #define SUB_PRIORITY_14 (14)
Priority: 14.
- #define SUB_PRIORITY_15 (15)
Priority: 15.

5.27.1 Detailed Description

The sub-group priority values.

See also

[MNVIC_vSetPriority](#)

[Group priorities](#)

5.27.2 Macro Definition Documentation

5.27.2.1 NO_SUB_PRIORITY

```
#define NO_SUB_PRIORITY (0)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

No sub-group priority.

Definition at line [285](#) of file [NVIC_interface.h](#).

5.27.2.2 SUB_PRIORITY_0

```
#define SUB_PRIORITY_0 (0)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

Priority: 0.

Definition at line [290](#) of file [NVIC_interface.h](#).

5.27.2.3 SUB_PRIORITY_1

```
#define SUB_PRIORITY_1 (1)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

Priority: 1.

Definition at line [295](#) of file [NVIC_interface.h](#).

5.27.2.4 SUB_PRIORITY_3

```
#define SUB_PRIORITY_3 (3)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

Priority: 3.

Definition at line 305 of file [NVIC_interface.h](#).

5.27.2.5 SUB_PRIORITY_2

```
#define SUB_PRIORITY_2 (2)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

Definition at line 300 of file [NVIC_interface.h](#).

5.27.2.6 SUB_PRIORITY_4

```
#define SUB_PRIORITY_4 (4)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

Priority: 4.

Definition at line 310 of file [NVIC_interface.h](#).

5.27.2.7 SUB_PRIORITY_5

```
#define SUB_PRIORITY_5 (5)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

Priority: 5.

Definition at line 315 of file [NVIC_interface.h](#).

5.27.2.8 SUB_PRIORITY_6

```
#define SUB_PRIORITY_6 (6)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

Priority: 6.

Definition at line 320 of file [NVIC_interface.h](#).

5.27.2.9 SUB_PRIORITY_7

```
#define SUB_PRIORITY_7 (7)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

Priority: 7.

Definition at line 325 of file [NVIC_interface.h](#).

5.27.2.10 SUB_PRIORITY_8

```
#define SUB_PRIORITY_8 (8)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

Priority: 8.

Definition at line 330 of file [NVIC_interface.h](#).

5.27.2.11 SUB_PRIORITY_9

```
#define SUB_PRIORITY_9 (9)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

Priority: 9.

Definition at line 335 of file [NVIC_interface.h](#).

5.27.2.12 SUB_PRIORITY_10

```
#define SUB_PRIORITY_10 (10)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

Priority: 10.

Definition at line 340 of file [NVIC_interface.h](#).

5.27.2.13 SUB_PRIORITY_11

```
#define SUB_PRIORITY_11 (11)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

Priority: 11.

Definition at line 345 of file [NVIC_interface.h](#).

5.27.2.14 SUB_PRIORITY_12

```
#define SUB_PRIORITY_12 (12)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

Priority: 12.

Definition at line 350 of file [NVIC_interface.h](#).

5.27.2.15 SUB_PRIORITY_13

```
#define SUB_PRIORITY_13 (13)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

Priority: 13.

Definition at line 355 of file [NVIC_interface.h](#).

5.27.2.16 SUB_PRIORITY_14

```
#define SUB_PRIORITY_14 (14)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

Priority: 14.

Definition at line 360 of file [NVIC_interface.h](#).

5.27.2.17 SUB_PRIORITY_15

```
#define SUB_PRIORITY_15 (15)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

Priority: 15.

Definition at line 365 of file [NVIC_interface.h](#).

5.28 NVIC Vector Table

NVIC vector table.

Macros

- #define [WWDG](#) (0)
Window Watchdog (WWDG) Interrupt.
- #define [EXTI16](#) (1)
EXTI Line 16 interrupt / PVD through EXTI line detection interrupt.
- #define [EXTI21](#) (2)
EXTI Line 21 interrupt / Tamper andTimeStamp interrupts through the EXTI line.
- #define [EXTI22](#) (3)
EXTI Line 22 interrupt / RTC (Real-time clock) wakeup interrupt through the EXTI line.
- #define [FLASH](#) (4)
Flash global interrupt.
- #define [RCC](#) (5)
RCC global interrupt.
- #define [EXTI0](#) (6)
EXTI Line 0 interrupt.
- #define [EXTI1](#) (7)
EXTI Line 1 interrupt.
- #define [EXTI2](#) (8)
EXTI Line 2 interrupt.
- #define [EXTI3](#) (9)
EXTI Line 3 interrupt.
- #define [EXTI4](#) (10)

- #define **EXTI Line 4 interrupt.**
- #define **DMA1_STREAM0** (11)
DMA1 (Direct Memory Access) Stream 0 global interrupt.
- #define **DMA1_STREAM1** (12)
DMA1 (Direct Memory Access) Stream 1 global interrupt.
- #define **DMA1_STREAM2** (13)
DMA1 (Direct Memory Access) Stream 2 global interrupt.
- #define **DMA1_STREAM3** (14)
DMA1 (Direct Memory Access) Stream 3 global interrupt.
- #define **DMA1_STREAM4** (15)
DMA1 (Direct Memory Access) Stream 4 global interrupt.
- #define **DMA1_STREAM5** (16)
DMA1 (Direct Memory Access) Stream 5 global interrupt.
- #define **DMA1_STREAM6** (17)
DMA1 (Direct Memory Access) Stream 6 global interrupt.
- #define **ADC** (18)
ADC (Analog-To-Digital Converter) ADC1 global interrupt.
- #define **EXTI9** (23)
EXTI Lines [9:5] interrupts.
- #define **TIM1_BRK_TIM9** (24)
TIM1 (Timer 1) break interrupt and TIM9 (Timer 9) global interrupt.
- #define **TIM1_UP_TIM10** (25)
TIM1 (Timer 1) update interrupt and TIM10 (Timer 10) global interrupt.
- #define **TIM1_TRG_COM_TIM11** (26)
TIM1 (Timer 1) trigger and commutation itnerrupts and TIM11 (Timer 11) global interrupts.
- #define **TIM1_CC** (27)
TIM1 (Timer 1) capture compare interrupt.
- #define **TIM2** (28)
TIM2 (Timer 2) global interrupt.
- #define **TIM3** (29)
TIM3 (Timer 3) global interrupt.
- #define **TIM4** (30)
TIM4 (Timer 4) global interrupt.
- #define **I2C1_EV** (31)
I2C1 (Inter-integrated Circuit 1) event interrupt.
- #define **I2C1_ER** (32)
I2C1 (Inter-integrated Circuit 1) error interrupt.
- #define **I2C2_EV** (33)
I2C2 (Inter-integrated Circuit 2) event interrupt.
- #define **I2C2_ER** (34)
I2C2 (Inter-integrated Circuit 2) error interrupt.
- #define **SPI1** (35)
SPI1 (Serial Peripheral Interface 1) global interrupt.
- #define **SPI2** (36)
SPI2 (Serial Peripheral Interface 2) global interrupt.
- #define **USART1** (37)
USART1 (Universal Synchronous/Asynchronous Receiver/Transmitter 1) global interrupt.
- #define **USART2** (38)
USART2 (Universal Synchronous/Asynchronous Receiver/Transmitter 2) global interrupt.
- #define **EXTI15_10** (30)
EXTI Line[15:10] interrupts.

- #define [EXTI17](#) (41)
EXTI Line 17 interrupt / RTC Alarms (A and B) through EXTI line interrupt.
- #define [EXTI18](#) (42)
EXTI Line 18 interrupt / USB On-The-Go FS Wakeup through EXTI line interrupt.
- #define [DMA1_STREAM7](#) (47)
DMA1 (Direct Memory Access) Stream 7 global interrupt.
- #define [SDIO](#) (49)
SDIO (Secure Digital Input Output) global interrupt.
- #define [TIM5](#) (50)
TIM5 (Timer 5) global interrupt.
- #define [SPI3](#) (51)
SPI3 (Serial Peripheral Interface 3) global interrupt.
- #define [DMA2_STREAM0](#) (56)
DMA2 (Direct Memory Access 2) Stream 0 global interrupt.
- #define [DMA2_STREAM1](#) (57)
DMA2 (Direct Memory Access 2) Stream 1 global interrupt.
- #define [DMA2_STREAM2](#) (58)
DMA2 (Direct Memory Access 2) Stream 2 global interrupt.
- #define [DMA2_STREAM5](#) (68)
DMA2 (Direct Memory Access 2) Stream 3 global interrupt.
- #define [DMA2_STREAM3](#) (59)
DMA2 (Direct Memory Access 2) Stream 6 global interrupt.
- #define [DMA2_STREAM4](#) (60)
DMA2 (Direct Memory Access 2) Stream 4 global interrupt.
- #define [OTG_FS](#) (67)
USB On The Go FS global interrupt.
- #define [DMA2_STREAM6](#) (69)
DMA2 (Direct Memory Access 2) Stream 7 global interrupt.
- #define [DMA2_STREAM7](#) (70)
DMA2 (Direct Memory Access 2) Stream 8 global interrupt.
- #define [USART6](#) (71)
USART6 (Universal Synchronous/Asynchronous Receiver/Transmitter 6) global interrupt.
- #define [I2C3_EV](#) (72)
I2C3 (Inter-integrated Circuit 3) event interrupt.
- #define [I2C3_ER](#) (73)
I2C3 (Inter-integrated Circuit 3) error interrupt.
- #define [FPU](#) (81)
FPU (Floating Point Unit) global interrupt.
- #define [SPI4](#) (84)
SPI4 (Serial Peripheral Interface 4) global interrupt.

5.28.1 Detailed Description

NVIC vector table.

NVIC vector table that contains each peripheral's priority.

The lower the value, the higher the priority

5.28.2 Macro Definition Documentation

5.28.2.1 WWDG

```
#define WWDG (0)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

Window Watchdog (WWDG) Interrupt.

Definition at line 379 of file [NVIC_interface.h](#).

5.28.2.2 EXTI16

```
#define EXTI16 (1)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

EXTI Line 16 interrupt / PVD through EXTI line detection interrupt.

Definition at line 384 of file [NVIC_interface.h](#).

5.28.2.3 EXTI21

```
#define EXTI21 (2)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

EXTI Line 21 interrupt / Tamper andTimeStamp interrupts through the EXTI line.

Definition at line 389 of file [NVIC_interface.h](#).

5.28.2.4 EXTI22

```
#define EXTI22 (3)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

EXTI Line 22 interrupt / RTC (Real-time clock) wakeup interrupt through the EXTI line.

Definition at line 394 of file [NVIC_interface.h](#).

5.28.2.5 FLASH

```
#define FLASH (4)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

Flash global interrupt.

Definition at line 399 of file [NVIC_interface.h](#).

5.28.2.6 RCC

```
#define RCC (5)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

RCC global interrupt.

Definition at line 404 of file [NVIC_interface.h](#).

5.28.2.7 EXTI0

```
#define EXTI0 (6)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

EXTI Line 0 interrupt.

Definition at line 409 of file [NVIC_interface.h](#).

5.28.2.8 EXTI1

```
#define EXTI1 (7)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

EXTI Line 1 interrupt.

Definition at line 414 of file [NVIC_interface.h](#).

5.28.2.9 EXTI2

```
#define EXTI2 (8)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

EXTI Line 2 interrupt.

Definition at line [419](#) of file [NVIC_interface.h](#).

5.28.2.10 EXTI3

```
#define EXTI3 (9)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

EXTI Line 3 interrupt.

Definition at line [424](#) of file [NVIC_interface.h](#).

5.28.2.11 EXTI4

```
#define EXTI4 (10)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

EXTI Line 4 interrupt.

Definition at line [429](#) of file [NVIC_interface.h](#).

5.28.2.12 DMA1_STREAM0

```
#define DMA1_STREAM0 (11)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

DMA1 (Direct Memory Access) Stream 0 global interrupt.

Definition at line [434](#) of file [NVIC_interface.h](#).

5.28.2.13 DMA1_STREAM1

```
#define DMA1_STREAM1 (12)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

DMA1 (Direct Memory Access) Steam 1 global interrupt.

Definition at line [439](#) of file [NVIC_interface.h](#).

5.28.2.14 DMA1_STREAM2

```
#define DMA1_STREAM2 (13)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

DMA1 (Direct Memory Access) Steam 2 global interrupt.

Definition at line [444](#) of file [NVIC_interface.h](#).

5.28.2.15 DMA1_STREAM3

```
#define DMA1_STREAM3 (14)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

DMA1 (Direct Memory Access) Steam 3 global interrupt.

Definition at line [449](#) of file [NVIC_interface.h](#).

5.28.2.16 DMA1_STREAM4

```
#define DMA1_STREAM4 (15)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

DMA1 (Direct Memory Access) Steam 4 global interrupt.

Definition at line [454](#) of file [NVIC_interface.h](#).

5.28.2.17 DMA1_STREAM5

```
#define DMA1_STREAM5 (16)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

DMA1 (Direct Memory Access) Steam 5 global interrupt.

Definition at line [459](#) of file [NVIC_interface.h](#).

5.28.2.18 DMA1_STREAM6

```
#define DMA1_STREAM6 (17)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

DMA1 (Direct Memory Access) Steam 6 global interrupt.

Definition at line [464](#) of file [NVIC_interface.h](#).

5.28.2.19 ADC

```
#define ADC (18)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

ADC (Analog-To-Digital Converter) ADC1 global interrupt.

Definition at line [469](#) of file [NVIC_interface.h](#).

5.28.2.20 EXTI9

```
#define EXTI9 (23)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

EXTI Lines [9:5] interrupts.

Definition at line [474](#) of file [NVIC_interface.h](#).

5.28.2.21 TIM1_BRK_TIM9

```
#define TIM1_BRK_TIM9 (24)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

TIM1 (Timer 1) break interrupt and TIM9 (Timer 9) global interrupt.

Definition at line [479](#) of file [NVIC_interface.h](#).

5.28.2.22 TIM1_UP_TIM10

```
#define TIM1_UP_TIM10 (25)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

TIM1 (Timer 1) update interrupt and TIM10 (Timer 10) global interrupt.

Definition at line [484](#) of file [NVIC_interface.h](#).

5.28.2.23 TIM1_TRG_COM_TIM11

```
#define TIM1_TRG_COM_TIM11 (26)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

TIM1 (Timer 1) trigger and commutation itnerrupts and TIM11 (Timer 11) global interrupts.

Definition at line [489](#) of file [NVIC_interface.h](#).

5.28.2.24 TIM1_CC

```
#define TIM1_CC (27)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

TIM1 (Timer 1) capture compare interrupt.

Definition at line [494](#) of file [NVIC_interface.h](#).

5.28.2.25 TIM2

```
#define TIM2 (28)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

TIM2 (Timer 2) global interrupt.

Definition at line 499 of file [NVIC_interface.h](#).

5.28.2.26 TIM3

```
#define TIM3 (29)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

TIM3 (Timer 3) global interrupt.

Definition at line 504 of file [NVIC_interface.h](#).

5.28.2.27 TIM4

```
#define TIM4 (30)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

TIM4 (Timer 4) global interrupt.

Definition at line 509 of file [NVIC_interface.h](#).

5.28.2.28 I2C1_EV

```
#define I2C1_EV (31)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

I2C1 (Inter-integrated Circuit 1) event interrupt.

Definition at line 514 of file [NVIC_interface.h](#).

5.28.2.29 I2C1_ER

```
#define I2C1_ER (32)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

I2C1 (Inter-integrated Circuit 1) error interrupt.

Definition at line 519 of file [NVIC_interface.h](#).

5.28.2.30 I2C2_EV

```
#define I2C2_EV (33)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

I2C2 (Inter-integrated Circuit 2) event interrupt.

Definition at line 524 of file [NVIC_interface.h](#).

5.28.2.31 I2C2_ER

```
#define I2C2_ER (34)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

I2C2 (Inter-integrated Circuit 2) error interrupt.

Definition at line 529 of file [NVIC_interface.h](#).

5.28.2.32 SPI1

```
#define SPI1 (35)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

SPI1 (Serial Peripheral Interface 1) global interrupt.

Definition at line 534 of file [NVIC_interface.h](#).

5.28.2.33 SPI2

```
#define SPI2 (36)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

SPI2 (Serial Peripheral Interface 2) global interrupt.

Definition at line [539](#) of file [NVIC_interface.h](#).

5.28.2.34 USART1

```
#define USART1 (37)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

USART1 (Universal Synchronous/Asynchronous Receiver/Transmitter 1) global interrupt.

Definition at line [544](#) of file [NVIC_interface.h](#).

5.28.2.35 USART2

```
#define USART2 (38)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

USART2 (Universal Synchronous/Asynchronous Receiver/Transmitter 2) global interrupt.

Definition at line [549](#) of file [NVIC_interface.h](#).

5.28.2.36 EXTI15_10

```
#define EXTI15_10 (30)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

EXTI Line[15:10] interrupts.

Definition at line [554](#) of file [NVIC_interface.h](#).

5.28.2.37 EXTI17

```
#define EXTI17 (41)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

EXTI Line 17 interrupt / RTC Alarms (A and B) through EXTI line interrupt.

Definition at line [559](#) of file [NVIC_interface.h](#).

5.28.2.38 EXTI18

```
#define EXTI18 (42)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

EXTI Line 18 interrupt / USB On-The-Go FS Wakeup through EXTI line interrupt.

Definition at line [564](#) of file [NVIC_interface.h](#).

5.28.2.39 DMA1_STREAM7

```
#define DMA1_STREAM7 (47)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

DMA1 (Direct Memory Access) Steam 7 global interrupt.

Definition at line [569](#) of file [NVIC_interface.h](#).

5.28.2.40 SDIO

```
#define SDIO (49)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

SDIO (Secure Digital Input Output) global interrupt.

Definition at line [574](#) of file [NVIC_interface.h](#).

5.28.2.41 TIM5

```
#define TIM5 (50)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

TIM5 (Timer 5) global interrupt.

Definition at line 579 of file [NVIC_interface.h](#).

5.28.2.42 SPI3

```
#define SPI3 (51)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

SPI3 (Serial Peripheral Interface 3) global interrupt.

Definition at line 584 of file [NVIC_interface.h](#).

5.28.2.43 DMA2_STREAM0

```
#define DMA2_STREAM0 (56)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

DMA2 (Direct Memory Access 2) Steam 0 global interrupt.

Definition at line 589 of file [NVIC_interface.h](#).

5.28.2.44 DMA2_STREAM1

```
#define DMA2_STREAM1 (57)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

DMA2 (Direct Memory Access 2) Steam 1 global interrupt.

Definition at line 594 of file [NVIC_interface.h](#).

5.28.2.45 DMA2_STREAM2

```
#define DMA2_STREAM2 (58)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

DMA2 (Direct Memory Access 2) Steam 2 global interrupt.

Definition at line [599](#) of file [NVIC_interface.h](#).

5.28.2.46 DMA2_STREAM5

```
#define DMA2_STREAM5 (68)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

DMA2 (Direct Memory Access 2) Steam 3 global interrupt.

DMA2 (Direct Memory Access 2) Steam 5 global interrupt.

Definition at line [619](#) of file [NVIC_interface.h](#).

5.28.2.47 DMA2_STREAM3

```
#define DMA2_STREAM3 (59)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

Definition at line [604](#) of file [NVIC_interface.h](#).

5.28.2.48 DMA2_STREAM4

```
#define DMA2_STREAM4 (60)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

DMA2 (Direct Memory Access 2) Steam 4 global interrupt.

Definition at line [609](#) of file [NVIC_interface.h](#).

5.28.2.49 OTG_FS

```
#define OTG_FS (67)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

USB On The Go FS global interrupt.

Definition at line [614](#) of file [NVIC_interface.h](#).

5.28.2.50 DMA2_STREAM6

```
#define DMA2_STREAM6 (69)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

DMA2 (Direct Memory Access 2) Steam 6 global interrupt.

Definition at line [624](#) of file [NVIC_interface.h](#).

5.28.2.51 DMA2_STREAM7

```
#define DMA2_STREAM7 (70)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

DMA2 (Direct Memory Access 2) Steam 7 global interrupt.

Definition at line [629](#) of file [NVIC_interface.h](#).

5.28.2.52 USART6

```
#define USART6 (71)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

USART6 (Universal Synchronous/Asynchronous Receiver/Transmitter 6) global interrupt.

Definition at line [634](#) of file [NVIC_interface.h](#).

5.28.2.53 I2C3_EV

```
#define I2C3_EV (72)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

I2C3 (Inter-integrated Circuit 3) event interrupt.

I2C3 (Inter-integrated Circuit 3) error interrupt.

5.28.2.54 I2C3_ER

```
#define I2C3_ER (73)

#include <COTS/MCAL/NVIC/NVIC_interface.h>
```

Definition at line [644](#) of file [NVIC_interface.h](#).

5.28.2.55 FPU

```
#define FPU (81)

#include <COTS/MCAL/NVIC/NVIC_interface.h>

FPU (Floating Point Unit) global interrupt.

Definition at line 649 of file NVIC\_interface.h.
```

5.28.2.56 SPI4

```
#define SPI4 (84)

#include <COTS/MCAL/NVIC/NVIC_interface.h>

SPI4 (Serial Peripheral Interface 4) global interrupt.

Definition at line 654 of file NVIC\_interface.h.
```

5.29 NVIC Addresses

NVIC registers base addresses.

Macros

- `#define NVIC_BASE_ADDRESS (0xE000E100)`
NVIC base address.
- `#define SCB_BASE_ADDRESS (0xE000ED00)`
SCB base address.

5.29.1 Detailed Description

NVIC registers base addresses.

5.29.2 Macro Definition Documentation

5.29.2.1 NVIC_BASE_ADDRESS

```
#define NVIC_BASE_ADDRESS (0xE000E100)

#include <COTS/MCAL/NVIC/NVIC_private.h>
```

NVIC base address.

Definition at line [212](#) of file [NVIC_private.h](#).

5.29.2.2 SCB_BASE_ADDRESS

```
#define SCB_BASE_ADDRESS (0xE000ED00)

#include <COTS/MCAL/NVIC/NVIC_private.h>
```

SCB base address.

Definition at line [219](#) of file [NVIC_private.h](#).

5.30 NVIC Registers

NVIC registers.

Macros

- `#define MSCB ((volatile P2VAR(SCB_MemoryMapType))(SCB_BASE_ADDRESS))`
- `#define MNVIC ((volatile P2VAR(NVIC_MemoryMapType))(NVIC_BASE_ADDRESS))`

5.30.1 Detailed Description

NVIC registers.

5.30.2 Macro Definition Documentation

5.30.2.1 MSCB

```
#define MSCB ((volatile P2VAR(SCB_MemoryMapType)) (SCB_BASE_ADDRESS))  
  
#include <COTS/MCAL/NVIC/NVIC_private.h>
```

MSCB register

Definition at line [234](#) of file [NVIC_private.h](#).

5.30.2.2 MNVIC

```
#define MNVIC ((volatile P2VAR(NVIC_MemoryMapType)) (NVIC_BASE_ADDRESS))  
  
#include <COTS/MCAL/NVIC/NVIC_private.h>
```

MNVIC register

Definition at line [240](#) of file [NVIC_private.h](#).

5.31 NVIC Settable Priorities

NVIC Settable Priorities for the hardware interrupts.

Macros

- #define PEND_SV (-6)
- #define SYSTICK (-5)
- #define SV_CALL (-4)
- #define MEMORY_MANAGE (-3)
- #define BUS_FAULT (-2)
- #define USAGE_FAULT (-1)

5.31.1 Detailed Description

NVIC Settable Priorities for the hardware interrupts.

5.31.2 Macro Definition Documentation

5.31.2.1 PEND_SV

```
#define PEND_SV (-6)

#include <COTS/MCAL/NVIC/NVIC_private.h>

Pendable request for system service

Definition at line 266 of file NVIC\_private.h.
```

5.31.2.2 SYSTICK

```
#define SYSTICK (-5)

#include <COTS/MCAL/NVIC/NVIC_private.h>

System tick timer

Definition at line 272 of file NVIC\_private.h.
```

5.31.2.3 SV_CALL

```
#define SV_CALL (-4)

#include <COTS/MCAL/NVIC/NVIC_private.h>

System service call via SWI instruction

Definition at line 278 of file NVIC\_private.h.
```

5.31.2.4 MEMORY_MANAGE

```
#define MEMORY_MANAGE (-3)

#include <COTS/MCAL/NVIC/NVIC_private.h>

Memory management

Pre-fetch fault, memory access fault
```

5.31.2.5 BUS_FAULT

```
#define BUS_FAULT (-2)

#include <COTS/MCAL/NVIC/NVIC_private.h>
```

Definition at line 290 of file [NVIC_private.h](#).

5.31.2.6 USAGE_FAULT

```
#define USAGE_FAULT (-1)

#include <COTS/MCAL/NVIC/NVIC_private.h>
```

Undefined instruction or illegal state

Definition at line 296 of file [NVIC_private.h](#).

5.32 ID options

Macros

- #define RCC_AHB1 1
AHB1 Bus.
- #define RCC_AHB2 2
AHB2 Bus.
- #define RCC_APB1 3
APB1 Bus.
- #define RCC_APB2 4
APB2 Bus.
- #define RCC_AHB1LPENR 5
peripheral clock enable in low power mode register

5.32.1 Detailed Description

5.32.2 Macro Definition Documentation

5.32.2.1 RCC_AHB1

```
#define RCC_AHB1 1

#include <COTS/MCAL/RCC/MRCC_interface.h>

AHB1 Bus.

Definition at line 51 of file MRCC\_interface.h.
```

5.32.2.2 RCC_AHB2

```
#define RCC_AHB2 2

#include <COTS/MCAL/RCC/MRCC_interface.h>

AHB2 Bus.
```

Definition at line [58](#) of file [MRCC_interface.h](#).

5.32.2.3 RCC_APB1

```
#define RCC_APB1 3

#include <COTS/MCAL/RCC/MRCC_interface.h>

APB1 Bus.
```

Definition at line [65](#) of file [MRCC_interface.h](#).

5.32.2.4 RCC_APB2

```
#define RCC_APB2 4

#include <COTS/MCAL/RCC/MRCC_interface.h>

APB2 Bus.
```

Definition at line [72](#) of file [MRCC_interface.h](#).

5.32.2.5 RCC_AHB1LPENR

```
#define RCC_AHB1LPENR 5

#include <COTS/MCAL/RCC/MRCC_interface.h>

peripheral clock enable in low power mode register

Definition at line 79 of file MRCC\_interface.h.
```

5.33 ID options

Macros

- #define AHB1ENR_DMA2EN 22
Enable CLK on DMA2 peripheral.
- #define AHB1ENR_DMA1EN 21
Enable CLK on DMA1 peripheral.
- #define AHB1ENR_CRCEN 12
Enable CLK on CRC peripheral.
- #define AHB1ENR_GPIOHEN 7
Enable CLK on GPIOH peripheral.
- #define AHB1ENR_GPIOEEN 4
Enable CLK on GPIOE peripheral.
- #define AHB1ENR_GPIODEN 3
Enable CLK on GPIOD peripheral.
- #define AHB1ENR_GPIOCEN 2
Enable CLK on GPIOC peripheral.
- #define AHB1ENR_GPIOBEN 1
Enable CLK on GPIOB peripheral.
- #define AHB1ENR_GPIOAEN 0
Enable CLK on GPIOA peripheral.
- #define AHB2ENR_OTGFSEN 7
Enable CLK on OTGFS peripheral.
- #define APB1ENR_PWREN 28
Enable CLK on PWR peripheral.
- #define APB1ENR_I2C3EN 23
Enable CLK on I2C3 peripheral.
- #define APB1ENR_I2C2EN 22
Enable CLK on I2C2 peripheral.
- #define APB1ENR_I2C1EN 21
Enable CLK on I2C1 peripheral.
- #define APB1ENR_USART2EN 17
Enable CLK on USART2 peripheral.
- #define APB1ENR_SPI3EN 15
Enable CLK on SPI3 peripheral.
- #define APB1ENR_SPI2EN 14
Enable CLK on SPI2 peripheral.
- #define APB1ENR_WWDGEN 11
Enable CLK on WWD peripheral.
- #define APB1ENR_TIM5EN 3
Enable CLK on TIM5 peripheral.
- #define APB1ENR_TIM4EN 2
Enable CLK on TIM4 peripheral.
- #define APB1ENR_TIM3EN 1
Enable CLK on TIM3 peripheral.
- #define APB1ENR_TIM2EN 0
Enable CLK on TIM2 peripheral.
- #define APB2ENR_TIM11EN 18
Enable CLK on TIM11 peripheral.

- #define APB2ENR_TIM10EN 17
Enable CLK on TIM10 peripheral.
- #define APB2ENR_TIM9EN 16
Enable CLK on TIM9 peripheral.
- #define APB2ENR_SYSCFGEN 14
Enable CLK on SYSCFG peripheral.
- #define APB2ENR_SPI4EN 13
Enable CLK on SPI4 peripheral.
- #define APB2ENR_SPI1EN 12
Enable CLK on SPI1 peripheral.
- #define APB2ENR_SDIOEN 11
Enable CLK on SDIO peripheral.
- #define APB2ENR_ADC1EN 8
Enable CLK on ADC1 peripheral.
- #define APB2ENR_USART6EN 5
Enable CLK on USART6 peripheral.
- #define APB2ENR_USART1EN 4
Enable CLK on USART1 peripheral.
- #define APB2ENR_TIM1EN 0
Enable CLK on TIM1 peripheral.
- #define AHB1LPENR_FLITFLPEN 15
Enable CLK on FLITFLP peripheral.

5.33.1 Detailed Description

5.33.2 Macro Definition Documentation

5.33.2.1 AHB1ENR_DMA2EN

```
#define AHB1ENR_DMA2EN 22

#include <COTS/MCAL/RCC/MRCC_interface.h>

Enable CLK on DMA2 peripheral.

Definition at line 94 of file MRCC_interface.h.
```

5.33.2.2 AHB1ENR_DMA1EN

```
#define AHB1ENR_DMA1EN 21

#include <COTS/MCAL/RCC/MRCC_interface.h>

Enable CLK on DMA1 peripheral.

Definition at line 101 of file MRCC_interface.h.
```

5.33.2.3 AHB1ENR_CRCEN

```
#define AHB1ENR_CRCEN 12

#include <COTS/MCAL/RCC/MRCC_interface.h>
```

Enable CLK on CRC peripheral.

Definition at line [108](#) of file [MRCC_interface.h](#).

5.33.2.4 AHB1ENR_GPIOHEN

```
#define AHB1ENR_GPIOHEN 7

#include <COTS/MCAL/RCC/MRCC_interface.h>
```

Enable CLK on GPIOH peripheral.

Definition at line [115](#) of file [MRCC_interface.h](#).

5.33.2.5 AHB1ENR_GPIOEEN

```
#define AHB1ENR_GPIOEEN 4

#include <COTS/MCAL/RCC/MRCC_interface.h>
```

Enable CLK on GPIOE peripheral.

Definition at line [122](#) of file [MRCC_interface.h](#).

5.33.2.6 AHB1ENR_GPIODEN

```
#define AHB1ENR_GPIODEN 3

#include <COTS/MCAL/RCC/MRCC_interface.h>
```

Enable CLK on GPIOD peripheral.

Definition at line [129](#) of file [MRCC_interface.h](#).

5.33.2.7 AHB1ENR_GPIOCEN

```
#define AHB1ENR_GPIOCEN 2

#include <COTS/MCAL/RCC/MRCC_interface.h>
```

Enable CLK on GPIOC peripheral.

Definition at line 136 of file [MRCC_interface.h](#).

5.33.2.8 AHB1ENR_GPIOBEN

```
#define AHB1ENR_GPIOBEN 1

#include <COTS/MCAL/RCC/MRCC_interface.h>
```

Enable CLK on GPIOB peripheral.

Definition at line 143 of file [MRCC_interface.h](#).

5.33.2.9 AHB1ENR_GPIOAEN

```
#define AHB1ENR_GPIOAEN 0

#include <COTS/MCAL/RCC/MRCC_interface.h>
```

Enable CLK on GPIOA peripheral.

Definition at line 150 of file [MRCC_interface.h](#).

5.33.2.10 AHB2ENR_OTGFSEN

```
#define AHB2ENR_OTGFSEN 7

#include <COTS/MCAL/RCC/MRCC_interface.h>
```

Enable CLK on OTGFS peripheral.

Definition at line 157 of file [MRCC_interface.h](#).

5.33.2.11 APB1ENR_PWREN

```
#define APB1ENR_PWREN 28  
  
#include <COTS/MCAL/RCC/MRCC_interface.h>
```

Enable CLK on PWR peripheral.

Definition at line 164 of file [MRCC_interface.h](#).

5.33.2.12 APB1ENR_I2C3EN

```
#define APB1ENR_I2C3EN 23  
  
#include <COTS/MCAL/RCC/MRCC_interface.h>
```

Enable CLK on I2C3 peripheral.

Definition at line 171 of file [MRCC_interface.h](#).

5.33.2.13 APB1ENR_I2C2EN

```
#define APB1ENR_I2C2EN 22  
  
#include <COTS/MCAL/RCC/MRCC_interface.h>
```

Enable CLK on I2C2 peripheral.

Definition at line 178 of file [MRCC_interface.h](#).

5.33.2.14 APB1ENR_I2C1EN

```
#define APB1ENR_I2C1EN 21  
  
#include <COTS/MCAL/RCC/MRCC_interface.h>
```

Enable CLK on I2C1 peripheral.

Definition at line 185 of file [MRCC_interface.h](#).

5.33.2.15 APB1ENR_USART2EN

```
#define APB1ENR_USART2EN 17  
  
#include <COTS/MCAL/RCC/MRCC_interface.h>
```

Enable CLK on USART2 peripheral.

Definition at line 192 of file [MRCC_interface.h](#).

5.33.2.16 APB1ENR_SPI3EN

```
#define APB1ENR_SPI3EN 15  
  
#include <COTS/MCAL/RCC/MRCC_interface.h>
```

Enable CLK on SPI3 peripheral.

Definition at line 199 of file [MRCC_interface.h](#).

5.33.2.17 APB1ENR_SPI2EN

```
#define APB1ENR_SPI2EN 14  
  
#include <COTS/MCAL/RCC/MRCC_interface.h>
```

Enable CLK on SPI2 peripheral.

Definition at line 206 of file [MRCC_interface.h](#).

5.33.2.18 APB1ENR_WWDGEN

```
#define APB1ENR_WWDGEN 11  
  
#include <COTS/MCAL/RCC/MRCC_interface.h>
```

Enable CLK on WWD peripheral.

Definition at line 213 of file [MRCC_interface.h](#).

5.33.2.19 APB1ENR_TIM5EN

```
#define APB1ENR_TIM5EN 3

#include <COTS/MCAL/RCC/MRCC_interface.h>
```

Enable CLK on TIM5 peripheral.

Definition at line [220](#) of file [MRCC_interface.h](#).

5.33.2.20 APB1ENR_TIM4EN

```
#define APB1ENR_TIM4EN 2

#include <COTS/MCAL/RCC/MRCC_interface.h>
```

Enable CLK on TIM4 peripheral.

Definition at line [227](#) of file [MRCC_interface.h](#).

5.33.2.21 APB1ENR_TIM3EN

```
#define APB1ENR_TIM3EN 1

#include <COTS/MCAL/RCC/MRCC_interface.h>
```

Enable CLK on TIM3 peripheral.

Definition at line [234](#) of file [MRCC_interface.h](#).

5.33.2.22 APB1ENR_TIM2EN

```
#define APB1ENR_TIM2EN 0

#include <COTS/MCAL/RCC/MRCC_interface.h>
```

Enable CLK on TIM2 peripheral.

Definition at line [241](#) of file [MRCC_interface.h](#).

5.33.2.23 APB2ENR_TIM11EN

```
#define APB2ENR_TIM11EN 18

#include <COTS/MCAL/RCC/MRCC_interface.h>
```

Enable CLK on TIM11 peripheral.

Definition at line [248](#) of file [MRCC_interface.h](#).

5.33.2.24 APB2ENR_TIM10EN

```
#define APB2ENR_TIM10EN 17

#include <COTS/MCAL/RCC/MRCC_interface.h>
```

Enable CLK on TIM10 peripheral.

Definition at line [255](#) of file [MRCC_interface.h](#).

5.33.2.25 APB2ENR_TIM9EN

```
#define APB2ENR_TIM9EN 16

#include <COTS/MCAL/RCC/MRCC_interface.h>
```

Enable CLK on TIM9 peripheral.

Definition at line [262](#) of file [MRCC_interface.h](#).

5.33.2.26 APB2ENR_SYSCFGGEN

```
#define APB2ENR_SYSCFGGEN 14

#include <COTS/MCAL/RCC/MRCC_interface.h>
```

Enable CLK on SYSCFG peripheral.

Definition at line [269](#) of file [MRCC_interface.h](#).

5.33.2.27 APB2ENR_SPI4EN

```
#define APB2ENR_SPI4EN 13

#include <COTS/MCAL/RCC/MRCC_interface.h>
```

Enable CLK on SPI4 peripheral.

Definition at line [276](#) of file [MRCC_interface.h](#).

5.33.2.28 APB2ENR_SPI1EN

```
#define APB2ENR_SPI1EN 12

#include <COTS/MCAL/RCC/MRCC_interface.h>
```

Enable CLK on SPI1 peripheral.

Definition at line [283](#) of file [MRCC_interface.h](#).

5.33.2.29 APB2ENR_SDIOEN

```
#define APB2ENR_SDIOEN 11

#include <COTS/MCAL/RCC/MRCC_interface.h>
```

Enable CLK on SDIO peripheral.

Definition at line [290](#) of file [MRCC_interface.h](#).

5.33.2.30 APB2ENR_ADC1EN

```
#define APB2ENR_ADC1EN 8

#include <COTS/MCAL/RCC/MRCC_interface.h>
```

Enable CLK on ADC1 peripheral.

Definition at line [297](#) of file [MRCC_interface.h](#).

5.33.2.31 APB2ENR_USART6EN

```
#define APB2ENR_USART6EN 5

#include <COTS/MCAL/RCC/MRCC_interface.h>
```

Enable CLK on USART6 peripheral.

Definition at line 304 of file [MRCC_interface.h](#).

5.33.2.32 APB2ENR_USART1EN

```
#define APB2ENR_USART1EN 4

#include <COTS/MCAL/RCC/MRCC_interface.h>
```

Enable CLK on USART1 peripheral.

Definition at line 311 of file [MRCC_interface.h](#).

5.33.2.33 APB2ENR_TIM1EN

```
#define APB2ENR_TIM1EN 0

#include <COTS/MCAL/RCC/MRCC_interface.h>
```

Enable CLK on TIM1 peripheral.

Definition at line 318 of file [MRCC_interface.h](#).

5.33.2.34 AHB1LPENR_FLITFLPEN

```
#define AHB1LPENR_FLITFLPEN 15

#include <COTS/MCAL/RCC/MRCC_interface.h>
```

Enable CLK on FLITFLP peripheral.

Definition at line 325 of file [MRCC_interface.h](#).

5.34 Systick Interval Mode Configuration

Macros

- #define **SINGLE_INTERVAL_MODE** (1)
Single interval mode.
- #define **PERIODIC_INTERVAL_MODE** (2)
Periodic interval mode.

5.34.1 Detailed Description

5.34.2 Macro Definition Documentation

5.34.2.1 SINGLE_INTERVAL_MODE

```
#define SINGLE_INTERVAL_MODE (1)

#include <COTS/MCAL/SysTick/SysTick_config.h>

Single interval mode.

Definition at line 52 of file SysTick\_config.h.
```

5.34.2.2 PERIODIC_INTERVAL_MODE

```
#define PERIODIC_INTERVAL_MODE (2)

#include <COTS/MCAL/SysTick/SysTick_config.h>

Periodic interval mode.

Definition at line 57 of file SysTick\_config.h.
```

5.35 Delay Units

Macros

- #define **MILLI_SEC** (1)

Delay in milli seconds.

- #define **MICRO_SEC** (2)

- #define **SEC** (3)

Delay in seconds.

5.35.1 Detailed Description

5.35.2 Macro Definition Documentation

5.35.2.1 MILLI_SEC

```
#define MILLI_SEC (1)

#include <COTS/MCAL/SysTick/SysTick_interface.h>
```

Delay in milli seconds.

Delay in micro seconds.

5.35.2.2 MICRO_SEC

```
#define MICRO_SEC (2)

#include <COTS/MCAL/SysTick/SysTick_interface.h>
```

Definition at line 143 of file [SysTick_interface.h](#).

5.35.2.3 SEC

```
#define SEC (3)

#include <COTS/MCAL/SysTick/SysTick_interface.h>
```

Delay in seconds.

Definition at line 149 of file [SysTick_interface.h](#).

5.36 Systick Addresses

Macros

- #define SysTick_BASE_ADDRESS (0xE000E010)

5.36.1 Detailed Description

5.36.2 Macro Definition Documentation

5.36.2.1 SysTick_BASE_ADDRESS

```
#define SysTick_BASE_ADDRESS (0xE000E010)

#include <COTS/MCAL/SysTick/SysTick_private.h>
```

Systick Base address

Definition at line 55 of file [SysTick_private.h](#).

5.37 Systick Registers

Macros

- #define [SysTick](#) ((volatile P2VAR(SysTick_Type))(SysTick_BASE_ADDRESS))

5.37.1 Detailed Description

5.37.2 Macro Definition Documentation

5.37.2.1 SysTick

```
#define SysTick ((volatile P2VAR(SysTick_Type))(SysTick_BASE_ADDRESS))

#include <COTS/MCAL/SysTick/SysTick_private.h>
```

Systick register

Definition at line 70 of file [SysTick_private.h](#).

5.38 Systick Register Bits Positions

Macros

- #define [COUNTFLAG](#) (16)
- #define [CLKSOURCE](#) (2)
- #define [TICKINT](#) (1)
- #define [COUNTER_ENABLE](#) (0)

5.38.1 Detailed Description

5.38.2 Macro Definition Documentation

5.38.2.1 COUNTFLAG

```
#define COUNTFLAG (16)

#include <COTS/MCAL/SysTick/SysTick_private.h>
```

This bit is responsible for keeping an eye on the timer whether it has counted to 0 or no, since the last time this was read It returns 1 if the timer counted to 0, since last time this was read

Definition at line 85 of file [SysTick_private.h](#).

5.38.2.2 CLKSOURCE

```
#define CLKSOURCE (2)

#include <COTS/MCAL/SysTick/SysTick_private.h>
```

This bit is responsible for selecting the clock source for the Systick peripheral

Definition at line 91 of file [SysTick_private.h](#).

5.38.2.3 TICKINT

```
#define TICKINT (1)

#include <COTS/MCAL/SysTick/SysTick_private.h>
```

This bit is responsible for controlling the exception request whether to enable or disable the request when reaching zero

Definition at line 97 of file [SysTick_private.h](#).

5.38.2.4 COUNTER_ENABLE

```
#define COUNTER_ENABLE (0)

#include <COTS/MCAL/SysTick/SysTick_private.h>
```

This bit is responsible for whether to enable the counter or no When this is enabled, the counter loads the reload value from the load register and then starts countdowning. On reaching 0, it sets the COUNTFLAG to 1 and optionally asserts the Systick depending on the value of TICKINT. Then, it loads the reload value again and begins counting.

See also

[SysTick_Type::VAL](#)
[TICKINT](#)

Definition at line 108 of file [SysTick_private.h](#).

5.39 Systick Clock Sources

Macros

- #define AHB_DividedBy8 (1)
- #define AHB (2)

5.39.1 Detailed Description

5.39.2 Macro Definition Documentation

5.39.2.1 AHB_DividedBy8

```
#define AHB_DividedBy8 (1)

#include <COTS/MCAL/SysTick/SysTick_private.h>
```

Set the clock source to divide the clock output of RCC by 8

Definition at line 120 of file [SysTick_private.h](#).

5.39.2.2 AHB

```
#define AHB (2)

#include <COTS/MCAL/SysTick/SysTick_private.h>
```

Set the clock source to the same as the clock output of RCC

Definition at line 126 of file [SysTick_private.h](#).

5.40 Systick Exception (Interrupt) Status

Macros

- #define Dont AssertRequest (1)
- #define AssertRequest (2)

5.40.1 Detailed Description

5.40.2 Macro Definition Documentation

5.40.2.1 Dont.AssertRequest

```
#define Dont.AssertRequest (1)

#include <COTS/MCAL/SysTick/SysTick_private.h>
```

Don't raise exception (interrupt) when the counter reaches 0

Definition at line 138 of file [SysTick_private.h](#).

5.40.2.2 AssertRequest

```
#define AssertRequest (2)

#include <COTS/MCAL/SysTick/SysTick_private.h>
```

Raise exception (interrupt) when the counter reaches 0

Definition at line 144 of file [SysTick_private.h](#).

Chapter 6

Data Structure Documentation

6.1 DCM_MotorConfiguration Struct Reference

DC Motor configuration structure for motor initialization DC Motor configuration for motor initialization for a certain port and pin. Since DC Motors have 2 sides to be controller, we will intilize each motor with 2 pins. Taking into consideration that the pins could be on different ports.

```
#include "COTS/HAL/DCMOTOR/DCM_interface.h"
```

Data Fields

- `u8_t Side1Port`
Initialize the DC Motor on a certain port.
- `u8_t Side1Pin`
Initialize the DC Motor on a certain pin.
- `u8_t Side2Port`
Initialize the DC Motor on a certain port.
- `u8_t Side2Pin`
Initialize the DC Motor on a certain pin.

6.1.1 Detailed Description

DC Motor configuration structure for motor initialization DC Motor configuration for motor initialization for a certain port and pin. Since DC Motors have 2 sides to be controller, we will intilize each motor with 2 pins. Taking into consideration that the pins could be on different ports.

Definition at line 20 of file [DCM_interface.h](#).

6.1.2 Field Documentation

6.1.2.1 Side1Port

`u8_t Side1Port`

Initialize the DC Motor on a certain port.

Definition at line 25 of file [DCM_interface.h](#).

Referenced by [HDCM_vInitMotor\(\)](#), [HDCM_vMoveBackward\(\)](#), [HDCM_vMoveForward\(\)](#), and [HDCM_vStopMotor\(\)](#).

6.1.2.2 Side1Pin

`u8_t Side1Pin`

Initialize the DC Motor on a certain pin.

Definition at line 29 of file [DCM_interface.h](#).

Referenced by [HDCM_vInitMotor\(\)](#), [HDCM_vMoveBackward\(\)](#), [HDCM_vMoveForward\(\)](#), and [HDCM_vStopMotor\(\)](#).

6.1.2.3 Side2Port

`u8_t Side2Port`

Initialize the DC Motor on a certain port.

Definition at line 33 of file [DCM_interface.h](#).

Referenced by [HDCM_vInitMotor\(\)](#), [HDCM_vMoveBackward\(\)](#), [HDCM_vMoveForward\(\)](#), and [HDCM_vStopMotor\(\)](#).

6.1.2.4 Side2Pin

`u8_t Side2Pin`

Initialize the DC Motor on a certain pin.

Definition at line 37 of file [DCM_interface.h](#).

Referenced by [HDCM_vInitMotor\(\)](#), [HDCM_vMoveBackward\(\)](#), [HDCM_vMoveForward\(\)](#), and [HDCM_vStopMotor\(\)](#).

The documentation for this struct was generated from the following file:

- COTS/HAL/DCMOTOR/[DCM_interface.h](#)

6.2 EXTI_ConfigType Struct Reference

Interrupt configuration structure to initialize the interrupt with.

```
#include "COTS/MCAL/EXTI/EXTI_interface.h"
```

Data Fields

- `u8_t LineNum`
Initialize the interrupt with a certain line ID.
- `u8_t PortNum`
Initialize the interrupt with a certain PORT.
- `u8_t TriggerStatus`
The trigger status to trigger when the interrupt occurs.

6.2.1 Detailed Description

Interrupt configuration structure to initialize the interrupt with.

Definition at line 18 of file [EXTI_interface.h](#).

6.2.2 Field Documentation

6.2.2.1 LineNum

`u8_t LineNum`

Initialize the interrupt with a certain line ID.

Definition at line 23 of file [EXTI_interface.h](#).

Referenced by [MEXTI_vInit_WithStruct\(\)](#).

6.2.2.2 PortNum

`u8_t PortNum`

Initialize the interrupt with a certain PORT.

Definition at line 28 of file [EXTI_interface.h](#).

Referenced by [MEXTI_vInit_WithStruct\(\)](#).

6.2.2.3 TriggerStatus

`u8_t TriggerStatus`

The trigger status to trigger when the interrupt occurs.

Definition at line 33 of file [EXTI_interface.h](#).

Referenced by [MEXTI_vInit_WithStruct\(\)](#).

The documentation for this struct was generated from the following file:

- COTS/MCAL/EXTI/[EXTI_interface.h](#)

6.3 EXTI_Type Struct Reference

EXTI Configuration structure for interrupt initialization process.

```
#include "COTS/MCAL/EXTI/EXTI_private.h"
```

Data Fields

- `u32_t IMR`
Interrupt mask register.
- `u32_t EMR`
Event mask register.
- `u32_t RTSR`
Rising trigger status register.
- `u32_t FTSR`
Falling trigger status register.
- `u32_t SWIER`
Software interrupt event register.
- `u32_t PR`
Pending register.

6.3.1 Detailed Description

EXTI Configuration structure for interrupt initialization process.

Definition at line 18 of file [EXTI_private.h](#).

6.3.2 Field Documentation

6.3.2.1 IMR

`u32_t IMR`

Interrupt mask register.

Definition at line 23 of file [EXTI_private.h](#).

6.3.2.2 EMR

`u32_t EMR`

Event mask register.

Definition at line 27 of file [EXTI_private.h](#).

6.3.2.3 RTSR

`u32_t RTSR`

Rising trigger status register.

Definition at line 31 of file [EXTI_private.h](#).

6.3.2.4 FTSR

`u32_t FTSR`

Falling trigger status register.

Definition at line 35 of file [EXTI_private.h](#).

6.3.2.5 SWIER

`u32_t SWIER`

Software interrupt event register.

Definition at line 39 of file [EXTI_private.h](#).

6.3.2.6 PR

`u32_t PR`

Pending register.

Definition at line 43 of file [EXTI_private.h](#).

The documentation for this struct was generated from the following file:

- COTS/MCAL/EXTI/[EXTI_private.h](#)

6.4 GPIOx_MemoryMapType Struct Reference

GPIO declaration structure for its registers.

```
#include "COTS/MCAL/GPIO/GPIO_private.h"
```

Data Fields

- `u32_t MODERx`
*Control PIN mode whether **INPUT** or **OUTPUT***
- `u32_t OTYPERx`
Select the output type between push-pull or open-drain.
- `u32_t OSPEEDRx`
Configure the speed that the PIN is connected to.
- `u32_t PUPDRx`
Control the pull-up or pull-down of the pin, despite its direction.
- `u32_t IDRx`
Read the input data.
- `u32_t ODRx`
Write the output data.
- `u32_t BSRRx`
(Re)set each bit in the output data register
- `u32_t LCKRx`
Lock the GPIO control registers.
- `u32_t AFRLx`
*Control alternate function **LOW** register.*
- `u32_t AFRHx`
*Control alternate function **HIGH** register.*

6.4.1 Detailed Description

GPIO declaration structure for its registers.

Definition at line 18 of file [GPIO_private.h](#).

6.4.2 Field Documentation

6.4.2.1 MODERx

`u32_t MODERx`

Control PIN mode whether **INPUT** or **OUTPUT**

Definition at line [23](#) of file [GPIO_private.h](#).

6.4.2.2 OTYPERx

`u32_t OTYPERx`

Select the output type between push-pull or open-drain.

Definition at line [28](#) of file [GPIO_private.h](#).

6.4.2.3 OSPEEDRx

`u32_t OSPEEDRx`

Configure the speed that the PIN is connected to.

Definition at line [33](#) of file [GPIO_private.h](#).

6.4.2.4 PUPDRx

`u32_t PUPDRx`

Control the pull-up or pull-down of the pin, despite its direction.

Definition at line [38](#) of file [GPIO_private.h](#).

6.4.2.5 IDRx

`u32_t IDRx`

Read the input data.

Definition at line [43](#) of file [GPIO_private.h](#).

6.4.2.6 ODRx

`u32_t` ODRx

Write the output data.

Definition at line 48 of file [GPIO_private.h](#).

6.4.2.7 BSRRx

`u32_t` BSRRx

(Re)set each bit in the output data register

Definition at line 53 of file [GPIO_private.h](#).

6.4.2.8 LCKRx

`u32_t` LCKRx

Lock the GPIO control registers.

Definition at line 58 of file [GPIO_private.h](#).

6.4.2.9 AFRLx

`u32_t` AFRLx

Control alternate function **LOW** register.

Definition at line 63 of file [GPIO_private.h](#).

6.4.2.10 AFRHx

`u32_t` AFRHx

Control alternate function **HIGH** register.

Definition at line 68 of file [GPIO_private.h](#).

The documentation for this struct was generated from the following file:

- COTS/MCAL/GPIO/[GPIO_private.h](#)

6.5 MGPIox_ConfigType Struct Reference

MDIO Configuration structure for a specific PIN initialization.

```
#include "COTS/MCAL/GPIO/GPIO_interface.h"
```

Data Fields

- [u8_t Port](#)
Configures a specific GPIO port.
- [u8_t Pin](#)
Configures a specific GPIO pin.
- [u8_t Mode](#)
Configures a specific GPIO mode.
- [u8_t OutputType](#)
Configures a specific GPIO output's type.
- [u8_t OutputSpeed](#)
Configures a specific GPIO output's speed.
- [u8_t InputType](#)
Configures a specific GPIO input type.
- [u8_t AF_Type](#)
Configures a specific GPIO Alternative function.

6.5.1 Detailed Description

MDIO Configuration structure for a specific PIN initialization.

Definition at line [18](#) of file [GPIO_interface.h](#).

6.5.2 Field Documentation

6.5.2.1 Port

[u8_t Port](#)

Configures a specific GPIO port.

Definition at line [23](#) of file [GPIO_interface.h](#).

Referenced by [MGPIox_vInit\(\)](#).

6.5.2.2 Pin

`u8_t Pin`

Configures a specific GPIO pin.

Definition at line 27 of file [GPIO_interface.h](#).

Referenced by [MGPIOx_vInit\(\)](#).

6.5.2.3 Mode

`u8_t Mode`

Configures a specific GPIO mode.

Definition at line 31 of file [GPIO_interface.h](#).

Referenced by [MGPIOx_vInit\(\)](#).

6.5.2.4 OutputType

`u8_t OutputType`

Configures a specific GPIO output's type.

Definition at line 35 of file [GPIO_interface.h](#).

Referenced by [MGPIOx_vInit\(\)](#).

6.5.2.5 OutputSpeed

`u8_t OutputSpeed`

Configures a specific GPIO output's speed.

Definition at line 39 of file [GPIO_interface.h](#).

Referenced by [MGPIOx_vInit\(\)](#).

6.5.2.6 InputType

`u8_t` `InputType`

Configures a specific GPIO input type.

Definition at line 43 of file [GPIO_interface.h](#).

Referenced by [MGPIOx_vInit\(\)](#).

6.5.2.7 AF_Type

`u8_t` `AF_Type`

Configures a specific GPIO Alternative function.

Definition at line 47 of file [GPIO_interface.h](#).

Referenced by [MGPIOx_vInit\(\)](#).

The documentation for this struct was generated from the following file:

- COTS/MCAL.GPIO/[GPIO_interface.h](#)

6.6 MSYSCFG_MemMap_t Struct Reference

System configuration structure to initialize the SYSCFG module with.

```
#include "COTS/MCAL/EXTI/SYSCFG_private.h"
```

Data Fields

- `u32_t MEMRMP`
Memory remap register.
- `u32_t PMC`
Peripheral mode configuration register.
- `u32_t EXTICR [4]`
External interrupt configuration 1-4 registers.
- `u32_t CMPCR`
Compensation cell control register.

6.6.1 Detailed Description

System configuration structure to initialize the SYSCFG module with.

Definition at line 19 of file [SYSCFG_private.h](#).

6.6.2 Field Documentation

6.6.2.1 MEMRMP

`u32_t` MEMRMP

Memory remap register.

This register is used for specific configurations on memory map

Definition at line 25 of file [SYSCFG_private.h](#).

6.6.2.2 PMC

`u32_t` PMC

Peripheral mode configuration register.

Definition at line 29 of file [SYSCFG_private.h](#).

6.6.2.3 EXTICR

`u32_t` EXTICR[4]

External interrupt configuration 1-4 registers.

Definition at line 33 of file [SYSCFG_private.h](#).

6.6.2.4 CMPCR

`u32_t` CMPCR

Compensation cell control register.

Definition at line 37 of file [SYSCFG_private.h](#).

The documentation for this struct was generated from the following file:

- COTS/MCAL/EXTI/[SYSCFG_private.h](#)

6.7 NVIC_MemoryMapType Struct Reference

NVIC configuration structure for NVIC memory map.

```
#include "COTS/MCAL/NVIC/NVIC_private.h"
```

Data Fields

- `u32_t ISERx [8]`
Interrupt set-enable register.
- `u32_t RESERVED0x [24]`
- `u32_t ICERx [8]`
Interrupt clear-enable register.
- `u32_t RSERVED1x [24]`
- `u32_t ISPRx [8]`
Interrupt set-pending register.
- `u32_t RESERVED2x [24]`
- `u32_t ICPRx [8]`
Interrupt clear-pending register.
- `u32_t RESERVED3x [24]`
- `u32_t IABRx [8]`
Interrupt active bit register.
- `u32_t RESERVED4x [56]`
- `u8_t IPRx [240]`
Interrupt priority register.

6.7.1 Detailed Description

NVIC configuration structure for NVIC memory map.

Definition at line 18 of file [NVIC_private.h](#).

6.7.2 Field Documentation

6.7.2.1 ISERx

`u32_t ISERx [8]`

Interrupt set-enable register.

This register is responsible for enabling the interrupt

Note

Writing 0 has no effect at all

Writing 1 enables the interrupt

See also

[ICERx](#)

Definition at line 27 of file [NVIC_private.h](#).

6.7.2.2 RESERVED0x

`u32_t` RESERVED0x [24]

Note

This is a reserved register

Definition at line 31 of file [NVIC_private.h](#).

6.7.2.3 ICERx

`u32_t` ICERx [8]

Interrupt clear-enable register.

This register is responsible for disabling the interrupt

Note

Writing 0 has no effect at all

Writing 1 disables the interrupt

See also

[ISERx](#)

Definition at line 39 of file [NVIC_private.h](#).

6.7.2.4 RSERVED1x

`u32_t` RSERVED1x [24]

Note

This is a reserved register

Definition at line 43 of file [NVIC_private.h](#).

6.7.2.5 ISPRx

`u32_t ISPRx[8]`

Interrupt set-pending register.

Mostly used in testing the NVIC module

Note

Writing 0 has no effect at all

Writing 1 changes the interrupt state to pending

Writing 1 on a certain bit that already has a value of 1, it stays pending and the value doesn't change

See also

[ICPRx](#)

Definition at line 53 of file [NVIC_private.h](#).

6.7.2.6 RESERVED2x

`u32_t RESERVED2x[24]`

Note

This is a reserved register

Definition at line 57 of file [NVIC_private.h](#).

6.7.2.7 ICPRx

`u32_t ICPRx[8]`

Interrupt clear-pending register.

Mostly used in testing the NVIC module

Note

Writing 0 has no effect at all

Writing 1 removes the pending interrupt state

Writing 1 on a certain bit that already has a value of 1, it stays pending and the value doesn't change

See also

[ISPRx](#)

Definition at line 67 of file [NVIC_private.h](#).

6.7.2.8 RESERVED3x

`u32_t` RESERVED3x[24]

Note

This is a reserved register

Definition at line 71 of file [NVIC_private.h](#).

6.7.2.9 IABRx

`u32_t` IABRx[8]

Interrupt active bit register.

Note

This is a read-only register

Definition at line 77 of file [NVIC_private.h](#).

6.7.2.10 RESERVED4x

`u32_t` RESERVED4x[56]

Note

This is a reserved register

Definition at line 81 of file [NVIC_private.h](#).

6.7.2.11 IPRx

`u8_t` IPRx[240]

Interrupt priority register.

See also

[Group priorities](#)

[Sub-Group priorities](#)

Note

Interrupt priority register

Definition at line 89 of file [NVIC_private.h](#).

The documentation for this struct was generated from the following file:

- COTS/MCAL/NVIC/[NVIC_private.h](#)

6.8 RCC_MemoryMapType Struct Reference

RCC declaration structure for its registers.

```
#include "COTS/MCAL/RCC/MRCC_private.h"
```

Data Fields

- **u32_t CR**
clock control register
- **u32_t PLLCFGR**
PLL configuration register.
- **u32_t CFGR**
clock configuration register
- **u32_t CIR**
clock interrupt register
- **u32_t AHB1RSTR**
AHB1 peripheral reset register.
- **u32_t AHB2RSTR**
AHB2 peripheral reset register.
- **u32_t Reserved1**
Reserved register.
- **u32_t Reserved2**
Reserved register.
- **u32_t APB1RSTR**
APB1 peripheral reset register.
- **u32_t APB2RSTR**
APB2 peripheral reset register.
- **u32_t Reserved3**
Reserved register.
- **u32_t Reserved4**
Reserved register.
- **u32_t AHB1ENR**
AHB1 peripheral clock enable register.
- **u32_t AHB2ENR**
AHB2 peripheral clock enable register.
- **u32_t Reserved5**
Reserved register.
- **u32_t Reserved6**
Reserved register.
- **u32_t APB1ENR**
APB1 peripheral clock enable register.
- **u32_t APB2ENR**
APB2 peripheral clock enable register.
- **u32_t Reserved7**
Reserved register.
- **u32_t Reserved8**
Reserved register.
- **u32_t AHB1LPENR**
AHB1 peripheral clock enable in low power mode register.

- `u32_t AHB2LPENR`
AHB2 peripheral clock enable in low power mode register.
- `u32_t Reserved9`
Reserved register.
- `u32_t Reserved10`
Reserved register.
- `u32_t APB1LPENR`
APB1 peripheral clock enable in low power mode register.
- `u32_t APB2LPENR`
APB2 peripheral clock enable in low power mode register.
- `u32_t Reserved11`
Reserved register.
- `u32_t Reserved12`
Reserved register.
- `u32_t BDCR`
RCC Backup domain control register.
- `u32_t CSR`
clock control & status register
- `u32_t Reserved13`
Reserved register.
- `u32_t Reserved14`
Reserved register.
- `u32_t SSCGR`
RCC spread spectrum clock generation register.
- `u32_t PLLI2SCFGR`
PLLI2S configuration register.
- `u32_t DCKCFGR`
Dedicated Clocks Configuration Register.

6.8.1 Detailed Description

RCC declaration structure for its registers.

Definition at line 24 of file [MRCC_private.h](#).

6.8.2 Field Documentation

6.8.2.1 CR

`u32_t CR`

clock control register

Definition at line 30 of file [MRCC_private.h](#).

6.8.2.2 PLLCFGR

`u32_t` PLLCFGR

PLL configuration register.

Definition at line 35 of file [MRCC_private.h](#).

6.8.2.3 CFGR

`u32_t` CFGR

clock configuration register

Definition at line 40 of file [MRCC_private.h](#).

6.8.2.4 CIR

`u32_t` CIR

clock interrupt register

Definition at line 45 of file [MRCC_private.h](#).

6.8.2.5 AHB1RSTR

`u32_t` AHB1RSTR

AHB1 peripheral reset register.

Definition at line 50 of file [MRCC_private.h](#).

6.8.2.6 AHB2RSTR

`u32_t` AHB2RSTR

AHB2 peripheral reset register.

Definition at line 55 of file [MRCC_private.h](#).

6.8.2.7 Reserved1

`u32_t` Reserved1

Reserved register.

Definition at line 60 of file [MRCC_private.h](#).

6.8.2.8 Reserved2

`u32_t` Reserved2

Reserved register.

Definition at line 65 of file [MRCC_private.h](#).

6.8.2.9 APB1RSTR

`u32_t` APB1RSTR

APB1 peripheral reset register.

Definition at line 70 of file [MRCC_private.h](#).

6.8.2.10 APB2RSTR

`u32_t` APB2RSTR

APB2 peripheral reset register.

Definition at line 75 of file [MRCC_private.h](#).

6.8.2.11 Reserved3

`u32_t` Reserved3

Reserved register.

Definition at line 80 of file [MRCC_private.h](#).

6.8.2.12 Reserved4

`u32_t` Reserved4

Reserved register.

Definition at line 85 of file [MRCC_private.h](#).

6.8.2.13 AHB1ENR

`u32_t` AHB1ENR

AHB1 peripheral clock enable register.

Definition at line 90 of file [MRCC_private.h](#).

6.8.2.14 AHB2ENR

`u32_t` AHB2ENR

AHB2 peripheral clock enable register.

Definition at line 95 of file [MRCC_private.h](#).

6.8.2.15 Reserved5

`u32_t` Reserved5

Reserved register.

Definition at line 100 of file [MRCC_private.h](#).

6.8.2.16 Reserved6

`u32_t` Reserved6

Reserved register.

Definition at line 105 of file [MRCC_private.h](#).

6.8.2.17 APB1ENR

`u32_t` APB1ENR

APB1 peripheral clock enable register.

Definition at line 110 of file [MRCC_private.h](#).

6.8.2.18 APB2ENR

`u32_t` APB2ENR

APB2 peripheral clock enable register.

Definition at line 115 of file [MRCC_private.h](#).

6.8.2.19 Reserved7

`u32_t` Reserved7

Reserved register.

Definition at line 120 of file [MRCC_private.h](#).

6.8.2.20 Reserved8

`u32_t` Reserved8

Reserved register.

Definition at line 125 of file [MRCC_private.h](#).

6.8.2.21 AHB1LPENR

`u32_t` AHB1LPENR

AHB1 peripheral clock enable in low power mode register.

Definition at line 130 of file [MRCC_private.h](#).

6.8.2.22 AHB2LPENR

`u32_t` AHB2LPENR

AHB2 peripheral clock enable in low power mode register.

Definition at line 135 of file [MRCC_private.h](#).

6.8.2.23 Reserved9

`u32_t` Reserved9

Reserved register.

Definition at line 140 of file [MRCC_private.h](#).

6.8.2.24 Reserved10

`u32_t` Reserved10

Reserved register.

Definition at line 145 of file [MRCC_private.h](#).

6.8.2.25 APB1LPENR

`u32_t` APB1LPENR

APB1 peripheral clock enable in low power mode register.

Definition at line 150 of file [MRCC_private.h](#).

6.8.2.26 APB2LPENR

`u32_t` APB2LPENR

APB2 peripheral clock enable in low power mode register.

Definition at line 155 of file [MRCC_private.h](#).

6.8.2.27 Reserved11

`u32_t` Reserved11

Reserved register.

Definition at line 160 of file [MRCC_private.h](#).

6.8.2.28 Reserved12

`u32_t` Reserved12

Reserved register.

Definition at line 165 of file [MRCC_private.h](#).

6.8.2.29 BDCR

`u32_t` BDCR

RCC Backup domain control register.

Definition at line 170 of file [MRCC_private.h](#).

6.8.2.30 CSR

`u32_t` CSR

clock control & status register

Definition at line 175 of file [MRCC_private.h](#).

6.8.2.31 Reserved13

`u32_t` Reserved13

Reserved register.

Definition at line 180 of file [MRCC_private.h](#).

6.8.2.32 Reserved14

`u32_t` Reserved14

Reserved register.

Definition at line 185 of file [MRCC_private.h](#).

6.8.2.33 SSCGR

`u32_t` SSCGR

RCC spread spectrum clock generation register.

Definition at line 190 of file [MRCC_private.h](#).

6.8.2.34 PLLI2SCFGR

`u32_t` PLLI2SCFGR

PLLI2S configuration register.

Definition at line 195 of file [MRCC_private.h](#).

6.8.2.35 DCKCFGR

`u32_t` DCKCFGR

Dedicated Clocks Configuration Register.

Definition at line 200 of file [MRCC_private.h](#).

The documentation for this struct was generated from the following file:

- COTS/MCAL/RCC/[MRCC_private.h](#)

6.9 SCB_MemoryMapType Struct Reference

System control block memory map structure.

```
#include "COTS/MCAL/NVIC/NVIC_private.h"
```

Data Fields

- [u32_t CPUID](#)
CPUID base register.
- [u32_t ICSR](#)
Interrupt control and state register.
- [u32_t VTOR](#)
Vector table offset register.
- [u32_t AIRCR](#)
Application interrupt and reset control register.
- [u32_t SCR](#)
System control register.
- [u32_t CCR](#)
configuration and control register
- [u32_t SHPR1](#)
System handler priority register 1.
- [u32_t SHPR2](#)
System handler priority register 2.
- [u32_t SHPR3](#)
System handler priority register 3.
- [u32_t SHCSR](#)
System handler control and state register.
- [u32_t CFSR](#)
Configurable fault status register.
- [u32_t HFSR](#)
Hart fault status register.
- [u32_t RESERVED](#)
- [u32_t MMFAR](#)
Memory management fault address register.
- [u32_t BFAR](#)
Bus fault address register.

6.9.1 Detailed Description

System control block memory map structure.

provides system implementation information, and system control.

This includes configuration, control, and reporting of the system exceptions

Definition at line 100 of file [NVIC_private.h](#).

6.9.2 Field Documentation

6.9.2.1 CPUID

`u32_t CPUID`

CPUID base register.

This register contains the the processor part number, version, and implementation information

Definition at line 106 of file [NVIC_private.h](#).

6.9.2.2 ICSR

`u32_t ICSR`

Interrupt control and state register.

This register is responsible for enabling the interrupt

Definition at line 111 of file [NVIC_private.h](#).

6.9.2.3 VTOR

`u32_t VTOR`

Vector table offset register.

Interrupt control and state register

Definition at line 116 of file [NVIC_private.h](#).

6.9.2.4 AIRCR

`u32_t AIRCR`

Application interrupt and reset control register.

This register provides priority grouping control for the exception model, endian status for data accesses, and reset control for the system

To write on this register, password 0x5FA must be written to the VECTKEY field, otherwise the write attempt is ignored

See also

[VECTKEY_PASSWORD](#) for the VECTKEY field password value

Definition at line 123 of file [NVIC_private.h](#).

6.9.2.5 SCR

`u32_t SCR`

System control register.

The SCR controls features of entry to and exit from low power state

Definition at line 128 of file [NVIC_private.h](#).

6.9.2.6 CCR

`u32_t CCR`

configuration and control register

The CCR controls entry to Thread mode and enables:

- The handlers for NMI, hard fault and faults escalated by FAULTMASK to ignore bus faults
- Trapping of divide by zero and unaligned accesses
- Access to the STIR by unprivileged software

Definition at line 136 of file [NVIC_private.h](#).

6.9.2.7 SHPR1

`u32_t SHPR1`

System handler priority register 1.

The SHPR1-SHPR3 registers set the priority level, 0 to 255 of the exception handlers that have configurable priority.

See also

[SHPR2](#)

[SHPR3](#)

Definition at line 143 of file [NVIC_private.h](#).

6.9.2.8 SHPR2

`u32_t SHPR2`

System handler priority register 2.

The SHPR1-SHPR3 registers set the priority level, 0 to 255 of the exception handlers that have configurable priority.

See also

[SHPR1](#)

[SHPR3](#)

Definition at line 150 of file [NVIC_private.h](#).

6.9.2.9 SHPR3

`u32_t SHPR3`

System handler priority register 3.

The SHPR1-SHPR3 registers set the priority level, 0 to 255 of the exception handlers that have configurable priority.

See also

[SHPR1](#)

[SHPR2](#)

Definition at line 157 of file [NVIC_private.h](#).

6.9.2.10 SHCSR

`u32_t SHCSR`

System handler control and state register.

The SHCSR enables the system handlers, and indicates:

- The pending status of the bus fault, memory management fault, and SVC exceptions
- The active status of the system handlers.

Note

If you disable a system handler and the corresponding fault occurs, the processor treats the fault as a hard fault.

Definition at line 165 of file [NVIC_private.h](#).

6.9.2.11 CFSR

`u32_t` CFSR

Configurable fault status register.

This register consists of 3 sub-registers:

- Usage Fault Status Register (UFSR)
- Bus Fault Status Register (BFSR)
- Memory Management Fault Status Register (MMFSR)
System handler control and state register

Definition at line 176 of file [NVIC_private.h](#).

6.9.2.12 HFSR

`u32_t` HFSR

Hard fault status register.

The HFSR gives information about events that activate the hard fault handler.

This register is read, write to clear.

This means that bits in the register read normally, but writing 1 to any bit clears that bit to 0

Definition at line 183 of file [NVIC_private.h](#).

6.9.2.13 RESERVED

`u32_t` RESERVED

Note

This is a reserved register

Definition at line 187 of file [NVIC_private.h](#).

6.9.2.14 MMFAR

`u32_t` MMFAR

Memory management fault address register.

Memory management fault address register

Definition at line 192 of file [NVIC_private.h](#).

6.9.2.15 BFAR

`u32_t` BFAR

Bus fault address register.

Bus fault address register

Definition at line 197 of file [NVIC_private.h](#).

The documentation for this struct was generated from the following file:

- COTS/MCAL/NVIC/[NVIC_private.h](#)

6.10 SysTick_Type Struct Reference

Systick memory map structure declaration for the Systick's registers.

```
#include "COTS/MCAL/SysTick/SysTick_private.h"
```

Data Fields

- `u32_t CTRL`
Systick control and status register.
- `u32_t LOAD`
Systick reload value register.
- `u32_t VAL`
Systick current value register.
- `u32_t CALIB`
Systick calibration value register.

6.10.1 Detailed Description

Systick memory map structure declaration for the Systick's registers.

Definition at line 18 of file [SysTick_private.h](#).

6.10.2 Field Documentation

6.10.2.1 CTRL

`u32_t CTRL`

Systick control and status register.

This register is responsible for enabling the systick features

Definition at line 24 of file [SysTick_private.h](#).

6.10.2.2 LOAD

`u32_t LOAD`

Systick reload value register.

This register is responsible for setting the countdown value

Definition at line 29 of file [SysTick_private.h](#).

6.10.2.3 VAL

`u32_t VAL`

Systick current value register.

This register holds the current value for the Systick counter

Definition at line 34 of file [SysTick_private.h](#).

6.10.2.4 CALIB

`u32_t CALIB`

Systick calibration value register.

SysTick calibration value register

Definition at line 39 of file [SysTick_private.h](#).

The documentation for this struct was generated from the following file:

- COTS/MCAL/SysTick/[SysTick_private.h](#)

Chapter 7

File Documentation

7.1 D:/GProject/ADAS_Project/README.md File Reference

7.2 COTS/HAL/DCMOTOR/DCM_config.h File Reference

This file contains the configuration information for the DC Motor module.

7.2.1 Detailed Description

This file contains the configuration information for the DC Motor module.

Author

Mohamed Alaa

Version

1.0

Date

8/12/2022

Definition in file [DCM_config.h](#).

7.3 DCM_config.h

[Go to the documentation of this file.](#)

```
00001
00009 /* Header file guard */
00010 #ifndef _DCM_config_H
00011 #define _DCM_config_H
00012
00013 #endif // _DCM_config_H
```

7.4 COTS/HAL/DCMOTOR/DCM_interface.h File Reference

This file contains the interfacing information for the DC Motor module.

Data Structures

- struct `DCM_MotorConfiguration`

DC Motor configuration structure for motor initialization DC Motor configuration for motor initialization for a certain port and pin. Since DC Motors have 2 sides to be controller, we will initialize each motor with 2 pins. Taking into consideration that the pins could be on different ports.

Functions

- void `HDCM_vInitMotor (P2VAR(DCM_MotorConfiguration) pMotorConfiguration)`
Initialize a motor Initialize a motor with a certain GPIO Port & Pin based on the passed motor configuration.
- void `HDCM_vMoveForward (P2VAR(DCM_MotorConfiguration) pMotorConfiguration)`
Move a certain motor forward.
- void `HDCM_vMoveBackward (P2VAR(DCM_MotorConfiguration) pMotorConfiguration)`
Move a certain motor backward.
- void `HDCM_vStopMotor (P2VAR(DCM_MotorConfiguration) pMotorConfiguration)`
Stop a certain motor's movement completely.

7.4.1 Detailed Description

This file contains the interfacing information for the DC Motor module.

Author

Mohamed Alaa

Version

1.0

Date

8/12/2022

Definition in file `DCM_interface.h`.

7.4.2 Function Documentation

7.4.2.1 HDCM_vInitMotor()

```
void HDCM_vInitMotor (
    P2VAR(DCM_MotorConfiguration) pMotorConfiguration )
```

Initialize a motor Initialize a motor with a certain GPIO Port & Pin based on the passed motor configuration.

Parameters

in	<i>pMotorConfiguration</i>	The motor configuration to be initialized
----	----------------------------	---

See also[DCM_MotorConfiguration](#)**7.4.2.2 HDCM_vMoveForward()**

```
void HDCM_vMoveForward (
    P2VAR(DCM_MotorConfiguration) pMotorConfiguration )
```

Move a certain motor forward.

Parameters

in	<i>pMotorConfiguration</i>	The motor's configuration to move forward
----	----------------------------	---

7.4.2.3 HDCM_vMoveBackward()

```
void HDCM_vMoveBackward (
    P2VAR(DCM_MotorConfiguration) pMotorConfiguration )
```

Move a certain motor backward.

Parameters

in	<i>pMotorConfiguration</i>	The motor's configuration to move backward
----	----------------------------	--

7.4.2.4 HDCM_vStopMotor()

```
void HDCM_vStopMotor (
    P2VAR(DCM_MotorConfiguration) pMotorConfiguration )
```

Stop a certain motor's movement completely.

Parameters

in	<i>pMotorConfiguration</i>	The motor's configuration to stop
----	----------------------------	-----------------------------------

7.5 DCM_interface.h

[Go to the documentation of this file.](#)

```
00001
00009 /* Header file guard */
00010 #ifndef _DCM_interface_H
00011 #define _DCM_interface_H
00012
00020 typedef struct
00021 {
00025     u8_t Side1Port;
00029     u8_t Side1Pin;
00033     u8_t Side2Port;
00037     u8_t Side2Pin;
00038 } DCM_MotorConfiguration;
00039
00040 /***** Functions prototypes *****/
00041 /* Functions prototypes */
00042 /*****
00043
00050 void HDCM_vInitMotor(P2VAR(DCM_MotorConfiguration) pMotorConfiguration);
00051
00056 void HDCM_vMoveForward(P2VAR(DCM_MotorConfiguration) pMotorConfiguration);
00057
00062 void HDCM_vMoveBackward(P2VAR(DCM_MotorConfiguration) pMotorConfiguration);
00063
00068 void HDCM_vStopMotor(P2VAR(DCM_MotorConfiguration) pMotorConfiguration);
00069
00070 #endif // _DCM_interface_H
```

7.6 COTS/HAL/DCMOTOR/DCM_private.h File Reference

This file contains the private information for the DC Motor module.

7.6.1 Detailed Description

This file contains the private information for the DC Motor module.

Author

Mohamed Alaa

Version

1.0

Date

8/12/2022

Definition in file [DCM_private.h](#).

7.7 DCM_private.h

[Go to the documentation of this file.](#)

```
00001
00009 /* Header file guard */
00010 #ifndef _DCM_private_H
00011 #define _DCM_private_H
00012
00013 #endif // _DCM_private_H
```

7.8 COTS/HAL/DCMOTOR/DCM_program.c File Reference

This file contains the source code of the interfacing information for the DC Motor module.

```
#include "../../LIB/LSTD_TYPES.h"
#include "../../LIB/LSTD_COMPILER.h"
#include "../../LIB/LSTD_VALUES.h"
#include "../../LIB/LSTD_BITMATH.h"
#include "../../MCAL/GPIO/GPIO_interface.h"
#include "DCM_private.h"
#include "DCM_interface.h"
#include "DCM_config.h"
```

Functions

- void [HDCM_vInitMotor](#) (DCM_MotorConfiguration *pMotorConfiguration)
- void [HDCM_vMoveForward](#) (DCM_MotorConfiguration *pMotorConfiguration)
- void [HDCM_vMoveBackward](#) (DCM_MotorConfiguration *pMotorConfiguration)
- void [HDCM_vStopMotor](#) (DCM_MotorConfiguration *pMotorConfiguration)

7.8.1 Detailed Description

This file contains the source code of the interfacing information for the DC Motor module.

Author

Mohamed Alaa

Version

1.0

Date

8/12/2022

Definition in file [DCM_program.c](#).

7.8.2 Function Documentation

7.8.2.1 HDCM_vInitMotor()

```
void HDCM_vInitMotor (
    DCM_MotorConfiguration * pMotorConfiguration )
```

Definition at line 31 of file [DCM_program.c](#).

```
00032 {
00033     VAR(MGPIox_ConfigType) DCMotorSide1Init =
00034     {
00035         .Port = pMotorConfiguration->Side1Port,
00036         .Pin = pMotorConfiguration->Side1Pin,
00037         .Mode = GPIOx_MODE_OUTPUT,
00038         .OutputType = GPIOx_PUSH_PULL,
00039         .OutputSpeed = GPIOx_LowSpeed,
00040         .InputType = GPIOx_NoPull
00041     };
00042
00043     VAR(MGPIox_ConfigType) DCMotorSide2Init =
00044     {
00045         .Port = pMotorConfiguration->Side2Port,
00046         .Pin = pMotorConfiguration->Side2Pin,
00047         .Mode = GPIOx_MODE_OUTPUT,
00048         .OutputType = GPIOx_PUSH_PULL,
00049         .OutputSpeed = GPIOx_LowSpeed,
00050         .InputType = GPIOx_NoPull
00051     };
00052
00053     MGPIox_vInit(&DCMotorSide1Init);
00054     MGPIox_vInit(&DCMotorSide2Init);
00055 }
```

References [GPIOx_LowSpeed](#), [GPIOx_MODE_OUTPUT](#), [GPIOx_NoPull](#), [GPIOx_PUSH_PULL](#), [MGPIox_vInit\(\)](#), [DCM_MotorConfiguration::Side1Pin](#), [DCM_MotorConfiguration::Side1Port](#), [DCM_MotorConfiguration::Side2Pin](#), [DCM_MotorConfiguration::Side2Port](#), and [VAR](#).

7.8.2.2 HDCM_vMoveForward()

```
void HDCM_vMoveForward (
    DCM_MotorConfiguration * pMotorConfiguration )
```

Definition at line 57 of file [DCM_program.c](#).

```
00058 {
00059     MGPIox_vSetValue(pMotorConfiguration->Side1Port, pMotorConfiguration->Side1Pin, GPIOx_HIGH);
00060     MGPIox_vSetValue(pMotorConfiguration->Side2Port, pMotorConfiguration->Side2Pin, GPIOx_LOW);
00061 }
```

References [GPIOx_HIGH](#), [GPIOx_LOW](#), [MGPIox_vSetValue\(\)](#), [DCM_MotorConfiguration::Side1Pin](#), [DCM_MotorConfiguration::Side1Port](#), [DCM_MotorConfiguration::Side2Pin](#), and [DCM_MotorConfiguration::Side2Port](#).

7.8.2.3 HDCM_vMoveBackward()

```
void HDCM_vMoveBackward (
    DCM_MotorConfiguration * pMotorConfiguration )
```

Definition at line 63 of file [DCM_program.c](#).

```
00064 {
00065     MGPIox_vSetValue(pMotorConfiguration->Side1Port, pMotorConfiguration->Side1Pin, GPIOx_LOW);
00066     MGPIox_vSetValue(pMotorConfiguration->Side2Port, pMotorConfiguration->Side2Pin, GPIOx_HIGH);
00067 }
```

References [GPIOx_HIGH](#), [GPIOx_LOW](#), [MGPIox_vSetValue\(\)](#), [DCM_MotorConfiguration::Side1Pin](#), [DCM_MotorConfiguration::Side1Port](#), [DCM_MotorConfiguration::Side2Pin](#), and [DCM_MotorConfiguration::Side2Port](#).

7.8.2.4 HDCM_vStopMotor()

```
void HDCM_vStopMotor (
    DCM_MotorConfiguration * pMotorConfiguration )
```

Definition at line 69 of file [DCM_program.c](#).

```
00070 {
00071     GPIOx_vSetValue(pMotorConfiguration->Side1Port, pMotorConfiguration->Side1Pin, GPIOx_LOW);
00072     GPIOx_vSetValue(pMotorConfiguration->Side2Port, pMotorConfiguration->Side2Pin, GPIOx_LOW);
00073 }
```

References [GPIOx_LOW](#), [GPIOx_vSetValue\(\)](#), [DCM_MotorConfiguration::Side1Pin](#), [DCM_MotorConfiguration::Side1Port](#), [DCM_MotorConfiguration::Side2Pin](#), and [DCM_MotorConfiguration::Side2Port](#).

7.9 DCM_program.c

[Go to the documentation of this file.](#)

```
00001
00009 /*****
0010  *          Include headers
0011 *****/
0012
0013 #include "../../LIB/LSTD_TYPES.h"
0014 #include "../../LIB/LSTD_COMPILER.h"
0015 #include "../../LIB/LSTD_VALUES.h"
0016 #include "../../LIB/LSTD_BITMATH.h"
0017 #include "../../MCAL/GPIO/GPIO_interface.h"
0018 #include "DCM_private.h"
0019 #include "DCM_interface.h"
0020 #include "DCM_config.h"
0021
0022 STATIC P2VAR(DCM_MotorConfiguration) GS_pMotor1;
0023 STATIC P2VAR(DCM_MotorConfiguration) GS_pMotor2;
0024 STATIC P2VAR(DCM_MotorConfiguration) GS_pMotor3;
0025 STATIC P2VAR(DCM_MotorConfiguration) GS_pMotor4;
0026
0027 /*****
0028  *          Functions' implementations
0029 *****/
0030
0031 FUNC(void) HDCM_vInitMotor(P2VAR(DCM_MotorConfiguration) pMotorConfiguration)
0032 {
0033     VAR(GPIOx_ConfigType) DCMotorSide1Init =
0034     {
0035         .Port = pMotorConfiguration->Side1Port,
0036         .Pin = pMotorConfiguration->Side1Pin,
0037         .Mode = GPIOx_MODE_OUTPUT,
0038         .OutputType = GPIOx_PUSH_PULL,
0039         .OutputSpeed = GPIOx_lowSpeed,
0040         .InputType = GPIOx_NoPull
0041     };
0042
0043     VAR(GPIOx_ConfigType) DCMotorSide2Init =
0044     {
0045         .Port = pMotorConfiguration->Side2Port,
0046         .Pin = pMotorConfiguration->Side2Pin,
0047         .Mode = GPIOx_MODE_OUTPUT,
0048         .OutputType = GPIOx_PUSH_PULL,
0049         .OutputSpeed = GPIOx_lowSpeed,
0050         .InputType = GPIOx_NoPull
0051     };
0052
0053     GPIOx_vInit(&DCMotorSide1Init);
0054     GPIOx_vInit(&DCMotorSide2Init);
0055 }
0056
0057 FUNC(void) HDCM_vMoveForward(P2VAR(DCM_MotorConfiguration) pMotorConfiguration)
0058 {
0059     GPIOx_vSetValue(pMotorConfiguration->Side1Port, pMotorConfiguration->Side1Pin, GPIOx_HIGH);
0060     GPIOx_vSetValue(pMotorConfiguration->Side2Port, pMotorConfiguration->Side2Pin, GPIOx_LOW);
0061 }
0062
0063 FUNC(void) HDCM_vMoveBackward(P2VAR(DCM_MotorConfiguration) pMotorConfiguration)
0064 {
0065     GPIOx_vSetValue(pMotorConfiguration->Side1Port, pMotorConfiguration->Side1Pin, GPIOx_LOW);
0066     GPIOx_vSetValue(pMotorConfiguration->Side2Port, pMotorConfiguration->Side2Pin, GPIOx_HIGH);
```

```

00067 }
00068
00069 FUNC(void) HDCM_vStopMotor(P2VAR(DCM_MotorConfiguration) pMotorConfiguration)
00070 {
00071     MGPIOx_vSetPinValue(pMotorConfiguration->Side1Port, pMotorConfiguration->Side1Pin, GPIOx_LOW);
00072     MGPIOx_vSetPinValue(pMotorConfiguration->Side2Port, pMotorConfiguration->Side2Pin, GPIOx_LOW);
00073 }

```

7.10 COTS/LIB/LSTD_BITMATH.h File Reference

This file contains the bit math manipulation macro-functions.

Macros

- #define [SET_BIT](#)(Reg, bitnum) (Reg) |= (1 << (bitnum))
- #define [CLR_BIT](#)(Reg, bitnum) (Reg) &= ~(1 << (bitnum))
- #define [TOGGLE_BIT](#)(Reg, bitnum) (Reg) ^= (1 << (bitnum))
- #define [GET_BIT](#)(Reg, bitnum) (((Reg)>>(bitnum)) & 1)
- #define [SET_BITS](#)(Reg, bits, bitnum, factor) (Reg) |= ((bits) << (bitnum * factor))
- #define [CLR_BITS](#)(Reg, bits, bitnum, factor) (Reg) &= ~((bits) << (bitnum * factor))
- #define [TOGGLE_BITS](#)(Reg, bits, bitnum, factor) (Reg) ^= ((bits) << (bitnum * factor))
- #define [GET_BITS](#)(Reg, bits, bitnum, factor) (((Reg) >> (bitnum * factor)) & (bits))

7.10.1 Detailed Description

This file contains the bit math manipulation macro-functions.

Author

Ali El Bana

Version

1.0

Date

10/29/2022

Definition in file [LSTD_BITMATH.h](#).

7.11 LSTD_BITMATH.h

[Go to the documentation of this file.](#)

```

00001
00009 #ifndef COTS_LIB_LSTD_BITMATH_H_
00010 #define COTS_LIB_LSTD_BITMATH_H_
00011
00023 #define SET_BIT(Reg, bitnum) (Reg) |= (1 << (bitnum))
00024
00030 #define CLR_BIT(Reg, bitnum) (Reg) &= ~(1 << (bitnum))
00031
00037 #define TOGGLE_BIT(Reg, bitnum) (Reg) ^= (1 << (bitnum))
00038
00044 #define GET_BIT(Reg, bitnum) (((Reg)>>(bitnum)) & 1)
00045
00059 #define SET_BITS(Reg, bits, bitnum, factor) (Reg) |= ((bits) << (bitnum * factor))
00060
00066 #define CLR_BITS(Reg, bits, bitnum, factor) (Reg) &= ~((bits) << (bitnum * factor))
00067
00073 #define TOGGLE_BITS(Reg, bits, bitnum, factor) (Reg) ^= ((bits) << (bitnum * factor))
00074
00080 #define GET_BITS(Reg, bits, bitnum, factor) (((Reg) >> (bitnum * factor)) & (bits))
00081
00084 #endif /* COTS_LIB_LSTD_BITMATH_H_ */

```

7.12 COTS/LIB/LSTD_COMPILER.h File Reference

This file contains the compiler standard macros.

Macros

- #define **VAR**(vartype) vartype
- #define **FUNC**(rettype) rettype
- #define **P2VAR**(ptrtype) ptrtype *
- #define **P2CONST**(ptrtype) const ptrtype *
- #define **CONSTP2VAR**(ptrtype) ptrtype * const
- #define **CONSTP2CONST**(ptrtype) const ptrtype * const
- #define **P2FUNC**(rettype, fctname) rettype (*fctname)
- #define **CONST**(consttype) const consttype
- #define **STATIC** static

7.12.1 Detailed Description

This file contains the compiler standard macros.

Author

Mohamed Alaa

Version

1.0

Date

11/04/2022

Definition in file [LSTD_COMPILER.h](#).

7.13 LSTD_COMPILER.h

[Go to the documentation of this file.](#)

```
00001
00009 #ifndef COTS_LIB_LSTD_COMPILER_H_
00010 #define COTS_LIB_LSTD_COMPILER_H_
00011
00023 #define VAR(vartype) vartype
00024
00030 #define FUNC(rettype) rettype
00031
00037 #define P2VAR(ptrtype) ptrtype *
00038
00044 #define P2CONST(ptrtype) const ptrtype *
00045
00051 #define CONSTP2VAR(ptrtype) ptrtype * const
00052
00058 #define CONSTP2CONST(ptrtype) const ptrtype * const
00059
00065 #define P2FUNC(rettype, fctname) rettype (*fctname)
00066
00072 #ifndef CONST
00073 #define CONST(consttype) const consttype
00074 #endif
00075
00081 #ifndef STATIC
00082 #define STATIC static
00083 #endif
00084
00087 #endif /* COTS_LIB_LSTD_COMPILER_H_ */
```

7.14 COTS/LIB/LSTD MCU UTILITIES.h File Reference

This file contains the MCU utility macro-functions.

Macros

- #define MY_MS_DELAY(T) do{ u32 Timer = (T * 500); while (Timer--) { asm ("nop"); } } while(0);

7.14.1 Detailed Description

This file contains the MCU utility macro-functions.

Author

Ali El Bana

Version

1.0

Date

10/29/2022

Definition in file [LSTD MCU UTILITIES.h](#).

7.15 LSTD MCU UTILITIES.h

[Go to the documentation of this file.](#)

```
00001
00009 #ifndef COTS_LIB_LSTD MCU UTILITIES_H_
00010 #define COTS_LIB_LSTD MCU UTILITIES_H_
00023 #define MY_MS_DELAY(T)    do{ u32 Timer = (T * 500); while (Timer--) { asm ("nop"); } } while(0);
00024
00027 #endif /* COTS_LIB_LSTD MCU UTILITIES_H_ */
```

7.16 COTS/LIB/LSTD TYPES.h File Reference

This file contains the standard types.

Typedefs

- typedef unsigned char [bool_t](#)
- typedef unsigned char [u8_t](#)
- typedef unsigned short int [u16_t](#)
- typedef unsigned long int [u32_t](#)
- typedef signed char [s8_t](#)
- typedef signed short int [s16_t](#)
- typedef signed long int [s32_t](#)
- typedef float [f32_t](#)
- typedef double [f64_t](#)

7.16.1 Detailed Description

This file contains the standard types.

Author

Ali El Bana

Version

1.0

Date

10/29/2022

Definition in file [LSTD_TYPES.h](#).

7.17 LSTD_TYPES.h

[Go to the documentation of this file.](#)

```
00001
00009 #ifndef COTS_LIB_LSTD_TYPES_H_
00010 #define COTS_LIB_LSTD_TYPES_H_
00011
00023 typedef unsigned char bool_t;
00024
00030 typedef unsigned char u8_t;
00031
00037 typedef unsigned short int u16_t;
00038
00044 typedef unsigned long int u32_t;
00045
00051 typedef signed char s8_t;
00052
00058 typedef signed short int s16_t;
00059
00065 typedef signed long int s32_t;
00066
00072 typedef float f132_t;
00073
00079 typedef double f164_t;
00080
00083 #endif /* COTS_LIB_LSTD_TYPES_H_ */
```

7.18 COTS/LIB/LSTD_VALUES.h File Reference

This file contains the standard values.

Macros

- `#define TRUE (1)`
- `#define FALSE (0)`
- `#define NULL ((P2VAR(void))0)`
- `#define INITIAL_ZERO (0)`
- `#define FLAG_SET (1)`
- `#define FLAG_CLEARED (0)`
- `#define RUN (1)`
- `#define STOP (0)`
- `#define PRESSED (1)`
- `#define RELEASED (0)`

7.18.1 Detailed Description

This file contains the standard values.

Author

Ali El Bana

Version

1.0

Date

10/29/2022

Definition in file [LSTD_VALUES.h](#).

7.19 LSTD_VALUES.h

[Go to the documentation of this file.](#)

```

00001
00009 #ifndef COTS_LIB_LSTD_VALUES_H_
00010 #define COTS_LIB_LSTD_VALUES_H_
00011
00018 #ifndef TRUE
00024 #define TRUE (1)
00025 #endif /* ifndef TRUE */
00026
00027 #ifndef FALSE
00033 #define FALSE (0)
00034 #endif /* ifndef FALSE */
00035
00036 #ifndef NULL
00043 #define NULL ((P2VAR(void))0)
00044 #endif /* ifndef NULL */
00045
00046 #ifndef NULL
00047 #define NULL ((P2VAR(void))0)
00048 #endif /* ifndef NULL */
00049
00050
00051 #ifndef INITIAL_ZERO
00057 #define INITIAL_ZERO (0)
00058 #endif /* ifndef INITIAL_ZERO */
00059
00060 #ifndef FLAG_SET
00066 #define FLAG_SET (1)
00067 #endif /* ifndef FLAG_SET */
00068
00069 #ifndef FLAG_CLEARED
00075 #define FLAG_CLEARED (0)
00076 #endif /* ifndef FLAG_CLEARED */
00077
00083 #ifndef RUN
00084 #define RUN (1)
00085 #endif /* ifndef RUN */
00086
00087
00093 #ifndef STOP
00094 #define STOP (0)
00095 #endif /* ifndef STOP */
00096
00102 #ifndef PRESSED
00103 #define PRESSED (1)
00104 #endif /* ifndef PRESSED */
00105
00106
00112 #ifndef RELEASED
00113 #define RELEASED (0)
00114 #endif /* ifndef RELEASED */
00115
00116
00119 #endif /* COTS_LIB_LSTD_VALUES_H_ */

```

7.20 COTS/MCAL/EXTI/EXTI_config.h File Reference

Macros

- #define EXTI_LINE0_EN ENABLE
- #define EXTI_LINE0_TRIGGER EXTI_FallingEdge
- #define EXTI_LINE1_EN ENABLE
- #define EXTI_LINE1_TRIGGER EXTI_RisingEdge
- #define EXTI_LINE2_EN ENABLE
- #define EXTI_LINE2_TRIGGER EXTI_RisingEdge
- #define EXTI_LINE3_EN ENABLE
- #define EXTI_LINE3_TRIGGER EXTI_RisingEdge
- #define EXTI_LINE4_EN ENABLE
- #define EXTI_LINE4_TRIGGER EXTI_RisingEdge
- #define EXTI_LINE5_EN ENABLE
- #define EXTI_LINE5_TRIGGER EXTI_RisingEdge
- #define EXTI_LINE6_EN ENABLE
- #define EXTI_LINE6_TRIGGER EXTI_RisingEdge
- #define EXTI_LINE7_EN ENABLE
- #define EXTI_LINE7_TRIGGER EXTI_RisingEdge
- #define EXTI_LINE8_EN ENABLE
- #define EXTI_LINE8_TRIGGER EXTI_RisingEdge
- #define EXTI_LINE9_EN ENABLE
- #define EXTI_LINE9_TRIGGER EXTI_RisingEdge
- #define EXTI_LINE10_EN ENABLE
- #define EXTI_LINE10_TRIGGER EXTI_RisingEdge
- #define EXTI_LINE11_EN ENABLE
- #define EXTI_LINE11_TRIGGER EXTI_RisingEdge
- #define EXTI_LINE12_EN ENABLE
- #define EXTI_LINE12_TRIGGER EXTI_RisingEdge
- #define EXTI_LINE13_EN ENABLE
- #define EXTI_LINE13_TRIGGER EXTI_RisingEdge
- #define EXTI_LINE14_EN ENABLE
- #define EXTI_LINE14_TRIGGER EXTI_OnChange
- #define EXTI_LINE15_EN ENABLE
- #define EXTI_LINE15_TRIGGER EXTI_RisingEdge

7.20.1 Macro Definition Documentation

7.20.1.1 EXTI_LINE0_EN

```
#define EXTI_LINE0_EN ENABLE
```

Definition at line 18 of file [EXTI_config.h](#).

7.20.1.2 EXTI_LINE0_TRIGGER

```
#define EXTI_LINE0_TRIGGER EXTI_FallingEdge
```

Options: EXTI_FallingEdge , EXTI_RisingEdge , EXTI_OnChange

Definition at line [28](#) of file [EXTI_config.h](#).

7.20.1.3 EXTI_LINE1_EN

```
#define EXTI_LINE1_EN ENABLE
```

Definition at line [36](#) of file [EXTI_config.h](#).

7.20.1.4 EXTI_LINE1_TRIGGER

```
#define EXTI_LINE1_TRIGGER EXTI_RisingEdge
```

Options: EXTI_FallingEdge , EXTI_RisingEdge , EXTI_OnChange

Definition at line [46](#) of file [EXTI_config.h](#).

7.20.1.5 EXTI_LINE2_EN

```
#define EXTI_LINE2_EN ENABLE
```

Definition at line [54](#) of file [EXTI_config.h](#).

7.20.1.6 EXTI_LINE2_TRIGGER

```
#define EXTI_LINE2_TRIGGER EXTI_RisingEdge
```

Options: EXTI_FallingEdge , EXTI_RisingEdge , EXTI_OnChange

Definition at line [64](#) of file [EXTI_config.h](#).

7.20.1.7 EXTI_LINE3_EN

```
#define EXTI_LINE3_EN ENABLE
```

Definition at line [72](#) of file [EXTI_config.h](#).

7.20.1.8 EXTI_LINE3_TRIGGER

```
#define EXTI_LINE3_TRIGGER EXTI_RisingEdge
```

Options: EXTI_FallingEdge , EXTI_RisingEdge , EXTI_OnChange

Definition at line [82](#) of file [EXTI_config.h](#).

7.20.1.9 EXTI_LINE4_EN

```
#define EXTI_LINE4_EN ENABLE
```

Definition at line [90](#) of file [EXTI_config.h](#).

7.20.1.10 EXTI_LINE4_TRIGGER

```
#define EXTI_LINE4_TRIGGER EXTI_RisingEdge
```

Options: EXTI_FallingEdge , EXTI_RisingEdge , EXTI_OnChange

Definition at line [100](#) of file [EXTI_config.h](#).

7.20.1.11 EXTI_LINE5_EN

```
#define EXTI_LINE5_EN ENABLE
```

Definition at line [108](#) of file [EXTI_config.h](#).

7.20.1.12 EXTI_LINE5_TRIGGER

```
#define EXTI_LINE5_TRIGGER EXTI_RisingEdge
```

Options: EXTI_FallingEdge , EXTI_RisingEdge , EXTI_OnChange

Definition at line [118](#) of file [EXTI_config.h](#).

7.20.1.13 EXTI_LINE6_EN

```
#define EXTI_LINE6_EN ENABLE
```

Definition at line 126 of file [EXTI_config.h](#).

7.20.1.14 EXTI_LINE6_TRIGGER

```
#define EXTI_LINE6_TRIGGER EXTI_RisingEdge
```

Options: EXTI_FallingEdge , EXTI_RisingEdge , EXTI_OnChange

Definition at line 136 of file [EXTI_config.h](#).

7.20.1.15 EXTI_LINE7_EN

```
#define EXTI_LINE7_EN ENABLE
```

Definition at line 144 of file [EXTI_config.h](#).

7.20.1.16 EXTI_LINE7_TRIGGER

```
#define EXTI_LINE7_TRIGGER EXTI_RisingEdge
```

Options: EXTI_FallingEdge , EXTI_RisingEdge , EXTI_OnChange

Definition at line 154 of file [EXTI_config.h](#).

7.20.1.17 EXTI_LINE8_EN

```
#define EXTI_LINE8_EN ENABLE
```

Definition at line 162 of file [EXTI_config.h](#).

7.20.1.18 EXTI_LINE8_TRIGGER

```
#define EXTI_LINE8_TRIGGER EXTI_RisingEdge
```

Options: EXTI_FallingEdge , EXTI_RisingEdge , EXTI_OnChange

Definition at line 172 of file [EXTI_config.h](#).

7.20.1.19 EXTI_LINE9_EN

```
#define EXTI_LINE9_EN ENABLE
```

Definition at line 180 of file [EXTI_config.h](#).

7.20.1.20 EXTI_LINE9_TRIGGER

```
#define EXTI_LINE9_TRIGGER EXTI_RisingEdge
```

Options: EXTI_FallingEdge , EXTI_RisingEdge , EXTI_OnChange

Definition at line 190 of file [EXTI_config.h](#).

7.20.1.21 EXTI_LINE10_EN

```
#define EXTI_LINE10_EN ENABLE
```

Definition at line 198 of file [EXTI_config.h](#).

7.20.1.22 EXTI_LINE10_TRIGGER

```
#define EXTI_LINE10_TRIGGER EXTI_RisingEdge
```

Options: EXTI_FallingEdge , EXTI_RisingEdge , EXTI_OnChange

Definition at line 208 of file [EXTI_config.h](#).

7.20.1.23 EXTI_LINE11_EN

```
#define EXTI_LINE11_EN ENABLE
```

Definition at line 216 of file [EXTI_config.h](#).

7.20.1.24 EXTI_LINE11_TRIGGER

```
#define EXTI_LINE11_TRIGGER EXTI_RisingEdge
```

Options: EXTI_FallingEdge , EXTI_RisingEdge , EXTI_OnChange

Definition at line 226 of file [EXTI_config.h](#).

7.20.1.25 EXTI_LINE12_EN

```
#define EXTI_LINE12_EN ENABLE
```

Definition at line [234](#) of file [EXTI_config.h](#).

7.20.1.26 EXTI_LINE12_TRIGGER

```
#define EXTI_LINE12_TRIGGER EXTI_RisingEdge
```

Options: EXTI_FallingEdge , EXTI_RisingEdge , EXTI_OnChange

Definition at line [244](#) of file [EXTI_config.h](#).

7.20.1.27 EXTI_LINE13_EN

```
#define EXTI_LINE13_EN ENABLE
```

Definition at line [251](#) of file [EXTI_config.h](#).

7.20.1.28 EXTI_LINE13_TRIGGER

```
#define EXTI_LINE13_TRIGGER EXTI_RisingEdge
```

Options: EXTI_FallingEdge , EXTI_RisingEdge , EXTI_OnChange

Definition at line [261](#) of file [EXTI_config.h](#).

7.20.1.29 EXTI_LINE14_EN

```
#define EXTI_LINE14_EN ENABLE
```

Definition at line [269](#) of file [EXTI_config.h](#).

7.20.1.30 EXTI_LINE14_TRIGGER

```
#define EXTI_LINE14_TRIGGER EXTI_OnChange
```

Options: EXTI_FallingEdge , EXTI_RisingEdge , EXTI_OnChange

Definition at line [279](#) of file [EXTI_config.h](#).

7.20.1.31 EXTI_LINE15_EN

```
#define EXTI_LINE15_EN ENABLE
```

Definition at line 287 of file [EXTI_config.h](#).

7.20.1.32 EXTI_LINE15_TRIGGER

```
#define EXTI_LINE15_TRIGGER EXTI_RisingEdge
```

Options: EXTI_FallingEdge , EXTI_RisingEdge , EXTI_OnChange

Definition at line 297 of file [EXTI_config.h](#).

7.21 EXTI_config.h

[Go to the documentation of this file.](#)

```
00001 /* FILENAME: EXTI_config
00002 * Author: Ali El Bana
00003 * Version: V1.0
00004 * DATE: Fri 09/02/2022
00005 */
00006 #ifndef _EXTI_config_H
00007 #define _EXTI_config_H
00008
00009
00010 /***** // EXTI Configurations *****/
00011
00012 /***** // Options *****/
00013
00014 /*options:
00015 *ENABLE
00016 *DISABLE
00017 */
00018 #define EXTI_LINE0_EN ENABLE
00019
00020 /***** // Line 0 *****/
00021
00022 #define EXTI_LINE0_TRIGGER EXTI_FallingEdge
00023
00024 /***** // Line 1 *****/
00025
00026 #define EXTI_LINE1_EN ENABLE
00027
00028 /***** // Line 2 *****/
00029
00030 #define EXTI_LINE2_TRIGGER EXTI_RisingEdge
00031
00032 /*options:
00033 *ENABLE
00034 *DISABLE
00035 */
00036 #define EXTI_LINE1_TRIGGER EXTI_RisingEdge
00037
00038 /***** // Line 3 *****/
00039
00040 #define EXTI_LINE3_TRIGGER EXTI_RisingEdge
00041
00042 /***** // Line 4 *****/
00043
00044 #define EXTI_LINE4_TRIGGER EXTI_RisingEdge
00045
00046 /***** // Line 5 *****/
00047
00048 #define EXTI_LINE5_TRIGGER EXTI_RisingEdge
00049
00050 /*options:
00051 *ENABLE
00052 *DISABLE
00053 */
00054 #define EXTI_LINE6_TRIGGER EXTI_RisingEdge
00055
00056 /***** // Line 7 *****/
00057
00058 #define EXTI_LINE7_TRIGGER EXTI_RisingEdge
00059
00060 /***** // Line 8 *****/
00061
00062 #define EXTI_LINE8_TRIGGER EXTI_RisingEdge
00063
00064 /***** // Line 9 *****/
00065
00066 #define EXTI_LINE9_TRIGGER EXTI_RisingEdge
00067
00068 /*options:
00069 *ENABLE
00070 *DISABLE
```

```
00071 */
00072 #define EXTI_LINE3_EN ENABLE
00073
00074 /*****
00075
00076 #define EXTI_LINE3_TRIGGER EXTI_RisingEdge
00077
00078 /*****
00079
00080 /*options:
00081 *ENABLE
00082 *DISABLE
00083 */
00084
00085
00086 #define EXTI_LINE4_EN ENABLE
00087
00088 /*****
00089
00090 #define EXTI_LINE4_TRIGGER EXTI_RisingEdge
00091
00092 /*****
00093
00094 #define EXTI_LINE5_EN ENABLE
00095
00096 /*****
00097
00098 #define EXTI_LINE5_TRIGGER EXTI_RisingEdge
00099
00100 /*****
00101
00102 /*****
00103
00104 /*options:
00105 *ENABLE
00106 *DISABLE
00107 */
00108 #define EXTI_LINE6_EN ENABLE
00109
00110 /*****
00111
00112 #define EXTI_LINE6_TRIGGER EXTI_RisingEdge
00113
00114 /*****
00115
00116 /*options:
00117 *ENABLE
00118 *DISABLE
00119 */
00120 /*****
00121
00122 /*options:
00123 *ENABLE
00124 *DISABLE
00125 */
00126 #define EXTI_LINE7_EN ENABLE
00127
00128 /*****
00129
00130 #define EXTI_LINE7_TRIGGER EXTI_RisingEdge
00131
00132 /*****
00133
00134 /*options:
00135 *ENABLE
00136 *DISABLE
00137 */
00138 /*****
00139
00140 /*options:
00141 *ENABLE
00142 *DISABLE
00143 */
00144 #define EXTI_LINE8_EN ENABLE
00145
00146 /*****
00147
00148 /*options:
00149 *ENABLE
00150 *DISABLE
00151 */
00152 #define EXTI_LINE8_TRIGGER EXTI_RisingEdge
00153
00154 /*****
00155
00156 /*****
00157
00158 /*options:
00159 *ENABLE
00160 *DISABLE
00161 */
00162 #define EXTI_LINE9_EN ENABLE
00163
00164 /*****
00165
00166 #define EXTI_LINE9_TRIGGER EXTI_RisingEdge
00167
00168 /*****
00169
00170 /*options:
00171 *ENABLE
00172 *DISABLE
00173 */
00174 /*****
00175
00176 /*options:
00177 *ENABLE
00178 *DISABLE
00179 */
00180 #define EXTI_LINE10_EN ENABLE
00181
00182 /*****
00183
00184 #define EXTI_LINE10_TRIGGER EXTI_RisingEdge
00185
00186 /*****
00187
00188 /*options:
00189 *ENABLE
00190 *DISABLE
00191 */
00192 /*****
00193
00194 /*options:
00195 *ENABLE
00196 *DISABLE
00197 */
00198 #define EXTI_LINE10_TRIGGER EXTI_RisingEdge
00199
```

```
00200 /******  
00201  
00208 #define EXTI_LINE10_TRIGGER EXTI_RisingEdge  
00209  
00210 /******  
00211  
00212 /*options:  
00213 *ENABLE  
00214 *DISABLE  
00215 */  
00216 #define EXTI_LINE11_EN ENABLE  
00217  
00218 /******  
00219  
00226 #define EXTI_LINE11_TRIGGER EXTI_RisingEdge  
00227  
00228 /******  
00229  
00230 /*options:  
00231 *ENABLE  
00232 *DISABLE  
00233 */  
00234 #define EXTI_LINE12_EN ENABLE  
00235  
00236 /******  
00237  
00244 #define EXTI_LINE12_TRIGGER EXTI_RisingEdge  
00245  
00246 /******  
00247 /*options:  
00248 *ENABLE  
00249 *DISABLE  
00250 */  
00251 #define EXTI_LINE13_EN ENABLE  
00252  
00253 /******  
00254  
00261 #define EXTI_LINE13_TRIGGER EXTI_RisingEdge  
00262  
00263 /******  
00264  
00265 /*options:  
00266 *ENABLE  
00267 *DISABLE  
00268 */  
00269 #define EXTI_LINE14_EN ENABLE  
00270  
00271 /******  
00272  
00279 #define EXTI_LINE14_TRIGGER EXTI_OnChange  
00280  
00281 /******  
00282  
00283 /*options:  
00284 *ENABLE  
00285 *DISABLE  
00286 */  
00287 #define EXTI_LINE15_EN ENABLE  
00288  
00289 /******  
00290  
00297 #define EXTI_LINE15_TRIGGER EXTI_RisingEdge  
00298  
00299 /******  
00300  
00301  
00302  
00303  
00304  
00305  
00306  
00307  
00308  
00309  
00310  
00311  
00312  
00313  
00314  
00315 #endif // _EXTI_config_H
```

7.22 COTS/MCAL/EXTI/EXTI_interface.h File Reference

This file contains the interfacing information for the EXTI module.

Data Structures

- struct `EXTI_ConfigType`
Interrupt configuration structure to initialize the interrupt with.

Macros

- `#define EXTI_LINE0 (0)`
Line 0.
- `#define EXTI_LINE1 (1)`
Line 1.
- `#define EXTI_LINE2 (2)`
Line 2.
- `#define EXTI_LINE3 (3)`
Line 3.
- `#define EXTI_LINE4 (4)`
Line 4.
- `#define EXTI_LINE5 (5)`
Line 5.
- `#define EXTI_LINE6 (6)`
Line 6.
- `#define EXTI_LINE7 (7)`
Line 7.
- `#define EXTI_LINE8 (8)`
Line 8.
- `#define EXTI_LINE9 (9)`
Line 9.
- `#define EXTI_LINE10 (10)`
Line 10.
- `#define EXTI_LINE11 (11)`
Line 11.
- `#define EXTI_LINE12 (12)`
Line 12.
- `#define EXTI_LINE13 (13)`
Line 13.
- `#define EXTI_LINE14 (14)`
Line 14.
- `#define EXTI_LINE15 (15)`
Line 15.
- `#define EXTI_FallingEdge (1)`
trigger interrupt on falling edge
- `#define EXTI_RisingEdge (2)`
trigger interrupt on rising edge
- `#define EXTI_OnChange (3)`
trigger interrupt on level change

Functions

- void [MEXTI_vInit \(void\)](#)
Initialize the EXTI module.
- void [MEXTI_vInit_WithStruct \(P2VAR\(EXTI_ConfigType\) A_xINTConfig\)](#)
Initialize the EXTI module with a certain configuration.
- void [MEXTI_vEnableLine \(VAR\(u8_t\) A_u8LineID, VAR\(u8_t\) A_u8TriggerStatus\)](#)
Enable the interrupt on a certain line ID with a certain trigger status.
- void [MEXTI_vDisableLine \(VAR\(u8_t\) A_u8LineID\)](#)
Disable the interrupt on a certain line ID.
- void [MEXTI_vSWITrigger \(VAR\(u8_t\) A_u8LineID\)](#)
perform a software event interrupt on a certain line
- void [MEXTI_vSetTrigger \(VAR\(u8_t\) A_u8LineID, VAR\(u8_t\) A_u8TriggerStatus\)](#)
Set the trigger status interrupt on a certain line ID.
- void [MEXTI_vSetCallback \(VAR\(u8_t\) A_u8LineID, void\(*A_vFptr\)\(void\)\)](#)
Set the callback when an interrupt occurs on a certain line ID.
- void [MSYSCFG_vSetEXTIPort \(VAR\(u8_t\) A_u8LineID, VAR\(u8_t\) A_u8PortID\)](#)
Enable the interrupt on a certain line in a certain port.

7.22.1 Detailed Description

This file contains the interfacing information for the EXTI module.

Author

Ali El Bana

Version

1.0

Date

09/02/2022

Definition in file [EXTI_interface.h](#).

7.22.2 Function Documentation

7.22.2.1 MEXTI_vInit()

```
void MEXTI_vInit (
    void )
```

Initialize the EXTI module.

Definition at line 42 of file [EXTI_program.c](#).

```
00043 {
00044     VAR(u8_t) L_u8ShiftedOffset = INITIAL_ZERO;
00046
00047 #if EXTI_LINE0_EN == ENABLE
00048     L_u8ShiftedOffset = 0;
00049 #elif EXTI_LINE1_EN == ENABLE
00050     L_u8ShiftedOffset = 1;
00051 #elif EXTI_LINE2_EN == ENABLE
00052     L_u8ShiftedOffset = 2;
00053 #elif EXTI_LINE3_EN == ENABLE
00054     L_u8ShiftedOffset = 3;
00055 #elif EXTI_LINE4_EN == ENABLE
00056     L_u8ShiftedOffset = 4;
00057 #elif EXTI_LINE5_EN == ENABLE
00058     L_u8ShiftedOffset = 5;
00059 #elif EXTI_LINE6_EN == ENABLE
00060     L_u8ShiftedOffset = 6;
00061 #elif EXTI_LINE7_EN == ENABLE
00062     L_u8ShiftedOffset = 7;
00063 #elif EXTI_LINE8_EN == ENABLE
00064     L_u8ShiftedOffset = 8;
00065 #elif EXTI_LINE9_EN == ENABLE
00066     L_u8ShiftedOffset = 9;
00067 #elif EXTI_LINE10_EN == ENABLE
00068     L_u8ShiftedOffset = 10;
00069 #elif EXTI_LINE11_EN == ENABLE
00070     L_u8ShiftedOffset = 11;
00071 #elif EXTI_LINE12_EN == ENABLE
00072     L_u8ShiftedOffset = 12;
00073 #elif EXTI_LINE13_EN == ENABLE
00074     L_u8ShiftedOffset = 13;
00075 #elif EXTI_LINE14_EN == ENABLE
00076     L_u8ShiftedOffset = 14;
00077 #elif EXTI_LINE15_EN == ENABLE
00078     L_u8ShiftedOffset = 15;
00079 #endif
00080
00081     // Get the enabled line ID.
00082     // GS_u8EXTI_EnabledLine_ID = L_u8ShiftedOffset;
00083
00084     // Clear all flags.
00085     MEXTI->PR = 0xffffffff;
00086
00087     // Enable EXTI on a line.
00088     MEXTI->IMR = 0;
00089
00090     MEXTI->IMR |= (ENABLE << L_u8ShiftedOffset);
00091
00092 // Set EXTI Triggered status on a line.
00093 #if EXTI_LINE0_TRIGGER == EXTI_RisingEdge
00094
00095     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00096
00097 #elif EXTI_LINE0_TRIGGER == EXTI_FallingEdge
00098     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00099
00100 #elif EXTI_LINE0_TRIGGER == EXTI_OnChange
00101     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00102     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00103
00105 #elif EXTI_LINE1_TRIGGER == EXTI_RisingEdge
00106     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00107
00108 #elif EXTI_LINE1_TRIGGER == EXTI_FallingEdge
00109     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00110
00111 #elif EXTI_LINE1_TRIGGER == EXTI_OnChange
00112     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00113     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00114
00116 #elif EXTI_LINE2_TRIGGER == EXTI_RisingEdge
00117     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00118
00119 #elif EXTI_LINE2_TRIGGER == EXTI_FallingEdge
```

```

00120     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00121
00122 #elif EXTI_LINE2_TRIGGER == EXTI_OnChange
00123     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00124     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00125
00127 #elif EXTI_LINE3_TRIGGER == EXTI_RisingEdge
00128     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00129
00130 #elif EXTI_LINE3_TRIGGER == EXTI_FallingEdge
00131     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00132
00133 #elif EXTI_LINE3_TRIGGER == EXTI_OnChange
00134     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00135     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00136
00138 #elif EXTI_LINE4_TRIGGER == EXTI_RisingEdge
00139     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00140
00141 #elif EXTI_LINE4_TRIGGER == EXTI_FallingEdge
00142     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00143
00144 #elif EXTI_LINE4_TRIGGER == EXTI_OnChange
00145     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00146     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00147
00149 #elif EXTI_LINES_TRIGGER == EXTI_RisingEdge
00150     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00151
00152 #elif EXTI_LINES_TRIGGER == EXTI_FallingEdge
00153     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00154
00155 #elif EXTI_LINES_TRIGGER == EXTI_OnChange
00156     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00157     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00158
00160 #elif EXTI_LINE6_TRIGGER == EXTI_RisingEdge
00161     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00162
00163 #elif EXTI_LINE6_TRIGGER == EXTI_FallingEdge
00164     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00165
00166 #elif EXTI_LINE6_TRIGGER == EXTI_OnChange
00167     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00168     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00169
00171 #elif EXTI_LINE7_TRIGGER == EXTI_RisingEdge
00172     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00173
00174 #elif EXTI_LINE7_TRIGGER == EXTI_FallingEdge
00175     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00176
00177 #elif EXTI_LINE7_TRIGGER == EXTI_OnChange
00178     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00179     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00180
00182 #elif EXTI_LINE8_TRIGGER == EXTI_RisingEdge
00183     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00184
00185 #elif EXTI_LINE8_TRIGGER == EXTI_FallingEdge
00186     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00187
00188 #elif EXTI_LINE8_TRIGGER == EXTI_OnChange
00189     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00190     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00191
00193 #elif EXTI_LINE9_TRIGGER == EXTI_RisingEdge
00194     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00195
00196 #elif EXTI_LINE9_TRIGGER == EXTI_FallingEdge
00197     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00198
00199 #elif EXTI_LINE9_TRIGGER == EXTI_OnChange
00200     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00201     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00202
00204 #elif EXTI_LINE10_TRIGGER == EXTI_RisingEdge
00205     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00206
00207 #elif EXTI_LINE10_TRIGGER == EXTI_FallingEdge
00208     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00209
00210 #elif EXTI_LINE10_TRIGGER == EXTI_OnChange
00211     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00212     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00213
00215 #elif EXTI_LINE11_TRIGGER == EXTI_RisingEdge

```

```

00216     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00217
00218 #elif EXTI_LINE11_TRIGGER == EXTI_FallingEdge
00219     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00220
00221 #elif EXTI_LINE11_TRIGGER == EXTI_OnChange
00222     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00223     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00224
00225 #elif EXTI_LINE12_TRIGGER == EXTI_RisingEdge
00226     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00227
00228 #elif EXTI_LINE12_TRIGGER == EXTI_FallingEdge
00229     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00230
00231 #elif EXTI_LINE12_TRIGGER == EXTI_OnChange
00232     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00233     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00234
00235 #elif EXTI_LINE13_TRIGGER == EXTI_RisingEdge
00236     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00237
00238 #elif EXTI_LINE13_TRIGGER == EXTI_FallingEdge
00239     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00240
00241 #elif EXTI_LINE13_TRIGGER == EXTI_OnChange
00242     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00243     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00244
00245 #elif EXTI_LINE14_TRIGGER == EXTI_RisingEdge
00246     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00247
00248 #elif EXTI_LINE14_TRIGGER == EXTI_FallingEdge
00249     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00250
00251 #elif EXTI_LINE14_TRIGGER == EXTI_OnChange
00252     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00253
00254 #elif EXTI_LINE14_TRIGGER == EXTI_RisingEdge
00255     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00256     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00257
00258 #elif EXTI_LINE15_TRIGGER == EXTI_RisingEdge
00259     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00260
00261 #elif EXTI_LINE15_TRIGGER == EXTI_FallingEdge
00262     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00263
00264 #elif EXTI_LINE15_TRIGGER == EXTI_OnChange
00265     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00266     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00267
00268 #endif
00270
00271 }

```

References [ENABLE](#), [INITIAL_ZERO](#), [MEXTI](#), [SET_BIT](#), and [VAR](#).

7.22.2.2 MEXTI_vInit_WithStruct()

```

void MEXTI_vInit_WithStruct (
    P2VAR(EXTI_ConfigType) A_xINTConfig )

```

Initialize the EXTI module with a certain configuration.

Parameters

in	A_xINTConfig	The configuration structure to initialize the interrupt with
----	--------------	--

7.22.2.3 MEXTI_vEnableLine()

```
void MEXTI_vEnableLine (
    VAR(u8_t) A_u8LineID,
    VAR(u8_t) A_u8TriggerStatus )
```

Enable the interrupt on a certain line ID with a certain trigger status.

Parameters

in	<i>A_u8LineID</i>	The line ID to enable the interrupt on
in	<i>A_u8TriggerStatus</i>	The trigger status to trigger when the interrupt occurs

7.22.2.4 MEXTI_vDisableLine()

```
void MEXTI_vDisableLine (
    VAR(u8_t) A_u8LineID )
```

Disable the interrupt on a certain line ID.

Parameters

in	<i>A_u8LineID</i>	Which line to disable the interrupt on
----	-------------------	--

7.22.2.5 MEXTI_vSWITrigger()

```
void MEXTI_vSWITrigger (
    VAR(u8_t) A_u8LineID )
```

perform a software event interrupt on a certain line

Parameters

in	<i>A_u8LineID</i>	The line ID to perform the software event interrupt on
----	-------------------	--

7.22.2.6 MEXTI_vSetTrigger()

```
void MEXTI_vSetTrigger (
    VAR(u8_t) A_u8LineID,
    VAR(u8_t) A_u8TriggerStatus )
```

Set the trigger status interrupt on a certain line ID.

Parameters

in	<i>A_u8LineID</i>	The line ID to enable the interrupt on
in	<i>A_u8TriggerStatus</i>	The trigger status to trigger when the interrupt occurs

7.22.2.7 MEXTI_vSetCallback()

```
void MEXTI_vSetCallback (
    VAR(u8_t) A_u8LineID,
    void(*)(void) A_vFptr )
```

Set the callback when an interrupt occurs on a certain line ID.

Parameters

in	<i>A_u8LineID</i>	The line ID to set the callback on
in	<i>A_vFptr</i>	The callback to call when the interrupt occurs on a certain line

7.22.2.8 MSYSCFG_vSetEXTIPort()

```
void MSYSCFG_vSetEXTIPort (
    VAR(u8_t) A_u8LineID,
    VAR(u8_t) A_u8PortID )
```

Enable the interrupt on a certain line in a certain port.

Parameters

in	<i>A_u8LineID</i>	The line ID to enable the interrupt onto
in	<i>A_u8PortID</i>	Port that the line belongs to

7.23 EXTI_interface.h

[Go to the documentation of this file.](#)

```
00001
00009 /* Header file guard */
00010 #ifndef _EXTI_interface_H
00011 #define _EXTI_interface_H
00012
00018 typedef struct
00019 {
00023     u8_t LineNum;
00024
00028     u8_t PortNum ;
00029
00033     u8_t TriggerStatus;
00034
```

```

00035
00036
00037 } EXTI_ConfigType;
00038
00039 /***** Functions prototypes *****/
00040 /* Functions prototypes */
00041 /*****
00042
00043 void MEXTI_vInit(void);
00044 void MEXTI_vInit_WithStruct(P2VAR(EXTI_ConfigType) A_xINTConfig);
00045
00046 void MEXTI_vEnableLine(VAR(u8_t) A_u8LineID, VAR(u8_t) A_u8TriggerStatus);
00047
00048 void MEXTI_vDisableLine(VAR(u8_t) A_u8LineID);
00049
00050 void MEXTI_vSWITrigger(VAR(u8_t) A_u8LineID);
00051
00052 void MEXTI_vSetTrigger(VAR(u8_t) A_u8LineID, VAR(u8_t) A_u8TriggerStatus);
00053
00054 void MEXTI_vSetCallback(VAR(u8_t) A_u8LineID, void (*A_vFptr)(void));
00055
00056 void MSYSCFG_vSetEXTIPort(VAR(u8_t) A_u8LineID, VAR(u8_t) A_u8PortID);
00057
00058
00059 #define EXTI_LINE0 (0)
00060 #define EXTI_LINE1 (1)
00061 #define EXTI_LINE2 (2)
00062 #define EXTI_LINE3 (3)
00063 #define EXTI_LINE4 (4)
00064 #define EXTI_LINE5 (5)
00065 #define EXTI_LINE6 (6)
00066 #define EXTI_LINE7 (7)
00067 #define EXTI_LINE8 (8)
00068 #define EXTI_LINE9 (9)
00069 #define EXTI_LINE10 (10)
00070 #define EXTI_LINE11 (11)
00071 #define EXTI_LINE12 (12)
00072 #define EXTI_LINE13 (13)
00073 #define EXTI_LINE14 (14)
00074 #define EXTI_LINE15 (15)
00075
00076
00077 #define EXTI_FallingEdge (1)
00078 #define EXTI_RisingEdge (2)
00079 #define EXTI_OnChange (3)
00080
00081 #endif // _EXTI_interface_H

```

7.24 COTS/MCAL/EXTI/EXTI_private.h File Reference

This file contains the private information related to the EXTI module.

Data Structures

- struct `EXTI_Type`
- EXTI Configuration structure for interrupt initialization process.*

Macros

- `#define EXTI_BASE_ADDRESS (0x40013C00)`
- `#define MEXTI ((volatile P2VAR(EXTI_Type))(EXTI_BASE_ADDRESS))`
- `#define ENABLE (1)`
- `#define DISABLE (2)`
- `#define EXTI_MAX_EXTI_NUM (22)`
- `#define EXTI IRQs 23`

7.24.1 Detailed Description

This file contains the private information related to the EXTI module.

Author

Ali El Bana

Version

1.0

Date

09/02/2022

Definition in file [EXTI_private.h](#).

7.24.2 Macro Definition Documentation

7.24.2.1 EXTI IRQs

```
#define EXTI_IRQs 23
```

Definition at line 109 of file [EXTI_private.h](#).

7.25 EXTI_private.h

[Go to the documentation of this file.](#)

```
00001 /* Header file guard */
00009 #ifndef _EXTI_private_H
00010 #define _EXTI_private_H
00012
00018 typedef struct
00019 {
00023     u32_t IMR;
00027     u32_t EMR;
00031     u32_t RTSR;
00035     u32_t FTSR;
00039     u32_t SWIER;
00043     u32_t PR;
00044 } EXTI_Type;
00045
00057 #define EXTI_BASE_ADDRESS (0x40013C00)
00071 #define MEXTI ((volatile P2VAR(EXTI_Type)) (EXTI_BASE_ADDRESS))
00085 #define ENABLE (1)
00091 #define DISABLE (2)
00105 #define EXTI_MAX_EXTI_NUM (22)
00109 #define EXTI_IRQs 23
00110
00111
00112 #endif // _EXTI_private_H
```

7.26 COTS/MCAL/EXTI/EXTI_program.c File Reference

This file contains the source code of the interfacing for the EXTI module.

```
#include "../../LIB/LSTD_TYPES.h"
#include "../../LIB/LSTD_COMPILER.h"
#include "../../LIB/LSTD_VALUES.h"
#include "../../LIB/LSTD_BITMATH.h"
#include "EXTI_interface.h"
#include "EXTI_private.h"
#include "EXTI_config.h"
#include "SYSCFG_private.h"
```

Functions

- void [MSYSCFG_vSetEXTIPort](#) (u8_t A_u8LineID, u8_t A_u8PortID)
- void [MEXTI_vInit](#) (void)
Initialize the EXTI module.
- void [MEXTI_vInit_WithStruct](#) (EXTI_ConfigType *A_xINTConfig)
- void [MEXTI_vEnableLine](#) (u8_t A_u8LineID, u8_t A_u8TriggerStatus)
- void [MEXTI_vDisableLine](#) (u8_t A_u8LineID)
- void [MEXTI_vSWITrigger](#) (u8_t A_u8LineID)
- void [MEXTI_vSetTrigger](#) (u8_t A_u8LineID, u8_t A_u8TriggerStatus)
- void [MEXTI_vSetCallback](#) (u8_t A_u8LineID, void(*A_vFptr)(void))
- void [EXTI0_IRQHandler](#) (void)
- void [EXTI1_IRQHandler](#) (void)
- void [EXTI2_IRQHandler](#) (void)
- void [EXTI3_IRQHandler](#) (void)
- void [EXTI4_IRQHandler](#) (void)
- void [EXTI9_5_IRQHandler](#) (void)
- void [EXTI15_10_IRQHandler](#) (void)

7.26.1 Detailed Description

This file contains the source code of the interfacing for the EXTI module.

Author

Ali El Bana

Version

1.0

Date

09/02/2022

Definition in file [EXTI_program.c](#).

7.26.2 Function Documentation

7.26.2.1 MSYSCFG_vSetEXTIPort()

```
void MSYSCFG_vSetEXTIPort (
    u8_t A_u8LineID,
    u8_t A_u8PortID )
```

Definition at line 30 of file [EXTI_program.c](#).

```
00031 {
00032     VAR(u8_t) L_u8Index = A_u8LineID / 4;
00033     VAR(u8_t) L_u8ShiftAmount = A_u8LineID % 4;
00034
00035     CLR_BITS(MSYSCFG->EXTICR[L_u8Index], 0b1111, L_u8ShiftAmount, 4);
00036     SET_BITS(MSYSCFG->EXTICR[L_u8Index], A_u8PortID, L_u8ShiftAmount, 4);
00037 }
```

References [CLR_BITS](#), [MSYSCFG](#), [SET_BITS](#), and [VAR](#).

Referenced by [MEXTI_vInit_WithStruct\(\)](#).

7.26.2.2 MEXTI_vInit()

```
void MEXTI_vInit (
    void )
```

Initialize the EXTI module.

Definition at line 42 of file [EXTI_program.c](#).

```
00043 {
00044
00045     VAR(u8_t) L_u8ShiftedOffset = INITIAL_ZERO;
00046
00047 #if EXTI_LINE0_EN == ENABLE
00048     L_u8ShiftedOffset = 0;
00049 #elif EXTI_LINE1_EN == ENABLE
00050     L_u8ShiftedOffset = 1;
00051 #elif EXTI_LINE2_EN == ENABLE
00052     L_u8ShiftedOffset = 2;
00053 #elif EXTI_LINE3_EN == ENABLE
00054     L_u8ShiftedOffset = 3;
00055 #elif EXTI_LINE4_EN == ENABLE
00056     L_u8ShiftedOffset = 4;
00057 #elif EXTI_LINE5_EN == ENABLE
00058     L_u8ShiftedOffset = 5;
00059 #elif EXTI_LINE6_EN == ENABLE
00060     L_u8ShiftedOffset = 6;
00061 #elif EXTI_LINE7_EN == ENABLE
00062     L_u8ShiftedOffset = 7;
00063 #elif EXTI_LINE8_EN == ENABLE
00064     L_u8ShiftedOffset = 8;
00065 #elif EXTI_LINE9_EN == ENABLE
00066     L_u8ShiftedOffset = 9;
00067 #elif EXTI_LINE10_EN == ENABLE
00068     L_u8ShiftedOffset = 10;
00069 #elif EXTI_LINE11_EN == ENABLE
00070     L_u8ShiftedOffset = 11;
00071 #elif EXTI_LINE12_EN == ENABLE
00072     L_u8ShiftedOffset = 12;
00073 #elif EXTI_LINE13_EN == ENABLE
00074     L_u8ShiftedOffset = 13;
00075 #elif EXTI_LINE14_EN == ENABLE
00076     L_u8ShiftedOffset = 14;
00077 #elif EXTI_LINE15_EN == ENABLE
00078     L_u8ShiftedOffset = 15;
```

```
00079 #endif
00080
00081     // Get the enabled line ID.
00082     // GS_u8EXTI_EnabledLine_ID = L_u8ShiftedOffset;
00083
00084     // Clear all flags.
00085     MEXTI->PR = 0xffffffff;
00086
00087     // Enable EXTI on a line.
00088     MEXTI->IMR = 0;
00089
00090     MEXTI->IMR |= (ENABLE << L_u8ShiftedOffset);
00091
00092 // Set EXTI Triggered status on a line.
00093 #if EXTI_LINE0_TRIGGER == EXTI_RisingEdge
00094
00095     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00096
00097 #elif EXTI_LINE0_TRIGGER == EXTI_FallingEdge
00098     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00099
00100 #elif EXTI_LINE0_TRIGGER == EXTI_OnChange
00101     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00102     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00103
00104 #elif EXTI_LINE1_TRIGGER == EXTI_RisingEdge
00105     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00106
00107 #elif EXTI_LINE1_TRIGGER == EXTI_FallingEdge
00108     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00109
00110 #elif EXTI_LINE1_TRIGGER == EXTI_OnChange
00111     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00112     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00113
00114 #elif EXTI_LINE2_TRIGGER == EXTI_RisingEdge
00115     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00116
00117 #elif EXTI_LINE2_TRIGGER == EXTI_FallingEdge
00118     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00119
00120 #elif EXTI_LINE2_TRIGGER == EXTI_OnChange
00121     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00122     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00123
00124 #elif EXTI_LINE3_TRIGGER == EXTI_RisingEdge
00125     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00126
00127 #elif EXTI_LINE3_TRIGGER == EXTI_FallingEdge
00128     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00129
00130 #elif EXTI_LINE3_TRIGGER == EXTI_OnChange
00131     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00132
00133 #elif EXTI_LINE4_TRIGGER == EXTI_RisingEdge
00134     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00135
00136 #elif EXTI_LINE4_TRIGGER == EXTI_FallingEdge
00137     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00138
00139 #elif EXTI_LINE4_TRIGGER == EXTI_OnChange
00140     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00141
00142 #elif EXTI_LINE5_TRIGGER == EXTI_RisingEdge
00143     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00144
00145 #elif EXTI_LINE5_TRIGGER == EXTI_FallingEdge
00146     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00147
00148 #elif EXTI_LINE5_TRIGGER == EXTI_OnChange
00149     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00150
00151 #elif EXTI_LINE5_TRIGGER == EXTI_RisingEdge
00152     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00153
00154 #elif EXTI_LINE5_TRIGGER == EXTI_FallingEdge
00155     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00156
00157 #elif EXTI_LINE5_TRIGGER == EXTI_OnChange
00158     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00159
00160 #elif EXTI_LINE6_TRIGGER == EXTI_RisingEdge
00161     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00162
00163 #elif EXTI_LINE6_TRIGGER == EXTI_FallingEdge
00164     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00165
00166 #elif EXTI_LINE6_TRIGGER == EXTI_OnChange
00167     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00168
00169 #elif EXTI_LINE7_TRIGGER == EXTI_RisingEdge
00170     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
```

```

00173
00174 #elif EXTI_LINE7_TRIGGER == EXTI_FallingEdge
00175     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00176
00177 #elif EXTI_LINE7_TRIGGER == EXTI_OnChange
00178     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00179     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00180
00182 #elif EXTI_LINE8_TRIGGER == EXTI_RisingEdge
00183     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00184
00185 #elif EXTI_LINE8_TRIGGER == EXTI_FallingEdge
00186     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00187
00188 #elif EXTI_LINE8_TRIGGER == EXTI_OnChange
00189     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00190     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00191
00193 #elif EXTI_LINE9_TRIGGER == EXTI_RisingEdge
00194     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00195
00196 #elif EXTI_LINE9_TRIGGER == EXTI_FallingEdge
00197     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00198
00199 #elif EXTI_LINE9_TRIGGER == EXTI_OnChange
00200     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00201     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00202
00204 #elif EXTI_LINE10_TRIGGER == EXTI_RisingEdge
00205     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00206
00207 #elif EXTI_LINE10_TRIGGER == EXTI_FallingEdge
00208     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00209
00210 #elif EXTI_LINE10_TRIGGER == EXTI_OnChange
00211     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00212     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00213
00215 #elif EXTI_LINE11_TRIGGER == EXTI_RisingEdge
00216     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00217
00218 #elif EXTI_LINE11_TRIGGER == EXTI_FallingEdge
00219     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00220
00221 #elif EXTI_LINE11_TRIGGER == EXTI_OnChange
00222     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00223     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00224
00226 #elif EXTI_LINE12_TRIGGER == EXTI_RisingEdge
00227     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00228
00229 #elif EXTI_LINE12_TRIGGER == EXTI_FallingEdge
00230     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00231
00232 #elif EXTI_LINE12_TRIGGER == EXTI_OnChange
00233     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00234     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00235
00237 #elif EXTI_LINE13_TRIGGER == EXTI_RisingEdge
00238     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00239
00240 #elif EXTI_LINE13_TRIGGER == EXTI_FallingEdge
00241     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00242
00243 #elif EXTI_LINE13_TRIGGER == EXTI_OnChange
00244     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00245     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00246
00248 #elif EXTI_LINE14_TRIGGER == EXTI_RisingEdge
00249     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00250
00251 #elif EXTI_LINE14_TRIGGER == EXTI_FallingEdge
00252     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00253
00254 #elif EXTI_LINE14_TRIGGER == EXTI_OnChange
00255     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00256     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00257
00259 #elif EXTI_LINE15_TRIGGER == EXTI_RisingEdge
00260     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00261
00262 #elif EXTI_LINE15_TRIGGER == EXTI_FallingEdge
00263     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00264
00265 #elif EXTI_LINE15_TRIGGER == EXTI_OnChange
00266     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00267     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);

```

```

00268
00269 #endif
00270
00271 }
```

References [ENABLE](#), [INITIAL_ZERO](#), [MEXTI](#), [SET_BIT](#), and [VAR](#).

7.26.2.3 MEXTI_vInit_WithStruct()

```
void MEXTI_vInit_WithStruct (
    EXTI_ConfigType * A_xINTConfig )
```

Definition at line 276 of file [EXTI_program.c](#).

```

00277 {
00278
00279     MEXTI_vEnableLine( A_xINTConfig->LineNum, A_xINTConfig->TriggerStatus ) ;
00280
00281     MSYSCFG_vSetEXTIPort( A_xINTConfig->LineNum, A_xINTConfig->PortNum ) ;
00282
00283 }
```

References [EXTI_ConfigType::LineNum](#), [MEXTI_vEnableLine\(\)](#), [MSYSCFG_vSetEXTIPort\(\)](#), [EXTI_ConfigType::PortNum](#), and [EXTI_ConfigType::TriggerStatus](#).

7.26.2.4 MEXTI_vEnableLine()

```
void MEXTI_vEnableLine (
    u8_t A_u8LineID,
    u8_t A_u8TriggerStatus )
```

Definition at line 288 of file [EXTI_program.c](#).

```

00289 {
00290
00291     if (A_u8LineID < EXTI_MAX_EXTI_NUM)
00292     {
00293
00294         SET_BIT(MEXTI->IMR, A_u8LineID);
00295
00296         MEXTI_vSetTrigger(A_u8LineID, A_u8TriggerStatus);
00297
00298     }
00299
00300 }
```

References [EXTI_MAX_EXTI_NUM](#), [MEXTI](#), [MEXTI_vSetTrigger\(\)](#), and [SET_BIT](#).

Referenced by [MEXTI_vInit_WithStruct\(\)](#).

7.26.2.5 MEXTI_vDisableLine()

```
void MEXTI_vDisableLine (
    u8_t A_u8LineID )
```

Definition at line 305 of file [EXTI_program.c](#).

```

00306 {
00307
00308     if (A_u8LineID < EXTI_MAX_EXTI_NUM)
00309     {
00310         CLR_BIT(MEXTI->IMR, A_u8LineID);
00311     }
00312
00313 }
```

References [CLR_BIT](#), [EXTI_MAX_EXTI_NUM](#), and [MEXTI](#).

7.26.2.6 MEXTI_vSWITrigger()

```
void MEXTI_vSWITrigger (
    u8_t A_u8LineID )
```

Definition at line 318 of file [EXTI_program.c](#).

```
00319 {
00320
00321     if (A_u8LineID < EXTI_MAX_EXTI_NUM)
00322     {
00323         SET_BIT(MEXTI->SWIER, A_u8LineID);
00324     }
00325
00326 }
```

References [EXTI_MAX_EXTI_NUM](#), [MEXTI](#), and [SET_BIT](#).

7.26.2.7 MEXTI_vSetTrigger()

```
void MEXTI_vSetTrigger (
    u8_t A_u8LineID,
    u8_t A_u8TriggerStatus )
```

Definition at line 331 of file [EXTI_program.c](#).

```
00332 {
00333
00334     if (A_u8LineID < EXTI_MAX_EXTI_NUM)
00335     {
00336
00337         switch (A_u8TriggerStatus)
00338         {
00339             case EXTI_RisingEdge:
00340                 SET_BIT(MEXTI->RTSR, A_u8LineID);
00341
00342                 CLR_BIT(MEXTI->FTSR, A_u8LineID);
00343                 break;
00344
00345             case EXTI_FallingEdge:
00346                 SET_BIT(MEXTI->FTSR, A_u8LineID);
00347
00348                 CLR_BIT(MEXTI->RTSR, A_u8LineID);
00349                 break;
00350
00351             case EXTI_OnChange:
00352                 SET_BIT(MEXTI->RTSR, A_u8LineID);
00353
00354                 SET_BIT(MEXTI->FTSR, A_u8LineID);
00355                 break;
00356         }
00357     }
00358
00359 }
```

References [CLR_BIT](#), [EXTI_FallingEdge](#), [EXTI_MAX_EXTI_NUM](#), [EXTI_OnChange](#), [EXTI_RisingEdge](#), [MEXTI](#), and [SET_BIT](#).

Referenced by [MEXTI_vEnableLine\(\)](#).

7.26.2.8 MEXTI_vSetCallback()

```
void MEXTI_vSetCallback (
    u8_t A_u8LineID,
    void(*) (void) A_vFptr )
```

Definition at line 364 of file [EXTI_program.c](#).

```
00365 {
00366
00367     if (A_u8LineID < EXTI_MAX_EXTI_NUM)
00368     {
00369         GS_vEXTI_Callback[A_u8LineID] = A_vFptr;
00370     }
00371
00372 }
```

References [EXTI_MAX_EXTI_NUM](#).

7.26.2.9 EXTI0_IRQHandler()

```
void EXTI0_IRQHandler (
    void )
```

Definition at line 377 of file [EXTI_program.c](#).

```
00378 {
00379
00380     if (GS_vEXTI_Callback[EXTI_LINE0] != NULL)
00381     {
00382         GS_vEXTI_Callback[EXTI_LINE0] ();
00383     }
00384
00385     // Clear the flag.
00386     SET_BIT (MEXTI->PR, EXTI_LINE0);
00387
00388 }
```

References [EXTI_LINE0](#), [MEXTI](#), [NULL](#), and [SET_BIT](#).

7.26.2.10 EXTI1_IRQHandler()

```
void EXTI1_IRQHandler (
    void )
```

Definition at line 393 of file [EXTI_program.c](#).

```
00394 {
00395
00396     if (GS_vEXTI_Callback[EXTI_LINE1] != NULL)
00397     {
00398         GS_vEXTI_Callback[EXTI_LINE1] ();
00399     }
00400
00401     // Clear the flag.
00402     SET_BIT (MEXTI->PR, EXTI_LINE1);
00403
00404 }
```

References [EXTI_LINE1](#), [MEXTI](#), [NULL](#), and [SET_BIT](#).

7.26.2.11 EXTI2_IRQHandler()

```
void EXTI2_IRQHandler (
    void )
```

Definition at line 409 of file [EXTI_program.c](#).

```
00410 {
00411
00412     if (GS_vEXTI_Callback[EXTI_LINE2] != NULL)
00413     {
00414         GS_vEXTI_Callback[EXTI_LINE2]();
00415     }
00416
00417     // Clear the flag.
00418     SET_BIT(MEXTI->PR, EXTI_LINE2);
00419
00420 }
```

References [EXTI_LINE2](#), [MEXTI](#), [NULL](#), and [SET_BIT](#).

7.26.2.12 EXTI3_IRQHandler()

```
void EXTI3_IRQHandler (
    void )
```

Definition at line 425 of file [EXTI_program.c](#).

```
00426 {
00427
00428     if (GS_vEXTI_Callback[EXTI_LINE3] != NULL)
00429     {
00430         GS_vEXTI_Callback[EXTI_LINE3]();
00431     }
00432
00433     // Clear the flag.
00434     SET_BIT(MEXTI->PR, EXTI_LINE3);
00435
00436 }
```

References [EXTI_LINE3](#), [MEXTI](#), [NULL](#), and [SET_BIT](#).

7.26.2.13 EXTI4_IRQHandler()

```
void EXTI4_IRQHandler (
    void )
```

Definition at line 441 of file [EXTI_program.c](#).

```
00442 {
00443
00444     if (GS_vEXTI_Callback[EXTI_LINE4] != NULL)
00445     {
00446         GS_vEXTI_Callback[EXTI_LINE4]();
00447     }
00448
00449     // Clear the flag.
00450     SET_BIT(MEXTI->PR, EXTI_LINE4);
00451
00452 }
```

References [EXTI_LINE4](#), [MEXTI](#), [NULL](#), and [SET_BIT](#).

7.26.2.14 EXTI9_5_IRQHandler()

```
void EXTI9_5_IRQHandler (
    void )
```

Definition at line 457 of file [EXTI_program.c](#).

```
00458 {
00459
00460 }
```

7.26.2.15 EXTI15_10_IRQHandler()

```
void EXTI15_10_IRQHandler (
    void )
```

Definition at line 465 of file [EXTI_program.c](#).

```
00466 {
00467
00468 }
```

7.27 EXTI_program.c

[Go to the documentation of this file.](#)

```
00001
00009 /*****
00010 /*           Include headers                         */
00011 *****/
00012 #include "../../../LIB/LSTD_TYPES.h"
00013 #include "../../../LIB/LSTD_COMPILER.h"
00014 #include "../../../LIB/LSTD_VALUES.h"
00015 #include "../../../LIB/LSTD_BITMATH.h"
00016
00017 #include "EXTI_interface.h"
00018 #include "EXTI_private.h"
00019 #include "EXTI_config.h"
00020 #include "SYSCFG_private.h"
00021
00022
00023 static void( *GS_vEXTI_Callback[EXTI_IRQs] ) (void) = {NULL} ;
00024
00025
00026 /*****
00027 /*           Functions implementations                  */
00028 *****/
00029
00030 FUNC(void) MSYSCFG_vSetEXTIPort(VAR(u8_t) A_u8LineID, VAR(u8_t) A_u8PortID)
00031 {
00032     VAR(u8_t) L_u8Index = A_u8LineID / 4;
00033     VAR(u8_t) L_u8ShiftAmount = A_u8LineID % 4;
00034
00035     CLR_BITs(MSYSCFG->EXTICR[L_u8Index], 0b1111, L_u8ShiftAmount, 4);
00036     SET_BITs(MSYSCFG->EXTICR[L_u8Index], A_u8PortID, L_u8ShiftAmount, 4);
00037 }
00038
00039
00040
00041
00042 FUNC(void) MEXTI_vInit(void)
00043 {
00044
00045     VAR(u8_t) L_u8ShiftedOffset = INITIAL_ZERO;
00046
00047 #if EXTI_LINE0_EN == ENABLE
00048     L_u8ShiftedOffset = 0;
00049 #elif EXTI_LINE1_EN == ENABLE
00050     L_u8ShiftedOffset = 1;
00051 #elif EXTI_LINE2_EN == ENABLE
```

```

00052     L_u8ShiftedOffset = 2;
00053 #elif EXTI_LINE3_EN == ENABLE
00054     L_u8ShiftedOffset = 3;
00055 #elif EXTI_LINE4_EN == ENABLE
00056     L_u8ShiftedOffset = 4;
00057 #elif EXTI_LINE5_EN == ENABLE
00058     L_u8ShiftedOffset = 5;
00059 #elif EXTI_LINE6_EN == ENABLE
00060     L_u8ShiftedOffset = 6;
00061 #elif EXTI_LINE7_EN == ENABLE
00062     L_u8ShiftedOffset = 7;
00063 #elif EXTI_LINE8_EN == ENABLE
00064     L_u8ShiftedOffset = 8;
00065 #elif EXTI_LINE9_EN == ENABLE
00066     L_u8ShiftedOffset = 9;
00067 #elif EXTI_LINE10_EN == ENABLE
00068     L_u8ShiftedOffset = 10;
00069 #elif EXTI_LINE11_EN == ENABLE
00070     L_u8ShiftedOffset = 11;
00071 #elif EXTI_LINE12_EN == ENABLE
00072     L_u8ShiftedOffset = 12;
00073 #elif EXTI_LINE13_EN == ENABLE
00074     L_u8ShiftedOffset = 13;
00075 #elif EXTI_LINE14_EN == ENABLE
00076     L_u8ShiftedOffset = 14;
00077 #elif EXTI_LINE15_EN == ENABLE
00078     L_u8ShiftedOffset = 15;
00079 #endif
00080
00081     // Get the enabled line ID.
00082     // GS_u8EXTI_EnabledLine_ID = L_u8ShiftedOffset;
00083
00084     // Clear all flags.
00085     MEXTI->PR = 0xffffffff;
00086
00087     // Enable EXTI on a line.
00088     MEXTI->IMR = 0;
00089
00090     MEXTI->IMR |= (ENABLE « L_u8ShiftedOffset);
00091
00092 // Set EXTI Triggered status on a line.
00093 #if EXTI_LINE0_TRIGGER == EXTI_RisingEdge
00094
00095     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00096
00097 #elif EXTI_LINE0_TRIGGER == EXTI_FallingEdge
00098     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00099
00100 #elif EXTI_LINE0_TRIGGER == EXTI_OnChange
00101     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00102     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00103
00104 #elif EXTI_LINE1_TRIGGER == EXTI_RisingEdge
00105     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00106
00107 #elif EXTI_LINE1_TRIGGER == EXTI_FallingEdge
00108     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00109
00110 #elif EXTI_LINE1_TRIGGER == EXTI_OnChange
00111     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00112     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00113
00114 #elif EXTI_LINE2_TRIGGER == EXTI_RisingEdge
00115     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00116
00117 #elif EXTI_LINE2_TRIGGER == EXTI_FallingEdge
00118     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00119
00120 #elif EXTI_LINE2_TRIGGER == EXTI_OnChange
00121     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00122     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00123
00124 #elif EXTI_LINE3_TRIGGER == EXTI_RisingEdge
00125     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00126
00127 #elif EXTI_LINE3_TRIGGER == EXTI_FallingEdge
00128     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00129
00130 #elif EXTI_LINE3_TRIGGER == EXTI_OnChange
00131     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00132
00133 #elif EXTI_LINE4_TRIGGER == EXTI_RisingEdge
00134     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00135
00136 #elif EXTI_LINE4_TRIGGER == EXTI_FallingEdge
00137     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00138
00139 #endif
00140
00141 #endif
00142

```

```
00143
00144 #elif EXTI_LINE4_TRIGGER == EXTI_OnChange
00145     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00146     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00147
00148 #elif EXTI_LINE5_TRIGGER == EXTI_RisingEdge
00149     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00150
00151 #elif EXTI_LINE5_TRIGGER == EXTI_FallingEdge
00152     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00153
00154 #elif EXTI_LINE5_TRIGGER == EXTI_OnChange
00155     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00156     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00157
00158 #elif EXTI_LINE6_TRIGGER == EXTI_RisingEdge
00159     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00160
00161 #elif EXTI_LINE6_TRIGGER == EXTI_FallingEdge
00162     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00163
00164 #elif EXTI_LINE6_TRIGGER == EXTI_OnChange
00165     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00166     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00167
00168 #elif EXTI_LINE7_TRIGGER == EXTI_RisingEdge
00169     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00170
00171 #elif EXTI_LINE7_TRIGGER == EXTI_FallingEdge
00172     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00173
00174 #elif EXTI_LINE7_TRIGGER == EXTI_OnChange
00175     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00176
00177 #elif EXTI_LINE7_TRIGGER == EXTI_OnChange
00178     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00179     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00180
00181 #elif EXTI_LINE8_TRIGGER == EXTI_RisingEdge
00182     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00183
00184 #elif EXTI_LINE8_TRIGGER == EXTI_FallingEdge
00185     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00186
00187 #elif EXTI_LINE8_TRIGGER == EXTI_OnChange
00188     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00189     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00190
00191 #elif EXTI_LINE9_TRIGGER == EXTI_RisingEdge
00192     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00193
00194 #elif EXTI_LINE9_TRIGGER == EXTI_FallingEdge
00195     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00196
00197 #elif EXTI_LINE9_TRIGGER == EXTI_OnChange
00198     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00199
00200 #elif EXTI_LINE10_TRIGGER == EXTI_RisingEdge
00201     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00202
00203 #elif EXTI_LINE10_TRIGGER == EXTI_FallingEdge
00204     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00205
00206 #elif EXTI_LINE10_TRIGGER == EXTI_OnChange
00207     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00208
00209 #elif EXTI_LINE10_TRIGGER == EXTI_OnChange
00210     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00211     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00212
00213 #elif EXTI_LINE11_TRIGGER == EXTI_RisingEdge
00214     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00215
00216 #elif EXTI_LINE11_TRIGGER == EXTI_FallingEdge
00217     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00218
00219 #elif EXTI_LINE11_TRIGGER == EXTI_OnChange
00220     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00221
00222 #elif EXTI_LINE11_TRIGGER == EXTI_OnChange
00223     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00224
00225 #elif EXTI_LINE12_TRIGGER == EXTI_RisingEdge
00226     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00227
00228 #elif EXTI_LINE12_TRIGGER == EXTI_FallingEdge
00229     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00230
00231 #elif EXTI_LINE12_TRIGGER == EXTI_OnChange
00232     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00233     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00234
00235 #elif EXTI_LINE13_TRIGGER == EXTI_RisingEdge
00236     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
```

```

00239
00240 #elif EXTI_LINE13_TRIGGER == EXTI_FallingEdge
00241     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00242
00243 #elif EXTI_LINE13_TRIGGER == EXTI_OnChange
00244     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00245     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00246
00248 #elif EXTI_LINE14_TRIGGER == EXTI_RisingEdge
00249     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00250
00251 #elif EXTI_LINE14_TRIGGER == EXTI_FallingEdge
00252     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00253
00254 #elif EXTI_LINE14_TRIGGER == EXTI_OnChange
00255     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00256     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00257
00259 #elif EXTI_LINE15_TRIGGER == EXTI_RisingEdge
00260     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00261
00262 #elif EXTI_LINE15_TRIGGER == EXTI_FallingEdge
00263     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00264
00265 #elif EXTI_LINE15_TRIGGER == EXTI_OnChange
00266     SET_BIT(MEXTI->RTSR, L_u8ShiftedOffset);
00267     SET_BIT(MEXTI->FTSR, L_u8ShiftedOffset);
00268
00269 #endif
00270 }
00272
00273
00274 //*****
00274
00275
00276 FUNC(void) MEXTI_vInit_WithStruct(P2VAR(EXTI_ConfigType) A_xINTConfig)
00277 {
00278
00279     MEXTI_vEnableLine( A_xINTConfig->LineNum, A_xINTConfig->TriggerStatus ) ;
00280
00281     MSYSCFG_vSetEXTIPort( A_xINTConfig->LineNum, A_xINTConfig->PortNum ) ;
00282
00283 }
00284
00285
00286 //*****
00287
00288 FUNC(void) MEXTI_vEnableLine(VAR(u8_t) A_u8LineID, VAR(u8_t) A_u8TriggerStatus)
00289 {
00290
00291     if (A_u8LineID < EXTI_MAX_EXTI_NUM)
00292     {
00293
00294         SET_BIT(MEXTI->IMR, A_u8LineID);
00295
00296         MEXTI_vSetTrigger(A_u8LineID, A_u8TriggerStatus);
00297
00298     }
00299
00300 }
00301
00302
00303 //*****
00304
00305 FUNC(void) MEXTI_vDisableLine(VAR(u8_t) A_u8LineID)
00306 {
00307
00308     if (A_u8LineID < EXTI_MAX_EXTI_NUM)
00309     {
00310         CLR_BIT(MEXTI->IMR, A_u8LineID);
00311     }
00312
00313 }
00314
00315
00316 //*****
00317
00318 FUNC(void) MEXTI_vSWITrigger(VAR(u8_t) A_u8LineID)
00319 {

```

```

00320
00321     if (A_u8LineID < EXTI_MAX_EXTI_NUM)
00322     {
00323         SET_BIT(MEXTI->SWIER, A_u8LineID);
00324     }
00325
00326 }
00327
00328 /*****
00329 ****/
00330
00331 FUNC(void) MEXTI_vSetTrigger(VAR(u8_t) A_u8LineID, VAR(u8_t) A_u8TriggerStatus)
00332 {
00333
00334     if (A_u8LineID < EXTI_MAX_EXTI_NUM)
00335     {
00336
00337         switch (A_u8TriggerStatus)
00338         {
00339             case EXTI_RisingEdge:
00340                 SET_BIT(MEXTI->RTSR, A_u8LineID);
00341
00342                 CLR_BIT(MEXTI->FTSR, A_u8LineID);
00343                 break;
00344
00345             case EXTI_FallingEdge:
00346                 SET_BIT(MEXTI->FTSR, A_u8LineID);
00347
00348                 CLR_BIT(MEXTI->RTSR, A_u8LineID);
00349                 break;
00350
00351             case EXTI_OnChange:
00352                 SET_BIT(MEXTI->RTSR, A_u8LineID);
00353
00354                 SET_BIT(MEXTI->FTSR, A_u8LineID);
00355                 break;
00356         }
00357     }
00358
00359 }
00360
00361 /*****
00362 ****/
00363
00364 FUNC(void) MEXTI_vSetCallback(VAR(u8_t) A_u8LineID, P2FUNC(VAR(void), A_vFptr) (void))
00365 {
00366
00367     if (A_u8LineID < EXTI_MAX_EXTI_NUM)
00368     {
00369         GS_vEXTI_Callback[A_u8LineID] = A_vFptr;
00370     }
00371
00372 }
00373
00374 /*****
00375 ****/
00376
00377 FUNC(void) EXTI0_IRQHandler(void)
00378 {
00379
00380     if (GS_vEXTI_Callback[EXTI_LINE0] != NULL)
00381     {
00382         GS_vEXTI_Callback[EXTI_LINE0]();
00383     }
00384
00385     // Clear the flag.
00386     SET_BIT(MEXTI->PR, EXTI_LINE0);
00387
00388 }
00389
00390 /*****
00391 ****/
00392
00393 FUNC(void) EXTI1_IRQHandler(void)
00394 {
00395
00396     if (GS_vEXTI_Callback[EXTI_LINE1] != NULL)
00397     {
00398         GS_vEXTI_Callback[EXTI_LINE1]();
00399

```

```

00399     }
00400
00401     // Clear the flag.
00402     SET_BIT(MEXTI->PR, EXTI_LINE1);
00403
00404 }
00405
00406
00407 /*****
00408 ****/
00409 FUNC(void) EXTI2_IRQHandler(void)
00410 {
00411
00412     if (GS_vEXTI_Callback[EXTI_LINE2] != NULL)
00413     {
00414         GS_vEXTI_Callback[EXTI_LINE2]();
00415     }
00416
00417     // Clear the flag.
00418     SET_BIT(MEXTI->PR, EXTI_LINE2);
00419
00420 }
00421
00422
00423 /*****
00424 ****/
00425 FUNC(void) EXTI3_IRQHandler(void)
00426 {
00427
00428     if (GS_vEXTI_Callback[EXTI_LINE3] != NULL)
00429     {
00430         GS_vEXTI_Callback[EXTI_LINE3]();
00431     }
00432
00433     // Clear the flag.
00434     SET_BIT(MEXTI->PR, EXTI_LINE3);
00435
00436 }
00437
00438
00439 /*****
00440 ****/
00441 FUNC(void) EXTI4_IRQHandler(void)
00442 {
00443
00444     if (GS_vEXTI_Callback[EXTI_LINE4] != NULL)
00445     {
00446         GS_vEXTI_Callback[EXTI_LINE4]();
00447     }
00448
00449     // Clear the flag.
00450     SET_BIT(MEXTI->PR, EXTI_LINE4);
00451
00452 }
00453
00454
00455 /*****
00456 ****/
00457 FUNC(void) EXTI9_5_IRQHandler(void)
00458 {
00459
00460 }
00461
00462
00463 /*****
00464 ****/
00465 FUNC(void) EXTI15_10_IRQHandler(void)
00466 {
00467
00468 }
00469
00470
00471 /*****
00472 ****/
00473

```

```
00474  
00475  
00476  
00477  
00478  
00479  
00480  
00481  
00482  
00483  
00484  
00485  
00486  
00487  
00488  
00489  
00490  
00491
```

7.28 COTS/MCAL/EXTI/SYSCFG_private.h File Reference

This file contains the private information regarding the SYSCFG.

Data Structures

- struct [MSYSCFG_MemMap_t](#)

System configuration structure to initialize the SYSCFG module with.

Macros

- #define [SYSCFG_BASE_ADDR](#) (0x40013800)
- #define [MSYSCFG](#) ((volatile [MSYSCFG_MemMap_t](#) *)([SYSCFG_BASE_ADDR](#)))

7.28.1 Detailed Description

This file contains the private information regarding the SYSCFG.

Author

Ali El Bana

This is needed to ensure that the clocked is enabled for the EXTI module

Version

1.0

Date

09/01/2022

Definition in file [SYSCFG_private.h](#).

7.29 SYSCFG_private.h

[Go to the documentation of this file.](#)

```
00001 /* Header file guard */
00010 #ifndef MCAL_EXTI_SYSCFG_PRV_H_
00011 #define MCAL_EXTI_SYSCFG_PRV_H_
00012
00013
00014 typedef struct
00015 {
00016     u32_t MEMRMP;
00017     u32_t PMC;
00018     u32_t EXTICR[4];
00019     u32_t CMPCR;
00020 } MSYSCFG_MemMap_t;
00021
00022 #define SYSCFG_BASE_ADDR (0x40013800)
00023 #define MSYSCFG ((volatile MSYSCFG_MemMap_t *) (SYSCFG_BASE_ADDR))
00024
00025 #endif /* MCAL_EXTI_SYSCFG_PRV_H_ */
```

7.30 COTS/MCAL/GPIO/GPIO_config.h File Reference

This file contains the GPIO configurations.

Macros

- `#define GPIOA_PIN_POS (0b1110000000000000)`
lock PA13, PA14 and PA15
- `#define GPIOB_PIN_POS (0b0000000000001110)`
lock PB2, PB3 and PB4
- `#define LCKK_BIT_POS (16U)`
Position of LCKK bit.

7.30.1 Detailed Description

This file contains the GPIO configurations.

Author

Ali El Bana & Mo Alaa

Version

2.0

Date

08/22/2022

Definition in file [GPIO_config.h](#).

7.30.2 Macro Definition Documentation

7.30.2.1 GPIOA_PIN_POS

```
#define GPIOA_PIN_POS (0b1110000000000000)
```

lock PA13, PA14 and PA15

Definition at line 22 of file [GPIO_config.h](#).

7.30.2.2 GPIOB_PIN_POS

```
#define GPIOB_PIN_POS (0b00000000000011100)
```

lock PB2, PB3 and PB4

Definition at line 29 of file [GPIO_config.h](#).

7.30.2.3 LCKK_BIT_POS

```
#define LCKK_BIT_POS (16U)
```

Position of LCKK bit.

Definition at line 36 of file [GPIO_config.h](#).

7.31 GPIO_config.h

[Go to the documentation of this file.](#)

```
00001
00009 #ifndef _GPIO_config_H
00010 #define _GPIO_config_H
00011
00012
00013 /***** // GPIOx configurations *****/
00014           // GPIOx configurations //
00015 /***** *****/
00016
00022 #define GPIOA_PIN_POS (0b1110000000000000) //I want to lock PA13,14,15.
00023
00029 #define GPIOB_PIN_POS (0b00000000000011100) //I want to lock PB2,3,4.
00030
00036 #define LCKK_BIT_POS (16U)                  //Position of LCKK bit.
00037
00038
00039 #endif // _GPIO_config_H
```

7.32 COTS/MCAL/GPIO/GPIO_interface.h File Reference

This file contains the interfacing information for the GPIO module.

Data Structures

- struct `MGPIOx_ConfigType`
MDIO Configuration structure for a specific PIN initialization.

Macros

- `#define GPIOx_MODE_INPUT` (0b00)
Control input mode.
- `#define GPIOx_MODE_OUTPUT` (0b01)
Control output mode.
- `#define GPIOx_MODE_AF` (0b10)
Control alternate function mode.
- `#define GPIOx_MODE_ANALOG` (0b11)
Control analog mode.
- `#define GPIO_PORTA` (0)
GPIO Port A.
- `#define GPIO_PORTB` (1)
GPIO Port B.
- `#define GPIO_PORTC` (2)
GPIO Port C.
- `#define GPIOx_OPENDRAIN` (1)
GPIO open-drain.
- `#define GPIOx_PUSHPULL` (2)
GPIO push-pull.
- `#define GPIOx_LowSpeed` (0b00)
GPIO low speed.
- `#define GPIOx_MediumSpeed` (0b01)
GPIO medium speed.
- `#define GPIOx_HighSpeed` (0b10)
GPIO high speed.
- `#define GPIOx_VeryHighSpeed` (0b11)
GPIO very high speed.
- `#define GPIOx_NoPull` (0b00)
GPIO No PULL.
- `#define GPIOx_PullUp` (0b01)
GPIO Pull UP.
- `#define GPIOx_PullDown` (0b10)
GPIO Pull Down.
- `#define GPIOx_HIGH` (1)
GPIO output high.
- `#define GPIOx_LOW` (2)
GPIO output low.
- `#define GPIOx_PIN0` (0)
GPIO PIN 0.

- #define **GPIOx_PIN1** (1)
 GPIO PIN 1.
- #define **GPIOx_PIN2** (2)
 GPIO PIN 2.
- #define **GPIOx_PIN3** (3)
 GPIO PIN 3.
- #define **GPIOx_PIN4** (4)
 GPIO PIN 4.
- #define **GPIOx_PIN5** (5)
 GPIO PIN 5.
- #define **GPIOx_PIN6** (6)
 GPIO PIN 6.
- #define **GPIOx_PIN7** (7)
 GPIO PIN 7.
- #define **GPIOx_PIN8** (8)
 GPIO PIN 8.
- #define **GPIOx_PIN9** (9)
 brief GPIO PIN 9
- #define **GPIOx_PIN10** (10)
 GPIO PIN 10.
- #define **GPIOx_PIN11** (11)
 GPIO PIN 11.
- #define **GPIOx_PIN12** (12)
 GPIO PIN 12.
- #define **GPIOx_PIN13** (13)
 GPIO PIN 13.
- #define **GPIOx_PIN14** (14)
 GPIO PIN 14.
- #define **GPIOx_PIN15** (15)
 GPIO PIN 15.
- #define **GPIOx_AF0** (0)
 GPIO Alternate function 0.
- #define **GPIOx_AF1** (1)
 GPIO Alternate function 1.
- #define **GPIOx_AF2** (2)
 GPIO Alternate function 2.
- #define **GPIOx_AF3** (3)
 GPIO Alternate function 3.
- #define **GPIOx_AF4** (4)
 GPIO Alternate function 4.
- #define **GPIOx_AF5** (5)
 GPIO Alternate function 5.
- #define **GPIOx_AF6** (6)
 GPIO Alternate function 6.
- #define **GPIOx_AF7** (7)
 GPIO Alternate function 7.
- #define **GPIOx_AF8** (8)
 GPIO Alternate function 8.
- #define **GPIOx_AF9** (9)
 GPIO Alternate function 9.
- #define **GPIOx_AF10** (10)

- #define GPIOx_AF11 (11)

GPIO Alternate function 10.
- #define GPIOx_AF12 (12)

GPIO Alternate function 11.
- #define GPIOx_AF13 (13)

GPIO Alternate function 12.
- #define GPIOx_AF14 (14)

GPIO Alternate function 13.
- #define GPIOx_AF15 (15)

GPIO Alternate function 14.
- #define GPIOx_AF16 (16)

GPIO Alternate function 15.

Functions

- void MGPIox_vLockedPins (void)

Locks the prohibited GPIO PINs.
- void MGPIox_vSetPinMode (VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID, VAR(u8_t) A_u8Mode)

Sets a certain pin's mode on a specific port.
- void MGPIox_vSetPinOutputType (VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID, VAR(u8_t) A_u8OutputType)

Sets a certain pin's output type on a specific port.
- void MGPIox_vSetPinOutputSpeed (VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID, VAR(u8_t) A_u8OutputSpeed)

Sets a certain pin's output speed on a specific port.
- void MGPIox_vSetPinInputPullType (VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID, VAR(u8_t) A_u8InputPullType)

Sets a certain pin's input pull type on a specific port.
- u8_t MGPIox_u8GetPinValue (VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID)

Gets the value currently on a certain pin.
- void MGPIox_vSetPinValue (VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID, VAR(u8_t) A_u8PinValue)

Sets a certain pin's output value on a specific port.
- void MGPIox_vSetResetAtomic (VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID, VAR(u8_t) A_u8SetResetPinValue)

Resets a certain pin's output value on a specific port.
- void MGPIox_vSetAlternateFunctionON (VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID, VAR(u8_t) A_u8AFID)

Applies an alternative function on a certain pin.
- void MGPIox_vSetPortConfigLock (VAR(u8_t) A_u8PortID)

Updates a port's configuration lock.
- void MGPIox_vInit (P2VAR(MGPIox_ConfigType) A_xPinConfig)

Initialize the GPIO with a certain configuration.
- void MGPIox_vTogglePinValue (VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID)

Toggles a certain pin's value on a certain port.

7.32.1 Detailed Description

This file contains the interfacing information for the GPIO module.

Author

Ali El Bana & Mo Alaa

Version

2.0

Date

11/9/2022

Definition in file [GPIO_interface.h](#).

7.32.2 Function Documentation

7.32.2.1 MGPIOX_vLockedPins()

```
void MGPIOX_vLockedPins (
    void )
```

Locks the prohibited GPIO PINs.

Definition at line 24 of file [GPIO_program.c](#).

```
00025 {
00026
00027     VAR(volatile u32_t) L_u32LockGPIOA = INITIAL_ZERO ;
00028
00029     /* Lock key write sequence */
00030
00031     /* WR LCKR[16] = '1' + LCKR[13,14,15] = *1* */
00032     L_u32LockGPIOA = ( (1UL << LCKK_BIT_POS) | (GPIOA_PIN_POS) ) ;
00033
00034     GPIOA->LCKRx = L_u32LockGPIOA ;
00035
00036     /* WR LCKR[16] = '0' + LCKR[13,14,15] should not change*/
00037     GPIOA->LCKRx = (GPIOA_PIN_POS) ;
00038
00039     /* WR LCKR[16] = '1'+ LCKR[13,14,15] should not change*/
00040     GPIOA->LCKRx = L_u32LockGPIOA ;
00041
00042     /* RD LCKR */
00043     L_u32LockGPIOA = GPIOA->LCKRx ;
00044
00045
00046     VAR(volatile u32_t) L_u32LockGPIOB = INITIAL_ZERO ;
00047
00048     /* Lock key write sequence */
00049
00050     /* WR LCKR[16] = '1' + LCKR[2,3,4] = *1* */
00051     L_u32LockGPIOB = ( (1UL << LCKK_BIT_POS) | (GPIOB_PIN_POS) ) ;
00052
00053     GPIOB->LCKRx = L_u32LockGPIOB ;
00054
00055     /* WR LCKR[16] = '0' + LCKR[2,3,4] should not change*/
00056     GPIOB->LCKRx = (GPIOB_PIN_POS) ;
00057
00058     /* WR LCKR[16] = '1' + LCKR[2,3,4] should not change*/
00059     GPIOB->LCKRx = L_u32LockGPIOB ;
00060
00061     /* RD LCKR */
00062     L_u32LockGPIOB = GPIOB->LCKRx ;
00063
00064
00065 }
```

References [GPIOA](#), [GPIOA_PIN_POS](#), [GPIOB](#), [GPIOB_PIN_POS](#), [INITIAL_ZERO](#), [LCKK_BIT_POS](#), and [VAR](#).

7.32.2.2 MGPIox_vSetPinMode()

```
void MGPIox_vSetPinMode (
    VAR(u8_t) A_u8PortID,
    VAR(u8_t) A_u8PinID,
    VAR(u8_t) A_u8Mode )
```

Sets a certain pin's mode on a specific port.

Parameters

in	A_u8PortID	The port that the pin belongs to
in	A_u8PinID	The pin to update its mode
in	A_u8Mode	The mode to apply the pin

7.32.2.3 MGPIox_vSetPinOutputType()

```
void MGPIox_vSetPinOutputType (
    VAR(u8_t) A_u8PortID,
    VAR(u8_t) A_u8PinID,
    VAR(u8_t) A_u8OutputType )
```

Sets a certain pin's output type on a specific port.

Parameters

in	A_u8PortID	The port that the pin belongs to
in	A_u8PinID	The pin to update its mode
in	A_u8OutputType	The output type to apply on the pin

7.32.2.4 MGPIox_vSetPinOutputSpeed()

```
void MGPIox_vSetPinOutputSpeed (
    VAR(u8_t) A_u8PortID,
    VAR(u8_t) A_u8PinID,
    VAR(u8_t) A_u8OutputSpeed )
```

Sets a certain pin's output speed on a specific port.

Parameters

in	A_u8PortID	The port that the pin belongs to
in	A_u8PinID	The pin to update its mode
in	A_u8OutputSpeed	The output speed to apply on the pin

7.32.2.5 MGPIOX_vSetPinInputPullType()

```
void MGPIOX_vSetPinInputPullType (
    VAR(u8_t) A_u8PortID,
    VAR(u8_t) A_u8PinID,
    VAR(u8_t) A_u8InputPullType )
```

Sets a certain pin's input pull type on a specific port.

Parameters

in	A_u8PortID	The port that the pin belongs to
in	A_u8PinID	The pin to update its mode
in	A_u8InputPullType	The input pull type to apply on the pin

7.32.2.6 MGPIOX_u8GetPinValue()

```
u8_t MGPIOX_u8GetPinValue (
    VAR(u8_t) A_u8PortID,
    VAR(u8_t) A_u8PinID )
```

Gets the value currently on a certain pin.

Parameters

in	A_u8PortID	The port that the pin belongs to
in	A_u8PinID	The pin to update its mode

Returns

The current value on the pin

7.32.2.7 MGPIOX_vSetPinValue()

```
void MGPIOX_vSetPinValue (
    VAR(u8_t) A_u8PortID,
    VAR(u8_t) A_u8PinID,
    VAR(u8_t) A_u8PinValue )
```

Sets a certain pin's output value on a specific port.

Parameters

in	<i>A_u8PortID</i>	The port that the pin belongs to
in	<i>A_u8PinID</i>	The pin to update its mode
in	<i>A_u8PinValue</i>	The set value to set on the pin

Referenced by [HDCM_vMoveBackward\(\)](#), [HDCM_vMoveForward\(\)](#), and [HDCM_vStopMotor\(\)](#).

7.32.2.8 MGPIOx_vSetResetAtomic()

```
void MGPIOx_vSetResetAtomic (
    VAR(u8_t) A_u8PortID,
    VAR(u8_t) A_u8PinID,
    VAR(u8_t) A_u8SetResetPinValue )
```

Resets a certain pin's output value on a specific port.

Parameters

in	<i>A_u8PortID</i>	The port that the pin belongs to
in	<i>A_u8PinID</i>	The pin to update its mode
in	<i>A_u8SetResetPinValue</i>	The reset value to set on the pin

7.32.2.9 MGPIOx_vSetAlternateFunctionON()

```
void MGPIOx_vSetAlternateFunctionON (
    VAR(u8_t) A_u8PortID,
    VAR(u8_t) A_u8PinID,
    VAR(u8_t) A_u8AFID )
```

Applies an alternative function on a certain pin.

Parameters

in	<i>A_u8PortID</i>	The port that the pin belongs to
in	<i>A_u8PinID</i>	The pin to update its mode
in	<i>A_u8AFID</i>	The alternative function to apply on the pin

7.32.2.10 MGPIOx_vSetPortConfigLock()

```
void MGPIOx_vSetPortConfigLock (
    VAR(u8_t) A_u8PortID )
```

Updates a port's configuration lock.

Parameters

in	<i>A_u8PortID</i>	The port to update the pin's mode
----	-------------------	-----------------------------------

7.32.2.11 MGPIox_vInit()

```
void MGPIox_vInit (
    P2VAR(MGPIox_ConfigType) A_xPinConfig )
```

Initialize the GPIO with a certain configuration.

Parameters

in	<i>A_xPinConfig</i>	The initialization configuration for the GPIO
----	---------------------	---

Referenced by [HDCM_vInitMotor\(\)](#).

7.32.2.12 MGPIox_vTogglePinValue()

```
void MGPIox_vTogglePinValue (
    VAR(u8_t) A_u8PortID,
    VAR(u8_t) A_u8PinID )
```

Toggles a certain's pin's value on a certain port.

Parameters

in	<i>A_u8PortID</i>	The port that the pin belongs to
in	<i>A_u8PinID</i>	The pin to toggle its value

7.33 GPIO_interface.h

[Go to the documentation of this file.](#)

```
00001
00009 /* Header file guard */
00010 #ifndef _GPIO_interface_H
00011 #define _GPIO_interface_H
00012
00018 typedef struct
00019 {
00023     u8_t Port;
00027     u8_t Pin;
00031     u8_t Mode;
00035     u8_t OutputType;
00039     u8_t OutputSpeed;
00043     u8_t InputType;
00047     u8_t AF_Type;
00048
```

```

00049 } MGPIox_ConfigType;
00050
00051 /****** Functions prototypes *****/
00052
00053 /***** */
00054
00058 void MGPIox_vLockedPins(void);
00059
00066 void MGPIox_vSetPinMode(VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID, VAR(u8_t) A_u8Mode);
00067
00074 void MGPIox_vSetPinOutputType(VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID, VAR(u8_t) A_u8OutputType);
00075
00082 void MGPIox_vSetPinOutputSpeed(VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID, VAR(u8_t) A_u8OutputSpeed);
00083
00090 void MGPIox_vSetPinInputPullType(VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID, VAR(u8_t)
    A_u8InputPullType);
00091
00098 u8_t MGPIox_u8GetPinValue(VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID);
00099
00106 void MGPIox_vSetPinValue(VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID, VAR(u8_t) A_u8PinValue);
00107
00114 void MGPIox_vSetResetAtomic(VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID, VAR(u8_t)
    A_u8SetResetPinValue);
00115
00122 void MGPIox_vSetAlternateFunctionON(VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID, VAR(u8_t) A_u8AFID);
00123
00128 void MGPIox_vSetPortConfigLock(VAR(u8_t) A_u8PortID);
00129
00134 void MGPIox_vInit(P2VAR(MGPIox_ConfigType) A_xPinConfig);
00135
00141 void MGPIox_vTogglePinValue(VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID);
00142
00143 /***** */
00144 /*           Interfacing macros           */
00145 /***** */
00146
00158 #define GPIOx_MODE_INPUT (0b00)
00159
00165 #define GPIOx_MODE_OUTPUT (0b01)
00166
00172 #define GPIOx_MODE_AF (0b10)
00173
00179 #define GPIOx_MODE_ANALOG (0b11)
00180
00194 #define GPIO_PORTA (0)
00195
00201 #define GPIO_PORTB (1)
00202
00208 #define GPIO_PORTC (2)
00209
00223 #define GPIOx_OPENDRAIN (1)
00224
00230 #define GPIOx_PUSH_PULL (2)
00231
00245 #define GPIOx_LowSpeed (0b00)
00246
00252 #define GPIOx_MediumSpeed (0b01)
00253
00259 #define GPIOx_HighSpeed (0b10)
00260
00266 #define GPIOx_VeryHighSpeed (0b11)
00267
00281 #define GPIOx_NoPull (0b00)
00282
00288 #define GPIOx_PullUp (0b01)
00289
00295 #define GPIOx_PullDown (0b10)
00296
00310 #define GPIOx_HIGH (1)
00311
00317 #define GPIOx_LOW (2)
00318
00332 #define GPIOx_PIN0 (0)
00333
00339 #define GPIOx_PIN1 (1)
00340
00346 #define GPIOx_PIN2 (2)
00347
00353 #define GPIOx_PIN3 (3)
00354
00360 #define GPIOx_PIN4 (4)
00361
00367 #define GPIOx_PIN5 (5)
00368
00374 #define GPIOx_PIN6 (6)
00375
00381 #define GPIOx_PIN7 (7)

```

```

00382
00388 #define GPIOx_PIN8 (8)
00389
00395 #define GPIOx_PIN9 (9)
00396
00402 #define GPIOx_PIN10 (10)
00403
00409 #define GPIOx_PIN11 (11)
00410
00416 #define GPIOx_PIN12 (12)
00417
00423 #define GPIOx_PIN13 (13)
00424
00430 #define GPIOx_PIN14 (14)
00431
00437 #define GPIOx_PIN15 (15)
00438
00452 #define GPIOx_AF0 (0)
00453
00459 #define GPIOx_AF1 (1)
00460
00466 #define GPIOx_AF2 (2)
00467
00473 #define GPIOx_AF3 (3)
00474
00480 #define GPIOx_AF4 (4)
00481
00487 #define GPIOx_AF5 (5)
00488
00494 #define GPIOx_AF6 (6)
00495
00501 #define GPIOx_AF7 (7)
00502
00508 #define GPIOx_AF8 (8)
00509
00515 #define GPIOx_AF9 (9)
00516
00522 #define GPIOx_AF10 (10)
00523
00529 #define GPIOx_AF11 (11)
00530
00536 #define GPIOx_AF12 (12)
00537
00543 #define GPIOx_AF13 (13)
00544
00550 #define GPIOx_AF14 (14)
00551
00557 #define GPIOx_AF15 (15)
00558
00561 #endif // _GPIO_interface_H

```

7.34 COTS/MCAL/GPIO/GPIO_private.h File Reference

This file contains the registers information and addresses for the GPIO module.

Data Structures

- struct [GPIOx_MemoryMapType](#)
GPIO declaration structure for its registers.

Macros

- #define [GPIOA_BASE_ADDRESS](#) (0x40020000)
- #define [GPIOB_BASE_ADDRESS](#) (0x40020400)
- #define [GPIOC_BASE_ADDRESS](#) (0x40020800)
- #define [GPIOA](#) ((volatile P2VAR(GPIOx_MemoryMapType))(GPIOA_BASE_ADDRESS))
- #define [GPIOB](#) ((volatile P2VAR(GPIOx_MemoryMapType))(GPIOB_BASE_ADDRESS))
- #define [GPIOC](#) ((volatile P2VAR(GPIOx_MemoryMapType))(GPIOC_BASE_ADDRESS))

7.34.1 Detailed Description

This file contains the registers information and addresses for the GPIO module.

Author

Ali El Bana & Mo Alaa

Version

2.0

Date

08/22/2022

Definition in file [GPIO_private.h](#).

7.35 GPIO_private.h

[Go to the documentation of this file.](#)

```
00001 /* Header file guard */
00009 #ifndef _GPIO_private_H
00010 #define _GPIO_private_H
00012
00018 typedef struct
00019 {
00023     u32_t MODERx;
00024
00028     u32_t OTYPERx;
00029
00033     u32_t OSPEEDRx;
00034
00038     u32_t PUPDRx;
00039
00043     u32_t IDRx;
00044
00048     u32_t ODRx;
00049
00053     u32_t BSRRx;
00054
00058     u32_t LCKRx;
00059
00063     u32_t AFRLx;
00064
00068     u32_t AFRHx;
00069
00070 } GPIOx_MemoryMapType;
00071
00083 #define GPIOA_BASE_ADDRESS (0x40020000)
00084
00090 #define GPIOB_BASE_ADDRESS (0x40020400)
00091
00097 #define GPIOC_BASE_ADDRESS (0x40020800)
00098
00112 #define GPIOA ((volatile P2VAR(GPIOx_MemoryMapType)) (GPIOA_BASE_ADDRESS))
00113
00119 #define GPIOB ((volatile P2VAR(GPIOx_MemoryMapType)) (GPIOB_BASE_ADDRESS))
00120
00126 #define GPIOC ((volatile P2VAR(GPIOx_MemoryMapType)) (GPIOC_BASE_ADDRESS))
00127
00130#endif // _GPIO_private_H
```

7.36 COTS/MCAL/GPIO(GPIO_program.c) File Reference

This file contains the source code of the interfacing for the GPIO modules.

```
#include "../../LIB/LSTD_TYPES.h"
#include "../../LIB/LSTD_COMPILER.h"
#include "../../LIB/LSTD_VALUES.h"
#include "../../LIB/LSTD_BITMATH.h"
#include "GPIO_interface.h"
#include "GPIO_private.h"
#include "GPIO_config.h"
```

Functions

- void [MGPIOx_vLockedPins](#) (void)
Locks the prohibited GPIO PINs.
- void [MGPIOx_vSetPinMode](#) ([u8_t](#) A_u8PortID, [u8_t](#) A_u8PinID, [u8_t](#) A_u8Mode)
- void [MGPIOx_vSetPinOutputType](#) ([u8_t](#) A_u8PortID, [u8_t](#) A_u8PinID, [u8_t](#) A_u8OutputType)
- void [MGPIOx_vSetPinOutputSpeed](#) ([u8_t](#) A_u8PortID, [u8_t](#) A_u8PinID, [u8_t](#) A_u8OutputSpeed)
- void [MGPIOx_vSetPinInputPullType](#) ([u8_t](#) A_u8PortID, [u8_t](#) A_u8PinID, [u8_t](#) A_u8InputPullType)
- [u8_t](#) [MGPIOx_u8GetPinValue](#) ([u8_t](#) A_u8PortID, [u8_t](#) A_u8PinID)
- void [MGPIOx_vSetPinValue](#) ([u8_t](#) A_u8PortID, [u8_t](#) A_u8PinID, [u8_t](#) A_u8PinValue)
- void [MGPIOx_vSetResetAtomic](#) ([u8_t](#) A_u8PortID, [u8_t](#) A_u8PinID, [u8_t](#) A_u8SetResetPinValue)
- void [MGPIOx_vSetAlternateFunctionON](#) ([u8_t](#) A_u8PortID, [u8_t](#) A_u8PinID, [u8_t](#) A_u8AFID)
- void [MGPIOx_vInit](#) ([MGPIOx_ConfigType](#) *A_xPinConfig)
- void [MGPIOx_vTogglePinValue](#) ([u8_t](#) A_u8PortID, [u8_t](#) A_u8PinID)

7.36.1 Detailed Description

This file contains the source code of the interfacing for the GPIO modules.

Author

Ali El Bana & Mo Alaa

Version

2.0

Date

08/22/2022

Definition in file [GPIO_program.c](#).

7.36.2 Function Documentation

7.36.2.1 MGPIOX_vLockedPins()

```
void MGPIOX_vLockedPins (
    void )
```

Locks the prohibited GPIO PINs.

Definition at line 24 of file [GPIO_program.c](#).

```
00025 {
00026
00027     VAR(volatile u32_t) L_u32LockGPIOA = INITIAL_ZERO ;
00028
00029     /* Lock key write sequence */
00030
00031     /* WR LCKR[16] = '1' + LCKR[13,14,15] = *1* */
00032     L_u32LockGPIOA = ( (1UL << LCKK_BIT_POS) | (GPIOA_PIN_POS) ) ;
00033
00034     GPIOA->LCKRx = L_u32LockGPIOA ;
00035
00036     /* WR LCKR[16] = '0' + LCKR[13,14,15] should not change*/
00037     GPIOA->LCKRx = (GPIOA_PIN_POS) ;
00038
00039     /* WR LCKR[16] = '1'+ LCKR[13,14,15] should not change*/
00040     GPIOA->LCKRx = L_u32LockGPIOA ;
00041
00042     /* RD LCKR */
00043     L_u32LockGPIOA = GPIOA->LCKRx ;
00044
00045
00046     VAR(volatile u32_t) L_u32LockGPIOB = INITIAL_ZERO ;
00047
00048     /* Lock key write sequence */
00049
00050     /* WR LCKR[16] = '1' + LCKR[2,3,4] = *1* */
00051     L_u32LockGPIOB = ( (1UL << LCKK_BIT_POS) | (GPIOB_PIN_POS) ) ;
00052
00053     GPIOB->LCKRx = L_u32LockGPIOB ;
00054
00055     /* WR LCKR[16] = '0' + LCKR[2,3,4] should not change*/
00056     GPIOB->LCKRx = (GPIOB_PIN_POS) ;
00057
00058     /* WR LCKR[16] = '1' + LCKR[2,3,4] should not change*/
00059     GPIOB->LCKRx = L_u32LockGPIOB ;
00060
00061     /* RD LCKR */
00062     L_u32LockGPIOB = GPIOB->LCKRx ;
00063
00064
00065 }
```

References [GPIOA](#), [GPIOA_PIN_POS](#), [GPIOB](#), [GPIOB_PIN_POS](#), [INITIAL_ZERO](#), [LCKK_BIT_POS](#), and [VAR](#).

7.36.2.2 MGPIOX_vSetPinMode()

```
void MGPIOX_vSetPinMode (
    u8_t A_u8PortID,
    u8_t A_u8PinID,
    u8_t A_u8Mode )
```

Definition at line 70 of file [GPIO_program.c](#).

```
00071 {
00072
00073     switch (A_u8PortID)
00074     {
00075         case GPIO_PORTA:
00076             CLR_BITS(GPIOA->MODERx, 0b11, A_u8PinID, 2);
00077             SET_BITS(GPIOA->MODERx, A_u8Mode, A_u8PinID, 2);
00078             break;
00079
00080         case GPIO_PORTB:
00081             CLR_BITS(GPIOB->MODERx, 0b11, A_u8PinID, 2);
00082             SET_BITS(GPIOB->MODERx, A_u8Mode, A_u8PinID, 2);
00083             break;
```

```

00084
00085     case GPIO_PORTC:
00086         CLR_BITS(GPIOC->MODERx, 0b11, A_u8PinID, 2);
00087         SET_BITS(GPIOC->MODERx, A_u8Mode, A_u8PinID, 2);
00088         break;
00089     }
00090
00091 }
```

References [CLR_BITS](#), [GPIO_PORTA](#), [GPIO_PORTB](#), [GPIO_PORTC](#), [GPIOA](#), [GPIOB](#), [GPIOC](#), and [SET_BITS](#).

Referenced by [MGPIOx_vInit\(\)](#).

7.36.2.3 MGPIOx_vSetPinOutputType()

```

void MGPIOx_vSetPinOutputType (
    u8_t A_u8PortID,
    u8_t A_u8PinID,
    u8_t A_u8OutputType )
```

Definition at line 96 of file [GPIO_program.c](#).

```

00097 {
00098     switch (A_u8OutputType)
00099     {
00100         case GPIOX_OPENDRAIN:
00101             switch (A_u8PortID)
00102             {
00103                 case GPIO_PORTA:
00104                     SET_BIT(GPIOA->OTYPERx, A_u8PinID);
00105                     break;
00106
00107                 case GPIO_PORTB:
00108                     SET_BIT(GPIOB->OTYPERx, A_u8PinID);
00109                     break;
00110
00111                 case GPIO_PORTC:
00112                     SET_BIT(GPIOC->OTYPERx, A_u8PinID);
00113                     break;
00114             }
00115             break;
00116
00117         case GPIOX_PUSH_PULL:
00118             switch(A_u8PortID)
00119             {
00120                 case GPIO_PORTA:
00121                     CLR_BIT(GPIOA->OTYPERx, A_u8PinID);
00122                     break;
00123
00124                 case GPIO_PORTB:
00125                     CLR_BIT(GPIOB->OTYPERx, A_u8PinID);
00126                     break;
00127
00128                 case GPIO_PORTC:
00129                     CLR_BIT(GPIOC->OTYPERx, A_u8PinID);
00130                     break;
00131             }
00132             break;
00133     }
00134 }
```

References [CLR_BIT](#), [GPIO_PORTA](#), [GPIO_PORTB](#), [GPIO_PORTC](#), [GPIOA](#), [GPIOB](#), [GPIOC](#), [GPIOx_OPENDRAIN](#), [GPIOx_PUSH_PULL](#), and [SET_BIT](#).

Referenced by [MGPIOx_vInit\(\)](#).

7.36.2.4 MGPIOX_vSetPinOutputSpeed()

```
void MGPIOX_vSetPinOutputSpeed (
    u8_t A_u8PortID,
    u8_t A_u8PinID,
    u8_t A_u8OutputSpeed )
```

Definition at line 141 of file [GPIO_program.c](#).

```
00142 {
00143
00144     switch (A_u8PortID)
00145     {
00146         case GPIO_PORTA:
00147             CLR_BITS(GPIOA->OSPEEDRx, 0b11, A_u8PinID, 2);
00148             SET_BITS(GPIOA->OSPEEDRx, A_u8OutputSpeed, A_u8PinID, 2);
00149             break;
00150
00151         case GPIO_PORTB:
00152             CLR_BITS(GPIOB->OSPEEDRx, 0b11, A_u8PinID, 2);
00153             SET_BITS(GPIOB->OSPEEDRx, A_u8OutputSpeed, A_u8PinID, 2);
00154             break;
00155
00156         case GPIO_PORTC:
00157             CLR_BITS(GPIOC->OSPEEDRx, 0b11, A_u8PinID, 2);
00158             SET_BITS(GPIOC->OSPEEDRx, A_u8OutputSpeed, A_u8PinID, 2);
00159             break;
00160     }
00161
00162 }
```

References [CLR_BITS](#), [GPIO_PORTA](#), [GPIO_PORTB](#), [GPIO_PORTC](#), [GPIOA](#), [GPIOB](#), [GPIOC](#), and [SET_BITS](#).

Referenced by [MGPIOX_vInit\(\)](#).

7.36.2.5 MGPIOX_vSetPinInputPullType()

```
void MGPIOX_vSetPinInputPullType (
    u8_t A_u8PortID,
    u8_t A_u8PinID,
    u8_t A_u8InputPullType )
```

Definition at line 167 of file [GPIO_program.c](#).

```
00168 {
00169
00170     switch (A_u8PortID)
00171     {
00172         case GPIO_PORTA:
00173             CLR_BITS(GPIOA->PUPDRx, 0b11, A_u8PinID, 2);
00174             SET_BITS(GPIOA->PUPDRx, A_u8InputPullType, A_u8PinID, 2);
00175             break;
00176
00177         case GPIO_PORTB:
00178             CLR_BITS(GPIOB->PUPDRx, 0b11, A_u8PinID, 2);
00179             SET_BITS(GPIOB->PUPDRx, A_u8InputPullType, A_u8PinID, 2);
00180             break;
00181
00182         case GPIO_PORTC:
00183             CLR_BITS(GPIOC->PUPDRx, 0b11, A_u8PinID, 2);
00184             SET_BITS(GPIOC->PUPDRx, A_u8InputPullType, A_u8PinID, 2);
00185             break;
00186     }
00187
00188 }
```

References [CLR_BITS](#), [GPIO_PORTA](#), [GPIO_PORTB](#), [GPIO_PORTC](#), [GPIOA](#), [GPIOB](#), [GPIOC](#), and [SET_BITS](#).

Referenced by [MGPIOX_vInit\(\)](#).

7.36.2.6 MGPIOX_u8GetPinValue()

```
u8_t MGPIOX_u8GetPinValue (
    u8_t A_u8PortID,
    u8_t A_u8PinID )
```

Definition at line 193 of file [GPIO_program.c](#).

```
00194 {
00195
00196     VAR(u8_t) L_u8PinValue = INITIAL_ZERO;
00197
00198     switch (A_u8PortID)
00199     {
00200         case GPIO_PORTA:
00201             L_u8PinValue = GET_BIT(GPIOA->IDRx, A_u8PinID);
00202             break;
00203
00204         case GPIO_PORTB:
00205             L_u8PinValue = GET_BIT(GPIOB->IDRx, A_u8PinID);
00206             break;
00207
00208         case GPIO_PORTC:
00209             L_u8PinValue = GET_BIT(GPIOC->IDRx, A_u8PinID);
00210             break;
00211     }
00212
00213     return L_u8PinValue;
00214
00215 }
```

References [GET_BIT](#), [GPIO_PORTA](#), [GPIO_PORTB](#), [GPIO_PORTC](#), [GPIOA](#), [GPIOB](#), [GPIOC](#), [INITIAL_ZERO](#), and [VAR](#).

7.36.2.7 MGPIOX_vSetPinValue()

```
void MGPIOX_vSetPinValue (
    u8_t A_u8PortID,
    u8_t A_u8PinID,
    u8_t A_u8PinValue )
```

Definition at line 220 of file [GPIO_program.c](#).

```
00221 {
00222
00223     switch (A_u8PinValue)
00224     {
00225         case GPIOX_HIGH:
00226             switch (A_u8PortID)
00227             {
00228                 case GPIO_PORTA:
00229                     SET_BIT(GPIOA->ODRx, A_u8PinID);
00230                     break;
00231
00232                 case GPIO_PORTB:
00233                     SET_BIT(GPIOB->ODRx, A_u8PinID);
00234                     break;
00235
00236                 case GPIO_PORTC:
00237                     SET_BIT(GPIOC->ODRx, A_u8PinID);
00238                     break;
00239             }
00240             break;
00241
00242         case GPIOX_LOW:
00243             switch (A_u8PortID)
00244             {
00245                 case GPIO_PORTA:
00246                     CLR_BIT(GPIOA->ODRx, A_u8PinID);
00247                     break;
00248
00249                 case GPIO_PORTB:
00250                     CLR_BIT(GPIOB->ODRx, A_u8PinID);
00251                     break;
00252             }
00253     }
00254 }
```

```

00252
00253     case GPIO_PORTC:
00254         CLR_BIT(GPIOC->ODRx, A_u8PinID);
00255         break;
00256     }
00257     break;
00258 }
00259
00260 }
```

References [CLR_BIT](#), [GPIO_PORTA](#), [GPIO_PORTB](#), [GPIO_PORTC](#), [GPIOA](#), [GPIOB](#), [GPIOC](#), [GPIOx_HIGH](#), [GPIOx_LOW](#), and [SET_BIT](#).

7.36.2.8 MGPIOx_vSetResetAtomic()

```

void MGPIOx_vSetResetAtomic (
    u8_t A_u8PortID,
    u8_t A_u8PinID,
    u8_t A_u8SetResetPinValue )
```

Definition at line 265 of file [GPIO_program.c](#).

```

00266 {
00267
00268     switch (A_u8PortID)
00269     {
00270
00271         case GPIO_PORTA:
00272
00273             switch (A_u8SetResetPinValue)
00274             {
00275                 case GPIOx_HIGH:
00276                     GPIOA->BSRRx = (1 << A_u8PinID);
00277                     break;
00278
00279                 case GPIOx_LOW:
00280                     GPIOA->BSRRx = (1 << (A_u8PinID + 16));
00281                     break;
00282             }
00283
00284             break;
00285
00286         case GPIO_PORTB:
00287
00288             switch (A_u8SetResetPinValue)
00289             {
00290                 case GPIOx_HIGH:
00291                     GPIOB->BSRRx = (1 << A_u8PinID);
00292                     break;
00293
00294                 case GPIOx_LOW:
00295                     GPIOB->BSRRx = (1 << (A_u8PinID + 16));
00296                     break;
00297             }
00298
00299             break;
00300
00301         case GPIO_PORTC:
00302
00303             switch (A_u8SetResetPinValue)
00304             {
00305                 case GPIOx_HIGH:
00306                     GPIOC->BSRRx = (1 << A_u8PinID);
00307                     break;
00308
00309                 case GPIOx_LOW:
00310                     GPIOC->BSRRx = (1 << (A_u8PinID + 16));
00311                     break;
00312             }
00313
00314             break;
00315     }
00316
00317 }
```

References [GPIO_PORTA](#), [GPIO_PORTB](#), [GPIO_PORTC](#), [GPIOA](#), [GPIOB](#), [GPIOC](#), [GPIOx_HIGH](#), and [GPIOx_LOW](#).

7.36.2.9 MGPIox_vSetAlternateFunctionON()

```
void MGPIox_vSetAlternateFunctionON (
    u8_t A_u8PortID,
    u8_t A_u8PinID,
    u8_t A_u8AFID )
```

Definition at line 322 of file [GPIO_program.c](#).

```
00323 {
00324
00325     switch (A_u8PortID)
00326     {
00327         case GPIO_PORTA:
00328             switch (A_u8PinID)
00329             {
00330                 case GPIOx_PIN0:
00331                 case GPIOx_PIN1:
00332                 case GPIOx_PIN2:
00333                 case GPIOx_PIN3:
00334                 case GPIOx_PIN4:
00335                 case GPIOx_PIN5:
00336                 case GPIOx_PIN6:
00337                 case GPIOx_PIN7:
00338                     CLR_BITS(GPIOA->AFRLx, 0b1111, A_u8PinID, 4);
00339                     SET_BITS(GPIOA->AFRLx, A_u8AFID, A_u8PinID, 4);
00340                     break;
00341
00342                 case GPIOx_PIN8:
00343                 case GPIOx_PIN9:
00344                 case GPIOx_PIN10:
00345                 case GPIOx_PIN11:
00346                 case GPIOx_PIN12:
00347                 case GPIOx_PIN13:
00348                 case GPIOx_PIN14:
00349                 case GPIOx_PIN15:
00350                     CLR_BITS(GPIOA->AFRHx, 0b1111, A_u8PinID, 4);
00351                     SET_BITS(GPIOA->AFRHx, A_u8AFID, A_u8PinID, 4);
00352                     break;
00353             }
00354             break;
00355
00356         case GPIO_PORTB:
00357             switch (A_u8PinID)
00358             {
00359                 case GPIOx_PIN0:
00360                 case GPIOx_PIN1:
00361                 case GPIOx_PIN2:
00362                 case GPIOx_PIN3:
00363                 case GPIOx_PIN4:
00364                 case GPIOx_PIN5:
00365                 case GPIOx_PIN6:
00366                 case GPIOx_PIN7:
00367                     CLR_BITS(GPIOB->AFRLx, 0b1111, A_u8PinID, 4);
00368                     SET_BITS(GPIOB->AFRLx, A_u8AFID, A_u8PinID, 4);
00369                     break;
00370
00371                 case GPIOx_PIN8:
00372                 case GPIOx_PIN9:
00373                 case GPIOx_PIN10:
00374                 case GPIOx_PIN11:
00375                 case GPIOx_PIN12:
00376                 case GPIOx_PIN13:
00377                 case GPIOx_PIN14:
00378                 case GPIOx_PIN15:
00379                     CLR_BITS(GPIOB->AFRHx, 0b1111, A_u8PinID-8, 4);
00380                     SET_BITS(GPIOB->AFRHx, A_u8AFID, A_u8PinID-8, 4);
00381                     break;
00382             }
00383             break;
00384
00385         case GPIO_PORTC:
00386             switch (A_u8PinID)
00387             {
00388                 case GPIOx_PIN0:
00389                 case GPIOx_PIN1:
00390                 case GPIOx_PIN2:
00391                 case GPIOx_PIN3:
00392                 case GPIOx_PIN4:
00393                 case GPIOx_PIN5:
00394                 case GPIOx_PIN6:
00395                 case GPIOx_PIN7:
00396                     CLR_BITS(GPIOC->AFRLx, 0b1111, A_u8PinID, 4);
00397                     SET_BITS(GPIOC->AFRLx, A_u8AFID, A_u8PinID, 4);
```

```

00398         break;
00399
00400     case GPIOx_PIN8:
00401     case GPIOx_PIN9:
00402     case GPIOx_PIN10:
00403     case GPIOx_PIN11:
00404     case GPIOx_PIN12:
00405     case GPIOx_PIN13:
00406     case GPIOx_PIN14:
00407     case GPIOx_PIN15:
00408         CLR_BITS(GPIOC->AFRHx, 0b1111, A_u8PinID, 4);
00409         SET_BITS(GPIOC->AFRHx, A_u8AFID, A_u8PinID, 4);
00410         break;
00411     }
00412     break;
00413 }
00414
00415 }
```

References [CLR_BITS](#), [GPIO_PORTA](#), [GPIO_PORTB](#), [GPIO_PORTC](#), [GPIOA](#), [GPIOB](#), [GPIOC](#), [GPIOx_PIN0](#), [GPIOx_PIN1](#), [GPIOx_PIN10](#), [GPIOx_PIN11](#), [GPIOx_PIN12](#), [GPIOx_PIN13](#), [GPIOx_PIN14](#), [GPIOx_PIN15](#), [GPIOx_PIN2](#), [GPIOx_PIN3](#), [GPIOx_PIN4](#), [GPIOx_PIN5](#), [GPIOx_PIN6](#), [GPIOx_PIN7](#), [GPIOx_PIN8](#), [GPIOx_PIN9](#), and [SET_BITS](#).

Referenced by [MGPIOx_vInit\(\)](#).

7.36.2.10 MGPIOx_vInit()

```
void MGPIOx_vInit (
    MGPIOx_ConfigType * A_xPinConfig )
```

Definition at line 420 of file [GPIO_program.c](#).

```

00421 {
00422
00423     MGPIOx_vSetPinMode           (A_xPinConfig->Port, A_xPinConfig->Pin, A_xPinConfig->Mode
00424     );
00425     MGPIOx_vSetPinOutputType    (A_xPinConfig->Port, A_xPinConfig->Pin, A_xPinConfig->OutputType
00426     );
00427     MGPIOx_vSetPinOutputSpeed   (A_xPinConfig->Port, A_xPinConfig->Pin, A_xPinConfig->OutputSpeed
00428     );
00429     MGPIOx_vSetPinInputPullType (A_xPinConfig->Port, A_xPinConfig->Pin, A_xPinConfig->InputType
00430     );
00431     MGPIOx_vSetAlternateFunctionON (A_xPinConfig->Port, A_xPinConfig->Pin, A_xPinConfig->AF_Type
00432     );
00433 }
```

References [MGPIOx_ConfigType::AF_Type](#), [MGPIOx_ConfigType::InputType](#), [MGPIOx_vSetAlternateFunctionON\(\)](#), [MGPIOx_vSetPinInputPullType\(\)](#), [MGPIOx_vSetPinMode\(\)](#), [MGPIOx_vSetPinOutputSpeed\(\)](#), [MGPIOx_vSetPinOutputType\(\)](#), [MGPIOx_ConfigType::Mode](#), [MGPIOx_ConfigType::OutputSpeed](#), [MGPIOx_ConfigType::OutputType](#), [MGPIOx_ConfigType::Pin](#), and [MGPIOx_ConfigType::Port](#).

7.36.2.11 MGPIOx_vTogglePinValue()

```
void MGPIOx_vTogglePinValue (
    u8_t A_u8PortID,
    u8_t A_u8PinID )
```

Definition at line 434 of file [GPIO_program.c](#).

```

00435 {
00436     switch (A_u8PortID)
00437     {
00438 }
```

```

00439     case GPIO_PORTA:
00440         TOGGLE_BIT(GPIOA->ODRx, A_u8PinID);
00441         break;
00442
00443     case GPIO_PORTB:
00444         TOGGLE_BIT(GPIOB->ODRx, A_u8PinID);
00445         break;
00446
00447     case GPIO_PORTC:
00448         TOGGLE_BIT(GPIOC->ODRx, A_u8PinID);
00449         break;
00450 }
00451 }
00452 }
```

References [GPIO_PORTA](#), [GPIO_PORTB](#), [GPIO_PORTC](#), [GPIOA](#), [GPIOB](#), [GPIOC](#), and [TOGGLE_BIT](#).

7.37 GPIO_program.c

[Go to the documentation of this file.](#)

```

00001
00009 /*****
00010 /*           Include headers           */
00011 *****/
00012 #include "../../../LIB/LSTD_TYPES.h"
00013 #include "../../../LIB/LSTD_COMPILER.h"
00014 #include "../../../LIB/LSTD_VALUES.h"
00015 #include "../../../LIB/LSTD_BITMATH.h"
00016 #include "GPIO_interface.h"
00017 #include "GPIO_private.h"
00018 #include "GPIO_config.h"
00019
00020 /*****
00021 /*           Functions' implementations      */
00022 *****/
00023
00024 FUNC(void) MGPIox_vLockedPins(void)
00025 {
00026
00027     VAR(volatile u32_t) L_u32LockGPIOA = INITIAL_ZERO ;
00028
00029     /* Lock key write sequence */
00030
00031     /* WR LCKR[16] = '1' + LCKR[13,14,15] = ↴1↖ */
00032     L_u32LockGPIOA = ( (1UL << LCKK_BIT_POS) | (GPIOA_PIN_POS) ) ;
00033
00034     GPIOA->LCKRx = L_u32LockGPIOA ;
00035
00036     /* WR LCKR[16] = '0' + LCKR[13,14,15] should not change*/
00037     GPIOA->LCKRx = (GPIOA_PIN_POS) ;
00038
00039     /* WR LCKR[16] = '1'+ LCKR[13,14,15] should not change*/
00040     GPIOA->LCKRx = L_u32LockGPIOA ;
00041
00042     /* RD LCKR */
00043     L_u32LockGPIOA = GPIOA->LCKRx ;
00044
00045
00046     VAR(volatile u32_t) L_u32LockGPIOB = INITIAL_ZERO ;
00047
00048     /* Lock key write sequence */
00049
00050     /* WR LCKR[16] = '1' + LCKR[2,3,4] = ↴1↖ */
00051     L_u32LockGPIOB = ( (1UL << LCKK_BIT_POS) | (GPIOB_PIN_POS) ) ;
00052
00053     GPIOB->LCKRx = L_u32LockGPIOB ;
00054
00055     /* WR LCKR[16] = '0' + LCKR[2,3,4] should not change*/
00056     GPIOB->LCKRx = (GPIOB_PIN_POS) ;
00057
00058     /* WR LCKR[16] = '1' + LCKR[2,3,4] should not change*/
00059     GPIOB->LCKRx = L_u32LockGPIOB ;
00060
00061     /* RD LCKR */
00062     L_u32LockGPIOB = GPIOB->LCKRx ;
00063
00064
00065 }
00066
00067 /*****
00068 *****/

```

```

00069
00070 FUNC(void) MGPIox_vSetPinMode(VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID, VAR(u8_t) A_u8Mode)
00071 {
00072
00073     switch (A_u8PortID)
00074     {
00075         case GPIO_PORTA:
00076             CLR_BITS(GPIOA->MODERx, 0b11, A_u8PinID, 2);
00077             SET_BITS(GPIOA->MODERx, A_u8Mode, A_u8PinID, 2);
00078             break;
00079
00080         case GPIO_PORTB:
00081             CLR_BITS(GPIOB->MODERx, 0b11, A_u8PinID, 2);
00082             SET_BITS(GPIOB->MODERx, A_u8Mode, A_u8PinID, 2);
00083             break;
00084
00085         case GPIO_PORTC:
00086             CLR_BITS(GPIOC->MODERx, 0b11, A_u8PinID, 2);
00087             SET_BITS(GPIOC->MODERx, A_u8Mode, A_u8PinID, 2);
00088             break;
00089     }
00090
00091 }
00092
00093 /*****
00094 *****/
00095
00096 FUNC(void) MGPIox_vSetPinOutputType(VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID, VAR(u8_t)
00097     A_u8OutputType)
00098 {
00099     switch (A_u8OutputType)
00100     {
00101         case GPIO_OPENDRAIN:
00102             switch (A_u8PortID)
00103             {
00104                 case GPIO_PORTA:
00105                     SET_BIT(GPIOA->OTYPERx, A_u8PinID);
00106                     break;
00107
00108                 case GPIO_PORTB:
00109                     SET_BIT(GPIOB->OTYPERx, A_u8PinID);
00110                     break;
00111
00112                 case GPIO_PORTC:
00113                     SET_BIT(GPIOC->OTYPERx, A_u8PinID);
00114                     break;
00115             }
00116             break;
00117
00118         case GPIO_PUSH_PULL:
00119             switch(A_u8PortID)
00120             {
00121                 case GPIO_PORTA:
00122                     CLR_BIT(GPIOA->OTYPERx, A_u8PinID);
00123                     break;
00124
00125                 case GPIO_PORTB:
00126                     CLR_BIT(GPIOB->OTYPERx, A_u8PinID);
00127                     break;
00128
00129                 case GPIO_PORTC:
00130                     CLR_BIT(GPIOC->OTYPERx, A_u8PinID);
00131                     break;
00132             }
00133             break;
00134     }
00135
00136 }
00137
00138 /*****
00139 *****/
00140
00141 FUNC(void) MGPIox_vSetPinOutputSpeed(VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID, VAR(u8_t)
00142     A_u8OutputSpeed)
00143 {
00144     switch (A_u8PortID)
00145     {
00146         case GPIO_PORTA:
00147             CLR_BITS(GPIOA->OSPEEDR, 0b11, A_u8PinID, 2);
00148             SET_BITS(GPIOA->OSPEEDR, A_u8OutputSpeed, A_u8PinID, 2);
00149             break;
00150
00151         case GPIO_PORTB:
00152             CLR_BITS(GPIOB->OSPEEDR, 0b11, A_u8PinID, 2);
00153             SET_BITS(GPIOB->OSPEEDR, A_u8OutputSpeed, A_u8PinID, 2);

```

```

00154         break;
00155
00156     case GPIO_PORTC:
00157         CLR_BITS(GPIOC->OSPEEDRx, 0b11, A_u8PinID, 2);
00158         SET_BITS(GPIOC->OSPEEDRx, A_u8OutputSpeed, A_u8PinID, 2);
00159         break;
00160     }
00161 }
00163
00164 /*****
00165 ****/
00166
00167 FUNC(void) MGPIOX_vSetPinInputPullType(VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID, VAR(u8_t)
A_u8InputPullType)
00168 {
00169
00170     switch (A_u8PortID)
00171     {
00172         case GPIO_PORTA:
00173             CLR_BITS(GPIOA->PUPDRx, 0b11, A_u8PinID, 2);
00174             SET_BITS(GPIOA->PUPDRx, A_u8InputPullType, A_u8PinID, 2);
00175             break;
00176
00177         case GPIO_PORTB:
00178             CLR_BITS(GPIOB->PUPDRx, 0b11, A_u8PinID, 2);
00179             SET_BITS(GPIOB->PUPDRx, A_u8InputPullType, A_u8PinID, 2);
00180             break;
00181
00182         case GPIO_PORTC:
00183             CLR_BITS(GPIOC->PUPDRx, 0b11, A_u8PinID, 2);
00184             SET_BITS(GPIOC->PUPDRx, A_u8InputPullType, A_u8PinID, 2);
00185             break;
00186     }
00187
00188 }
00189
00190 *****/
00191 *****/
00192
00193 FUNC(u8_t) MGPIOX_u8GetPinValue(VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID)
00194 {
00195
00196     VAR(u8_t) L_u8PinValue = INITIAL_ZERO;
00197
00198     switch (A_u8PortID)
00199     {
00200         case GPIO_PORTA:
00201             L_u8PinValue = GET_BIT(GPIOA->IDRx, A_u8PinID);
00202             break;
00203
00204         case GPIO_PORTB:
00205             L_u8PinValue = GET_BIT(GPIOB->IDRx, A_u8PinID);
00206             break;
00207
00208         case GPIO_PORTC:
00209             L_u8PinValue = GET_BIT(GPIOC->IDRx, A_u8PinID);
00210             break;
00211     }
00212
00213     return L_u8PinValue;
00214
00215 }
00216
00217 *****/
00218 *****/
00219
00220 FUNC(void) MGPIOX_vSetValue(VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID, VAR(u8_t) A_u8PinValue)
00221 {
00222
00223     switch (A_u8PinValue)
00224     {
00225         case GPIOx_HIGH:
00226             switch (A_u8PortID)
00227             {
00228                 case GPIO_PORTA:
00229                     SET_BIT(GPIOA->ODRx, A_u8PinID);
00230                     break;
00231
00232                 case GPIO_PORTB:
00233                     SET_BIT(GPIOB->ODRx, A_u8PinID);
00234                     break;
00235
00236                 case GPIO_PORTC:
00237                     SET_BIT(GPIOC->ODRx, A_u8PinID);
00238                     break;
00239             }
00240     }
00241
00242 }
```

```

00240         break;
00241
00242     case GPIOx_LOW:
00243         switch( A_u8PortID )
00244     {
00245         case GPIO_PORTA:
00246             CLR_BIT(GPIOA->ODRx, A_u8PinID);
00247             break;
00248
00249         case GPIO_PORTB:
00250             CLR_BIT(GPIOB->ODRx, A_u8PinID);
00251             break;
00252
00253         case GPIO_PORTC:
00254             CLR_BIT(GPIOC->ODRx, A_u8PinID);
00255             break;
00256     }
00257     break;
00258 }
00259
00260 }
00261
00262 /*****
00263 *****/
00264
00265 FUNC(void) MGPIox_vSetResetAtomic(VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID, VAR(u8_t)
00266     A_u8SetResetPinValue)
00267 {
00268     switch (A_u8PortID)
00269     {
00270
00271         case GPIO_PORTA:
00272
00273             switch (A_u8SetResetPinValue)
00274         {
00275             case GPIOx_HIGH:
00276                 GPIOA->BSRRx = (1 << A_u8PinID);
00277                 break;
00278
00279             case GPIOx_LOW:
00280                 GPIOA->BSRRx = (1 << (A_u8PinID + 16));
00281                 break;
00282         }
00283
00284         break;
00285
00286         case GPIO_PORTB:
00287
00288             switch (A_u8SetResetPinValue)
00289         {
00290             case GPIOx_HIGH:
00291                 GPIOB->BSRRx = (1 << A_u8PinID);
00292                 break;
00293
00294             case GPIOx_LOW:
00295                 GPIOB->BSRRx = (1 << (A_u8PinID + 16));
00296                 break;
00297         }
00298
00299         break;
00300
00301         case GPIO_PORTC:
00302
00303             switch (A_u8SetResetPinValue)
00304         {
00305             case GPIOx_HIGH:
00306                 GPIOC->BSRRx = (1 << A_u8PinID);
00307                 break;
00308
00309             case GPIOx_LOW:
00310                 GPIOC->BSRRx = (1 << (A_u8PinID + 16));
00311                 break;
00312         }
00313
00314         break;
00315     }
00316
00317 }
00318
00319 /*****
00320 *****/
00321
00322 FUNC(void) MGPIox_vSetAlternateFunctionON(VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID, VAR(u8_t)
00323     A_u8AFID)
00324 {

```

```

00325     switch (A_u8PortID)
00326     {
00327         case GPIO_PORTA:
00328             switch (A_u8PinID)
00329             {
00330                 case GPIOx_PIN0:
00331                 case GPIOx_PIN1:
00332                 case GPIOx_PIN2:
00333                 case GPIOx_PIN3:
00334                 case GPIOx_PIN4:
00335                 case GPIOx_PIN5:
00336                 case GPIOx_PIN6:
00337                 case GPIOx_PIN7:
00338                     CLR_BITS(GPIOA->AFRLx, 0b1111, A_u8PinID, 4);
00339                     SET_BITS(GPIOA->AFRLx, A_u8AFID, A_u8PinID, 4);
00340                     break;
00341
00342                 case GPIOx_PIN8:
00343                 case GPIOx_PIN9:
00344                 case GPIOx_PIN10:
00345                 case GPIOx_PIN11:
00346                 case GPIOx_PIN12:
00347                 case GPIOx_PIN13:
00348                 case GPIOx_PIN14:
00349                 case GPIOx_PIN15:
00350                     CLR_BITS(GPIOA->AFRHx, 0b1111, A_u8PinID, 4);
00351                     SET_BITS(GPIOA->AFRHx, A_u8AFID, A_u8PinID, 4);
00352                     break;
00353             }
00354             break;
00355
00356         case GPIO_PORTB:
00357             switch (A_u8PinID)
00358             {
00359                 case GPIOx_PIN0:
00360                 case GPIOx_PIN1:
00361                 case GPIOx_PIN2:
00362                 case GPIOx_PIN3:
00363                 case GPIOx_PIN4:
00364                 case GPIOx_PIN5:
00365                 case GPIOx_PIN6:
00366                 case GPIOx_PIN7:
00367                     CLR_BITS(GPIOB->AFRLx, 0b1111, A_u8PinID, 4);
00368                     SET_BITS(GPIOB->AFRLx, A_u8AFID, A_u8PinID, 4);
00369                     break;
00370
00371                 case GPIOx_PIN8:
00372                 case GPIOx_PIN9:
00373                 case GPIOx_PIN10:
00374                 case GPIOx_PIN11:
00375                 case GPIOx_PIN12:
00376                 case GPIOx_PIN13:
00377                 case GPIOx_PIN14:
00378                 case GPIOx_PIN15:
00379                     CLR_BITS(GPIOB->AFRHx, 0b1111, A_u8PinID-8, 4);
00380                     SET_BITS(GPIOB->AFRHx, A_u8AFID, A_u8PinID-8, 4);
00381                     break;
00382             }
00383             break;
00384
00385         case GPIO_PORTC:
00386             switch (A_u8PinID)
00387             {
00388                 case GPIOx_PIN0:
00389                 case GPIOx_PIN1:
00390                 case GPIOx_PIN2:
00391                 case GPIOx_PIN3:
00392                 case GPIOx_PIN4:
00393                 case GPIOx_PIN5:
00394                 case GPIOx_PIN6:
00395                 case GPIOx_PIN7:
00396                     CLR_BITS(GPIOC->AFRLx, 0b1111, A_u8PinID, 4);
00397                     SET_BITS(GPIOC->AFRLx, A_u8AFID, A_u8PinID, 4);
00398                     break;
00399
00400                 case GPIOx_PIN8:
00401                 case GPIOx_PIN9:
00402                 case GPIOx_PIN10:
00403                 case GPIOx_PIN11:
00404                 case GPIOx_PIN12:
00405                 case GPIOx_PIN13:
00406                 case GPIOx_PIN14:
00407                 case GPIOx_PIN15:
00408                     CLR_BITS(GPIOC->AFRHx, 0b1111, A_u8PinID, 4);
00409                     SET_BITS(GPIOC->AFRHx, A_u8AFID, A_u8PinID, 4);
00410                     break;
00411             }

```

```

00412         break;
00413     }
00414
00415 }
00416
00417 /*****
00418 *****/
00419
00420 FUNC(void) MGPIOx_vInit(P2VAR(MGPIOx_ConfigType) A_xPinConfig)
00421 {
00422
00423     MGPIOx_vSetPinMode          (A_xPinConfig->Port, A_xPinConfig->Pin, A_xPinConfig->Mode
00424 );
00425     MGPIOx_vSetPinOutputType    (A_xPinConfig->Port, A_xPinConfig->Pin, A_xPinConfig->OutputType
00426 );
00427     MGPIOx_vSetPinOutputSpeed   (A_xPinConfig->Port, A_xPinConfig->Pin, A_xPinConfig->OutputSpeed
00428 );
00429     MGPIOx_vSetPinInputPullType (A_xPinConfig->Port, A_xPinConfig->Pin, A_xPinConfig->InputType
00430 );
00431     MGPIOx_vSetAlternateFunctionON (A_xPinConfig->Port, A_xPinConfig->Pin, A_xPinConfig->AF_Type
00432 );
00433
00434 FUNC(void) MGPIOx_vTogglePinValue(VAR(u8_t) A_u8PortID, VAR(u8_t) A_u8PinID)
00435 {
00436
00437     switch (A_u8PortID)
00438     {
00439         case GPIO_PORTA:
00440             TOGGLE_BIT(GPIOA->ODRx, A_u8PinID);
00441             break;
00442
00443         case GPIO_PORTB:
00444             TOGGLE_BIT(GPIOB->ODRx, A_u8PinID);
00445             break;
00446
00447         case GPIO_PORTC:
00448             TOGGLE_BIT(GPIOC->ODRx, A_u8PinID);
00449             break;
00450     }
00451
00452 }
00453
00454 /*****
00455 *****/
00456
00457
00458

```

7.38 COTS/MCAL/NVIC/NVIC_config.h File Reference

7.39 NVIC_config.h

Go to the documentation of this file.

```

00001 /* FILENAME: NVIC_config
00002 * Author: Ali El Bana
00003 * Version: V1.0
00004 * DATE: Thu 08/25/2022
00005 */
00006 #ifndef _NVIC_config_H
00007 #define _NVIC_config_H
00008
00009
00010
00011
00012
00013 #endif // _NVIC_config_H

```

7.40 COTS/MCAL/NVIC/NVIC_interface.h File Reference

This file contains the interface information and addresses for the NVIC module.

Macros

- `#define _16GROUP_NoSub_Priorities (0b011)`
16 groups and 0 sub-groups
- `#define _8GROUP_2Sub_Priorities (0b100)`
8 groups and 2 sub-groups
- `#define _4GROUP_4Sub_Priorities (0b101)`
4 groups and 4 sub-groups
- `#define _2GROUP_8Sub_Priorities (0b110)`
2 groups and 8 sub-groups
- `#define NoGROUP_16Sub_Priorities (0b111)`
0 groups and 16 sub-groups
- `#define GROUP_4BITS (0b011)`
16 groups and 0 sub-groups
- `#define GROUP_3BITS (0b100)`
8 groups and 2 sub-groups
- `#define GROUP_2BITS (0b101)`
4 groups and 4 sub-groups
- `#define GROUP_1BITS (0b110)`
2 groups and 8 sub-groups
- `#define GROUP_0BITS (0b111)`
0 groups and 16 sub-groups
- `#define NO_GROUP_PRIORITY (0)`
No group priority.
- `#define GROUP_PRIORITY_0 (0)`
Priority: 0.
- `#define GROUP_PRIORITY_1 (1)`
Priority: 1.
- `#define GROUP_PRIORITY_2 (2)`
Priority: 2.
- `#define GROUP_PRIORITY_3 (3)`
Priority: 3.
- `#define GROUP_PRIORITY_4 (4)`
Priority: 4.
- `#define GROUP_PRIORITY_5 (5)`
Priority: 5.
- `#define GROUP_PRIORITY_6 (6)`
Priority: 6.
- `#define GROUP_PRIORITY_7 (7)`
Priority: 7.
- `#define GROUP_PRIORITY_8 (8)`
Priority: 8.
- `#define GROUP_PRIORITY_9 (9)`
Priority: 9.
- `#define GROUP_PRIORITY_10 (10)`
Priority: 10.
- `#define GROUP_PRIORITY_11 (11)`
Priority: 11.
- `#define GROUP_PRIORITY_12 (12)`
Priority: 12.
- `#define GROUP_PRIORITY_13 (13)`

- #define GROUP_PRIORITY_14 (14)
Priority: 13.
- #define GROUP_PRIORITY_15 (15)
Priority: 14.
- #define NO_SUB_PRIORITY (0)
No sub-group priority.
- #define SUB_PRIORITY_0 (0)
Priority: 0.
- #define SUB_PRIORITY_1 (1)
Priority: 1.
- #define SUB_PRIORITY_2 (2)
Priority: 2.
- #define SUB_PRIORITY_3 (3)
Priority: 3.
- #define SUB_PRIORITY_4 (4)
Priority: 4.
- #define SUB_PRIORITY_5 (5)
Priority: 5.
- #define SUB_PRIORITY_6 (6)
Priority: 6.
- #define SUB_PRIORITY_7 (7)
Priority: 7.
- #define SUB_PRIORITY_8 (8)
Priority: 8.
- #define SUB_PRIORITY_9 (9)
Priority: 9.
- #define SUB_PRIORITY_10 (10)
Priority: 10.
- #define SUB_PRIORITY_11 (11)
Priority: 11.
- #define SUB_PRIORITY_12 (12)
Priority: 12.
- #define SUB_PRIORITY_13 (13)
Priority: 13.
- #define SUB_PRIORITY_14 (14)
Priority: 14.
- #define SUB_PRIORITY_15 (15)
Priority: 15.
- #define WWDG (0)
Window Watchdog (WWDG) Interrupt.
- #define EXTI16 (1)
EXTI Line 16 interrupt / PVD through EXTI line detection interrupt.
- #define EXTI21 (2)
EXTI Line 21 interrupt / Tamper andTimeStamp interrupts through the EXTI line.
- #define EXTI22 (3)
EXTI Line 22 interrupt / RTC (Real-time clock) wakeup interrupt through the EXTI line.
- #define FLASH (4)
Flash global interrupt.
- #define RCC (5)
RCC global interrupt.
- #define EXTIO (6)

- #define **EXTI1** (7)
 - EXTI Line 0 interrupt.*
- #define **EXTI2** (8)
 - EXTI Line 1 interrupt.*
- #define **EXTI3** (9)
 - EXTI Line 2 interrupt.*
- #define **EXTI4** (10)
 - EXTI Line 3 interrupt.*
- #define **DMA1_STREAM0** (11)
 - DMA1 (Direct Memory Access) Steam 0 global interrupt.*
- #define **DMA1_STREAM1** (12)
 - DMA1 (Direct Memory Access) Steam 1 global interrupt.*
- #define **DMA1_STREAM2** (13)
 - DMA1 (Direct Memory Access) Steam 2 global interrupt.*
- #define **DMA1_STREAM3** (14)
 - DMA1 (Direct Memory Access) Steam 3 global interrupt.*
- #define **DMA1_STREAM4** (15)
 - DMA1 (Direct Memory Access) Steam 4 global interrupt.*
- #define **DMA1_STREAM5** (16)
 - DMA1 (Direct Memory Access) Steam 5 global interrupt.*
- #define **DMA1_STREAM6** (17)
 - DMA1 (Direct Memory Access) Steam 6 global interrupt.*
- #define **ADC** (18)
 - ADC (Analog-To-Digital Converter) ADC1 global interrupt.*
- #define **EXTI9** (23)
 - EXTI Lines [9:5] interrupts.*
- #define **TIM1_BRK_TIM9** (24)
 - TIM1 (Timer 1) break interrupt and TIM9 (Timer 9) global interrupt.*
- #define **TIM1_UP_TIM10** (25)
 - TIM1 (Timer 1) update interrupt and TIM10 (Timer 10) global interrupt.*
- #define **TIM1_TRG_COM_TIM11** (26)
 - TIM1 (Timer 1) trigger and commutation itnerrupts and TIM11 (Timer 11) global interrupts.*
- #define **TIM1_CC** (27)
 - TIM1 (Timer 1) capture compare interrupt.*
- #define **TIM2** (28)
 - TIM2 (Timer 2) global interrupt.*
- #define **TIM3** (29)
 - TIM3 (Timer 3) global interrupt.*
- #define **TIM4** (30)
 - TIM4 (Timer 4) global interrupt.*
- #define **I2C1_EV** (31)
 - I2C1 (Inter-integrated Circuit 1) event interrupt.*
- #define **I2C1_ER** (32)
 - I2C1 (Inter-integrated Circuit 1) error interrupt.*
- #define **I2C2_EV** (33)
 - I2C2 (Inter-integrated Circuit 2) event interrupt.*
- #define **I2C2_ER** (34)
 - I2C2 (Inter-integrated Circuit 2) error interrupt.*
- #define **SPI1** (35)
 - SPI1 (Serial Peripheral Interface 1) global interrupt.*

- #define **SPI2** (36)
SPI2 (Serial Peripheral Interface 2) global interrupt.
- #define **USART1** (37)
USART1 (Universal Synchronous/Asynchronous Receiver/Transmitter 1) global interrupt.
- #define **USART2** (38)
USART2 (Universal Synchronous/Asynchronous Receiver/Transmitter 2) global interrupt.
- #define **EXTI15_10** (30)
EXTI Line[15:10] interrupts.
- #define **EXTI17** (41)
EXTI Line 17 interrupt / RTC Alarms (A and B) through EXTI line interrupt.
- #define **EXTI18** (42)
EXTI Line 18 interrupt / USB On-The-Go FS Wakeup through EXTI line interrupt.
- #define **DMA1_STREAM7** (47)
DMA1 (Direct Memory Access) Stream 7 global interrupt.
- #define **SDIO** (49)
SDIO (Secure Digital Input Output) global interrupt.
- #define **TIM5** (50)
TIM5 (Timer 5) global interrupt.
- #define **SPI3** (51)
SPI3 (Serial Peripheral Interface 3) global interrupt.
- #define **DMA2_STREAM0** (56)
DMA2 (Direct Memory Access 2) Stream 0 global interrupt.
- #define **DMA2_STREAM1** (57)
DMA2 (Direct Memory Access 2) Stream 1 global interrupt.
- #define **DMA2_STREAM2** (58)
DMA2 (Direct Memory Access 2) Stream 2 global interrupt.
- #define **DMA2_STREAM3** (59)
DMA2 (Direct Memory Access 2) Stream 3 global interrupt.
- #define **DMA2_STREAM4** (60)
DMA2 (Direct Memory Access 2) Stream 4 global interrupt.
- #define **OTG_FS** (67)
USB On The Go FS global interrupt.
- #define **DMA2_STREAM5** (68)
DMA2 (Direct Memory Access 2) Stream 5 global interrupt.
- #define **DMA2_STREAM6** (69)
DMA2 (Direct Memory Access 2) Stream 6 global interrupt.
- #define **DMA2_STREAM7** (70)
DMA2 (Direct Memory Access 2) Stream 7 global interrupt.
- #define **USART6** (71)
USART6 (Universal Synchronous/Asynchronous Receiver/Transmitter 6) global interrupt.
- #define **I2C3_EV** (72)
I2C3 (Inter-integrated Circuit 3) event interrupt.
- #define **I2C3_ER** (73)
- #define **FPU** (81)
FPU (Floating Point Unit) global interrupt.
- #define **SPI4** (84)
SPI4 (Serial Peripheral Interface 4) global interrupt.

Functions

- void [MNVIC_vEnablePeriphral](#) ([VAR\(u8_t\)](#) A_u8INTID)

Enable the interrupt of a certain peripheral in NVIC.
- void [MNVIC_vDisablePeriphral](#) ([VAR\(u8_t\)](#) A_u8INTID)

Disable the interrupt of a certain peripheral in NVIC.
- void [MNVIC_vSetPendingFlag](#) ([VAR\(u8_t\)](#) A_u8INTID)

Set the pending flag of the peripheral.
- void [MNVIC_vClearPendingFlag](#) ([VAR\(u8_t\)](#) A_u8INTID)

Clear the pending flag of the peripheral.
- [u8_t MNVIC_u8GetActive](#) ([VAR\(u8_t\)](#) A_u8INTID)

Get the current state of the interrupt active flag of a certain peripheral in NVIC.
- void [MNVIC_vSetPriorityConfig](#) ([VAR\(u8_t\)](#) A_u8PriorityOption)

Configures the group and sub-group priority configuration.
- void [MNVIC_vSetPriority](#) ([VAR\(s8_t\)](#) A_s8INTID, [VAR\(u8_t\)](#) A_u8GroupPriority, [VAR\(u8_t\)](#) A_u8SubPriority)

Set the group and the sub-group priorities for the required interrupt.
- [u32_t NVIC_GetPriority](#) ([VAR\(s8_t\)](#) A_s8INTID)

Get a certain interrupt's priority.

7.40.1 Detailed Description

This file contains the interface information and addresses for the NVIC module.

Author

Ali El Bana

Version

2.0

Date

08/25/2022

Definition in file [NVIC_interface.h](#).

7.40.2 Function Documentation

7.40.2.1 MNVIC_vEnablePeriphral()

```
void MNVIC_vEnablePeriphral (
    VAR\(u8\_t\) A_u8INTID )
```

Enable the interrupt of a certain peripheral in NVIC.

Parameters

in	A_u8INTID	The peripheral's interrupt ID
----	-----------	-------------------------------

See also[MNVIC_vDisablePeriphral](#)**7.40.2.2 MNVIC_vDisablePeriphral()**

```
void MNVIC_vDisablePeriphral (
    VAR(u8_t) A_u8INTID )
```

Disable the interrupt of a certain peripheral in NVIC.

Parameters

in	A_u8INTID	The peripheral's interrupt ID
----	-----------	-------------------------------

See also[MNVIC_vEnablePeriphral](#)**7.40.2.3 MNVIC_vSetPendingFlag()**

```
void MNVIC_vSetPendingFlag (
    VAR(u8_t) A_u8INTID )
```

Set the pending flag of the peripheral.

Note

Used mostly for testing

Parameters

in	A_u8INTID	The peripheral's interrupt ID
----	-----------	-------------------------------

See also[MNVIC_vClearPendingFlag](#)

7.40.2.4 MNVIC_vClearPendingFlag()

```
void MNVIC_vClearPendingFlag (
    VAR(u8_t) A_u8INTID )
```

Clear the pending flag of the peripheral.

Note

Used mostly for testing

Parameters

in	A_u8INTID	The peripheral's interrupt ID
----	-----------	-------------------------------

See also

[MNVIC_vSetPendingFlag](#)

7.40.2.5 MNVIC_u8GetActive()

```
u8_t MNVIC_u8GetActive (
    VAR(u8_t) A_u8INTID )
```

Get the current state of the interrupt active flag of a certain peripheral in NVIC.

Parameters

in	A_u8INTID	The peripheral's interrupt ID
----	-----------	-------------------------------

Returns

The current state of the interrupt active flag

7.40.2.6 MNVIC_vSetPriorityConfig()

```
void MNVIC_vSetPriorityConfig (
    VAR(u8_t) A_u8PriorityOption )
```

Configures the group and sub-group priority configuration.

Parameters

in	A_u8PriorityOption	The peripheral's interrupt ID
----	--------------------	-------------------------------

7.40.2.7 MNVIC_vSetPriority()

```
void MNVIC_vSetPriority (
    VAR(s8_t) A_s8INTID,
    VAR(u8_t) A_u8GroupPriority,
    VAR(u8_t) A_u8SubPriority )
```

Set the group and the sub-group priorities for the required interrupt.

Parameters

in	<i>A_s8INTID</i>	The port that the pin belongs to
in	<i>A_u8GroupPriority</i>	The pin to update its mode
in	<i>A_u8SubPriority</i>	The alternative function to apply on the pin

See also

[Group priorities](#)

[Sub-Group priorities](#)

7.40.2.8 NVIC_GetPriority()

```
u32_t NVIC_GetPriority (
    VAR(s8_t) A_s8INTID )
```

Get a certain interrupt's priority.

Parameters

in	<i>A_s8INTID</i>	
----	------------------	--

Returns

The priority for the corresponding interrupt

7.41 NVIC_interface.h

[Go to the documentation of this file.](#)

```
00001
00009 /* Header file guard */
00010 #ifndef _NVIC_interface_H
00011 #define _NVIC_interface_H
00012
00013 /***** */
00014 /*          Functions prototypes          */
00015 /***** */
```

```

00016
00022 void MNVIC_vEnablePeripheral(VAR(u8_t) A_u8INTID);
00023
00029 void MNVIC_vDisablePeripheral(VAR(u8_t) A_u8INTID);
00030
00037 void MNVIC_vSetPendingFlag(VAR(u8_t) A_u8INTID);
00038
00045 void MNVIC_vClearPendingFlag(VAR(u8_t) A_u8INTID);
00046
00052 u8_t MNVIC_u8GetActive(VAR(u8_t) A_u8INTID);
00053
00058 void MNVIC_vSetPriorityConfig(VAR(u8_t) A_u8PriorityOption);
00059
00068 void MNVIC_vSetPriority(VAR(s8_t) A_s8INTID, VAR(u8_t) A_u8GroupPriority, VAR(u8_t) A_u8SubPriority);
00069
00075 u32_t NVIC_GetPriority(VAR(s8_t) A_s8INTID);
00076
00077 /***** Interfacing macros *****/
00078 /* ****
00079 ****/
00080
00097 #define _16GROUP_NoSub_Priorities (0b011)
00105 #define _8GROUP_2Sub_Priorities (0b100)
00113 #define _4GROUP_4Sub_Priorities (0b101)
00121 #define _2GROUP_8Sub_Priorities (0b110)
00129 #define NoGROUP_16Sub_Priorities (0b111)
00147 #define GROUP_4BITS (0b011)
00154 #define GROUP_3BITS (0b100)
00161 #define GROUP_2BITS (0b101)
00168 #define GROUP_1BITS (0b110)
00175 #define GROUP_0BITS (0b111)
00189 #define NO_GROUP_PRIORITY (0)
00194 #define GROUP_PRIORITY_0 (0)
00199 #define GROUP_PRIORITY_1 (1)
00204 #define GROUP_PRIORITY_2 (2)
00209 #define GROUP_PRIORITY_3 (3)
00214 #define GROUP_PRIORITY_4 (4)
00219 #define GROUP_PRIORITY_5 (5)
00224 #define GROUP_PRIORITY_6 (6)
00229 #define GROUP_PRIORITY_7 (7)
00234 #define GROUP_PRIORITY_8 (8)
00239 #define GROUP_PRIORITY_9 (9)
00244 #define GROUP_PRIORITY_10 (10)
00249 #define GROUP_PRIORITY_11 (11)
00254 #define GROUP_PRIORITY_12 (12)
00259 #define GROUP_PRIORITY_13 (13)
00264 #define GROUP_PRIORITY_14 (14)
00269 #define GROUP_PRIORITY_15 (15)
00285 #define NO_SUB_PRIORITY (0)
00290 #define SUB_PRIORITY_0 (0)
00295 #define SUB_PRIORITY_1 (1)
00300 #define SUB_PRIORITY_2 (2)
00305 #define SUB_PRIORITY_3 (3)
00310 #define SUB_PRIORITY_4 (4)
00315 #define SUB_PRIORITY_5 (5)
00320 #define SUB_PRIORITY_6 (6)
00325 #define SUB_PRIORITY_7 (7)
00330 #define SUB_PRIORITY_8 (8)
00335 #define SUB_PRIORITY_9 (9)
00340 #define SUB_PRIORITY_10 (10)
00345 #define SUB_PRIORITY_11 (11)
00350 #define SUB_PRIORITY_12 (12)
00355 #define SUB_PRIORITY_13 (13)
00360 #define SUB_PRIORITY_14 (14)
00365 #define SUB_PRIORITY_15 (15)
00379 #define WWDG (0)
00384 #define EXTI16 (1)
00389 #define EXTI21 (2)
00394 #define EXTI22 (3)
00399 #define FLASH (4)
00404 #define RCC (5)
00409 #define EXTI0 (6)
00414 #define EXTI1 (7)
00419 #define EXTI2 (8)
00424 #define EXTI3 (9)
00429 #define EXTI4 (10)
00434 #define DMA1_STREAM0 (11)
00439 #define DMA1_STREAM1 (12)
00444 #define DMA1_STREAM2 (13)
00449 #define DMA1_STREAM3 (14)
00454 #define DMA1_STREAM4 (15)
00459 #define DMA1_STREAM5 (16)
00464 #define DMA1_STREAM6 (17)
00469 #define ADC (18)
00474 #define EXTI9 (23)
00479 #define TIM1_BRK_TIM9 (24)
00484 #define TIM1_UP_TIM10 (25)

```

```

00489 #define TIM1_TRG_COM_TIM11 (26)
00494 #define TIM1_CC (27)
00499 #define TIM2 (28)
00504 #define TIM3 (29)
00509 #define TIM4 (30)
00514 #define I2C1_EV (31)
00519 #define I2C1_ER (32)
00524 #define I2C2_EV (33)
00529 #define I2C2_ER (34)
00534 #define SPI1 (35)
00539 #define SPI2 (36)
00544 #define USART1 (37)
00549 #define USART2 (38)
00554 #define EXTI15_10 (30)
00559 #define EXTI17 (41)
00564 #define EXTI18 (42)
00569 #define DMA1_STREAM7 (47)
00574 #define SDIO (49)
00579 #define TIM5 (50)
00584 #define SPI3 (51)
00589 #define DMA2_STREAM0 (56)
00594 #define DMA2_STREAM1 (57)
00599 #define DMA2_STREAM2 (58)
00604 #define DMA2_STREAM3 (59)
00609 #define DMA2_STREAM4 (60)
00614 #define OTG_FS (67)
00619 #define DMA2_STREAM5 (68)
00624 #define DMA2_STREAM6 (69)
00629 #define DMA2_STREAM7 (70)
00634 #define USART6 (71)
00639 #define I2C3_EV (72)
00644 #define I2C3_ER (73)
00649 #define FPU (81)
00654 #define SPI4 (84)
00657 #endif // _NVIC_interface_H

```

7.42 COTS/MCAL/NVIC/NVIC_private.h File Reference

This file contains the registers information and addresses for the NVIC module.

Data Structures

- struct **NVIC_MemoryMapType**
NVIC configuration structure for NVIC memory map.
- struct **SCB_MemoryMapType**
System control block memory map structure.

Macros

- #define **NVIC_BASE_ADDRESS** (0xE000E100)
NVIC base address.
- #define **SCB_BASE_ADDRESS** (0xE000ED00)
SCB base address.
- #define **MSCB** ((volatile P2VAR(SCB_MemoryMapType))(SCB_BASE_ADDRESS))
- #define **MNVIC** ((volatile P2VAR(NVIC_MemoryMapType))(NVIC_BASE_ADDRESS))
- #define **MNVIC_STIR** *((volatile 32 *)0xE000EF00))
- #define **VECTKEY_PASSWORD** (0x05FA0000)
VECTKEY Password.
- #define **PEND_SV** (-6)
- #define **SYSTICK** (-5)
- #define **SV_CALL** (-4)
- #define **MEMORY_MANAGE** (-3)
- #define **BUS_FAULT** (-2)
- #define **USAGE_FAULT** (-1)

7.42.1 Detailed Description

This file contains the registers information and addresses for the NVIC module.

Author

Ali El Bana

Version

2.0

Date

08/22/2022

Definition in file [NVIC_private.h](#).

7.42.2 Macro Definition Documentation

7.42.2.1 MNVIC_STIR

```
#define MNVIC_STIR *((volatile 32 *) (0xE000EF00))
```

Definition at line [243](#) of file [NVIC_private.h](#).

7.42.2.2 VECTKEY_PASSWORD

```
#define VECTKEY_PASSWORD (0x05FA0000)
```

VECTKEY Password.

This is the VECTKEY field password that must provides on every write attempt on the AIRCR register

See also

[SCB_MemoryMapType::AIRCR](#)

Definition at line [252](#) of file [NVIC_private.h](#).

7.43 NVIC_private.h

[Go to the documentation of this file.](#)

```

00001 /* Header file guard */
00010 #ifndef _NVIC_private_H
00011 #define _NVIC_private_H
00012
00018 typedef struct
00019 {
00027     u32_t ISERx[8];
00031     u32_t RESERVED0x[24];
00039     u32_t ICERx[8];
00043     u32_t RSERVED1x[24];
00053     u32_t ISPRx[8];
00057     u32_t RESERVED2x[24];
00067     u32_t ICPRx[8];
00071     u32_t RESERVED3x[24];
00077     u32_t IABRx[8];
00081     u32_t RESERVED4x[56];
00089     u8_t IPRx[240];
00090
00091 } NVIC_MemoryMapType;
00092
00100 typedef struct
00101 {
00106     u32_t CPUID;
00111     u32_t ICSR;
00116     u32_t VTOR;
00123     u32_t AIRCR;
00128     u32_t SCR;
00136     u32_t CCR;
00143     u32_t SHPR1;
00150     u32_t SHPR2;
00157     u32_t SHPR3;
00165     u32_t SHCSR;
00176     u32_t CFSR;
00183     u32_t HFSR;
00187     u32_t RESERVED;
00192     u32_t MMFAR;
00197     u32_t BFAR;
00198 } SCB_MemoryMapType;
00199
00212 #define NVIC_BASE_ADDRESS (0xE000E100)
00213
00219 #define SCB_BASE_ADDRESS (0xE000ED00)
00234 #define MSCB ((volatile P2VAR(SCB_MemoryMapType)) (SCB_BASE_ADDRESS))
00240 #define MNVIC ((volatile P2VAR(NVIC_MemoryMapType)) (NVIC_BASE_ADDRESS))
00243 #define MNVIC_STIR *((volatile 32 *) (0xE000EF00))
00244
00252 #define VECTKEY_PASSWORD (0x05FA0000)
00253
00266 #define PEND_SV (-6)
00272 #define SYSTICK (-5)
00278 #define SV_CALL (-4)
00284 #define MEMORY_MANAGE (-3)
00290 #define BUSFAULT (-2)
00296 #define USAGE_FAULT (-1)
00299 #endif // _NVIC_private_H

```

7.44 COTS/MCAL/NVIC/NVIC_program.c File Reference

This file contains the source code of the interfacing for the NVIC modules.

```

#include "../../LIB/LSTD_TYPES.h"
#include "../../LIB/LSTD_COMPILER.h"
#include "../../LIB/LSTD_VALUES.h"
#include "../../LIB/LSTD_BITMATH.h"
#include "NVIC_interface.h"
#include "NVIC_private.h"
#include "NVIC_config.h"

```

Macros

- #define REGISTER_SIZE (32)

Functions

- void MNVIC_vEnablePeripheral (u8_t A_u8INTID)
- void MNVIC_vDisablePeripheral (u8_t A_u8INTID)
- void MNVIC_vSetPendingFlag (u8_t A_u8INTID)
- void MNVIC_vClearPendingFlag (u8_t A_u8INTID)
- u8_t MNVIC_u8GetActive (u8_t A_u8INTID)
- void MNVIC_vSetPriorityConfig (u8_t A_u8PriorityOption)
- void MNVIC_vSetPriority (s8_t A_s8INTID, u8_t A_u8GroupPriority, u8_t A_u8SubPriority)
- u32_t NVIC_GetPriority (s8_t A_s8INTID)

7.44.1 Detailed Description

This file contains the source code of the interfacing for the NVIC modules.

Author

Ali El Bana

Version

2.0

Date

08/25/2022

Definition in file [NVIC_program.c](#).

7.44.2 Macro Definition Documentation

7.44.2.1 REGISTER_SIZE

```
#define REGISTER_SIZE (32)
```

Definition at line [27](#) of file [NVIC_program.c](#).

7.44.3 Function Documentation

7.44.3.1 MNVIC_vEnablePeriphral()

```
void MNVIC_vEnablePeriphral (
    u8_t A_u8INTID )
```

Definition at line 33 of file [NVIC_program.c](#).

```
00034 {
00035     MNVIC->ISERx[A_u8INTID / REGISTER_SIZE] = (1 << (A_u8INTID % REGISTER_SIZE));
00036 }
```

References [MNVIC](#), and [REGISTER_SIZE](#).

7.44.3.2 MNVIC_vDisablePeriphral()

```
void MNVIC_vDisablePeriphral (
    u8_t A_u8INTID )
```

Definition at line 41 of file [NVIC_program.c](#).

```
00042 {
00043     MNVIC->ICERx[A_u8INTID / REGISTER_SIZE] = (1 << (A_u8INTID % REGISTER_SIZE));
00044 }
```

References [MNVIC](#), and [REGISTER_SIZE](#).

7.44.3.3 MNVIC_vSetPendingFlag()

```
void MNVIC_vSetPendingFlag (
    u8_t A_u8INTID )
```

Definition at line 49 of file [NVIC_program.c](#).

```
00050 {
00051     MNVIC->ISPRx[A_u8INTID / REGISTER_SIZE] = (1 << (A_u8INTID % REGISTER_SIZE));
00052 }
```

References [MNVIC](#), and [REGISTER_SIZE](#).

7.44.3.4 MNVIC_vClearPendingFlag()

```
void MNVIC_vClearPendingFlag (
    u8_t A_u8INTID )
```

Definition at line 57 of file [NVIC_program.c](#).

```
00058 {
00059     MNVIC->ICPRx[A_u8INTID / REGISTER_SIZE] = (1 << (A_u8INTID % REGISTER_SIZE));
00060 }
```

References [MNVIC](#), and [REGISTER_SIZE](#).

7.44.3.5 MNVIC_u8GetActive()

```
u8_t MNVIC_u8GetActive (
    u8_t A_u8INTID )
```

Definition at line 65 of file [NVIC_program.c](#).

```
00066 {
00067     VAR(u8_t) L_u8Active = INITIAL_ZERO;
00068     L_u8Active = GET_BIT((MNVIC->IABRx[A_u8INTID / REGISTER_SIZE]), (A_u8INTID % REGISTER_SIZE));
00069     return L_u8Active;
00070 }
```

References [GET_BIT](#), [INITIAL_ZERO](#), [MNVIC](#), [REGISTER_SIZE](#), and [VAR](#).

7.44.3.6 MNVIC_vSetPriorityConfig()

```
void MNVIC_vSetPriorityConfig (
    u8_t A_u8PriorityOption )
```

Definition at line 75 of file [NVIC_program.c](#).

```
00076 {
00077     GS_u32GroupConf = (VECTKEY_PASSWORD | (A_u8PriorityOption << 8));
00078     MSCB->AIRCR = GS_u32GroupConf;
00079 }
```

References [MSCB](#), and [VECTKEY_PASSWORD](#).

7.44.3.7 MNVIC_vSetPriority()

```
void MNVIC_vSetPriority (
    s8_t A_s8INTID,
    u8_t A_u8GroupPriority,
    u8_t A_u8SubPriority )
```

Definition at line 84 of file [NVIC_program.c](#).

```
00085 {
00086     VAR(u8_t) L_u8Priority = INITIAL_ZERO;
00087     L_u8Priority = A_u8SubPriority | (A_u8GroupPriority << ((GS_u32GroupConf - 0x05FA0300) / 0x100));
00088
00089 // Core Peripheral
00090 if (A_s8INTID < 0)
00091 {
00092     if ((A_s8INTID == MEMORY_MANAGE) || (A_s8INTID == BUS_FAULT) || (A_s8INTID == USAGE_FAULT))
00093     {
00094         A_s8INTID += 3;
00095         MSCB->SHPR1 = (L_u8Priority) << ((8 * A_s8INTID) + 4);
00096     }
00097     else if (A_s8INTID == SV_CALL)
00098     {
00099         A_s8INTID += 7;
00100         MSCB->SHPR2 = (L_u8Priority) << ((8 * A_s8INTID) + 4);
00101     }
00102     else if (A_s8INTID == PEND_SV || A_s8INTID == SYSTICK)
00103     {
00104         A_s8INTID += 8;
00105         MSCB->SHPR3 = (L_u8Priority) << ((8 * A_s8INTID) + 4);
00106     }
00107     else
00108     {
00109         /* Do nothing */
00110     }
00111 }
00112 // External Peripheral
```

```

00113     else if (A_s8INTID >= 0)
00114     {
00115         MNVIC->IPRx[A_s8INTID] = (L_u8Priority << 4);
00116     }
00117     else
00118     {
00119         /* Do nothing */
00120     }
00121 }
```

References [BUS_FAULT](#), [INITIAL_ZERO](#), [MEMORY_MANAGE](#), [MNVIC](#), [MSCB](#), [PEND_SV](#), [SV_CALL](#), [SYSTICK](#), [USAGE_FAULT](#), and [VAR](#).

7.44.3.8 NVIC_GetPriority()

```

u32_t NVIC_GetPriority (
    s8_t A_s8INTID )
```

Definition at line 126 of file [NVIC_program.c](#).

```

00127 {
00128     VAR(u32_t) L_u32ThePriorityIS = INITIAL_ZERO;
00129
00130     // Core Peripheral
00131     if (A_s8INTID < 0)
00132     {
00133         if ((A_s8INTID == MEMORY_MANAGE) || (A_s8INTID == BUS_FAULT) || (A_s8INTID == USAGE_FAULT))
00134         {
00135             A_s8INTID += 3;
00136             L_u32ThePriorityIS = MSCB->SHPR1 >> ((8 * A_s8INTID) + 4);
00137         }
00138         else if (A_s8INTID == SV_CALL)
00139         {
00140             A_s8INTID += 7;
00141             L_u32ThePriorityIS = MSCB->SHPR2 >> ((8 * A_s8INTID) + 4);
00142         }
00143         else if (A_s8INTID == PEND_SV || A_s8INTID == SYSTICK)
00144         {
00145             A_s8INTID += 8;
00146             L_u32ThePriorityIS = MSCB->SHPR3 >> ((8 * A_s8INTID) + 4);
00147         }
00148         else
00149         {
00150             /* Do nothing */
00151         }
00152     }
00153     // External Peripheral
00154     else if (A_s8INTID >= 0)
00155     {
00156         L_u32ThePriorityIS = (MNVIC->IPRx[A_s8INTID] >> 4);
00157     }
00158     else
00159     {
00160         /* Do nothing */
00161     }
00162
00163     return L_u32ThePriorityIS;
00164 }
```

References [BUS_FAULT](#), [INITIAL_ZERO](#), [MEMORY_MANAGE](#), [MNVIC](#), [MSCB](#), [PEND_SV](#), [SV_CALL](#), [SYSTICK](#), [USAGE_FAULT](#), and [VAR](#).

7.45 NVIC_program.c

[Go to the documentation of this file.](#)

```

00001
00009 /***** Include headers *****/
0010 /*                                         */
0011 /*****                                         */
0012 */
```

```

00013 #include "../../LIB/LSTD_TYPES.h"
00014 #include "../../LIB/LSTD_COMPILER.h"
00015 #include "../../LIB/LSTD_VALUES.h"
00016 #include "../../LIB/LSTD_BITMATH.h"
00017 #include "NVIC_interface.h"
00018 #include "NVIC_private.h"
00019 #include "NVIC_config.h"
00020
00021 /***** Global functions ****/
00022 /*          Functions' implementations          */
00023 /***** */
00024
00025 STATIC VAR(u32_t) GS_u32GroupConf = INITIAL_ZERO;
00026
00027 #define REGISTER_SIZE (32)
00028
00029 /***** */
00030 /*          Functions' implementations          */
00031 /***** */
00032
00033 FUNC(void) MNVIC_vEnablePeriphral(VAR(u8_t) A_u8INTID)
00034 {
00035     MNVIC->ISERx[A_u8INTID / REGISTER_SIZE] = (1 << (A_u8INTID % REGISTER_SIZE));
00036 }
00037
00038
00039 /***** */
00040
00041 FUNC(void) MNVIC_vDisablePeriphral(VAR(u8_t) A_u8INTID)
00042 {
00043     MNVIC->ICERx[A_u8INTID / REGISTER_SIZE] = (1 << (A_u8INTID % REGISTER_SIZE));
00044 }
00045
00046
00047
00048
00049 FUNC(void) MNVIC_vSetPendingFlag(VAR(u8_t) A_u8INTID)
00050 {
00051     MNVIC->ISPRx[A_u8INTID / REGISTER_SIZE] = (1 << (A_u8INTID % REGISTER_SIZE));
00052 }
00053
00054
00055
00056
00057 FUNC(void) MNVIC_vClearPendingFlag(VAR(u8_t) A_u8INTID)
00058 {
00059     MNVIC->ICPRx[A_u8INTID / REGISTER_SIZE] = (1 << (A_u8INTID % REGISTER_SIZE));
00060 }
00061
00062
00063
00064
00065 FUNC(u8_t) MNVIC_u8GetActive(VAR(u8_t) A_u8INTID)
00066 {
00067     VAR(u8_t) L_u8Active = INITIAL_ZERO;
00068     L_u8Active = GET_BIT((MNVIC->IABRx[A_u8INTID / REGISTER_SIZE]), (A_u8INTID % REGISTER_SIZE));
00069     return L_u8Active;
00070 }
00071
00072
00073
00074
00075 FUNC(void) MNVIC_vSetPriorityConfig(VAR(u8_t) A_u8PriorityOption)
00076 {
00077     GS_u32GroupConf = (VECTKEY_PASSWORD | (A_u8PriorityOption << 8));
00078     MSCB->AIRCR = GS_u32GroupConf;
00079 }
00080
00081
00082
00083
00084 FUNC(void) MNVIC_vSetPriority(VAR(s8_t) A_s8INTID, VAR(u8_t) A_u8GroupPriority, VAR(u8_t)
00085     A_u8SubPriority)
00086 {
00087     VAR(u8_t) L_u8Priority = INITIAL_ZERO;

```

```

00087     L_u8Priority = A_u8SubPriority | (A_u8GroupPriority << ((GS_u32GroupConf - 0x05FA0300) / 0x100));
00088
00089     // Core Peripheral
00090     if (A_s8INTID < 0)
00091     {
00092         if ((A_s8INTID == MEMORY_MANAGE) || (A_s8INTID == BUS_FAULT) || (A_s8INTID == USAGE_FAULT))
00093         {
00094             A_s8INTID += 3;
00095             MSCB->SHPR1 = (L_u8Priority) << ((8 * A_s8INTID) + 4);
00096         }
00097         else if (A_s8INTID == SV_CALL)
00098         {
00099             A_s8INTID += 7;
00100             MSCB->SHPR2 = (L_u8Priority) << ((8 * A_s8INTID) + 4);
00101         }
00102         else if (A_s8INTID == PEND_SV || A_s8INTID == SYSTICK)
00103         {
00104             A_s8INTID += 8;
00105             MSCB->SHPR3 = (L_u8Priority) << ((8 * A_s8INTID) + 4);
00106         }
00107         else
00108         {
00109             /* Do nothing */
00110         }
00111     }
00112     // External Peripheral
00113     else if (A_s8INTID >= 0)
00114     {
00115         MNVIC->IPRx[A_s8INTID] = (L_u8Priority << 4);
00116     }
00117     else
00118     {
00119         /* Do nothing */
00120     }
00121 }
00122
00123 /*****
00124 ****/
00125
00126 VAR(u32_t) NVIC_GetPriority(VAR(s8_t) A_s8INTID)
00127 {
00128     VAR(u32_t) L_u32ThePriorityIS = INITIAL_ZERO;
00129
00130     // Core Peripheral
00131     if (A_s8INTID < 0)
00132     {
00133         if ((A_s8INTID == MEMORY_MANAGE) || (A_s8INTID == BUS_FAULT) || (A_s8INTID == USAGE_FAULT))
00134         {
00135             A_s8INTID += 3;
00136             L_u32ThePriorityIS = MSCB->SHPR1 >> ((8 * A_s8INTID) + 4);
00137         }
00138         else if (A_s8INTID == SV_CALL)
00139         {
00140             A_s8INTID += 7;
00141             L_u32ThePriorityIS = MSCB->SHPR2 >> ((8 * A_s8INTID) + 4);
00142         }
00143         else if (A_s8INTID == PEND_SV || A_s8INTID == SYSTICK)
00144         {
00145             A_s8INTID += 8;
00146             L_u32ThePriorityIS = MSCB->SHPR3 >> ((8 * A_s8INTID) + 4);
00147         }
00148         else
00149         {
00150             /* Do nothing */
00151         }
00152     }
00153     // External Peripheral
00154     else if (A_s8INTID >= 0)
00155     {
00156         L_u32ThePriorityIS = (MNVIC->IPRx[A_s8INTID] >> 4);
00157     }
00158     else
00159     {
00160         /* Do nothing */
00161     }
00162
00163     return L_u32ThePriorityIS;
00164 }
00165
00166 /*****
00167 ****/

```

7.46 COTS/MCAL/RCC/MRCC_config.h File Reference

This file contains the RCC configurations.

Macros

- #define PLLI2S DISABLE
- #define PLL DISABLE
- #define CSS DISABLE
- #define HSEBYP NOTBYBASED
- #define HSE_EN DISABLE
- #define HSI_EN ENABLE
- #define PLLQ Equal_2
- #define PLLSRC HSE
- #define PLLP PLLCLK
- #define PLLN Equal_200
- #define PLLM Equal_2
- #define MCO2 SYSCLK
- #define MCO2PRE NoDivision
- #define MCO1PRE NoDivision
- #define I2SSRC PLLI2SCLK
- #define MCO1 HSE
- #define RTCPRE HSEby2
- #define PPRE2 NoDivision
- #define PPRE1 NoDivision
- #define HPRE SYSCLKby2
- #define SWS HSI
- #define SW HSI
- #define DMA2CLK DISABLE
- #define DMA1CLK DISABLE
- #define CRCCLK DISABLE
- #define GPIOHCLK DISABLE
- #define GPIOECLK DISABLE
- #define GPIODCLK DISABLE
- #define GPIOCCLK DISABLE
- #define GPIOBCLK DISABLE
- #define GPIOACLK DISABLE
- #define OTGFS DISABLE
- #define PWREN DISABLE
- #define I2C3EN DISABLE
- #define I2C2EN DISABLE
- #define I2C1EN DISABLE
- #define USART2EN DISABLE
- #define SPI3EN DISABLE
- #define SPI2EN DISABLE
- #define WWDGEN DISABLE
- #define TIM5EN DISABLE
- #define TIM4EN DISABLE
- #define TIM3EN DISABLE
- #define TIM2EN DISABLE
- #define TIM11EN DISABLE
- #define TIM10EN DISABLE
- #define TIM9EN DISABLE

- #define SYSCFGEN DISABLE
- #define SPI4EN DISABLE
- #define SPI1EN DISABLE
- #define SDIOEN DISABLE
- #define ADC1EN DISABLE
- #define USART6EN DISABLE
- #define USART1EN DISABLE
- #define TIM1EN DISABLE

7.46.1 Detailed Description

This file contains the RCC configurations.

Author

Ali El Bana & Mo Alaa

Version

2.0

Date

11/9/2022

Definition in file [MRCC_config.h](#).

7.46.2 Macro Definition Documentation

7.46.2.1 PLLI2S

```
#define PLLI2S DISABLE
```

Definition at line 21 of file [MRCC_config.h](#).

7.46.2.2 PLL

```
#define PLL DISABLE
```

Definition at line 29 of file [MRCC_config.h](#).

7.46.2.3 CSS

```
#define CSS DISABLE
```

Definition at line 37 of file [MRCC_config.h](#).

7.46.2.4 HSEBYP

```
#define HSEBYP NOTBYBASED
```

Definition at line 45 of file [MRCC_config.h](#).

7.46.2.5 HSE_EN

```
#define HSE_EN DISABLE
```

Definition at line 53 of file [MRCC_config.h](#).

7.46.2.6 HSI_EN

```
#define HSI_EN ENABLE
```

Definition at line 61 of file [MRCC_config.h](#).

7.46.2.7 PLLQ

```
#define PLLQ Equal_2
```

Definition at line 84 of file [MRCC_config.h](#).

7.46.2.8 PLLSRC

```
#define PLLSRC HSE
```

Definition at line 92 of file [MRCC_config.h](#).

7.46.2.9 PLLP

```
#define PLLP PLLCLK
```

Definition at line 102 of file [MRCC_config.h](#).

7.46.2.10 PLLN

```
#define PLLN Equal_200
```

Definition at line 173 of file [MRCC_config.h](#).

7.46.2.11 PLLM

```
#define PLLM Equal_2
```

Definition at line 240 of file [MRCC_config.h](#).

7.46.2.12 MCO2

```
#define MCO2 SYSCLK
```

Definition at line 253 of file [MRCC_config.h](#).

7.46.2.13 MCO2PRE

```
#define MCO2PRE NoDivision
```

Definition at line 264 of file [MRCC_config.h](#).

7.46.2.14 MCO1PRE

```
#define MCO1PRE NoDivision
```

Definition at line 275 of file [MRCC_config.h](#).

7.46.2.15 I2SSRC

```
#define I2SSRC PLLI2SCLK
```

Definition at line 283 of file [MRCC_config.h](#).

7.46.2.16 MCO1

```
#define MCO1 HSE
```

Definition at line 293 of file [MRCC_config.h](#).

7.46.2.17 RTCPRE

```
#define RTCPRE HSEby2
```

Definition at line 332 of file [MRCC_config.h](#).

7.46.2.18 PPRE2

```
#define PPRE2 NoDivision
```

Definition at line 343 of file [MRCC_config.h](#).

7.46.2.19 PPRE1

```
#define PPRE1 NoDivision
```

Definition at line 354 of file [MRCC_config.h](#).

7.46.2.20 HPRE

```
#define HPRE SYSCLKby2
```

Definition at line 369 of file [MRCC_config.h](#).

7.46.2.21 SWS

```
#define SWS HSI
```

Definition at line 378 of file [MRCC_config.h](#).

7.46.2.22 SW

```
#define SW HSI
```

Definition at line 387 of file [MRCC_config.h](#).

7.46.2.23 DMA2CLK

```
#define DMA2CLK DISABLE
```

Definition at line 398 of file [MRCC_config.h](#).

7.46.2.24 DMA1CLK

```
#define DMA1CLK DISABLE
```

Definition at line 406 of file [MRCC_config.h](#).

7.46.2.25 CRCCLK

```
#define CRCCLK DISABLE
```

Definition at line 414 of file [MRCC_config.h](#).

7.46.2.26 GPIOHCLK

```
#define GPIOHCLK DISABLE
```

Definition at line 422 of file [MRCC_config.h](#).

7.46.2.27 GPIOECLK

```
#define GPIOECLK DISABLE
```

Definition at line [430](#) of file [MRCC_config.h](#).

7.46.2.28 GPIODCLK

```
#define GPIODCLK DISABLE
```

Definition at line [438](#) of file [MRCC_config.h](#).

7.46.2.29 GPIOCCLK

```
#define GPIOCCLK DISABLE
```

Definition at line [446](#) of file [MRCC_config.h](#).

7.46.2.30 GPIOBCLK

```
#define GPIOBCLK DISABLE
```

Definition at line [454](#) of file [MRCC_config.h](#).

7.46.2.31 GPIOACLK

```
#define GPIOACLK DISABLE
```

Definition at line [462](#) of file [MRCC_config.h](#).

7.46.2.32 OTGFS

```
#define OTGFS DISABLE
```

Definition at line [473](#) of file [MRCC_config.h](#).

7.46.2.33 PWREN

```
#define PWREN DISABLE
```

Definition at line [484](#) of file [MRCC_config.h](#).

7.46.2.34 I2C3EN

```
#define I2C3EN DISABLE
```

Definition at line [492](#) of file [MRCC_config.h](#).

7.46.2.35 I2C2EN

```
#define I2C2EN DISABLE
```

Definition at line [500](#) of file [MRCC_config.h](#).

7.46.2.36 I2C1EN

```
#define I2C1EN DISABLE
```

Definition at line [508](#) of file [MRCC_config.h](#).

7.46.2.37 USART2EN

```
#define USART2EN DISABLE
```

Definition at line [516](#) of file [MRCC_config.h](#).

7.46.2.38 SPI3EN

```
#define SPI3EN DISABLE
```

Definition at line [524](#) of file [MRCC_config.h](#).

7.46.2.39 SPI2EN

```
#define SPI2EN DISABLE
```

Definition at line 532 of file [MRCC_config.h](#).

7.46.2.40 WWDGEN

```
#define WWDGEN DISABLE
```

Definition at line 540 of file [MRCC_config.h](#).

7.46.2.41 TIM5EN

```
#define TIM5EN DISABLE
```

Definition at line 548 of file [MRCC_config.h](#).

7.46.2.42 TIM4EN

```
#define TIM4EN DISABLE
```

Definition at line 556 of file [MRCC_config.h](#).

7.46.2.43 TIM3EN

```
#define TIM3EN DISABLE
```

Definition at line 564 of file [MRCC_config.h](#).

7.46.2.44 TIM2EN

```
#define TIM2EN DISABLE
```

Definition at line 572 of file [MRCC_config.h](#).

7.46.2.45 TIM11EN

```
#define TIM11EN DISABLE
```

Definition at line 583 of file [MRCC_config.h](#).

7.46.2.46 TIM10EN

```
#define TIM10EN DISABLE
```

Definition at line 591 of file [MRCC_config.h](#).

7.46.2.47 TIM9EN

```
#define TIM9EN DISABLE
```

Definition at line 599 of file [MRCC_config.h](#).

7.46.2.48 SYSCFGGEN

```
#define SYSCFGGEN DISABLE
```

Definition at line 607 of file [MRCC_config.h](#).

7.46.2.49 SPI4EN

```
#define SPI4EN DISABLE
```

Definition at line 615 of file [MRCC_config.h](#).

7.46.2.50 SPI1EN

```
#define SPI1EN DISABLE
```

Definition at line 623 of file [MRCC_config.h](#).

7.46.2.51 SDIOEN

```
#define SDIOEN DISABLE
```

Definition at line [631](#) of file [MRCC_config.h](#).

7.46.2.52 ADC1EN

```
#define ADC1EN DISABLE
```

Definition at line [639](#) of file [MRCC_config.h](#).

7.46.2.53 USART6EN

```
#define USART6EN DISABLE
```

Definition at line [647](#) of file [MRCC_config.h](#).

7.46.2.54 USART1EN

```
#define USART1EN DISABLE
```

Definition at line [655](#) of file [MRCC_config.h](#).

7.46.2.55 TIM1EN

```
#define TIM1EN DISABLE
```

Definition at line [663](#) of file [MRCC_config.h](#).

7.47 MRCC_config.h

[Go to the documentation of this file.](#)

```
00001
00009 #ifndef MCAL_RCC_MRCC_CONFIG_H_
0010 #define MCAL_RCC_MRCC_CONFIG_H_
0011
0012
0013 /***** RCC_CR configurations *****/
0014 /*          RCC_CR configurations          */
0015 /***** RCC_CR configurations *****/
0016
0017 /*options:
0018 *ENABLE
0019 *DISABLE
0020 */
0021 #define PLLI2S DISABLE
0022
0023 /***** CSS configurations *****/
0024
0025 /*options:
0026 *ENABLE
0027 *DISABLE
0028 */
0029 #define PLL DISABLE
0030
0031 /***** HSEBYP configurations *****/
0032
0033 /*options:
0034 *ENABLE
0035 *DISABLE
0036 */
0037 #define CSS DISABLE
0038
0039 /***** HSE_EN configurations *****/
0040
0041 /*options:
0042 *BYBASED
0043 *NOTBYBASED
0044 */
0045 #define HSEBYP NOTBYBASED
0046
0047 /***** HSE_EN configurations *****/
0048
0049 /*options:
0050 *ENABLE
0051 *DISABLE
0052 */
0053 #define HSE_EN DISABLE
0054
0055 /***** HSI_EN configurations *****/
0056
0057 /*options:
0058 *ENABLE
0059 *DISABLE
0060 */
0061 #define HSI_EN ENABLE
0062
0063
0064 /***** RCC_PLLCFGR configurations ***/
0065 // RCC_PLLCFGR configurations //
0066 /***** RCC_PLLCFGR configurations ***/
0067
0068 /*options:
0069 *Equal_2
0070 *Equal_3
0071 *Equal_4
0072 *Equal_5
0073 *Equal_6
0074 *Equal_7
0075 *Equal_8
0076 *Equal_9
0077 *Equal_10
0078 *Equal_11
0079 *Equal_12
0080 *Equal_13
0081 *Equal_14
0082 *Equal_15
0083 */
0084 #define PLLQ Equal_2
0085
0086 /***** HSI configurations *****/
0087
0088 /*options:
0089 *HSI
```

```
00090 *HSE
00091 */
00092 #define PLLSRC HSE
00093
00094 /***** */
00095
00096 /*options:
00097 *Equal_2
00098 *Equal_4
00099 *Equal_6
00100 *Equal_8
00101 */
00102 #define PLLP PLLCLK
00103
00104 /***** */
00105
00106 /*options:
00107 *Equal_192
00108 *Equal_193
00109 *Equal_194
00110 *Equal_195
00111 *Equal_196
00112 *Equal_197
00113 *Equal_198
00114 *Equal_199
00115 *Equal_200
00116 *Equal_201
00117 *Equal_202
00118 *Equal_203
00119 *Equal_204
00120 *Equal_205
00121 *Equal_206
00122 *Equal_207
00123 *Equal_208
00124 *Equal_209
00125 *Equal_210
00126
00127 *Equal_300
00128 *Equal_301
00129 *Equal_302
00130 *Equal_303
00131 *Equal_304
00132 *Equal_305
00133 *Equal_306
00134 *Equal_307
00135 *Equal_308
00136 *Equal_309
00137 *Equal_310
00138
00139 *Equal_400
00140 *Equal_401
00141 *Equal_402
00142 *Equal_403
00143 *Equal_404
00144 *Equal_405
00145 *Equal_406
00146 *Equal_407
00147 *Equal_408
00148 *Equal_409
00149 *Equal_410
00150 *Equal_411
00151 *Equal_412
00152 *Equal_413
00153 *Equal_414
00154 *Equal_415
00155 *Equal_416
00156 *Equal_417
00157 *Equal_418
00158 *Equal_419
00159 *Equal_420
00160 *Equal_421
00161 *Equal_422
00162 *Equal_423
00163 *Equal_424
00164 *Equal_425
00165 *Equal_426
00166 *Equal_427
00167 *Equal_428
00168 *Equal_429
00169 *Equal_430
00170 *Equal_431
00171 *Equal_432
00172 */
00173 #define PLLN Equal_200
00174
00175 /***** */
00176
```

```
00177 /*options:  
00178 *Equal_2  
00179 *Equal_4  
00180 *Equal_5  
00181 *Equal_6  
00182 *Equal_7  
00183 *Equal_8  
00184 *Equal_9  
00185 *Equal_10  
00186 *Equal_11  
00187 *Equal_12  
00188 *Equal_13  
00189 *Equal_14  
00190 *Equal_15  
00191 *Equal_16  
00192 *Equal_17  
00193 *Equal_18  
00194 *Equal_19  
00195 *Equal_20  
00196 *Equal_21  
00197 *Equal_22  
00198 *Equal_23  
00199 *Equal_24  
00200 *Equal_25  
00201 *Equal_26  
00202 *Equal_27  
00203 *Equal_28  
00204 *Equal_29  
00205 *Equal_30  
00206 *Equal_31  
00207 *Equal_32  
00208 *Equal_33  
00209 *Equal_34  
00210 *Equal_35  
00211 *Equal_36  
00212 *Equal_37  
00213 *Equal_38  
00214 *Equal_39  
00215 *Equal_40  
00216 *Equal_41  
00217 *Equal_42  
00218 *Equal_43  
00219 *Equal_44  
00220 *Equal_45  
00221 *Equal_46  
00222 *Equal_47  
00223 *Equal_48  
00224 *Equal_49  
00225 *Equal_50  
00226 *Equal_51  
00227 *Equal_52  
00228 *Equal_53  
00229 *Equal_54  
00230 *Equal_55  
00231 *Equal_56  
00232 *Equal_57  
00233 *Equal_58  
00234 *Equal_59  
00235 *Equal_60  
00236 *Equal_61  
00237 *Equal_62  
00238 *Equal_63  
00239 */  
00240 #define PLLM Equal_2  
00241  
00242  
00243 /******  
00244 // RCC_CFGR configurations //  
00245 ******/  
00246  
00247 /*options:  
00248 *SYSCLK  
00249 *PLL12SCLK  
00250 *HSE  
00251 *PLLCLK  
00252 */  
00253 #define MCO2 SYSCLK  
00254  
00255 /******  
00256  
00257 /*options:  
00258 *NoDivision  
00259 *DivisionBy2  
00260 *DivisionBy3  
00261 *DivisionBy4  
00262 *DivisionBy5  
00263 */
```

```
00264 #define MCO2PRE NoDivision
00265
00266 /******
00267 *****/
00268 /*options:
00269 *NoDivision
00270 *DivisionBy2
00271 *DivisionBy3
00272 *DivisionBy4
00273 *DivisionBy5
00274 */
00275 #define MCO1PRE NoDivision
00276
00277 /******
00278 *****/
00279 /*options:
00280 *PLL12SCLK
00281 *I2S_CKIN
00282 */
00283 #define I2SSRC PLL12SCLK
00284
00285 /******
00286 *****/
00287 /*options:
00288 *HSI
00289 *LSE
00290 *HSE
00291 *PLLCLK
00292 */
00293 #define MCO1 HSE
00294
00295 /******
00296 *****/
00297 /*options:
00298 *NoCLK0
00299 *NoCLK1
00300 *HSEby2
00301 *HSEby3
00302 *HSEby4
00303 *HSEby5
00304 *HSEby6
00305 *HSEby7
00306 *HSEby8
00307 *HSEby9
00308 *HSEby10
00309 *HSEby11
00310 *HSEby12
00311 *HSEby13
00312 *HSEby14
00313 *HSEby15
00314 *HSEby16
00315 *HSEby17
00316 *HSEby18
00317 *HSEby19
00318 *HSEby20
00319 *HSEby21
00320 *HSEby22
00321 *HSEby22
00322 *HSEby23
00323 *HSEby24
00324 *HSEby25
00325 *HSEby26
00326 *HSEby27
00327 *HSEby28
00328 *HSEby29
00329 *HSEby30
00330 *HSEby31
00331 */
00332 #define RTCPRE HSEby2
00333
00334 /******
00335 *****/
00336 /*options:
00337 *NoDivision
00338 *AHBby2
00339 *AHBby4
00340 *AHBBy8
00341 *AHBby16
00342 */
00343 #define PPREG2 NoDivision
00344
00345 /******
00346 *****/
00347 /*options:
00348 *NoDivision
00349 *AHBby2
00350 *AHBby4
```

```
00351 *AHBby8
00352 *AHBby16
00353 */
00354 #define PPRE1 NoDivision
00355
00356 /***** */
00357
00358 /*options:
00359 *NoDivision
00360 *SYSCLKby2
00361 *SYSCLKby4
00362 *SYSCLKby8
00363 *SYSCLKby16
00364 *SYSCLKby64
00365 *SYSCLKby128
00366 *SYSCLKby256
00367 *SYSCLKby512
00368 */
00369 #define HPRE SYSCLKby2
00370
00371 /***** */
00372
00373 /*options:
00374 *HSI
00375 *HSE
00376 *PLLCLK
00377 */
00378 #define SWS HSI
00379
00380 /***** */
00381
00382 /*options:
00383 *HSI
00384 *HSE
00385 *PLLCLK
00386 */
00387 #define SW HSI
00388
00389
00390 /***** */
00391 // RCC_AHB1ENR configurations //
00392 /***** */
00393
00394 /*options:
00395 *ENABLE
00396 *DISABLE
00397 */
00398 #define DMA2CLK DISABLE
00399
00400 /***** */
00401
00402 /*options:
00403 *ENABLE
00404 *DISABLE
00405 */
00406 #define DMA1CLK DISABLE
00407
00408 /***** */
00409
00410 /*options:
00411 *ENABLE
00412 *DISABLE
00413 */
00414 #define CRCCLK DISABLE
00415
00416 /***** */
00417
00418 /*options:
00419 *ENABLE
00420 *DISABLE
00421 */
00422 #define GPIOHCLK DISABLE
00423
00424 /***** */
00425
00426 /*options:
00427 *ENABLE
00428 *DISABLE
00429 */
00430 #define GPIOECLK DISABLE
00431
00432 /***** */
00433
00434 /*options:
00435 *ENABLE
00436 *DISABLE
00437 */
```

```
00438 #define GPIODCLK DISABLE
00439 /*****
00440 ****/
00441 /*options:
00442 *ENABLE
00443 *DISABLE
00444 */
00445 #define GPIOCCLK DISABLE
00446 /*****
00447 ****/
00448 /*options:
00449 *ENABLE
00450 *DISABLE
00451 */
00452 #define GPIOBCLK DISABLE
00453 /*****
00454 #define GPIOACLK DISABLE
00455 /*****
00456 ****/
00457 /*options:
00458 *ENABLE
00459 *DISABLE
00460 */
00461 #define OTGFS DISABLE
00462 /*****
00463 ****/
00464 /*options:
00465 // RCC_AHB2ENR configurations //*/
00466 /*****
00467 ****/
00468 /*options:
00469 *ENABLE
00470 *DISABLE
00471 */
00472 #define PWREN DISABLE
00473 /*****
00474 ****/
00475 /*options:
00476 // RCC_APB1ENR configurations //*/
00477 /*****
00478 ****/
00479 /*options:
00480 *ENABLE
00481 *DISABLE
00482 */
00483 #define I2C3EN DISABLE
00484 /*****
00485 ****/
00486 /*options:
00487 *ENABLE
00488 *DISABLE
00489 */
00490 #define I2C2EN DISABLE
00491 /*****
00492 ****/
00493 /*options:
00494 ****/
00495 /*options:
00496 *ENABLE
00497 *DISABLE
00498 */
00499 #define I2C1EN DISABLE
00500 /*****
00501 ****/
00502 /*options:
00503 *ENABLE
00504 *DISABLE
00505 */
00506 #define USART2EN DISABLE
00507 /*****
00508 ****/
00509 /*options:
00510 ****/
00511 /*options:
00512 *ENABLE
00513 *DISABLE
00514 */
00515 #define USART1EN DISABLE
00516 /*****
00517 ****/
00518 /*options:
00519 *ENABLE
00520 *DISABLE
00521 */
00522 #define SPI3EN DISABLE
00523 /*****
00524 ****/
```

```
00525 /******  
00526 /*options:  
00527 *ENABLE  
00528 *DISABLE  
00529 */  
00530 #define SPI2EN DISABLE  
00531  
00532 /******  
00533 /*options:  
00534 *ENABLE  
00535 *DISABLE  
00536 */  
00537 #define WWDGEN DISABLE  
00538  
00539 /******  
00540 /*options:  
00541 *ENABLE  
00542 *DISABLE  
00543 */  
00544 /*options:  
00545 *ENABLE  
00546 *DISABLE  
00547 */  
00548 #define TIM5EN DISABLE  
00549  
00550 /******  
00551 /*options:  
00552 *ENABLE  
00553 *DISABLE  
00554 */  
00555 #define TIM4EN DISABLE  
00556  
00557 /******  
00558 /*options:  
00559 *ENABLE  
00560 *DISABLE  
00561 */  
00562 #define TIM3EN DISABLE  
00563  
00564 /******  
00565 /*options:  
00566 *ENABLE  
00567 *DISABLE  
00568 */  
00569 #define TIM2EN DISABLE  
00570  
00571 /******  
00572 /*options:  
00573 *ENABLE  
00574 *DISABLE  
00575 */  
00576 // RCC_APB2ENR configurations //  
00577 /******  
00578 /*options:  
00579 *ENABLE  
00580 *DISABLE  
00581 */  
00582 #define TIM11EN DISABLE  
00583  
00584 /******  
00585 /*options:  
00586 *ENABLE  
00587 *DISABLE  
00588 */  
00589 #define TIM10EN DISABLE  
00590  
00591 /******  
00592 /*options:  
00593 *ENABLE  
00594 *DISABLE  
00595 */  
00596 #define TIM9EN DISABLE  
00597  
00598 /******  
00599 /*options:  
00600 *ENABLE  
00601 *DISABLE  
00602 */  
00603 /*options:  
00604 *ENABLE  
00605 *DISABLE  
00606 */  
00607 #define SYSCFGEN DISABLE  
00608  
00609 /******  
00610 /*options:  
00611 */
```

```

00612 /*ENABLE
00613 *DISABLE
00614 */
00615 #define SPI4EN DISABLE
00616
00617 /***** */
00618
00619 /*options:
00620 *ENABLE
00621 *DISABLE
00622 */
00623 #define SPILEN DISABLE
00624
00625 /***** */
00626
00627 /*options:
00628 *ENABLE
00629 *DISABLE
00630 */
00631 #define SDIOEN DISABLE
00632
00633 /***** */
00634
00635 /*options:
00636 *ENABLE
00637 *DISABLE
00638 */
00639 #define ADC1EN DISABLE
00640
00641 /***** */
00642
00643 /*options:
00644 *ENABLE
00645 *DISABLE
00646 */
00647 #define USART6EN DISABLE
00648
00649 /***** */
00650
00651 /*options:
00652 *ENABLE
00653 *DISABLE
00654 */
00655 #define USART1EN DISABLE
00656
00657 /***** */
00658
00659 /*options:
00660 *ENABLE
00661 *DISABLE
00662 */
00663 #define TIM1EN DISABLE
00664
00665 /***** */
00666
00667
00668
00669
00670
00671
00672
00673
00674
00675
00676
00677
00678
00679
00680
00681
00682 #endif /* MCAL_RCC_MRCC_CONFIG_H */

```

7.48 COTS/MCAL/RCC/MRCC_interface.h File Reference

This file contains the interfacing information for the RCC module.

Macros

- `#define RCC_AHB1 1`

- #define **RCC_AHB2** 2
 - AHB2 Bus.*
- #define **RCC_APB1** 3
 - APB1 Bus.*
- #define **RCC_APB2** 4
 - APB2 Bus.*
- #define **RCC_AHB1LPENR** 5
 - peripheral clock enable in low power mode register*
- #define **AHB1ENR_DMA2EN** 22
 - Enable CLK on DMA2 peripheral.*
- #define **AHB1ENR_DMA1EN** 21
 - Enable CLK on DMA1 peripheral.*
- #define **AHB1ENR_CRCEN** 12
 - Enable CLK on CRC peripheral.*
- #define **AHB1ENR_GPIOHEN** 7
 - Enable CLK on GPIOH peripheral.*
- #define **AHB1ENR_GPIOEEN** 4
 - Enable CLK on GPIOE peripheral.*
- #define **AHB1ENR_GPIODEN** 3
 - Enable CLK on GPIOD peripheral.*
- #define **AHB1ENR_GPIOCEN** 2
 - Enable CLK on GPIOC peripheral.*
- #define **AHB1ENR_GPIOBEN** 1
 - Enable CLK on GPIOB peripheral.*
- #define **AHB1ENR_GPIOAEN** 0
 - Enable CLK on GPIOA peripheral.*
- #define **AHB2ENR_OTGFSEN** 7
 - Enable CLK on OTGFS peripheral.*
- #define **APB1ENR_PWREN** 28
 - Enable CLK on PWR peripheral.*
- #define **APB1ENR_I2C3EN** 23
 - Enable CLK on I2C3 peripheral.*
- #define **APB1ENR_I2C2EN** 22
 - Enable CLK on I2C2 peripheral.*
- #define **APB1ENR_I2C1EN** 21
 - Enable CLK on I2C1 peripheral.*
- #define **APB1ENR_USART2EN** 17
 - Enable CLK on USART2 peripheral.*
- #define **APB1ENR_SPI3EN** 15
 - Enable CLK on SPI3 peripheral.*
- #define **APB1ENR_SPI2EN** 14
 - Enable CLK on SPI2 peripheral.*
- #define **APB1ENR_WWDGEN** 11
 - Enable CLK on WWD peripheral.*
- #define **APB1ENR_TIM5EN** 3
 - Enable CLK on TIM5 peripheral.*
- #define **APB1ENR_TIM4EN** 2
 - Enable CLK on TIM4 peripheral.*
- #define **APB1ENR_TIM3EN** 1
 - Enable CLK on TIM3 peripheral.*

- #define APB1ENR_TIM2EN 0
Enable CLK on TIM2 peripheral.
- #define APB2ENR_TIM11EN 18
Enable CLK on TIM11 peripheral.
- #define APB2ENR_TIM10EN 17
Enable CLK on TIM10 peripheral.
- #define APB2ENR_TIM9EN 16
Enable CLK on TIM9 peripheral.
- #define APB2ENR_SYSCFGEN 14
Enable CLK on SYSCFG peripheral.
- #define APB2ENR_SPI4EN 13
Enable CLK on SPI4 peripheral.
- #define APB2ENR_SPI1EN 12
Enable CLK on SPI1 peripheral.
- #define APB2ENR_SDIOEN 11
Enable CLK on SDIO peripheral.
- #define APB2ENR_ADC1EN 8
Enable CLK on ADC1 peripheral.
- #define APB2ENR_USART6EN 5
Enable CLK on USART6 peripheral.
- #define APB2ENR_USART1EN 4
Enable CLK on USART1 peripheral.
- #define APB2ENR_TIM1EN 0
Enable CLK on TIM1 peripheral.
- #define AHB1LPENR_FLITFLPEN 15
Enable CLK on FLITFLP peripheral.

Functions

- void MRCC_vInit (void)
Initialize the RCC with a certain configurations.
- void MRCC_vEnablePeriphralCLK (VAR(u32_t) A_u32BusID, VAR(u32_t) A_u32PeriphralID)
Enabling the CLK on a specific Peripheral.
- void MRCC_vDisablePeriphralCLK (VAR(u32_t) A_u32BusID, VAR(u32_t) A_u32PeriphralID)
Disabling the CLK on a specific Peripheral.

7.48.1 Detailed Description

This file contains the interfacing information for the RCC module.

Author

Ali El Bana & Mo Alaa

Version

2.0

Date

11/9/2022

Definition in file [MRCC_interface.h](#).

7.48.2 Function Documentation

7.48.2.1 MRCC_vInit()

```
void MRCC_vInit (
    void )
```

Initialize the RCC with a certain configurations.

Definition at line 29 of file [MRCC_program.c](#).

```
00030 {
00031     // PLLI2S (ON/OFF).
00032 #if PLLI2S == ENABLE
00033     SET_BIT( RCC->CR, RCC_CR_PLLI2SON ) ;
00036
00037 #elif PLLI2S == DISABLE
00038     CLR_BIT( RCC->CR, RCC_CR_PLLI2SON ) ;
00040
00041 #endif
00042
00043
00044     // PLL (ON/OFF).
00045 #if PLL == ENABLE
00046     SET_BIT( RCC->CR, RCC_CR_PLLON ) ;
00048
00049 #elif PLL == DISABLE
00050     CLR_BIT( RCC->CR, RCC_CR_PLLON ) ;
00052
00053 #endif
00054
00055
00056     // CSS (ON/OFF).
00057 #if CSS == ENABLE
00058     SET_BIT( RCC->CR, RCC_CR_CSSON ) ;
00060
00061 #elif CSS == DISABLE
00062     CLR_BIT( RCC->CR, RCC_CR_CSSON ) ;
00064
00065 #endif
00066
00067
00068     // HSEBYP.
00069 #if HSEBYP == BYBASED
00070     SET_BIT( RCC->CR, RCC_CR_HSEBYP ) ;
00072
00073 #elif HSEBYP == NOTBYBASED
00074     CLR_BIT( RCC->CR, RCC_CR_HSEBYP ) ;
00076
00077 #endif
00078
00079
00080     // Select CLK switch (HSI/HSE/PLL).
00081 #if SW == HSI
00082
00083     CLR_BIT( RCC->CFGR, RCC_CFGR_SW_b0 ) ;
00084     CLR_BIT( RCC->CFGR, RCC_CFGR_SW_b1 ) ;
00085
00086 #elif SW == HSE
00087
00088     SET_BIT( RCC->CFGR, RCC_CFGR_SW_b0 ) ;
00089     CLR_BIT( RCC->CFGR, RCC_CFGR_SW_b1 ) ;
00090
00091 #elif SW == PLLCLK
00092
00093     CLR_BIT( RCC->CFGR, RCC_CFGR_SW_b0 ) ;
00094     SET_BIT( RCC->CFGR, RCC_CFGR_SW_b1 ) ;
00095
```

```

00096 #endif
00097
00098     // Select CLK switch status (HSI/HSE/PLL).
00099 #if SWS == HSI
00100         CLR_BIT( RCC->CFGR, RCC_CFGR_SWS_b0 ) ;
00101         CLR_BIT( RCC->CFGR, RCC_CFGR_SWS_b1 ) ;
00103
00104 #elif SWS == HSE
00105
00106         SET_BIT( RCC->CFGR, RCC_CFGR_SWS_b0 ) ;
00107         CLR_BIT( RCC->CFGR, RCC_CFGR_SWS_b1 ) ;
00108
00109 #elif SWS == PLLCLK
00110
00111         CLR_BIT( RCC->CFGR, RCC_CFGR_SWS_b0 ) ;
00112         SET_BIT( RCC->CFGR, RCC_CFGR_SWS_b1 ) ;
00113
00114 #endif
00115
00116
00117     // MCO2 selection:
00118 #if MCO2 == SYSCLK
00119
00120         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2_b0 ) ;
00121         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2_b1 ) ;
00122
00123 #elif MCO2 == PLLI2SCLK
00124
00125         SET_BIT( RCC->CFGR, RCC_CFGR_MOC2_b0 ) ;
00126         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2_b1 ) ;
00127
00128 #elif MCO2 == HSE
00129
00130         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2_b0 ) ;
00131         SET_BIT( RCC->CFGR, RCC_CFGR_MOC2_b1 ) ;
00132
00133 #elif MCO2 == PLLCLK
00134
00135         SET_BIT( RCC->CFGR, RCC_CFGR_MOC2_b0 ) ;
00136         SET_BIT( RCC->CFGR, RCC_CFGR_MOC2_b1 ) ;
00137
00138 #endif
00139
00140
00141     // MCO2 prescaler:
00142 #if MCO2PRE == NoDivision
00143
00144         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b0 ) ;
00145         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b1 ) ;
00146         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b2 ) ;
00147
00148 #elif MCO2PRE == DivisionBy2
00149
00150         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b0 ) ;
00151         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b1 ) ;
00152         SET_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b2 ) ;
00153
00154 #elif MCO2PRE == DivisionBy3
00155
00156         SET_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b0 ) ;
00157         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b1 ) ;
00158         SET_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b2 ) ;
00159
00160 #elif MCO2PRE == DivisionBy4
00161
00162         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b0 ) ;
00163         SET_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b1 ) ;
00164         SET_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b2 ) ;
00165
00166 #elif MCO2PRE == DivisionBy5
00167
00168         SET_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b0 ) ;
00169         SET_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b1 ) ;
00170         SET_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b2 ) ;
00171
00172 #endif
00173
00174
00175     // MCO1 selection:
00176 #if MCO1 == HSI
00177
00178         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1_b0 ) ;
00179         CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1_b1 ) ;
00180
00181 #elif MCO1 == LSE
00182

```

```

00183     SET_BIT( RCC->CFGR, RCC_CFGR_MOC1_b0 ) ;
00184     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1_b1 ) ;
00185
00186 #elif MCO1 == HSE
00187
00188     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1_b0 ) ;
00189     SET_BIT( RCC->CFGR, RCC_CFGR_MOC1_b1 ) ;
00190
00191 #elif MCO1 == PLLCLK
00192
00193     SET_BIT( RCC->CFGR, RCC_CFGR_MOC1_b0 ) ;
00194     SET_BIT( RCC->CFGR, RCC_CFGR_MOC1_b1 ) ;
00195
00196 #endif
00197
00198
00199 // MCO1 prescaler:
00200 #if MCO1PRE == NoDivision
00201
00202     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b0 ) ;
00203     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b1 ) ;
00204     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b2 ) ;
00205
00206 #elif MCO1PRE == DivisionBy2
00207
00208     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b0 ) ;
00209     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b1 ) ;
00210     SET_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b2 ) ;
00211
00212 #elif MCO1PRE == DivisionBy3
00213
00214     SET_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b0 ) ;
00215     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b1 ) ;
00216     SET_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b2 ) ;
00217
00218 #elif MCO1PRE == DivisionBy4
00219
00220     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b0 ) ;
00221     SET_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b1 ) ;
00222     SET_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b2 ) ;
00223
00224 #elif MCO1PRE == DivisionBy5
00225
00226     SET_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b0 ) ;
00227     SET_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b1 ) ;
00228     SET_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b2 ) ;
00229
00230 #endif
00231
00232
00233 // AHB prescalers:
00234 #if HPRE == NoDivision
00235
00236     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00237     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00238     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00239     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00240
00241 #elif HPRE == SYSCLKby2
00242
00243     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00244     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00245     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00246     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00247
00248 #elif HPRE == SYSCLKby4
00249
00250     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00251     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00252     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00253     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00254
00255 #elif HPRE == SYSCLKby8
00256
00257     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00258     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00259     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00260     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00261
00262 #elif HPRE == SYSCLKby16
00263
00264     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00265     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00266     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00267     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00268
00269 #elif HPRE == SYSCLKby64

```

```

00270
00271     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00272     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00273     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00274     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00275
00276 #elif HPRE == SYSCLKby128
00277
00278     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00279     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00280     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00281     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00282
00283 #elif HPRE == SYSCLKby256
00284
00285     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00286     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00287     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00288     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00289
00290 #elif HPRE == SYSCLKby512
00291
00292     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00293     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00294     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00295     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00296
00297 #endif
00298
00299
00300 // APB1 prescalers:
00301 #if PPREG == NoDivision
00302
00303     CLR_BIT( RCC->CFGR, RCC_CFGR_PPREG_b0 ) ;
00304     CLR_BIT( RCC->CFGR, RCC_CFGR_PPREG_b1 ) ;
00305     CLR_BIT( RCC->CFGR, RCC_CFGR_PPREG_b2 ) ;
00306
00307 #elif PPREG == AHBby2
00308
00309     CLR_BIT( RCC->CFGR, RCC_CFGR_PPREG_b0 ) ;
00310     CLR_BIT( RCC->CFGR, RCC_CFGR_PPREG_b1 ) ;
00311     SET_BIT( RCC->CFGR, RCC_CFGR_PPREG_b2 ) ;
00312
00313 #elif PPREG == AHBby4
00314
00315     SET_BIT( RCC->CFGR, RCC_CFGR_PPREG_b0 ) ;
00316     CLR_BIT( RCC->CFGR, RCC_CFGR_PPREG_b1 ) ;
00317     SET_BIT( RCC->CFGR, RCC_CFGR_PPREG_b2 ) ;
00318
00319 #elif PPREG == AHBby8
00320
00321     CLR_BIT( RCC->CFGR, RCC_CFGR_PPREG_b0 ) ;
00322     SET_BIT( RCC->CFGR, RCC_CFGR_PPREG_b1 ) ;
00323     SET_BIT( RCC->CFGR, RCC_CFGR_PPREG_b2 ) ;
00324
00325 #elif PPREG == AHBby16
00326
00327     SET_BIT( RCC->CFGR, RCC_CFGR_PPREG_b0 ) ;
00328     SET_BIT( RCC->CFGR, RCC_CFGR_PPREG_b1 ) ;
00329     SET_BIT( RCC->CFGR, RCC_CFGR_PPREG_b2 ) ;
00330
00331 #endif
00332
00333 // APB2 prescalers:
00334 #if PPREG == NoDivision
00335
00336     CLR_BIT( RCC->CFGR, RCC_CFGR_PPREG_b0 ) ;
00337     CLR_BIT( RCC->CFGR, RCC_CFGR_PPREG_b1 ) ;
00338     CLR_BIT( RCC->CFGR, RCC_CFGR_PPREG_b2 ) ;
00339
00340 #elif PPREG == AHBby2
00341
00342     CLR_BIT( RCC->CFGR, RCC_CFGR_PPREG_b0 ) ;
00343     CLR_BIT( RCC->CFGR, RCC_CFGR_PPREG_b1 ) ;
00344     SET_BIT( RCC->CFGR, RCC_CFGR_PPREG_b2 ) ;
00345
00346 #elif PPREG == AHBby4
00347
00348     SET_BIT( RCC->CFGR, RCC_CFGR_PPREG_b0 ) ;
00349     CLR_BIT( RCC->CFGR, RCC_CFGR_PPREG_b1 ) ;
00350     SET_BIT( RCC->CFGR, RCC_CFGR_PPREG_b2 ) ;
00351
00352 #elif PPREG == AHBby8
00353
00354     CLR_BIT( RCC->CFGR, RCC_CFGR_PPREG_b0 ) ;
00355     SET_BIT( RCC->CFGR, RCC_CFGR_PPREG_b1 ) ;
00356     SET_BIT( RCC->CFGR, RCC_CFGR_PPREG_b2 ) ;

```

```

00357
00358 #elif PPRE2 == AHBy16
00359
00360     SET_BIT( RCC->CFGR, RCC_CFGR_PPRE2_b0 ) ;
00361     SET_BIT( RCC->CFGR, RCC_CFGR_PPRE2_b1 ) ;
00362     SET_BIT( RCC->CFGR, RCC_CFGR_PPRE2_b2 ) ;
00363
00364 #endif
00365
00366
00367 // PLL configurations:
00368
00369
00370
00371     // Enable the selected CLK (HSI ON/HSE ON/PLL ON): //
00372     // HSE (ON/OFF).
00373 #if HSE_EN == ENABLE
00374
00375     SET_BIT( RCC->CR, RCC_CR_HSEON ) ;
00376
00377 #elif HSE_EN == DISABLE
00378
00379     CLR_BIT( RCC->CR, RCC_CR_HSEON ) ;
00380
00381 #endif
00382
00383
00384 // HSI (ON/OFF).
00385 #if HSI_EN == ENABLE
00386
00387     SET_BIT( RCC->CR, RCC_CR_HSION ) ;
00388
00389 #elif HSI_EN == DISABLE
00390
00391     CLR_BIT( RCC->CR, RCC_CR_HSION ) ;
00392
00393 #endif
00394
00395 }

```

References [CLR_BIT](#), [RCC](#), [RCC_CFGR_HPRE_b0](#), [RCC_CFGR_HPRE_b1](#), [RCC_CFGR_HPRE_b2](#), [RCC_CFGR_HPRE_b3](#), [RCC_CFGR_MOC1_b0](#), [RCC_CFGR_MOC1_b1](#), [RCC_CFGR_MOC1PRE_b0](#), [RCC_CFGR_MOC1PRE_b1](#), [RCC_CFGR_MOC1PRE_b2](#), [RCC_CFGR_MOC2_b0](#), [RCC_CFGR_MOC2_b1](#), [RCC_CFGR_MOC2PRE_b0](#), [RCC_CFGR_MOC2PRE_b1](#), [RCC_CFGR_MOC2PRE_b2](#), [RCC_CFGR_PPREG1_b0](#), [RCC_CFGR_PPREG1_b1](#), [RCC_CFGR_PPREG1_b2](#), [RCC_CFGR_PPREG2_b0](#), [RCC_CFGR_PPREG2_b1](#), [RCC_CFGR_PPREG2_b2](#), [RCC_CFGR_SW_b0](#), [RCC_CFGR_SW_b1](#), [RCC_CFGR_SWS_b0](#), [RCC_CFGR_SWS_b1](#), [RCC_CR_CS0N](#), [RCC_CR_HSEBYP](#), [RCC_CR_HSEON](#), [RCC_CR_HS0N](#), [RCC_CR_PLI2SON](#), [RCC_CR_PLL0N](#), and [SET_BIT](#).

7.48.2.2 MRCC_vEnablePeriphralCLK()

```

void MRCC_vEnablePeriphralCLK (
    VAR(u32_t) A_u32BusID,
    VAR(u32_t) A_u32PeriphralID )

```

Enabling the CLK on a specific Peripheral.

Parameters

in	A_u32BusID	Bus ID.
in	A_u32PeriphralID	Peripheral ID.

7.48.2.3 MRCC_vDisablePeriphralCLK()

```
void MRCC_vDisablePeriphralCLK (
    VAR(u32_t) A_u32BusID,
    VAR(u32_t) A_u32PeriphralID )
```

Disabling the CLK on a specific Peripheral.

Parameters

in	A_u32BusID	Bus ID.
in	A_u32PeriphralID	Peripheral ID.

7.49 MRCC_interface.h

[Go to the documentation of this file.](#)

```
00001
00009 /* Header file guard */
00010 #ifndef MCAL_RCC_MRCC_INTERFACE_H_
00011 #define MCAL_RCC_MRCC_INTERFACE_H_
00012
00013 /***** Functions prototypes ****/
00014 /*          Functions prototypes          */
00015 /***** Interfacing macros ****/
00016
00020 void MRCC_vInit(void);
00021
00027 void MRCC_vEnablePeriphralCLK(VAR(u32_t) A_u32BusID, VAR(u32_t) A_u32PeriphralID);
00028
00034 void MRCC_vDisablePeriphralCLK(VAR(u32_t) A_u32BusID, VAR(u32_t) A_u32PeriphralID);
00035
00036 /***** Interfacing macros ****/
00037 /*          Interfacing macros          */
00038 /***** Macros ****/
00039
00051 #define RCC_AHB1 1
00052
00058 #define RCC_AHB2 2
00059
00065 #define RCC_APB1 3
00066
00072 #define RCC_APB2 4
00073
00079 #define RCC_AHB1LPENR 5
00080
00094 #define AHB1ENR_DMA2EN 22
00095
00101 #define AHB1ENR_DMA1EN 21
00102
00108 #define AHB1ENR_CRCEN 12
00109
00115 #define AHB1ENR_GPIOHEN 7
00116
00122 #define AHB1ENR_GPIOEEN 4
00123
00129 #define AHB1ENR_GPIODEN 3
00130
00136 #define AHB1ENR_GPIOCEN 2
00137
00143 #define AHB1ENR_GPIOBEN 1
00144
00150 #define AHB1ENR_GPIOAEN 0
00151
00157 #define AHB2ENR_OTGFSEN 7
00158
00164 #define APB1ENR_PWREN 28
00165
00171 #define APB1ENR_I2C3EN 23
00172
00178 #define APB1ENR_I2C2EN 22
00179
00185 #define APB1ENR_I2C1EN 21
```

```

00186
00192 #define APB1ENR_USART2EN 17
00193
00199 #define APB1ENR_SPI3EN 15
00200
00206 #define APB1ENR_SPI2EN 14
00207
00213 #define APB1ENR_WWDGEN 11
00214
00220 #define APB1ENR_TIM5EN 3
00221
00227 #define APB1ENR_TIM4EN 2
00228
00234 #define APB1ENR_TIM3EN 1
00235
00241 #define APB1ENR_TIM2EN 0
00242
00248 #define APB2ENR_TIM11EN 18
00249
00255 #define APB2ENR_TIM10EN 17
00256
00262 #define APB2ENR_TIM9EN 16
00263
00269 #define APB2ENR_SYSCFGEN 14
00270
00276 #define APB2ENR_SPI4EN 13
00277
00283 #define APB2ENR_SPI1EN 12
00284
00290 #define APB2ENR_SDIOEN 11
00291
00297 #define APB2ENR_ADC1EN 8
00298
00304 #define APB2ENR_USART6EN 5
00305
00311 #define APB2ENR_USART1EN 4
00312
00318 #define APB2ENR_TIM1EN 0
00319
00325 #define AHB1LPENR_FLITFLPEN 15
00326
00329 #endif /* MCAL_RCC_MRCC_INTERFACE_H_ */

```

7.50 COTS/MCAL/RCC/MRCC_private.h File Reference

This file contains the registers information and addresses for the RCC module.

Data Structures

- struct [RCC_MemoryMapType](#)
RCC declaration structure for its registers.

Macros

- `#define RCC_BASE_ADDRESS 0x40023800`
- `#define RCC (volatile P2VAR(RCC_MemoryMapType)) (RCC_BASE_ADDRESS))`
- `#define RCC_CR_PLLI2SRDY 27`
- `#define RCC_CR_PLLI2SON 26`
- `#define RCC_CR_PLLRDY 25`
- `#define RCC_CR_PLLON 24`
- `#define RCC_CR_CSSON 19`
- `#define RCC_CR_HSEBYP 18`
- `#define RCC_CR_HSERDY 17`
- `#define RCC_CR_HSEON 16`
- `#define RCC_CR_HSIRDY 1`
- `#define RCC_CR_HSION 0`

- #define RCC_PLLCFGR_PLLQ_b0 24
- #define RCC_PLLCFGR_PLLQ_b1 25
- #define RCC_PLLCFGR_PLLQ_b2 26
- #define RCC_PLLCFGR_PLLQ_b3 27
- #define RCC_PLLCFGR_PLLSRC 22
- #define RCC_PLLCFGR_PLLP_b0 16
- #define RCC_PLLCFGR_PLLP_b1 17
- #define RCC_PLLCFGR_PLLN_b0 6
- #define RCC_PLLCFGR_PLLN_b1 7
- #define RCC_PLLCFGR_PLLN_b2 8
- #define RCC_PLLCFGR_PLLN_b3 9
- #define RCC_PLLCFGR_PLLN_b4 10
- #define RCC_PLLCFGR_PLLN_b5 11
- #define RCC_PLLCFGR_PLLN_b6 12
- #define RCC_PLLCFGR_PLLN_b7 13
- #define RCC_PLLCFGR_PLLN_b8 14
- #define RCC_PLLCFGR_PLLM_b0 0
- #define RCC_PLLCFGR_PLLM_b1 1
- #define RCC_PLLCFGR_PLLM_b2 2
- #define RCC_PLLCFGR_PLLM_b3 3
- #define RCC_PLLCFGR_PLLM_b4 4
- #define RCC_PLLCFGR_PLLM_b5 5
- #define RCC_CFGR_MOC2_b0 30
- #define RCC_CFGR_MOC2_b1 31
- #define RCC_CFGR_MOC2PRE_b0 27
- #define RCC_CFGR_MOC2PRE_b1 28
- #define RCC_CFGR_MOC2PRE_b2 29
- #define RCC_CFGR_MOC1PRE_b0 24
- #define RCC_CFGR_MOC1PRE_b1 25
- #define RCC_CFGR_MOC1PRE_b2 26
- #define RCC_CFGR_I2SSRC 23
- #define RCC_CFGR_MOC1_b0 21
- #define RCC_CFGR_MOC1_b1 22
- #define RCC_CFGR_RTCPRE_b0 16
- #define RCC_CFGR_RTCPRE_b1 17
- #define RCC_CFGR_RTCPRE_b2 18
- #define RCC_CFGR_RTCPRE_b3 19
- #define RCC_CFGR_RTCPRE_b4 20
- #define RCC_CFGR_PPREG2_b0 13
- #define RCC_CFGR_PPREG2_b1 14
- #define RCC_CFGR_PPREG2_b2 15
- #define RCC_CFGR_PPREG1_b0 10
- #define RCC_CFGR_PPREG1_b1 11
- #define RCC_CFGR_PPREG1_b2 12
- #define RCC_CFGR_HPRE_b0 4
- #define RCC_CFGR_HPRE_b1 5
- #define RCC_CFGR_HPRE_b2 6
- #define RCC_CFGR_HPRE_b3 7
- #define RCC_CFGR_SWS_b0 2
- #define RCC_CFGR_SWS_b1 3
- #define RCC_CFGR_SW_b0 0
- #define RCC_CFGR_SW_b1 1
- #define RCC_AHB1ENR_DMA2EN 22
- #define RCC_AHB1ENR_DMA1EN 21
- #define RCC_AHB1ENR_CRCEN 12

- #define RCC_AHB1ENR_GPIOHEN 7
- #define RCC_AHB1ENR_GPIOEEN 4
- #define RCC_AHB1ENR_GPIODEN 3
- #define RCC_AHB1ENR_GPIOCEN 2
- #define RCC_AHB1ENR_GPIOBEN 1
- #define RCC_AHB1ENR_GPIOAEN 0
- #define RCC_AHB2ENR_OTGFSEN 7
- #define RCC_APB1ENR_PWREN 28
- #define RCC_APB1ENR_I2C3EN 23
- #define RCC_APB1ENR_I2C2EN 22
- #define RCC_APB1ENR_I2C1EN 21
- #define RCC_APB1ENR_USART2EN 17
- #define RCC_APB1ENR_SPI3EN 15
- #define RCC_APB1ENR_SPI2EN 14
- #define RCC_APB1ENR_WWDGEN 11
- #define RCC_APB1ENR_TIM5EN 3
- #define RCC_APB1ENR_TIM4EN 2
- #define RCC_APB1ENR_TIM3EN 1
- #define RCC_APB1ENR_TIM2EN 0
- #define RCC_APB2ENR_TIM11EN 18
- #define RCC_APB2ENR_TIM10EN 17
- #define RCC_APB2ENR_TIM9EN 16
- #define RCC_APB2ENR_SYSCFGEN 14
- #define RCC_APB2ENR_SPI4EN 13
- #define RCC_APB2ENR_SPI1EN 12
- #define RCC_APB2ENR_SDIOEN 11
- #define RCC_APB2ENR_ADC1EN 8
- #define RCC_APB2ENR_USART6EN 5
- #define RCC_APB2ENR_USART1EN 4
- #define RCC_APB2ENR_TIM1EN 0
- #define RCC_AHB1LPENR_FLITFLPEN 15
- #define ENABLE 1
- #define DISABLE 2
- #define BYBASED 1
- #define NOTBYBASED 2
- #define SYSCLK 1
- #define PLLI2SCLK 5
- #define HSE 3
- #define PLLCLK 4
- #define NoDivision 1
- #define DivisionBy2 2
- #define DivisionBy3 3
- #define DivisionBy4 4
- #define DivisionBy5 5
- #define I2S_CKIN 2
- #define HSI 1
- #define LSE 2
- #define PLLCLK 4
- #define SYSCLKby2 2
- #define SYSCLKby4 3
- #define SYSCLKby8 4
- #define SYSCLKby16 5
- #define SYSCLKby64 6
- #define SYSCLKby128 7
- #define SYSCLKby256 8

- #define SYSCLKby512 9
- #define NoCLK0 1
- #define NoCLK1 2
- #define HSEby2 3
- #define HSEby3 4
- #define HSEby4 5
- #define HSEby5 6
- #define HSEby6 7
- #define HSEby7 8
- #define HSEby8 9
- #define HSEby9 10
- #define HSEby10 11
- #define HSEby11 12
- #define HSEby12 13
- #define HSEby13 14
- #define HSEby14 15
- #define HSEby15 16
- #define HSEby16 17
- #define HSEby17 18
- #define HSEby18 19
- #define HSEby19 20
- #define HSEby20 21
- #define HSEby21 22
- #define HSEby22 23
- #define HSEby23 24
- #define HSEby24 25
- #define HSEby25 26
- #define HSEby26 27
- #define HSEby27 28
- #define HSEby28 39
- #define HSEby29 30
- #define HSEby30 31
- #define HSEby31 32
- #define AHBby2 1
- #define AHBby4 2
- #define AHBby8 3
- #define AHBby16 4
- #define Equal_0 0
- #define Equal_1 1
- #define Equal_2 2
- #define Equal_3 3
- #define Equal_4 4
- #define Equal_5 5
- #define Equal_6 6
- #define Equal_7 7
- #define Equal_8 8
- #define Equal_9 9
- #define Equal_10 10
- #define Equal_11 11
- #define Equal_12 12
- #define Equal_13 13
- #define Equal_14 14
- #define Equal_15 15
- #define Equal_16 16
- #define Equal_17 17

- #define Equal_18 18
- #define Equal_19 19
- #define Equal_20 20
- #define Equal_21 21
- #define Equal_22 22
- #define Equal_23 23
- #define Equal_24 24
- #define Equal_25 25
- #define Equal_26 26
- #define Equal_27 27
- #define Equal_28 28
- #define Equal_29 29
- #define Equal_30 30
- #define Equal_31 31
- #define Equal_32 32
- #define Equal_33 33
- #define Equal_34 34
- #define Equal_35 35
- #define Equal_36 36
- #define Equal_37 37
- #define Equal_38 38
- #define Equal_39 39
- #define Equal_40 40
- #define Equal_41 41
- #define Equal_42 42
- #define Equal_43 43
- #define Equal_44 44
- #define Equal_45 45
- #define Equal_46 46
- #define Equal_47 47
- #define Equal_48 48
- #define Equal_49 49
- #define Equal_50 50
- #define Equal_51 51
- #define Equal_52 52
- #define Equal_53 53
- #define Equal_54 54
- #define Equal_55 55
- #define Equal_56 56
- #define Equal_57 57
- #define Equal_58 58
- #define Equal_59 59
- #define Equal_60 60
- #define Equal_61 61
- #define Equal_62 62
- #define Equal_63 63

7.50.1 Detailed Description

This file contains the registers information and addresses for the RCC module.

Author

Ali El Bana & Mo Alaa

Version

2.0

Date

11/9/2022

Definition in file [MRCC_private.h](#).

7.50.2 Macro Definition Documentation

7.50.2.1 RCC_BASE_ADDRESS

```
#define RCC_BASE_ADDRESS 0x40023800
```

RCC Base Address

Definition at line 211 of file [MRCC_private.h](#).

7.50.2.2 RCC

```
#define RCC ( (volatile P2VAR(RCC_MemoryMapType) ) (RCC_BASE_ADDRESS) )
```

RCC register

Definition at line 218 of file [MRCC_private.h](#).

7.50.2.3 RCC_CR_PLLI2SRDY

```
#define RCC_CR_PLLI2SRDY 27
```

Definition at line 225 of file [MRCC_private.h](#).

7.50.2.4 RCC_CR_PLLI2SON

```
#define RCC_CR_PLLI2SON 26
```

Definition at line [226](#) of file [MRCC_private.h](#).

7.50.2.5 RCC_CR_PLLRDY

```
#define RCC_CR_PLLRDY 25
```

Definition at line [227](#) of file [MRCC_private.h](#).

7.50.2.6 RCC_CR_PLLON

```
#define RCC_CR_PLLON 24
```

Definition at line [228](#) of file [MRCC_private.h](#).

7.50.2.7 RCC_CR_CSSON

```
#define RCC_CR_CSSON 19
```

Definition at line [229](#) of file [MRCC_private.h](#).

7.50.2.8 RCC_CR_HSEBYP

```
#define RCC_CR_HSEBYP 18
```

Definition at line [230](#) of file [MRCC_private.h](#).

7.50.2.9 RCC_CR_HSERDY

```
#define RCC_CR_HSERDY 17
```

Definition at line [231](#) of file [MRCC_private.h](#).

7.50.2.10 RCC_CR_HSEON

```
#define RCC_CR_HSEON 16
```

Definition at line [232](#) of file [MRCC_private.h](#).

7.50.2.11 RCC_CR_HSIRDY

```
#define RCC_CR_HSIRDY 1
```

Definition at line [233](#) of file [MRCC_private.h](#).

7.50.2.12 RCC_CR_HSION

```
#define RCC_CR_HSION 0
```

Definition at line [234](#) of file [MRCC_private.h](#).

7.50.2.13 RCC_PLLCFGR_PLLQ_b0

```
#define RCC_PLLCFGR_PLLQ_b0 24
```

Definition at line [237](#) of file [MRCC_private.h](#).

7.50.2.14 RCC_PLLCFGR_PLLQ_b1

```
#define RCC_PLLCFGR_PLLQ_b1 25
```

Definition at line [238](#) of file [MRCC_private.h](#).

7.50.2.15 RCC_PLLCFGR_PLLQ_b2

```
#define RCC_PLLCFGR_PLLQ_b2 26
```

Definition at line [239](#) of file [MRCC_private.h](#).

7.50.2.16 RCC_PLLCFGR_PLLQ_b3

```
#define RCC_PLLCFGR_PLLQ_b3 27
```

Definition at line 240 of file [MRCC_private.h](#).

7.50.2.17 RCC_PLLCFGR_PLLSRC

```
#define RCC_PLLCFGR_PLLSRC 22
```

Definition at line 241 of file [MRCC_private.h](#).

7.50.2.18 RCC_PLLCFGR_PLLP_b0

```
#define RCC_PLLCFGR_PLLP_b0 16
```

Definition at line 242 of file [MRCC_private.h](#).

7.50.2.19 RCC_PLLCFGR_PLLP_b1

```
#define RCC_PLLCFGR_PLLP_b1 17
```

Definition at line 243 of file [MRCC_private.h](#).

7.50.2.20 RCC_PLLCFGR_PLLN_b0

```
#define RCC_PLLCFGR_PLLN_b0 6
```

Definition at line 244 of file [MRCC_private.h](#).

7.50.2.21 RCC_PLLCFGR_PLLN_b1

```
#define RCC_PLLCFGR_PLLN_b1 7
```

Definition at line 245 of file [MRCC_private.h](#).

7.50.2.22 RCC_PLLCFGR_PLLN_b2

```
#define RCC_PLLCFGR_PLLN_b2 8
```

Definition at line [246](#) of file [MRCC_private.h](#).

7.50.2.23 RCC_PLLCFGR_PLLN_b3

```
#define RCC_PLLCFGR_PLLN_b3 9
```

Definition at line [247](#) of file [MRCC_private.h](#).

7.50.2.24 RCC_PLLCFGR_PLLN_b4

```
#define RCC_PLLCFGR_PLLN_b4 10
```

Definition at line [248](#) of file [MRCC_private.h](#).

7.50.2.25 RCC_PLLCFGR_PLLN_b5

```
#define RCC_PLLCFGR_PLLN_b5 11
```

Definition at line [249](#) of file [MRCC_private.h](#).

7.50.2.26 RCC_PLLCFGR_PLLN_b6

```
#define RCC_PLLCFGR_PLLN_b6 12
```

Definition at line [250](#) of file [MRCC_private.h](#).

7.50.2.27 RCC_PLLCFGR_PLLN_b7

```
#define RCC_PLLCFGR_PLLN_b7 13
```

Definition at line [251](#) of file [MRCC_private.h](#).

7.50.2.28 RCC_PLLCFGR_PLLN_b8

```
#define RCC_PLLCFGR_PLLN_b8 14
```

Definition at line 252 of file [MRCC_private.h](#).

7.50.2.29 RCC_PLLCFGR_PLLM_b0

```
#define RCC_PLLCFGR_PLLM_b0 0
```

Definition at line 253 of file [MRCC_private.h](#).

7.50.2.30 RCC_PLLCFGR_PLLM_b1

```
#define RCC_PLLCFGR_PLLM_b1 1
```

Definition at line 254 of file [MRCC_private.h](#).

7.50.2.31 RCC_PLLCFGR_PLLM_b2

```
#define RCC_PLLCFGR_PLLM_b2 2
```

Definition at line 255 of file [MRCC_private.h](#).

7.50.2.32 RCC_PLLCFGR_PLLM_b3

```
#define RCC_PLLCFGR_PLLM_b3 3
```

Definition at line 256 of file [MRCC_private.h](#).

7.50.2.33 RCC_PLLCFGR_PLLM_b4

```
#define RCC_PLLCFGR_PLLM_b4 4
```

Definition at line 257 of file [MRCC_private.h](#).

7.50.2.34 RCC_PLLCFGR_PLLM_b5

```
#define RCC_PLLCFGR_PLLM_b5 5
```

Definition at line [258](#) of file [MRCC_private.h](#).

7.50.2.35 RCC_CFGR_MOC2_b0

```
#define RCC_CFGR_MOC2_b0 30
```

Definition at line [261](#) of file [MRCC_private.h](#).

7.50.2.36 RCC_CFGR_MOC2_b1

```
#define RCC_CFGR_MOC2_b1 31
```

Definition at line [262](#) of file [MRCC_private.h](#).

7.50.2.37 RCC_CFGR_MOC2PRE_b0

```
#define RCC_CFGR_MOC2PRE_b0 27
```

Definition at line [263](#) of file [MRCC_private.h](#).

7.50.2.38 RCC_CFGR_MOC2PRE_b1

```
#define RCC_CFGR_MOC2PRE_b1 28
```

Definition at line [264](#) of file [MRCC_private.h](#).

7.50.2.39 RCC_CFGR_MOC2PRE_b2

```
#define RCC_CFGR_MOC2PRE_b2 29
```

Definition at line [265](#) of file [MRCC_private.h](#).

7.50.2.40 RCC_CFGR_MOC1PRE_b0

```
#define RCC_CFGR_MOC1PRE_b0 24
```

Definition at line [266](#) of file [MRCC_private.h](#).

7.50.2.41 RCC_CFGR_MOC1PRE_b1

```
#define RCC_CFGR_MOC1PRE_b1 25
```

Definition at line [267](#) of file [MRCC_private.h](#).

7.50.2.42 RCC_CFGR_MOC1PRE_b2

```
#define RCC_CFGR_MOC1PRE_b2 26
```

Definition at line [268](#) of file [MRCC_private.h](#).

7.50.2.43 RCC_CFGR_I2SSRC

```
#define RCC_CFGR_I2SSRC 23
```

Definition at line [269](#) of file [MRCC_private.h](#).

7.50.2.44 RCC_CFGR_MOC1_b0

```
#define RCC_CFGR_MOC1_b0 21
```

Definition at line [270](#) of file [MRCC_private.h](#).

7.50.2.45 RCC_CFGR_MOC1_b1

```
#define RCC_CFGR_MOC1_b1 22
```

Definition at line [271](#) of file [MRCC_private.h](#).

7.50.2.46 RCC_CFGR_RTCPRE_b0

```
#define RCC_CFGR_RTCPRE_b0 16
```

Definition at line [272](#) of file [MRCC_private.h](#).

7.50.2.47 RCC_CFGR_RTCPRE_b1

```
#define RCC_CFGR_RTCPRE_b1 17
```

Definition at line [273](#) of file [MRCC_private.h](#).

7.50.2.48 RCC_CFGR_RTCPRE_b2

```
#define RCC_CFGR_RTCPRE_b2 18
```

Definition at line [274](#) of file [MRCC_private.h](#).

7.50.2.49 RCC_CFGR_RTCPRE_b3

```
#define RCC_CFGR_RTCPRE_b3 19
```

Definition at line [275](#) of file [MRCC_private.h](#).

7.50.2.50 RCC_CFGR_RTCPRE_b4

```
#define RCC_CFGR_RTCPRE_b4 20
```

Definition at line [276](#) of file [MRCC_private.h](#).

7.50.2.51 RCC_CFGR_PPREG2_b0

```
#define RCC_CFGR_PPREG2_b0 13
```

Definition at line [277](#) of file [MRCC_private.h](#).

7.50.2.52 RCC_CFGR_PPREG_b1

```
#define RCC_CFGR_PPREG_b1 14
```

Definition at line [278](#) of file [MRCC_private.h](#).

7.50.2.53 RCC_CFGR_PPREG_b2

```
#define RCC_CFGR_PPREG_b2 15
```

Definition at line [279](#) of file [MRCC_private.h](#).

7.50.2.54 RCC_CFGR_PPREG1_b0

```
#define RCC_CFGR_PPREG1_b0 10
```

Definition at line [280](#) of file [MRCC_private.h](#).

7.50.2.55 RCC_CFGR_PPREG1_b1

```
#define RCC_CFGR_PPREG1_b1 11
```

Definition at line [281](#) of file [MRCC_private.h](#).

7.50.2.56 RCC_CFGR_PPREG1_b2

```
#define RCC_CFGR_PPREG1_b2 12
```

Definition at line [282](#) of file [MRCC_private.h](#).

7.50.2.57 RCC_CFGR_HPRE_b0

```
#define RCC_CFGR_HPRE_b0 4
```

Definition at line [283](#) of file [MRCC_private.h](#).

7.50.2.58 RCC_CFGR_HPRE_b1

```
#define RCC_CFGR_HPRE_b1 5
```

Definition at line [284](#) of file [MRCC_private.h](#).

7.50.2.59 RCC_CFGR_HPRE_b2

```
#define RCC_CFGR_HPRE_b2 6
```

Definition at line [285](#) of file [MRCC_private.h](#).

7.50.2.60 RCC_CFGR_HPRE_b3

```
#define RCC_CFGR_HPRE_b3 7
```

Definition at line [286](#) of file [MRCC_private.h](#).

7.50.2.61 RCC_CFGR_SWS_b0

```
#define RCC_CFGR_SWS_b0 2
```

Definition at line [287](#) of file [MRCC_private.h](#).

7.50.2.62 RCC_CFGR_SWS_b1

```
#define RCC_CFGR_SWS_b1 3
```

Definition at line [288](#) of file [MRCC_private.h](#).

7.50.2.63 RCC_CFGR_SW_b0

```
#define RCC_CFGR_SW_b0 0
```

Definition at line [289](#) of file [MRCC_private.h](#).

7.50.2.64 RCC_CFGR_SW_b1

```
#define RCC_CFGR_SW_b1 1
```

Definition at line 290 of file [MRCC_private.h](#).

7.50.2.65 RCC_AHB1ENR_DMA2EN

```
#define RCC_AHB1ENR_DMA2EN 22
```

Definition at line 293 of file [MRCC_private.h](#).

7.50.2.66 RCC_AHB1ENR_DMA1EN

```
#define RCC_AHB1ENR_DMA1EN 21
```

Definition at line 294 of file [MRCC_private.h](#).

7.50.2.67 RCC_AHB1ENR_CRCEN

```
#define RCC_AHB1ENR_CRCEN 12
```

Definition at line 295 of file [MRCC_private.h](#).

7.50.2.68 RCC_AHB1ENR_GPIOHEN

```
#define RCC_AHB1ENR_GPIOHEN 7
```

Definition at line 296 of file [MRCC_private.h](#).

7.50.2.69 RCC_AHB1ENR_GPIOEEN

```
#define RCC_AHB1ENR_GPIOEEN 4
```

Definition at line 297 of file [MRCC_private.h](#).

7.50.2.70 RCC_AHB1ENR_GPIODEN

```
#define RCC_AHB1ENR_GPIODEN 3
```

Definition at line [298](#) of file [MRCC_private.h](#).

7.50.2.71 RCC_AHB1ENR_GPIOCEN

```
#define RCC_AHB1ENR_GPIOCEN 2
```

Definition at line [299](#) of file [MRCC_private.h](#).

7.50.2.72 RCC_AHB1ENR_GPIOBEN

```
#define RCC_AHB1ENR_GPIOBEN 1
```

Definition at line [300](#) of file [MRCC_private.h](#).

7.50.2.73 RCC_AHB1ENR_GPIOAEN

```
#define RCC_AHB1ENR_GPIOAEN 0
```

Definition at line [301](#) of file [MRCC_private.h](#).

7.50.2.74 RCC_AHB2ENR_OTGFSEN

```
#define RCC_AHB2ENR_OTGFSEN 7
```

Definition at line [304](#) of file [MRCC_private.h](#).

7.50.2.75 RCC_APB1ENR_PWREN

```
#define RCC_APB1ENR_PWREN 28
```

Definition at line [307](#) of file [MRCC_private.h](#).

7.50.2.76 RCC_APB1ENR_I2C3EN

```
#define RCC_APB1ENR_I2C3EN 23
```

Definition at line 308 of file [MRCC_private.h](#).

7.50.2.77 RCC_APB1ENR_I2C2EN

```
#define RCC_APB1ENR_I2C2EN 22
```

Definition at line 309 of file [MRCC_private.h](#).

7.50.2.78 RCC_APB1ENR_I2C1EN

```
#define RCC_APB1ENR_I2C1EN 21
```

Definition at line 310 of file [MRCC_private.h](#).

7.50.2.79 RCC_APB1ENR_USART2EN

```
#define RCC_APB1ENR_USART2EN 17
```

Definition at line 311 of file [MRCC_private.h](#).

7.50.2.80 RCC_APB1ENR_SPI3EN

```
#define RCC_APB1ENR_SPI3EN 15
```

Definition at line 312 of file [MRCC_private.h](#).

7.50.2.81 RCC_APB1ENR_SPI2EN

```
#define RCC_APB1ENR_SPI2EN 14
```

Definition at line 313 of file [MRCC_private.h](#).

7.50.2.82 RCC_APB1ENR_WWDGEN

```
#define RCC_APB1ENR_WWDGEN 11
```

Definition at line 314 of file [MRCC_private.h](#).

7.50.2.83 RCC_APB1ENR_TIM5EN

```
#define RCC_APB1ENR_TIM5EN 3
```

Definition at line 315 of file [MRCC_private.h](#).

7.50.2.84 RCC_APB1ENR_TIM4EN

```
#define RCC_APB1ENR_TIM4EN 2
```

Definition at line 316 of file [MRCC_private.h](#).

7.50.2.85 RCC_APB1ENR_TIM3EN

```
#define RCC_APB1ENR_TIM3EN 1
```

Definition at line 317 of file [MRCC_private.h](#).

7.50.2.86 RCC_APB1ENR_TIM2EN

```
#define RCC_APB1ENR_TIM2EN 0
```

Definition at line 318 of file [MRCC_private.h](#).

7.50.2.87 RCC_APB2ENR_TIM11EN

```
#define RCC_APB2ENR_TIM11EN 18
```

Definition at line 321 of file [MRCC_private.h](#).

7.50.2.88 RCC_APB2ENR_TIM10EN

```
#define RCC_APB2ENR_TIM10EN 17
```

Definition at line 322 of file [MRCC_private.h](#).

7.50.2.89 RCC_APB2ENR_TIM9EN

```
#define RCC_APB2ENR_TIM9EN 16
```

Definition at line 323 of file [MRCC_private.h](#).

7.50.2.90 RCC_APB2ENR_SYSCFGEN

```
#define RCC_APB2ENR_SYSCFGEN 14
```

Definition at line 324 of file [MRCC_private.h](#).

7.50.2.91 RCC_APB2ENR_SPI4EN

```
#define RCC_APB2ENR_SPI4EN 13
```

Definition at line 325 of file [MRCC_private.h](#).

7.50.2.92 RCC_APB2ENR_SPI1EN

```
#define RCC_APB2ENR_SPI1EN 12
```

Definition at line 326 of file [MRCC_private.h](#).

7.50.2.93 RCC_APB2ENR_SDIOEN

```
#define RCC_APB2ENR_SDIOEN 11
```

Definition at line 327 of file [MRCC_private.h](#).

7.50.2.94 RCC_APB2ENR_ADC1EN

```
#define RCC_APB2ENR_ADC1EN 8
```

Definition at line 328 of file [MRCC_private.h](#).

7.50.2.95 RCC_APB2ENR_USART6EN

```
#define RCC_APB2ENR_USART6EN 5
```

Definition at line 329 of file [MRCC_private.h](#).

7.50.2.96 RCC_APB2ENR_USART1EN

```
#define RCC_APB2ENR_USART1EN 4
```

Definition at line 330 of file [MRCC_private.h](#).

7.50.2.97 RCC_APB2ENR_TIM1EN

```
#define RCC_APB2ENR_TIM1EN 0
```

Definition at line 331 of file [MRCC_private.h](#).

7.50.2.98 RCC_AHB1LPENR_FLITFLPEN

```
#define RCC_AHB1LPENR_FLITFLPEN 15
```

Definition at line 333 of file [MRCC_private.h](#).

7.50.2.99 ENABLE

```
#define ENABLE 1
```

Definition at line 340 of file [MRCC_private.h](#).

7.50.2.100 DISABLE

```
#define DISABLE 2
```

Definition at line 341 of file [MRCC_private.h](#).

7.50.2.101 BYBASED

```
#define BYBASED 1
```

Definition at line 343 of file [MRCC_private.h](#).

7.50.2.102 NOTBYBASED

```
#define NOTBYBASED 2
```

Definition at line 344 of file [MRCC_private.h](#).

7.50.2.103 SYSCLK

```
#define SYSCLK 1
```

Definition at line 346 of file [MRCC_private.h](#).

7.50.2.104 PLLI2SCLK

```
#define PLLI2SCLK 5
```

Definition at line 347 of file [MRCC_private.h](#).

7.50.2.105 HSE

```
#define HSE 3
```

Definition at line 348 of file [MRCC_private.h](#).

7.50.2.106 PLLCLK [1/2]

```
#define PLLCLK 4
```

Definition at line 362 of file [MRCC_private.h](#).

7.50.2.107 NoDivision

```
#define NoDivision 1
```

Definition at line 351 of file [MRCC_private.h](#).

7.50.2.108 DivisionBy2

```
#define DivisionBy2 2
```

Definition at line 352 of file [MRCC_private.h](#).

7.50.2.109 DivisionBy3

```
#define DivisionBy3 3
```

Definition at line 353 of file [MRCC_private.h](#).

7.50.2.110 DivisionBy4

```
#define DivisionBy4 4
```

Definition at line 354 of file [MRCC_private.h](#).

7.50.2.111 DivisionBy5

```
#define DivisionBy5 5
```

Definition at line 355 of file [MRCC_private.h](#).

7.50.2.112 I2S_CKIN

```
#define I2S_CKIN 2
```

Definition at line 358 of file [MRCC_private.h](#).

7.50.2.113 HSI

```
#define HSI 1
```

Definition at line 360 of file [MRCC_private.h](#).

7.50.2.114 LSE

```
#define LSE 2
```

Definition at line 361 of file [MRCC_private.h](#).

7.50.2.115 PLLCLK [2/2]

```
#define PLLCLK 4
```

Definition at line 362 of file [MRCC_private.h](#).

7.50.2.116 SYSCLKby2

```
#define SYSCLKby2 2
```

Definition at line 365 of file [MRCC_private.h](#).

7.50.2.117 SYSCLKby4

```
#define SYSCLKby4 3
```

Definition at line 366 of file [MRCC_private.h](#).

7.50.2.118 SYSCLKby8

```
#define SYSCLKby8 4
```

Definition at line [367](#) of file [MRCC_private.h](#).

7.50.2.119 SYSCLKby16

```
#define SYSCLKby16 5
```

Definition at line [368](#) of file [MRCC_private.h](#).

7.50.2.120 SYSCLKby64

```
#define SYSCLKby64 6
```

Definition at line [369](#) of file [MRCC_private.h](#).

7.50.2.121 SYSCLKby128

```
#define SYSCLKby128 7
```

Definition at line [370](#) of file [MRCC_private.h](#).

7.50.2.122 SYSCLKby256

```
#define SYSCLKby256 8
```

Definition at line [371](#) of file [MRCC_private.h](#).

7.50.2.123 SYSCLKby512

```
#define SYSCLKby512 9
```

Definition at line [372](#) of file [MRCC_private.h](#).

7.50.2.124 NoCLK0

```
#define NoCLK0 1
```

Definition at line 375 of file [MRCC_private.h](#).

7.50.2.125 NoCLK1

```
#define NoCLK1 2
```

Definition at line 376 of file [MRCC_private.h](#).

7.50.2.126 HSEby2

```
#define HSEby2 3
```

Definition at line 377 of file [MRCC_private.h](#).

7.50.2.127 HSEby3

```
#define HSEby3 4
```

Definition at line 378 of file [MRCC_private.h](#).

7.50.2.128 HSEby4

```
#define HSEby4 5
```

Definition at line 379 of file [MRCC_private.h](#).

7.50.2.129 HSEby5

```
#define HSEby5 6
```

Definition at line 380 of file [MRCC_private.h](#).

7.50.2.130 HSEby6

```
#define HSEby6 7
```

Definition at line 381 of file [MRCC_private.h](#).

7.50.2.131 HSEby7

```
#define HSEby7 8
```

Definition at line 382 of file [MRCC_private.h](#).

7.50.2.132 HSEby8

```
#define HSEby8 9
```

Definition at line 383 of file [MRCC_private.h](#).

7.50.2.133 HSEby9

```
#define HSEby9 10
```

Definition at line 384 of file [MRCC_private.h](#).

7.50.2.134 HSEby10

```
#define HSEby10 11
```

Definition at line 385 of file [MRCC_private.h](#).

7.50.2.135 HSEby11

```
#define HSEby11 12
```

Definition at line 386 of file [MRCC_private.h](#).

7.50.2.136 HSEby12

```
#define HSEby12 13
```

Definition at line [387](#) of file [MRCC_private.h](#).

7.50.2.137 HSEby13

```
#define HSEby13 14
```

Definition at line [388](#) of file [MRCC_private.h](#).

7.50.2.138 HSEby14

```
#define HSEby14 15
```

Definition at line [389](#) of file [MRCC_private.h](#).

7.50.2.139 HSEby15

```
#define HSEby15 16
```

Definition at line [390](#) of file [MRCC_private.h](#).

7.50.2.140 HSEby16

```
#define HSEby16 17
```

Definition at line [391](#) of file [MRCC_private.h](#).

7.50.2.141 HSEby17

```
#define HSEby17 18
```

Definition at line [392](#) of file [MRCC_private.h](#).

7.50.2.142 HSEby18

```
#define HSEby18 19
```

Definition at line [393](#) of file [MRCC_private.h](#).

7.50.2.143 HSEby19

```
#define HSEby19 20
```

Definition at line [394](#) of file [MRCC_private.h](#).

7.50.2.144 HSEby20

```
#define HSEby20 21
```

Definition at line [395](#) of file [MRCC_private.h](#).

7.50.2.145 HSEby21

```
#define HSEby21 22
```

Definition at line [396](#) of file [MRCC_private.h](#).

7.50.2.146 HSEby22

```
#define HSEby22 23
```

Definition at line [397](#) of file [MRCC_private.h](#).

7.50.2.147 HSEby23

```
#define HSEby23 24
```

Definition at line [398](#) of file [MRCC_private.h](#).

7.50.2.148 HSEby24

```
#define HSEby24 25
```

Definition at line [399](#) of file [MRCC_private.h](#).

7.50.2.149 HSEby25

```
#define HSEby25 26
```

Definition at line [400](#) of file [MRCC_private.h](#).

7.50.2.150 HSEby26

```
#define HSEby26 27
```

Definition at line [401](#) of file [MRCC_private.h](#).

7.50.2.151 HSEby27

```
#define HSEby27 28
```

Definition at line [402](#) of file [MRCC_private.h](#).

7.50.2.152 HSEby28

```
#define HSEby28 39
```

Definition at line [403](#) of file [MRCC_private.h](#).

7.50.2.153 HSEby29

```
#define HSEby29 30
```

Definition at line [404](#) of file [MRCC_private.h](#).

7.50.2.154 HSEby30

```
#define HSEby30 31
```

Definition at line [405](#) of file [MRCC_private.h](#).

7.50.2.155 HSEby31

```
#define HSEby31 32
```

Definition at line [406](#) of file [MRCC_private.h](#).

7.50.2.156 AHBby2

```
#define AHBby2 1
```

Definition at line [409](#) of file [MRCC_private.h](#).

7.50.2.157 AHBby4

```
#define AHBby4 2
```

Definition at line [410](#) of file [MRCC_private.h](#).

7.50.2.158 AHBby8

```
#define AHBby8 3
```

Definition at line [411](#) of file [MRCC_private.h](#).

7.50.2.159 AHBby16

```
#define AHBby16 4
```

Definition at line [412](#) of file [MRCC_private.h](#).

7.50.2.160 Equal_0

```
#define Equal_0 0
```

Definition at line 415 of file [MRCC_private.h](#).

7.50.2.161 Equal_1

```
#define Equal_1 1
```

Definition at line 416 of file [MRCC_private.h](#).

7.50.2.162 Equal_2

```
#define Equal_2 2
```

Definition at line 417 of file [MRCC_private.h](#).

7.50.2.163 Equal_3

```
#define Equal_3 3
```

Definition at line 418 of file [MRCC_private.h](#).

7.50.2.164 Equal_4

```
#define Equal_4 4
```

Definition at line 419 of file [MRCC_private.h](#).

7.50.2.165 Equal_5

```
#define Equal_5 5
```

Definition at line 420 of file [MRCC_private.h](#).

7.50.2.166 Equal_6

```
#define Equal_6 6
```

Definition at line [421](#) of file [MRCC_private.h](#).

7.50.2.167 Equal_7

```
#define Equal_7 7
```

Definition at line [422](#) of file [MRCC_private.h](#).

7.50.2.168 Equal_8

```
#define Equal_8 8
```

Definition at line [423](#) of file [MRCC_private.h](#).

7.50.2.169 Equal_9

```
#define Equal_9 9
```

Definition at line [424](#) of file [MRCC_private.h](#).

7.50.2.170 Equal_10

```
#define Equal_10 10
```

Definition at line [425](#) of file [MRCC_private.h](#).

7.50.2.171 Equal_11

```
#define Equal_11 11
```

Definition at line [426](#) of file [MRCC_private.h](#).

7.50.2.172 Equal_12

```
#define Equal_12 12
```

Definition at line [427](#) of file [MRCC_private.h](#).

7.50.2.173 Equal_13

```
#define Equal_13 13
```

Definition at line [428](#) of file [MRCC_private.h](#).

7.50.2.174 Equal_14

```
#define Equal_14 14
```

Definition at line [429](#) of file [MRCC_private.h](#).

7.50.2.175 Equal_15

```
#define Equal_15 15
```

Definition at line [430](#) of file [MRCC_private.h](#).

7.50.2.176 Equal_16

```
#define Equal_16 16
```

Definition at line [431](#) of file [MRCC_private.h](#).

7.50.2.177 Equal_17

```
#define Equal_17 17
```

Definition at line [432](#) of file [MRCC_private.h](#).

7.50.2.178 Equal_18

```
#define Equal_18 18
```

Definition at line [433](#) of file [MRCC_private.h](#).

7.50.2.179 Equal_19

```
#define Equal_19 19
```

Definition at line [434](#) of file [MRCC_private.h](#).

7.50.2.180 Equal_20

```
#define Equal_20 20
```

Definition at line [435](#) of file [MRCC_private.h](#).

7.50.2.181 Equal_21

```
#define Equal_21 21
```

Definition at line [436](#) of file [MRCC_private.h](#).

7.50.2.182 Equal_22

```
#define Equal_22 22
```

Definition at line [437](#) of file [MRCC_private.h](#).

7.50.2.183 Equal_23

```
#define Equal_23 23
```

Definition at line [438](#) of file [MRCC_private.h](#).

7.50.2.184 Equal_24

```
#define Equal_24 24
```

Definition at line [439](#) of file [MRCC_private.h](#).

7.50.2.185 Equal_25

```
#define Equal_25 25
```

Definition at line [440](#) of file [MRCC_private.h](#).

7.50.2.186 Equal_26

```
#define Equal_26 26
```

Definition at line [441](#) of file [MRCC_private.h](#).

7.50.2.187 Equal_27

```
#define Equal_27 27
```

Definition at line [442](#) of file [MRCC_private.h](#).

7.50.2.188 Equal_28

```
#define Equal_28 28
```

Definition at line [443](#) of file [MRCC_private.h](#).

7.50.2.189 Equal_29

```
#define Equal_29 29
```

Definition at line [444](#) of file [MRCC_private.h](#).

7.50.2.190 Equal_30

```
#define Equal_30 30
```

Definition at line [445](#) of file [MRCC_private.h](#).

7.50.2.191 Equal_31

```
#define Equal_31 31
```

Definition at line [446](#) of file [MRCC_private.h](#).

7.50.2.192 Equal_32

```
#define Equal_32 32
```

Definition at line [447](#) of file [MRCC_private.h](#).

7.50.2.193 Equal_33

```
#define Equal_33 33
```

Definition at line [448](#) of file [MRCC_private.h](#).

7.50.2.194 Equal_34

```
#define Equal_34 34
```

Definition at line [449](#) of file [MRCC_private.h](#).

7.50.2.195 Equal_35

```
#define Equal_35 35
```

Definition at line [450](#) of file [MRCC_private.h](#).

7.50.2.196 Equal_36

```
#define Equal_36 36
```

Definition at line [451](#) of file [MRCC_private.h](#).

7.50.2.197 Equal_37

```
#define Equal_37 37
```

Definition at line [452](#) of file [MRCC_private.h](#).

7.50.2.198 Equal_38

```
#define Equal_38 38
```

Definition at line [453](#) of file [MRCC_private.h](#).

7.50.2.199 Equal_39

```
#define Equal_39 39
```

Definition at line [454](#) of file [MRCC_private.h](#).

7.50.2.200 Equal_40

```
#define Equal_40 40
```

Definition at line [455](#) of file [MRCC_private.h](#).

7.50.2.201 Equal_41

```
#define Equal_41 41
```

Definition at line [456](#) of file [MRCC_private.h](#).

7.50.2.202 Equal_42

```
#define Equal_42 42
```

Definition at line [457](#) of file [MRCC_private.h](#).

7.50.2.203 Equal_43

```
#define Equal_43 43
```

Definition at line [458](#) of file [MRCC_private.h](#).

7.50.2.204 Equal_44

```
#define Equal_44 44
```

Definition at line [459](#) of file [MRCC_private.h](#).

7.50.2.205 Equal_45

```
#define Equal_45 45
```

Definition at line [460](#) of file [MRCC_private.h](#).

7.50.2.206 Equal_46

```
#define Equal_46 46
```

Definition at line [461](#) of file [MRCC_private.h](#).

7.50.2.207 Equal_47

```
#define Equal_47 47
```

Definition at line [462](#) of file [MRCC_private.h](#).

7.50.2.208 Equal_48

```
#define Equal_48 48
```

Definition at line [463](#) of file [MRCC_private.h](#).

7.50.2.209 Equal_49

```
#define Equal_49 49
```

Definition at line [464](#) of file [MRCC_private.h](#).

7.50.2.210 Equal_50

```
#define Equal_50 50
```

Definition at line [465](#) of file [MRCC_private.h](#).

7.50.2.211 Equal_51

```
#define Equal_51 51
```

Definition at line [466](#) of file [MRCC_private.h](#).

7.50.2.212 Equal_52

```
#define Equal_52 52
```

Definition at line [467](#) of file [MRCC_private.h](#).

7.50.2.213 Equal_53

```
#define Equal_53 53
```

Definition at line [468](#) of file [MRCC_private.h](#).

7.50.2.214 Equal_54

```
#define Equal_54 54
```

Definition at line [469](#) of file [MRCC_private.h](#).

7.50.2.215 Equal_55

```
#define Equal_55 55
```

Definition at line [470](#) of file [MRCC_private.h](#).

7.50.2.216 Equal_56

```
#define Equal_56 56
```

Definition at line [471](#) of file [MRCC_private.h](#).

7.50.2.217 Equal_57

```
#define Equal_57 57
```

Definition at line [472](#) of file [MRCC_private.h](#).

7.50.2.218 Equal_58

```
#define Equal_58 58
```

Definition at line [473](#) of file [MRCC_private.h](#).

7.50.2.219 Equal_59

```
#define Equal_59 59
```

Definition at line [474](#) of file [MRCC_private.h](#).

7.50.2.220 Equal_60

```
#define Equal_60 60
```

Definition at line 475 of file [MRCC_private.h](#).

7.50.2.221 Equal_61

```
#define Equal_61 61
```

Definition at line 476 of file [MRCC_private.h](#).

7.50.2.222 Equal_62

```
#define Equal_62 62
```

Definition at line 477 of file [MRCC_private.h](#).

7.50.2.223 Equal_63

```
#define Equal_63 63
```

Definition at line 478 of file [MRCC_private.h](#).

7.51 MRCC_private.h

[Go to the documentation of this file.](#)

```
00001
00009 /* Header file guard */
00010 #ifndef MCAL_RCC_MRCC_PRIVATE_H_
00011 #define MCAL_RCC_MRCC_PRIVATE_H_
00012
00013
00014
00015 /***** Peripherals declaration *****/
00016 /* Peripherals declaration */
00017 /***** *****/
00018
00024 typedef struct
00025 {
00026
00030     u32_t CR;
00031
00035     u32_t PLLCFGR;
00036
00040     u32_t CFGR;
00041
00045     u32_t CIR;
00046
00050     u32_t AHB1RSTR;
00051
00055     u32_t AHB2RSTR;
00056
```

```

00060     u32_t Reserved1 ;  

00061  

00065     u32_t Reserved2 ;  

00066  

00070     u32_t APB1RSTR ;  

00071  

00075     u32_t APB2RSTR ;  

00076  

00080     u32_t Reserved3 ;  

00081  

00085     u32_t Reserved4 ;  

00086  

00090     u32_t AHB1ENR ;  

00091  

00095     u32_t AHB2ENR ;  

00096  

00100     u32_t Reserved5 ;  

00101  

00105     u32_t Reserved6 ;  

00106  

00110     u32_t APB1ENR ;  

00111  

00115     u32_t APB2ENR ;  

00116  

00120     u32_t Reserved7 ;  

00121  

00125     u32_t Reserved8 ;  

00126  

00130     u32_t AHB1LPENR ;  

00131  

00135     u32_t AHB2LPENR ;  

00136  

00140     u32_t Reserved9 ;  

00141  

00145     u32_t Reserved10 ;  

00146  

00150     u32_t APB1LPENR ;  

00151  

00155     u32_t APB2LPENR ;  

00156  

00160     u32_t Reserved11 ;  

00161  

00165     u32_t Reserved12 ;  

00166  

00170     u32_t BDCR ;  

00171  

00175     u32_t CSR ;  

00176  

00180     u32_t Reserved13 ;  

00181  

00185     u32_t Reserved14 ;  

00186  

00190     u32_t SSCGR ;  

00191  

00195     u32_t PLLI2SCFGR ;  

00196  

00200     u32_t DCKCFGR ;  

00201  

00202  

00203 } RCC_MemoryMapType ;  

00204  

00205  

00211 #define RCC_BASE_ADDRESS 0x40023800  

00212  

00218 #define RCC ( volatile P2VAR(RCC_MemoryMapType) ) (RCC_BASE_ADDRESS) // Is a pointer to the struct.  

00219  

00220  

00221 /*****  

00222 /* Peripheral registers */  

00223 *****/  

00224  

00225 #define RCC_CR_PLLI2SRDY 27  

00226 #define RCC_CR_PLLI2SON 26  

00227 #define RCC_CR_PLLRDY 25  

00228 #define RCC_CR_PLLON 24  

00229 #define RCC_CR_CSSTON 19  

00230 #define RCC_CR_HSEBYP 18  

00231 #define RCC_CR_HSERTDY 17  

00232 #define RCC_CR_HSEON 16  

00233 #define RCC_CR_HSIRDY 1  

00234 #define RCC_CR_HSION 0  

00235  

00236  

00237 #define RCC_PLLCFGRLLQ_b0 24  

00238 #define RCC_PLLCFGRLLQ_b1 25  

00239 #define RCC_PLLCFGRLLQ_b2 26  

00240 #define RCC_PLLCFGRLLQ_b3 27

```

```
00241 #define RCC_PLLCFGRL_PLLSRC 22
00242 #define RCC_PLLCFGRL_PLLP_b0 16
00243 #define RCC_PLLCFGRL_PLLP_b1 17
00244 #define RCC_PLLCFGRL_PLLN_b0 6
00245 #define RCC_PLLCFGRL_PLLN_b1 7
00246 #define RCC_PLLCFGRL_PLLN_b2 8
00247 #define RCC_PLLCFGRL_PLLN_b3 9
00248 #define RCC_PLLCFGRL_PLLN_b4 10
00249 #define RCC_PLLCFGRL_PLLN_b5 11
00250 #define RCC_PLLCFGRL_PLLN_b6 12
00251 #define RCC_PLLCFGRL_PLLN_b7 13
00252 #define RCC_PLLCFGRL_PLLN_b8 14
00253 #define RCC_PLLCFGRL_PLLM_b0 0
00254 #define RCC_PLLCFGRL_PLLM_b1 1
00255 #define RCC_PLLCFGRL_PLLM_b2 2
00256 #define RCC_PLLCFGRL_PLLM_b3 3
00257 #define RCC_PLLCFGRL_PLLM_b4 4
00258 #define RCC_PLLCFGRL_PLLM_b5 5
00259
00260
00261 #define RCC_CFGR_MOC2_b0 30
00262 #define RCC_CFGR_MOC2_b1 31
00263 #define RCC_CFGR_MOC2PRE_b0 27
00264 #define RCC_CFGR_MOC2PRE_b1 28
00265 #define RCC_CFGR_MOC2PRE_b2 29
00266 #define RCC_CFGR_MOC1PRE_b0 24
00267 #define RCC_CFGR_MOC1PRE_b1 25
00268 #define RCC_CFGR_MOC1PRE_b2 26
00269 #define RCC_CFGR_I2SSRC 23
00270 #define RCC_CFGR_MOC1_b0 21
00271 #define RCC_CFGR_MOC1_b1 22
00272 #define RCC_CFGR_RTCPRE_b0 16
00273 #define RCC_CFGR_RTCPRE_b1 17
00274 #define RCC_CFGR_RTCPRE_b2 18
00275 #define RCC_CFGR_RTCPRE_b3 19
00276 #define RCC_CFGR_RTCPRE_b4 20
00277 #define RCC_CFGR_PPREG2_b0 13
00278 #define RCC_CFGR_PPREG2_b1 14
00279 #define RCC_CFGR_PPREG2_b2 15
00280 #define RCC_CFGR_PPREG1_b0 10
00281 #define RCC_CFGR_PPREG1_b1 11
00282 #define RCC_CFGR_PPREG1_b2 12
00283 #define RCC_CFGR_HPRE_b0 4
00284 #define RCC_CFGR_HPRE_b1 5
00285 #define RCC_CFGR_HPRE_b2 6
00286 #define RCC_CFGR_HPRE_b3 7
00287 #define RCC_CFGR_SWS_b0 2
00288 #define RCC_CFGR_SWS_b1 3
00289 #define RCC_CFGR_SW_b0 0
00290 #define RCC_CFGR_SW_b1 1
00291
00292
00293 #define RCC_AHB1ENR_DMA2EN 22
00294 #define RCC_AHB1ENR_DMA1EN 21
00295 #define RCC_AHB1ENR_CRCEN 12
00296 #define RCC_AHB1ENR_GPIOHEN 7
00297 #define RCC_AHB1ENR_GPIOEEN 4
00298 #define RCC_AHB1ENR_GPIODEN 3
00299 #define RCC_AHB1ENR_GPIOCEN 2
00300 #define RCC_AHB1ENR_GPIOBEN 1
00301 #define RCC_AHB1ENR_GPIOAEN 0
00302
00303
00304 #define RCC_AHB2ENR_OTGFSEN 7
00305
00306
00307 #define RCC_APB1ENR_PWREN 28
00308 #define RCC_APB1ENR_I2C3EN 23
00309 #define RCC_APB1ENR_I2C2EN 22
00310 #define RCC_APB1ENR_I2C1EN 21
00311 #define RCC_APB1ENR_USART2EN 17
00312 #define RCC_APB1ENR_SPI3EN 15
00313 #define RCC_APB1ENR_SPI2EN 14
00314 #define RCC_APB1ENR_WWDGEN 11
00315 #define RCC_APB1ENR_TIM5EN 3
00316 #define RCC_APB1ENR_TIM4EN 2
00317 #define RCC_APB1ENR_TIM3EN 1
00318 #define RCC_APB1ENR_TIM2EN 0
00319
00320
00321 #define RCC_APB2ENR_TIM11EN 18
00322 #define RCC_APB2ENR_TIM10EN 17
00323 #define RCC_APB2ENR_TIM9EN 16
00324 #define RCC_APB2ENR_SYSCFGEN 14
00325 #define RCC_APB2ENR_SPI4EN 13
00326 #define RCC_APB2ENR_SPI1EN 12
00327 #define RCC_APB2ENR_SDIOEN 11
```

```

00328 #define RCC_APB2ENR_ADC1EN      8
00329 #define RCC_APB2ENR_USART6EN    5
00330 #define RCC_APB2ENR_USART1EN    4
00331 #define RCC_APB2ENR_TIM1EN     0
00332
00333 #define RCC_AHB1LPENR_FLITFLPEN 15
00334
00335
00336 /***** Config.h Macros *****/
00337 /*                                         */
00338 /*****                                         *****/
00339
00340 #define ENABLE   1
00341 #define DISABLE  2
00342
00343 #define BYBASED   1
00344 #define NOTBYBASED 2
00345
00346 #define SYSCLK      1
00347 #define PLLI2SCLK   5
00348 #define HSE        3
00349 #define PLLCLK     4
00350
00351 #define NoDivision 1
00352 #define DivisionBy2 2
00353 #define DivisionBy3 3
00354 #define DivisionBy4 4
00355 #define DivisionBy5 5
00356
00357
00358 #define I2S_CKIN  2
00359
00360 #define HSI      1
00361 #define LSE      2
00362 #define PLLCLK   4
00363
00364 // AHB prescalers:
00365 #define SYSCLKby2  2  // (0b1000)
00366 #define SYSCLKby4  3  // (0b1001)
00367 #define SYSCLKby8  4  // (0b1010)
00368 #define SYSCLKby16 5  // (0b1011)
00369 #define SYSCLKby64 6  // (0b1100)
00370 #define SYSCLKby128 7 // (0b1101)
00371 #define SYSCLKby256 8 // (0b1110)
00372 #define SYSCLKby512 9 // (0b1111)
00373
00374
00375 #define NoCLK0   1
00376 #define NoCLK1   2
00377 #define HSEby2   3
00378 #define HSEby3   4
00379 #define HSEby4   5
00380 #define HSEby5   6
00381 #define HSEby6   7
00382 #define HSEby7   8
00383 #define HSEby8   9
00384 #define HSEby9   10
00385 #define HSEby10  11
00386 #define HSEby11  12
00387 #define HSEby12  13
00388 #define HSEby13  14
00389 #define HSEby14  15
00390 #define HSEby15  16
00391 #define HSEby16  17
00392 #define HSEby17  18
00393 #define HSEby18  19
00394 #define HSEby19  20
00395 #define HSEby20  21
00396 #define HSEby21  22
00397 #define HSEby22  23
00398 #define HSEby23  24
00399 #define HSEby24  25
00400 #define HSEby25  26
00401 #define HSEby26  27
00402 #define HSEby27  28
00403 #define HSEby28  39
00404 #define HSEby29  30
00405 #define HSEby30  31
00406 #define HSEby31  32
00407
00408
00409 #define AHBby2  1
00410 #define AHBby4  2
00411 #define AHBby8  3
00412 #define AHBby16 4
00413
00414

```

```

00415 #define Equal_0      0
00416 #define Equal_1      1
00417 #define Equal_2      2
00418 #define Equal_3      3
00419 #define Equal_4      4
00420 #define Equal_5      5
00421 #define Equal_6      6
00422 #define Equal_7      7
00423 #define Equal_8      8
00424 #define Equal_9      9
00425 #define Equal_10     10
00426 #define Equal_11     11
00427 #define Equal_12     12
00428 #define Equal_13     13
00429 #define Equal_14     14
00430 #define Equal_15     15
00431 #define Equal_16     16
00432 #define Equal_17     17
00433 #define Equal_18     18
00434 #define Equal_19     19
00435 #define Equal_20     20
00436 #define Equal_21     21
00437 #define Equal_22     22
00438 #define Equal_23     23
00439 #define Equal_24     24
00440 #define Equal_25     25
00441 #define Equal_26     26
00442 #define Equal_27     27
00443 #define Equal_28     28
00444 #define Equal_29     29
00445 #define Equal_30     30
00446 #define Equal_31     31
00447 #define Equal_32     32
00448 #define Equal_33     33
00449 #define Equal_34     34
00450 #define Equal_35     35
00451 #define Equal_36     36
00452 #define Equal_37     37
00453 #define Equal_38     38
00454 #define Equal_39     39
00455 #define Equal_40     40
00456 #define Equal_41     41
00457 #define Equal_42     42
00458 #define Equal_43     43
00459 #define Equal_44     44
00460 #define Equal_45     45
00461 #define Equal_46     46
00462 #define Equal_47     47
00463 #define Equal_48     48
00464 #define Equal_49     49
00465 #define Equal_50     50
00466 #define Equal_51     51
00467 #define Equal_52     52
00468 #define Equal_53     53
00469 #define Equal_54     54
00470 #define Equal_55     55
00471 #define Equal_56     56
00472 #define Equal_57     57
00473 #define Equal_58     58
00474 #define Equal_59     59
00475 #define Equal_60     60
00476 #define Equal_61     61
00477 #define Equal_62     62
00478 #define Equal_63     63
00479
00480
00481
00482
00483
00484 #endif /* MCAL_RCC_MRCC_PRIVATE_H */

```

7.52 COTS/MCAL/RCC/MRCC_program.c File Reference

This file contains the source code of the interfacing for the RCC module.

```

#include "../../LIB/LSTD_TYPES.h"
#include "../../LIB/LSTD_COMPILER.h"
#include "../../LIB/LSTD_VALUES.h"
#include "../../LIB/LSTD_BITMATH.h"

```

```
#include "MRCC_private.h"
#include "MRCC_interface.h"
#include "MRCC_config.h"
```

Functions

- void [MRCC_vInit](#) (void)
Initialize the RCC with a certain configurations.
- void [MRCC_vEnablePeriphralCLK](#) ([u32_t](#) A_u32BusID, [u32_t](#) A_u32PeriphralID)
- void [MRCC_vDisablePeriphralCLK](#) ([u32_t](#) A_u32BusID, [u32_t](#) A_u32PeriphralID)

7.52.1 Detailed Description

This file contains the source code of the interfacing for the RCC module.

Author

Ali El Bana & Mo Alaa

Version

2.0

Date

11/9/2022

Definition in file [MRCC_program.c](#).

7.52.2 Function Documentation

7.52.2.1 MRCC_vInit()

```
void MRCC_vInit (
    void )
```

Initialize the RCC with a certain configurations.

Definition at line 29 of file [MRCC_program.c](#).

```
00030 {
00031
00032     // PLLI2S (ON/OFF).
00033 #if PLLI2S == ENABLE
00034
00035     SET_BIT( RCC->CR, RCC_CR_PLLI2SON ) ;
00036
00037 #elif PLLI2S == DISABLE
00038
00039     CLR_BIT( RCC->CR, RCC_CR_PLLI2SON ) ;
00040
```

```
00041 #endif
00042
00043
00044 // PLL (ON/OFF).
00045 #if PLL == ENABLE
00046     SET_BIT( RCC->CR, RCC_CR_PLLON ) ;
00048
00049 #elif PLL == DISABLE
00050     CLR_BIT( RCC->CR, RCC_CR_PLLON ) ;
00052
00053 #endif
00054
00055
00056 // CSS (ON/OFF).
00057 #if CSS == ENABLE
00058     SET_BIT( RCC->CR, RCC_CR_CSSON ) ;
00060
00061 #elif CSS == DISABLE
00062     CLR_BIT( RCC->CR, RCC_CR_CSSON ) ;
00064
00065 #endif
00066
00067
00068 // HSEBYP.
00069 #if HSEBYP == BYBASED
00070     SET_BIT( RCC->CR, RCC_CR_HSEBYP ) ;
00072
00073 #elif HSEBYP == NOTBYBASED
00074     CLR_BIT( RCC->CR, RCC_CR_HSEBYP ) ;
00076
00077 #endif
00078
00079
00080 // Select CLK switch (HSI/HSE/PLL).
00081 #if SW == HSI
00082
00083     CLR_BIT( RCC->CFGR, RCC_CFGR_SW_b0 ) ;
00084     CLR_BIT( RCC->CFGR, RCC_CFGR_SW_b1 ) ;
00085
00086 #elif SW == HSE
00087
00088     SET_BIT( RCC->CFGR, RCC_CFGR_SW_b0 ) ;
00089     CLR_BIT( RCC->CFGR, RCC_CFGR_SW_b1 ) ;
00090
00091 #elif SW == PLLCLK
00092
00093     CLR_BIT( RCC->CFGR, RCC_CFGR_SW_b0 ) ;
00094     SET_BIT( RCC->CFGR, RCC_CFGR_SW_b1 ) ;
00095
00096 #endif
00097
00098 // Select CLK switch status (HSI/HSE/PLL).
00099 #if SWS == HSI
00100
00101     CLR_BIT( RCC->CFGR, RCC_CFGR_SWS_b0 ) ;
00102     CLR_BIT( RCC->CFGR, RCC_CFGR_SWS_b1 ) ;
00103
00104 #elif SWS == HSE
00105
00106     SET_BIT( RCC->CFGR, RCC_CFGR_SWS_b0 ) ;
00107     CLR_BIT( RCC->CFGR, RCC_CFGR_SWS_b1 ) ;
00108
00109 #elif SWS == PLLCLK
00110
00111     CLR_BIT( RCC->CFGR, RCC_CFGR_SWS_b0 ) ;
00112     SET_BIT( RCC->CFGR, RCC_CFGR_SWS_b1 ) ;
00113
00114 #endif
00115
00116
00117 // MCO2 selection:
00118 #if MCO2 == SYSCLK
00119
00120     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2_b0 ) ;
00121     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2_b1 ) ;
00122
00123 #elif MCO2 == PLLI2SCLK
00124
00125     SET_BIT( RCC->CFGR, RCC_CFGR_MOC2_b0 ) ;
00126     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2_b1 ) ;
00127
```

```

00128 #elif MCO2 == HSE
00129     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2_b0 ) ;
00130     SET_BIT( RCC->CFGR, RCC_CFGR_MOC2_b1 ) ;
00132
00133 #elif MCO2 == PLLCLK
00134
00135     SET_BIT( RCC->CFGR, RCC_CFGR_MOC2_b0 ) ;
00136     SET_BIT( RCC->CFGR, RCC_CFGR_MOC2_b1 ) ;
00137
00138 #endif
00139
00140
00141     // MCO2 prescaler:
00142 #if MCO2PRE == NoDivision
00143
00144     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b0 ) ;
00145     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b1 ) ;
00146     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b2 ) ;
00147
00148 #elif MCO2PRE == DivisionBy2
00149
00150     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b0 ) ;
00151     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b1 ) ;
00152     SET_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b2 ) ;
00153
00154 #elif MCO2PRE == DivisionBy3
00155
00156     SET_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b0 ) ;
00157     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b1 ) ;
00158     SET_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b2 ) ;
00159
00160 #elif MCO2PRE == DivisionBy4
00161
00162     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b0 ) ;
00163     SET_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b1 ) ;
00164     SET_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b2 ) ;
00165
00166 #elif MCO2PRE == DivisionBy5
00167
00168     SET_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b0 ) ;
00169     SET_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b1 ) ;
00170     SET_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b2 ) ;
00171
00172 #endif
00173
00174
00175     // MCO1 selection:
00176 #if MCO1 == HSI
00177
00178     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1_b0 ) ;
00179     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1_b1 ) ;
00180
00181 #elif MCO1 == LSE
00182
00183     SET_BIT( RCC->CFGR, RCC_CFGR_MOC1_b0 ) ;
00184     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1_b1 ) ;
00185
00186 #elif MCO1 == HSE
00187
00188     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1_b0 ) ;
00189     SET_BIT( RCC->CFGR, RCC_CFGR_MOC1_b1 ) ;
00190
00191 #elif MCO1 == PLLCLK
00192
00193     SET_BIT( RCC->CFGR, RCC_CFGR_MOC1_b0 ) ;
00194     SET_BIT( RCC->CFGR, RCC_CFGR_MOC1_b1 ) ;
00195
00196 #endif
00197
00198
00199     // MCO1 prescaler:
00200 #if MCO1PRE == NoDivision
00201
00202     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b0 ) ;
00203     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b1 ) ;
00204     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b2 ) ;
00205
00206 #elif MCO1PRE == DivisionBy2
00207
00208     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b0 ) ;
00209     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b1 ) ;
00210     SET_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b2 ) ;
00211
00212 #elif MCO1PRE == DivisionBy3
00213
00214     SET_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b0 ) ;

```

```

00215     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b1 ) ;
00216     SET_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b2 ) ;
00217
00218 #elif MCO1PRE == DivisionBy4
00219     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b0 ) ;
00220     SET_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b1 ) ;
00221     SET_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b2 ) ;
00222
00223 #elif MCO1PRE == DivisionBy5
00224     SET_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b0 ) ;
00225     SET_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b1 ) ;
00226     SET_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b2 ) ;
00227
00228 #endif
00229
00230 // AHB prescalers:
00231
00232 #if HPRE == NoDivision
00233
00234 #if HPRE == NoDivision
00235     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00236     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00237     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00238     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00239
00240 #elif HPRE == SYSCLKby2
00241     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00242     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00243     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00244     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00245
00246 #elif HPRE == SYSCLKby4
00247     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00248     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00249     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00250     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00251
00252 #elif HPRE == SYSCLKby8
00253     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00254     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00255     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00256     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00257
00258 #elif HPRE == SYSCLKby16
00259     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00260     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00261     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00262     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00263
00264 #elif HPRE == SYSCLKby64
00265     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00266     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00267     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00268     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00269
00270 #elif HPRE == SYSCLKby128
00271     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00272     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00273     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00274     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00275
00276 #elif HPRE == SYSCLKby256
00277     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00278     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00279     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00280     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00281
00282 #elif HPRE == SYSCLKby512
00283     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00284     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00285     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00286     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00287
00288 #endif
00289
00290 // APB1 prescalers:
00291 #if PPRE1 == NoDivision
00292     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00293     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00294     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00295     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00296
00297 #endif
00298
00299
00300 // APB2 prescalers:
00301 #if PPRE2 == NoDivision

```

```

00302
00303     CLR_BIT( RCC->CFGR, RCC_CFGR_PPREG_b0 ) ;
00304     CLR_BIT( RCC->CFGR, RCC_CFGR_PPREG_b1 ) ;
00305     CLR_BIT( RCC->CFGR, RCC_CFGR_PPREG_b2 ) ;
00306
00307 #elif PPREG == AHBby2
00308
00309     CLR_BIT( RCC->CFGR, RCC_CFGR_PPREG_b0 ) ;
00310     CLR_BIT( RCC->CFGR, RCC_CFGR_PPREG_b1 ) ;
00311     SET_BIT( RCC->CFGR, RCC_CFGR_PPREG_b2 ) ;
00312
00313 #elif PPREG == AHBby4
00314
00315     SET_BIT( RCC->CFGR, RCC_CFGR_PPREG_b0 ) ;
00316     CLR_BIT( RCC->CFGR, RCC_CFGR_PPREG_b1 ) ;
00317     SET_BIT( RCC->CFGR, RCC_CFGR_PPREG_b2 ) ;
00318
00319 #elif PPREG == AHBby8
00320
00321     CLR_BIT( RCC->CFGR, RCC_CFGR_PPREG_b0 ) ;
00322     SET_BIT( RCC->CFGR, RCC_CFGR_PPREG_b1 ) ;
00323     SET_BIT( RCC->CFGR, RCC_CFGR_PPREG_b2 ) ;
00324
00325 #elif PPREG == AHBby16
00326
00327     SET_BIT( RCC->CFGR, RCC_CFGR_PPREG_b0 ) ;
00328     SET_BIT( RCC->CFGR, RCC_CFGR_PPREG_b1 ) ;
00329     SET_BIT( RCC->CFGR, RCC_CFGR_PPREG_b2 ) ;
00330
00331 #endif
00332
00333 // APB2 prescalers:
00334 #if PPREG2 == NoDivision
00335
00336     CLR_BIT( RCC->CFGR, RCC_CFGR_PPREG2_b0 ) ;
00337     CLR_BIT( RCC->CFGR, RCC_CFGR_PPREG2_b1 ) ;
00338     CLR_BIT( RCC->CFGR, RCC_CFGR_PPREG2_b2 ) ;
00339
00340 #elif PPREG2 == AHBby2
00341
00342     CLR_BIT( RCC->CFGR, RCC_CFGR_PPREG2_b0 ) ;
00343     CLR_BIT( RCC->CFGR, RCC_CFGR_PPREG2_b1 ) ;
00344     SET_BIT( RCC->CFGR, RCC_CFGR_PPREG2_b2 ) ;
00345
00346 #elif PPREG2 == AHBby4
00347
00348     SET_BIT( RCC->CFGR, RCC_CFGR_PPREG2_b0 ) ;
00349     CLR_BIT( RCC->CFGR, RCC_CFGR_PPREG2_b1 ) ;
00350     SET_BIT( RCC->CFGR, RCC_CFGR_PPREG2_b2 ) ;
00351
00352 #elif PPREG2 == AHBby8
00353
00354     CLR_BIT( RCC->CFGR, RCC_CFGR_PPREG2_b0 ) ;
00355     SET_BIT( RCC->CFGR, RCC_CFGR_PPREG2_b1 ) ;
00356     SET_BIT( RCC->CFGR, RCC_CFGR_PPREG2_b2 ) ;
00357
00358 #elif PPREG2 == AHBby16
00359
00360     SET_BIT( RCC->CFGR, RCC_CFGR_PPREG2_b0 ) ;
00361     SET_BIT( RCC->CFGR, RCC_CFGR_PPREG2_b1 ) ;
00362     SET_BIT( RCC->CFGR, RCC_CFGR_PPREG2_b2 ) ;
00363
00364 #endif
00365
00366
00367 // PLL configurations:
00368
00369
00370
00371 // Enable the selected CLK (HSI ON/HSE ON/PLL ON): //
00372 // HSE (ON/OFF).
00373 #if HSE_EN == ENABLE
00374
00375     SET_BIT( RCC->CR, RCC_CR_HSEON ) ;
00376
00377 #elif HSE_EN == DISABLE
00378
00379     CLR_BIT( RCC->CR, RCC_CR_HSEON ) ;
00380
00381 #endif
00382
00383
00384 // HSI (ON/OFF).
00385 #if HSI_EN == ENABLE
00386
00387     SET_BIT( RCC->CR, RCC_CR_HSION ) ;
00388

```

```

00389 #elif HST_EN == DISABLE
00390     CLR_BIT( RCC->CR, RCC_CR_HSION ) ;
00392
00393 #endif
00394
00395 }

```

References [CLR_BIT](#), [RCC](#), [RCC_CFGR_HPRE_b0](#), [RCC_CFGR_HPRE_b1](#), [RCC_CFGR_HPRE_b2](#), [RCC_CFGR_HPRE_b3](#), [RCC_CFGR_MOC1_b0](#), [RCC_CFGR_MOC1_b1](#), [RCC_CFGR_MOC1PRE_b0](#), [RCC_CFGR_MOC1PRE_b1](#), [RCC_CFGR_MOC1PRE_b2](#), [RCC_CFGR_MOC2_b0](#), [RCC_CFGR_MOC2_b1](#), [RCC_CFGR_MOC2PRE_b0](#), [RCC_CFGR_MOC2PRE_b1](#), [RCC_CFGR_MOC2PRE_b2](#), [RCC_CFGR_PPREG1_b0](#), [RCC_CFGR_PPREG1_b1](#), [RCC_CFGR_PPREG1_b2](#), [RCC_CFGR_PPREG2_b0](#), [RCC_CFGR_PPREG2_b1](#), [RCC_CFGR_PPREG2_b2](#), [RCC_CFGR_SW_b0](#), [RCC_CFGR_SW_b1](#), [RCC_CFGR_SWS_b0](#), [RCC_CFGR_SWS_b1](#), [RCC_CR_CSSEN](#), [RCC_CR_HSEBYP](#), [RCC_CR_HSEON](#), [RCC_CR_HSION](#), [RCC_CR_PLLI2SON](#), [RCC_CR_PLLON](#), and [SET_BIT](#).

7.52.2.2 MRCC_vEnablePeriphralCLK()

```

void MRCC_vEnablePeriphralCLK (
    u32_t A_u32BusID,
    u32_t A_u32PeriphralID )

```

Definition at line 400 of file [MRCC_program.c](#).

```

00401 {
00402
00403     switch( A_u32BusID )
00404     {
00405
00406         case RCC_AHB1 :
00407             SET_BIT( RCC->AHB1ENR, A_u32PeriphralID ) ;
00408             break ;
00409
00410         case RCC_AHB2 :
00411             SET_BIT( RCC->AHB2ENR, A_u32PeriphralID ) ;
00412             break ;
00413
00414         case RCC_APB1 :
00415             SET_BIT( RCC->APB1ENR, A_u32PeriphralID ) ;
00416             break ;
00417
00418         case RCC_APB2 :
00419             SET_BIT( RCC->APB2ENR, A_u32PeriphralID ) ;
00420             break ;
00421
00422         case RCC_AHB1LPENR:
00423             SET_BIT( RCC->AHB1LPENR, A_u32PeriphralID ) ;
00424             break ;
00425
00426         default:
00427             // Error wrong Bus ID
00428             break ;
00429
00430     }
00431
00432 }
00433
00434 }
00435
00436 }
00437
00438 }
00439
00440 }
00441
00442 }
00443
00444 }
00445
00446 }
00447
00448 }

```

References [RCC](#), [RCC_AHB1](#), [RCC_AHB1LPENR](#), [RCC_AHB2](#), [RCC_APB1](#), [RCC_APB2](#), and [SET_BIT](#).

7.52.2.3 MRCC_vDisablePeriphralCLK()

```
void MRCC_vDisablePeriphralCLK (
    u32_t A_u32BusID,
    u32_t A_u32PeriphralID )
```

Definition at line 453 of file [MRCC_program.c](#).

```
00454 {
00455
00456     switch( A_u32BusID )
00457     {
00458
00459         case RCC_AHB1 :
00460             CLR_BIT( RCC->AHB1ENR, A_u32PeriphralID ) ;
00461             break ;
00462
00463         case RCC_AHB2 :
00464             CLR_BIT( RCC->AHB2ENR, A_u32PeriphralID ) ;
00465             break ;
00466
00467         case RCC_APB1 :
00468             CLR_BIT( RCC->APB1ENR, A_u32PeriphralID ) ;
00469             break ;
00470
00471         case RCC_APB2 :
00472             CLR_BIT( RCC->APB2ENR, A_u32PeriphralID ) ;
00473             break ;
00474
00475         case RCC_AHB1LPENR:
00476             CLR_BIT( RCC->AHB1LPENR, A_u32PeriphralID ) ;
00477             break ;
00478
00479         default:
00480             // Error wrong Bus ID
00481             break ;
00482
00483     }
00484 }
```

References [CLR_BIT](#), [RCC](#), [RCC_AHB1](#), [RCC_AHB1LPENR](#), [RCC_AHB2](#), [RCC_APB1](#), and [RCC_APB2](#).

7.53 MRCC_program.c

[Go to the documentation of this file.](#)

```
00001 /*********************************************************************
00012  *          Include headers
00013 /*********************************************************************
00014 #include "../../../LIB/LSTD_TYPES.h"
00015 #include "../../../LIB/LSTD_COMPILER.h"
00016 #include "../../../LIB/LSTD_VALUES.h"
00017 #include "../../../LIB/LSTD_BITMATH.h"
00018
00019 #include "MRCC_private.h"
00020 #include "MRCC_interface.h"
00021 #include "MRCC_config.h"
00022
```

```

00023
00024
00025 /****** Functions implementations *****/
00026 /*
00027 ***** */
00028
00029 FUNC(void) MRCC_vInit( void )
00030 {
00031     // PLLI2S (ON/OFF).
00032 #if PLLI2S == ENABLE
00033     SET_BIT( RCC->CR, RCC_CR_PLLI2SON ) ;
00034
00035 #elif PLLI2S == DISABLE
00036     CLR_BIT( RCC->CR, RCC_CR_PLLI2SON ) ;
00037
00038 #endif
00039
00040
00041
00042
00043
00044     // PLL (ON/OFF).
00045 #if PLL == ENABLE
00046     SET_BIT( RCC->CR, RCC_CR_PLLON ) ;
00047
00048 #elif PLL == DISABLE
00049     CLR_BIT( RCC->CR, RCC_CR_PLLON ) ;
00050
00051 #endif
00052
00053
00054
00055
00056     // CSS (ON/OFF).
00057 #if CSS == ENABLE
00058     SET_BIT( RCC->CR, RCC_CR_CSSON ) ;
00059
00060 #elif CSS == DISABLE
00061     CLR_BIT( RCC->CR, RCC_CR_CSSON ) ;
00062
00063 #endif
00064
00065
00066
00067
00068     // HSEBYP.
00069 #if HSEBYP == BYBASED
00070     SET_BIT( RCC->CR, RCC_CR_HSEBYP ) ;
00071
00072 #elif HSEBYP == NOTBYBASED
00073     CLR_BIT( RCC->CR, RCC_CR_HSEBYP ) ;
00074
00075 #endif
00076
00077
00078
00079
00080     // Select CLK switch (HSI/HSE/PLL).
00081 #if SW == HSI
00082     CLR_BIT( RCC->CFGR, RCC_CFGR_SW_b0 ) ;
00083     CLR_BIT( RCC->CFGR, RCC_CFGR_SW_b1 ) ;
00084
00085 #elif SW == HSE
00086
00087     SET_BIT( RCC->CFGR, RCC_CFGR_SW_b0 ) ;
00088     CLR_BIT( RCC->CFGR, RCC_CFGR_SW_b1 ) ;
00089
00090
00091 #elif SW == PLLCLK
00092
00093     CLR_BIT( RCC->CFGR, RCC_CFGR_SW_b0 ) ;
00094     SET_BIT( RCC->CFGR, RCC_CFGR_SW_b1 ) ;
00095
00096 #endif
00097
00098     // Select CLK switch status (HSI/HSE/PLL).
00099 #if SWS == HSI
00100
00101     CLR_BIT( RCC->CFGR, RCC_CFGR_SWS_b0 ) ;
00102     CLR_BIT( RCC->CFGR, RCC_CFGR_SWS_b1 ) ;
00103
00104 #elif SWS == HSE
00105
00106     SET_BIT( RCC->CFGR, RCC_CFGR_SWS_b0 ) ;
00107     CLR_BIT( RCC->CFGR, RCC_CFGR_SWS_b1 ) ;
00108
00109 #elif SWS == PLLCLK

```

```

00110
00111     CLR_BIT( RCC->CFGR, RCC_CFGR_SWS_b0 ) ;
00112     SET_BIT( RCC->CFGR, RCC_CFGR_SWS_b1 ) ;
00113
00114 #endif
00115
00116
00117 // MCO2 selection:
00118 #if MCO2 == SYSCLK
00119
00120     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2_b0 ) ;
00121     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2_b1 ) ;
00122
00123 #elif MCO2 == PLLI2SCLK
00124
00125     SET_BIT( RCC->CFGR, RCC_CFGR_MOC2_b0 ) ;
00126     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2_b1 ) ;
00127
00128 #elif MCO2 == HSE
00129
00130     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2_b0 ) ;
00131     SET_BIT( RCC->CFGR, RCC_CFGR_MOC2_b1 ) ;
00132
00133 #elif MCO2 == PLLCLK
00134
00135     SET_BIT( RCC->CFGR, RCC_CFGR_MOC2_b0 ) ;
00136     SET_BIT( RCC->CFGR, RCC_CFGR_MOC2_b1 ) ;
00137
00138 #endif
00139
00140
00141 // MCO2 prescaler:
00142 #if MCO2PRE == NoDivision
00143
00144     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b0 ) ;
00145     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b1 ) ;
00146     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b2 ) ;
00147
00148 #elif MCO2PRE == DivisionBy2
00149
00150     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b0 ) ;
00151     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b1 ) ;
00152     SET_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b2 ) ;
00153
00154 #elif MCO2PRE == DivisionBy3
00155
00156     SET_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b0 ) ;
00157     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b1 ) ;
00158     SET_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b2 ) ;
00159
00160 #elif MCO2PRE == DivisionBy4
00161
00162     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b0 ) ;
00163     SET_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b1 ) ;
00164     SET_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b2 ) ;
00165
00166 #elif MCO2PRE == DivisionBy5
00167
00168     SET_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b0 ) ;
00169     SET_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b1 ) ;
00170     SET_BIT( RCC->CFGR, RCC_CFGR_MOC2PRE_b2 ) ;
00171
00172 #endif
00173
00174
00175 // MCO1 selection:
00176 #if MCO1 == HSI
00177
00178     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1_b0 ) ;
00179     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1_b1 ) ;
00180
00181 #elif MCO1 == LSE
00182
00183     SET_BIT( RCC->CFGR, RCC_CFGR_MOC1_b0 ) ;
00184     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1_b1 ) ;
00185
00186 #elif MCO1 == HSE
00187
00188     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1_b0 ) ;
00189     SET_BIT( RCC->CFGR, RCC_CFGR_MOC1_b1 ) ;
00190
00191 #elif MCO1 == PLLCLK
00192
00193     SET_BIT( RCC->CFGR, RCC_CFGR_MOC1_b0 ) ;
00194     SET_BIT( RCC->CFGR, RCC_CFGR_MOC1_b1 ) ;
00195
00196 #endif

```

```

00197
00198
00199 // MCO1 prescaler:
00200 #if MCO1PRE == NoDivision
00201
00202     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b0 ) ;
00203     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b1 ) ;
00204     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b2 ) ;
00205
00206 #elif MCO1PRE == DivisionBy2
00207
00208     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b0 ) ;
00209     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b1 ) ;
00210     SET_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b2 ) ;
00211
00212 #elif MCO1PRE == DivisionBy3
00213
00214     SET_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b0 ) ;
00215     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b1 ) ;
00216     SET_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b2 ) ;
00217
00218 #elif MCO1PRE == DivisionBy4
00219
00220     CLR_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b0 ) ;
00221     SET_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b1 ) ;
00222     SET_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b2 ) ;
00223
00224 #elif MCO1PRE == DivisionBy5
00225
00226     SET_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b0 ) ;
00227     SET_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b1 ) ;
00228     SET_BIT( RCC->CFGR, RCC_CFGR_MOC1PRE_b2 ) ;
00229
00230 #endif
00231
00232
00233 // AHB prescalers:
00234 #if HPRE == NoDivision
00235
00236     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00237     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00238     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00239     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00240
00241 #elif HPRE == SYSCLKby2
00242
00243     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00244     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00245     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00246     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00247
00248 #elif HPRE == SYSCLKby4
00249
00250     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00251     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00252     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00253     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00254
00255 #elif HPRE == SYSCLKby8
00256
00257     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00258     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00259     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00260     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00261
00262 #elif HPRE == SYSCLKby16
00263
00264     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00265     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00266     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00267     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00268
00269 #elif HPRE == SYSCLKby64
00270
00271     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00272     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00273     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00274     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00275
00276 #elif HPRE == SYSCLKby128
00277
00278     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00279     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00280     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00281     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00282
00283 #elif HPRE == SYSCLKby256

```

```

00284     CLR_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00285     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00286     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00287     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00288
00289
00290 #elif HPRE == SYSCLKby512
00291     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b0 ) ;
00292     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b1 ) ;
00293     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b2 ) ;
00294     SET_BIT( RCC->CFGR, RCC_CFGR_HPRE_b3 ) ;
00295
00296
00297 #endif
00298
00299
00300 // APB1 prescalers:
00301 #if PPREG == NoDivision
00302     CLR_BIT( RCC->CFGR, RCC_CFGR_PPREG_b0 ) ;
00303     CLR_BIT( RCC->CFGR, RCC_CFGR_PPREG_b1 ) ;
00304     CLR_BIT( RCC->CFGR, RCC_CFGR_PPREG_b2 ) ;
00305
00306
00307 #elif PPREG == AHBby2
00308     CLR_BIT( RCC->CFGR, RCC_CFGR_PPREG_b0 ) ;
00309     CLR_BIT( RCC->CFGR, RCC_CFGR_PPREG_b1 ) ;
00310     SET_BIT( RCC->CFGR, RCC_CFGR_PPREG_b2 ) ;
00311
00312
00313 #elif PPREG == AHBby4
00314     SET_BIT( RCC->CFGR, RCC_CFGR_PPREG_b0 ) ;
00315     CLR_BIT( RCC->CFGR, RCC_CFGR_PPREG_b1 ) ;
00316     SET_BIT( RCC->CFGR, RCC_CFGR_PPREG_b2 ) ;
00317
00318
00319 #elif PPREG == AHBby8
00320     CLR_BIT( RCC->CFGR, RCC_CFGR_PPREG_b0 ) ;
00321     SET_BIT( RCC->CFGR, RCC_CFGR_PPREG_b1 ) ;
00322     SET_BIT( RCC->CFGR, RCC_CFGR_PPREG_b2 ) ;
00323
00324
00325 #elif PPREG == AHBby16
00326     SET_BIT( RCC->CFGR, RCC_CFGR_PPREG_b0 ) ;
00327     SET_BIT( RCC->CFGR, RCC_CFGR_PPREG_b1 ) ;
00328     SET_BIT( RCC->CFGR, RCC_CFGR_PPREG_b2 ) ;
00329
00330
00331 #endif
00332
00333 // APB2 prescalers:
00334 #if PPREG2 == NoDivision
00335
00336     CLR_BIT( RCC->CFGR, RCC_CFGR_PPREG2_b0 ) ;
00337     CLR_BIT( RCC->CFGR, RCC_CFGR_PPREG2_b1 ) ;
00338     CLR_BIT( RCC->CFGR, RCC_CFGR_PPREG2_b2 ) ;
00339
00340 #elif PPREG2 == AHBby2
00341     CLR_BIT( RCC->CFGR, RCC_CFGR_PPREG2_b0 ) ;
00342     CLR_BIT( RCC->CFGR, RCC_CFGR_PPREG2_b1 ) ;
00343     SET_BIT( RCC->CFGR, RCC_CFGR_PPREG2_b2 ) ;
00344
00345
00346 #elif PPREG2 == AHBby4
00347
00348     SET_BIT( RCC->CFGR, RCC_CFGR_PPREG2_b0 ) ;
00349     CLR_BIT( RCC->CFGR, RCC_CFGR_PPREG2_b1 ) ;
00350     SET_BIT( RCC->CFGR, RCC_CFGR_PPREG2_b2 ) ;
00351
00352
00353 #elif PPREG2 == AHBby8
00354     CLR_BIT( RCC->CFGR, RCC_CFGR_PPREG2_b0 ) ;
00355     SET_BIT( RCC->CFGR, RCC_CFGR_PPREG2_b1 ) ;
00356     SET_BIT( RCC->CFGR, RCC_CFGR_PPREG2_b2 ) ;
00357
00358 #elif PPREG2 == AHBby16
00359     SET_BIT( RCC->CFGR, RCC_CFGR_PPREG2_b0 ) ;
00360     SET_BIT( RCC->CFGR, RCC_CFGR_PPREG2_b1 ) ;
00361     SET_BIT( RCC->CFGR, RCC_CFGR_PPREG2_b2 ) ;
00362
00363
00364 #endif
00365
00366
00367 // PLL configurations:
00368
00369
00370

```

```
00371     // Enable the selected CLK (HSI ON/HSE ON/PLL ON): //
00372     // HSE (ON/OFF).
00373 #if HSE_EN == ENABLE
00374
00375     SET_BIT( RCC->CR, RCC_CR_HSEON ) ;
00376
00377 #elif HSE_EN == DISABLE
00378
00379     CLR_BIT( RCC->CR, RCC_CR_HSEON ) ;
00380
00381 #endif
00382
00383
00384     // HSI (ON/OFF).
00385 #if HSI_EN == ENABLE
00386
00387     SET_BIT( RCC->CR, RCC_CR_HSION ) ;
00388
00389 #elif HSI_EN == DISABLE
00390
00391     CLR_BIT( RCC->CR, RCC_CR_HSION ) ;
00392
00393 #endif
00394
00395 }
00396
00397 /*****
00398 *****/
00400 FUNC(void) MRCC_vEnablePeriphralCLK( VAR(u32_t) A_u32BusID, VAR(u32_t) A_u32PeriphralID )
00401 {
00402
00403     switch( A_u32BusID )
00404     {
00405
00406         case RCC_AHB1 :
00407
00408             SET_BIT( RCC->AHB1ENR, A_u32PeriphralID ) ;
00409
00410             break ;
00411
00412
00413         case RCC_AHB2 :
00414
00415             SET_BIT( RCC->AHB2ENR, A_u32PeriphralID ) ;
00416
00417
00418         case RCC_APB1 :
00419
00420             SET_BIT( RCC->APB1ENR, A_u32PeriphralID ) ;
00421
00422             break ;
00423
00424
00425         case RCC_APB2 :
00426
00427             SET_BIT( RCC->APB2ENR, A_u32PeriphralID ) ;
00428
00429             break ;
00430
00431
00432         case RCC_AHB1LPENR:
00433
00434             SET_BIT( RCC->AHB1LPENR, A_u32PeriphralID ) ;
00435
00436             break ;
00437
00438
00439         default:
00440
00441             // Error wrong Bus ID
00442
00443             break ;
00444
00445     }
00446 }
00447
00448 }
00449
00450 *****/
00451 *****/
00452 FUNC(void) MRCC_vDisablePeriphralCLK( VAR(u32_t) A_u32BusID, VAR(u32_t) A_u32PeriphralID )
00453
00454 {
00455
00456     switch( A_u32BusID )
00457     {
```

```

00458
00459     case RCC_AHB1 :
00460         CLR_BIT( RCC->AHB1ENR, A_u32PeriphralID ) ;
00461         break ;
00462
00463     case RCC_AHB2 :
00464         CLR_BIT( RCC->AHB2ENR, A_u32PeriphralID ) ;
00465         break ;
00466
00467     case RCC_APB1 :
00468         CLR_BIT( RCC->APB1ENR, A_u32PeriphralID ) ;
00469         break ;
00470
00471     case RCC_APB2 :
00472         CLR_BIT( RCC->APB2ENR, A_u32PeriphralID ) ;
00473         break ;
00474
00475     case RCC_AHB1LPENR:
00476         CLR_BIT( RCC->AHB1LPENR, A_u32PeriphralID ) ;
00477         break ;
00478
00479     default:
00480         // Error wrong Bus ID
00481         break ;
00482
00483     }
00484
00485 }
00486
00487
00488
00489
00490
00491
00492
00493
00494
00495
00496
00497
00498
00499
00500
00501
00502
00503
00504
00505 ****
00506 ****
00507
00508
00509
00510
00511
00512
00513
00514
00515

```

7.54 COTS/MCAL/SysTick/SysTick_config.h File Reference

Macros

- #define CLK_SOURCE AHB_DividedBy8
- #define Exception_Request Dont AssertRequest
- #define SINGLE_INTERVAL_MODE (1)

Single interval mode.
- #define PERIODIC_INTERVAL_MODE (2)

Periodic interval mode.
- #define MAX_TICKS 16777216

7.54.1 Macro Definition Documentation

7.54.1.1 Exception_Request

```
#define Exception_Request Dont AssertRequest
```

Options: Dont AssertRequest AssertRequest

Definition at line 38 of file [SysTick_config.h](#).

7.54.1.2 MAX_TICKS

```
#define MAX_TICKS 16777216
```

Definition at line 61 of file [SysTick_config.h](#).

7.55 SysTick_config.h

[Go to the documentation of this file.](#)

```
00001 /* FILENAME: SysTick_config
00002 * Author: Ali El Bana
00003 * Version: V1.0
00004 * DATE: Sun 09/04/2022
00005 */
00006 #ifndef _SysTick_config_H
00007 #define _SysTick_config_H
00008
00009 /*****
00010 // SysTick Configurations //
00011 *****/
00012
00024 /*options:
00025 *AHB_DividedBy8
00026 *AHB
00027 */
00028 #define CLK_SOURCE AHB_DividedBy8
00031 /*****
00032
00038 #define Exception_Request Dont AssertRequest
00039
00040 /*****
00041
00052 #define SINGLE_INTERVAL_MODE (1)
00057 #define PERIODIC_INTERVAL_MODE (2)
00061 #define MAX_TICKS 16777216
00062
00063
00064 #endif // _SysTick_config_H
```

7.56 COTS/MCAL/SysTick/SysTick_interface.h File Reference

This file contains the interfacing information for the Systick module.

Macros

- `#define BUSY_TICK_TIME (1000)`
Set the time you want to count when using the busy wait function.
- `#define SINGLE_INTERVAL_TICK_TIME (1000)`
Set the time you want to count when using the single interval function.
- `#define PERIODIC_INTERVAL_TICK_TIME (1000)`
Set the time you want to count when using the periodic interval function.
- `#define MILLI_SEC (1)`
Delay in milli seconds.
- `#define MICRO_SEC (2)`
- `#define SEC (3)`
Delay in seconds.

Functions

- `void MSysTick_vInit (void)`
Initialize the Systick module.
- `void MSysTick_vSetBusyWait (VAR(u32_t) A_u32Ticks)`
Synchronous delay function.
- `void MSysTick_vDelay (VAR(u32_t) A_u32Ticks, VAR(u32_t) A_u32TickType)`
Synchronous delay function.
- `void MSysTick_vDelayMicroSec (VAR(u32_t) A_u32Ticks)`
Synchronous delay function.
- `void MSysTick_vDelayMilliSec (VAR(u32_t) A_u32Ticks)`
Synchronous delay function.
- `void MSysTick_vDelaySec (VAR(u32_t) A_u32Ticks)`
Synchronous delay function.
- `void MSysTick_vSetSingleInterval (VAR(u32_t) A_u32Ticks, void(*A_Fptr)(void))`
Asynchronous 1-cycle delay function.
- `void MSysTick_vSetPeriodicInterval (VAR(u32_t) A_u32Ticks, void(*A_Fptr)(void))`
Asynchronous repetitive delay function.
- `void MSysTick_vStopInterval (void)`
Stop the interval delay function.
- `u32_t MSysTick_u32GetElapsedTime (void)`
Return the elapsed time since the start of the countdown.
- `u32_t MSysTick_u32GetRemainingTime (void)`
Return the remaining time in the systick register.
- `void MSysTick_vEnable (void)`
Enable the Systick timer.
- `void MSysTick_vDisable (void)`
Disable the Systick timer.
- `void MSysTick_vEnableException (void)`
Enable the Systick Exception.

7.56.1 Detailed Description

This file contains the interfacing information for the Systick module.

Author

Ali El Bana

Version

1.0

Date

09/04/2022

Definition in file [SysTick_interface.h](#).

7.56.2 Macro Definition Documentation

7.56.2.1 BUSY_TICK_TIME

```
#define BUSY_TICK_TIME (1000)
```

Set the time you want to count when using the busy wait function.

Definition at line 113 of file [SysTick_interface.h](#).

7.56.2.2 SINGLE_INTERVAL_TICK_TIME

```
#define SINGLE_INTERVAL_TICK_TIME (1000)
```

Set the time you want to count when using the single interval function.

Definition at line 119 of file [SysTick_interface.h](#).

7.56.2.3 PERIODIC_INTERVAL_TICK_TIME

```
#define PERIODIC_INTERVAL_TICK_TIME (1000)
```

Set the time you want to count when using the periodic interval function.

Definition at line 125 of file [SysTick_interface.h](#).

7.56.3 Function Documentation

7.56.3.1 MSysTick_vInit()

```
void MSysTick_vInit (
    void )
```

Initialize the Systick module.

Definition at line 35 of file [SysTick_program.c](#).

```
00036 {
00037
00038     // Reset timer value.
00039     SysTick->VAL = INITIAL_ZERO;
00040
00041     // Choose the input CLK source.
00042 #if CLK_SOURCE == AHB_DividedBy8
00043     CLR_BIT(SysTick->CTRL, CLKSOURCE);
00044
00045 #elif CLK_SOURCE == AHB
00046     SET_BIT(SysTick->CTRL, CLKSOURCE);
00047 #endif
00048
00049     // SysTick exception request enable.
00050 #if Exception_Request == Dont AssertRequest
00051     CLR_BIT( SysTick->CTRL, TICKINT );
00052
00053 #elif Exception_Request == AssertRequest
00054     SET_BIT( SysTick->CTRL, TICKINT );
00055 #endif
00056
00057 }
```

References [CLKSOURCE](#), [CLR_BIT](#), [INITIAL_ZERO](#), [SET_BIT](#), [SysTick](#), and [TICKINT](#).

7.56.3.2 MSysTick_vSetBusyWait()

```
void MSysTick_vSetBusyWait (
    VAR(u32_t) A_u32Ticks )
```

Synchronous delay function.

Parameters

in	<i>A_u32Ticks</i>	Number of ticks to delay for
----	-------------------	------------------------------

7.56.3.3 MSysTick_vDelay()

```
void MSysTick_vDelay (
    VAR(u32_t) A_u32Ticks,
    VAR(u32_t) A_u32TickType )
```

Synchronous delay function.

Parameters

in	<i>A_u32Ticks</i>	Number of ticks to delay for
in	<i>A_u32TickType</i>	Type of the delay

7.56.3.4 MSysTick_vDelayMicroSec()

```
void MSysTick_vDelayMicroSec (
    VAR(u32_t) A_u32Ticks )
```

Synchronous delay function.

Parameters

in	<i>A_u32Ticks</i>	Number of ticks to delay in micro seconds for
----	-------------------	---

7.56.3.5 MSysTick_vDelayMilliSec()

```
void MSysTick_vDelayMilliSec (
    VAR(u32_t) A_u32Ticks )
```

Synchronous delay function.

Parameters

in	<i>A_u32Ticks</i>	Number of ticks to delay in milli seconds for
----	-------------------	---

7.56.3.6 MSysTick_vDelaySec()

```
void MSysTick_vDelaySec (
    VAR(u32_t) A_u32Ticks )
```

Synchronous delay function.

Parameters

in	<i>A_u32Ticks</i>	Number of ticks to delay in seconds for
----	-------------------	---

7.56.3.7 MSysTick_vSetSingleInterval()

```
void MSysTick_vSetSingleInterval (
    VAR(u32_t) A_u32Ticks,
    void(*)(void) A_Fptr )
```

Asynchronous 1-cycle delay function.

This function counts down the ticks number, and when it reaches 0, it stops, unlike `MSysTick_vSetPeriodicInterval`

Parameters

in	<i>A_u32Ticks</i>	Number of ticks to delay for
in	<i>A_Fptr</i>	Callback function to call when the countdown has finished

7.56.3.8 MSysTick_vSetPeriodicInterval()

```
void MSysTick_vSetPeriodicInterval (
    VAR(u32_t) A_u32Ticks,
    void(*)(void) A_Fptr )
```

Asynchronous repetitive delay function.

This function counts down the ticks number, and when it reaches 0, it starts the countdown again

Parameters

in	<i>A_u32Ticks</i>	Number of ticks to delay for
in	<i>A_Fptr</i>	Callback function to call when the countdown has finished

See also

- [MSysTick_vSetSingleInterval](#) for a 1-cycle delay function
- [MSysTick_vSetBusyWait](#) for a synchronous delay function
- [MSysTick_vStopInterval](#) to stop the interval delay function

7.56.3.9 MSysTick_vStopInterval()

```
void MSysTick_vStopInterval (
    void )
```

Stop the interval delay function.

See also

[MSysTick_vSetPeriodicInterval](#) for a interval delay function

Definition at line 302 of file [SysTick_program.c](#).

```
00303 {
00304
00305     // Disable the IRQ.
00306     SET_BIT( SysTick->CTRL, TICKINT ) ;
00307
00308     // Stop the timer.
00309     SET_BIT( SysTick->CTRL, TICKINT ) ;
00310
00311     // Clear the LOAD and VAL registers
00312     SysTick->LOAD = INITIAL_ZERO;
00313     SysTick->VAL = INITIAL_ZERO;
00314
00315 }
```

References [INITIAL_ZERO](#), [SET_BIT](#), [SysTick](#), and [TICKINT](#).

7.56.3.10 MSysTick_u32GetElapsedTime()

```
u32_t MSysTick_u32GetElapsedTime (
    void )
```

Return the elapsed time since the start of the countdown.

Returns

Return the elapsed time since the start of the countdown

Definition at line 320 of file [SysTick_program.c](#).

```
00321 {
00322     VAR(u32_t) L_u32ElapsedTime = INITIAL_ZERO;
00323
00324     L_u32ElapsedTime = (SysTick->LOAD - SysTick->VAL);
00325
00326     return L_u32ElapsedTime;
00327 }
```

References [INITIAL_ZERO](#), [SysTick](#), and [VAR](#).

7.56.3.11 MSysTick_u32GetRemainingTime()

```
u32_t MSysTick_u32GetRemainingTime (
    void )
```

Return the remaining time in the systick register.

Returns

The remaining time in the Systick register

Definition at line 332 of file [SysTick_program.c](#).

```
00333 {
00334     VAR(u32_t) L_u32RemainingTime = INITIAL_ZERO;
00335
00336     L_u32RemainingTime = SysTick->VAL;
00337
00338     return L_u32RemainingTime;
00339 }
```

References [INITIAL_ZERO](#), [SysTick](#), and [VAR](#).

7.56.3.12 MSysTick_vEnable()

```
void MSysTick_vEnable (
    void )
```

Enable the Systick timer.

Definition at line 344 of file [SysTick_program.c](#).

```
00345 {
00346
00347     // Start Timer.
00348     SET_BIT(SysTick->CTRL, COUNTER_ENABLE);
00349
00350 }
```

References [COUNTER_ENABLE](#), [SET_BIT](#), and [SysTick](#).

7.56.3.13 MSysTick_vDisable()

```
void MSysTick_vDisable (
    void )
```

Disable the Systick timer.

Definition at line 355 of file [SysTick_program.c](#).

```
00356 {
00357
00358     // Disable the peripheral interrupt.
00359     SET_BIT( SysTick->CTRL, TICKINT );
00360
00361     // Stop the timer.
00362     CLR_BIT(SysTick->CTRL, COUNTER_ENABLE);
00363
00364 }
```

References [CLR_BIT](#), [COUNTER_ENABLE](#), [SET_BIT](#), [SysTick](#), and [TICKINT](#).

7.56.3.14 MSysTick_vEnableException()

```
void MSysTick_vEnableException (
    void )
```

Enable the Systick Exception.

Definition at line 27 of file [SysTick_program.c](#).

```
00028 {
00029     SET_BIT(SysTick->CTRL, TICKINT);
00030 }
```

References [SET_BIT](#), [SysTick](#), and [TICKINT](#).

7.57 SysTick_interface.h

[Go to the documentation of this file.](#)

```

00001
00009 /* Header file guard */
0010 #ifndef _SysTick_interface_H
0011 #define _SysTick_interface_H
0012
0013 /*****
0014 *          Functions prototypes
0015 *****/
0016
0020 void MSysTick_vInit(void);
0021
0026 void MSysTick_vSetBusyWait(VAR(u32_t) A_u32Ticks);
0027
0033 void MSysTick_vDelay(VAR(u32_t) A_u32Ticks, VAR(u32_t) A_u32TickType);
0034
0039 void MSysTick_vDelayMicroSec(VAR(u32_t) A_u32Ticks);
0040
0045 void MSysTick_vDelayMilliSec(VAR(u32_t) A_u32Ticks);
0046
0051 void MSysTick_vDelaySec(VAR(u32_t) A_u32Ticks);
0052
0059 void MSysTick_vSetSingleInterval(VAR(u32_t) A_u32Ticks, void (*A_Fptr)(void));
0060
0070 void MSysTick_vSetPeriodicInterval(VAR(u32_t) A_u32Ticks, void (*A_Fptr)(void));
0071
0076 void MSysTick_vStopInterval(void);
0077
0082 u32_t MSysTick_u32GetElapsedTime(void);
0083
0088 u32_t MSysTick_u32GetRemainingTime(void);
0089
0093 void MSysTick_vEnable(void);
0094
0098 void MSysTick_vDisable(void);
0099
0103 void MSysTick_vEnableException(void);
0104
0105 /*****
0106 *          Interfacing macros
0107 *****/
0108
0113 #define BUSY_TICK_TIME (1000)
0114
0119 #define SINGLE_INTERVAL_TICK_TIME (1000)
0120
0125 #define PERIODIC_INTERVAL_TICK_TIME (1000)
0126
0137 #define MILLI_SEC (1)
0138
0143 #define MICRO_SEC (2)
0144
0149 #define SEC (3)
0152#endif // _SysTick_interface_H

```

7.58 COTS/MCAL/SysTick/SysTick_private.h File Reference

This file contains the registers information and addresses for the Systick module.

Data Structures

- struct [SysTick_Type](#)
Systick memory map structure declaration for the Systick's registers.

Macros

- #define SysTick_BASE_ADDRESS (0xE000E010)
- #define SysTick ((volatile P2VAR(SysTick_Type))(SysTick_BASE_ADDRESS))
- #define COUNTFLAG (16)
- #define CLKSOURCE (2)
- #define TICKINT (1)
- #define COUNTER_ENABLE (0)
- #define AHB_DividedBy8 (1)
- #define AHB (2)
- #define Dont_ASSERTRequest (1)
- #define ASSERTRequest (2)

7.58.1 Detailed Description

This file contains the registers information and addresses for the Systick module.

Author

Ali El Bana

Version

1.0

Date

09/04/2022

Definition in file [SysTick_private.h](#).

7.59 SysTick_private.h

[Go to the documentation of this file.](#)

```

00001
00009 /* Header file guard */
00010 #ifndef _SysTick_private_H
00011 #define _SysTick_private_H
00012
00018 typedef struct
00019 {
00024     u32_t CTRL;
00029     u32_t LOAD;
00034     u32_t VAL;
00039     u32_t CALIB;
00040
00041 } SysTick_Type;
00042
00055 #define SysTick_BASE_ADDRESS (0xE000E010)
00056
00070 #define SysTick ((volatile P2VAR(SysTick_Type))(SysTick_BASE_ADDRESS))
00071
00085 #define COUNTFLAG (16)
00086
00091 #define CLKSOURCE (2)
00092
00097 #define TICKINT (1)
00098
00108 #define COUNTER_ENABLE (0)
00120 #define AHB_DividedBy8 (1)
00121
00126 #define AHB (2)
00138 #define Dont_ASSERTRequest (1)
00139
00144 #define ASSERTRequest (2)
00147 #endif // _SysTick_private_H

```

7.60 COTS/MCAL/SysTick/SysTick_program.c File Reference

This file contains the source code of the interfacing for the Systick modules.

```
#include "../../LIB/LSTD_TYPES.h"
#include "../../LIB/LSTD_COMPILER.h"
#include "../../LIB/LSTD_VALUES.h"
#include "../../LIB/LSTD_BITMATH.h"
#include "SysTick_interface.h"
#include "SysTick_private.h"
#include "SysTick_config.h"
```

Functions

- void [MSysTick_vEnableException](#) (void)
Enable the Systick Exception.
- void [MSysTick_vInit](#) (void)
Initialize the Systick module.
- void [MSysTick_vSetBusyWait](#) (u32_t A_u32Ticks)
- void [MSysTick_vDelay](#) (u32_t A_u32Ticks, u32_t A_u32TickType)
- void [MSysTick_vDelayMicroSec](#) (u32_t A_u32Ticks)
- void [MSysTick_vDelayMilliSec](#) (u32_t A_u32Ticks)
- void [MSysTick_vDelaySec](#) (u32_t A_u32Ticks)
- void [MSysTick_vSetSingleInterval](#) (u32_t A_u32Ticks, void(*A_Fptr)(void))
- void [MSysTick_vSetPeriodicInterval](#) (u32_t A_u32Ticks, void(*A_Fptr)(void))
- void [MSysTick_vStopInterval](#) (void)
Stop the interval delay function.
- [u32_t MSysTick_u32GetElapsedTime](#) (void)
Return the elapsed time since the start of the countdown.
- [u32_t MSysTick_u32GetRemainingTime](#) (void)
Return the remaining time in the systick register.
- void [MSysTick_vEnable](#) (void)
Enable the Systick timer.
- void [MSysTick_vDisable](#) (void)
Disable the Systick timer.
- void [SysTick_Handler](#) (void)

7.60.1 Detailed Description

This file contains the source code of the interfacing for the Systick modules.

Author

Ali El Bana

Version

1.0

Date

09/04/2022

Definition in file [SysTick_program.c](#).

7.60.2 Function Documentation

7.60.2.1 MSysTick_vEnableException()

```
void MSysTick_vEnableException (
    void )
```

Enable the Systick Exception.

Definition at line 27 of file [SysTick_program.c](#).

```
00028 {
00029     SET_BIT(SysTick->CTRL, TICKINT);
00030 }
```

References [SET_BIT](#), [SysTick](#), and [TICKINT](#).

7.60.2.2 MSysTick_vInit()

```
void MSysTick_vInit (
    void )
```

Initialize the Systick module.

Definition at line 35 of file [SysTick_program.c](#).

```
00036 {
00037
00038     // Reset timer value.
00039     SysTick->VAL = INITIAL_ZERO;
00040
00041     // Choose the input CLK source.
00042 #if CLK_SOURCE == AHB_DividedBy8
00043     CLR_BIT(SysTick->CTRL, CLKSOURCE);
00044
00045 #elif CLK_SOURCE == AHB
00046     SET_BIT(SysTick->CTRL, CLKSOURCE);
00047 #endif
00048
00049     // SysTick exception request enable.
00050 #if Exception_Request == Dont AssertRequest
00051     CLR_BIT( SysTick->CTRL, TICKINT ) ;
00052
00053 #elif Exception_Request == AssertRequest
00054     SET_BIT( SysTick->CTRL, TICKINT ) ;
00055 #endif
00056
00057 }
```

References [CLKSOURCE](#), [CLR_BIT](#), [INITIAL_ZERO](#), [SET_BIT](#), [SysTick](#), and [TICKINT](#).

7.60.2.3 MSysTick_vSetBusyWait()

```
void MSysTick_vSetBusyWait (
    u32_t A_u32Ticks )
```

Definition at line 62 of file [SysTick_program.c](#).

```
00063 {
00064     // Load Ticks to the load register.
00065     SysTick->LOAD = A_u32Ticks;
00066
00067     // Reset timer value.
00068     SysTick->VAL = INITIAL_ZERO;
00069
00070     // Start Timer.
00071     SET_BIT( SysTick->CTRL, COUNTER_ENABLE ) ;
00072
00073     // Wait till the flag is raised (= 1).
00074     while( GET_BIT(SysTick->CTRL, COUNTFLAG) == FLAG_CLEARED ) ;
00075
00076     // Stop the timer.
00077     CLR_BIT( SysTick->CTRL, COUNTER_ENABLE ) ;
00078
00079     // Clear the LOAD and VAL registers
00080     SysTick->LOAD = INITIAL_ZERO;
00081     SysTick->VAL = INITIAL_ZERO;
00082
00083 }
00084 }
```

References [CLR_BIT](#), [COUNTER_ENABLE](#), [COUNTFLAG](#), [FLAG_CLEARED](#), [GET_BIT](#), [INITIAL_ZERO](#), [SET_BIT](#), and [SysTick](#).

7.60.2.4 MSysTick_vDelay()

```
void MSysTick_vDelay (
    u32_t A_u32Ticks,
    u32_t A_u32TickType )
```

Definition at line 88 of file [SysTick_program.c](#).

```
00089 {
00090
00091     VAR(u32_t) l_u32TickNum = INITIAL_ZERO ;
00092
00093     if( A_u32TickType == MILLI_SEC )
00094     {
00095         l_u32TickNum = (A_u32Ticks * 1000) - 1 ;
00096     }
00097
00098     else if( A_u32TickType == MICRO_SEC )
00099     {
00100         l_u32TickNum = A_u32Ticks - 1 ;
00101     }
00102
00103     else if( A_u32TickType == SEC )
00104     {
00105         l_u32TickNum = (A_u32Ticks * 1000000) - 1 ;
00106     }
00107
00108     else
00109     {
00110         // error
00111     }
00112
00113     if (l_u32TickNum < MAX_TICKS)
00114     {
00115
00116         // Load Ticks to the load register.
00117         SysTick->LOAD = l_u32TickNum;
00118
00119         // Reset timer value.
00120         SysTick->VAL = INITIAL_ZERO;
00121
00122         // Start Timer.
```

```

00123     SET_BIT( SysTick->CTRL, COUNTER_ENABLE ) ;
00124
00125     // Wait till the flag is raised (= 1).
00126     while( GET_BIT(SysTick->CTRL, COUNTFLAG) == FLAG_CLEARED ) ;
00127
00128     // Stop the timer.
00129     CLR_BIT( SysTick->CTRL, COUNTER_ENABLE ) ;
00130
00131     // Clear the LOAD and VAL registers
00132     SysTick->LOAD = INITIAL_ZERO;
00133     SysTick->VAL = INITIAL_ZERO;
00134
00135 }
00136
00137 }
```

References [CLR_BIT](#), [COUNTER_ENABLE](#), [COUNTFLAG](#), [FLAG_CLEARED](#), [GET_BIT](#), [INITIAL_ZERO](#), [MAX_TICKS](#), [MICRO_SEC](#), [MILLI_SEC](#), [SEC](#), [SET_BIT](#), [SysTick](#), and [VAR](#).

7.60.2.5 MSysTick_vDelayMicroSec()

```

void MSysTick_vDelayMicroSec (
    u32_t A_u32Ticks )
```

Definition at line 142 of file [SysTick_program.c](#).

```

00143 {
00144
00145     VAR(u32_t) l_u32TickNum = INITIAL_ZERO ;
00146
00147     l_u32TickNum = (A_u32Ticks) - 1 ;
00148
00149     if (l_u32TickNum < MAX_TICKS)
00150     {
00151
00152         // Load Ticks to the load register.
00153         SysTick->LOAD = l_u32TickNum;
00154
00155         // Reset timer value.
00156         SysTick->VAL = INITIAL_ZERO;
00157
00158         // Start Timer.
00159         SET_BIT( SysTick->CTRL, COUNTER_ENABLE ) ;
00160
00161         // Wait till the flag is raised (= 1).
00162         while( GET_BIT(SysTick->CTRL, COUNTFLAG) == FLAG_CLEARED ) ;
00163
00164         // Stop the timer.
00165         CLR_BIT( SysTick->CTRL, COUNTER_ENABLE ) ;
00166
00167         // Clear the LOAD and VAL registers
00168         SysTick->LOAD = INITIAL_ZERO;
00169         SysTick->VAL = INITIAL_ZERO;
00170
00171     }
00172
00173 }
```

References [CLR_BIT](#), [COUNTER_ENABLE](#), [COUNTFLAG](#), [FLAG_CLEARED](#), [GET_BIT](#), [INITIAL_ZERO](#), [MAX_TICKS](#), [SET_BIT](#), [SysTick](#), and [VAR](#).

7.60.2.6 MSysTick_vDelayMilliSec()

```

void MSysTick_vDelayMilliSec (
    u32_t A_u32Ticks )
```

Definition at line 178 of file [SysTick_program.c](#).

```

00179 {
00180     VAR(u32_t) l_u32TickNum = INITIAL_ZERO ;
00182
00183     l_u32TickNum = (A_u32Ticks * 1000) - 1 ;
00184
00185     if (l_u32TickNum < MAX_TICKS)
00186     {
00187
00188         // Load Ticks to the load register.
00189         SysTick->LOAD = l_u32TickNum;
00190
00191         // Reset timer value.
00192         SysTick->VAL = INITIAL_ZERO;
00193
00194         // Start Timer.
00195         SET_BIT( SysTick->CTRL, COUNTER_ENABLE ) ;
00196
00197         // Wait till the flag is raised (= 1).
00198         while( GET_BIT(SysTick->CTRL, COUNTFLAG) == FLAG_CLEARED ) ;
00199
00200         // Stop the timer.
00201         CLR_BIT( SysTick->CTRL, COUNTER_ENABLE ) ;
00202
00203         // Clear the LOAD and VAL registers
00204         SysTick->LOAD = INITIAL_ZERO;
00205         SysTick->VAL = INITIAL_ZERO;
00206
00207     }
00208
00209 }
```

References [CLR_BIT](#), [COUNTER_ENABLE](#), [COUNTFLAG](#), [FLAG_CLEARED](#), [GET_BIT](#), [INITIAL_ZERO](#), [MAX_TICKS](#), [SET_BIT](#), [SysTick](#), and [VAR](#).

7.60.2.7 MSysTick_vDelaySec()

```
void MSysTick_vDelaySec (
    u32_t A_u32Ticks )
```

Definition at line 214 of file [SysTick_program.c](#).

```

00215 {
00216
00217     VAR(u32_t) l_u32TickNum = INITIAL_ZERO ;
00218
00219     l_u32TickNum = (A_u32Ticks * 1000000) - 1 ;
00220
00221     if (l_u32TickNum < MAX_TICKS)
00222     {
00223
00224         // Load Ticks to the load register.
00225         SysTick->LOAD = l_u32TickNum;
00226
00227         // Reset timer value.
00228         SysTick->VAL = INITIAL_ZERO;
00229
00230         // Start Timer.
00231         SET_BIT( SysTick->CTRL, COUNTER_ENABLE ) ;
00232
00233         // Wait till the flag is raised (= 1).
00234         while( GET_BIT(SysTick->CTRL, COUNTFLAG) == FLAG_CLEARED ) ;
00235
00236         // Stop the timer.
00237         CLR_BIT( SysTick->CTRL, COUNTER_ENABLE ) ;
00238
00239         // Clear the LOAD and VAL registers
00240         SysTick->LOAD = INITIAL_ZERO;
00241         SysTick->VAL = INITIAL_ZERO;
00242
00243     }
00244
00245 }
```

References [CLR_BIT](#), [COUNTER_ENABLE](#), [COUNTFLAG](#), [FLAG_CLEARED](#), [GET_BIT](#), [INITIAL_ZERO](#), [MAX_TICKS](#), [SET_BIT](#), [SysTick](#), and [VAR](#).

7.60.2.8 MSysTick_vSetSingleInterval()

```
void MSysTick_vSetSingleInterval (
    u32_t A_u32Ticks,
    void(*)(void) A_Fptr )
```

Definition at line 250 of file [SysTick_program.c](#).

```
00251 {
00252
00253     // Save the callback.
00254     GS_vCallbackFunc = A_Fptr ;
00255
00256     // Reset timer value.
00257     SysTick->VAL = INITIAL_ZERO ;
00258
00259     // Load Ticks to the load register.
00260     SysTick->LOAD = A_u32Ticks ;
00261
00262     // Start Timer.
00263     SET_BIT( SysTick->CTRL, COUNTER_ENABLE ) ;
00264
00265     // Set interval mode to single.
00266     GS_u8MyIntervalMode = SINGLE_INTERVAL_MODE ;
00267
00268     // Enable the IRQ.
00269     SET_BIT( SysTick->CTRL, TICKINT ) ;
00270
00271 }
```

References [COUNTER_ENABLE](#), [INITIAL_ZERO](#), [SET_BIT](#), [SINGLE_INTERVAL_MODE](#), [SysTick](#), and [TICKINT](#).

7.60.2.9 MSysTick_vSetPeriodicInterval()

```
void MSysTick_vSetPeriodicInterval (
    u32_t A_u32Ticks,
    void(*)(void) A_Fptr )
```

Definition at line 276 of file [SysTick_program.c](#).

```
00277 {
00278
00279     // Save the callback.
00280     GS_vCallbackFunc = A_Fptr;
00281
00282     // Reset timer value.
00283     SysTick->VAL = INITIAL_ZERO ;
00284
00285     // Load Ticks to the load register.
00286     SysTick->LOAD = A_u32Ticks;
00287
00288     // Start Timer.
00289     SET_BIT( SysTick->CTRL, COUNTER_ENABLE ) ;
00290
00291     // Set interval mode to single.
00292     GS_u8MyIntervalMode = PERIODIC_INTERVAL_MODE;
00293
00294     // Enable the IRQ.
00295     SET_BIT( SysTick->CTRL, TICKINT ) ;
00296
00297 }
```

References [COUNTER_ENABLE](#), [INITIAL_ZERO](#), [PERIODIC_INTERVAL_MODE](#), [SET_BIT](#), [SysTick](#), and [TICKINT](#).

7.60.2.10 MSysTick_vStopInterval()

```
void MSysTick_vStopInterval (
    void )
```

Stop the interval delay function.

See also

[MSysTick_vSetPeriodicInterval](#) for a interval delay function

Definition at line 302 of file [SysTick_program.c](#).

```
00303 {
00304     // Disable the IRQ.
00305     SET_BIT( SysTick->CTRL, TICKINT ) ;
00307
00308     // Stop the timer.
00309     SET_BIT( SysTick->CTRL, TICKINT ) ;
00310
00311     // Clear the LOAD and VAL registers
00312     SysTick->LOAD = INITIAL_ZERO;
00313     SysTick->VAL = INITIAL_ZERO;
00314
00315 }
```

References [INITIAL_ZERO](#), [SET_BIT](#), [SysTick](#), and [TICKINT](#).

7.60.2.11 MSysTick_u32GetElapsedTime()

```
u32_t MSysTick_u32GetElapsedTime (
    void )
```

Return the elapsed time since the start of the countdown.

Returns

Return the elapsed time since the start of the countdown

Definition at line 320 of file [SysTick_program.c](#).

```
00321 {
00322     VAR(u32_t) L_u32ElapsedTime = INITIAL_ZERO;
00323
00324     L_u32ElapsedTime = (SysTick->LOAD - SysTick->VAL);
00325
00326     return L_u32ElapsedTime;
00327 }
```

References [INITIAL_ZERO](#), [SysTick](#), and [VAR](#).

7.60.2.12 MSysTick_u32GetRemainingTime()

```
u32_t MSysTick_u32GetRemainingTime (
    void )
```

Return the remaining time in the systick register.

Returns

The remaining time in the Systick register

Definition at line 332 of file [SysTick_program.c](#).

```
00333 {
00334     VAR(u32_t) L_u32RemainingTime = INITIAL_ZERO;
00335
00336     L_u32RemainingTime = SysTick->VAL;
00337
00338     return L_u32RemainingTime;
00339 }
```

References [INITIAL_ZERO](#), [SysTick](#), and [VAR](#).

7.60.2.13 MSysTick_vEnable()

```
void MSysTick_vEnable (
    void )
```

Enable the Systick timer.

Definition at line 344 of file [SysTick_program.c](#).

```
00345 {
00346
00347     // Start Timer.
00348     SET_BIT(SysTick->CTRL, COUNTER_ENABLE);
00349
00350 }
```

References [COUNTER_ENABLE](#), [SET_BIT](#), and [SysTick](#).

7.60.2.14 MSysTick_vDisable()

```
void MSysTick_vDisable (
    void )
```

Disable the Systick timer.

Definition at line 355 of file [SysTick_program.c](#).

```
00356 {
00357
00358     // Disable the peripheral interrupt.
00359     SET_BIT( SysTick->CTRL, TICKINT );
00360
00361     // Stop the timer.
00362     CLR_BIT(SysTick->CTRL, COUNTER_ENABLE);
00363
00364 }
```

References [CLR_BIT](#), [COUNTER_ENABLE](#), [SET_BIT](#), [SysTick](#), and [TICKINT](#).

7.60.2.15 SysTick_Handler()

```
void SysTick_Handler (
    void )
```

Definition at line 369 of file [SysTick_program.c](#).

```
00370 {
00371     volatile VAR(u8_t) L_u8ReadFlag = INITIAL_ZERO;
00373
00374     if (GS_vCallbackFunc != NULL)
00375     {
00376         // Callback notification.
00377         GS_vCallbackFunc();
00378     }
00379
00380     if (GS_u8MyIntervalMode == SINGLE_INTERVAL_MODE)
00381     {
00382         // Disable the IRQ.
00383         CLR_BIT( SysTick->CTRL, TICKINT ) ;
00384
00385         // Stop the timer.
00386         CLR_BIT( SysTick->CTRL, COUNTER_ENABLE ) ;
00387
00388         // Clear the LOAD and VAL registers
00389         SysTick->LOAD = INITIAL_ZERO;
00390         SysTick->VAL = INITIAL_ZERO;
00391
00392         GS_u8MyIntervalMode = PERIODIC_INTERVAL_MODE;
00393     }
00394
00395     // Clear IRQ flag.
00396     L_u8ReadFlag = GET_BIT(SysTick->CTRL, COUNTFLAG);
00397
00398 }
```

References [CLR_BIT](#), [COUNTER_ENABLE](#), [COUNTFLAG](#), [GET_BIT](#), [INITIAL_ZERO](#), [NULL](#), [PERIODIC_INTERVAL_MODE](#), [SINGLE_INTERVAL_MODE](#), [SysTick](#), [TICKINT](#), and [VAR](#).

7.61 SysTick_program.c

[Go to the documentation of this file.](#)

```
00001
00009 /***** Include headers *****/
0010 /*           Include headers           */
0011 /***** Include headers *****/
0012 #include "../../LIB/LSTD_TYPES.h"
0013 #include "../../LIB/LSTD_COMPILER.h"
0014 #include "../../LIB/LSTD_VALUES.h"
0015 #include "../../LIB/LSTD_BITMATH.h"
0016 #include "SysTick_interface.h"
0017 #include "SysTick_private.h"
0018 #include "SysTick_config.h"
0019
0020 STATIC P2FUNC(VAR(void), GS_vCallbackFunc) (void) = NULL ;
0021
0022 STATIC VAR(u8_t) GS_u8MyIntervalMode = INITIAL_ZERO ;
0023
0024
0025
0026
0027 FUNC(void) MSysTick_vEnableException(void)
0028 {
0029     SET_BIT(SysTick->CTRL, TICKINT);
0030 }
0031
0032
0033
0034
0035 FUNC(void) MSysTick_vInit(void)
0036 {
0037 }
```

```

00038     // Reset timer value.
00039     SysTick->VAL = INITIAL_ZERO;
00040
00041     // Choose the input CLK source.
00042 #if CLK_SOURCE == AHB_DividedBy8
00043     CLR_BIT(SysTick->CTRL, CLKSOURCE);
00044
00045 #elif CLK_SOURCE == AHB
00046     SET_BIT(SysTick->CTRL, CLKSOURCE);
00047 #endif
00048
00049     // SysTick exception request enable.
00050 #if Exception_Request == Dont AssertRequest
00051     CLR_BIT( SysTick->CTRL, TICKINT ) ;
00052
00053 #elif Exception_Request == AssertRequest
00054     SET_BIT( SysTick->CTRL, TICKINT ) ;
00055 #endif
00056
00057 }
00058
00059
00060 /******
00061 *****/
00062 FUNC(void) MSysTick_vSetBusyWait(VAR(u32_t) A_u32Ticks)
00063 {
00064
00065     // Load Ticks to the load register.
00066     SysTick->LOAD = A_u32Ticks;
00067
00068     // Reset timer value.
00069     SysTick->VAL = INITIAL_ZERO;
00070
00071     // Start Timer.
00072     SET_BIT( SysTick->CTRL, COUNTER_ENABLE ) ;
00073
00074     // Wait till the flag is raised (= 1).
00075     while( GET_BIT(SysTick->CTRL, COUNTFLAG) == FLAG_CLEARED ) ;
00076
00077     // Stop the timer.
00078     CLR_BIT( SysTick->CTRL, COUNTER_ENABLE ) ;
00079
00080     // Clear the LOAD and VAL registers
00081     SysTick->LOAD = INITIAL_ZERO;
00082     SysTick->VAL = INITIAL_ZERO;
00083
00084 }
00085
00086
00087 /******
00088 *****/
00088 FUNC(void) MSysTick_vDelay(VAR(u32_t) A_u32Ticks, VAR(u32_t) A_u32TickType )
00089 {
00090
00091     VAR(u32_t) l_u32TickNum = INITIAL_ZERO ;
00092
00093     if( A_u32TickType == MILLI_SEC )
00094     {
00095         l_u32TickNum = (A_u32Ticks * 1000) - 1 ;
00096     }
00097
00098     else if( A_u32TickType == MICRO_SEC )
00099     {
00100         l_u32TickNum = A_u32Ticks - 1 ;
00101     }
00102
00103     else if( A_u32TickType == SEC )
00104     {
00105         l_u32TickNum = (A_u32Ticks * 1000000) - 1 ;
00106     }
00107
00108     else
00109     {
00110         // error
00111     }
00112
00113     if (l_u32TickNum < MAX_TICKS)
00114     {
00115
00116         // Load Ticks to the load register.
00117         SysTick->LOAD = l_u32TickNum;
00118
00119         // Reset timer value.
00120         SysTick->VAL = INITIAL_ZERO;

```

```

00121     // Start Timer.
00122     SET_BIT( SysTick->CTRL, COUNTER_ENABLE ) ;
00123
00124     // Wait till the flag is raised (= 1).
00125     while( GET_BIT(SysTick->CTRL, COUNTFLAG) == FLAG_CLEARED ) ;
00126
00127     // Stop the timer.
00128     CLR_BIT( SysTick->CTRL, COUNTER_ENABLE ) ;
00129
00130     // Clear the LOAD and VAL registers
00131     SysTick->LOAD = INITIAL_ZERO;
00132     SysTick->VAL = INITIAL_ZERO;
00133
00134 }
00135
00136 }
00137 }
00138
00139
00140
00141
00142 FUNC(void) MSysTick_vDelayMicroSec( VAR(u32_t) A_u32Ticks )
00143 {
00144
00145     VAR(u32_t) l_u32TickNum = INITIAL_ZERO ;
00146
00147     l_u32TickNum = (A_u32Ticks) - 1 ;
00148
00149     if (l_u32TickNum < MAX_TICKS)
00150     {
00151
00152         // Load Ticks to the load register.
00153         SysTick->LOAD = l_u32TickNum;
00154
00155         // Reset timer value.
00156         SysTick->VAL = INITIAL_ZERO;
00157
00158         // Start Timer.
00159         SET_BIT( SysTick->CTRL, COUNTER_ENABLE ) ;
00160
00161         // Wait till the flag is raised (= 1).
00162         while( GET_BIT(SysTick->CTRL, COUNTFLAG) == FLAG_CLEARED ) ;
00163
00164         // Stop the timer.
00165         CLR_BIT( SysTick->CTRL, COUNTER_ENABLE ) ;
00166
00167         // Clear the LOAD and VAL registers
00168         SysTick->LOAD = INITIAL_ZERO;
00169         SysTick->VAL = INITIAL_ZERO;
00170
00171     }
00172
00173 }
00174
00175
00176
00177
00178 FUNC(void) MSysTick_vDelayMilliSec( VAR(u32_t) A_u32Ticks )
00179 {
00180
00181     VAR(u32_t) l_u32TickNum = INITIAL_ZERO ;
00182
00183     l_u32TickNum = (A_u32Ticks * 1000) - 1 ;
00184
00185     if (l_u32TickNum < MAX_TICKS)
00186     {
00187
00188         // Load Ticks to the load register.
00189         SysTick->LOAD = l_u32TickNum;
00190
00191         // Reset timer value.
00192         SysTick->VAL = INITIAL_ZERO;
00193
00194         // Start Timer.
00195         SET_BIT( SysTick->CTRL, COUNTER_ENABLE ) ;
00196
00197         // Wait till the flag is raised (= 1).
00198         while( GET_BIT(SysTick->CTRL, COUNTFLAG) == FLAG_CLEARED ) ;
00199
00200         // Stop the timer.
00201         CLR_BIT( SysTick->CTRL, COUNTER_ENABLE ) ;
00202
00203         // Clear the LOAD and VAL registers

```

```

00204     SysTick->LOAD = INITIAL_ZERO;
00205     SysTick->VAL = INITIAL_ZERO;
00206
00207 }
00208
00209 }
00210
00211 //*****
00212 //*****
00213
00214 FUNC(void) MSysTick_vDelaySec( VAR(u32_t) A_u32Ticks )
00215 {
00216
00217     VAR(u32_t) l_u32TickNum = INITIAL_ZERO ;
00218
00219     l_u32TickNum = (A_u32Ticks * 1000000) - 1 ;
00220
00221     if (l_u32TickNum < MAX_TICKS)
00222     {
00223
00224         // Load Ticks to the load register.
00225         SysTick->LOAD = l_u32TickNum;
00226
00227         // Reset timer value.
00228         SysTick->VAL = INITIAL_ZERO;
00229
00230         // Start Timer.
00231         SET_BIT( SysTick->CTRL, COUNTER_ENABLE ) ;
00232
00233         // Wait till the flag is raised (= 1).
00234         while( GET_BIT(SysTick->CTRL, COUNTFLAG) == FLAG_CLEARED ) ;
00235
00236         // Stop the timer.
00237         CLR_BIT( SysTick->CTRL, COUNTER_ENABLE ) ;
00238
00239         // Clear the LOAD and VAL registers
00240         SysTick->LOAD = INITIAL_ZERO;
00241         SysTick->VAL = INITIAL_ZERO;
00242
00243     }
00244
00245 }
00246
00247 //*****
00248 //*****
00249
00250 FUNC(void) MSysTick_vSetSingleInterval(VAR(u32_t) A_u32Ticks, P2FUNC(VAR(void), A_Fptr)(void))
00251 {
00252
00253     // Save the callback.
00254     GS_vCallbackFunc = A_Fptr ;
00255
00256     // Reset timer value.
00257     SysTick->VAL = INITIAL_ZERO ;
00258
00259     // Load Ticks to the load register.
00260     SysTick->LOAD = A_u32Ticks ;
00261
00262     // Start Timer.
00263     SET_BIT( SysTick->CTRL, COUNTER_ENABLE ) ;
00264
00265     // Set interval mode to single.
00266     GS_u8MyIntervalMode = SINGLE_INTERVAL_MODE ;
00267
00268     // Enable the IRQ.
00269     SET_BIT( SysTick->CTRL, TICKINT ) ;
00270
00271 }
00272
00273 //*****
00274 //*****
00275
00276 FUNC(void) MSysTick_vSetPeriodicInterval(VAR(u32_t) A_u32Ticks, P2FUNC(VAR(void), A_Fptr)(void))
00277 {
00278
00279     // Save the callback.
00280     GS_vCallbackFunc = A_Fptr;
00281
00282     // Reset timer value.
00283     SysTick->VAL = INITIAL_ZERO ;
00284

```

```
00285 // Load Ticks to the load register.  
00286 SysTick->LOAD = A_u32Ticks;  
00287  
00288 // Start Timer.  
00289 SET_BIT( SysTick->CTRL, COUNTER_ENABLE ) ;  
00290  
00291 // Set interval mode to single.  
00292 GS_u8MyIntervalMode = PERIODIC_INTERVAL_MODE;  
00293  
00294 // Enable the IRQ.  
00295 SET_BIT( SysTick->CTRL, TICKINT ) ;  
00296  
00297 }  
00298  
00299  
00300 //*****  
00301 //*****  
00302 FUNC(void) MSysTick_vStopInterval(void)  
00303 {  
00304  
00305 // Disable the IRQ.  
00306 SET_BIT( SysTick->CTRL, TICKINT ) ;  
00307  
00308 // Stop the timer.  
00309 SET_BIT( SysTick->CTRL, TICKINT ) ;  
00310  
00311 // Clear the LOAD and VAL registers  
00312 SysTick->LOAD = INITIAL_ZERO;  
00313 SysTick->VAL = INITIAL_ZERO;  
00314  
00315 }  
00316  
00317  
00318 //*****  
00319  
00320 VAR(u32_t) MSysTick_u32GetElapsedTime(void)  
00321 {  
00322     VAR(u32_t) L_u32ElapsedTime = INITIAL_ZERO;  
00323  
00324     L_u32ElapsedTime = (SysTick->LOAD - SysTick->VAL);  
00325  
00326     return L_u32ElapsedTime;  
00327 }  
00328  
00329  
00330 //*****  
00331  
00332 VAR(u32_t) MSysTick_u32GetRemainingTime(void)  
00333 {  
00334     VAR(u32_t) L_u32RemainingTime = INITIAL_ZERO;  
00335  
00336     L_u32RemainingTime = SysTick->VAL;  
00337  
00338     return L_u32RemainingTime;  
00339 }  
00340  
00341  
00342 //*****  
00343  
00344 FUNC(void) MSysTick_vEnable(void)  
00345 {  
00346  
00347     // Start Timer.  
00348     SET_BIT(SysTick->CTRL, COUNTER_ENABLE);  
00349  
00350 }  
00351  
00352  
00353  
00354  
00355 FUNC(void) MSysTick_vDisable(void)  
00356 {  
00357  
00358     // Disable the peripheral interrupt.  
00359     SET_BIT( SysTick->CTRL, TICKINT ) ;  
00360  
00361     // Stop the timer.
```

```
00362     CLR_BIT(SysTick->CTRL, COUNTER_ENABLE);
00363
00364 }
00365
00366 //*****
00367 //*****
00368
00369 FUNC(void) SysTick_Handler(void)
00370 {
00371
00372     volatile VAR(u8_t) L_u8ReadFlag = INITIAL_ZERO;
00373
00374     if (GS_vCallbackFunc != NULL)
00375     {
00376         // Callback notification.
00377         GS_vCallbackFunc();
00378     }
00379
00380     if (GS_u8MyIntervalMode == SINGLE_INTERVAL_MODE)
00381     {
00382         // Disable the IRQ.
00383         CLR_BIT( SysTick->CTRL, TICKINT ) ;
00384
00385         // Stop the timer.
00386         CLR_BIT( SysTick->CTRL, COUNTER_ENABLE ) ;
00387
00388         // Clear the LOAD and VAL registers
00389         SysTick->LOAD = INITIAL_ZERO;
00390         SysTick->VAL = INITIAL_ZERO;
00391
00392         GS_u8MyIntervalMode = PERIODIC_INTERVAL_MODE;
00393     }
00394
00395     // Clear IRQ flag.
00396     L_u8ReadFlag = GET_BIT(SysTick->CTRL, COUNTFLAG);
00397
00398 }
00399
00400 //*****
00401 //*****
```

Index

_16GROUP_NoSub_Priorities
 Interrupt priority grouping, 48

_2GROUP_8Sub_Priorities
 Interrupt priority grouping, 49

_4GROUP_4Sub_Priorities
 Interrupt priority grouping, 49

_8GROUP_2Sub_Priorities
 Interrupt priority grouping, 48

ADC
 NVIC Vector Table, 69

ADC1EN
 MRCC_config.h, 234

AF_Type
 MGPIOx_ConfigType, 111

AFRHx
 GPIOx_MemoryMapType, 108

AFLRx
 GPIOx_MemoryMapType, 108

AHB
 Systick Clock Sources, 98

AHB1ENR
 RCC_MemoryMapType, 121

AHB1ENR_CRCEN
 ID options, 85

AHB1ENR_DMA1EN
 ID options, 85

AHB1ENR_DMA2EN
 ID options, 85

AHB1ENR_GPIOAEN
 ID options, 87

AHB1ENR_GPIOBEN
 ID options, 87

AHB1ENR_GPIOCEN
 ID options, 86

AHB1ENR_GPIODEN
 ID options, 86

AHB1ENR_GPIOEEN
 ID options, 86

AHB1ENR_GPIOHEN
 ID options, 86

AHB1LPENR
 RCC_MemoryMapType, 122

AHB1LPENR_FLITFLPEN
 ID options, 93

AHB1RSTR
 RCC_MemoryMapType, 119

AHB2ENR
 RCC_MemoryMapType, 121

AHB2ENR_OTGFSEN
 ID options, 87

AHB2LPENR
 RCC_MemoryMapType, 122

AHB2RSTR
 RCC_MemoryMapType, 119

AHB_DividedBy8
 Systick Clock Sources, 98

AHBby16
 MRCC_private.h, 282

AHBby2
 MRCC_private.h, 282

AHBby4
 MRCC_private.h, 282

AHBby8
 MRCC_private.h, 282

AIRCR
 SCB_MemoryMapType, 127

APB1ENR
 RCC_MemoryMapType, 121

APB1ENR_I2C1EN
 ID options, 88

APB1ENR_I2C2EN
 ID options, 88

APB1ENR_I2C3EN
 ID options, 88

APB1ENR_PWREN
 ID options, 87

APB1ENR_SPI2EN
 ID options, 89

APB1ENR_SPI3EN
 ID options, 89

APB1ENR_TIM2EN
 ID options, 90

APB1ENR_TIM3EN
 ID options, 90

APB1ENR_TIM4EN
 ID options, 90

APB1ENR_TIM5EN
 ID options, 89

APB1ENR_USART2EN
 ID options, 88

APB1ENR_WWDGEN
 ID options, 89

APB1LPENR
 RCC_MemoryMapType, 123

APB1RSTR
 RCC_MemoryMapType, 120

APB2ENR
 RCC_MemoryMapType, 122

APB2ENR_ADC1EN
 ID options, 92

APB2ENR_SDIOEN
 ID options, 92

APB2ENR_SPI1EN
 ID options, 92

APB2ENR_SPI4EN
 ID options, 91

APB2ENR_SYSCFGEN
 ID options, 91

APB2ENR_TIM10EN
 ID options, 91

APB2ENR_TIM11EN
 ID options, 90

APB2ENR_TIM1EN
 ID options, 93

APB2ENR_TIM9EN
 ID options, 91

APB2ENR_USART1EN
 ID options, 93

APB2ENR_USART6EN
 ID options, 92

APB2LPENR
 RCC_MemoryMapType, 123

APB2RSTR
 RCC_MemoryMapType, 120

AssertRequest
 Systick Exception (Interrupt) Status, 99

BDCR
 RCC_MemoryMapType, 124

BFAR
 SCB_MemoryMapType, 131

Bit Group Manipulation Math Macros, 10
 CLR_BITs, 11
 GET_BITs, 11
 SET_BITs, 11
 TOGGLE_BITs, 11

Bit Manipulation Math Macros, 9
 CLR_BIT, 9
 GET_BIT, 10
 SET_BIT, 9
 TOGGLE_BIT, 10

bool_t
 Standard types, 15

BSRRx
 GPIOx_MemoryMapType, 108

BUS_FAULT
 NVIC Settable Priorities, 81

BUSY_TICK_TIME
 SysTick_interface.h, 313

BYBASED
 MRCC_private.h, 273

CALIB
 SysTick_Type, 132

CCR
 SCB_MemoryMapType, 128

CFG_R
 RCC_MemoryMapType, 119

CFSR
 SCB_MemoryMapType, 129

CIR
 RCC_MemoryMapType, 119

CLK_SOURCE
 GPIO Addresses, 46

CLKSOURCE
 Systick Register Bits Positions, 97

CLR_BIT
 Bit Manipulation Math Macros, 9

CLR_BITs
 Bit Group Manipulation Math Macros, 11

CMPCR
 MSYSCFG_MemMap_t, 112

Compiler standard macros, 12
 CONST, 14
 CONSTP2CONST, 13
 CONSTP2VAR, 13
 FUNC, 12
 P2CONST, 13
 P2FUNC, 14
 P2VAR, 13
 STATIC, 14
 VAR, 12

CONST
 Compiler standard macros, 14

CONSTP2CONST
 Compiler standard macros, 13

CONSTP2VAR
 Compiler standard macros, 13

COTS/HAL/DCMOTOR/DCM_config.h, 133

COTS/HAL/DCMOTOR/DCM_interface.h, 134, 136

COTS/HAL/DCMOTOR/DCM_private.h, 136

COTS/HAL/DCMOTOR/DCM_program.c, 137, 139

COTS/LIB/LSTD_BITMATH.h, 140

COTS/LIB/LSTD_COMPILER.h, 141

COTS/LIB/LSTD MCU UTILITIES.h, 142

COTS/LIB/LSTD TYPES.h, 142, 143

COTS/LIB/LSTD VALUES.h, 143, 144

COTS/MCAL/EXTI/EXTI_config.h, 145, 151

COTS/MCAL/EXTI/EXTI_interface.h, 154, 160

COTS/MCAL/EXTI/EXTI_private.h, 161, 162

COTS/MCAL/EXTI/EXTI_program.c, 163, 171

COTS/MCAL/EXTI/SYSCFG_private.h, 177, 178

COTS/MCAL/GPIO/GPIO_config.h, 178, 179

COTS/MCAL/GPIO/GPIO_interface.h, 180, 188

COTS/MCAL/GPIO/GPIO_private.h, 190, 191

COTS/MCAL/GPIO/GPIO_program.c, 192, 200

COTS/MCAL/NVIC/NVIC_config.h, 205

COTS/MCAL/NVIC/NVIC_interface.h, 205, 213

COTS/MCAL/NVIC/NVIC_private.h, 215, 217

COTS/MCAL/NVIC/NVIC_program.c, 217, 221

COTS/MCAL/RCC/MRCC_config.h, 224, 235

COTS/MCAL/RCC/MRCC_interface.h, 242, 250

COTS/MCAL/RCC/MRCC_private.h, 251, 293

COTS/MCAL/RCC/MRCC_program.c, 297, 304

COTS/MCAL/SysTick/SysTick_config.h, 310, 311

COTS/MCAL/SysTick/SysTick_interface.h, 311, 319
COTS/MCAL/SysTick/SysTick_private.h, 319, 320
COTS/MCAL/SysTick/SysTick_program.c, 321, 329
COUNTER_ENABLE
 Systick Register Bits Positions, 97
COUNTFLAG
 Systick Register Bits Positions, 96
CPUID
 SCB_MemoryMapType, 126
CR
 RCC_MemoryMapType, 118
CRCCLK
 MRCC_config.h, 229
CSR
 RCC_MemoryMapType, 124
CSS
 MRCC_config.h, 225
CTRL
 SysTick_Type, 132
D:/GProject/ADAS_Project/README.md, 133
DCKCFG
 RCC_MemoryMapType, 125
DCM_interface.h
 HDCM_vInitMotor, 134
 HDCM_vMoveBackward, 135
 HDCM_vMoveForward, 135
 HDCM_vStopMotor, 135
DCM_MotorConfiguration, 101
 Side1Pin, 102
 Side1Port, 101
 Side2Pin, 102
 Side2Port, 102
DCM_program.c
 HDCM_vInitMotor, 137
 HDCM_vMoveBackward, 138
 HDCM_vMoveForward, 138
 HDCM_vStopMotor, 138
Delay Units, 94
 MICRO_SEC, 95
 MILLI_SEC, 95
 SEC, 95
DISABLE
 EXTI Lines Status, 28
 MRCC_private.h, 272
DivisionBy2
 MRCC_private.h, 274
DivisionBy3
 MRCC_private.h, 274
DivisionBy4
 MRCC_private.h, 274
DivisionBy5
 MRCC_private.h, 274
DMA1_STREAM0
 NVIC Vector Table, 67
DMA1_STREAM1
 NVIC Vector Table, 67
DMA1_STREAM2
 NVIC Vector Table, 68
DMA1_STREAM3
 NVIC Vector Table, 68
DMA1_STREAM4
 NVIC Vector Table, 68
DMA1_STREAM5
 NVIC Vector Table, 68
DMA1_STREAM6
 NVIC Vector Table, 69
DMA1_STREAM7
 NVIC Vector Table, 74
DMA1CLK
 MRCC_config.h, 229
DMA2_STREAM0
 NVIC Vector Table, 75
DMA2_STREAM1
 NVIC Vector Table, 75
DMA2_STREAM2
 NVIC Vector Table, 75
DMA2_STREAM3
 NVIC Vector Table, 76
DMA2_STREAM4
 NVIC Vector Table, 76
DMA2_STREAM5
 NVIC Vector Table, 76
DMA2_STREAM6
 NVIC Vector Table, 77
DMA2_STREAM7
 NVIC Vector Table, 77
DMA2CLK
 MRCC_config.h, 229
Dont AssertRequest
 Systick Exception (Interrupt) Status, 98
EMR
 EXTI_Type, 105
ENABLE
 EXTI Lines Status, 28
 MRCC_private.h, 272
Equal_0
 MRCC_private.h, 282
Equal_1
 MRCC_private.h, 283
Equal_10
 MRCC_private.h, 284
Equal_11
 MRCC_private.h, 284
Equal_12
 MRCC_private.h, 284
Equal_13
 MRCC_private.h, 285
Equal_14
 MRCC_private.h, 285
Equal_15
 MRCC_private.h, 285
Equal_16
 MRCC_private.h, 285
Equal_17
 MRCC_private.h, 285
Equal_18
 MRCC_private.h, 285

MRCC_private.h, 285
Equal_19
 MRCC_private.h, 286
Equal_2
 MRCC_private.h, 283
Equal_20
 MRCC_private.h, 286
Equal_21
 MRCC_private.h, 286
Equal_22
 MRCC_private.h, 286
Equal_23
 MRCC_private.h, 286
Equal_24
 MRCC_private.h, 286
Equal_25
 MRCC_private.h, 287
Equal_26
 MRCC_private.h, 287
Equal_27
 MRCC_private.h, 287
Equal_28
 MRCC_private.h, 287
Equal_29
 MRCC_private.h, 287
Equal_3
 MRCC_private.h, 283
Equal_30
 MRCC_private.h, 287
Equal_31
 MRCC_private.h, 288
Equal_32
 MRCC_private.h, 288
Equal_33
 MRCC_private.h, 288
Equal_34
 MRCC_private.h, 288
Equal_35
 MRCC_private.h, 288
Equal_36
 MRCC_private.h, 288
Equal_37
 MRCC_private.h, 289
Equal_38
 MRCC_private.h, 289
Equal_39
 MRCC_private.h, 289
Equal_4
 MRCC_private.h, 283
Equal_40
 MRCC_private.h, 289
Equal_41
 MRCC_private.h, 289
Equal_42
 MRCC_private.h, 289
Equal_43
 MRCC_private.h, 290
Equal_44

 MRCC_private.h, 290
Equal_45
 MRCC_private.h, 290
Equal_46
 MRCC_private.h, 290
Equal_47
 MRCC_private.h, 290
Equal_48
 MRCC_private.h, 290
Equal_49
 MRCC_private.h, 291
Equal_5
 MRCC_private.h, 283
Equal_50
 MRCC_private.h, 291
Equal_51
 MRCC_private.h, 291
Equal_52
 MRCC_private.h, 291
Equal_53
 MRCC_private.h, 291
Equal_54
 MRCC_private.h, 291
Equal_55
 MRCC_private.h, 292
Equal_56
 MRCC_private.h, 292
Equal_57
 MRCC_private.h, 292
Equal_58
 MRCC_private.h, 292
Equal_59
 MRCC_private.h, 292
Equal_6
 MRCC_private.h, 283
Equal_60
 MRCC_private.h, 292
Equal_61
 MRCC_private.h, 293
Equal_62
 MRCC_private.h, 293
Equal_63
 MRCC_private.h, 293
Equal_7
 MRCC_private.h, 284
Equal_8
 MRCC_private.h, 284
Equal_9
 MRCC_private.h, 284
Exception_Request
 SysTick_config.h, 310
EXTI Line Settings, 28
 EXTI_MAX_EXTI_NUM, 28
EXTI Lines Status, 28
 DISABLE, 28
 ENABLE, 28
EXTI Memory Addresses, 27
 EXTI_BASE_ADDRESS, 27

EXTI Registers, 27
 MEXTI, 27
EXTI0
 NVIC Vector Table, 66
EXTI0_IRQHandler
 EXTI_program.c, 169
EXTI1
 NVIC Vector Table, 66
EXTI15_10
 NVIC Vector Table, 73
EXTI15_10_IRQHandler
 EXTI_program.c, 171
EXTI16
 NVIC Vector Table, 65
EXTI17
 NVIC Vector Table, 73
EXTI18
 NVIC Vector Table, 74
EXTI1_IRQHandler
 EXTI_program.c, 169
EXTI2
 NVIC Vector Table, 66
EXTI21
 NVIC Vector Table, 65
EXTI22
 NVIC Vector Table, 65
EXTI2_IRQHandler
 EXTI_program.c, 169
EXTI3
 NVIC Vector Table, 67
EXTI3_IRQHandler
 EXTI_program.c, 170
EXTI4
 NVIC Vector Table, 67
EXTI4_IRQHandler
 EXTI_program.c, 170
EXTI9
 NVIC Vector Table, 69
EXTI9_5_IRQHandler
 EXTI_program.c, 170
EXTI_BASE_ADDRESS
 EXTI Memory Addresses, 27
EXTI_config.h
 EXTI_LINE0_EN, 145
 EXTI_LINE0_TRIGGER, 145
 EXTI_LINE10_EN, 149
 EXTI_LINE10_TRIGGER, 149
 EXTI_LINE11_EN, 149
 EXTI_LINE11_TRIGGER, 149
 EXTI_LINE12_EN, 149
 EXTI_LINE12_TRIGGER, 150
 EXTI_LINE13_EN, 150
 EXTI_LINE13_TRIGGER, 150
 EXTI_LINE14_EN, 150
 EXTI_LINE14_TRIGGER, 150
 EXTI_LINE15_EN, 150
 EXTI_LINE15_TRIGGER, 151
 EXTI_LINE1_EN, 146
 EXTI_LINE1_TRIGGER, 146
 EXTI_LINE2_EN, 146
 EXTI_LINE2_TRIGGER, 146
 EXTI_LINE3_EN, 146
 EXTI_LINE3_TRIGGER, 147
 EXTI_LINE4_EN, 147
 EXTI_LINE4_TRIGGER, 147
 EXTI_LINE5_EN, 147
 EXTI_LINE5_TRIGGER, 147
 EXTI_LINE6_EN, 147
 EXTI_LINE6_TRIGGER, 148
 EXTI_LINE7_EN, 148
 EXTI_LINE7_TRIGGER, 148
 EXTI_LINE8_EN, 148
 EXTI_LINE8_TRIGGER, 148
 EXTI_LINE9_EN, 148
 EXTI_LINE9_TRIGGER, 149
 EXTI_ConfigType, 103
 LineNum, 103
 PortNum, 103
 TriggerStatus, 103
 EXTI_FallingEdge
 Interrupt trigger status, 26
 EXTI_interface.h
 MEXTI_vDisableLine, 159
 MEXTI_vEnableLine, 158
 MEXTI_vInit, 155
 MEXTI_vInit_WithStruct, 158
 MEXTI_vSetCallback, 160
 MEXTI_vSetTrigger, 159
 MEXTI_vSWITrigger, 159
 MSYSCFG_vSetEXTIPort, 160
 EXTI IRQs
 EXTI_private.h, 162
 EXTI_LINE0
 Interrupt line IDs, 22
 EXTI_LINE0_EN
 EXTI_config.h, 145
 EXTI_LINE0_TRIGGER
 EXTI_config.h, 145
 EXTI_LINE1
 Interrupt line IDs, 22
 EXTI_LINE10
 Interrupt line IDs, 24
 EXTI_LINE10_EN
 EXTI_config.h, 149
 EXTI_LINE10_TRIGGER
 EXTI_config.h, 149
 EXTI_LINE11
 Interrupt line IDs, 24
 EXTI_LINE11_EN
 EXTI_config.h, 149
 EXTI_LINE11_TRIGGER
 EXTI_config.h, 149
 EXTI_LINE12
 Interrupt line IDs, 24
 EXTI_LINE12_EN
 EXTI_config.h, 149

EXTI_LINE12_TRIGGER
 EXTI_config.h, 150
EXTI_LINE13
 Interrupt line IDs, 25
EXTI_LINE13_EN
 EXTI_config.h, 150
EXTI_LINE13_TRIGGER
 EXTI_config.h, 150
EXTI_LINE14
 Interrupt line IDs, 25
EXTI_LINE14_EN
 EXTI_config.h, 150
EXTI_LINE14_TRIGGER
 EXTI_config.h, 150
EXTI_LINE15
 Interrupt line IDs, 25
EXTI_LINE15_EN
 EXTI_config.h, 150
EXTI_LINE15_TRIGGER
 EXTI_config.h, 151
EXTI_LINE1_EN
 EXTI_config.h, 146
EXTI_LINE1_TRIGGER
 EXTI_config.h, 146
EXTI_LINE2
 Interrupt line IDs, 22
EXTI_LINE2_EN
 EXTI_config.h, 146
EXTI_LINE2_TRIGGER
 EXTI_config.h, 146
EXTI_LINE3
 Interrupt line IDs, 22
EXTI_LINE3_EN
 EXTI_config.h, 146
EXTI_LINE3_TRIGGER
 EXTI_config.h, 147
EXTI_LINE4
 Interrupt line IDs, 22
EXTI_LINE4_EN
 EXTI_config.h, 147
EXTI_LINE4_TRIGGER
 EXTI_config.h, 147
EXTI_LINE5
 Interrupt line IDs, 23
EXTI_LINE5_EN
 EXTI_config.h, 147
EXTI_LINE5_TRIGGER
 EXTI_config.h, 147
EXTI_LINE6
 Interrupt line IDs, 23
EXTI_LINE6_EN
 EXTI_config.h, 147
EXTI_LINE6_TRIGGER
 EXTI_config.h, 148
EXTI_LINE7
 Interrupt line IDs, 23
EXTI_LINE7_EN
 EXTI_config.h, 148
EXTI_LINE7_TRIGGER
 EXTI_config.h, 148
EXTI_LINE8
 Interrupt line IDs, 23
EXTI_LINE8_EN
 EXTI_config.h, 148
EXTI_LINE8_TRIGGER
 EXTI_config.h, 148
EXTI_LINE9
 Interrupt line IDs, 24
EXTI_LINE9_EN
 EXTI_config.h, 148
EXTI_LINE9_TRIGGER
 EXTI_config.h, 149
EXTI_MAX_EXTI_NUM
 EXTI Line Settings, 28
EXTI_OnChange
 Interrupt trigger status, 26
EXTI_private.h
 EXTI IRQs, 162
EXTI_program.c
 EXTI0_IRQHandler, 169
 EXTI15_10_IRQHandler, 171
 EXTI1_IRQHandler, 169
 EXTI2_IRQHandler, 169
 EXTI3_IRQHandler, 170
 EXTI4_IRQHandler, 170
 EXTI9_5_IRQHandler, 170
 MEXTI_vDisableLine, 167
 MEXTI_vEnableLine, 167
 MEXTI_vInit, 164
 MEXTI_vInit_WithStruct, 167
 MEXTI_vSetCallback, 168
 MEXTI_vSetTrigger, 168
 MEXTI_vSWITrigger, 167
 MSYSCFG_vSetEXTIPort, 164
EXTI_RisingEdge
 Interrupt trigger status, 26
EXTI_Type, 104
 EMR, 105
 FTSR, 105
 IMR, 104
 PR, 105
 RTSR, 105
 SWIER, 105
EXTICR
 MSYSCFG_MemMap_t, 112
FALSE
 Standard values, 18
f32_t
 Standard types, 17
f64_t
 Standard types, 17
FLAG_CLEARED
 Standard values, 19
FLAG_SET
 Standard values, 19
FLASH

NVIC Vector Table, 65
FPU
 NVIC Vector Table, 78
FTSR
 EXTI_Type, 105
FUNC
 Compiler standard macros, 12
GET_BIT
 Bit Manipulation Math Macros, 10
GET_BITS
 Bit Group Manipulation Math Macros, 11
GPIO Addresses, 45
 CLK_SOURCE, 46
 GPIOA_BASE_ADDRESS, 45
 GPIOB_BASE_ADDRESS, 46
 GPIOC_BASE_ADDRESS, 46
GPIO Alternate Functions, 41
 GPIOx_AF0, 41
 GPIOx_AF1, 41
 GPIOx_AF10, 44
 GPIOx_AF11, 44
 GPIOx_AF12, 44
 GPIOx_AF13, 44
 GPIOx_AF14, 45
 GPIOx_AF15, 45
 GPIOx_AF2, 42
 GPIOx_AF3, 42
 GPIOx_AF4, 42
 GPIOx_AF5, 42
 GPIOx_AF6, 43
 GPIOx_AF7, 43
 GPIOx_AF8, 43
 GPIOx_AF9, 43
GPIO Modes, 30
 GPIOx_MODE_AF, 30
 GPIOx_MODE_ANALOG, 31
 GPIOx_MODE_INPUT, 30
 GPIOx_MODE_OUTPUT, 30
GPIO Output Pins, 36
 GPIOx_PIN0, 37
 GPIOx_PIN1, 37
 GPIOx_PIN10, 39
 GPIOx_PIN11, 39
 GPIOx_PIN12, 39
 GPIOx_PIN13, 40
 GPIOx_PIN14, 40
 GPIOx_PIN15, 40
 GPIOx_PIN2, 37
 GPIOx_PIN3, 37
 GPIOx_PIN4, 37
 GPIOx_PIN5, 38
 GPIOx_PIN6, 38
 GPIOx_PIN7, 38
 GPIOx_PIN8, 38
 GPIOx_PIN9, 39
GPIO Output Types, 32
 GPIOx_OPENDRAIN, 32
 GPIOx_PUSH_PULL, 32
 GPIO Output Values, 35
 GPIOx_HIGH, 35
 GPIOx_LOW, 35
 GPIO PIN Speed, 33
 GPIOx_HighSpeed, 33
 GPIOx_LowSpeed, 33
 GPIOx_MediumSpeed, 33
 GPIOx_VeryHighSpeed, 34
 GPIO Ports, 31
 GPIO_PORTA, 31
 GPIO_PORTB, 31
 GPIO_PORTC, 32
 GPIO Pull Types, 34
 GPIOx_NoPull, 34
 GPIOx_PullDown, 35
 GPIOx_PullUp, 34
 GPIO Registers, 46
 GPIOA, 47
 GPIOB, 47
 GPIOC, 47
 GPIO_config.h
 GPIOA_PIN_POS, 179
 GPIOB_PIN_POS, 179
 LCKK_BIT_POS, 179
 GPIO_interface.h
 MGPIOx_u8GetPinValue, 185
 MGPIOx_vInit, 188
 MGPIOx_vLockedPins, 183
 MGPIOx_vSetAlternateFunctionON, 186
 MGPIOx_vSetPinInputPullType, 185
 MGPIOx_vSetPinMode, 183
 MGPIOx_vSetPinOutputSpeed, 184
 MGPIOx_vSetPinOutputType, 184
 MGPIOx_vSetPinValue, 185
 MGPIOx_vSetPortConfigLock, 186
 MGPIOx_vSetResetAtomic, 186
 MGPIOx_vTogglePinValue, 188
 GPIO_PORTA
 GPIO Ports, 31
 GPIO_PORTB
 GPIO Ports, 31
 GPIO_PORTC
 GPIO Ports, 32
 GPIO_program.c
 MGPIOx_u8GetPinValue, 195
 MGPIOx_vInit, 199
 MGPIOx_vLockedPins, 192
 MGPIOx_vSetAlternateFunctionON, 197
 MGPIOx_vSetPinInputPullType, 195
 MGPIOx_vSetPinMode, 193
 MGPIOx_vSetPinOutputSpeed, 194
 MGPIOx_vSetPinOutputType, 194
 MGPIOx_vSetPinValue, 196
 MGPIOx_vSetResetAtomic, 197
 MGPIOx_vTogglePinValue, 199
 GPIOA
 GPIO Registers, 47
 GPIOA_BASE_ADDRESS

GPIO Addresses, 45
GPIOA_PIN_POS
 GPIO_config.h, 179
GPIOACLK
 MRCC_config.h, 230
GPIOB
 GPIO Registers, 47
GPIOB_BASE_ADDRESS
 GPIO Addresses, 46
GPIOB_PIN_POS
 GPIO_config.h, 179
GPIOBCLK
 MRCC_config.h, 230
GPIOC
 GPIO Registers, 47
GPIOC_BASE_ADDRESS
 GPIO Addresses, 46
GPIOCCLK
 MRCC_config.h, 230
GPIODCLK
 MRCC_config.h, 230
GPIOECLK
 MRCC_config.h, 229
GPIOHCLK
 MRCC_config.h, 229
GPIOx_AF0
 GPIO Alternate Functions, 41
GPIOx_AF1
 GPIO Alternate Functions, 41
GPIOx_AF10
 GPIO Alternate Functions, 44
GPIOx_AF11
 GPIO Alternate Functions, 44
GPIOx_AF12
 GPIO Alternate Functions, 44
GPIOx_AF13
 GPIO Alternate Functions, 44
GPIOx_AF14
 GPIO Alternate Functions, 45
GPIOx_AF15
 GPIO Alternate Functions, 45
GPIOx_AF2
 GPIO Alternate Functions, 42
GPIOx_AF3
 GPIO Alternate Functions, 42
GPIOx_AF4
 GPIO Alternate Functions, 42
GPIOx_AF5
 GPIO Alternate Functions, 42
GPIOx_AF6
 GPIO Alternate Functions, 43
GPIOx_AF7
 GPIO Alternate Functions, 43
GPIOx_AF8
 GPIO Alternate Functions, 43
GPIOx_AF9
 GPIO Alternate Functions, 43
GPIOx_HIGH
 GPIO Output Values, 35
GPIOx_HighSpeed
 GPIO PIN Speed, 33
GPIOx_LOW
 GPIO Output Values, 35
GPIOx_LowSpeed
 GPIO PIN Speed, 33
GPIOx_MediumSpeed
 GPIO PIN Speed, 33
GPIOx_MemoryMapType, 106
 AFRHx, 108
 AFRLx, 108
 BSRRx, 108
 IDRx, 107
 LCKRx, 108
 MODERx, 107
 ODRx, 107
 OSPEEDRx, 107
 OTYPERx, 107
 PUPDRx, 107
GPIOx_MODE_AF
 GPIO Modes, 30
GPIOx_MODE_ANALOG
 GPIO Modes, 31
GPIOx_MODE_INPUT
 GPIO Modes, 30
GPIOx_MODE_OUTPUT
 GPIO Modes, 30
GPIOx_NoPull
 GPIO Pull Types, 34
GPIOx_OPENDRAIN
 GPIO Output Types, 32
GPIOx_PIN0
 GPIO Output PINs, 37
GPIOx_PIN1
 GPIO Output PINs, 37
GPIOx_PIN10
 GPIO Output PINs, 39
GPIOx_PIN11
 GPIO Output PINs, 39
GPIOx_PIN12
 GPIO Output PINs, 39
GPIOx_PIN13
 GPIO Output PINs, 40
GPIOx_PIN14
 GPIO Output PINs, 40
GPIOx_PIN15
 GPIO Output PINs, 40
GPIOx_PIN2
 GPIO Output PINs, 37
GPIOx_PIN3
 GPIO Output PINs, 37
GPIOx_PIN4
 GPIO Output PINs, 37
GPIOx_PIN5
 GPIO Output PINs, 38
GPIOx_PIN6
 GPIO Output PINs, 38

GPIOx_PIN7
 GPIO Output PINs, 38
GPIOx_PIN8
 GPIO Output PINs, 38
GPIOx_PIN9
 GPIO Output PINs, 39
GPIOx_PullDown
 GPIO Pull Types, 35
GPIOx_PullUp
 GPIO Pull Types, 34
GPIOx_PUSH_PULL
 GPIO Output Types, 32
GPIOx_VeryHighSpeed
 GPIO PIN Speed, 34
Group priorities, 51
 GROUP_PRIORITY_0, 53
 GROUP_PRIORITY_1, 53
 GROUP_PRIORITY_10, 55
 GROUP_PRIORITY_11, 55
 GROUP_PRIORITY_12, 56
 GROUP_PRIORITY_13, 56
 GROUP_PRIORITY_14, 56
 GROUP_PRIORITY_15, 56
 GROUP_PRIORITY_2, 53
 GROUP_PRIORITY_3, 53
 GROUP_PRIORITY_4, 54
 GROUP_PRIORITY_5, 54
 GROUP_PRIORITY_6, 54
 GROUP_PRIORITY_7, 54
 GROUP_PRIORITY_8, 55
 GROUP_PRIORITY_9, 55
 NO_GROUP_PRIORITY, 52
GROUP_0BITS
 Interrupt priority grouping, 51
GROUP_1BITS
 Interrupt priority grouping, 51
GROUP_2BITS
 Interrupt priority grouping, 50
GROUP_3BITS
 Interrupt priority grouping, 50
GROUP_4BITS
 Interrupt priority grouping, 50
GROUP_PRIORITY_0
 Group priorities, 53
GROUP_PRIORITY_1
 Group priorities, 53
GROUP_PRIORITY_10
 Group priorities, 55
GROUP_PRIORITY_11
 Group priorities, 55
GROUP_PRIORITY_12
 Group priorities, 56
GROUP_PRIORITY_13
 Group priorities, 56
GROUP_PRIORITY_14
 Group priorities, 56
GROUP_PRIORITY_15
 Group priorities, 56
GROUP_PRIORITY_2
 Group priorities, 53
GROUP_PRIORITY_3
 Group priorities, 53
GROUP_PRIORITY_4
 Group priorities, 54
GROUP_PRIORITY_5
 Group priorities, 54
GROUP_PRIORITY_6
 Group priorities, 54
GROUP_PRIORITY_7
 Group priorities, 54
GROUP_PRIORITY_8
 Group priorities, 55
GROUP_PRIORITY_9
 Group priorities, 55
HDCM_vInitMotor
 DCM_interface.h, 134
 DCM_program.c, 137
HDCM_vMoveBackward
 DCM_interface.h, 135
 DCM_program.c, 138
HDCM_vMoveForward
 DCM_interface.h, 135
 DCM_program.c, 138
HDCM_vStopMotor
 DCM_interface.h, 135
 DCM_program.c, 138
HFSR
 SCB_MemoryMapType, 130
HPRE
 MRCC_config.h, 228
HSE
 MRCC_private.h, 273
HSE_EN
 MRCC_config.h, 226
HSEby10
 MRCC_private.h, 278
HSEby11
 MRCC_private.h, 278
HSEby12
 MRCC_private.h, 278
HSEby13
 MRCC_private.h, 279
HSEby14
 MRCC_private.h, 279
HSEby15
 MRCC_private.h, 279
HSEby16
 MRCC_private.h, 279
HSEby17
 MRCC_private.h, 279
HSEby18
 MRCC_private.h, 279
HSEby19
 MRCC_private.h, 280
HSEby2
 MRCC_private.h, 277

HSEby20
 MRCC_private.h, 280

HSEby21
 MRCC_private.h, 280

HSEby22
 MRCC_private.h, 280

HSEby23
 MRCC_private.h, 280

HSEby24
 MRCC_private.h, 280

HSEby25
 MRCC_private.h, 281

HSEby26
 MRCC_private.h, 281

HSEby27
 MRCC_private.h, 281

HSEby28
 MRCC_private.h, 281

HSEby29
 MRCC_private.h, 281

HSEby3
 MRCC_private.h, 277

HSEby30
 MRCC_private.h, 281

HSEby31
 MRCC_private.h, 282

HSEby4
 MRCC_private.h, 277

HSEby5
 MRCC_private.h, 277

HSEby6
 MRCC_private.h, 277

HSEby7
 MRCC_private.h, 278

HSEby8
 MRCC_private.h, 278

HSEby9
 MRCC_private.h, 278

HSEBYP
 MRCC_config.h, 226

HSI
 MRCC_private.h, 275

HSI_EN
 MRCC_config.h, 226

I2C1_ER
 NVIC Vector Table, 71

I2C1_EV
 NVIC Vector Table, 71

I2C1EN
 MRCC_config.h, 231

I2C2_ER
 NVIC Vector Table, 72

I2C2_EV
 NVIC Vector Table, 72

I2C2EN
 MRCC_config.h, 231

I2C3_ER
 NVIC Vector Table, 78

I2C3_EV
 NVIC Vector Table, 77

I2C3EN
 MRCC_config.h, 231

I2S_CKIN
 MRCC_private.h, 274

I2SSRC
 MRCC_config.h, 227

IABRx
 NVIC_MemoryMapType, 116

ICERx
 NVIC_MemoryMapType, 114

ICPRx
 NVIC_MemoryMapType, 115

ICSR
 SCB_MemoryMapType, 127

ID options, 82, 84
 AHB1ENR_CRCEN, 85
 AHB1ENR_DMA1EN, 85
 AHB1ENR_DMA2EN, 85
 AHB1ENR_GPIOAEN, 87
 AHB1ENR_GPIOBEN, 87
 AHB1ENR_GPIOCEN, 86
 AHB1ENR_GPIODEN, 86
 AHB1ENR_GPIOEEN, 86
 AHB1ENR_GPIOHEN, 86
 AHB1LPENR_FLITFLPEN, 93
 AHB2ENR_OTGFSEN, 87
 APB1ENR_I2C1EN, 88
 APB1ENR_I2C2EN, 88
 APB1ENR_I2C3EN, 88
 APB1ENR_PWREN, 87
 APB1ENR_SPI2EN, 89
 APB1ENR_SPI3EN, 89
 APB1ENR_TIM2EN, 90
 APB1ENR_TIM3EN, 90
 APB1ENR_TIM4EN, 90
 APB1ENR_TIM5EN, 89
 APB1ENR_USART2EN, 88
 APB1ENR_WWDGEN, 89
 APB2ENR_ADC1EN, 92
 APB2ENR_SDIOEN, 92
 APB2ENR_SPI1EN, 92
 APB2ENR_SPI4EN, 91
 APB2ENR_SYSCFGEN, 91
 APB2ENR_TIM10EN, 91
 APB2ENR_TIM11EN, 90
 APB2ENR_TIM1EN, 93
 APB2ENR_TIM9EN, 91
 APB2ENR_USART1EN, 93
 APB2ENR_USART6EN, 92
 RCC_AHB1, 82
 RCC_AHB1LPENR, 83
 RCC_AHB2, 82
 RCC_APB1, 83
 RCC_APB2, 83

IDRx
 GPIOx_MemoryMapType, 107

IMR
 EXTI_Type, 104
INITIAL_ZERO
 Standard values, 19
InputType
 MGPIOx_ConfigType, 110
Interrupt line IDs, 21
 EXTI_LINE0, 22
 EXTI_LINE1, 22
 EXTI_LINE10, 24
 EXTI_LINE11, 24
 EXTI_LINE12, 24
 EXTI_LINE13, 25
 EXTI_LINE14, 25
 EXTI_LINE15, 25
 EXTI_LINE2, 22
 EXTI_LINE3, 22
 EXTI_LINE4, 22
 EXTI_LINE5, 23
 EXTI_LINE6, 23
 EXTI_LINE7, 23
 EXTI_LINE8, 23
 EXTI_LINE9, 24
Interrupt priority grouping, 47
 _16GROUP_NoSub_Priorities, 48
 _2GROUP_8Sub_Priorities, 49
 _4GROUP_4Sub_Priorities, 49
 _8GROUP_2Sub_Priorities, 48
GROUP_0BITS, 51
GROUP_1BITS, 51
GROUP_2BITS, 50
GROUP_3BITS, 50
GROUP_4BITS, 50
NoGROUP_16Sub_Priorities, 49
Interrupt trigger status, 26
 EXTI_FallingEdge, 26
 EXTI_OnChange, 26
 EXTI_RisingEdge, 26
IPRx
 NVIC_MemoryMapType, 116
ISERx
 NVIC_MemoryMapType, 113
ISPRx
 NVIC_MemoryMapType, 114
LCKK_BIT_POS
 GPIO_config.h, 179
LCKRx
 GPIOx_MemoryMapType, 108
LineNum
 EXTI_ConfigType, 103
LOAD
 SysTick_Type, 132
LSE
 MRCC_private.h, 275
MAX_TICKS
 SysTick_config.h, 311
MCO1
 MRCC_config.h, 228
MCO1PRE
 MRCC_config.h, 227
MCO2
 MRCC_config.h, 227
MCO2PRE
 MRCC_config.h, 227
MEMORY_MANAGE
 NVIC_Settable Priorities, 81
MEMRMP
 MSYSCFG_MemMap_t, 112
MEXTI
 EXTI Registers, 27
MEXTI_vDisableLine
 EXTI_interface.h, 159
 EXTI_program.c, 167
MEXTI_vEnableLine
 EXTI_interface.h, 158
 EXTI_program.c, 167
MEXTI_vInit
 EXTI_interface.h, 155
 EXTI_program.c, 164
MEXTI_vInit_WithStruct
 EXTI_interface.h, 158
 EXTI_program.c, 167
MEXTI_vSetCallback
 EXTI_interface.h, 160
 EXTI_program.c, 168
MEXTI_vSetTrigger
 EXTI_interface.h, 159
 EXTI_program.c, 168
MEXTI_vSWITrigger
 EXTI_interface.h, 159
 EXTI_program.c, 167
MGPIOx_ConfigType, 109
 AF_Type, 111
 InputType, 110
 Mode, 110
 OutputSpeed, 110
 OutputType, 110
 Pin, 109
 Port, 109
MGPIOx_u8GetPinValue
 GPIO_interface.h, 185
 GPIO_program.c, 195
MGPIOx_vInit
 GPIO_interface.h, 188
 GPIO_program.c, 199
MGPIOx_vLockedPins
 GPIO_interface.h, 183
 GPIO_program.c, 192
MGPIOx_vSetAlternateFunctionON
 GPIO_interface.h, 186
 GPIO_program.c, 197
MGPIOx_vSetPinInputPullType
 GPIO_interface.h, 185
 GPIO_program.c, 195
MGPIOx_vSetPinMode

GPIO_interface.h, 183
 GPIO_program.c, 193
MGPIOx_vSetPinOutputSpeed
 GPIO_interface.h, 184
 GPIO_program.c, 194
MGPIOx_vSetPinOutputType
 GPIO_interface.h, 184
 GPIO_program.c, 194
MGPIOx_vSetPinValue
 GPIO_interface.h, 185
 GPIO_program.c, 196
MGPIOx_vSetPortConfigLock
 GPIO_interface.h, 186
MGPIOx_vSetResetAtomic
 GPIO_interface.h, 186
 GPIO_program.c, 197
MGPIOx_vTogglePinValue
 GPIO_interface.h, 188
 GPIO_program.c, 199
MICRO_SEC
 Delay Units, 95
 MILLI_SEC
 Delay Units, 95
MMFAR
 SCB_MemoryMapType, 130
MNVIC
 NVIC Registers, 80
MNVIC_STIR
 NVIC_private.h, 216
MNVIC_u8GetActive
 NVIC_interface.h, 212
 NVIC_program.c, 219
MNVIC_vClearPendingFlag
 NVIC_interface.h, 211
 NVIC_program.c, 219
MNVIC_vDisablePeriphral
 NVIC_interface.h, 211
 NVIC_program.c, 219
MNVIC_vEnablePeriphral
 NVIC_interface.h, 210
 NVIC_program.c, 218
MNVIC_vSetPendingFlag
 NVIC_interface.h, 211
 NVIC_program.c, 219
MNVIC_vSetPriority
 NVIC_interface.h, 213
 NVIC_program.c, 220
MNVIC_vSetPriorityConfig
 NVIC_interface.h, 212
 NVIC_program.c, 220
Mode
 MGPIOx_ConfigType, 110
MODERx
 GPIOx_MemoryMapType, 107
MRCC_config.h
 ADC1EN, 234
 CRCCLK, 229
 CSS, 225
 DMA1CLK, 229
 DMA2CLK, 229
 GPIOACKL, 230
 GPIOBCLK, 230
 GPIOCCLK, 230
 GPIODCLK, 230
 GPIOECLK, 229
 GPIOHCLK, 229
 HPRE, 228
 HSE_EN, 226
 HSEBYP, 226
 HSI_EN, 226
 I2C1EN, 231
 I2C2EN, 231
 I2C3EN, 231
 I2SSRC, 227
 MCO1, 228
 MCO1PRE, 227
 MCO2, 227
 MCO2PRE, 227
 OTGFS, 230
 PLL, 225
 PLL2S, 225
 PLLM, 227
 PLLN, 227
 PLLP, 226
 PLLQ, 226
 PLLSRC, 226
 PPRE1, 228
 PPRE2, 228
 PWREN, 230
 RTCPRE, 228
 SDIOEN, 233
 SPI1EN, 233
 SPI2EN, 231
 SPI3EN, 231
 SPI4EN, 233
 SW, 229
 SWS, 228
 SYSCFGEN, 233
 TIM10EN, 233
 TIM11EN, 232
 TIM1EN, 234
 TIM2EN, 232
 TIM3EN, 232
 TIM4EN, 232
 TIM5EN, 232
 TIM9EN, 233
 USART1EN, 234
 USART2EN, 231
 USART6EN, 234
 WWDGEN, 232
MRCC_interface.h
 MRCC_vDisablePeriphralCLK, 249
 MRCC_vEnablePeriphralCLK, 249
 MRCC_vInit, 245
MRCC_private.h
 AHBby16, 282

AHBby2, 282
AHBby4, 282
AHBby8, 282
BYBASED, 273
DISABLE, 272
DivisionBy2, 274
DivisionBy3, 274
DivisionBy4, 274
DivisionBy5, 274
ENABLE, 272
Equal_0, 282
Equal_1, 283
Equal_10, 284
Equal_11, 284
Equal_12, 284
Equal_13, 285
Equal_14, 285
Equal_15, 285
Equal_16, 285
Equal_17, 285
Equal_18, 285
Equal_19, 286
Equal_2, 283
Equal_20, 286
Equal_21, 286
Equal_22, 286
Equal_23, 286
Equal_24, 286
Equal_25, 287
Equal_26, 287
Equal_27, 287
Equal_28, 287
Equal_29, 287
Equal_3, 283
Equal_30, 287
Equal_31, 288
Equal_32, 288
Equal_33, 288
Equal_34, 288
Equal_35, 288
Equal_36, 288
Equal_37, 289
Equal_38, 289
Equal_39, 289
Equal_4, 283
Equal_40, 289
Equal_41, 289
Equal_42, 289
Equal_43, 290
Equal_44, 290
Equal_45, 290
Equal_46, 290
Equal_47, 290
Equal_48, 290
Equal_49, 291
Equal_5, 283
Equal_50, 291
Equal_51, 291
Equal_52, 291
Equal_53, 291
Equal_54, 291
Equal_55, 292
Equal_56, 292
Equal_57, 292
Equal_58, 292
Equal_59, 292
Equal_6, 283
Equal_60, 292
Equal_61, 293
Equal_62, 293
Equal_63, 293
Equal_7, 284
Equal_8, 284
Equal_9, 284
HSE, 273
HSEby10, 278
HSEby11, 278
HSEby12, 278
HSEby13, 279
HSEby14, 279
HSEby15, 279
HSEby16, 279
HSEby17, 279
HSEby18, 279
HSEby19, 280
HSEby2, 277
HSEby20, 280
HSEby21, 280
HSEby22, 280
HSEby23, 280
HSEby24, 280
HSEby25, 281
HSEby26, 281
HSEby27, 281
HSEby28, 281
HSEby29, 281
HSEby3, 277
HSEby30, 281
HSEby31, 282
HSEby4, 277
HSEby5, 277
HSEby6, 277
HSEby7, 278
HSEby8, 278
HSEby9, 278
HSI, 275
I2S_CKIN, 274
LSE, 275
NoCLK0, 276
NoCLK1, 277
NoDivision, 274
NOTBYBASED, 273
PLLCLK, 273, 275
PLLl2SCLK, 273
RCC, 256
RCC_AHB1ENR_CRCEN, 267

RCC_AHB1ENR_DMA1EN, 267
 RCC_AHB1ENR_DMA2EN, 267
 RCC_AHB1ENR_GPIOAEN, 268
 RCC_AHB1ENR_GPIOBEN, 268
 RCC_AHB1ENR_GPIOCEN, 268
 RCC_AHB1ENR_GPIODEN, 267
 RCC_AHB1ENR_GPIOEEN, 267
 RCC_AHB1ENR_GPIOHEN, 267
 RCC_AHB1LPENR_FLITFLPEN, 272
 RCC_AHB2ENR_OTGFSEN, 268
 RCC_APB1ENR_I2C1EN, 269
 RCC_APB1ENR_I2C2EN, 269
 RCC_APB1ENR_I2C3EN, 268
 RCC_APB1ENR_PWREN, 268
 RCC_APB1ENR_SPI2EN, 269
 RCC_APB1ENR_SPI3EN, 269
 RCC_APB1ENR_TIM2EN, 270
 RCC_APB1ENR_TIM3EN, 270
 RCC_APB1ENR_TIM4EN, 270
 RCC_APB1ENR_TIM5EN, 270
 RCC_APB1ENR_USART2EN, 269
 RCC_APB1ENR_WWDGEN, 269
 RCC_APB2ENR_ADC1EN, 271
 RCC_APB2ENR_SDIOEN, 271
 RCC_APB2ENR_SPI1EN, 271
 RCC_APB2ENR_SPI4EN, 271
 RCC_APB2ENR_SYSCFGEN, 271
 RCC_APB2ENR_TIM10EN, 270
 RCC_APB2ENR_TIM11EN, 270
 RCC_APB2ENR_TIM1EN, 272
 RCC_APB2ENR_TIM9EN, 271
 RCC_APB2ENR_USART1EN, 272
 RCC_APB2ENR_USART6EN, 272
 RCC_BASE_ADDRESS, 256
 RCC_CFGR_HPRE_b0, 265
 RCC_CFGR_HPRE_b1, 265
 RCC_CFGR_HPRE_b2, 266
 RCC_CFGR_HPRE_b3, 266
 RCC_CFGR_I2SSRC, 263
 RCC_CFGR_MOC1_b0, 263
 RCC_CFGR_MOC1_b1, 263
 RCC_CFGR_MOC1PRE_b0, 262
 RCC_CFGR_MOC1PRE_b1, 263
 RCC_CFGR_MOC1PRE_b2, 263
 RCC_CFGR_MOC2_b0, 262
 RCC_CFGR_MOC2_b1, 262
 RCC_CFGR_MOC2PRE_b0, 262
 RCC_CFGR_MOC2PRE_b1, 262
 RCC_CFGR_MOC2PRE_b2, 262
 RCC_CFGR_PPREG1_b0, 265
 RCC_CFGR_PPREG1_b1, 265
 RCC_CFGR_PPREG1_b2, 265
 RCC_CFGR_PPREG2_b0, 264
 RCC_CFGR_PPREG2_b1, 264
 RCC_CFGR_PPREG2_b2, 265
 RCC_CFGR_RTCPRE_b0, 263
 RCC_CFGR_RTCPRE_b1, 264
 RCC_CFGR_RTCPRE_b2, 264
 RCC_CFGR_RTCPRE_b3, 264
 RCC_CFGR_RTCPRE_b4, 264
 RCC_CFGR_SW_b0, 266
 RCC_CFGR_SW_b1, 266
 RCC_CFGR_SWS_b0, 266
 RCC_CFGR_SWS_b1, 266
 RCC_CR_CSSON, 257
 RCC_CR_HSEBYP, 257
 RCC_CR_HSEON, 257
 RCC_CR_HSERDY, 257
 RCC_CR_HSION, 258
 RCC_CR_HSIRDY, 258
 RCC_CR_PLLI2SON, 256
 RCC_CR_PLLI2SRDY, 256
 RCC_CR_PLLON, 257
 RCC_CR_PLLRDY, 257
 RCC_PLLCFG_PLLM_b0, 261
 RCC_PLLCFG_PLLM_b1, 261
 RCC_PLLCFG_PLLM_b2, 261
 RCC_PLLCFG_PLLM_b3, 261
 RCC_PLLCFG_PLLM_b4, 261
 RCC_PLLCFG_PLLM_b5, 261
 RCC_PLLCFG_PLLN_b0, 259
 RCC_PLLCFG_PLLN_b1, 259
 RCC_PLLCFG_PLLN_b2, 259
 RCC_PLLCFG_PLLN_b3, 260
 RCC_PLLCFG_PLLN_b4, 260
 RCC_PLLCFG_PLLN_b5, 260
 RCC_PLLCFG_PLLN_b6, 260
 RCC_PLLCFG_PLLN_b7, 260
 RCC_PLLCFG_PLLN_b8, 260
 RCC_PLLCFG_PLLP_b0, 259
 RCC_PLLCFG_PLLP_b1, 259
 RCC_PLLCFG_PLLQ_b0, 258
 RCC_PLLCFG_PLLQ_b1, 258
 RCC_PLLCFG_PLLQ_b2, 258
 RCC_PLLCFG_PLLQ_b3, 258
 RCC_PLLCFG_PLLSRC, 259
 SYSCLK, 273
 SYSCLKby128, 276
 SYSCLKby16, 276
 SYSCLKby2, 275
 SYSCLKby256, 276
 SYSCLKby4, 275
 SYSCLKby512, 276
 SYSCLKby64, 276
 SYSCLKby8, 275
 MRCC_program.c
 MRCC_vDisablePeriphralCLK, 303
 MRCC_vEnablePeriphralCLK, 303
 MRCC_vInit, 298
 MRCC_vDisablePeriphralCLK
 MRCC_interface.h, 249
 MRCC_program.c, 303
 MRCC_vEnablePeriphralCLK
 MRCC_interface.h, 249
 MRCC_program.c, 303
 MRCC_vInit

MRCC_interface.h, 245
MRCC_program.c, 298
MSCB
 NVIC Registers, 80
MSYSCFG
 SYSCFG Memory Registers, 29
MSYSCFG_MemMap_t, 111
 CMPCR, 112
 EXTICR, 112
 MEMRMP, 112
 PMC, 112
MSYSCFG_vSetEXTIPort
 EXTI_interface.h, 160
 EXTI_program.c, 164
MSysTick_u32GetElapsedTime
 SysTick_interface.h, 317
 SysTick_program.c, 327
MSysTick_u32GetRemainingTime
 SysTick_interface.h, 317
 SysTick_program.c, 327
MSysTick_vDelay
 SysTick_interface.h, 314
 SysTick_program.c, 323
MSysTick_vDelayMicroSec
 SysTick_interface.h, 315
 SysTick_program.c, 324
MSysTick_vDelayMilliSec
 SysTick_interface.h, 315
 SysTick_program.c, 324
MSysTick_vDelaySec
 SysTick_interface.h, 315
 SysTick_program.c, 325
MSysTick_vDisable
 SysTick_interface.h, 318
 SysTick_program.c, 328
MSysTick_vEnable
 SysTick_interface.h, 317
 SysTick_program.c, 328
MSysTick_vEnableException
 SysTick_interface.h, 318
 SysTick_program.c, 322
MSysTick_vInit
 SysTick_interface.h, 314
 SysTick_program.c, 322
MSysTick_vSetBusyWait
 SysTick_interface.h, 314
 SysTick_program.c, 322
MSysTick_vSetPeriodicInterval
 SysTick_interface.h, 316
 SysTick_program.c, 326
MSysTick_vSetSingleInterval
 SysTick_interface.h, 315
 SysTick_program.c, 325
MSysTick_vStopInterval
 SysTick_interface.h, 316
 SysTick_program.c, 326
MY_MS_DELAY
 Utilities macros, 15
NO_GROUP_PRIORITY
 Group priorities, 52
NO_SUB_PRIORITY
 Sub-Group priorities, 58
NoCLK0
 MRCC_private.h, 276
NoCLK1
 MRCC_private.h, 277
NoDivision
 MRCC_private.h, 274
NoGROUP_16Sub_Priorities
 Interrupt priority grouping, 49
NOTBYBASED
 MRCC_private.h, 273
NULL
 Standard values, 19
NVIC Addresses, 78
 NVIC_BASE_ADDRESS, 79
 SCB_BASE_ADDRESS, 79
NVIC Registers, 79
 MNVIC, 80
 MSCB, 80
NVIC Settable Priorities, 80
 BUSFAULT, 81
 MEMORYMANAGE, 81
 PENDSV, 81
 SVCALL, 81
 SYSTICK, 81
 USAGEFAULT, 82
NVIC Vector Table, 62
 ADC, 69
 DMA1_STREAM0, 67
 DMA1_STREAM1, 67
 DMA1_STREAM2, 68
 DMA1_STREAM3, 68
 DMA1_STREAM4, 68
 DMA1_STREAM5, 68
 DMA1_STREAM6, 69
 DMA1_STREAM7, 74
 DMA2_STREAM0, 75
 DMA2_STREAM1, 75
 DMA2_STREAM2, 75
 DMA2_STREAM3, 76
 DMA2_STREAM4, 76
 DMA2_STREAM5, 76
 DMA2_STREAM6, 77
 DMA2_STREAM7, 77
 EXTI0, 66
 EXTI1, 66
 EXTI15_10, 73
 EXTI16, 65
 EXTI17, 73
 EXTI18, 74
 EXTI2, 66
 EXTI21, 65
 EXTI22, 65
 EXTI3, 67
 EXTI4, 67

EXTI9, 69
 FLASH, 65
 FPU, 78
 I2C1_ER, 71
 I2C1_EV, 71
 I2C2_ER, 72
 I2C2_EV, 72
 I2C3_ER, 78
 I2C3_EV, 77
 OTG_FS, 76
 RCC, 66
 SDIO, 74
 SPI1, 72
 SPI2, 72
 SPI3, 75
 SPI4, 78
 TIM1_BRK_TIM9, 69
 TIM1_CC, 70
 TIM1_TRG_COM_TIM11, 70
 TIM1_UP_TIM10, 70
 TIM2, 70
 TIM3, 71
 TIM4, 71
 TIM5, 74
 USART1, 73
 USART2, 73
 USART6, 77
 WWDG, 64
NVIC_BASE_ADDRESS
 NVIC Addresses, 79
NVIC_GetPriority
 NVIC_interface.h, 213
 NVIC_program.c, 221
NVIC_interface.h
 MNVIC_u8GetActive, 212
 MNVIC_vClearPendingFlag, 211
 MNVIC_vDisablePeriphral, 211
 MNVIC_vEnablePeriphral, 210
 MNVIC_vSetPendingFlag, 211
 MNVIC_vSetPriority, 213
 MNVIC_vSetPriorityConfig, 212
 NVIC_GetPriority, 213
NVIC_MemoryMapType, 113
 IABRx, 116
 ICERx, 114
 ICPRx, 115
 IPRx, 116
 ISERx, 113
 ISPRx, 114
 RESERVED0x, 113
 RESERVED2x, 115
 RESERVED3x, 115
 RESERVED4x, 116
 RSERVED1x, 114
NVIC_private.h
 MNVIC_STIR, 216
 VECTKEY_PASSWORD, 216
NVIC_program.c

MNVIC_u8GetActive, 219
 MNVIC_vClearPendingFlag, 219
 MNVIC_vDisablePeriphral, 219
 MNVIC_vEnablePeriphral, 218
 MNVIC_vSetPendingFlag, 219
 MNVIC_vSetPriority, 220
 MNVIC_vSetPriorityConfig, 220
 NVIC_GetPriority, 221
 REGISTER_SIZE, 218

ODRx
 GPIOx_MemoryMapType, 107

OSPEEDRx
 GPIOx_MemoryMapType, 107

OTG_FS
 NVIC Vector Table, 76

OTGFS
 MRCC_config.h, 230

OTYPERx
 GPIOx_MemoryMapType, 107

OutputSpeed
 MGPIox_ConfigType, 110

OutputType
 MGPIox_ConfigType, 110

P2CONST
 Compiler standard macros, 13

P2FUNC
 Compiler standard macros, 14

P2VAR
 Compiler standard macros, 13

PEND_SV
 NVIC Settable Priorities, 81

PERIODIC_INTERVAL_MODE
 SysTick Interval Mode Configuration, 94

PERIODIC_INTERVAL_TICK_TIME
 SysTick_interface.h, 313

Pin
 MGPIox_ConfigType, 109

PLL
 MRCC_config.h, 225

PLLCFGR
 RCC_MemoryMapType, 118

PLLCLK
 MRCC_private.h, 273, 275

PLLI2S
 MRCC_config.h, 225

PLLI2SCFGR
 RCC_MemoryMapType, 125

PLLI2SCLK
 MRCC_private.h, 273

PLLM
 MRCC_config.h, 227

PLLN
 MRCC_config.h, 227

PLLP
 MRCC_config.h, 226

PLLQ
 MRCC_config.h, 226

PLLSRC
 MRCC_config.h, 226

PMC
 MSYSCFG_MemMap_t, 112

Port
 MGPIOx_ConfigType, 109

PortNum
 EXTI_ConfigType, 103

PPRE1
 MRCC_config.h, 228

PPRE2
 MRCC_config.h, 228

PR
 EXTI_Type, 105

PRESSED
 Standard values, 20

PUPDRx
 GPIOx_MemoryMapType, 107

PWREN
 MRCC_config.h, 230

RCC
 MRCC_private.h, 256
 NVIC Vector Table, 66

RCC_AHB1
 ID options, 82

RCC_AHB1ENR_CRCEN
 MRCC_private.h, 267

RCC_AHB1ENR_DMA1EN
 MRCC_private.h, 267

RCC_AHB1ENR_DMA2EN
 MRCC_private.h, 267

RCC_AHB1ENR_GPIOAEN
 MRCC_private.h, 268

RCC_AHB1ENR_GPIOBEN
 MRCC_private.h, 268

RCC_AHB1ENR_GPIOCEN
 MRCC_private.h, 268

RCC_AHB1ENR_GPIODEN
 MRCC_private.h, 267

RCC_AHB1ENR_GPIOEEN
 MRCC_private.h, 267

RCC_AHB1ENR_GPIOHEN
 MRCC_private.h, 267

RCC_AHB1LPENR
 ID options, 83

RCC_AHB1LPENR_FLITFLPEN
 MRCC_private.h, 272

RCC_AHB2
 ID options, 82

RCC_AHB2ENR_OTGFSEN
 MRCC_private.h, 268

RCC_APB1
 ID options, 83

RCC_APB1ENR_I2C1EN
 MRCC_private.h, 269

RCC_APB1ENR_I2C2EN
 MRCC_private.h, 269

RCC_APB1ENR_I2C3EN
 MRCC_private.h, 268

RCC_APB1ENR_PWREN
 MRCC_private.h, 268

RCC_APB1ENR_SPI2EN
 MRCC_private.h, 269

RCC_APB1ENR_SPI3EN
 MRCC_private.h, 269

RCC_APB1ENR_TIM2EN
 MRCC_private.h, 270

RCC_APB1ENR_TIM3EN
 MRCC_private.h, 270

RCC_APB1ENR_TIM4EN
 MRCC_private.h, 270

RCC_APB1ENR_TIM5EN
 MRCC_private.h, 270

RCC_APB1ENR_USART2EN
 MRCC_private.h, 269

RCC_APB1ENR_WWDGEN
 MRCC_private.h, 269

RCC_APB2
 ID options, 83

RCC_APB2ENR_ADC1EN
 MRCC_private.h, 271

RCC_APB2ENR_SDIOEN
 MRCC_private.h, 271

RCC_APB2ENR_SPI1EN
 MRCC_private.h, 271

RCC_APB2ENR_SPI4EN
 MRCC_private.h, 271

RCC_APB2ENR_SYSCFGEN
 MRCC_private.h, 271

RCC_APB2ENR_TIM10EN
 MRCC_private.h, 270

RCC_APB2ENR_TIM11EN
 MRCC_private.h, 270

RCC_APB2ENR_TIM1EN
 MRCC_private.h, 272

RCC_APB2ENR_TIM9EN
 MRCC_private.h, 271

RCC_APB2ENR_USART1EN
 MRCC_private.h, 272

RCC_APB2ENR_USART6EN
 MRCC_private.h, 272

RCC_BASE_ADDRESS
 MRCC_private.h, 256

RCC_CFGR_HPREG_b0
 MRCC_private.h, 265

RCC_CFGR_HPREG_b1
 MRCC_private.h, 265

RCC_CFGR_HPREG_b2
 MRCC_private.h, 266

RCC_CFGR_HPREG_b3
 MRCC_private.h, 266

RCC_CFGR_I2SSRC
 MRCC_private.h, 263

RCC_CFGR_MOC1_b0
 MRCC_private.h, 263

RCC_CFGR_MOC1_b1
 MRCC_private.h, 263

MRCC_private.h, 263
 RCC_CFGR_MOC1PRE_b0
 MRCC_private.h, 262
 RCC_CFGR_MOC1PRE_b1
 MRCC_private.h, 263
 RCC_CFGR_MOC1PRE_b2
 MRCC_private.h, 263
 RCC_CFGR_MOC2_b0
 MRCC_private.h, 262
 RCC_CFGR_MOC2_b1
 MRCC_private.h, 262
 RCC_CFGR_MOC2PRE_b0
 MRCC_private.h, 262
 RCC_CFGR_MOC2PRE_b1
 MRCC_private.h, 262
 RCC_CFGR_MOC2PRE_b2
 MRCC_private.h, 262
 RCC_CFGR_PPREG_b0
 MRCC_private.h, 265
 RCC_CFGR_PPREG_b1
 MRCC_private.h, 265
 RCC_CFGR_PPREG_b2
 MRCC_private.h, 265
 RCC_CFGR_PPREG_b3
 MRCC_private.h, 265
 RCC_CFGR_PPREG_b4
 MRCC_private.h, 265
 RCC_CFGR_PPREG_b5
 MRCC_private.h, 265
 RCC_CFGR_RTCPRE_b0
 MRCC_private.h, 263
 RCC_CFGR_RTCPRE_b1
 MRCC_private.h, 264
 RCC_CFGR_RTCPRE_b2
 MRCC_private.h, 264
 RCC_CFGR_RTCPRE_b3
 MRCC_private.h, 264
 RCC_CFGR_RTCPRE_b4
 MRCC_private.h, 264
 RCC_CFGR_SW_b0
 MRCC_private.h, 266
 RCC_CFGR_SW_b1
 MRCC_private.h, 266
 RCC_CFGR_SWS_b0
 MRCC_private.h, 266
 RCC_CFGR_SWS_b1
 MRCC_private.h, 266
 RCC_CR_CSSON
 MRCC_private.h, 257
 RCC_CR_HSEBYP
 MRCC_private.h, 257
 RCC_CR_HSEON
 MRCC_private.h, 257
 RCC_CR_HSIRDY
 MRCC_private.h, 257
 RCC_CR_HSION
 MRCC_private.h, 258
 RCC_CR_HSIRDY
 MRCC_private.h, 258
 RCC_CR_PLLCFGR_PLLM_b0
 MRCC_private.h, 261
 RCC_PLLCFGR_PLLM_b1
 MRCC_private.h, 261
 RCC_PLLCFGR_PLLM_b2
 MRCC_private.h, 261
 RCC_PLLCFGR_PLLM_b3
 MRCC_private.h, 261
 RCC_PLLCFGR_PLLM_b4
 MRCC_private.h, 261
 RCC_PLLCFGR_PLLM_b5
 MRCC_private.h, 261
 RCC_PLLCFGR_PLNN_b0

MRCC_private.h, 259
RCC_PLLCFGR_PLLN_b1
 MRCC_private.h, 259
RCC_PLLCFGR_PLLN_b2
 MRCC_private.h, 259
RCC_PLLCFGR_PLLN_b3
 MRCC_private.h, 260
RCC_PLLCFGR_PLLN_b4
 MRCC_private.h, 260
RCC_PLLCFGR_PLLN_b5
 MRCC_private.h, 260
RCC_PLLCFGR_PLLN_b7
 MRCC_private.h, 260
RCC_PLLCFGR_PLLN_b8
 MRCC_private.h, 260
RCC_PLLCFGR_PLLP_b0
 MRCC_private.h, 259
RCC_PLLCFGR_PLLP_b1
 MRCC_private.h, 259
RCC_PLLCFGR_PLLQ_b0
 MRCC_private.h, 258
RCC_PLLCFGR_PLLQ_b1
 MRCC_private.h, 258
RCC_PLLCFGR_PLLQ_b2
 MRCC_private.h, 258
RCC_PLLCFGR_PLLQ_b3
 MRCC_private.h, 258
RCC_PLLCFGR_PLLSRC
 MRCC_private.h, 259
REGISTER_SIZE
 NVIC_program.c, 218
RELEASED
 Standard values, 20
RESERVED
 SCB_MemoryMapType, 130
RESERVED0x
 NVIC_MemoryMapType, 113
Reserved1
 RCC_MemoryMapType, 119
Reserved10
 RCC_MemoryMapType, 123
Reserved11
 RCC_MemoryMapType, 123
Reserved12
 RCC_MemoryMapType, 124
Reserved13
 RCC_MemoryMapType, 124
Reserved14
 RCC_MemoryMapType, 124
Reserved2
 RCC_MemoryMapType, 120
RESERVED2x
 NVIC_MemoryMapType, 115
Reserved3
 RCC_MemoryMapType, 120
RESERVED3x
 NVIC_MemoryMapType, 115
NVIC_MemoryMapType, 115
Reserved4
 RCC_MemoryMapType, 120
RESERVED4x
 NVIC_MemoryMapType, 116
Reserved5
 RCC_MemoryMapType, 121
Reserved6
 RCC_MemoryMapType, 121
Reserved7
 RCC_MemoryMapType, 122
Reserved8
 RCC_MemoryMapType, 122
Reserved9
 RCC_MemoryMapType, 123
RSERVED1x
 NVIC_MemoryMapType, 114
RTCPRE
 MRCC_config.h, 228
RTSR
 EXTI_Type, 105
RUN
 Standard values, 20
s16_t
 Standard types, 17
s32_t
 Standard types, 17
s8_t
 Standard types, 16
SCB_BASE_ADDRESS
 NVIC Addresses, 79
SCB_MemoryMapType, 125
 AIRCR, 127
 BFAR, 131
 CCR, 128
 CFSR, 129
 CPUID, 126
 HFSR, 130
 ICSR, 127
 MMFAR, 130
 RESERVED, 130
 SCR, 127
 SHCSR, 129
 SHPR1, 128
 SHPR2, 128
 SHPR3, 129
 VTOR, 127
SCR
 SCB_MemoryMapType, 127
SDIO
 NVIC Vector Table, 74
SDIOEN
 MRCC_config.h, 233
SEC
 Delay Units, 95
SET_BIT
 Bit Manipulation Math Macros, 9
SET_BITS

Bit Group Manipulation Math Macros, 11
SHCSR
 SCB_MemoryMapType, 129
SHPR1
 SCB_MemoryMapType, 128
SHPR2
 SCB_MemoryMapType, 128
SHPR3
 SCB_MemoryMapType, 129
Side1Pin
 DCM_MotorConfiguration, 102
Side1Port
 DCM_MotorConfiguration, 101
Side2Pin
 DCM_MotorConfiguration, 102
Side2Port
 DCM_MotorConfiguration, 102
SINGLE_INTERVAL_MODE
 Systick Interval Mode Configuration, 94
SINGLE_INTERVAL_TICK_TIME
 SysTick_interface.h, 313
SPI1
 NVIC Vector Table, 72
SPI1EN
 MRCC_config.h, 233
SPI2
 NVIC Vector Table, 72
SPI2EN
 MRCC_config.h, 231
SPI3
 NVIC Vector Table, 75
SPI3EN
 MRCC_config.h, 231
SPI4
 NVIC Vector Table, 78
SPI4EN
 MRCC_config.h, 233
SSCGR
 RCC_MemoryMapType, 125
Standard types, 15
 bool_t, 15
 fl32_t, 17
 fl64_t, 17
 s16_t, 17
 s32_t, 17
 s8_t, 16
 u16_t, 16
 u32_t, 16
 u8_t, 16
Standard values, 18
 FALSE, 18
 FLAG_CLEARED, 19
 FLAG_SET, 19
 INITIAL_ZERO, 19
 NULL, 19
 PRESSED, 20
 RELEASED, 20
 RUN, 20
STOP, 20
TRUE, 18
STATIC
 Compiler standard macros, 14
STOP
 Standard values, 20
Sub-Group priorities, 57
 NO_SUB_PRIORITY, 58
 SUB_PRIORITY_0, 58
 SUB_PRIORITY_1, 58
 SUB_PRIORITY_10, 60
 SUB_PRIORITY_11, 61
 SUB_PRIORITY_12, 61
 SUB_PRIORITY_13, 61
 SUB_PRIORITY_14, 61
 SUB_PRIORITY_15, 62
 SUB_PRIORITY_2, 59
 SUB_PRIORITY_3, 58
 SUB_PRIORITY_4, 59
 SUB_PRIORITY_5, 59
 SUB_PRIORITY_6, 59
 SUB_PRIORITY_7, 60
 SUB_PRIORITY_8, 60
 SUB_PRIORITY_9, 60
SUB_PRIORITY_0
 Sub-Group priorities, 58
SUB_PRIORITY_1
 Sub-Group priorities, 58
SUB_PRIORITY_10
 Sub-Group priorities, 60
SUB_PRIORITY_11
 Sub-Group priorities, 61
SUB_PRIORITY_12
 Sub-Group priorities, 61
SUB_PRIORITY_13
 Sub-Group priorities, 61
SUB_PRIORITY_14
 Sub-Group priorities, 61
SUB_PRIORITY_15
 Sub-Group priorities, 62
SUB_PRIORITY_2
 Sub-Group priorities, 59
SUB_PRIORITY_3
 Sub-Group priorities, 58
SUB_PRIORITY_4
 Sub-Group priorities, 59
SUB_PRIORITY_5
 Sub-Group priorities, 59
SUB_PRIORITY_6
 Sub-Group priorities, 59
SUB_PRIORITY_7
 Sub-Group priorities, 60
SUB_PRIORITY_8
 Sub-Group priorities, 60
SUB_PRIORITY_9
 Sub-Group priorities, 60
SV_CALL
 NVIC Settable Priorities, 81

SW
 MRCC_config.h, 229

SWIER
 EXTI_Type, 105

SWS
 MRCC_config.h, 228

SYSCFG Memory Addresses, 29
 SYSCFG_BASE_ADDR, 29

SYSCFG Memory Registers, 29
 MSYSCFG, 29

SYSCFG_BASE_ADDR
 SYSCFG Memory Addresses, 29

SYSCFGEN
 MRCC_config.h, 233

SYSCLK
 MRCC_private.h, 273

SYSCLKby128
 MRCC_private.h, 276

SYSCLKby16
 MRCC_private.h, 276

SYSCLKby2
 MRCC_private.h, 275

SYSCLKby256
 MRCC_private.h, 276

SYSCLKby4
 MRCC_private.h, 275

SYSCLKby512
 MRCC_private.h, 276

SYSCLKby64
 MRCC_private.h, 276

SYSCLKby8
 MRCC_private.h, 275

SYSTICK
 NVIC Settable Priorities, 81

Systick
 Systick Registers, 96

Systick Addresses, 95
 SysTick_BASE_ADDRESS, 95

Systick Clock Sources, 98
 AHB, 98
 AHB_DividedBy8, 98

Systick Exception (Interrupt) Status, 98
 AssertRequest, 99
 Dont AssertRequest, 98

Systick Interval Mode Configuration, 94
 PERIODIC_INTERVAL_MODE, 94
 SINGLE_INTERVAL_MODE, 94

Systick Register Bits Positions, 96
 CLKSOURCE, 97
 COUNTER_ENABLE, 97
 COUNTFLAG, 96
 TICKINT, 97

Systick Registers, 96
 SysTick, 96

SysTick_BASE_ADDRESS
 Systick Addresses, 95

SysTick_config.h
 Exception_Request, 310

 MAX_TICKS, 311

SysTick_Handler
 SysTick_program.c, 328

SysTick_interface.h
 BUSY_TICK_TIME, 313
 MSysTick_u32GetElapsedTime, 317
 MSysTick_u32GetRemainingTime, 317
 MSysTick_vDelay, 314
 MSysTick_vDelayMicroSec, 315
 MSysTick_vDelayMilliSec, 315
 MSysTick_vDelaySec, 315
 MSysTick_vDisable, 318
 MSysTick_vEnable, 317
 MSysTick_vEnableException, 318
 MSysTick_vInit, 314
 MSysTick_vSetBusyWait, 314
 MSysTick_vSetPeriodicInterval, 316
 MSysTick_vSetSingleInterval, 315
 MSysTick_vStopInterval, 316
 PERIODIC_INTERVAL_TICK_TIME, 313
 SINGLE_INTERVAL_TICK_TIME, 313

SysTick_program.c
 MSysTick_u32GetElapsedTime, 327
 MSysTick_u32GetRemainingTime, 327
 MSysTick_vDelay, 323
 MSysTick_vDelayMicroSec, 324
 MSysTick_vDelayMilliSec, 324
 MSysTick_vDelaySec, 325
 MSysTick_vDisable, 328
 MSysTick_vEnable, 328
 MSysTick_vEnableException, 322
 MSysTick_vInit, 322
 MSysTick_vSetBusyWait, 322
 MSysTick_vSetPeriodicInterval, 326
 MSysTick_vSetSingleInterval, 325
 MSysTick_vStopInterval, 326
 SysTick_Handler, 328

SysTick_Type, 131
 CALIB, 132
 CTRL, 132
 LOAD, 132
 VAL, 132

TICKINT
 Systick Register Bits Positions, 97

TIM10EN
 MRCC_config.h, 233

TIM11EN
 MRCC_config.h, 232

TIM1_BRK_TIM9
 NVIC Vector Table, 69

TIM1_CC
 NVIC Vector Table, 70

TIM1_TRG_COM_TIM11
 NVIC Vector Table, 70

TIM1_UP_TIM10
 NVIC Vector Table, 70

TIM1EN
 MRCC_config.h, 234

TIM2
 NVIC Vector Table, 70

TIM2EN
 MRCC_config.h, 232

TIM3
 NVIC Vector Table, 71

TIM3EN
 MRCC_config.h, 232

TIM4
 NVIC Vector Table, 71

TIM4EN
 MRCC_config.h, 232

TIM5
 NVIC Vector Table, 74

TIM5EN
 MRCC_config.h, 232

TIM9EN
 MRCC_config.h, 233

TOGGLE_BIT
 Bit Manipulation Math Macros, 10

TOGGLE_BITs
 Bit Group Manipulation Math Macros, 11

TriggerStatus
 EXTI_ConfigType, 103

TRUE
 Standard values, 18

u16_t
 Standard types, 16

u32_t
 Standard types, 16

u8_t
 Standard types, 16

USAGE_FAULT
 NVIC Settable Priorities, 82

USART1
 NVIC Vector Table, 73

USART1EN
 MRCC_config.h, 234

USART2
 NVIC Vector Table, 73

USART2EN
 MRCC_config.h, 231

USART6
 NVIC Vector Table, 77

USART6EN
 MRCC_config.h, 234

Utilities macros, 15
 MY_MS_DELAY, 15

VAL
 SysTick_Type, 132

VAR
 Compiler standard macros, 12

VECTKEY_PASSWORD
 NVIC_private.h, 216

VTOR
 SCB_MemoryMapType, 127