

# BASKETBALL LAUNCHER

Generated by Doxygen 1.10.0



<b>1 MAIN PAGE</b>	<b>1</b>
1.1 MOTIVATION . . . . .	1
1.2 OBJECTIVE . . . . .	1
1.3 IDEATION DESIGN . . . . .	1
1.4 CAD MODELING . . . . .	2
1.5 LAUNCHER . . . . .	2
1.6 CONTROLLER . . . . .	2
1.7 FUTURE IMPROVEMENT . . . . .	2
1.8 YOUTUBE REFERENCE . . . . .	2
1.9 REPOSITORY REFERENCE . . . . .	3
1.10 CONTACT INFO . . . . .	3
<b>2 CONTROLLER</b>	<b>5</b>
2.1 FINAL DESIGN . . . . .	5
2.2 CONTROLLER PCBA . . . . .	5
2.3 FINITE STATE MACHINE . . . . .	5
2.4 BILL OF MATERIAL - 3D PRINTED PARTS . . . . .	6
2.5 BILL OF MATERIAL - ELECTRONICS . . . . .	6
<b>3 LAUNCHER</b>	<b>7</b>
3.1 FINAL DESIGN . . . . .	7
3.2 LAUNCHER PCBA . . . . .	7
3.3 FINITE STATE MACHINE . . . . .	8
3.4 BILL OF MATERIAL - CYCLOIDAL GEARBOX . . . . .	9
3.5 BILL OF MATERIAL - MECHANICAL . . . . .	9
3.6 BILL OF MATERIAL - 3D PRINTED PARTS . . . . .	10
3.7 BILL OF MATERIAL - ELECTRONICS . . . . .	10
<b>4 REPORTS</b>	<b>11</b>
4.1 PROPOSAL . . . . .	11
4.2 BILL OF MATERIAL . . . . .	11
4.3 REFERENCE MANUAL . . . . .	11
4.4 OVERALL REPORT . . . . .	11
<b>5 Topic Index</b>	<b>13</b>
5.1 Topics . . . . .	13
<b>6 Data Structure Index</b>	<b>15</b>
6.1 Data Structures . . . . .	15
<b>7 File Index</b>	<b>17</b>
7.1 File List . . . . .	17
<b>8 Topic Documentation</b>	<b>19</b>
8.1 CMSIS . . . . .	19

8.1.1 Detailed Description . . . . .	19
8.1.2 Stm32f4xx_system . . . . .	19
8.1.2.1 Detailed Description . . . . .	19
8.1.2.2 STM32F4xx_System_Private_Includes . . . . .	19
8.1.2.3 STM32F4xx_System_Private_TypesDefinitions . . . . .	20
8.1.2.4 STM32F4xx_System_Private_Defines . . . . .	20
8.1.2.5 STM32F4xx_System_Private_Macros . . . . .	20
8.1.2.6 STM32F4xx_System_Private_Variables . . . . .	20
8.1.2.7 STM32F4xx_System_Private_FunctionPrototypes . . . . .	21
8.1.2.8 STM32F4xx_System_Private_Functions . . . . .	21
<b>9 Data Structure Documentation</b>	<b>25</b>
9.1 D4215X Struct Reference . . . . .	25
9.1.1 Detailed Description . . . . .	25
9.1.2 Field Documentation . . . . .	25
9.1.2.1 channel . . . . .	25
9.1.2.2 speed . . . . .	25
9.1.2.3 timer . . . . .	26
9.2 LEDX Struct Reference . . . . .	26
9.2.1 Detailed Description . . . . .	26
9.2.2 Field Documentation . . . . .	26
9.2.2.1 GPIOx . . . . .	26
9.2.2.2 PIN . . . . .	26
9.3 MPU6050 Struct Reference . . . . .	27
9.3.1 Detailed Description . . . . .	27
9.3.2 Field Documentation . . . . .	27
9.3.2.1 addr . . . . .	27
9.3.2.2 dt . . . . .	27
9.3.2.3 gX . . . . .	27
9.3.2.4 gX_offset . . . . .	27
9.3.2.5 gY . . . . .	28
9.3.2.6 gY_offset . . . . .	28
9.3.2.7 gZ . . . . .	28
9.3.2.8 gZ_offset . . . . .	28
9.3.2.9 hi2c . . . . .	28
9.3.2.10 status . . . . .	28
9.4 RadioX Struct Reference . . . . .	28
9.4.1 Detailed Description . . . . .	29
9.4.2 Field Documentation . . . . .	29
9.4.2.1 channel1 . . . . .	29
9.4.2.2 channel2 . . . . .	29
9.4.2.3 counter1 . . . . .	29

9.4.2.4 counter2 . . . . .	29
9.4.2.5 pulseWidth . . . . .	29
9.4.2.6 sum1 . . . . .	30
9.4.2.7 sum2 . . . . .	30
9.4.2.8 timer . . . . .	30
9.5 StepperX Struct Reference . . . . .	30
9.5.1 Detailed Description . . . . .	30
9.5.2 Field Documentation . . . . .	30
9.5.2.1 DIR_PIN . . . . .	30
9.5.2.2 EN_PIN . . . . .	31
9.5.2.3 GPIOx . . . . .	31
9.5.2.4 STP_PIN . . . . .	31
9.6 SwitchX Struct Reference . . . . .	31
9.6.1 Detailed Description . . . . .	31
9.6.2 Field Documentation . . . . .	31
9.6.2.1 GPIOx . . . . .	31
9.6.2.2 PIN . . . . .	32
9.6.2.3 status . . . . .	32
<b>10 File Documentation</b>	<b>33</b>
10.1 Ball_Launcher_Controller/Core/Inc/main_C.h File Reference . . . . .	33
10.1.1 Detailed Description . . . . .	33
10.1.2 Function Documentation . . . . .	33
10.1.2.1 Error_Handler() . . . . .	33
10.2 Ball_Launcher_Controller/Core/Inc/mpu6050.h File Reference . . . . .	34
10.2.1 Detailed Description . . . . .	34
10.2.2 Function Documentation . . . . .	34
10.2.2.1 mpu6050_calibrate() . . . . .	34
10.2.2.2 mpu6050_get_gX() . . . . .	35
10.2.2.3 mpu6050_get_gY() . . . . .	35
10.2.2.4 mpu6050_get_gZ() . . . . .	35
10.2.2.5 mpu6050_get_X() . . . . .	36
10.2.2.6 mpu6050_get_Y() . . . . .	36
10.2.2.7 mpu6050_get_Z() . . . . .	36
10.2.2.8 mpu6050_init() . . . . .	37
10.2.2.9 mpu6050_update() . . . . .	37
10.3 Ball_Launcher_Controller/Core/Inc/stm32f4xx_hal_conf_C.h File Reference . . . . .	37
10.3.1 Detailed Description . . . . .	40
10.3.2 Macro Definition Documentation . . . . .	40
10.3.2.1 assert_param . . . . .	40
10.3.2.2 DATA_CACHE_ENABLE . . . . .	40
10.3.2.3 DP83848_PHY_ADDRESS . . . . .	40

10.3.2.4 ETH_RX_BUF_SIZE . . . . .	40
10.3.2.5 ETH_RXBUFNB . . . . .	41
10.3.2.6 ETH_TX_BUF_SIZE . . . . .	41
10.3.2.7 ETH_TXBUFNB . . . . .	41
10.3.2.8 EXTERNAL_CLOCK_VALUE . . . . .	41
10.3.2.9 HAL_CORTEX_MODULE_ENABLED . . . . .	41
10.3.2.10 HAL_DMA_MODULE_ENABLED . . . . .	41
10.3.2.11 HAL_EXTI_MODULE_ENABLED . . . . .	41
10.3.2.12 HAL_FLASH_MODULE_ENABLED . . . . .	41
10.3.2.13 HAL_GPIO_MODULE_ENABLED . . . . .	41
10.3.2.14 HAL_I2C_MODULE_ENABLED . . . . .	42
10.3.2.15 HAL_MODULE_ENABLED . . . . .	42
10.3.2.16 HAL_PWR_MODULE_ENABLED . . . . .	42
10.3.2.17 HAL_RCC_MODULE_ENABLED . . . . .	42
10.3.2.18 HAL_UART_MODULE_ENABLED . . . . .	42
10.3.2.19 HSE_STARTUP_TIMEOUT . . . . .	42
10.3.2.20 HSE_VALUE . . . . .	42
10.3.2.21 HSI_VALUE . . . . .	42
10.3.2.22 INSTRUCTION_CACHE_ENABLE . . . . .	43
10.3.2.23 LSE_STARTUP_TIMEOUT . . . . .	43
10.3.2.24 LSE_VALUE . . . . .	43
10.3.2.25 LSI_VALUE . . . . .	43
10.3.2.26 MAC_ADDR0 . . . . .	43
10.3.2.27 MAC_ADDR1 . . . . .	43
10.3.2.28 MAC_ADDR2 . . . . .	43
10.3.2.29 MAC_ADDR3 . . . . .	43
10.3.2.30 MAC_ADDR4 . . . . .	44
10.3.2.31 MAC_ADDR5 . . . . .	44
10.3.2.32 PHY_AUTONEGO_COMPLETE . . . . .	44
10.3.2.33 PHY_AUTONEGOTIATION . . . . .	44
10.3.2.34 PHY_BCR . . . . .	44
10.3.2.35 PHY_BSR . . . . .	44
10.3.2.36 PHY_CONFIG_DELAY . . . . .	44
10.3.2.37 PHY_DUPLEX_STATUS . . . . .	44
10.3.2.38 PHY_FULLDUPLEX_100M . . . . .	45
10.3.2.39 PHY_FULLDUPLEX_10M . . . . .	45
10.3.2.40 PHY_HALFDUPLEX_100M . . . . .	45
10.3.2.41 PHY_HALFDUPLEX_10M . . . . .	45
10.3.2.42 PHY_ISOLATE . . . . .	45
10.3.2.43 PHY_JABBER_DETECTION . . . . .	45
10.3.2.44 PHY_LINKED_STATUS . . . . .	45
10.3.2.45 PHY_LOOPBACK . . . . .	46

10.3.2.46 PHY_POWERDOWN . . . . .	46
10.3.2.47 PHY_READ_TO . . . . .	46
10.3.2.48 PHY_RESET . . . . .	46
10.3.2.49 PHY_RESET_DELAY . . . . .	46
10.3.2.50 PHY_RESTART_AUTONEGOTIATION . . . . .	46
10.3.2.51 PHY_SPEED_STATUS . . . . .	46
10.3.2.52 PHY_SR . . . . .	46
10.3.2.53 PHY_WRITE_TO . . . . .	47
10.3.2.54 PREFETCH_ENABLE . . . . .	47
10.3.2.55 TICK_INT_PRIORITY . . . . .	47
10.3.2.56 USE_HAL_ADC_REGISTER_CALLBACKS . . . . .	47
10.3.2.57 USE_HAL_CAN_REGISTER_CALLBACKS . . . . .	47
10.3.2.58 USE_HAL_CEC_REGISTER_CALLBACKS . . . . .	47
10.3.2.59 USE_HAL_Cryp_REGISTER_CALLBACKS . . . . .	47
10.3.2.60 USE_HAL_DAC_REGISTER_CALLBACKS . . . . .	47
10.3.2.61 USE_HAL_DCI_REGISTER_CALLBACKS . . . . .	47
10.3.2.62 USE_HAL_DFSDM_REGISTER_CALLBACKS . . . . .	47
10.3.2.63 USE_HAL_DMA2D_REGISTER_CALLBACKS . . . . .	48
10.3.2.64 USE_HAL_DSI_REGISTER_CALLBACKS . . . . .	48
10.3.2.65 USE_HAL_ETH_REGISTER_CALLBACKS . . . . .	48
10.3.2.66 USE_HAL_FMPI2C_REGISTER_CALLBACKS . . . . .	48
10.3.2.67 USE_HAL_FMPMBUS_REGISTER_CALLBACKS . . . . .	48
10.3.2.68 USE_HAL_HASH_REGISTER_CALLBACKS . . . . .	48
10.3.2.69 USE_HAL_HCD_REGISTER_CALLBACKS . . . . .	48
10.3.2.70 USE_HAL_I2C_REGISTER_CALLBACKS . . . . .	48
10.3.2.71 USE_HAL_I2S_REGISTER_CALLBACKS . . . . .	48
10.3.2.72 USE_HAL_IRDA_REGISTER_CALLBACKS . . . . .	48
10.3.2.73 USE_HAL_LPTIM_REGISTER_CALLBACKS . . . . .	49
10.3.2.74 USE_HAL_LTDC_REGISTER_CALLBACKS . . . . .	49
10.3.2.75 USE_HAL_MMC_REGISTER_CALLBACKS . . . . .	49
10.3.2.76 USE_HAL_NAND_REGISTER_CALLBACKS . . . . .	49
10.3.2.77 USE_HAL_NOR_REGISTER_CALLBACKS . . . . .	49
10.3.2.78 USE_HAL_PCCARD_REGISTER_CALLBACKS . . . . .	49
10.3.2.79 USE_HAL_PCD_REGISTER_CALLBACKS . . . . .	49
10.3.2.80 USE_HAL_QSPI_REGISTER_CALLBACKS . . . . .	49
10.3.2.81 USE_HAL_RNG_REGISTER_CALLBACKS . . . . .	49
10.3.2.82 USE_HAL_RTC_REGISTER_CALLBACKS . . . . .	49
10.3.2.83 USE_HAL_SAI_REGISTER_CALLBACKS . . . . .	50
10.3.2.84 USE_HAL_SD_REGISTER_CALLBACKS . . . . .	50
10.3.2.85 USE_HAL_SDRAM_REGISTER_CALLBACKS . . . . .	50
10.3.2.86 USE_HAL_SMARTCARD_REGISTER_CALLBACKS . . . . .	50
10.3.2.87 USE_HAL_SMBUS_REGISTER_CALLBACKS . . . . .	50

10.3.2.88 USE_HAL_SPDIFRX_REGISTER_CALLBACKS . . . . .	50
10.3.2.89 USE_HAL_SPI_REGISTER_CALLBACKS . . . . .	50
10.3.2.90 USE_HAL_SRAM_REGISTER_CALLBACKS . . . . .	50
10.3.2.91 USE_HAL_TIM_REGISTER_CALLBACKS . . . . .	50
10.3.2.92 USE_HAL_UART_REGISTER_CALLBACKS . . . . .	50
10.3.2.93 USE_HAL_USART_REGISTER_CALLBACKS . . . . .	51
10.3.2.94 USE_HAL_WWDG_REGISTER_CALLBACKS . . . . .	51
10.3.2.95 USE_RTOS . . . . .	51
10.3.2.96 USE_SPI_CRC . . . . .	51
10.3.2.97 VDD_VALUE . . . . .	51
10.4 Ball_Launcher_Controller/Core/Inc/stm32f4xx_it_C.h File Reference . . . . .	51
10.4.1 Detailed Description . . . . .	52
10.4.2 Function Documentation . . . . .	52
10.4.2.1 BusFault_Handler() . . . . .	52
10.4.2.2 DebugMon_Handler() . . . . .	52
10.4.2.3 HardFault_Handler() . . . . .	52
10.4.2.4 MemManage_Handler() . . . . .	52
10.4.2.5 NMI_Handler() . . . . .	52
10.4.2.6 PendSV_Handler() . . . . .	53
10.4.2.7 SVC_Handler() . . . . .	53
10.4.2.8 SysTick_Handler() . . . . .	53
10.4.2.9 UsageFault_Handler() . . . . .	53
10.5 Ball_Launcher_Controller/Core/Src/main_C.c File Reference . . . . .	53
10.5.1 Detailed Description . . . . .	54
10.5.2 Function Documentation . . . . .	54
10.5.2.1 Error_Handler() . . . . .	54
10.5.2.2 main() . . . . .	54
10.5.2.3 SystemClock_Config() . . . . .	55
10.5.3 Variable Documentation . . . . .	55
10.5.3.1 buffer . . . . .	55
10.5.3.2 gX . . . . .	55
10.5.3.3 gY . . . . .	55
10.5.3.4 gZ . . . . .	55
10.5.3.5 hi2c1 . . . . .	55
10.5.3.6 huart1 . . . . .	56
10.5.3.7 huart2 . . . . .	56
10.5.3.8 move . . . . .	56
10.5.3.9 shot . . . . .	56
10.5.3.10 stat . . . . .	56
10.5.3.11 STATE . . . . .	56
10.5.3.12 STATE_0_INIT . . . . .	56
10.5.3.13 STATE_1_ERROR . . . . .	56



10.5.3.14 STATE_2_IMU . . . . .	56
10.5.3.15 STATE_3_BUTTON_LED . . . . .	56
10.5.3.16 STATE_4_TRANSFER . . . . .	57
10.6 Ball_Launcher_Controller/Core/Src/mpu6050.c File Reference . . . . .	57
10.6.1 Detailed Description . . . . .	57
10.6.2 Function Documentation . . . . .	57
10.6.2.1 mpu6050_calibrate() . . . . .	57
10.6.2.2 mpu6050_get_gX() . . . . .	58
10.6.2.3 mpu6050_get_gY() . . . . .	58
10.6.2.4 mpu6050_get_gZ() . . . . .	58
10.6.2.5 mpu6050_init() . . . . .	58
10.6.2.6 mpu6050_update() . . . . .	59
10.7 Ball_Launcher_Controller/Core/Src/stm32f4xx_hal_msp_C.c File Reference . . . . .	59
10.7.1 Detailed Description . . . . .	60
10.7.2 Function Documentation . . . . .	60
10.7.2.1 HAL_I2C_MspDeInit() . . . . .	60
10.7.2.2 HAL_I2C_MspInit() . . . . .	60
10.7.2.3 HAL_MspInit() . . . . .	61
10.7.2.4 HAL_UART_MspDeInit() . . . . .	61
10.7.2.5 HAL_UART_MspInit() . . . . .	61
10.8 Ball_Launcher_Controller/Core/Src/stm32f4xx_it_C.c File Reference . . . . .	62
10.8.1 Detailed Description . . . . .	62
10.8.2 Function Documentation . . . . .	62
10.8.2.1 BusFault_Handler() . . . . .	62
10.8.2.2 DebugMon_Handler() . . . . .	63
10.8.2.3 HardFault_Handler() . . . . .	63
10.8.2.4 MemManage_Handler() . . . . .	63
10.8.2.5 NMI_Handler() . . . . .	63
10.8.2.6 PendSV_Handler() . . . . .	63
10.8.2.7 SVC_Handler() . . . . .	63
10.8.2.8 SysTick_Handler() . . . . .	63
10.8.2.9 UsageFault_Handler() . . . . .	64
10.9 Ball_Launcher_Controller/Core/Src/syscalls_C.c File Reference . . . . .	64
10.9.1 Detailed Description . . . . .	65
10.9.2 Function Documentation . . . . .	65
10.9.2.1 __attribute__() . . . . .	65
10.9.2.2 __io_getchar() . . . . .	65
10.9.2.3 __io_putchar() . . . . .	65
10.9.2.4 _close() . . . . .	65
10.9.2.5 _execve() . . . . .	65
10.9.2.6 _exit() . . . . .	66
10.9.2.7 _fork() . . . . .	66

10.9.2.8 _fstat()	66
10.9.2.9 _getpid()	66
10.9.2.10 _isatty()	66
10.9.2.11 _kill()	66
10.9.2.12 _link()	66
10.9.2.13 _lseek()	66
10.9.2.14 _open()	67
10.9.2.15 _stat()	67
10.9.2.16 _times()	67
10.9.2.17 _unlink()	67
10.9.2.18 _wait()	67
10.9.2.19 initialise_monitor_handles()	67
10.9.3 Variable Documentation	67
10.9.3.1 environ	67
10.10 Ball_Launcher_Controller/Core/Src/systemem_C.c File Reference	67
10.10.1 Detailed Description	68
10.10.2 Function Documentation	68
10.10.2.1 _sbrk()	68
10.11 Ball_Launcher_Controller/Core/Src/system_stm32f4xx_C.c File Reference	69
10.11.1 Detailed Description	69
10.12 Ball_Launcher_Main/Core/Inc/d4215.h File Reference	70
10.12.1 Detailed Description	70
10.12.2 Function Documentation	70
10.12.2.1 D4215_init()	70
10.12.2.2 D4215_set()	71
10.13 Ball_Launcher_Main/Core/Inc/led.h File Reference	71
10.13.1 Detailed Description	71
10.13.2 Function Documentation	72
10.13.2.1 LED_init()	72
10.13.2.2 LED_off()	72
10.13.2.3 LED_on()	72
10.13.2.4 LED_toggle()	73
10.14 Ball_Launcher_Main/Core/Inc/main.h File Reference	73
10.14.1 Detailed Description	73
10.14.2 Function Documentation	73
10.14.2.1 Error_Handler()	73
10.14.2.2 HAL_TIM_MspPostInit()	74
10.15 Ball_Launcher_Main/Core/Inc/radio_driver.h File Reference	74
10.15.1 Detailed Description	74
10.15.2 Function Documentation	74
10.15.2.1 capturePulseWidth()	74
10.15.2.2 Radio_getPulseWidth()	75

10.15.2.3 Radio_init() . . . . .	75
10.15.2.4 update() . . . . .	76
10.16 Ball_Launcher_Main/Core/Inc/stepper_driver.h File Reference . . . . .	76
10.16.1 Detailed Description . . . . .	77
10.16.2 Function Documentation . . . . .	77
10.16.2.1 Delay_us() . . . . .	77
10.16.2.2 Stepper_disable() . . . . .	77
10.16.2.3 Stepper_enable() . . . . .	77
10.16.2.4 Stepper_init() . . . . .	77
10.16.2.5 Stepper_setspeed() . . . . .	78
10.16.2.6 SysTick_Init() . . . . .	78
10.17 Ball_Launcher_Main/Core/Inc/stm32f4xx_hal_conf.h File Reference . . . . .	79
10.17.1 Macro Definition Documentation . . . . .	81
10.17.1.1 assert_param . . . . .	81
10.17.1.2 DATA_CACHE_ENABLE . . . . .	81
10.17.1.3 DP83848_PHY_ADDRESS . . . . .	81
10.17.1.4 ETH_RX_BUF_SIZE . . . . .	81
10.17.1.5 ETH_RXBUFNB . . . . .	81
10.17.1.6 ETH_TX_BUF_SIZE . . . . .	82
10.17.1.7 ETH_TXBUFNB . . . . .	82
10.17.1.8 EXTERNAL_CLOCK_VALUE . . . . .	82
10.17.1.9 HAL_CORTEX_MODULE_ENABLED . . . . .	82
10.17.1.10 HAL_DMA_MODULE_ENABLED . . . . .	82
10.17.1.11 HAL_EXTI_MODULE_ENABLED . . . . .	82
10.17.1.12 HAL_FLASH_MODULE_ENABLED . . . . .	82
10.17.1.13 HAL_GPIO_MODULE_ENABLED . . . . .	82
10.17.1.14 HAL_MODULE_ENABLED . . . . .	82
10.17.1.15 HAL_PWR_MODULE_ENABLED . . . . .	83
10.17.1.16 HAL_RCC_MODULE_ENABLED . . . . .	83
10.17.1.17 HAL_TIM_MODULE_ENABLED . . . . .	83
10.17.1.18 HAL_UART_MODULE_ENABLED . . . . .	83
10.17.1.19 HSE_STARTUP_TIMEOUT . . . . .	83
10.17.1.20 HSE_VALUE . . . . .	83
10.17.1.21 HSI_VALUE . . . . .	83
10.17.1.22 INSTRUCTION_CACHE_ENABLE . . . . .	83
10.17.1.23 LSE_STARTUP_TIMEOUT . . . . .	84
10.17.1.24 LSE_VALUE . . . . .	84
10.17.1.25 LSI_VALUE . . . . .	84
10.17.1.26 MAC_ADDR0 . . . . .	84
10.17.1.27 MAC_ADDR1 . . . . .	84
10.17.1.28 MAC_ADDR2 . . . . .	84
10.17.1.29 MAC_ADDR3 . . . . .	84

10.17.1.30 MAC_ADDR4 . . . . .	84
10.17.1.31 MAC_ADDR5 . . . . .	85
10.17.1.32 PHY_AUTONEGO_COMPLETE . . . . .	85
10.17.1.33 PHY_AUTONEGOTIATION . . . . .	85
10.17.1.34 PHY_BCR . . . . .	85
10.17.1.35 PHY_BSR . . . . .	85
10.17.1.36 PHY_CONFIG_DELAY . . . . .	85
10.17.1.37 PHY_DUPLEX_STATUS . . . . .	85
10.17.1.38 PHY_FULLDUPLEX_100M . . . . .	86
10.17.1.39 PHY_FULLDUPLEX_10M . . . . .	86
10.17.1.40 PHY_HALFDUPLEX_100M . . . . .	86
10.17.1.41 PHY_HALFDUPLEX_10M . . . . .	86
10.17.1.42 PHY_ISOLATE . . . . .	86
10.17.1.43 PHY_JABBER_DETECTION . . . . .	86
10.17.1.44 PHY_LINKED_STATUS . . . . .	86
10.17.1.45 PHY_LOOPBACK . . . . .	87
10.17.1.46 PHY_POWERDOWN . . . . .	87
10.17.1.47 PHY_READ_TO . . . . .	87
10.17.1.48 PHY_RESET . . . . .	87
10.17.1.49 PHY_RESET_DELAY . . . . .	87
10.17.1.50 PHY_RESTART_AUTONEGOTIATION . . . . .	87
10.17.1.51 PHY_SPEED_STATUS . . . . .	87
10.17.1.52 PHY_SR . . . . .	87
10.17.1.53 PHY_WRITE_TO . . . . .	88
10.17.1.54 PREFETCH_ENABLE . . . . .	88
10.17.1.55 TICK_INT_PRIORITY . . . . .	88
10.17.1.56 USE_HAL_ADC_REGISTER_CALLBACKS . . . . .	88
10.17.1.57 USE_HAL_CAN_REGISTER_CALLBACKS . . . . .	88
10.17.1.58 USE_HAL_CEC_REGISTER_CALLBACKS . . . . .	88
10.17.1.59 USE_HAL_CRYPT_REGISTER_CALLBACKS . . . . .	88
10.17.1.60 USE_HAL_DAC_REGISTER_CALLBACKS . . . . .	88
10.17.1.61 USE_HAL_DCMI_REGISTER_CALLBACKS . . . . .	88
10.17.1.62 USE_HAL_DFSDM_REGISTER_CALLBACKS . . . . .	88
10.17.1.63 USE_HAL_DMA2D_REGISTER_CALLBACKS . . . . .	89
10.17.1.64 USE_HAL_DSI_REGISTER_CALLBACKS . . . . .	89
10.17.1.65 USE_HAL_ETH_REGISTER_CALLBACKS . . . . .	89
10.17.1.66 USE_HAL_FMPI2C_REGISTER_CALLBACKS . . . . .	89
10.17.1.67 USE_HAL_FMPSMBUS_REGISTER_CALLBACKS . . . . .	89
10.17.1.68 USE_HAL_HASH_REGISTER_CALLBACKS . . . . .	89
10.17.1.69 USE_HAL_HCD_REGISTER_CALLBACKS . . . . .	89
10.17.1.70 USE_HAL_I2C_REGISTER_CALLBACKS . . . . .	89
10.17.1.71 USE_HAL_I2S_REGISTER_CALLBACKS . . . . .	89

10.17.1.72 USE_HAL_IRDA_REGISTER_CALLBACKS . . . . .	89
10.17.1.73 USE_HAL_LPTIM_REGISTER_CALLBACKS . . . . .	90
10.17.1.74 USE_HAL_LTDC_REGISTER_CALLBACKS . . . . .	90
10.17.1.75 USE_HAL_MMC_REGISTER_CALLBACKS . . . . .	90
10.17.1.76 USE_HAL_NAND_REGISTER_CALLBACKS . . . . .	90
10.17.1.77 USE_HAL_NOR_REGISTER_CALLBACKS . . . . .	90
10.17.1.78 USE_HAL_PCCARD_REGISTER_CALLBACKS . . . . .	90
10.17.1.79 USE_HAL_PCD_REGISTER_CALLBACKS . . . . .	90
10.17.1.80 USE_HAL_QSPI_REGISTER_CALLBACKS . . . . .	90
10.17.1.81 USE_HAL_RNG_REGISTER_CALLBACKS . . . . .	90
10.17.1.82 USE_HAL_RTC_REGISTER_CALLBACKS . . . . .	90
10.17.1.83 USE_HAL_SAI_REGISTER_CALLBACKS . . . . .	91
10.17.1.84 USE_HAL_SD_REGISTER_CALLBACKS . . . . .	91
10.17.1.85 USE_HAL_SDRAM_REGISTER_CALLBACKS . . . . .	91
10.17.1.86 USE_HAL_SMARTCARD_REGISTER_CALLBACKS . . . . .	91
10.17.1.87 USE_HAL_SMBUS_REGISTER_CALLBACKS . . . . .	91
10.17.1.88 USE_HAL_SPDIFRX_REGISTER_CALLBACKS . . . . .	91
10.17.1.89 USE_HAL_SPI_REGISTER_CALLBACKS . . . . .	91
10.17.1.90 USE_HAL_SRAM_REGISTER_CALLBACKS . . . . .	91
10.17.1.91 USE_HAL_TIM_REGISTER_CALLBACKS . . . . .	91
10.17.1.92 USE_HAL_UART_REGISTER_CALLBACKS . . . . .	91
10.17.1.93 USE_HAL_USART_REGISTER_CALLBACKS . . . . .	92
10.17.1.94 USE_HAL_WWDG_REGISTER_CALLBACKS . . . . .	92
10.17.1.95 USE_RTOS . . . . .	92
10.17.1.96 USE_SPI_CRC . . . . .	92
10.17.1.97 VDD_VALUE . . . . .	92
10.18 Ball_Launcher_Main/Core/Inc/stm32f4xx_it.h File Reference . . . . .	92
10.18.1 Detailed Description . . . . .	93
10.18.2 Function Documentation . . . . .	93
10.18.2.1 BusFault_Handler() . . . . .	93
10.18.2.2 DebugMon_Handler() . . . . .	93
10.18.2.3 HardFault_Handler() . . . . .	93
10.18.2.4 MemManage_Handler() . . . . .	93
10.18.2.5 NMI_Handler() . . . . .	93
10.18.2.6 PendSV_Handler() . . . . .	94
10.18.2.7 SVC_Handler() . . . . .	94
10.18.2.8 SysTick_Handler() . . . . .	94
10.18.2.9 TIM4_IRQHandler() . . . . .	94
10.18.2.10 UsageFault_Handler() . . . . .	94
10.18.2.11 USART1_IRQHandler() . . . . .	94
10.19 Ball_Launcher_Main/Core/Inc/switch.h File Reference . . . . .	94
10.19.1 Detailed Description . . . . .	95

10.19.2 Function Documentation . . . . .	95
10.19.2.1 Switch_getStatus() . . . . .	95
10.19.2.2 Switch_init() . . . . .	95
10.20 Ball_Launcher_Main/Core/Src/d4215.c File Reference . . . . .	96
10.20.1 Detailed Description . . . . .	96
10.20.2 Function Documentation . . . . .	96
10.20.2.1 D4215_init() . . . . .	96
10.20.2.2 D4215_set() . . . . .	97
10.21 Ball_Launcher_Main/Core/Src/led.c File Reference . . . . .	97
10.21.1 Detailed Description . . . . .	98
10.21.2 Function Documentation . . . . .	98
10.21.2.1 LED_init() . . . . .	98
10.21.2.2 LED_off() . . . . .	98
10.21.2.3 LED_on() . . . . .	98
10.21.2.4 LED_toggle() . . . . .	99
10.22 Ball_Launcher_Main/Core/Src/main.c File Reference . . . . .	99
10.22.1 Detailed Description . . . . .	100
10.22.2 Function Documentation . . . . .	101
10.22.2.1 dataProcess() . . . . .	101
10.22.2.2 Error_Handler() . . . . .	101
10.22.2.3 HAL_TIM_IC_CaptureCallback() . . . . .	101
10.22.2.4 HAL_UART_RxCpltCallback() . . . . .	101
10.22.2.5 main() . . . . .	101
10.22.2.6 map() . . . . .	101
10.22.2.7 speedCalculate() . . . . .	102
10.22.2.8 SystemClock_Config() . . . . .	102
10.22.3 Variable Documentation . . . . .	102
10.22.3.1 angleTarget . . . . .	102
10.22.3.2 CAL . . . . .	102
10.22.3.3 charIn . . . . .	102
10.22.3.4 data . . . . .	102
10.22.3.5 dt . . . . .	103
10.22.3.6 ESTOP . . . . .	103
10.22.3.7 fall . . . . .	103
10.22.3.8 HOME . . . . .	103
10.22.3.9 htim2 . . . . .	103
10.22.3.10 htim4 . . . . .	103
10.22.3.11 huart1 . . . . .	103
10.22.3.12 huart6 . . . . .	103
10.22.3.13 idx . . . . .	103
10.22.3.14 MOVE . . . . .	103
10.22.3.15 pitch . . . . .	104

10.22.3.16 pitchDirection . . . . .	104
10.22.3.17 pitchDirectionSW . . . . .	104
10.22.3.18 rise . . . . .	104
10.22.3.19 SHOT . . . . .	104
10.22.3.20 STATE . . . . .	104
10.22.3.21 STATE_0_INIT . . . . .	104
10.22.3.22 STATE_1_HUB . . . . .	104
10.22.3.23 STATE_2_ESTOP . . . . .	104
10.22.3.24 STATE_3_STEPPER . . . . .	104
10.22.3.25 STATE_4_BLDC . . . . .	105
10.22.3.26 SW . . . . .	105
10.22.3.27 SW1 . . . . .	105
10.22.3.28 SW2 . . . . .	105
10.22.3.29 SW3 . . . . .	105
10.22.3.30 tf . . . . .	105
10.22.3.31 ti . . . . .	105
10.22.3.32 total . . . . .	105
10.22.3.33 yaw . . . . .	105
10.22.3.34 yawDirection . . . . .	105
10.22.3.35 yawDirectionSW . . . . .	106
10.23 Ball_Launcher_Main/Core/Src/radio_driver.c File Reference . . . . .	106
10.23.1 Detailed Description . . . . .	106
10.23.2 Function Documentation . . . . .	106
10.23.2.1 capturePulseWidth() . . . . .	106
10.23.2.2 Radio_getPulseWidth() . . . . .	107
10.23.2.3 Radio_init() . . . . .	107
10.23.2.4 update() . . . . .	107
10.24 Ball_Launcher_Main/Core/Src/stepper_driver.c File Reference . . . . .	108
10.24.1 Detailed Description . . . . .	108
10.24.2 Function Documentation . . . . .	108
10.24.2.1 Delay_us() . . . . .	108
10.24.2.2 Stepper_disable() . . . . .	109
10.24.2.3 Stepper_enable() . . . . .	109
10.24.2.4 Stepper_init() . . . . .	109
10.24.2.5 Stepper_setspeed() . . . . .	110
10.24.2.6 SysTick_Init() . . . . .	110
10.25 Ball_Launcher_Main/Core/Src/stm32f4xx_hal_msp.c File Reference . . . . .	110
10.25.1 Detailed Description . . . . .	111
10.25.2 Function Documentation . . . . .	111
10.25.2.1 HAL_MspInit() . . . . .	111
10.25.2.2 HAL_TIM_IC_MspDeInit() . . . . .	111
10.25.2.3 HAL_TIM_IC_MspInit() . . . . .	112

10.25.2.4 HAL_TIM_MspPostInit()	112
10.25.2.5 HAL_TIM_PWM_MspDeInit()	112
10.25.2.6 HAL_TIM_PWM_MspInit()	113
10.25.2.7 HAL_UART_MspDeInit()	113
10.25.2.8 HAL_UART_MspInit()	113
10.26 Ball_Launcher_Main/Core/Src/stm32f4xx_it.c File Reference	114
10.26.1 Detailed Description	115
10.26.2 Function Documentation	115
10.26.2.1 BusFault_Handler()	115
10.26.2.2 DebugMon_Handler()	115
10.26.2.3 HardFault_Handler()	115
10.26.2.4 MemManage_Handler()	115
10.26.2.5 NMI_Handler()	115
10.26.2.6 PendSV_Handler()	116
10.26.2.7 SVC_Handler()	116
10.26.2.8 SysTick_Handler()	116
10.26.2.9 TIM4_IRQHandler()	116
10.26.2.10 UsageFault_Handler()	116
10.26.2.11 USART1_IRQHandler()	116
10.26.3 Variable Documentation	116
10.26.3.1 htim4	116
10.26.3.2 huart1	117
10.27 Ball_Launcher_Main/Core/Src/switch.c File Reference	117
10.27.1 Detailed Description	117
10.27.2 Function Documentation	117
10.27.2.1 Switch_getStatus()	117
10.27.2.2 Switch_init()	118
10.28 Ball_Launcher_Main/Core/Src/syscalls.c File Reference	118
10.28.1 Detailed Description	119
10.28.2 Function Documentation	119
10.28.2.1 __attribute__()	119
10.28.2.2 __io_getchar()	119
10.28.2.3 __io_putchar()	120
10.28.2.4 _close()	120
10.28.2.5 _execve()	120
10.28.2.6 _exit()	120
10.28.2.7 _fork()	120
10.28.2.8 _fstat()	120
10.28.2.9 _getpid()	120
10.28.2.10 _isatty()	120
10.28.2.11 _kill()	121
10.28.2.12 _link()	121



---

10.28.2.13 _lseek()	121
10.28.2.14 _open()	121
10.28.2.15 _stat()	121
10.28.2.16 _times()	121
10.28.2.17 _unlink()	121
10.28.2.18 _wait()	121
10.28.2.19 initialise_monitor_handles()	122
10.28.3 Variable Documentation	122
10.28.3.1 environ	122
10.29 Ball_Launcher_Main/Core/Src/systemem.c File Reference	122
10.29.1 Detailed Description	122
10.29.2 Function Documentation	123
10.29.2.1 _sbrk()	123
10.30 Ball_Launcher_Main/Core/Src/system_stm32f4xx.c File Reference	123
10.30.1 Detailed Description	124
10.31 pages/controller.c File Reference	124
10.31.1 Detailed Description	124
10.32 pages/launcher.c File Reference	124
10.32.1 Detailed Description	125
10.33 pages/mainpage.c File Reference	125
10.33.1 Detailed Description	125
10.34 pages/report.c File Reference	125
10.34.1 Detailed Description	125
<b>Index</b>	<b>127</b>



# Chapter 1

## MAIN PAGE

PICTURE UPDATE NEEDED

### 1.1 MOTIVATION

The motivation for this project is to build a prototype for a full-scale basketball-playing robot. The project will start with a small-scale version to test the concept and gather insights for future improvements. This approach will provide an opportunity to develop customized PCBAs that can control the robot for its intended purposes.

### 1.2 OBJECTIVE

The objective of the project includes integrating the mechanical design with PCBA electronic component. This project will primarily use 3D-printed parts, along with shafts and bearings to support the robot's housing. The PCBAs will be designed using Fusion 360 software and manufactured by JLCPCB. The programming for the STM32 microcontroller will be done using the C language.

Note: The image below is provided solely for illustrative purposes and does not represent the actual project.

The project is divided into two main assemblies:

- The controller will use an STM32 BlackPill microcontroller with an IMU sensor attached to the user's hand. The controller will send signals via Bluetooth to the launcher, allowing the launcher to receive the data and spin according to the IMU sensor's input.
- The launcher will be the main assembly, receiving data from the controller and moving in 2 degrees of freedom (yaw and pitch angles). It will then shoot the basketball when the controller send launching flag. The launcher will use a customized PCBA centered around the STM32F401CEU6 chip, which will act as the core component to determine movement.

### 1.3 IDEATION DESIGN

The overall design of the two main assemblies in the project is sketched and listed below for reference. The sketch represents a rough idea of how all components are connected and communicate with each other. The PDF of the sketch is also attached with higher solution can be found in by access through the picture below

## 1.4 CAD MODELING

The design is inspired by the Peashooter from the Plants vs. Zombies mobile game. In terms of mechanical design, custom machined parts are limited to facilitate rapid prototyping and maintain focus on electronics and programming. Most custom parts will be 3D printed to save time and effort, except for bearings and shafts, which will be purchased from McMaster-Carr as off-the-shelf components. The CAD modeling phase of the project was completed early to allow the main focus to be on electronics and program implementation. The CAD figure below shows the final design of the launcher. The SolidWorks CAD model can access through the CAD sketch below.

## 1.5 LAUNCHER

The launcher will be the main assembly, receiving data from the controller and moving in 2 degrees of freedom (yaw and pitch angles). It will then shoot the basketball when the controller sends out signals. The launcher will use a customized PCBA centered around the STM32F401CEU6 chip, which will act as the core component to determine movement. The launcher includes a cycloidal gearbox and stepper motors with a planetary gearbox for precise movement control. Additionally, the launcher is equipped with brushless motors and 3 limit switches to ensure accurate positioning. More detail about the launcher Bill of Materials, design, source code can be found by access through the launcher picture below:

## 1.6 CONTROLLER

The controller will use an STM32 BlackPill microcontroller with an IMU sensor attached to the user's hand. It will send signals via Bluetooth to the launcher, allowing the launcher to receive the data and rotate according to the IMU sensor's input. The data transfer includes the velocity of the angles, and the communication uses a UART connection between the HC-05 Bluetooth module. The controller also features buttons and an indicator LED, all mounted on a 3D-printed handle bracket. More detail about the controller's Bill of Materials, design, source code can be found by access through the launcher picture below:

## 1.7 FUTURE IMPROVEMENT

Even though, the project is built and run appropriately according to the project's objectives. The following improvements can be considered for future version. Especially, for the upcoming full scale basketball launcher

- Improve the IMU to a 9 DOF sensor that can sense accurate absolute angle
- Improve stronger stepper driver that can allow faster and stronger rotation motion
- Increase moving angle of the launcher to more than 120°

## 1.8 YOUTUBE REFERENCE

## 1.9 REPOSITORY REFERENCE

You may find it more useful to read through the website exploring the source code. All code that will be referenced in this portfolio relate to project is accessible through:

NOTE: You may find the Controller's code with a small specification "\_C" at the end of the file that stands for Controller code. Files' name and the files' header code need to be renamed without that term in case the projects are recreated. The inconvenience is due to the drawback of uploading codes in 2 projects to one GitHub Repository.

For example:

- **main\_C.c** (p. 53) means the main code file for the controller. Needs to be changed to **main.c** (p. 99) before running in any IDE.

## 1.10 CONTACT INFO

**Author:** Vinh Vo

**Email:** vinhvo.career@gmail.com

**LinkedIn:** <https://www.linkedin.com/in/vinhvo98/>

**Youtube:** [https://www.youtube.com/channel/UCh\\_4F4CJVqvAhHmCMTvIb-w](https://www.youtube.com/channel/UCh_4F4CJVqvAhHmCMTvIb-w)

**Major:** MS Mechanical Engineer at Cal Poly San Luis Obispo

**Date:** May 15, 2024



## Chapter 2

# CONTROLLER

### 2.1 FINAL DESIGN

The controller will use an STM32 BlackPill microcontroller with an IMU sensor attached to the user's hand. It will send signals via Bluetooth to the launcher, allowing the launcher to receive the data and rotate according to the IMU sensor's input. The data transfer includes the velocity of the angles, and the communication uses a UART connection between the HC-05 Bluetooth module. The controller also features buttons and an indicator LED, all mounted on a 3D-printed handle bracket.

### 2.2 CONTROLLER PCBA

Below are the overall schematic and footprint of the PCBA that is used to integrate the STM32 BlackPill, mpu6050 IMU, HC-05, and LEDs together. The PDF of this schematic is also attached with higher resolution can be found in the picture below

### 2.3 FINITE STATE MACHINE

State	Description	Actions	Transitions
STATE0	INIT - Attempt to connect to IMU	- Attempt to connect - Turn on LED1 if successful	- If unsuccessful, transition to <b>STATE 1</b> - If successful, transition to <b>STATE 2</b>
STATE1	ERROR- Failed to connect to IMU	- Toggle LED1 if failed to connect to the IMU	- Always transition to <b>STATE 0</b>
STATE2	IMU - Read data from the IMU	- Continuously read the data from the IMU	- Always transition to <b>STATE 3</b>
STATE3	BUTTON_LED - Read buttons & update LEDs	- Continuously read the button - Update the LEDs	- Always transition back to <b>STATE 4</b>
STATE4	TRANSFER - Send data to UART1 and UART6	- Send data to UART1 (bluetooth) - UART6 (debug)	- Always transition back to <b>STATE 2</b>

## **2.4 BILL OF MATERIAL - 3D PRINTED PARTS**

Most of the 3D printed parts are printed using BambuLab XC1 3D printer and BambuLab filaments. All the CAD 3D model files can be found here for reference:

## **2.5 BILL OF MATERIAL - ELECTRONICS**

All of the components for electronics are listed in the attached PDF file here below



## Chapter 3

# LAUNCHER

### 3.1 FINAL DESIGN

The launcher will be the main assembly, receiving data from the controller and moving in 2 degrees of freedom (yaw and pitch angles). It will then shoot the basketball when the controller sends out signals. The launcher will use a customized PCBA centered around the STM32F401CEU6 chip, which will act as the core component to determine movement. The launcher includes a cycloidal gearbox and stepper motors with a planetary gearbox for precise movement control. Additionally, the launcher is equipped with brushless motors and 3 limit switches to ensure accurate positioning.

--> UPDATE PICTURE

### 3.2 LAUNCHER PCBA

This PCB is the main component of the project, serving as the centerpiece of the launcher. It operates the launcher based on commands received from the controller. The PDF of this schematic is also attached with higher resolution can be found in [PDF SCHEMATIC](#). Below are all of the PCB schematic contents, totaling 4 pages, described in the table below.

Page	Description
1	Power source schematic for power the pcb, drivers, mcu, etc
2	All connection to power and program for STM32F401CEU6 chip
3	All connection to power and program for DRV8825 stepper motor driver
4	Additional pins + test pads and connectors for limit switches + stepper motors

### 3.3 FINITE STATE MACHINE

State	Description	Actions	Transitions
STATE0	INIT	- Init all objects in the system	- Always transition to <b>STATE 1</b>
STATE1	HOME POSITION CALIBRATION	- Launcher finding limit switches and homing	- Always transition to <b>STATE 2</b> if done calibration - Otherwise stay in <b>STATE 1</b>
STATE2	DECISION HUB	- Jumping to different state according to condition	- Move to <b>STATE 3</b> if MOVE == 1 & SHOT == 0 - Move to <b>STATE 4</b> if SHOT == 1 & MOVE == 0 - Move to <b>STATE 5</b> if hit any switches - Move to <b>STATE 6</b> if ESTOP == 1
STATE3	STEPPER RUN	- Running stepper motor	- Always transition to <b>STATE 2</b>
STATE4	BLDC RUN	- Running BLDC Motor until fast speed	- Always transition to <b>STATE 2</b>
STATE5	LIMIT SWITCH	- Return to home position after hitting the switches	- Stay in <b>STATE 5</b> until done homing - Then transition to <b>STATE 2</b> when done
STATE6	ESTOP	- Turn off everything if we hitting the ESTOP	- Stop the launcher immediately

### 3.4 BILL OF MATERIAL - CYCLOIDAL GEARBOX

#### CYCLOIDAL GEARBOX DEMONSTRATION

Components	Description	Image
Bearing	OD = 55 mm + ID = 35 mm + t = 10 mm	
Bearing	OD = 47 mm + ID = 35 mm + t = 7 mm	
Bearing	OD = 37 mm + ID = 25 mm + t = 7 mm	
Bearing	OD = 15 mm + ID = 6 mm + t = 5 mm	
Bolt	M6 + L = 70 mm + Bolt	
Nut	M6 + t = 5 mm + Nut	
Nut	M6 + t = 5 mm + Lock Nut	

### 3.5 BILL OF MATERIAL - MECHANICAL

Components	Description	Image
NEMA17	NEMA17 stepper motor	
NEMA17 + GEAR BOX	NEMA17 stepper motor + planetary gearbox	
D4215 BLDC	D4215 BLDC 650 Kvmotor	
UBEC ESC	Hobby ESC control for BLDC	
O-RING	Friction O-Ring	
SWITCH	Switch for BLDC	
BASKETBALL	5 inches basketball	
LIMIT SWITCH	Limit switches for safety	
BATTERY	Lipo battery 11.1 V	

### **3.6 BILL OF MATERIAL - 3D PRINTED PARTS**

Most of the 3D printed parts are printed using BambuLab XC1 3D printer and BambuLab filaments. All the CAD 3D model files can be found here for reference:

### **3.7 BILL OF MATERIAL - ELECTRONICS**

All of the components for electronics are listed in the attached PDF file here below

## **Chapter 4**

# **REPORTS**

### **4.1 PROPOSAL**

### **4.2 BILL OF MATERIAL**

### **4.3 REFERENCE MANUAL**

### **4.4 OVERALL REPORT**



# Chapter 5

## Topic Index

### 5.1 Topics

Here is a list of all topics with brief descriptions:

CMSIS . . . . .	19
Stm32f4xx_system . . . . .	19
STM32F4xx_System_Private_Includes . . . . .	19
STM32F4xx_System_Private_TypesDefinitions . . . . .	20
STM32F4xx_System_Private_Defines . . . . .	20
STM32F4xx_System_Private_Macros . . . . .	20
STM32F4xx_System_Private_Variables . . . . .	20
STM32F4xx_System_Private_FunctionPrototypes . . . . .	21
STM32F4xx_System_Private_Functions . . . . .	21





## Chapter 6

# Data Structure Index

### 6.1 Data Structures

Here are the data structures with brief descriptions:

<b>D4215X</b>	Structure to define a D4215 motor with timer, channel, and speed . . . . .	25
<b>LEDX</b>	Structure to define an LED with GPIO port and pin . . . . .	26
<b>MPU6050</b>	Struct representing a mpu6050 imu sensor . . . . .	27
<b>RadioX</b>	Structure to define a radio driver with timer, channels, and pulse width parameters . . . . .	28
<b>StepperX</b>	Structure to define a stepper motor driver with GPIO ports and pins . . . . .	30
<b>SwitchX</b>	Structure to define a switch with GPIO port, pin, and status . . . . .	31



# Chapter 7

## File Index

### 7.1 File List

Here is a list of all files with brief descriptions:

Ball_Launcher_Controller/Core/Inc/ <b>main_C.h</b>	
Header for <b>main_C.c</b> (p. 53) file. This file contains the common defines of the application . . .	33
Ball_Launcher_Controller/Core/Inc/ <b>mpu6050.h</b>	
Header for <b>mpu6050.c</b> (p. 57) file. This file contains the common defines of the application . .	34
Ball_Launcher_Controller/Core/Inc/ <b>stm32f4xx_hal_conf_C.h</b>	
HAL configuration template file. This file should be copied to the application folder and renamed to <b>stm32f4xx_hal_conf_C.h</b> (p. 37) . . . . .	37
Ball_Launcher_Controller/Core/Inc/ <b>stm32f4xx_it_C.h</b>	
This file contains the headers of the interrupt handlers . . . . .	51
Ball_Launcher_Controller/Core/Src/ <b>main_C.c</b>	
Main program body . . . . .	53
Ball_Launcher_Controller/Core/Src/ <b>mpu6050.c</b>	
Implementation of mpu6050 sensor functions . . . . .	57
Ball_Launcher_Controller/Core/Src/ <b>stm32f4xx_hal_msp_C.c</b>	
This file provides code for the MSP Initialization and de-Initialization codes . . . . .	59
Ball_Launcher_Controller/Core/Src/ <b>stm32f4xx_it_C.c</b>	
Interrupt Service Routines . . . . .	62
Ball_Launcher_Controller/Core/Src/ <b>syscalls_C.c</b>	
STM32CubeIDE Minimal System calls file . . . . .	64
Ball_Launcher_Controller/Core/Src/ <b>sysmem_C.c</b>	
STM32CubeIDE System Memory calls file . . . . .	67
Ball_Launcher_Controller/Core/Src/ <b>system_stm32f4xx_C.c</b>	
CMSIS Cortex-M4 Device Peripheral Access Layer System Source File . . . . .	69
Ball_Launcher_Main/Core/Inc/ <b>d4215.h</b>	
Header for <b>d4215.c</b> (p. 96) file. This file contains the common defines of the application . . . .	70
Ball_Launcher_Main/Core/Inc/ <b>led.h</b>	
Header for <b>led.c</b> (p. 97) file. This file contains the common defines of the application . . . . .	71
Ball_Launcher_Main/Core/Inc/ <b>main.h</b>	
Header for <b>main.c</b> (p. 99) file. This file contains the common defines of the application . . . . .	73
Ball_Launcher_Main/Core/Inc/ <b>radio_driver.h</b>	
Header for <b>radio_driver.c</b> (p. 106) file. This file contains the common defines of the application	74
Ball_Launcher_Main/Core/Inc/ <b>stepper_driver.h</b>	
Header for <b>stepper_driver.c</b> (p. 108) file. This file contains the common defines of the application	76
Ball_Launcher_Main/Core/Inc/ <b>stm32f4xx_hal_conf.h</b> . . . . .	79
Ball_Launcher_Main/Core/Inc/ <b>stm32f4xx_it.h</b>	
This file contains the headers of the interrupt handlers . . . . .	92

Ball_Launcher_Main/Core/Inc/ <b>switch.h</b>	
Header for <b>switch.c</b> (p. 117) file. This file contains the common defines of the application . . .	94
Ball_Launcher_Main/Core/Src/ <b>d4215.c</b>	
Source file for D4215 motor operations. This file contains the implementation of the functions for initializing and controlling a D4215 motor . . . . .	96
Ball_Launcher_Main/Core/Src/ <b>led.c</b>	
Source file for LED operations. This file contains the implementation of the functions for initializing and controlling an LED . . . . .	97
Ball_Launcher_Main/Core/Src/ <b>main.c</b>	
Main program body . . . . .	99
Ball_Launcher_Main/Core/Src/ <b>radio_driver.c</b>	
Source file for radio driver operations. This file contains the implementation of the functions for initializing and handling a radio driver . . . . .	106
Ball_Launcher_Main/Core/Src/ <b>stepper_driver.c</b>	
Source file for stepper motor driver operations. This file contains the implementation of the functions for initializing and controlling a stepper motor . . . . .	108
Ball_Launcher_Main/Core/Src/ <b>stm32f4xx_hal_msp.c</b>	
This file provides code for the MSP Initialization and de-Initialization codes . . . . .	110
Ball_Launcher_Main/Core/Src/ <b>stm32f4xx_it.c</b>	
Interrupt Service Routines . . . . .	114
Ball_Launcher_Main/Core/Src/ <b>switch.c</b>	
Source file for switch operations. This file contains the implementation of the functions for initializing and getting the status of a switch . . . . .	117
Ball_Launcher_Main/Core/Src/ <b>syscalls.c</b>	
STM32CubeIDE Minimal System calls file . . . . .	118
Ball_Launcher_Main/Core/Src/ <b>sysmem.c</b>	
STM32CubeIDE System Memory calls file . . . . .	122
Ball_Launcher_Main/Core/Src/ <b>system_stm32f4xx.c</b>	
CMSIS Cortex-M4 Device Peripheral Access Layer System Source File . . . . .	123
pages/ <b>controller.c</b>	
Controller Component Documentation . . . . .	124
pages/ <b>launcher.c</b>	
Launcher Component Documentation . . . . .	124
pages/ <b>mainpage.c</b>	
Main Page Documentation for Project . . . . .	125
pages/ <b>report.c</b>	
Launcher Documentation . . . . .	125

# Chapter 8

## Topic Documentation

### 8.1 CMSIS

#### Topics

- `Stm32f4xx_system`

#### 8.1.1 Detailed Description

#### 8.1.2 `Stm32f4xx_system`

#### Topics

- `STM32F4xx_System_Private_Includes`
- `STM32F4xx_System_Private_TypesDefinitions`
- `STM32F4xx_System_Private_Defines`
- `STM32F4xx_System_Private_Macros`
- `STM32F4xx_System_Private_Variables`
- `STM32F4xx_System_Private_FunctionPrototypes`
- `STM32F4xx_System_Private_Functions`

#### 8.1.2.1 Detailed Description

#### 8.1.2.2 `STM32F4xx_System_Private_Includes`

#### Macros

- `#define HSE_VALUE ((uint32_t)25000000)`
- `#define HSI_VALUE ((uint32_t)16000000)`
- `#define HSE_VALUE ((uint32_t)25000000)`
- `#define HSI_VALUE ((uint32_t)16000000)`

#### 8.1.2.2.1 Detailed Description

#### 8.1.2.2.2 Macro Definition Documentation

##### 8.1.2.2.2.1 HSE\_VALUE [1/2]

```
#define HSE_VALUE ((uint32_t)25000000)
```

Default value of the External oscillator in Hz

##### 8.1.2.2.2.2 HSE\_VALUE [2/2]

```
#define HSE_VALUE ((uint32_t)25000000)
```

Default value of the External oscillator in Hz

##### 8.1.2.2.2.3 HSI\_VALUE [1/2]

```
#define HSI_VALUE ((uint32_t)16000000)
```

Value of the Internal oscillator in Hz

##### 8.1.2.2.2.4 HSI\_VALUE [2/2]

```
#define HSI_VALUE ((uint32_t)16000000)
```

Value of the Internal oscillator in Hz

#### 8.1.2.3 STM32F4xx\_System\_Private\_TypesDefinitions

#### 8.1.2.4 STM32F4xx\_System\_Private\_Defines

#### 8.1.2.5 STM32F4xx\_System\_Private\_Macros

#### 8.1.2.6 STM32F4xx\_System\_Private\_Variables

#### Variables

- uint32\_t **SystemCoreClock** = 16000000
- const uint8\_t **AHBPrescTable** [16] = {0, 0, 0, 0, 0, 0, 0, 0, 1, 2, 3, 4, 6, 7, 8, 9}
- const uint8\_t **APBPrescTable** [8] = {0, 0, 0, 0, 1, 2, 3, 4}
- uint32\_t **SystemCoreClock** = 16000000
- const uint8\_t **AHBPrescTable** [16] = {0, 0, 0, 0, 0, 0, 0, 0, 1, 2, 3, 4, 6, 7, 8, 9}
- const uint8\_t **APBPrescTable** [8] = {0, 0, 0, 0, 1, 2, 3, 4}

### 8.1.2.6.1 Detailed Description

### 8.1.2.6.2 Variable Documentation

#### 8.1.2.6.2.1 AHBPrescTable [1/2]

```
const uint8_t AHBPrescTable[16] = {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 2, 3, 4, 6, 7, 8, 9}
```

#### 8.1.2.6.2.2 AHBPrescTable [2/2]

```
const uint8_t AHBPrescTable[16] = {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 2, 3, 4, 6, 7, 8, 9}
```

#### 8.1.2.6.2.3 APBPrescTable [1/2]

```
const uint8_t APBPrescTable[8] = {0, 0, 0, 0, 1, 2, 3, 4}
```

#### 8.1.2.6.2.4 APBPrescTable [2/2]

```
const uint8_t APBPrescTable[8] = {0, 0, 0, 0, 1, 2, 3, 4}
```

#### 8.1.2.6.2.5 SystemCoreClock [1/2]

```
uint32_t SystemCoreClock = 16000000
```

#### 8.1.2.6.2.6 SystemCoreClock [2/2]

```
uint32_t SystemCoreClock = 16000000
```

### 8.1.2.7 STM32F4xx\_System\_Private\_FunctionPrototypes

### 8.1.2.8 STM32F4xx\_System\_Private\_Functions

#### Functions

- void **SystemInit** (void)  
*Setup the microcontroller system Initialize the FPU setting, vector table location and External memory configuration.*
- void **SystemCoreClockUpdate** (void)  
*Update SystemCoreClock variable according to Clock Register Values. The SystemCoreClock variable contains the core clock (HCLK), it can be used by the user application to setup the SysTick timer or configure other parameters.*

### 8.1.2.8.1 Detailed Description

### 8.1.2.8.2 Function Documentation

#### 8.1.2.8.2.1 SystemCoreClockUpdate()

```
void SystemCoreClockUpdate (
    void )
```

Update SystemCoreClock variable according to Clock Register Values. The SystemCoreClock variable contains the core clock (HCLK), it can be used by the user application to setup the SysTick timer or configure other parameters.

#### Note

Each time the core clock (HCLK) changes, this function must be called to update SystemCoreClock variable value. Otherwise, any configuration based on this variable will be incorrect.

- The system frequency computed by this function is not the real frequency in the chip. It is calculated based on the predefined constant and the selected clock source:

- If SYSCLK source is HSI, SystemCoreClock will contain the **HSI\_VALUE(\*)** (p. 20)
- If SYSCLK source is HSE, SystemCoreClock will contain the **HSE\_VALUE(\*\*)** (p. 20)
- If SYSCLK source is PLL, SystemCoreClock will contain the **HSE\_VALUE(\*\*)** (p. 20) or **HSI\_VALUE(\*)** (p. 20) multiplied/divided by the PLL factors.

(\*) HSI\_VALUE is a constant defined in **stm32f4xx\_hal\_conf.h** (p. 79) file (default value 16 MHz) but the real value may vary depending on the variations in voltage and temperature.

(\*\*) HSE\_VALUE is a constant defined in **stm32f4xx\_hal\_conf.h** (p. 79) file (its value depends on the application requirements), user has to ensure that HSE\_VALUE is same as the real frequency of the crystal used. Otherwise, this function may have wrong result.

- The result of this function could be not correct when using fractional value for HSE crystal.

#### Parameters

None	
------	--

#### Return values

None	
------	--

#### 8.1.2.8.2.2 SystemInit()

```
void SystemInit (
    void )
```



Setup the microcontroller system Initialize the FPU setting, vector table location and External memory configuration.

**Parameters**

<i>None</i>	
-------------	--

**Return values**

<i>None</i>	
-------------	--

## Chapter 9

# Data Structure Documentation

### 9.1 D4215X Struct Reference

Structure to define a D4215 motor with timer, channel, and speed.

```
#include <d4215.h>
```

#### Data Fields

- TIM\_HandleTypeDef \* **timer**
- uint32\_t **channel**
- int32\_t **speed**

#### 9.1.1 Detailed Description

Structure to define a D4215 motor with timer, channel, and speed.

#### 9.1.2 Field Documentation

##### 9.1.2.1 channel

```
uint32_t channel
```

Timer channel for the motor

##### 9.1.2.2 speed

```
int32_t speed
```

Speed of the motor

### 9.1.2.3 timer

```
TIM_HandleTypeDef* timer
```

Timer handle for the motor

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

- Ball\_Launcher\_Main/Core/Inc/ **d4215.h**

## 9.2 LEDX Struct Reference

Structure to define an LED with GPIO port and pin.

```
#include <led.h>
```

### Data Fields

- GPIO\_TypeDef \* **GPIOx**
- uint16\_t **PIN**

### 9.2.1 Detailed Description

Structure to define an LED with GPIO port and pin.

### 9.2.2 Field Documentation

#### 9.2.2.1 GPIOx

```
GPIO_TypeDef* GPIOx
```

GPIO port of the LED

#### 9.2.2.2 PIN

```
uint16_t PIN
```

GPIO pin of the LED

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

- Ball\_Launcher\_Main/Core/Inc/ **led.h**

## 9.3 MPU6050 Struct Reference

Struct representing a mpu6050 imu sensor.

```
#include <mpu6050.h>
```

### Data Fields

- I2C\_HandleTypeDef \* **hi2c**
- HAL\_StatusTypeDef **status**
- uint16\_t **addr**
- uint16\_t **dt**
- int16\_t **gX**
- int16\_t **gY**
- int16\_t **gZ**
- int16\_t **gX\_offset**
- int16\_t **gY\_offset**
- int16\_t **gZ\_offset**

### 9.3.1 Detailed Description

Struct representing a mpu6050 imu sensor.

### 9.3.2 Field Documentation

#### 9.3.2.1 addr

```
uint16_t addr
```

I2C address

#### 9.3.2.2 dt

```
uint16_t dt
```

Delta time

#### 9.3.2.3 gX

```
int16_t gX
```

Gyroscope X axis data

#### 9.3.2.4 gX\_offset

```
int16_t gX_offset
```

Gyroscope X axis offset

#### 9.3.2.5 gY

`int16_t gY`

Gyroscope Y axis data

#### 9.3.2.6 gY\_offset

`int16_t gY_offset`

Gyroscope Y axis offset

#### 9.3.2.7 gZ

`int16_t gZ`

Gyroscope Z axis data

#### 9.3.2.8 gZ\_offset

`int16_t gZ_offset`

Gyroscope Z axis offset

#### 9.3.2.9 hi2c

`I2C_HandleTypeDef* hi2c`

I2C handle

#### 9.3.2.10 status

`HAL_StatusTypeDef status`

HAL status

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

- Ball\_Launcher\_Controller/Core/Inc/ **mpu6050.h**

## 9.4 RadioX Struct Reference

Structure to define a radio driver with timer, channels, and pulse width parameters.

```
#include <radio_driver.h>
```

## Data Fields

- TIM\_HandleTypeDef \* **timer**
- uint32\_t **channel1**
- uint32\_t **channel2**
- uint32\_t **sum1**
- uint32\_t **sum2**
- uint32\_t **counter1**
- uint32\_t **counter2**
- double **pulseWidth**

### 9.4.1 Detailed Description

Structure to define a radio driver with timer, channels, and pulse width parameters.

### 9.4.2 Field Documentation

#### 9.4.2.1 channel1

```
uint32_t channel1
```

Timer channel 1

#### 9.4.2.2 channel2

```
uint32_t channel2
```

Timer channel 2

#### 9.4.2.3 counter1

```
uint32_t counter1
```

Counter for channel 1

#### 9.4.2.4 counter2

```
uint32_t counter2
```

Counter for channel 2

#### 9.4.2.5 pulseWidth

```
double pulseWidth
```

Pulse width

#### 9.4.2.6 sum1

```
uint32_t sum1
```

Sum of pulse widths for channel 1

#### 9.4.2.7 sum2

```
uint32_t sum2
```

Sum of pulse widths for channel 2

#### 9.4.2.8 timer

```
TIM_HandleTypeDef* timer
```

Timer handle for the radio driver

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

- Ball\_Launcher\_Main/Core/Inc/ **radio\_driver.h**

## 9.5 StepperX Struct Reference

Structure to define a stepper motor driver with GPIO ports and pins.

```
#include <stepper_driver.h>
```

### Data Fields

- GPIO\_TypeDef \* **GPIOx**
- uint16\_t **EN\_PIN**
- uint16\_t **DIR\_PIN**
- uint16\_t **STP\_PIN**

### 9.5.1 Detailed Description

Structure to define a stepper motor driver with GPIO ports and pins.

### 9.5.2 Field Documentation

#### 9.5.2.1 DIR\_PIN

```
uint16_t DIR_PIN
```

Direction pin of the stepper motor driver



### 9.5.2.2 EN\_PIN

```
uint16_t EN_PIN
```

Enable pin of the stepper motor driver

### 9.5.2.3 GPIOx

```
GPIO_TypeDef* GPIOx
```

GPIO port of the stepper motor driver

### 9.5.2.4 STP\_PIN

```
uint16_t STP_PIN
```

Step pin of the stepper motor driver

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

- Ball\_Launcher\_Main/Core/Inc/ **stepper\_driver.h**

## 9.6 SwitchX Struct Reference

Structure to define a switch with GPIO port, pin, and status.

```
#include <switch.h>
```

### Data Fields

- GPIO\_TypeDef \* **GPIOx**
- uint16\_t **PIN**
- uint16\_t **status**

### 9.6.1 Detailed Description

Structure to define a switch with GPIO port, pin, and status.

### 9.6.2 Field Documentation

#### 9.6.2.1 GPIOx

```
GPIO_TypeDef* GPIOx
```

GPIO port of the switch

### 9.6.2.2 PIN

`uint16_t PIN`

GPIO pin of the switch

### 9.6.2.3 status

`uint16_t status`

Status of the switch

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

- Ball\_Launcher\_Main/Core/Inc/ **switch.h**

# Chapter 10

## File Documentation

### 10.1 Ball\_Launcher\_Controller/Core/Inc/main\_C.h File Reference

Header for **main\_C.c** (p. 53) file. This file contains the common defines of the application.

```
#include "stm32f4xx_hal.h"
```

#### Functions

- void **Error\_Handler** (void)

*This function is executed in case of error occurrence.*

#### 10.1.1 Detailed Description

Header for **main\_C.c** (p. 53) file. This file contains the common defines of the application.

Attention

Copyright (c) 2024 STMicroelectronics. All rights reserved.

This software is licensed under terms that can be found in the LICENSE file in the root directory of this software component. If no LICENSE file comes with this software, it is provided AS-IS.

#### 10.1.2 Function Documentation

##### 10.1.2.1 Error\_Handler()

```
void Error_Handler (  
    void )
```

This function is executed in case of error occurrence.

## Return values

None	
------	--

## 10.2 Ball\_Launcher\_Controller/Core/Inc/mpu6050.h File Reference

Header for **mpu6050.c** (p. 57) file. This file contains the common defines of the application.

```
#include <stdio.h>
#include <stdint.h>
#include "stm32f4xx_hal.h"
```

### Data Structures

- struct **MPU6050**  
*Struct representing a mpu6050 imu sensor.*

### Functions

- uint16\_t **mpu6050\_init** ( **MPU6050** \*imux, I2C\_HandleTypeDef \*hi2c)  
*Initializes the MPU6050 (p. 27) sensor.*
- void **mpu6050\_calibrate** ( **MPU6050** \*imux)  
*Calibrates the MPU6050 (p. 27) sensor.*
- void **mpu6050\_update** ( **MPU6050** \*imux)  
*Updates the MPU6050 (p. 27) sensor data.*
- int16\_t **mpu6050\_get\_gX** ( **MPU6050** \*imux)  
*Gets the calibrated X-axis gyroscope data.*
- int16\_t **mpu6050\_get\_gY** ( **MPU6050** \*imux)  
*Gets the calibrated Y-axis gyroscope data.*
- int16\_t **mpu6050\_get\_gZ** ( **MPU6050** \*imux)  
*Gets the calibrated Z-axis gyroscope data.*
- int32\_t **mpu6050\_get\_X** ( **MPU6050** \*imux)  
*Gets the X-axis accelerometer data.*
- int32\_t **mpu6050\_get\_Y** ( **MPU6050** \*imux)  
*Gets the Y-axis accelerometer data.*
- int32\_t **mpu6050\_get\_Z** ( **MPU6050** \*imux)  
*Gets the Z-axis accelerometer data.*

### 10.2.1 Detailed Description

Header for **mpu6050.c** (p. 57) file. This file contains the common defines of the application.

### 10.2.2 Function Documentation

#### 10.2.2.1 mpu6050\_calibrate()

```
void mpu6050_calibrate (
    MPU6050 * imux )
```

Calibrates the **MPU6050** (p. 27) sensor.

## Parameters

<i>imux</i>	Pointer to the <b>MPU6050</b> (p. 27) structure.
-------------	--

**10.2.2.2 mpu6050\_get\_gX()**

```
int16_t mpu6050_get_gX (  
    MPU6050 * imux )
```

Gets the calibrated X-axis gyroscope data.

## Parameters

<i>imux</i>	Pointer to the <b>MPU6050</b> (p. 27) structure.
-------------	--

## Returns

int16\_t Calibrated X-axis gyroscope data.

**10.2.2.3 mpu6050\_get\_gY()**

```
int16_t mpu6050_get_gY (  
    MPU6050 * imux )
```

Gets the calibrated Y-axis gyroscope data.

## Parameters

<i>imux</i>	Pointer to the <b>MPU6050</b> (p. 27) structure.
-------------	--

## Returns

int16\_t Calibrated Y-axis gyroscope data.

**10.2.2.4 mpu6050\_get\_gZ()**

```
int16_t mpu6050_get_gZ (  
    MPU6050 * imux )
```

Gets the calibrated Z-axis gyroscope data.

## Parameters

<i>imux</i>	Pointer to the <b>MPU6050</b> (p. 27) structure.
-------------	--

**Returns**

int16\_t Calibrated Z-axis gyroscope data.

**10.2.2.5 mpu6050\_get\_X()**

```
int32_t mpu6050_get_X (
    MPU6050 * imux )
```

Gets the X-axis accelerometer data.

**Parameters**

<i>imux</i>	Pointer to the <b>MPU6050</b> (p. 27) structure.
-------------	--

**Returns**

int32\_t X-axis accelerometer data.

**10.2.2.6 mpu6050\_get\_Y()**

```
int32_t mpu6050_get_Y (
    MPU6050 * imux )
```

Gets the Y-axis accelerometer data.

**Parameters**

<i>imux</i>	Pointer to the <b>MPU6050</b> (p. 27) structure.
-------------	--

**Returns**

int32\_t Y-axis accelerometer data.

**10.2.2.7 mpu6050\_get\_Z()**

```
int32_t mpu6050_get_Z (
    MPU6050 * imux )
```

Gets the Z-axis accelerometer data.

**Parameters**

<i>imux</i>	Pointer to the <b>MPU6050</b> (p. 27) structure.
-------------	--

**Returns**

int32\_t Z-axis accelerometer data.

**10.2.2.8 mpu6050\_init()**

```
uint16_t mpu6050_init (
    MPU6050 * imux,
    I2C_HandleTypeDef * hi2c )
```

Initializes the **MPU6050** (p. 27) sensor.

**Parameters**

<i>imux</i>	Pointer to the <b>MPU6050</b> (p. 27) structure.
<i>hi2c</i>	Pointer to the I2C handle structure.

**Returns**

uint16\_t Returns 1 if initialization is successful, otherwise 0.

**10.2.2.9 mpu6050\_update()**

```
void mpu6050_update (
    MPU6050 * imux )
```

Updates the **MPU6050** (p. 27) sensor data.

**Parameters**

<i>imux</i>	Pointer to the <b>MPU6050</b> (p. 27) structure.
<i>dt</i>	Time interval in milliseconds.

## 10.3 Ball\_Launcher\_Controller/Core/Inc/stm32f4xx\_hal\_conf\_C.h File Reference

HAL configuration template file. This file should be copied to the application folder and renamed to **stm32f4xx\_hal\_conf\_C.h** (p. 37).

```
#include "stm32f4xx_hal_rcc.h"
#include "stm32f4xx_hal_gpio.h"
#include "stm32f4xx_hal_exti.h"
#include "stm32f4xx_hal_dma.h"
#include "stm32f4xx_hal_cortex.h"
#include "stm32f4xx_hal_flash.h"
#include "stm32f4xx_hal_i2c.h"
#include "stm32f4xx_hal_pwr.h"
#include "stm32f4xx_hal_uart.h"
```

## Macros

- **#define HAL\_MODULE\_ENABLED**

*This is the list of modules to be used in the HAL driver.*

- **#define HAL\_I2C\_MODULE\_ENABLED**
- **#define HAL\_UART\_MODULE\_ENABLED**
- **#define HAL\_GPIO\_MODULE\_ENABLED**
- **#define HAL\_EXTI\_MODULE\_ENABLED**
- **#define HAL\_DMA\_MODULE\_ENABLED**
- **#define HAL\_RCC\_MODULE\_ENABLED**
- **#define HAL\_FLASH\_MODULE\_ENABLED**
- **#define HAL\_PWR\_MODULE\_ENABLED**
- **#define HAL\_CORTEX\_MODULE\_ENABLED**
- **#define HSE\_VALUE 25000000U**

*Adjust the value of External High Speed oscillator (HSE) used in your application. This value is used by the RCC HAL module to compute the system frequency (when HSE is used as system clock source, directly or through the PLL).*

- **#define HSE\_STARTUP\_TIMEOUT 100U**
- **#define HSI\_VALUE ((uint32\_t)16000000U)**

*Internal High Speed oscillator (HSI) value. This value is used by the RCC HAL module to compute the system frequency (when HSI is used as system clock source, directly or through the PLL).*

- **#define LSI\_VALUE 32000U**

*Internal Low Speed oscillator (LSI) value.*

- **#define LSE\_VALUE 32768U**

*External Low Speed oscillator (LSE) value.*

- **#define LSE\_STARTUP\_TIMEOUT 5000U**
- **#define EXTERNAL\_CLOCK\_VALUE 12288000U**

*External clock source for I2S peripheral This value is used by the I2S HAL module to compute the I2S clock source frequency, this source is inserted directly through I2S\_CKIN pad.*

- **#define VDD\_VALUE 3300U**

*This is the HAL system configuration section.*

- **#define TICK\_INT\_PRIORITY 15U**
- **#define USE\_RTOS 0U**
- **#define PREFETCH\_ENABLE 1U**
- **#define INSTRUCTION\_CACHE\_ENABLE 1U**
- **#define DATA\_CACHE\_ENABLE 1U**
- **#define USE\_HAL\_ADC\_REGISTER\_CALLBACKS 0U** /\* ADC register callback disabled \*/
- **#define USE\_HAL\_CAN\_REGISTER\_CALLBACKS 0U** /\* CAN register callback disabled \*/
- **#define USE\_HAL\_CEC\_REGISTER\_CALLBACKS 0U** /\* CEC register callback disabled \*/
- **#define USE\_HAL\_Cryp\_REGISTER\_CALLBACKS 0U** /\* CRYP register callback disabled \*/
- **#define USE\_HAL\_DAC\_REGISTER\_CALLBACKS 0U** /\* DAC register callback disabled \*/
- **#define USE\_HAL\_DCMI\_REGISTER\_CALLBACKS 0U** /\* DCMI register callback disabled \*/
- **#define USE\_HAL\_DFSDM\_REGISTER\_CALLBACKS 0U** /\* DFSDM register callback disabled \*/
- **#define USE\_HAL\_DMA2D\_REGISTER\_CALLBACKS 0U** /\* DMA2D register callback disabled \*/
- **#define USE\_HAL\_DSI\_REGISTER\_CALLBACKS 0U** /\* DSI register callback disabled \*/
- **#define USE\_HAL\_ETH\_REGISTER\_CALLBACKS 0U** /\* ETH register callback disabled \*/
- **#define USE\_HAL\_HASH\_REGISTER\_CALLBACKS 0U** /\* HASH register callback disabled \*/
- **#define USE\_HAL\_HCD\_REGISTER\_CALLBACKS 0U** /\* HCD register callback disabled \*/
- **#define USE\_HAL\_I2C\_REGISTER\_CALLBACKS 0U** /\* I2C register callback disabled \*/
- **#define USE\_HAL\_FMPI2C\_REGISTER\_CALLBACKS 0U** /\* FMPI2C register callback disabled \*/
- **#define USE\_HAL\_FMPMBUS\_REGISTER\_CALLBACKS 0U** /\* FMPMBUS register callback disabled \*/
- **#define USE\_HAL\_I2S\_REGISTER\_CALLBACKS 0U** /\* I2S register callback disabled \*/
- **#define USE\_HAL\_IRDA\_REGISTER\_CALLBACKS 0U** /\* IRDA register callback disabled \*/
- **#define USE\_HAL\_LPTIM\_REGISTER\_CALLBACKS 0U** /\* LPTIM register callback disabled \*/



- `#define USE_HAL_LTDC_REGISTER_CALLBACKS 0U /* LTDC register callback disabled */`
- `#define USE_HAL_MMC_REGISTER_CALLBACKS 0U /* MMC register callback disabled */`
- `#define USE_HAL_NAND_REGISTER_CALLBACKS 0U /* NAND register callback disabled */`
- `#define USE_HAL_NOR_REGISTER_CALLBACKS 0U /* NOR register callback disabled */`
- `#define USE_HAL_PCCARD_REGISTER_CALLBACKS 0U /* PCCARD register callback disabled */`
- `#define USE_HAL_PCD_REGISTER_CALLBACKS 0U /* PCD register callback disabled */`
- `#define USE_HAL_QSPI_REGISTER_CALLBACKS 0U /* QSPI register callback disabled */`
- `#define USE_HAL_RNG_REGISTER_CALLBACKS 0U /* RNG register callback disabled */`
- `#define USE_HAL_RTC_REGISTER_CALLBACKS 0U /* RTC register callback disabled */`
- `#define USE_HAL_SAI_REGISTER_CALLBACKS 0U /* SAI register callback disabled */`
- `#define USE_HAL_SD_REGISTER_CALLBACKS 0U /* SD register callback disabled */`
- `#define USE_HAL_SMARTCARD_REGISTER_CALLBACKS 0U /* SMARTCARD register callback disabled */`
- `#define USE_HAL_SDRAM_REGISTER_CALLBACKS 0U /* SDRAM register callback disabled */`
- `#define USE_HAL_SRAM_REGISTER_CALLBACKS 0U /* SRAM register callback disabled */`
- `#define USE_HAL_SPDIFRX_REGISTER_CALLBACKS 0U /* SPDIFRX register callback disabled */`
- `#define USE_HAL_SMBUS_REGISTER_CALLBACKS 0U /* SMBUS register callback disabled */`
- `#define USE_HAL_SPI_REGISTER_CALLBACKS 0U /* SPI register callback disabled */`
- `#define USE_HAL_TIM_REGISTER_CALLBACKS 0U /* TIM register callback disabled */`
- `#define USE_HAL_UART_REGISTER_CALLBACKS 0U /* UART register callback disabled */`
- `#define USE_HAL_USART_REGISTER_CALLBACKS 0U /* USART register callback disabled */`
- `#define USE_HAL_WWDG_REGISTER_CALLBACKS 0U /* WWDG register callback disabled */`
- `#define MAC_ADDR0 2U`

*Uncomment the line below to expanse the "assert\_param" macro in the HAL drivers code.*

- `#define MAC_ADDR1 0U`
- `#define MAC_ADDR2 0U`
- `#define MAC_ADDR3 0U`
- `#define MAC_ADDR4 0U`
- `#define MAC_ADDR5 0U`
- `#define ETH_RX_BUF_SIZE ETH_MAX_PACKET_SIZE /* buffer size for receive */`
- `#define ETH_TX_BUF_SIZE ETH_MAX_PACKET_SIZE /* buffer size for transmit */`
- `#define ETH_RXBUFNB 4U /* 4 Rx buffers of size ETH_RX_BUF_SIZE */`
- `#define ETH_TXBUFNB 4U /* 4 Tx buffers of size ETH_TX_BUF_SIZE */`
- `#define DP83848_PHY_ADDRESS`
- `#define PHY_RESET_DELAY 0x000000FFU`
- `#define PHY_CONFIG_DELAY 0x00000FFFU`
- `#define PHY_READ_TO 0x0000FFFFU`
- `#define PHY_WRITE_TO 0x0000FFFFU`
- `#define PHY_BCR ((uint16_t)0x0000U)`
- `#define PHY_BSR ((uint16_t)0x0001U)`
- `#define PHY_RESET ((uint16_t)0x8000U)`
- `#define PHY_LOOPBACK ((uint16_t)0x4000U)`
- `#define PHY_FULLDUPLEX_100M ((uint16_t)0x2100U)`
- `#define PHY_HALFDUPLEX_100M ((uint16_t)0x2000U)`
- `#define PHY_FULLDUPLEX_10M ((uint16_t)0x0100U)`
- `#define PHY_HALFDUPLEX_10M ((uint16_t)0x0000U)`
- `#define PHY_AUTONEGOTIATION ((uint16_t)0x1000U)`
- `#define PHY_RESTART_AUTONEGOTIATION ((uint16_t)0x0200U)`
- `#define PHY_POWERDOWN ((uint16_t)0x0800U)`
- `#define PHY_ISOLATE ((uint16_t)0x0400U)`
- `#define PHY_AUTONEGO_COMPLETE ((uint16_t)0x0020U)`
- `#define PHY_LINKED_STATUS ((uint16_t)0x0004U)`
- `#define PHY_JABBER_DETECTION ((uint16_t)0x0002U)`
- `#define PHY_SR ((uint16_t))`

- `#define PHY_SPEED_STATUS ((uint16_t))`
- `#define PHY_DUPLEX_STATUS ((uint16_t))`
- `#define USE_SPI_CRC 0U`
- `#define assert_param(expr) ((void)0U)`

*Include module's header file.*

### 10.3.1 Detailed Description

HAL configuration template file. This file should be copied to the application folder and renamed to **stm32f4xx\_hal\_conf\_C.h** (p. 37).

#### Author

MCD Application Team

#### Attention

Copyright (c) 2017 STMicroelectronics. All rights reserved.

This software is licensed under terms that can be found in the LICENSE file in the root directory of this software component. If no LICENSE file comes with this software, it is provided AS-IS.

### 10.3.2 Macro Definition Documentation

#### 10.3.2.1 assert\_param

```
#define assert_param(  
    expr ) ((void)0U)
```

Include module's header file.

#### 10.3.2.2 DATA\_CACHE\_ENABLE

```
#define DATA_CACHE_ENABLE 1U
```

#### 10.3.2.3 DP83848\_PHY\_ADDRESS

```
#define DP83848_PHY_ADDRESS
```

#### 10.3.2.4 ETH\_RX\_BUF\_SIZE

```
#define ETH_RX_BUF_SIZE ETH_MAX_PACKET_SIZE /* buffer size for receive */
```

### 10.3.2.5 ETH\_RXBUFNB

```
#define ETH_RXBUFNB 4U /* 4 Rx buffers of size ETH_RX_BUF_SIZE */
```

### 10.3.2.6 ETH\_TX\_BUF\_SIZE

```
#define ETH_TX_BUF_SIZE ETH_MAX_PACKET_SIZE /* buffer size for transmit */
```

### 10.3.2.7 ETH\_TXBUFNB

```
#define ETH_TXBUFNB 4U /* 4 Tx buffers of size ETH_TX_BUF_SIZE */
```

### 10.3.2.8 EXTERNAL\_CLOCK\_VALUE

```
#define EXTERNAL_CLOCK_VALUE 12288000U
```

External clock source for I2S peripheral This value is used by the I2S HAL module to compute the I2S clock source frequency, this source is inserted directly through I2S\_CKIN pad.

Value of the External audio frequency in Hz

### 10.3.2.9 HAL\_CORTEX\_MODULE\_ENABLED

```
#define HAL_CORTEX_MODULE_ENABLED
```

### 10.3.2.10 HAL\_DMA\_MODULE\_ENABLED

```
#define HAL_DMA_MODULE_ENABLED
```

### 10.3.2.11 HAL\_EXTI\_MODULE\_ENABLED

```
#define HAL_EXTI_MODULE_ENABLED
```

### 10.3.2.12 HAL\_FLASH\_MODULE\_ENABLED

```
#define HAL_FLASH_MODULE_ENABLED
```

### 10.3.2.13 HAL\_GPIO\_MODULE\_ENABLED

```
#define HAL_GPIO_MODULE_ENABLED
```

#### 10.3.2.14 HAL\_I2C\_MODULE\_ENABLED

```
#define HAL_I2C_MODULE_ENABLED
```

#### 10.3.2.15 HAL\_MODULE\_ENABLED

```
#define HAL_MODULE_ENABLED
```

This is the list of modules to be used in the HAL driver.

#### 10.3.2.16 HAL\_PWR\_MODULE\_ENABLED

```
#define HAL_PWR_MODULE_ENABLED
```

#### 10.3.2.17 HAL\_RCC\_MODULE\_ENABLED

```
#define HAL_RCC_MODULE_ENABLED
```

#### 10.3.2.18 HAL\_UART\_MODULE\_ENABLED

```
#define HAL_UART_MODULE_ENABLED
```

#### 10.3.2.19 HSE\_STARTUP\_TIMEOUT

```
#define HSE_STARTUP_TIMEOUT 100U
```

Time out for HSE start up, in ms

#### 10.3.2.20 HSE\_VALUE

```
#define HSE_VALUE 25000000U
```

Adjust the value of External High Speed oscillator (HSE) used in your application. This value is used by the RCC HAL module to compute the system frequency (when HSE is used as system clock source, directly or through the PLL).

Value of the External oscillator in Hz

#### 10.3.2.21 HSI\_VALUE

```
#define HSI_VALUE ((uint32_t)16000000U)
```

Internal High Speed oscillator (HSI) value. This value is used by the RCC HAL module to compute the system frequency (when HSI is used as system clock source, directly or through the PLL).

Value of the Internal oscillator in Hz

### 10.3.2.22 INSTRUCTION\_CACHE\_ENABLE

```
#define INSTRUCTION_CACHE_ENABLE 1U
```

### 10.3.2.23 LSE\_STARTUP\_TIMEOUT

```
#define LSE_STARTUP_TIMEOUT 5000U
```

Time out for LSE start up, in ms

### 10.3.2.24 LSE\_VALUE

```
#define LSE_VALUE 32768U
```

External Low Speed oscillator (LSE) value.

< Value of the Internal Low Speed oscillator in Hz The real value may vary depending on the variations in voltage and temperature. Value of the External Low Speed oscillator in Hz

### 10.3.2.25 LSI\_VALUE

```
#define LSI_VALUE 32000U
```

Internal Low Speed oscillator (LSI) value.

LSI Typical Value in Hz

### 10.3.2.26 MAC\_ADDR0

```
#define MAC_ADDR0 2U
```

Uncomment the line below to expanse the "assert\_param" macro in the HAL drivers code.

### 10.3.2.27 MAC\_ADDR1

```
#define MAC_ADDR1 0U
```

### 10.3.2.28 MAC\_ADDR2

```
#define MAC_ADDR2 0U
```

### 10.3.2.29 MAC\_ADDR3

```
#define MAC_ADDR3 0U
```

#### 10.3.2.30 MAC\_ADDR4

```
#define MAC_ADDR4 0U
```

#### 10.3.2.31 MAC\_ADDR5

```
#define MAC_ADDR5 0U
```

#### 10.3.2.32 PHY\_AUTONEGO\_COMPLETE

```
#define PHY_AUTONEGO_COMPLETE ((uint16_t)0x0020U)
```

Auto-Negotiation process completed

#### 10.3.2.33 PHY\_AUTONEGOTIATION

```
#define PHY_AUTONEGOTIATION ((uint16_t)0x1000U)
```

Enable auto-negotiation function

#### 10.3.2.34 PHY\_BCR

```
#define PHY_BCR ((uint16_t)0x0000U)
```

Transceiver Basic Control Register

#### 10.3.2.35 PHY\_BSR

```
#define PHY_BSR ((uint16_t)0x0001U)
```

Transceiver Basic Status Register

#### 10.3.2.36 PHY\_CONFIG\_DELAY

```
#define PHY_CONFIG_DELAY 0x00000FFFU
```

#### 10.3.2.37 PHY\_DUPLEX\_STATUS

```
#define PHY_DUPLEX_STATUS ((uint16_t))
```

PHY Duplex mask

#### 10.3.2.38 PHY\_FULLDUPLEX\_100M

```
#define PHY_FULLDUPLEX_100M ((uint16_t)0x2100U)
```

Set the full-duplex mode at 100 Mb/s

#### 10.3.2.39 PHY\_FULLDUPLEX\_10M

```
#define PHY_FULLDUPLEX_10M ((uint16_t)0x0100U)
```

Set the full-duplex mode at 10 Mb/s

#### 10.3.2.40 PHY\_HALFDUPLEX\_100M

```
#define PHY_HALFDUPLEX_100M ((uint16_t)0x2000U)
```

Set the half-duplex mode at 100 Mb/s

#### 10.3.2.41 PHY\_HALFDUPLEX\_10M

```
#define PHY_HALFDUPLEX_10M ((uint16_t)0x0000U)
```

Set the half-duplex mode at 10 Mb/s

#### 10.3.2.42 PHY\_ISOLATE

```
#define PHY_ISOLATE ((uint16_t)0x0400U)
```

Isolate PHY from MII

#### 10.3.2.43 PHY\_JABBER\_DETECTION

```
#define PHY_JABBER_DETECTION ((uint16_t)0x0002U)
```

Jabber condition detected

#### 10.3.2.44 PHY\_LINKED\_STATUS

```
#define PHY_LINKED_STATUS ((uint16_t)0x0004U)
```

Valid link established

#### 10.3.2.45 PHY\_LOOPBACK

```
#define PHY_LOOPBACK ((uint16_t)0x4000U)
```

Select loop-back mode

#### 10.3.2.46 PHY\_POWERDOWN

```
#define PHY_POWERDOWN ((uint16_t)0x0800U)
```

Select the power down mode

#### 10.3.2.47 PHY\_READ\_TO

```
#define PHY_READ_TO 0x0000FFFFU
```

#### 10.3.2.48 PHY\_RESET

```
#define PHY_RESET ((uint16_t)0x8000U)
```

PHY Reset

#### 10.3.2.49 PHY\_RESET\_DELAY

```
#define PHY_RESET_DELAY 0x000000FFU
```

#### 10.3.2.50 PHY\_RESTART\_AUTONEGOTIATION

```
#define PHY_RESTART_AUTONEGOTIATION ((uint16_t)0x0200U)
```

Restart auto-negotiation function

#### 10.3.2.51 PHY\_SPEED\_STATUS

```
#define PHY_SPEED_STATUS ((uint16_t))
```

PHY Speed mask

#### 10.3.2.52 PHY\_SR

```
#define PHY_SR ((uint16_t))
```

PHY status register Offset



### 10.3.2.53 PHY\_WRITE\_TO

```
#define PHY_WRITE_TO 0x0000FFFFU
```

### 10.3.2.54 PREFETCH\_ENABLE

```
#define PREFETCH_ENABLE 1U
```

### 10.3.2.55 TICK\_INT\_PRIORITY

```
#define TICK_INT_PRIORITY 15U
```

tick interrupt priority

### 10.3.2.56 USE\_HAL\_ADC\_REGISTER\_CALLBACKS

```
#define USE_HAL_ADC_REGISTER_CALLBACKS 0U /* ADC register callback disabled */
```

### 10.3.2.57 USE\_HAL\_CAN\_REGISTER\_CALLBACKS

```
#define USE_HAL_CAN_REGISTER_CALLBACKS 0U /* CAN register callback disabled */
```

### 10.3.2.58 USE\_HAL\_CEC\_REGISTER\_CALLBACKS

```
#define USE_HAL_CEC_REGISTER_CALLBACKS 0U /* CEC register callback disabled */
```

### 10.3.2.59 USE\_HAL\_CRYPT\_REGISTER\_CALLBACKS

```
#define USE_HAL_CRYPT_REGISTER_CALLBACKS 0U /* CRYPT register callback disabled */
```

### 10.3.2.60 USE\_HAL\_DAC\_REGISTER\_CALLBACKS

```
#define USE_HAL_DAC_REGISTER_CALLBACKS 0U /* DAC register callback disabled */
```

### 10.3.2.61 USE\_HAL\_DCMI\_REGISTER\_CALLBACKS

```
#define USE_HAL_DCMI_REGISTER_CALLBACKS 0U /* DCMI register callback disabled */
```

### 10.3.2.62 USE\_HAL\_DFSDM\_REGISTER\_CALLBACKS

```
#define USE_HAL_DFSDM_REGISTER_CALLBACKS 0U /* DFSDM register callback disabled */
```

### 10.3.2.63 USE\_HAL\_DMA2D\_REGISTER\_CALLBACKS

```
#define USE_HAL_DMA2D_REGISTER_CALLBACKS 0U /* DMA2D register callback disabled */
```

### 10.3.2.64 USE\_HAL\_DSI\_REGISTER\_CALLBACKS

```
#define USE_HAL_DSI_REGISTER_CALLBACKS 0U /* DSI register callback disabled */
```

### 10.3.2.65 USE\_HAL\_ETH\_REGISTER\_CALLBACKS

```
#define USE_HAL_ETH_REGISTER_CALLBACKS 0U /* ETH register callback disabled */
```

### 10.3.2.66 USE\_HAL\_FMPI2C\_REGISTER\_CALLBACKS

```
#define USE_HAL_FMPI2C_REGISTER_CALLBACKS 0U /* FMPI2C register callback disabled */
```

### 10.3.2.67 USE\_HAL\_FMPMBUS\_REGISTER\_CALLBACKS

```
#define USE_HAL_FMPMBUS_REGISTER_CALLBACKS 0U /* FMPMBUS register callback disabled */
```

### 10.3.2.68 USE\_HAL\_HASH\_REGISTER\_CALLBACKS

```
#define USE_HAL_HASH_REGISTER_CALLBACKS 0U /* HASH register callback disabled */
```

### 10.3.2.69 USE\_HAL\_HCD\_REGISTER\_CALLBACKS

```
#define USE_HAL_HCD_REGISTER_CALLBACKS 0U /* HCD register callback disabled */
```

### 10.3.2.70 USE\_HAL\_I2C\_REGISTER\_CALLBACKS

```
#define USE_HAL_I2C_REGISTER_CALLBACKS 0U /* I2C register callback disabled */
```

### 10.3.2.71 USE\_HAL\_I2S\_REGISTER\_CALLBACKS

```
#define USE_HAL_I2S_REGISTER_CALLBACKS 0U /* I2S register callback disabled */
```

### 10.3.2.72 USE\_HAL\_IRDA\_REGISTER\_CALLBACKS

```
#define USE_HAL_IRDA_REGISTER_CALLBACKS 0U /* IRDA register callback disabled */
```

### 10.3.2.73 USE\_HAL\_LPTIM\_REGISTER\_CALLBACKS

```
#define USE_HAL_LPTIM_REGISTER_CALLBACKS 0U /* LPTIM register callback disabled */
```

### 10.3.2.74 USE\_HAL\_LTDC\_REGISTER\_CALLBACKS

```
#define USE_HAL_LTDC_REGISTER_CALLBACKS 0U /* LTDC register callback disabled */
```

### 10.3.2.75 USE\_HAL\_MMC\_REGISTER\_CALLBACKS

```
#define USE_HAL_MMC_REGISTER_CALLBACKS 0U /* MMC register callback disabled */
```

### 10.3.2.76 USE\_HAL\_NAND\_REGISTER\_CALLBACKS

```
#define USE_HAL_NAND_REGISTER_CALLBACKS 0U /* NAND register callback disabled */
```

### 10.3.2.77 USE\_HAL\_NOR\_REGISTER\_CALLBACKS

```
#define USE_HAL_NOR_REGISTER_CALLBACKS 0U /* NOR register callback disabled */
```

### 10.3.2.78 USE\_HAL\_PCCARD\_REGISTER\_CALLBACKS

```
#define USE_HAL_PCCARD_REGISTER_CALLBACKS 0U /* PCCARD register callback disabled */
```

### 10.3.2.79 USE\_HAL\_PCD\_REGISTER\_CALLBACKS

```
#define USE_HAL_PCD_REGISTER_CALLBACKS 0U /* PCD register callback disabled */
```

### 10.3.2.80 USE\_HAL\_QSPI\_REGISTER\_CALLBACKS

```
#define USE_HAL_QSPI_REGISTER_CALLBACKS 0U /* QSPI register callback disabled */
```

### 10.3.2.81 USE\_HAL\_RNG\_REGISTER\_CALLBACKS

```
#define USE_HAL_RNG_REGISTER_CALLBACKS 0U /* RNG register callback disabled */
```

### 10.3.2.82 USE\_HAL\_RTC\_REGISTER\_CALLBACKS

```
#define USE_HAL_RTC_REGISTER_CALLBACKS 0U /* RTC register callback disabled */
```

### 10.3.2.83 USE\_HAL\_SAI\_REGISTER\_CALLBACKS

```
#define USE_HAL_SAI_REGISTER_CALLBACKS 0U /* SAI register callback disabled */
```

### 10.3.2.84 USE\_HAL\_SD\_REGISTER\_CALLBACKS

```
#define USE_HAL_SD_REGISTER_CALLBACKS 0U /* SD register callback disabled */
```

### 10.3.2.85 USE\_HAL\_SDRAM\_REGISTER\_CALLBACKS

```
#define USE_HAL_SDRAM_REGISTER_CALLBACKS 0U /* SDRAM register callback disabled */
```

### 10.3.2.86 USE\_HAL\_SMARTCARD\_REGISTER\_CALLBACKS

```
#define USE_HAL_SMARTCARD_REGISTER_CALLBACKS 0U /* SMARTCARD register callback disabled */
```

### 10.3.2.87 USE\_HAL\_SMBUS\_REGISTER\_CALLBACKS

```
#define USE_HAL_SMBUS_REGISTER_CALLBACKS 0U /* SMBUS register callback disabled */
```

### 10.3.2.88 USE\_HAL\_SPDIFRX\_REGISTER\_CALLBACKS

```
#define USE_HAL_SPDIFRX_REGISTER_CALLBACKS 0U /* SPDIFRX register callback disabled */
```

### 10.3.2.89 USE\_HAL\_SPI\_REGISTER\_CALLBACKS

```
#define USE_HAL_SPI_REGISTER_CALLBACKS 0U /* SPI register callback disabled */
```

### 10.3.2.90 USE\_HAL\_SRAM\_REGISTER\_CALLBACKS

```
#define USE_HAL_SRAM_REGISTER_CALLBACKS 0U /* SRAM register callback disabled */
```

### 10.3.2.91 USE\_HAL\_TIM\_REGISTER\_CALLBACKS

```
#define USE_HAL_TIM_REGISTER_CALLBACKS 0U /* TIM register callback disabled */
```

### 10.3.2.92 USE\_HAL\_UART\_REGISTER\_CALLBACKS

```
#define USE_HAL_UART_REGISTER_CALLBACKS 0U /* UART register callback disabled */
```

**10.3.2.93 USE\_HAL\_USART\_REGISTER\_CALLBACKS**

```
#define USE_HAL_USART_REGISTER_CALLBACKS 0U /* USART register callback disabled */
```

**10.3.2.94 USE\_HAL\_WWDG\_REGISTER\_CALLBACKS**

```
#define USE_HAL_WWDG_REGISTER_CALLBACKS 0U /* WWDG register callback disabled */
```

**10.3.2.95 USE\_RTOS**

```
#define USE_RTOS 0U
```

**10.3.2.96 USE\_SPI\_CRC**

```
#define USE_SPI_CRC 0U
```

**10.3.2.97 VDD\_VALUE**

```
#define VDD_VALUE 3300U
```

This is the HAL system configuration section.

Value of VDD in mv

**10.4 Ball\_Launcher\_Controller/Core/Inc/stm32f4xx\_it\_C.h File Reference**

This file contains the headers of the interrupt handlers.

**Functions**

- void **NMI\_Handler** (void)  
*This function handles Non maskable interrupt.*
- void **HardFault\_Handler** (void)  
*This function handles Hard fault interrupt.*
- void **MemManage\_Handler** (void)  
*This function handles Memory management fault.*
- void **BusFault\_Handler** (void)  
*This function handles Pre-fetch fault, memory access fault.*
- void **UsageFault\_Handler** (void)  
*This function handles Undefined instruction or illegal state.*
- void **SVC\_Handler** (void)  
*This function handles System service call via SWI instruction.*
- void **DebugMon\_Handler** (void)  
*This function handles Debug monitor.*
- void **PendSV\_Handler** (void)  
*This function handles Pendable request for system service.*
- void **SysTick\_Handler** (void)  
*This function handles System tick timer.*

### 10.4.1 Detailed Description

This file contains the headers of the interrupt handlers.

#### Attention

Copyright (c) 2024 STMicroelectronics. All rights reserved.

This software is licensed under terms that can be found in the LICENSE file in the root directory of this software component. If no LICENSE file comes with this software, it is provided AS-IS.

### 10.4.2 Function Documentation

#### 10.4.2.1 BusFault\_Handler()

```
void BusFault_Handler (
    void )
```

This function handles Pre-fetch fault, memory access fault.

#### 10.4.2.2 DebugMon\_Handler()

```
void DebugMon_Handler (
    void )
```

This function handles Debug monitor.

#### 10.4.2.3 HardFault\_Handler()

```
void HardFault_Handler (
    void )
```

This function handles Hard fault interrupt.

#### 10.4.2.4 MemManage\_Handler()

```
void MemManage_Handler (
    void )
```

This function handles Memory management fault.

#### 10.4.2.5 NMI\_Handler()

```
void NMI_Handler (
    void )
```

This function handles Non maskable interrupt.

#### 10.4.2.6 PendSV\_Handler()

```
void PendSV_Handler (  
    void )
```

This function handles Pendable request for system service.

#### 10.4.2.7 SVC\_Handler()

```
void SVC_Handler (  
    void )
```

This function handles System service call via SWI instruction.

#### 10.4.2.8 SysTick\_Handler()

```
void SysTick_Handler (  
    void )
```

This function handles System tick timer.

#### 10.4.2.9 UsageFault\_Handler()

```
void UsageFault_Handler (  
    void )
```

This function handles Undefined instruction or illegal state.

## 10.5 Ball\_Launcher\_Controller/Core/Src/main\_C.c File Reference

Main program body.

```
#include "main.h"  
#include "mpu6050.h"  
#include <stdio.h>  
#include <string.h>
```

### Functions

- void **SystemClock\_Config** (void)  
*System Clock Configuration.*
- int **main** (void)  
*The application entry point.*
- void **Error\_Handler** (void)  
*This function is executed in case of error occurrence.*

## Variables

- I2C\_HandleTypeDef **hi2c1**
- UART\_HandleTypeDef **huart1**
- UART\_HandleTypeDef **huart2**
- int16\_t **gX** = 0
- int16\_t **gY** = 0
- int16\_t **gZ** = 0
- uint16\_t **stat** = 0
- uint16\_t **shot** = 0
- uint16\_t **move** = 0
- uint16\_t **STATE** = 0
- uint16\_t **STATE\_0\_INIT** = 0
- uint16\_t **STATE\_1\_ERROR** = 1
- uint16\_t **STATE\_2\_IMU** = 2
- uint16\_t **STATE\_3\_BUTTON\_LED** = 3
- uint16\_t **STATE\_4\_TRANSFER** = 4
- char **buffer** [100]

### 10.5.1 Detailed Description

Main program body.

#### Attention

Copyright (c) 2024 STMicroelectronics. All rights reserved.

This software is licensed under terms that can be found in the LICENSE file in the root directory of this software component. If no LICENSE file comes with this software, it is provided AS-IS.

### 10.5.2 Function Documentation

#### 10.5.2.1 Error\_Handler()

```
void Error_Handler (
    void )
```

This function is executed in case of error occurrence.

#### Return values

None	
------	--

#### 10.5.2.2 main()

```
int main (
    void )
```



The application entry point.

Return values

<i>int</i>	
------------	--

### 10.5.2.3 SystemClock\_Config()

```
void SystemClock_Config (  
    void )
```

System Clock Configuration.

Return values

<i>None</i>	
-------------	--

Configure the main internal regulator output voltage

Initializes the RCC Oscillators according to the specified parameters in the RCC\_OscInitTypeDef structure.

Initializes the CPU, AHB and APB buses clocks

## 10.5.3 Variable Documentation

### 10.5.3.1 buffer

```
char buffer[100]
```

### 10.5.3.2 gX

```
int16_t gX = 0
```

### 10.5.3.3 gY

```
int16_t gY = 0
```

### 10.5.3.4 gZ

```
int16_t gZ = 0
```

### 10.5.3.5 hi2c1

```
I2C_HandleTypeDef hi2c1
```

#### 10.5.3.6 huart1

```
UART_HandleTypeDef huart1
```

#### 10.5.3.7 huart2

```
UART_HandleTypeDef huart2
```

#### 10.5.3.8 move

```
uint16_t move = 0
```

#### 10.5.3.9 shot

```
uint16_t shot = 0
```

#### 10.5.3.10 stat

```
uint16_t stat = 0
```

#### 10.5.3.11 STATE

```
uint16_t STATE = 0
```

#### 10.5.3.12 STATE\_0\_INIT

```
uint16_t STATE_0_INIT = 0
```

#### 10.5.3.13 STATE\_1\_ERROR

```
uint16_t STATE_1_ERROR = 1
```

#### 10.5.3.14 STATE\_2\_IMU

```
uint16_t STATE_2_IMU = 2
```

#### 10.5.3.15 STATE\_3\_BUTTON\_LED

```
uint16_t STATE_3_BUTTON_LED = 3
```

### 10.5.3.16 STATE\_4\_TRANSFER

```
uint16_t STATE_4_TRANSFER = 4
```

## 10.6 Ball\_Launcher\_Controller/Core/Src/mpu6050.c File Reference

Implementation of mpu6050 sensor functions.

```
#include "mpu6050.h"
```

### Functions

- `uint16_t mpu6050_init ( MPU6050 *imux, I2C_HandleTypeDef *hi2c)`  
*Initializes the MPU6050 (p. 27) sensor.*
- `void mpu6050_calibrate ( MPU6050 *imux)`  
*Calibrates the MPU6050 (p. 27) sensor.*
- `void mpu6050_update ( MPU6050 *imux)`  
*Updates the MPU6050 (p. 27) sensor data.*
- `int16_t mpu6050_get_gX ( MPU6050 *imux)`  
*Gets the calibrated X-axis gyroscope data.*
- `int16_t mpu6050_get_gY ( MPU6050 *imux)`  
*Gets the calibrated Y-axis gyroscope data.*
- `int16_t mpu6050_get_gZ ( MPU6050 *imux)`  
*Gets the calibrated Z-axis gyroscope data.*

### 10.6.1 Detailed Description

Implementation of mpu6050 sensor functions.

Created on: May 3, 2024 Author: vvinh

### 10.6.2 Function Documentation

#### 10.6.2.1 mpu6050\_calibrate()

```
void mpu6050_calibrate (
    MPU6050 * imux )
```

Calibrates the **MPU6050** (p. 27) sensor.

#### Parameters

<i>imux</i>	Pointer to the <b>MPU6050</b> (p. 27) structure.
-------------	--

### 10.6.2.2 mpu6050\_get\_gX()

```
int16_t mpu6050_get_gX (
    MPU6050 * imux )
```

Gets the calibrated X-axis gyroscope data.

#### Parameters

<i>imux</i>	Pointer to the <b>MPU6050</b> (p. 27) structure.
-------------	--

#### Returns

int16\_t Calibrated X-axis gyroscope data.

### 10.6.2.3 mpu6050\_get\_gY()

```
int16_t mpu6050_get_gY (
    MPU6050 * imux )
```

Gets the calibrated Y-axis gyroscope data.

#### Parameters

<i>imux</i>	Pointer to the <b>MPU6050</b> (p. 27) structure.
-------------	--

#### Returns

int16\_t Calibrated Y-axis gyroscope data.

### 10.6.2.4 mpu6050\_get\_gZ()

```
int16_t mpu6050_get_gZ (
    MPU6050 * imux )
```

Gets the calibrated Z-axis gyroscope data.

#### Parameters

<i>imux</i>	Pointer to the <b>MPU6050</b> (p. 27) structure.
-------------	--

#### Returns

int16\_t Calibrated Z-axis gyroscope data.

### 10.6.2.5 mpu6050\_init()

```
uint16_t mpu6050_init (
```

```

    MPU6050 * imux,
    I2C_HandleTypeDef * hi2c )

```

Initializes the **MPU6050** (p. 27) sensor.

#### Parameters

<i>imux</i>	Pointer to the <b>MPU6050</b> (p. 27) structure.
<i>hi2c</i>	Pointer to the I2C handle structure.

#### Returns

uint16\_t Returns 1 if initialization is successful, otherwise 0.

#### 10.6.2.6 mpu6050\_update()

```

void mpu6050_update (
    MPU6050 * imux )

```

Updates the **MPU6050** (p. 27) sensor data.

#### Parameters

<i>imux</i>	Pointer to the <b>MPU6050</b> (p. 27) structure.
<i>dt</i>	Time interval in milliseconds.

## 10.7 Ball\_Launcher\_Controller/Core/Src/stm32f4xx\_hal\_msp\_C.c File Reference

This file provides code for the MSP Initialization and de-Initialization codes.

```
#include "main.h"
```

#### Functions

- void **HAL\_MspInit** (void)
- void **HAL\_I2C\_MspInit** (I2C\_HandleTypeDef \*hi2c)  
*I2C MSP Initialization This function configures the hardware resources used in this example.*
- void **HAL\_I2C\_MspDeInit** (I2C\_HandleTypeDef \*hi2c)  
*I2C MSP De-Initialization This function freeze the hardware resources used in this example.*
- void **HAL\_UART\_MspInit** (UART\_HandleTypeDef \*huart)  
*UART MSP Initialization This function configures the hardware resources used in this example.*
- void **HAL\_UART\_MspDeInit** (UART\_HandleTypeDef \*huart)  
*UART MSP De-Initialization This function freeze the hardware resources used in this example.*

### 10.7.1 Detailed Description

This file provides code for the MSP Initialization and de-Initialization codes.

#### Attention

Copyright (c) 2024 STMicroelectronics. All rights reserved.

This software is licensed under terms that can be found in the LICENSE file in the root directory of this software component. If no LICENSE file comes with this software, it is provided AS-IS.

### 10.7.2 Function Documentation

#### 10.7.2.1 HAL\_I2C\_MspDeInit()

```
void HAL_I2C_MspDeInit (
    I2C_HandleTypeDef * hi2c )
```

I2C MSP De-Initialization This function freeze the hardware resources used in this example.

##### Parameters

<i>hi2c</i>	I2C handle pointer
-------------	--------------------

##### Return values

<i>None</i>	
-------------	--

I2C1 GPIO Configuration PB6 -----> I2C1\_SCL PB7 -----> I2C1\_SDA

#### 10.7.2.2 HAL\_I2C\_MspInit()

```
void HAL_I2C_MspInit (
    I2C_HandleTypeDef * hi2c )
```

I2C MSP Initialization This function configures the hardware resources used in this example.

##### Parameters

<i>hi2c</i>	I2C handle pointer
-------------	--------------------

##### Return values

<i>None</i>	
-------------	--

I2C1 GPIO Configuration PB6 -----> I2C1\_SCL PB7 -----> I2C1\_SDA

### 10.7.2.3 HAL\_MspInit()

```
void HAL_MspInit (
    void )
```

Initializes the Global MSP.

### 10.7.2.4 HAL\_UART\_MspDeInit()

```
void HAL_UART_MspDeInit (
    UART_HandleTypeDef * huart )
```

UART MSP De-Initialization This function freeze the hardware resources used in this example.

#### Parameters

<i>huart</i>	UART handle pointer
--------------	---------------------

#### Return values

<i>None</i>	
-------------	--

USART1 GPIO Configuration PA9 -----> USART1\_TX PA10 -----> USART1\_RX

USART2 GPIO Configuration PA2 -----> USART2\_TX PA3 -----> USART2\_RX

### 10.7.2.5 HAL\_UART\_MspInit()

```
void HAL_UART_MspInit (
    UART_HandleTypeDef * huart )
```

UART MSP Initialization This function configures the hardware resources used in this example.

#### Parameters

<i>huart</i>	UART handle pointer
--------------	---------------------

#### Return values

<i>None</i>	
-------------	--

USART1 GPIO Configuration PA9 -----> USART1\_TX PA10 -----> USART1\_RX

USART2 GPIO Configuration PA2 -----> USART2\_TX PA3 -----> USART2\_RX

## 10.8 Ball\_Launcher\_Controller/Core/Src/stm32f4xx\_it\_C.c File Reference

Interrupt Service Routines.

```
#include "main.h"
#include "stm32f4xx_it.h"
```

### Functions

- void **NMI\_Handler** (void)  
*This function handles Non maskable interrupt.*
- void **HardFault\_Handler** (void)  
*This function handles Hard fault interrupt.*
- void **MemManage\_Handler** (void)  
*This function handles Memory management fault.*
- void **BusFault\_Handler** (void)  
*This function handles Pre-fetch fault, memory access fault.*
- void **UsageFault\_Handler** (void)  
*This function handles Undefined instruction or illegal state.*
- void **SVC\_Handler** (void)  
*This function handles System service call via SWI instruction.*
- void **DebugMon\_Handler** (void)  
*This function handles Debug monitor.*
- void **PendSV\_Handler** (void)  
*This function handles Pendable request for system service.*
- void **SysTick\_Handler** (void)  
*This function handles System tick timer.*

### 10.8.1 Detailed Description

Interrupt Service Routines.

Attention

Copyright (c) 2024 STMicroelectronics. All rights reserved.

This software is licensed under terms that can be found in the LICENSE file in the root directory of this software component. If no LICENSE file comes with this software, it is provided AS-IS.

### 10.8.2 Function Documentation

#### 10.8.2.1 BusFault\_Handler()

```
void BusFault_Handler (
    void )
```

This function handles Pre-fetch fault, memory access fault.



### 10.8.2.2 DebugMon\_Handler()

```
void DebugMon_Handler (  
    void )
```

This function handles Debug monitor.

### 10.8.2.3 HardFault\_Handler()

```
void HardFault_Handler (  
    void )
```

This function handles Hard fault interrupt.

### 10.8.2.4 MemManage\_Handler()

```
void MemManage_Handler (  
    void )
```

This function handles Memory management fault.

### 10.8.2.5 NMI\_Handler()

```
void NMI_Handler (  
    void )
```

This function handles Non maskable interrupt.

### 10.8.2.6 PendSV\_Handler()

```
void PendSV_Handler (  
    void )
```

This function handles Pendable request for system service.

### 10.8.2.7 SVC\_Handler()

```
void SVC_Handler (  
    void )
```

This function handles System service call via SWI instruction.

### 10.8.2.8 SysTick\_Handler()

```
void SysTick_Handler (  
    void )
```

This function handles System tick timer.

### 10.8.2.9 UsageFault\_Handler()

```
void UsageFault_Handler (
    void )
```

This function handles Undefined instruction or illegal state.

## 10.9 Ball\_Launcher\_Controller/Core/Src/syscalls\_C.c File Reference

STM32CubeIDE Minimal System calls file.

```
#include <sys/stat.h>
#include <stdlib.h>
#include <errno.h>
#include <stdio.h>
#include <signal.h>
#include <time.h>
#include <sys/time.h>
#include <sys/times.h>
```

### Functions

- int **\_\_io\_putchar** (int ch) **\_\_attribute\_\_((weak))**
- int **\_\_io\_getchar** (void)
- void **initialise\_monitor\_handles** ()
- int **\_getpid** (void)
- int **\_kill** (int pid, int sig)
- void **\_exit** (int status)
- **\_\_attribute\_\_((weak))**
- int **\_close** (int file)
- int **\_fstat** (int file, struct **stat** \*st)
- int **\_isatty** (int file)
- int **\_lseek** (int file, int ptr, int dir)
- int **\_open** (char \*path, int flags,...)
- int **\_wait** (int \*status)
- int **\_unlink** (char \*name)
- int **\_times** (struct tms \*buf)
- int **\_stat** (char \*file, struct **stat** \*st)
- int **\_link** (char \*old, char \*new)
- int **\_fork** (void)
- int **\_execve** (char \*name, char \*\*argv, char \*\*env)

### Variables

- char \*\* **environ** = \_\_env

## 10.9.1 Detailed Description

STM32CubeIDE Minimal System calls file.

### Author

Auto-generated by STM32CubeIDE

```
For more information about which c-functions
need which of these lowlevel functions
please consult the Newlib libc-manual
```

### Attention

Copyright (c) 2020-2024 STMicroelectronics. All rights reserved.

This software is licensed under terms that can be found in the LICENSE file in the root directory of this software component. If no LICENSE file comes with this software, it is provided AS-IS.

## 10.9.2 Function Documentation

### 10.9.2.1 \_\_attribute\_\_()

```
__attribute__ (
    (weak) )
```

### 10.9.2.2 \_\_io\_getchar()

```
int __io_getchar (
    void ) [extern]
```

### 10.9.2.3 \_\_io\_putchar()

```
int __io_putchar (
    int ch ) [extern]
```

### 10.9.2.4 \_close()

```
int _close (
    int file )
```

### 10.9.2.5 \_execve()

```
int _execve (
    char * name,
    char ** argv,
    char ** env )
```

#### 10.9.2.6 `_exit()`

```
void _exit (
    int status )
```

#### 10.9.2.7 `_fork()`

```
int _fork (
    void )
```

#### 10.9.2.8 `_fstat()`

```
int _fstat (
    int file,
    struct stat * st )
```

#### 10.9.2.9 `_getpid()`

```
int _getpid (
    void )
```

#### 10.9.2.10 `_isatty()`

```
int _isatty (
    int file )
```

#### 10.9.2.11 `_kill()`

```
int _kill (
    int pid,
    int sig )
```

#### 10.9.2.12 `_link()`

```
int _link (
    char * old,
    char * new )
```

#### 10.9.2.13 `_lseek()`

```
int _lseek (
    int file,
    int ptr,
    int dir )
```

#### 10.9.2.14 `_open()`

```
int _open (
    char * path,
    int flags,
    ... )
```

#### 10.9.2.15 `_stat()`

```
int _stat (
    char * file,
    struct stat * st )
```

#### 10.9.2.16 `_times()`

```
int _times (
    struct tms * buf )
```

#### 10.9.2.17 `_unlink()`

```
int _unlink (
    char * name )
```

#### 10.9.2.18 `_wait()`

```
int _wait (
    int * status )
```

#### 10.9.2.19 `initialise_monitor_handles()`

```
void initialise_monitor_handles ( )
```

### 10.9.3 Variable Documentation

#### 10.9.3.1 `environ`

```
char** environ = __env
```

## 10.10 Ball\_Launcher\_Controller/Core/Src/systemem\_C.c File Reference

STM32CubeIDE System Memory calls file.

```
#include <errno.h>
#include <stdint.h>
```

## Functions

- void \* **\_sbrk** (ptrdiff\_t incr)  
*\_sbrk()* (p. 68) allocates memory to the newlib heap and is used by malloc and others from the C library

### 10.10.1 Detailed Description

STM32CubeIDE System Memory calls file.

#### Author

Generated by STM32CubeIDE

For more information about which C functions  
need which of these lowlevel functions  
please consult the newlib libc manual

#### Attention

Copyright (c) 2024 STMicroelectronics. All rights reserved.

This software is licensed under terms that can be found in the LICENSE file in the root directory of this software component. If no LICENSE file comes with this software, it is provided AS-IS.

### 10.10.2 Function Documentation

#### 10.10.2.1 \_sbrk()

```
void * _sbrk (  
    ptrdiff_t incr )
```

**\_sbrk()** (p. 68) allocates memory to the newlib heap and is used by malloc and others from the C library

```
* #####  
* # .data # .bss #          newlib heap          #          MSP stack          #  
* #          #          #          #          # Reserved by _Min_Stack_Size #  
* #####  
* ^-- RAM start          ^-- _end          _estack, RAM end --^  
*
```

This implementation starts allocating at the '\_end' linker symbol The '\_Min\_Stack\_Size' linker symbol reserves a memory for the MSP stack The implementation considers '\_estack' linker symbol to be RAM end NOTE: If the MSP stack, at any point during execution, grows larger than the reserved size, please increase the '\_Min\_Stack\_Size'.

#### Parameters

<i>incr</i>	Memory size
-------------	-------------

**Returns**

Pointer to allocated memory

## 10.11 Ball\_Launcher\_Controller/Core/Src/system\_stm32f4xx\_C.c File Reference

CMSIS Cortex-M4 Device Peripheral Access Layer System Source File.

```
#include "stm32f4xx.h"
```

**Macros**

- **#define HSE\_VALUE** ((uint32\_t)25000000)
- **#define HSI\_VALUE** ((uint32\_t)16000000)

**Functions**

- void **SystemInit** (void)  
*Setup the microcontroller system Initialize the FPU setting, vector table location and External memory configuration.*
- void **SystemCoreClockUpdate** (void)  
*Update SystemCoreClock variable according to Clock Register Values. The SystemCoreClock variable contains the core clock (HCLK), it can be used by the user application to setup the SysTick timer or configure other parameters.*

**Variables**

- uint32\_t **SystemCoreClock** = 16000000
- const uint8\_t **AHBPrescTable** [16] = {0, 0, 0, 0, 0, 0, 0, 0, 1, 2, 3, 4, 6, 7, 8, 9}
- const uint8\_t **APBPrescTable** [8] = {0, 0, 0, 0, 1, 2, 3, 4}

### 10.11.1 Detailed Description

CMSIS Cortex-M4 Device Peripheral Access Layer System Source File.

**Author**

MCD Application Team

This file provides two functions and one global variable to be called from user application:

- **SystemInit()** (p. 22): This function is called at startup just after reset and before branch to main program. This call is made inside the "startup\_stm32f4xx.s" file.
- **SystemCoreClock** variable: Contains the core clock (HCLK), it can be used by the user application to setup the SysTick timer or configure other parameters.
- **SystemCoreClockUpdate()** (p. 22): Updates the variable SystemCoreClock and must be called whenever the core clock is changed during program execution.

**Attention**

Copyright (c) 2017 STMicroelectronics. All rights reserved.

This software is licensed under terms that can be found in the LICENSE file in the root directory of this software component. If no LICENSE file comes with this software, it is provided AS-IS.

## 10.12 Ball\_Launcher\_Main/Core/Inc/d4215.h File Reference

Header for **d4215.c** (p. 96) file. This file contains the common defines of the application.

```
#include <stdio.h>
#include <stdint.h>
#include "stm32f4xx_hal.h"
```

### Data Structures

- struct **D4215X**  
*Structure to define a D4215 motor with timer, channel, and speed.*

### Functions

- void **D4215\_init** ( **D4215X** \*bldcx, TIM\_HandleTypeDef \*timer, uint32\_t channel)  
*Initializes the D4215 motor with the specified timer and channel.*
- void **D4215\_set** ( **D4215X** \*bldcx, int32\_t speed)  
*Sets the speed of the D4215 motor.*

### 10.12.1 Detailed Description

Header for **d4215.c** (p. 96) file. This file contains the common defines of the application.

Created on: May 1, 2024

Author: vvinh

### 10.12.2 Function Documentation

#### 10.12.2.1 D4215\_init()

```
void D4215_init (
    D4215X * bldcx,
    TIM_HandleTypeDef * timer,
    uint32_t channel )
```

Initializes the D4215 motor with the specified timer and channel.

#### Parameters

<i>bldcx</i>	Pointer to the <b>D4215X</b> (p. 25) structure
<i>timer</i>	Timer handle
<i>channel</i>	Timer channel

< Start PWM for the specified timer and channel



### 10.12.2.2 D4215\_set()

```
void D4215_set (
    D4215X * bldcx,
    int32_t spd )
```

Sets the speed of the D4215 motor.

#### Parameters

<i>bldcx</i>	Pointer to the <b>D4215X</b> (p. 25) structure
<i>speed</i>	Speed of the motor (range: 0 to 100)
<i>bldcx</i>	Pointer to the <b>D4215X</b> (p. 25) structure
<i>spd</i>	Speed of the motor (range: 5 to 10)

< Calculate the duty cycle based on the input speed

< Set the duty cycle for the PWM

## 10.13 Ball\_Launcher\_Main/Core/Inc/led.h File Reference

Header for **led.c** (p. 97) file. This file contains the common defines of the application.

```
#include <stdio.h>
#include <stdint.h>
#include "stm32f4xx_hal.h"
```

### Data Structures

- struct **LEDX**  
*Structure to define an LED with GPIO port and pin.*

### Functions

- void **LED\_init** ( **LEDX** \*LEDx, GPIO\_TypeDef \*GPIO, uint16\_t PIN)  
*Initializes the LED with the specified GPIO port and pin.*
- void **LED\_on** ( **LEDX** \*LEDx)  
*Turns on the LED.*
- void **LED\_off** ( **LEDX** \*LEDx)  
*Turns off the LED.*
- void **LED\_toggle** ( **LEDX** \*LEDx)  
*Toggles the LED state.*

### 10.13.1 Detailed Description

Header for **led.c** (p. 97) file. This file contains the common defines of the application.

Created on: May 17, 2024

Author: vvinh

## 10.13.2 Function Documentation

### 10.13.2.1 LED\_init()

```
void LED_init (
    LEDX * LEDx,
    GPIO_TypeDef * GPIO,
    uint16_t PIN )
```

Initializes the LED with the specified GPIO port and pin.

#### Parameters

<i>LEDx</i>	Pointer to the <b>LEDX</b> (p. 26) structure
<i>GPIO</i>	GPIO port
<i>PIN</i>	GPIO pin

< Assign the GPIO port

< Assign the GPIO pin

### 10.13.2.2 LED\_off()

```
void LED_off (
    LEDX * LEDx )
```

Turns off the LED.

#### Parameters

<i>LEDx</i>	Pointer to the <b>LEDX</b> (p. 26) structure
-------------	--

< Set the pin low to turn off the LED

### 10.13.2.3 LED\_on()

```
void LED_on (
    LEDX * LEDx )
```

Turns on the LED.

#### Parameters

<i>LEDx</i>	Pointer to the <b>LEDX</b> (p. 26) structure
-------------	--

< Set the pin high to turn on the LED

### 10.13.2.4 LED\_toggle()

```
void LED_toggle (
    LEDX * LEDx )
```

Toggles the LED state.

#### Parameters

<i>LEDx</i>	Pointer to the <b>LEDX</b> (p. 26) structure
-------------	--

< Toggle the pin to change the LED state

## 10.14 Ball\_Launcher\_Main/Core/Inc/main.h File Reference

Header for **main.c** (p. 99) file. This file contains the common defines of the application.

```
#include "stm32f4xx_hal.h"
```

### Functions

- void **HAL\_TIM\_MspPostInit** (TIM\_HandleTypeDef \*htim)
- void **Error\_Handler** (void)

*This function is executed in case of error occurrence.*

### 10.14.1 Detailed Description

Header for **main.c** (p. 99) file. This file contains the common defines of the application.

#### Attention

Copyright (c) 2024 STMicroelectronics. All rights reserved.

This software is licensed under terms that can be found in the LICENSE file in the root directory of this software component. If no LICENSE file comes with this software, it is provided AS-IS.

### 10.14.2 Function Documentation

#### 10.14.2.1 Error\_Handler()

```
void Error_Handler (
    void )
```

This function is executed in case of error occurrence.

## Return values

None	
------	--

## 10.14.2.2 HAL\_TIM\_MspPostInit()

```
void HAL_TIM_MspPostInit (
    TIM_HandleTypeDef * htim )
```

TIM2 GPIO Configuration PA1 -----> TIM2\_CH2 PA2 -----> TIM2\_CH3

## 10.15 Ball\_Launcher\_Main/Core/Inc/radio\_driver.h File Reference

Header for **radio\_driver.c** (p. 106) file. This file contains the common defines of the application.

```
#include <stdio.h>
#include <stdint.h>
#include "stm32f4xx_hal.h"
```

## Data Structures

- struct **RadioX**

*Structure to define a radio driver with timer, channels, and pulse width parameters.*

## Functions

- void **Radio\_init** ( **RadioX** \*radio, TIM\_HandleTypeDef \*timer, uint32\_t channel1, uint32\_t channel2)  
*Initializes the radio driver with the specified timer and channels.*
- void **capturePulseWidth** ( **RadioX** \*radio)  
*Captures the pulse width for the radio driver.*
- void **update** ( **RadioX** \*radio, int pw1, int pw2)  
*Updates the radio driver with pulse width values.*
- double **Radio\_getPulseWidth** ( **RadioX** \*radio)  
*Gets the pulse width of the radio driver.*

## 10.15.1 Detailed Description

Header for **radio\_driver.c** (p. 106) file. This file contains the common defines of the application.

Created on: May 2, 2024

Author: vvinh

## 10.15.2 Function Documentation

## 10.15.2.1 capturePulseWidth()

```
void capturePulseWidth (
    RadioX * radio )
```

Captures the pulse width for the radio driver.

## Parameters

<i>radio</i>	Pointer to the <b>RadioX</b> (p. 28) structure
--------------	--

< Read captured value for channel 1

< Read captured value for channel 2

< Calculate the pulse width

## 10.15.2.2 Radio\_getPulseWidth()

```
double Radio_getPulseWidth (
    RadioX * radio )
```

Gets the pulse width of the radio driver.

## Parameters

<i>radio</i>	Pointer to the <b>RadioX</b> (p. 28) structure
--------------	--

## Returns

double Pulse width value

## Parameters

<i>radio</i>	Pointer to the <b>RadioX</b> (p. 28) structure
--------------	--

## Returns

double Pulse width value in milliseconds

## 10.15.2.3 Radio\_init()

```
void Radio_init (
    RadioX * radio,
    TIM_HandleTypeDef * timer,
    uint32_t channel1,
    uint32_t channel2 )
```

Initializes the radio driver with the specified timer and channels.

## Parameters

<i>radio</i>	Pointer to the <b>RadioX</b> (p. 28) structure
<i>timer</i>	Timer handle
<i>channel1</i>	Timer channel 1
<i>channel2</i>	Timer channel 2

< Start input capture interrupt for channel 1

< Start input capture interrupt for channel 2

#### 10.15.2.4 update()

```
void update (
    RadioX * radio,
    int pw1,
    int pw2 )
```

Updates the radio driver with pulse width values.

##### Parameters

<i>radio</i>	Pointer to the <b>RadioX</b> (p. 28) structure
<i>pw1</i>	Pulse width value for channel 1
<i>pw2</i>	Pulse width value for channel 2

## 10.16 Ball\_Launcher\_Main/Core/Inc/stepper\_driver.h File Reference

Header for **stepper\_driver.c** (p. 108) file. This file contains the common defines of the application.

```
#include <stdio.h>
#include <stdint.h>
#include "stm32f4xx_hal.h"
```

### Data Structures

- struct **StepperX**  
*Structure to define a stepper motor driver with GPIO ports and pins.*

### Functions

- void **Stepper\_init** ( **StepperX** \*stepperx, GPIO\_TypeDef \*GPIO, uint16\_t EN\_PIN, uint16\_t DIR\_PIN, uint16\_t STP\_PIN)  
*Initializes the stepper motor driver with the specified GPIO port and pins.*
- void **Stepper\_enable** ( **StepperX** \*stepperx)  
*Enables the stepper motor driver.*
- void **Stepper\_disable** ( **StepperX** \*stepperx)  
*Disables the stepper motor driver.*
- void **Stepper\_setspeed** ( **StepperX** \*stepperx, uint16\_t speed, uint8\_t dir)  
*Sets the speed and direction of the stepper motor.*
- void **SysTick\_Init** (void)  
*Initializes the system tick for delay functions.*
- void **Delay\_us** (uint32\_t us)  
*Delays the program execution for a specified number of microseconds.*

## 10.16.1 Detailed Description

Header for **stepper\_driver.c** (p. 108) file. This file contains the common defines of the application.

Created on: May 17, 2024

Author: vvinh

## 10.16.2 Function Documentation

### 10.16.2.1 Delay\_us()

```
void Delay_us (
    uint32_t us )
```

Delays the program execution for a specified number of microseconds.

#### Parameters

<i>us</i>	Number of microseconds to delay
-----------	---------------------------------

### 10.16.2.2 Stepper\_disable()

```
void Stepper_disable (
    StepperX * stepperx )
```

Disables the stepper motor driver.

#### Parameters

<i>stepperx</i>	Pointer to the <b>StepperX</b> (p. 30) structure
-----------------	--

### 10.16.2.3 Stepper\_enable()

```
void Stepper_enable (
    StepperX * stepperx )
```

Enables the stepper motor driver.

#### Parameters

<i>stepperx</i>	Pointer to the <b>StepperX</b> (p. 30) structure
-----------------	--

### 10.16.2.4 Stepper\_init()

```
void Stepper_init (
```

```

    StepperX * stepperx,
    GPIO_TypeDef * GPIO,
    uint16_t EN_PIN,
    uint16_t DIR_PIN,
    uint16_t STP_PIN )

```

Initializes the stepper motor driver with the specified GPIO port and pins.

#### Parameters

<i>stepperx</i>	Pointer to the <b>StepperX</b> (p. 30) structure
<i>GPIO</i>	GPIO port
<i>EN_PIN</i>	Enable pin
<i>DIR_PIN</i>	Direction pin
<i>STP_PIN</i>	Step pin

#### 10.16.2.5 Stepper\_setspeed()

```

void Stepper_setspeed (
    StepperX * stepperx,
    uint16_t speed,
    uint8_t dir )

```

Sets the speed and direction of the stepper motor.

#### Parameters

<i>stepperx</i>	Pointer to the <b>StepperX</b> (p. 30) structure
<i>speed</i>	Speed of the stepper motor
<i>dir</i>	Direction of the stepper motor (1 for one direction, 0 for the opposite)
<i>stepperx</i>	Pointer to the <b>StepperX</b> (p. 30) structure
<i>speed</i>	Speed of the stepper motor (delay between steps)
<i>dir</i>	Direction of the stepper motor (1 for one direction, 0 for the opposite)

< Set the direction pin

< Set the step pin high

< Delay to control the speed

< Set the step pin low

< Delay to control the speed

#### 10.16.2.6 SysTick\_Init()

```

void SysTick_Init (
    void )

```

Initializes the system tick for delay functions.

Initializes the system tick for delay functions.



## 10.17 Ball\_Launcher\_Main/Core/Inc/stm32f4xx\_hal\_conf.h File Reference

```
#include "stm32f4xx_hal_rcc.h"
#include "stm32f4xx_hal_gpio.h"
#include "stm32f4xx_hal_exti.h"
#include "stm32f4xx_hal_dma.h"
#include "stm32f4xx_hal_cortex.h"
#include "stm32f4xx_hal_flash.h"
#include "stm32f4xx_hal_pwr.h"
#include "stm32f4xx_hal_tim.h"
#include "stm32f4xx_hal_uart.h"
```

### Macros

- **#define HAL\_MODULE\_ENABLED**

*This is the list of modules to be used in the HAL driver.*

- **#define HAL\_TIM\_MODULE\_ENABLED**
- **#define HAL\_UART\_MODULE\_ENABLED**
- **#define HAL\_GPIO\_MODULE\_ENABLED**
- **#define HAL\_EXTI\_MODULE\_ENABLED**
- **#define HAL\_DMA\_MODULE\_ENABLED**
- **#define HAL\_RCC\_MODULE\_ENABLED**
- **#define HAL\_FLASH\_MODULE\_ENABLED**
- **#define HAL\_PWR\_MODULE\_ENABLED**
- **#define HAL\_CORTEX\_MODULE\_ENABLED**
- **#define HSE\_VALUE 25000000U**

*Adjust the value of External High Speed oscillator (HSE) used in your application. This value is used by the RCC HAL module to compute the system frequency (when HSE is used as system clock source, directly or through the PLL).*

- **#define HSE\_STARTUP\_TIMEOUT 100U**
- **#define HSI\_VALUE ((uint32\_t)16000000U)**

*Internal High Speed oscillator (HSI) value. This value is used by the RCC HAL module to compute the system frequency (when HSI is used as system clock source, directly or through the PLL).*

- **#define LSI\_VALUE 32000U**

*Internal Low Speed oscillator (LSI) value.*

- **#define LSE\_VALUE 32768U**

*External Low Speed oscillator (LSE) value.*

- **#define LSE\_STARTUP\_TIMEOUT 5000U**
- **#define EXTERNAL\_CLOCK\_VALUE 12288000U**

*External clock source for I2S peripheral This value is used by the I2S HAL module to compute the I2S clock source frequency, this source is inserted directly through I2S\_CKIN pad.*

- **#define VDD\_VALUE 3300U**

*This is the HAL system configuration section.*

- **#define TICK\_INT\_PRIORITY 15U**
- **#define USE\_RTOS 0U**
- **#define PREFETCH\_ENABLE 1U**
- **#define INSTRUCTION\_CACHE\_ENABLE 1U**
- **#define DATA\_CACHE\_ENABLE 1U**
- **#define USE\_HAL\_ADC\_REGISTER\_CALLBACKS 0U** /\* ADC register callback disabled \*/
- **#define USE\_HAL\_CAN\_REGISTER\_CALLBACKS 0U** /\* CAN register callback disabled \*/
- **#define USE\_HAL\_CEC\_REGISTER\_CALLBACKS 0U** /\* CEC register callback disabled \*/
- **#define USE\_HAL\_Cryp\_REGISTER\_CALLBACKS 0U** /\* CRYP register callback disabled \*/

- **#define USE\_HAL\_DAC\_REGISTER\_CALLBACKS** 0U /\* DAC register callback disabled \*/
- **#define USE\_HAL\_DCMI\_REGISTER\_CALLBACKS** 0U /\* DCMI register callback disabled \*/
- **#define USE\_HAL\_DFSDM\_REGISTER\_CALLBACKS** 0U /\* DFSDM register callback disabled \*/
- **#define USE\_HAL\_DMA2D\_REGISTER\_CALLBACKS** 0U /\* DMA2D register callback disabled \*/
- **#define USE\_HAL\_DSI\_REGISTER\_CALLBACKS** 0U /\* DSI register callback disabled \*/
- **#define USE\_HAL\_ETH\_REGISTER\_CALLBACKS** 0U /\* ETH register callback disabled \*/
- **#define USE\_HAL\_HASH\_REGISTER\_CALLBACKS** 0U /\* HASH register callback disabled \*/
- **#define USE\_HAL\_HCD\_REGISTER\_CALLBACKS** 0U /\* HCD register callback disabled \*/
- **#define USE\_HAL\_I2C\_REGISTER\_CALLBACKS** 0U /\* I2C register callback disabled \*/
- **#define USE\_HAL\_FMPI2C\_REGISTER\_CALLBACKS** 0U /\* FMPI2C register callback disabled \*/
- **#define USE\_HAL\_FMPMBUS\_REGISTER\_CALLBACKS** 0U /\* FMPMBUS register callback disabled \*/
- **#define USE\_HAL\_I2S\_REGISTER\_CALLBACKS** 0U /\* I2S register callback disabled \*/
- **#define USE\_HAL\_IRDA\_REGISTER\_CALLBACKS** 0U /\* IRDA register callback disabled \*/
- **#define USE\_HAL\_LPTIM\_REGISTER\_CALLBACKS** 0U /\* LPTIM register callback disabled \*/
- **#define USE\_HAL\_LTDC\_REGISTER\_CALLBACKS** 0U /\* LTDC register callback disabled \*/
- **#define USE\_HAL\_MMC\_REGISTER\_CALLBACKS** 0U /\* MMC register callback disabled \*/
- **#define USE\_HAL\_NAND\_REGISTER\_CALLBACKS** 0U /\* NAND register callback disabled \*/
- **#define USE\_HAL\_NOR\_REGISTER\_CALLBACKS** 0U /\* NOR register callback disabled \*/
- **#define USE\_HAL\_PCCARD\_REGISTER\_CALLBACKS** 0U /\* PCCARD register callback disabled \*/
- **#define USE\_HAL\_PCD\_REGISTER\_CALLBACKS** 0U /\* PCD register callback disabled \*/
- **#define USE\_HAL\_QSPI\_REGISTER\_CALLBACKS** 0U /\* QSPI register callback disabled \*/
- **#define USE\_HAL\_RNG\_REGISTER\_CALLBACKS** 0U /\* RNG register callback disabled \*/
- **#define USE\_HAL\_RTC\_REGISTER\_CALLBACKS** 0U /\* RTC register callback disabled \*/
- **#define USE\_HAL\_SAI\_REGISTER\_CALLBACKS** 0U /\* SAI register callback disabled \*/
- **#define USE\_HAL\_SD\_REGISTER\_CALLBACKS** 0U /\* SD register callback disabled \*/
- **#define USE\_HAL\_SMARTCARD\_REGISTER\_CALLBACKS** 0U /\* SMARTCARD register callback disabled \*/
- **#define USE\_HAL\_SDRAM\_REGISTER\_CALLBACKS** 0U /\* SDRAM register callback disabled \*/
- **#define USE\_HAL\_SRAM\_REGISTER\_CALLBACKS** 0U /\* SRAM register callback disabled \*/
- **#define USE\_HAL\_SPDIFRX\_REGISTER\_CALLBACKS** 0U /\* SPDIFRX register callback disabled \*/
- **#define USE\_HAL\_SMBUS\_REGISTER\_CALLBACKS** 0U /\* SMBUS register callback disabled \*/
- **#define USE\_HAL\_SPI\_REGISTER\_CALLBACKS** 0U /\* SPI register callback disabled \*/
- **#define USE\_HAL\_TIM\_REGISTER\_CALLBACKS** 0U /\* TIM register callback disabled \*/
- **#define USE\_HAL\_UART\_REGISTER\_CALLBACKS** 0U /\* UART register callback disabled \*/
- **#define USE\_HAL\_USART\_REGISTER\_CALLBACKS** 0U /\* USART register callback disabled \*/
- **#define USE\_HAL\_WWDG\_REGISTER\_CALLBACKS** 0U /\* WWDG register callback disabled \*/
- **#define MAC\_ADDR0** 2U

*Uncomment the line below to expanse the "assert\_param" macro in the HAL drivers code.*

- **#define MAC\_ADDR1** 0U
- **#define MAC\_ADDR2** 0U
- **#define MAC\_ADDR3** 0U
- **#define MAC\_ADDR4** 0U
- **#define MAC\_ADDR5** 0U
- **#define ETH\_RX\_BUF\_SIZE** ETH\_MAX\_PACKET\_SIZE /\* **buffer** size for receive \*/
- **#define ETH\_TX\_BUF\_SIZE** ETH\_MAX\_PACKET\_SIZE /\* **buffer** size for transmit \*/
- **#define ETH\_RXBUFNB** 4U /\* 4 Rx buffers of size **ETH\_RX\_BUF\_SIZE** \*/
- **#define ETH\_TXBUFNB** 4U /\* 4 Tx buffers of size **ETH\_TX\_BUF\_SIZE** \*/
- **#define DP83848\_PHY\_ADDRESS**
- **#define PHY\_RESET\_DELAY** 0x000000FFU
- **#define PHY\_CONFIG\_DELAY** 0x00000FFFU
- **#define PHY\_READ\_TO** 0x0000FFFFU
- **#define PHY\_WRITE\_TO** 0x0000FFFFU
- **#define PHY\_BCR** ((uint16\_t)0x0000U)

- `#define PHY_BSR ((uint16_t)0x0001U)`
- `#define PHY_RESET ((uint16_t)0x8000U)`
- `#define PHY_LOOPBACK ((uint16_t)0x4000U)`
- `#define PHY_FULLDUPLEX_100M ((uint16_t)0x2100U)`
- `#define PHY_HALFDUPLEX_100M ((uint16_t)0x2000U)`
- `#define PHY_FULLDUPLEX_10M ((uint16_t)0x0100U)`
- `#define PHY_HALFDUPLEX_10M ((uint16_t)0x0000U)`
- `#define PHY_AUTONEGOTIATION ((uint16_t)0x1000U)`
- `#define PHY_RESTART_AUTONEGOTIATION ((uint16_t)0x0200U)`
- `#define PHY_POWERDOWN ((uint16_t)0x0800U)`
- `#define PHY_ISOLATE ((uint16_t)0x0400U)`
- `#define PHY_AUTONEGO_COMPLETE ((uint16_t)0x0020U)`
- `#define PHY_LINKED_STATUS ((uint16_t)0x0004U)`
- `#define PHY_JABBER_DETECTION ((uint16_t)0x0002U)`
- `#define PHY_SR ((uint16_t))`
- `#define PHY_SPEED_STATUS ((uint16_t))`
- `#define PHY_DUPLEX_STATUS ((uint16_t))`
- `#define USE_SPI_CRC 0U`
- `#define assert_param(expr) ((void)0U)`

*Include module's header file.*

## 10.17.1 Macro Definition Documentation

### 10.17.1.1 `assert_param`

```
#define assert_param(  
    expr ) ((void)0U)
```

Include module's header file.

### 10.17.1.2 `DATA_CACHE_ENABLE`

```
#define DATA_CACHE_ENABLE 1U
```

### 10.17.1.3 `DP83848_PHY_ADDRESS`

```
#define DP83848_PHY_ADDRESS
```

### 10.17.1.4 `ETH_RX_BUF_SIZE`

```
#define ETH_RX_BUF_SIZE ETH_MAX_PACKET_SIZE /* buffer size for receive */
```

### 10.17.1.5 `ETH_RXBUFNB`

```
#define ETH_RXBUFNB 4U /* 4 Rx buffers of size ETH_RX_BUF_SIZE */
```

### 10.17.1.6 ETH\_TX\_BUF\_SIZE

```
#define ETH_TX_BUF_SIZE ETH_MAX_PACKET_SIZE /* buffer size for transmit */
```

### 10.17.1.7 ETH\_TXBUFNB

```
#define ETH_TXBUFNB 4U /* 4 Tx buffers of size ETH_TX_BUF_SIZE */
```

### 10.17.1.8 EXTERNAL\_CLOCK\_VALUE

```
#define EXTERNAL_CLOCK_VALUE 12288000U
```

External clock source for I2S peripheral This value is used by the I2S HAL module to compute the I2S clock source frequency, this source is inserted directly through I2S\_CKIN pad.

Value of the External audio frequency in Hz

### 10.17.1.9 HAL\_CORTEX\_MODULE\_ENABLED

```
#define HAL_CORTEX_MODULE_ENABLED
```

### 10.17.1.10 HAL\_DMA\_MODULE\_ENABLED

```
#define HAL_DMA_MODULE_ENABLED
```

### 10.17.1.11 HAL\_EXTI\_MODULE\_ENABLED

```
#define HAL_EXTI_MODULE_ENABLED
```

### 10.17.1.12 HAL\_FLASH\_MODULE\_ENABLED

```
#define HAL_FLASH_MODULE_ENABLED
```

### 10.17.1.13 HAL\_GPIO\_MODULE\_ENABLED

```
#define HAL_GPIO_MODULE_ENABLED
```

### 10.17.1.14 HAL\_MODULE\_ENABLED

```
#define HAL_MODULE_ENABLED
```

This is the list of modules to be used in the HAL driver.

**10.17.1.15 HAL\_PWR\_MODULE\_ENABLED**

```
#define HAL_PWR_MODULE_ENABLED
```

**10.17.1.16 HAL\_RCC\_MODULE\_ENABLED**

```
#define HAL_RCC_MODULE_ENABLED
```

**10.17.1.17 HAL\_TIM\_MODULE\_ENABLED**

```
#define HAL_TIM_MODULE_ENABLED
```

**10.17.1.18 HAL\_UART\_MODULE\_ENABLED**

```
#define HAL_UART_MODULE_ENABLED
```

**10.17.1.19 HSE\_STARTUP\_TIMEOUT**

```
#define HSE_STARTUP_TIMEOUT 100U
```

Time out for HSE start up, in ms

**10.17.1.20 HSE\_VALUE**

```
#define HSE_VALUE 25000000U
```

Adjust the value of External High Speed oscillator (HSE) used in your application. This value is used by the RCC HAL module to compute the system frequency (when HSE is used as system clock source, directly or through the PLL).

Value of the External oscillator in Hz

**10.17.1.21 HSI\_VALUE**

```
#define HSI_VALUE ((uint32_t)16000000U)
```

Internal High Speed oscillator (HSI) value. This value is used by the RCC HAL module to compute the system frequency (when HSI is used as system clock source, directly or through the PLL).

Value of the Internal oscillator in Hz

**10.17.1.22 INSTRUCTION\_CACHE\_ENABLE**

```
#define INSTRUCTION_CACHE_ENABLE 1U
```

#### 10.17.1.23 LSE\_STARTUP\_TIMEOUT

```
#define LSE_STARTUP_TIMEOUT 5000U
```

Time out for LSE start up, in ms

#### 10.17.1.24 LSE\_VALUE

```
#define LSE_VALUE 32768U
```

External Low Speed oscillator (LSE) value.

< Value of the Internal Low Speed oscillator in Hz The real value may vary depending on the variations in voltage and temperature. Value of the External Low Speed oscillator in Hz

#### 10.17.1.25 LSI\_VALUE

```
#define LSI_VALUE 32000U
```

Internal Low Speed oscillator (LSI) value.

LSI Typical Value in Hz

#### 10.17.1.26 MAC\_ADDR0

```
#define MAC_ADDR0 2U
```

Uncomment the line below to expanse the "assert\_param" macro in the HAL drivers code.

#### 10.17.1.27 MAC\_ADDR1

```
#define MAC_ADDR1 0U
```

#### 10.17.1.28 MAC\_ADDR2

```
#define MAC_ADDR2 0U
```

#### 10.17.1.29 MAC\_ADDR3

```
#define MAC_ADDR3 0U
```

#### 10.17.1.30 MAC\_ADDR4

```
#define MAC_ADDR4 0U
```

#### 10.17.1.31 MAC\_ADDR5

```
#define MAC_ADDR5 0U
```

#### 10.17.1.32 PHY\_AUTONEGO\_COMPLETE

```
#define PHY_AUTONEGO_COMPLETE ((uint16_t)0x0020U)
```

Auto-Negotiation process completed

#### 10.17.1.33 PHY\_AUTONEGOTIATION

```
#define PHY_AUTONEGOTIATION ((uint16_t)0x1000U)
```

Enable auto-negotiation function

#### 10.17.1.34 PHY\_BCR

```
#define PHY_BCR ((uint16_t)0x0000U)
```

Transceiver Basic Control Register

#### 10.17.1.35 PHY\_BSR

```
#define PHY_BSR ((uint16_t)0x0001U)
```

Transceiver Basic Status Register

#### 10.17.1.36 PHY\_CONFIG\_DELAY

```
#define PHY_CONFIG_DELAY 0x00000FFFU
```

#### 10.17.1.37 PHY\_DUPLEX\_STATUS

```
#define PHY_DUPLEX_STATUS ((uint16_t))
```

PHY Duplex mask

**10.17.1.38 PHY\_FULLDUPLEX\_100M**

```
#define PHY_FULLDUPLEX_100M ((uint16_t)0x2100U)
```

Set the full-duplex mode at 100 Mb/s

**10.17.1.39 PHY\_FULLDUPLEX\_10M**

```
#define PHY_FULLDUPLEX_10M ((uint16_t)0x0100U)
```

Set the full-duplex mode at 10 Mb/s

**10.17.1.40 PHY\_HALFDUPLEX\_100M**

```
#define PHY_HALFDUPLEX_100M ((uint16_t)0x2000U)
```

Set the half-duplex mode at 100 Mb/s

**10.17.1.41 PHY\_HALFDUPLEX\_10M**

```
#define PHY_HALFDUPLEX_10M ((uint16_t)0x0000U)
```

Set the half-duplex mode at 10 Mb/s

**10.17.1.42 PHY\_ISOLATE**

```
#define PHY_ISOLATE ((uint16_t)0x0400U)
```

Isolate PHY from MII

**10.17.1.43 PHY\_JABBER\_DETECTION**

```
#define PHY_JABBER_DETECTION ((uint16_t)0x0002U)
```

Jabber condition detected

**10.17.1.44 PHY\_LINKED\_STATUS**

```
#define PHY_LINKED_STATUS ((uint16_t)0x0004U)
```

Valid link established



#### 10.17.1.45 PHY\_LOOPBACK

```
#define PHY_LOOPBACK ((uint16_t)0x4000U)
```

Select loop-back mode

#### 10.17.1.46 PHY\_POWERDOWN

```
#define PHY_POWERDOWN ((uint16_t)0x0800U)
```

Select the power down mode

#### 10.17.1.47 PHY\_READ\_TO

```
#define PHY_READ_TO 0x0000FFFFU
```

#### 10.17.1.48 PHY\_RESET

```
#define PHY_RESET ((uint16_t)0x8000U)
```

PHY Reset

#### 10.17.1.49 PHY\_RESET\_DELAY

```
#define PHY_RESET_DELAY 0x000000FFU
```

#### 10.17.1.50 PHY\_RESTART\_AUTONEGOTIATION

```
#define PHY_RESTART_AUTONEGOTIATION ((uint16_t)0x0200U)
```

Restart auto-negotiation function

#### 10.17.1.51 PHY\_SPEED\_STATUS

```
#define PHY_SPEED_STATUS ((uint16_t))
```

PHY Speed mask

#### 10.17.1.52 PHY\_SR

```
#define PHY_SR ((uint16_t))
```

PHY status register Offset

#### 10.17.1.53 PHY\_WRITE\_TO

```
#define PHY_WRITE_TO 0x0000FFFFU
```

#### 10.17.1.54 PREFETCH\_ENABLE

```
#define PREFETCH_ENABLE 1U
```

#### 10.17.1.55 TICK\_INT\_PRIORITY

```
#define TICK_INT_PRIORITY 15U
```

tick interrupt priority

#### 10.17.1.56 USE\_HAL\_ADC\_REGISTER\_CALLBACKS

```
#define USE_HAL_ADC_REGISTER_CALLBACKS 0U /* ADC register callback disabled */
```

#### 10.17.1.57 USE\_HAL\_CAN\_REGISTER\_CALLBACKS

```
#define USE_HAL_CAN_REGISTER_CALLBACKS 0U /* CAN register callback disabled */
```

#### 10.17.1.58 USE\_HAL\_CEC\_REGISTER\_CALLBACKS

```
#define USE_HAL_CEC_REGISTER_CALLBACKS 0U /* CEC register callback disabled */
```

#### 10.17.1.59 USE\_HAL\_Cryp\_REGISTER\_CALLBACKS

```
#define USE_HAL_Cryp_REGISTER_CALLBACKS 0U /* CRYPT register callback disabled */
```

#### 10.17.1.60 USE\_HAL\_DAC\_REGISTER\_CALLBACKS

```
#define USE_HAL_DAC_REGISTER_CALLBACKS 0U /* DAC register callback disabled */
```

#### 10.17.1.61 USE\_HAL\_DCMI\_REGISTER\_CALLBACKS

```
#define USE_HAL_DCMI_REGISTER_CALLBACKS 0U /* DCMI register callback disabled */
```

#### 10.17.1.62 USE\_HAL\_DFSDM\_REGISTER\_CALLBACKS

```
#define USE_HAL_DFSDM_REGISTER_CALLBACKS 0U /* DFSDM register callback disabled */
```

#### 10.17.1.63 USE\_HAL\_DMA2D\_REGISTER\_CALLBACKS

```
#define USE_HAL_DMA2D_REGISTER_CALLBACKS 0U /* DMA2D register callback disabled */
```

#### 10.17.1.64 USE\_HAL\_DSI\_REGISTER\_CALLBACKS

```
#define USE_HAL_DSI_REGISTER_CALLBACKS 0U /* DSI register callback disabled */
```

#### 10.17.1.65 USE\_HAL\_ETH\_REGISTER\_CALLBACKS

```
#define USE_HAL_ETH_REGISTER_CALLBACKS 0U /* ETH register callback disabled */
```

#### 10.17.1.66 USE\_HAL\_FMPI2C\_REGISTER\_CALLBACKS

```
#define USE_HAL_FMPI2C_REGISTER_CALLBACKS 0U /* FMPI2C register callback disabled */
```

#### 10.17.1.67 USE\_HAL\_FMPMBUS\_REGISTER\_CALLBACKS

```
#define USE_HAL_FMPMBUS_REGISTER_CALLBACKS 0U /* FMPMBUS register callback disabled */
```

#### 10.17.1.68 USE\_HAL\_HASH\_REGISTER\_CALLBACKS

```
#define USE_HAL_HASH_REGISTER_CALLBACKS 0U /* HASH register callback disabled */
```

#### 10.17.1.69 USE\_HAL\_HCD\_REGISTER\_CALLBACKS

```
#define USE_HAL_HCD_REGISTER_CALLBACKS 0U /* HCD register callback disabled */
```

#### 10.17.1.70 USE\_HAL\_I2C\_REGISTER\_CALLBACKS

```
#define USE_HAL_I2C_REGISTER_CALLBACKS 0U /* I2C register callback disabled */
```

#### 10.17.1.71 USE\_HAL\_I2S\_REGISTER\_CALLBACKS

```
#define USE_HAL_I2S_REGISTER_CALLBACKS 0U /* I2S register callback disabled */
```

#### 10.17.1.72 USE\_HAL\_IRDA\_REGISTER\_CALLBACKS

```
#define USE_HAL_IRDA_REGISTER_CALLBACKS 0U /* IRDA register callback disabled */
```

#### 10.17.1.73 USE\_HAL\_LPTIM\_REGISTER\_CALLBACKS

```
#define USE_HAL_LPTIM_REGISTER_CALLBACKS 0U /* LPTIM register callback disabled */
```

#### 10.17.1.74 USE\_HAL\_LTDC\_REGISTER\_CALLBACKS

```
#define USE_HAL_LTDC_REGISTER_CALLBACKS 0U /* LTDC register callback disabled */
```

#### 10.17.1.75 USE\_HAL\_MMC\_REGISTER\_CALLBACKS

```
#define USE_HAL_MMC_REGISTER_CALLBACKS 0U /* MMC register callback disabled */
```

#### 10.17.1.76 USE\_HAL\_NAND\_REGISTER\_CALLBACKS

```
#define USE_HAL_NAND_REGISTER_CALLBACKS 0U /* NAND register callback disabled */
```

#### 10.17.1.77 USE\_HAL\_NOR\_REGISTER\_CALLBACKS

```
#define USE_HAL_NOR_REGISTER_CALLBACKS 0U /* NOR register callback disabled */
```

#### 10.17.1.78 USE\_HAL\_PCCARD\_REGISTER\_CALLBACKS

```
#define USE_HAL_PCCARD_REGISTER_CALLBACKS 0U /* PCCARD register callback disabled */
```

#### 10.17.1.79 USE\_HAL\_PCD\_REGISTER\_CALLBACKS

```
#define USE_HAL_PCD_REGISTER_CALLBACKS 0U /* PCD register callback disabled */
```

#### 10.17.1.80 USE\_HAL\_QSPI\_REGISTER\_CALLBACKS

```
#define USE_HAL_QSPI_REGISTER_CALLBACKS 0U /* QSPI register callback disabled */
```

#### 10.17.1.81 USE\_HAL\_RNG\_REGISTER\_CALLBACKS

```
#define USE_HAL_RNG_REGISTER_CALLBACKS 0U /* RNG register callback disabled */
```

#### 10.17.1.82 USE\_HAL\_RTC\_REGISTER\_CALLBACKS

```
#define USE_HAL_RTC_REGISTER_CALLBACKS 0U /* RTC register callback disabled */
```

#### 10.17.1.83 USE\_HAL\_SAI\_REGISTER\_CALLBACKS

```
#define USE_HAL_SAI_REGISTER_CALLBACKS 0U /* SAI register callback disabled */
```

#### 10.17.1.84 USE\_HAL\_SD\_REGISTER\_CALLBACKS

```
#define USE_HAL_SD_REGISTER_CALLBACKS 0U /* SD register callback disabled */
```

#### 10.17.1.85 USE\_HAL\_SDRAM\_REGISTER\_CALLBACKS

```
#define USE_HAL_SDRAM_REGISTER_CALLBACKS 0U /* SDRAM register callback disabled */
```

#### 10.17.1.86 USE\_HAL\_SMARTCARD\_REGISTER\_CALLBACKS

```
#define USE_HAL_SMARTCARD_REGISTER_CALLBACKS 0U /* SMARTCARD register callback disabled */
```

#### 10.17.1.87 USE\_HAL\_SMBUS\_REGISTER\_CALLBACKS

```
#define USE_HAL_SMBUS_REGISTER_CALLBACKS 0U /* SMBUS register callback disabled */
```

#### 10.17.1.88 USE\_HAL\_SPDIFRX\_REGISTER\_CALLBACKS

```
#define USE_HAL_SPDIFRX_REGISTER_CALLBACKS 0U /* SPDIFRX register callback disabled */
```

#### 10.17.1.89 USE\_HAL\_SPI\_REGISTER\_CALLBACKS

```
#define USE_HAL_SPI_REGISTER_CALLBACKS 0U /* SPI register callback disabled */
```

#### 10.17.1.90 USE\_HAL\_SRAM\_REGISTER\_CALLBACKS

```
#define USE_HAL_SRAM_REGISTER_CALLBACKS 0U /* SRAM register callback disabled */
```

#### 10.17.1.91 USE\_HAL\_TIM\_REGISTER\_CALLBACKS

```
#define USE_HAL_TIM_REGISTER_CALLBACKS 0U /* TIM register callback disabled */
```

#### 10.17.1.92 USE\_HAL\_UART\_REGISTER\_CALLBACKS

```
#define USE_HAL_UART_REGISTER_CALLBACKS 0U /* UART register callback disabled */
```

### 10.17.1.93 USE\_HAL\_USART\_REGISTER\_CALLBACKS

```
#define USE_HAL_USART_REGISTER_CALLBACKS 0U /* USART register callback disabled */
```

### 10.17.1.94 USE\_HAL\_WWDG\_REGISTER\_CALLBACKS

```
#define USE_HAL_WWDG_REGISTER_CALLBACKS 0U /* WWDG register callback disabled */
```

### 10.17.1.95 USE\_RTOS

```
#define USE_RTOS 0U
```

### 10.17.1.96 USE\_SPI\_CRC

```
#define USE_SPI_CRC 0U
```

### 10.17.1.97 VDD\_VALUE

```
#define VDD_VALUE 3300U
```

This is the HAL system configuration section.

Value of VDD in mv

## 10.18 Ball\_Launcher\_Main/Core/Inc/stm32f4xx\_it.h File Reference

This file contains the headers of the interrupt handlers.

### Functions

- void **NMI\_Handler** (void)  
*This function handles Non maskable interrupt.*
- void **HardFault\_Handler** (void)  
*This function handles Hard fault interrupt.*
- void **MemManage\_Handler** (void)  
*This function handles Memory management fault.*
- void **BusFault\_Handler** (void)  
*This function handles Pre-fetch fault, memory access fault.*
- void **UsageFault\_Handler** (void)  
*This function handles Undefined instruction or illegal state.*
- void **SVC\_Handler** (void)  
*This function handles System service call via SWI instruction.*
- void **DebugMon\_Handler** (void)  
*This function handles Debug monitor.*
- void **PendSV\_Handler** (void)  
*This function handles Pendable request for system service.*
- void **SysTick\_Handler** (void)  
*This function handles System tick timer.*
- void **TIM4\_IRQHandler** (void)  
*This function handles TIM4 global interrupt.*
- void **USART1\_IRQHandler** (void)  
*This function handles USART1 global interrupt.*

## 10.18.1 Detailed Description

This file contains the headers of the interrupt handlers.

### Attention

Copyright (c) 2024 STMicroelectronics. All rights reserved.

This software is licensed under terms that can be found in the LICENSE file in the root directory of this software component. If no LICENSE file comes with this software, it is provided AS-IS.

## 10.18.2 Function Documentation

### 10.18.2.1 BusFault\_Handler()

```
void BusFault_Handler (
    void )
```

This function handles Pre-fetch fault, memory access fault.

### 10.18.2.2 DebugMon\_Handler()

```
void DebugMon_Handler (
    void )
```

This function handles Debug monitor.

### 10.18.2.3 HardFault\_Handler()

```
void HardFault_Handler (
    void )
```

This function handles Hard fault interrupt.

### 10.18.2.4 MemManage\_Handler()

```
void MemManage_Handler (
    void )
```

This function handles Memory management fault.

### 10.18.2.5 NMI\_Handler()

```
void NMI_Handler (
    void )
```

This function handles Non maskable interrupt.

#### 10.18.2.6 PendSV\_Handler()

```
void PendSV_Handler (
    void )
```

This function handles Pendable request for system service.

#### 10.18.2.7 SVC\_Handler()

```
void SVC_Handler (
    void )
```

This function handles System service call via SWI instruction.

#### 10.18.2.8 SysTick\_Handler()

```
void SysTick_Handler (
    void )
```

This function handles System tick timer.

#### 10.18.2.9 TIM4\_IRQHandler()

```
void TIM4_IRQHandler (
    void )
```

This function handles TIM4 global interrupt.

#### 10.18.2.10 UsageFault\_Handler()

```
void UsageFault_Handler (
    void )
```

This function handles Undefined instruction or illegal state.

#### 10.18.2.11 USART1\_IRQHandler()

```
void USART1_IRQHandler (
    void )
```

This function handles USART1 global interrupt.

## 10.19 Ball\_Launcher\_Main/Core/Inc/switch.h File Reference

Header for **switch.c** (p. 117) file. This file contains the common defines of the application.

```
#include <stdio.h>
#include <stdint.h>
#include "stm32f4xx_hal.h"
```



## Data Structures

- struct **SwitchX**

*Structure to define a switch with GPIO port, pin, and status.*

## Functions

- void **Switch\_init** ( **SwitchX** \*switchx, GPIO\_TypeDef \*GPIO, uint16\_t PIN)  
*Initializes the switch with the specified GPIO port and pin.*
- uint8\_t **Switch\_getStatus** ( **SwitchX** \*switchx)  
*Gets the status of the switch.*

### 10.19.1 Detailed Description

Header for **switch.c** (p. 117) file. This file contains the common defines of the application.

Created on: May 17, 2024

Author: vvinh

### 10.19.2 Function Documentation

#### 10.19.2.1 Switch\_getStatus()

```
uint8_t Switch_getStatus (  
    SwitchX * switchx )
```

Gets the status of the switch.

##### Parameters

<i>switchx</i>	Pointer to the <b>SwitchX</b> (p. 31) structure
----------------	---

##### Returns

uint8\_t Status of the switch (1 if pressed, 0 if not pressed)

#### 10.19.2.2 Switch\_init()

```
void Switch_init (  
    SwitchX * switchx,  
    GPIO_TypeDef * GPIO,  
    uint16_t PIN )
```

Initializes the switch with the specified GPIO port and pin.

## Parameters

<i>switchx</i>	Pointer to the <b>SwitchX</b> (p. 31) structure
<i>GPIO</i>	GPIO port
<i>PIN</i>	GPIO pin

< Assign the GPIO port

< Assign the GPIO pin

< Initialize the status to 0

## 10.20 Ball\_Launcher\_Main/Core/Src/d4215.c File Reference

Source file for D4215 motor operations. This file contains the implementation of the functions for initializing and controlling a D4215 motor.

```
#include "d4215.h"
#include "stm32f4xx_hal.h"
#include <stdio.h>
#include <stdint.h>
```

### Functions

- void **D4215\_init** ( **D4215X** \*bldcx, TIM\_HandleTypeDef \*timer, uint32\_t channel)  
*Initializes the D4215 motor with the specified timer and channel.*
- void **D4215\_set** ( **D4215X** \*bldcx, int32\_t spd)  
*Sets the speed of the D4215 motor.*

### 10.20.1 Detailed Description

Source file for D4215 motor operations. This file contains the implementation of the functions for initializing and controlling a D4215 motor.

Created on: May 1, 2024

Author: vvinh

### 10.20.2 Function Documentation

#### 10.20.2.1 D4215\_init()

```
void D4215_init (
    D4215X * bldcx,
    TIM_HandleTypeDef * timer,
    uint32_t channel )
```

Initializes the D4215 motor with the specified timer and channel.

## Parameters

<i>bldcx</i>	Pointer to the <b>D4215X</b> (p. 25) structure
<i>timer</i>	Timer handle
<i>channel</i>	Timer channel

< Start PWM for the specified timer and channel

## 10.20.2.2 D4215\_set()

```
void D4215_set (
    D4215X * bldcx,
    int32_t spd )
```

Sets the speed of the D4215 motor.

## Parameters

<i>bldcx</i>	Pointer to the <b>D4215X</b> (p. 25) structure
<i>spd</i>	Speed of the motor (range: 5 to 10)

< Calculate the duty cycle based on the input speed

< Set the duty cycle for the PWM

## 10.21 Ball\_Launcher\_Main/Core/Src/led.c File Reference

Source file for LED operations. This file contains the implementation of the functions for initializing and controlling an LED.

```
#include "led.h"
#include "stm32f4xx_hal.h"
#include <stdio.h>
#include <stdint.h>
```

## Functions

- void **LED\_init** ( **LEDX** \*LEDx, GPIO\_TypeDef \*GPIO, uint16\_t PIN)  
*Initializes the LED with the specified GPIO port and pin.*
- void **LED\_on** ( **LEDX** \*LEDx)  
*Turns on the LED.*
- void **LED\_off** ( **LEDX** \*LEDx)  
*Turns off the LED.*
- void **LED\_toggle** ( **LEDX** \*LEDx)  
*Toggles the LED state.*

### 10.21.1 Detailed Description

Source file for LED operations. This file contains the implementation of the functions for initializing and controlling an LED.

Created on: May 17, 2024

Author: vvinh

### 10.21.2 Function Documentation

#### 10.21.2.1 LED\_init()

```
void LED_init (
    LEDX * LEDx,
    GPIO_TypeDef * GPIO,
    uint16_t PIN )
```

Initializes the LED with the specified GPIO port and pin.

##### Parameters

<i>LEDx</i>	Pointer to the <b>LEDX</b> (p. 26) structure
<i>GPIO</i>	GPIO port
<i>PIN</i>	GPIO pin

< Assign the GPIO port

< Assign the GPIO pin

#### 10.21.2.2 LED\_off()

```
void LED_off (
    LEDX * LEDx )
```

Turns off the LED.

##### Parameters

<i>LEDx</i>	Pointer to the <b>LEDX</b> (p. 26) structure
-------------	--

< Set the pin low to turn off the LED

#### 10.21.2.3 LED\_on()

```
void LED_on (
    LEDX * LEDx )
```

Turns on the LED.

## Parameters

<i>LEDx</i>	Pointer to the <b>LEDX</b> (p. 26) structure
-------------	--

< Set the pin high to turn on the LED

## 10.21.2.4 LED\_toggle()

```
void LED_toggle (
    LEDX * LEDx )
```

Toggles the LED state.

## Parameters

<i>LEDx</i>	Pointer to the <b>LEDX</b> (p. 26) structure
-------------	--

< Toggle the pin to change the LED state

## 10.22 Ball\_Launcher\_Main/Core/Src/main.c File Reference

Main program body.

```
#include "main.h"
#include <stdio.h>
#include <stdint.h>
#include <string.h>
#include "stm32f4xx_hal.h"
#include "stepper_driver.h"
#include "switch.h"
#include "d4215.h"
#include "led.h"
```

## Functions

- void **SystemClock\_Config** (void)  
*System Clock Configuration.*
- void **HAL\_UART\_RxCpltCallback** (UART\_HandleTypeDef \*huart)
- void **dataProcess** (char \* data)
- uint16\_t **speedCalculate** (uint16\_t first, uint16\_t second, uint16\_t third)
- uint16\_t **map** (uint16\_t x, uint16\_t in\_min, uint16\_t in\_max, uint16\_t out\_min, uint16\_t out\_max)
- int **main** (void)  
*The application entry point.*
- void **HAL\_TIM\_IC\_CaptureCallback** (TIM\_HandleTypeDef \*htim)
- void **Error\_Handler** (void)  
*This function is executed in case of error occurrence.*

## Variables

- TIM\_HandleTypeDef **htim2**
- TIM\_HandleTypeDef **htim4**
- UART\_HandleTypeDef **huart1**
- UART\_HandleTypeDef **huart6**
- char **charIn**
- char **data** [20]
- uint32\_t **ti** = 0
- uint32\_t **tf** = 0
- uint32\_t **dt** = 0
- uint16\_t **yaw** = 0
- uint16\_t **pitch** = 0
- uint16\_t **angleTarget** = 0
- uint16\_t **rise** = 0
- uint16\_t **fall** = 0
- uint16\_t **total** = 0
- uint8\_t **yawDirection** = 0
- uint8\_t **pitchDirection** = 0
- uint8\_t **yawDirectionSW** = 0
- uint8\_t **pitchDirectionSW** = 0
- uint8\_t **idx** = 0
- uint8\_t **MOVE** = 0
- uint8\_t **SHOT** = 0
- uint8\_t **ESTOP** = 0
- uint8\_t **CAL** = 0
- uint8\_t **HOME** = 0
- uint8\_t **SW1** = 0
- uint8\_t **SW2** = 0
- uint8\_t **SW3** = 0
- uint8\_t **SW** = 0
- uint8\_t **STATE** = 0
- uint8\_t **STATE\_0\_INIT** = 0
- uint8\_t **STATE\_1\_HUB** = 1
- uint8\_t **STATE\_2\_ESTOP** = 2
- uint8\_t **STATE\_3\_STEPPER** = 3
- uint8\_t **STATE\_4\_BLDC** = 4

### 10.22.1 Detailed Description

Main program body.

#### Attention

Copyright (c) 2024 STMicroelectronics. All rights reserved.

This software is licensed under terms that can be found in the LICENSE file in the root directory of this software component. If no LICENSE file comes with this software, it is provided AS-IS.

## 10.22.2 Function Documentation

### 10.22.2.1 dataProcess()

```
void dataProcess (
    char * data )
```

### 10.22.2.2 Error\_Handler()

```
void Error_Handler (
    void )
```

This function is executed in case of error occurrence.

Return values

None	
------	--

### 10.22.2.3 HAL\_TIM\_IC\_CaptureCallback()

```
void HAL_TIM_IC_CaptureCallback (
    TIM_HandleTypeDef * htim )
```

### 10.22.2.4 HAL\_UART\_RxCpltCallback()

```
void HAL_UART_RxCpltCallback (
    UART_HandleTypeDef * huart )
```

### 10.22.2.5 main()

```
int main (
    void )
```

The application entry point.

Return values

int	
-----	--

### 10.22.2.6 map()

```
uint16_t map (
    uint16_t x,
    uint16_t in_min,
    uint16_t in_max,
```

```
uint16_t out_min,
uint16_t out_max )
```

### 10.22.2.7 speedCalculate()

```
uint16_t speedCalculate (
    uint16_t first,
    uint16_t second,
    uint16_t third )
```

### 10.22.2.8 SystemClock\_Config()

```
void SystemClock_Config (
    void )
```

System Clock Configuration.

#### Return values

None	
------	--

Configure the main internal regulator output voltage

Initializes the RCC Oscillators according to the specified parameters in the RCC\_OscInitTypeDef structure.

Initializes the CPU, AHB and APB buses clocks

## 10.22.3 Variable Documentation

### 10.22.3.1 angleTarget

```
uint16_t angleTarget = 0
```

### 10.22.3.2 CAL

```
uint8_t CAL = 0
```

### 10.22.3.3 charIn

```
char charIn
```

### 10.22.3.4 data

```
char data[20]
```



#### 10.22.3.5 dt

```
uint32_t dt = 0
```

#### 10.22.3.6 ESTOP

```
uint8_t ESTOP = 0
```

#### 10.22.3.7 fall

```
uint16_t fall = 0
```

#### 10.22.3.8 HOME

```
uint8_t HOME = 0
```

#### 10.22.3.9 htim2

```
TIM_HandleTypeDef htim2
```

#### 10.22.3.10 htim4

```
TIM_HandleTypeDef htim4
```

#### 10.22.3.11 huart1

```
UART_HandleTypeDef huart1
```

#### 10.22.3.12 huart6

```
UART_HandleTypeDef huart6
```

#### 10.22.3.13 idx

```
uint8_t idx = 0
```

#### 10.22.3.14 MOVE

```
uint8_t MOVE = 0
```

**10.22.3.15 pitch**

```
uint16_t pitch = 0
```

**10.22.3.16 pitchDirection**

```
uint8_t pitchDirection = 0
```

**10.22.3.17 pitchDirectionSW**

```
uint8_t pitchDirectionSW = 0
```

**10.22.3.18 rise**

```
uint16_t rise = 0
```

**10.22.3.19 SHOT**

```
uint8_t SHOT = 0
```

**10.22.3.20 STATE**

```
uint8_t STATE = 0
```

**10.22.3.21 STATE\_0\_INIT**

```
uint8_t STATE_0_INIT = 0
```

**10.22.3.22 STATE\_1\_HUB**

```
uint8_t STATE_1_HUB = 1
```

**10.22.3.23 STATE\_2\_ESTOP**

```
uint8_t STATE_2_ESTOP = 2
```

**10.22.3.24 STATE\_3\_STEPPER**

```
uint8_t STATE_3_STEPPER = 3
```

**10.22.3.25 STATE\_4\_BLDC**

```
uint8_t STATE_4_BLDC = 4
```

**10.22.3.26 SW**

```
uint8_t SW = 0
```

**10.22.3.27 SW1**

```
uint8_t SW1 = 0
```

**10.22.3.28 SW2**

```
uint8_t SW2 = 0
```

**10.22.3.29 SW3**

```
uint8_t SW3 = 0
```

**10.22.3.30 tf**

```
uint32_t tf = 0
```

**10.22.3.31 ti**

```
uint32_t ti = 0
```

**10.22.3.32 total**

```
uint16_t total = 0
```

**10.22.3.33 yaw**

```
uint16_t yaw = 0
```

**10.22.3.34 yawDirection**

```
uint8_t yawDirection = 0
```

### 10.22.3.35 yawDirectionSW

```
uint8_t yawDirectionSW = 0
```

## 10.23 Ball\_Launcher\_Main/Core/Src/radio\_driver.c File Reference

Source file for radio driver operations. This file contains the implementation of the functions for initializing and handling a radio driver.

```
#include "radio_driver.h"
#include "stm32f4xx_hal.h"
#include <stdio.h>
#include <stdint.h>
```

### Functions

- void **Radio\_init** ( **RadioX** \*radio, TIM\_HandleTypeDef \*timer, uint32\_t channel1, uint32\_t channel2)  
*Initializes the radio driver with the specified timer and channels.*
- void **capturePulseWidth** ( **RadioX** \*radio)  
*Captures the pulse width for the radio driver.*
- void **update** ( **RadioX** \*radio, int pw1, int pw2)  
*Updates the radio driver with pulse width values.*
- double **Radio\_getPulseWidth** ( **RadioX** \*radio)  
*Gets the pulse width of the radio driver.*

### 10.23.1 Detailed Description

Source file for radio driver operations. This file contains the implementation of the functions for initializing and handling a radio driver.

Created on: May 2, 2024

Author: vvinh

### 10.23.2 Function Documentation

#### 10.23.2.1 capturePulseWidth()

```
void capturePulseWidth (
    RadioX * radio )
```

Captures the pulse width for the radio driver.

#### Parameters

<i>radio</i>	Pointer to the <b>RadioX</b> (p. 28) structure
--------------	--

- < Read captured value for channel 1
- < Read captured value for channel 2
- < Calculate the pulse width

#### 10.23.2.2 Radio\_getPulseWidth()

```
double Radio_getPulseWidth (
    RadioX * radio )
```

Gets the pulse width of the radio driver.

##### Parameters

<i>radio</i>	Pointer to the <b>RadioX</b> (p. 28) structure
--------------	--

##### Returns

double Pulse width value in milliseconds

#### 10.23.2.3 Radio\_init()

```
void Radio_init (
    RadioX * radio,
    TIM_HandleTypeDef * timer,
    uint32_t channel1,
    uint32_t channel2 )
```

Initializes the radio driver with the specified timer and channels.

##### Parameters

<i>radio</i>	Pointer to the <b>RadioX</b> (p. 28) structure
<i>timer</i>	Timer handle
<i>channel1</i>	Timer channel 1
<i>channel2</i>	Timer channel 2

- < Start input capture interrupt for channel 1
- < Start input capture interrupt for channel 2

#### 10.23.2.4 update()

```
void update (
    RadioX * radio,
    int pw1,
    int pw2 )
```

Updates the radio driver with pulse width values.

## Parameters

<i>radio</i>	Pointer to the <b>RadioX</b> (p. 28) structure
<i>pw1</i>	Pulse width value for channel 1
<i>pw2</i>	Pulse width value for channel 2

## 10.24 Ball\_Launcher\_Main/Core/Src/stepper\_driver.c File Reference

Source file for stepper motor driver operations. This file contains the implementation of the functions for initializing and controlling a stepper motor.

```
#include "stepper_driver.h"
#include "stm32f4xx_hal.h"
#include <stdio.h>
#include <stdint.h>
```

## Functions

- void **Stepper\_init** ( **StepperX** \*stepperx, GPIO\_TypeDef \*GPIO, uint16\_t EN\_PIN, uint16\_t DIR\_PIN, uint16\_t STP\_PIN)  
*Initializes the stepper motor driver with the specified GPIO port and pins.*
- void **Stepper\_enable** ( **StepperX** \*stepperx)  
*Enables the stepper motor driver.*
- void **Stepper\_disable** ( **StepperX** \*stepperx)  
*Disables the stepper motor driver.*
- void **Stepper\_setspeed** ( **StepperX** \*stepperx, uint16\_t speed, uint8\_t dir)  
*Sets the speed and direction of the stepper motor.*
- void **SysTick\_Init** (void)  
*Initializes the system tick for microsecond delays.*
- void **Delay\_us** (uint32\_t us)  
*Delays the program execution for a specified number of microseconds.*

### 10.24.1 Detailed Description

Source file for stepper motor driver operations. This file contains the implementation of the functions for initializing and controlling a stepper motor.

Created on: May 17, 2024

Author: vvinh

### 10.24.2 Function Documentation

#### 10.24.2.1 Delay\_us()

```
void Delay_us (
    uint32_t us )
```

Delays the program execution for a specified number of microseconds.

## Parameters

<i>us</i>	Number of microseconds to delay
-----------	---------------------------------

### 10.24.2.2 Stepper\_disable()

```
void Stepper_disable (
    StepperX * stepperx )
```

Disables the stepper motor driver.

## Parameters

<i>stepperx</i>	Pointer to the <b>StepperX</b> (p. 30) structure
-----------------	--

### 10.24.2.3 Stepper\_enable()

```
void Stepper_enable (
    StepperX * stepperx )
```

Enables the stepper motor driver.

## Parameters

<i>stepperx</i>	Pointer to the <b>StepperX</b> (p. 30) structure
-----------------	--

### 10.24.2.4 Stepper\_init()

```
void Stepper_init (
    StepperX * stepperx,
    GPIO_TypeDef * GPIO,
    uint16_t EN_PIN,
    uint16_t DIR_PIN,
    uint16_t STP_PIN )
```

Initializes the stepper motor driver with the specified GPIO port and pins.

## Parameters

<i>stepperx</i>	Pointer to the <b>StepperX</b> (p. 30) structure
<i>GPIO</i>	GPIO port
<i>EN_PIN</i>	Enable pin
<i>DIR_PIN</i>	Direction pin
<i>STP_PIN</i>	Step pin

### 10.24.2.5 Stepper\_setspeed()

```
void Stepper_setspeed (
    StepperX * stepperx,
    uint16_t speed,
    uint8_t dir )
```

Sets the speed and direction of the stepper motor.

#### Parameters

<i>stepperx</i>	Pointer to the <b>StepperX</b> (p. 30) structure
<i>speed</i>	Speed of the stepper motor (delay between steps)
<i>dir</i>	Direction of the stepper motor (1 for one direction, 0 for the opposite)

< Set the direction pin

< Set the step pin high

< Delay to control the speed

< Set the step pin low

< Delay to control the speed

### 10.24.2.6 SysTick\_Init()

```
void SysTick_Init (
    void )
```

Initializes the system tick for microsecond delays.

Initializes the system tick for delay functions.

## 10.25 Ball\_Launcher\_Main/Core/Src/stm32f4xx\_hal\_msp.c File Reference

This file provides code for the MSP Initialization and de-Initialization codes.

```
#include "main.h"
```



## Functions

- void **HAL\_TIM\_MspPostInit** (TIM\_HandleTypeDef \*htim)
- void **HAL\_MspInit** (void)
- void **HAL\_TIM\_PWM\_MspInit** (TIM\_HandleTypeDef \*htim\_pwm)  
*TIM\_PWM MSP Initialization This function configures the hardware resources used in this example.*
- void **HAL\_TIM\_IC\_MspInit** (TIM\_HandleTypeDef \*htim\_ic)  
*TIM\_IC MSP Initialization This function configures the hardware resources used in this example.*
- void **HAL\_TIM\_PWM\_MspDeInit** (TIM\_HandleTypeDef \*htim\_pwm)  
*TIM\_PWM MSP De-Initialization This function freeze the hardware resources used in this example.*
- void **HAL\_TIM\_IC\_MspDeInit** (TIM\_HandleTypeDef \*htim\_ic)  
*TIM\_IC MSP De-Initialization This function freeze the hardware resources used in this example.*
- void **HAL\_UART\_MspInit** (UART\_HandleTypeDef \*huart)  
*UART MSP Initialization This function configures the hardware resources used in this example.*
- void **HAL\_UART\_MspDeInit** (UART\_HandleTypeDef \*huart)  
*UART MSP De-Initialization This function freeze the hardware resources used in this example.*

## 10.25.1 Detailed Description

This file provides code for the MSP Initialization and de-Initialization codes.

### Attention

Copyright (c) 2024 STMicroelectronics. All rights reserved.

This software is licensed under terms that can be found in the LICENSE file in the root directory of this software component. If no LICENSE file comes with this software, it is provided AS-IS.

## 10.25.2 Function Documentation

### 10.25.2.1 HAL\_MspInit()

```
void HAL_MspInit (
    void )
```

Initializes the Global MSP.

### 10.25.2.2 HAL\_TIM\_IC\_MspDeInit()

```
void HAL_TIM_IC_MspDeInit (
    TIM_HandleTypeDef * htim_ic )
```

TIM\_IC MSP De-Initialization This function freeze the hardware resources used in this example.

**Parameters**

<i>htim_</i> <i>_ic</i>	TIM_IC handle pointer
----------------------------	-----------------------

**Return values**

<i>None</i>	
-------------	--

TIM4 GPIO Configuration PB6 -----> TIM4\_CH1

**10.25.2.3 HAL\_TIM\_IC\_MspInit()**

```
void HAL_TIM_IC_MspInit (
    TIM_HandleTypeDef * htim_ic )
```

TIM\_IC MSP Initialization This function configures the hardware resources used in this example.

**Parameters**

<i>htim_</i> <i>_ic</i>	TIM_IC handle pointer
----------------------------	-----------------------

**Return values**

<i>None</i>	
-------------	--

TIM4 GPIO Configuration PB6 -----> TIM4\_CH1

**10.25.2.4 HAL\_TIM\_MspPostInit()**

```
void HAL_TIM_MspPostInit (
    TIM_HandleTypeDef * htim )
```

TIM2 GPIO Configuration PA1 -----> TIM2\_CH2 PA2 -----> TIM2\_CH3

**10.25.2.5 HAL\_TIM\_PWM\_MspDeInit()**

```
void HAL_TIM_PWM_MspDeInit (
    TIM_HandleTypeDef * htim_pwm )
```

TIM\_PWM MSP De-Initialization This function freeze the hardware resources used in this example.

**Parameters**

<i>htim_pwm</i>	TIM_PWM handle pointer
-----------------	------------------------

## Return values

<i>None</i>	
-------------	--

**10.25.2.6 HAL\_TIM\_PWM\_MspInit()**

```
void HAL_TIM_PWM_MspInit (
    TIM_HandleTypeDef * htim_pwm )
```

TIM\_PWM MSP Initialization This function configures the hardware resources used in this example.

## Parameters

<i>htim_pwm</i>	TIM_PWM handle pointer
-----------------	------------------------

## Return values

<i>None</i>	
-------------	--

**10.25.2.7 HAL\_UART\_MspDeInit()**

```
void HAL_UART_MspDeInit (
    UART_HandleTypeDef * huart )
```

UART MSP De-Initialization This function freeze the hardware resources used in this example.

## Parameters

<i>huart</i>	UART handle pointer
--------------	---------------------

## Return values

<i>None</i>	
-------------	--

USART1 GPIO Configuration PA9 -----> USART1\_TX PA10 -----> USART1\_RX

USART6 GPIO Configuration PA11 -----> USART6\_TX PA12 -----> USART6\_RX

**10.25.2.8 HAL\_UART\_MspInit()**

```
void HAL_UART_MspInit (
    UART_HandleTypeDef * huart )
```

UART MSP Initialization This function configures the hardware resources used in this example.

**Parameters**

<i>huart</i>	UART handle pointer
--------------	---------------------

**Return values**

<i>None</i>	
-------------	--

USART1 GPIO Configuration PA9 -----> USART1\_TX PA10 -----> USART1\_RX

USART6 GPIO Configuration PA11 -----> USART6\_TX PA12 -----> USART6\_RX

## 10.26 Ball\_Launcher\_Main/Core/Src/stm32f4xx\_it.c File Reference

Interrupt Service Routines.

```
#include "main.h"
#include "stm32f4xx_it.h"
```

**Functions**

- void **NMI\_Handler** (void)  
*This function handles Non maskable interrupt.*
- void **HardFault\_Handler** (void)  
*This function handles Hard fault interrupt.*
- void **MemManage\_Handler** (void)  
*This function handles Memory management fault.*
- void **BusFault\_Handler** (void)  
*This function handles Pre-fetch fault, memory access fault.*
- void **UsageFault\_Handler** (void)  
*This function handles Undefined instruction or illegal state.*
- void **SVC\_Handler** (void)  
*This function handles System service call via SWI instruction.*
- void **DebugMon\_Handler** (void)  
*This function handles Debug monitor.*
- void **PendSV\_Handler** (void)  
*This function handles Pendable request for system service.*
- void **SysTick\_Handler** (void)  
*This function handles System tick timer.*
- void **TIM4\_IRQHandler** (void)  
*This function handles TIM4 global interrupt.*
- void **USART1\_IRQHandler** (void)  
*This function handles USART1 global interrupt.*

**Variables**

- TIM\_HandleTypeDef **htim4**
- UART\_HandleTypeDef **huart1**

## 10.26.1 Detailed Description

Interrupt Service Routines.

Attention

Copyright (c) 2024 STMicroelectronics. All rights reserved.

This software is licensed under terms that can be found in the LICENSE file in the root directory of this software component. If no LICENSE file comes with this software, it is provided AS-IS.

## 10.26.2 Function Documentation

### 10.26.2.1 BusFault\_Handler()

```
void BusFault_Handler (
    void )
```

This function handles Pre-fetch fault, memory access fault.

### 10.26.2.2 DebugMon\_Handler()

```
void DebugMon_Handler (
    void )
```

This function handles Debug monitor.

### 10.26.2.3 HardFault\_Handler()

```
void HardFault_Handler (
    void )
```

This function handles Hard fault interrupt.

### 10.26.2.4 MemManage\_Handler()

```
void MemManage_Handler (
    void )
```

This function handles Memory management fault.

### 10.26.2.5 NMI\_Handler()

```
void NMI_Handler (
    void )
```

This function handles Non maskable interrupt.

#### 10.26.2.6 PendSV\_Handler()

```
void PendSV_Handler (  
    void )
```

This function handles Pendable request for system service.

#### 10.26.2.7 SVC\_Handler()

```
void SVC_Handler (  
    void )
```

This function handles System service call via SWI instruction.

#### 10.26.2.8 SysTick\_Handler()

```
void SysTick_Handler (  
    void )
```

This function handles System tick timer.

#### 10.26.2.9 TIM4\_IRQHandler()

```
void TIM4_IRQHandler (  
    void )
```

This function handles TIM4 global interrupt.

#### 10.26.2.10 UsageFault\_Handler()

```
void UsageFault_Handler (  
    void )
```

This function handles Undefined instruction or illegal state.

#### 10.26.2.11 USART1\_IRQHandler()

```
void USART1_IRQHandler (  
    void )
```

This function handles USART1 global interrupt.

### 10.26.3 Variable Documentation

#### 10.26.3.1 htim4

```
TIM_HandleTypeDef htim4 [extern]
```

### 10.26.3.2 huart1

```
UART_HandleTypeDef huart1  [extern]
```

## 10.27 Ball\_Launcher\_Main/Core/Src/switch.c File Reference

Source file for switch operations. This file contains the implementation of the functions for initializing and getting the status of a switch.

```
#include "switch.h"
#include "stm32f4xx_hal.h"
#include <stdio.h>
#include <stdint.h>
```

### Functions

- void **Switch\_init** ( **SwitchX** \*switchx, GPIO\_TypeDef \*GPIO, uint16\_t PIN)  
*Initializes the switch with the specified GPIO port and pin.*
- uint8\_t **Switch\_getStatus** ( **SwitchX** \*switchx)  
*Gets the status of the switch.*

### 10.27.1 Detailed Description

Source file for switch operations. This file contains the implementation of the functions for initializing and getting the status of a switch.

Created on: May 17, 2024

Author: vvinh

### 10.27.2 Function Documentation

#### 10.27.2.1 Switch\_getStatus()

```
uint8_t Switch_getStatus (
    SwitchX * switchx )
```

Gets the status of the switch.

#### Parameters

<i>switchx</i>	Pointer to the <b>SwitchX</b> (p. 31) structure
----------------	---

**Returns**

uint8\_t Status of the switch (1 if pressed, 0 if not pressed)

**10.27.2.2 Switch\_init()**

```
void Switch_init (
    SwitchX * switchx,
    GPIO_TypeDef * GPIO,
    uint16_t PIN )
```

Initializes the switch with the specified GPIO port and pin.

**Parameters**

<i>switchx</i>	Pointer to the <b>SwitchX</b> (p. 31) structure
<i>GPIO</i>	GPIO port
<i>PIN</i>	GPIO pin

< Assign the GPIO port

< Assign the GPIO pin

< Initialize the status to 0

**10.28 Ball\_Launcher\_Main/Core/Src/syscalls.c File Reference**

STM32CubeIDE Minimal System calls file.

```
#include <sys/stat.h>
#include <stdlib.h>
#include <errno.h>
#include <stdio.h>
#include <signal.h>
#include <time.h>
#include <sys/time.h>
#include <sys/times.h>
```

**Functions**

- int **\_\_io\_putchar** (int ch) **\_\_attribute\_\_((weak))**
- int **\_\_io\_getchar** (void)
- void **initialise\_monitor\_handles** ()
- int **\_getpid** (void)
- int **\_kill** (int pid, int sig)
- void **\_exit** (int status)
- **\_\_attribute\_\_((weak))**
- int **\_close** (int file)
- int **\_fstat** (int file, struct **stat** \*st)



- int **\_isatty** (int file)
- int **\_lseek** (int file, int ptr, int dir)
- int **\_open** (char \*path, int flags,...)
- int **\_wait** (int \*status)
- int **\_unlink** (char \*name)
- int **\_times** (struct tms \*buf)
- int **\_stat** (char \*file, struct **stat** \*st)
- int **\_link** (char \*old, char \*new)
- int **\_fork** (void)
- int **\_execve** (char \*name, char \*\*argv, char \*\*env)

## Variables

- char \*\* **environ** = \_\_env

## 10.28.1 Detailed Description

STM32CubeIDE Minimal System calls file.

### Author

Auto-generated by STM32CubeIDE

For more information about which c-functions  
need which of these lowlevel functions  
please consult the Newlib libc-manual

### Attention

Copyright (c) 2020-2024 STMicroelectronics. All rights reserved.

This software is licensed under terms that can be found in the LICENSE file in the root directory of this software component. If no LICENSE file comes with this software, it is provided AS-IS.

## 10.28.2 Function Documentation

### 10.28.2.1 \_\_attribute\_\_()

```
__attribute__ (
    (weak) )
```

### 10.28.2.2 \_\_io\_getchar()

```
int __io_getchar (
    void ) [extern]
```

### 10.28.2.3 `__io_putchar()`

```
int __io_putchar (
    int ch ) [extern]
```

### 10.28.2.4 `_close()`

```
int _close (
    int file )
```

### 10.28.2.5 `_execve()`

```
int _execve (
    char * name,
    char ** argv,
    char ** env )
```

### 10.28.2.6 `_exit()`

```
void _exit (
    int status )
```

### 10.28.2.7 `_fork()`

```
int _fork (
    void )
```

### 10.28.2.8 `_fstat()`

```
int _fstat (
    int file,
    struct stat * st )
```

### 10.28.2.9 `_getpid()`

```
int _getpid (
    void )
```

### 10.28.2.10 `_isatty()`

```
int _isatty (
    int file )
```

**10.28.2.11 \_kill()**

```
int _kill (
    int pid,
    int sig )
```

**10.28.2.12 \_link()**

```
int _link (
    char * old,
    char * new )
```

**10.28.2.13 \_lseek()**

```
int _lseek (
    int file,
    int ptr,
    int dir )
```

**10.28.2.14 \_open()**

```
int _open (
    char * path,
    int flags,
    ... )
```

**10.28.2.15 \_stat()**

```
int _stat (
    char * file,
    struct stat * st )
```

**10.28.2.16 \_times()**

```
int _times (
    struct tms * buf )
```

**10.28.2.17 \_unlink()**

```
int _unlink (
    char * name )
```

**10.28.2.18 \_wait()**

```
int _wait (
    int * status )
```

### 10.28.2.19 initialise\_monitor\_handles()

```
void initialise_monitor_handles ( )
```

## 10.28.3 Variable Documentation

### 10.28.3.1 environ

```
char** environ = __env
```

## 10.29 Ball\_Launcher\_Main/Core/Src/sysmem.c File Reference

STM32CubeIDE System Memory calls file.

```
#include <errno.h>
#include <stdint.h>
```

### Functions

- void \* **\_sbrk** (ptrdiff\_t incr)  
*\_sbrk()* (p. 123) allocates memory to the newlib heap and is used by malloc and others from the C library

### 10.29.1 Detailed Description

STM32CubeIDE System Memory calls file.

#### Author

Generated by STM32CubeIDE

```
For more information about which C functions
need which of these lowlevel functions
please consult the newlib libc manual
```

#### Attention

Copyright (c) 2024 STMicroelectronics. All rights reserved.

This software is licensed under terms that can be found in the LICENSE file in the root directory of this software component. If no LICENSE file comes with this software, it is provided AS-IS.

## 10.29.2 Function Documentation

### 10.29.2.1 \_sbrk()

```
void * _sbrk (
    ptrdiff_t incr )
```

**\_sbrk()** (p. 123) allocates memory to the newlib heap and is used by malloc and others from the C library

```
* #####
* # .data # .bss #          newlib heap          #          MSP stack          #
* #          #          #          # Reserved by _Min_Stack_Size #
* #####
* ^-- RAM start          ^-- _end          _estack, RAM end --^
*
```

This implementation starts allocating at the '\_end' linker symbol The '\_Min\_Stack\_Size' linker symbol reserves a memory for the MSP stack The implementation considers '\_estack' linker symbol to be RAM end NOTE: If the MSP stack, at any point during execution, grows larger than the reserved size, please increase the '\_Min\_Stack\_Size'.

#### Parameters

<i>incr</i>	Memory size
-------------	-------------

#### Returns

Pointer to allocated memory

## 10.30 Ball\_Launcher\_Main/Core/Src/system\_stm32f4xx.c File Reference

CMSIS Cortex-M4 Device Peripheral Access Layer System Source File.

```
#include "stm32f4xx.h"
```

#### Macros

- **#define HSE\_VALUE** ((uint32\_t)25000000)
- **#define HSI\_VALUE** ((uint32\_t)16000000)

#### Functions

- void **SystemInit** (void)  
*Setup the microcontroller system Initialize the FPU setting, vector table location and External memory configuration.*
- void **SystemCoreClockUpdate** (void)  
*Update SystemCoreClock variable according to Clock Register Values. The SystemCoreClock variable contains the core clock (HCLK), it can be used by the user application to setup the SysTick timer or configure other parameters.*

## Variables

- uint32\_t **SystemCoreClock** = 16000000
- const uint8\_t **AHBPrescTable** [16] = {0, 0, 0, 0, 0, 0, 0, 0, 1, 2, 3, 4, 6, 7, 8, 9}
- const uint8\_t **APBPrescTable** [8] = {0, 0, 0, 0, 1, 2, 3, 4}

### 10.30.1 Detailed Description

CMSIS Cortex-M4 Device Peripheral Access Layer System Source File.

#### Author

MCD Application Team

This file provides two functions and one global variable to be called from user application:

- **SystemInit()** (p. 22): This function is called at startup just after reset and before branch to main program. This call is made inside the "startup\_stm32f4xx.s" file.
- **SystemCoreClock** variable: Contains the core clock (HCLK), it can be used by the user application to setup the SysTick timer or configure other parameters.
- **SystemCoreClockUpdate()** (p. 22): Updates the variable SystemCoreClock and must be called whenever the core clock is changed during program execution.

#### Attention

Copyright (c) 2017 STMicroelectronics. All rights reserved.

This software is licensed under terms that can be found in the LICENSE file in the root directory of this software component. If no LICENSE file comes with this software, it is provided AS-IS.

## 10.31 pages/controller.c File Reference

Controller Component Documentation.

### 10.31.1 Detailed Description

Controller Component Documentation.

## 10.32 pages/launcher.c File Reference

Launcher Component Documentation.

### 10.32.1 Detailed Description

Launcher Component Documentation.

## 10.33 pages/mainpage.c File Reference

Main Page Documentation for Project.

### 10.33.1 Detailed Description

Main Page Documentation for Project.

## 10.34 pages/report.c File Reference

Launcher Documentation.

### 10.34.1 Detailed Description

Launcher Documentation.





# Index

- `__attribute__`
    - `syscalls.c`, 119
    - `syscalls_C.c`, 65
  - `__io_getchar`
    - `syscalls.c`, 119
    - `syscalls_C.c`, 65
  - `__io_putchar`
    - `syscalls.c`, 119
    - `syscalls_C.c`, 65
  - `_close`
    - `syscalls.c`, 120
    - `syscalls_C.c`, 65
  - `_execve`
    - `syscalls.c`, 120
    - `syscalls_C.c`, 65
  - `_exit`
    - `syscalls.c`, 120
    - `syscalls_C.c`, 65
  - `_fork`
    - `syscalls.c`, 120
    - `syscalls_C.c`, 66
  - `_fstat`
    - `syscalls.c`, 120
    - `syscalls_C.c`, 66
  - `_getpid`
    - `syscalls.c`, 120
    - `syscalls_C.c`, 66
  - `_isatty`
    - `syscalls.c`, 120
    - `syscalls_C.c`, 66
  - `_kill`
    - `syscalls.c`, 120
    - `syscalls_C.c`, 66
  - `_link`
    - `syscalls.c`, 121
    - `syscalls_C.c`, 66
  - `_lseek`
    - `syscalls.c`, 121
    - `syscalls_C.c`, 66
  - `_open`
    - `syscalls.c`, 121
    - `syscalls_C.c`, 66
  - `_sbrk`
    - `sysmem.c`, 123
    - `sysmem_C.c`, 68
  - `_stat`
    - `syscalls.c`, 121
    - `syscalls_C.c`, 67
  - `_times`
    - `syscalls.c`, 121
    - `syscalls_C.c`, 67
- `_unlink`
  - `syscalls.c`, 121
  - `syscalls_C.c`, 67
- `_wait`
  - `syscalls.c`, 121
  - `syscalls_C.c`, 67
- `addr`
  - `MPU6050`, 27
- `AHBPrescTable`
  - `STM32F4xx_System_Private_Variables`, 21
- `angleTarget`
  - `main.c`, 102
- `APBPrescTable`
  - `STM32F4xx_System_Private_Variables`, 21
- `assert_param`
  - `stm32f4xx_hal_conf.h`, 81
  - `stm32f4xx_hal_conf_C.h`, 40
- `Ball_Launcher_Controller/Core/Inc/main_C.h`, 33
- `Ball_Launcher_Controller/Core/Inc/mpu6050.h`, 34
- `Ball_Launcher_Controller/Core/Inc/stm32f4xx_hal_conf_C.h`, 37
- `Ball_Launcher_Controller/Core/Inc/stm32f4xx_it_C.h`, 51
- `Ball_Launcher_Controller/Core/Src/main_C.c`, 53
- `Ball_Launcher_Controller/Core/Src/mpu6050.c`, 57
- `Ball_Launcher_Controller/Core/Src/stm32f4xx_hal_msp_C.c`, 59
- `Ball_Launcher_Controller/Core/Src/stm32f4xx_it_C.c`, 62
- `Ball_Launcher_Controller/Core/Src/syscalls_C.c`, 64
- `Ball_Launcher_Controller/Core/Src/sysmem_C.c`, 67
- `Ball_Launcher_Controller/Core/Src/system_stm32f4xx_C.c`, 69
- `Ball_Launcher_Main/Core/Inc/d4215.h`, 70
- `Ball_Launcher_Main/Core/Inc/led.h`, 71
- `Ball_Launcher_Main/Core/Inc/main.h`, 73
- `Ball_Launcher_Main/Core/Inc/radio_driver.h`, 74
- `Ball_Launcher_Main/Core/Inc/stepper_driver.h`, 76
- `Ball_Launcher_Main/Core/Inc/stm32f4xx_hal_conf.h`, 79
- `Ball_Launcher_Main/Core/Inc/stm32f4xx_it.h`, 92
- `Ball_Launcher_Main/Core/Inc/switch.h`, 94
- `Ball_Launcher_Main/Core/Src/d4215.c`, 96
- `Ball_Launcher_Main/Core/Src/led.c`, 97
- `Ball_Launcher_Main/Core/Src/main.c`, 99
- `Ball_Launcher_Main/Core/Src/radio_driver.c`, 106

- Ball\_Launcher\_Main/Core/Src/stepper\_driver.c, 108
- Ball\_Launcher\_Main/Core/Src/stm32f4xx\_hal\_msp.c, 110
- Ball\_Launcher\_Main/Core/Src/stm32f4xx\_it.c, 114
- Ball\_Launcher\_Main/Core/Src/switch.c, 117
- Ball\_Launcher\_Main/Core/Src/syscalls.c, 118
- Ball\_Launcher\_Main/Core/Src/sysmem.c, 122
- Ball\_Launcher\_Main/Core/Src/system\_stm32f4xx.c, 123
- buffer
  - main\_C.c, 55
- BusFault\_Handler
  - stm32f4xx\_it.c, 115
  - stm32f4xx\_it.h, 93
  - stm32f4xx\_it\_C.c, 62
  - stm32f4xx\_it\_C.h, 52
- CAL
  - main.c, 102
- capturePulseWidth
  - radio\_driver.c, 106
  - radio\_driver.h, 74
- channel
  - D4215X, 25
- channel1
  - RadioX, 29
- channel2
  - RadioX, 29
- charIn
  - main.c, 102
- CMSIS, 19
- CONTROLLER, 5
- counter1
  - RadioX, 29
- counter2
  - RadioX, 29
- d4215.c
  - D4215\_init, 96
  - D4215\_set, 97
- d4215.h
  - D4215\_init, 70
  - D4215\_set, 70
- D4215\_init
  - d4215.c, 96
  - d4215.h, 70
- D4215\_set
  - d4215.c, 97
  - d4215.h, 70
- D4215X, 25
  - channel, 25
  - speed, 25
  - timer, 25
- data
  - main.c, 102
- DATA\_CACHE\_ENABLE
  - stm32f4xx\_hal\_conf.h, 81
  - stm32f4xx\_hal\_conf\_C.h, 40
- dataProcess
  - main.c, 101
- DebugMon\_Handler
  - stm32f4xx\_it.c, 115
  - stm32f4xx\_it.h, 93
  - stm32f4xx\_it\_C.c, 62
  - stm32f4xx\_it\_C.h, 52
- Delay\_us
  - stepper\_driver.c, 108
  - stepper\_driver.h, 77
- DIR\_PIN
  - StepperX, 30
- DP83848\_PHY\_ADDRESS
  - stm32f4xx\_hal\_conf.h, 81
  - stm32f4xx\_hal\_conf\_C.h, 40
- dt
  - main.c, 102
  - MPU6050, 27
- EN\_PIN
  - StepperX, 30
- environ
  - syscalls.c, 122
  - syscalls\_C.c, 67
- Error\_Handler
  - main.c, 101
  - main.h, 73
  - main\_C.c, 54
  - main\_C.h, 33
- ESTOP
  - main.c, 103
- ETH\_RX\_BUF\_SIZE
  - stm32f4xx\_hal\_conf.h, 81
  - stm32f4xx\_hal\_conf\_C.h, 40
- ETH\_RXBUFNB
  - stm32f4xx\_hal\_conf.h, 81
  - stm32f4xx\_hal\_conf\_C.h, 40
- ETH\_TX\_BUF\_SIZE
  - stm32f4xx\_hal\_conf.h, 81
  - stm32f4xx\_hal\_conf\_C.h, 41
- ETH\_TXBUFNB
  - stm32f4xx\_hal\_conf.h, 82
  - stm32f4xx\_hal\_conf\_C.h, 41
- EXTERNAL\_CLOCK\_VALUE
  - stm32f4xx\_hal\_conf.h, 82
  - stm32f4xx\_hal\_conf\_C.h, 41
- fall
  - main.c, 103
- GPIOx
  - LEDX, 26
  - StepperX, 31
  - SwitchX, 31
- gX
  - main\_C.c, 55
  - MPU6050, 27
- gX\_offset
  - MPU6050, 27
- gY

- main\_C.c, 55
- MPU6050, 27
- gY\_offset
  - MPU6050, 28
- gZ
  - main\_C.c, 55
  - MPU6050, 28
- gZ\_offset
  - MPU6050, 28
- HAL\_CORTEX\_MODULE\_ENABLED
  - stm32f4xx\_hal\_conf.h, 82
  - stm32f4xx\_hal\_conf\_C.h, 41
- HAL\_DMA\_MODULE\_ENABLED
  - stm32f4xx\_hal\_conf.h, 82
  - stm32f4xx\_hal\_conf\_C.h, 41
- HAL\_EXTI\_MODULE\_ENABLED
  - stm32f4xx\_hal\_conf.h, 82
  - stm32f4xx\_hal\_conf\_C.h, 41
- HAL\_FLASH\_MODULE\_ENABLED
  - stm32f4xx\_hal\_conf.h, 82
  - stm32f4xx\_hal\_conf\_C.h, 41
- HAL\_GPIO\_MODULE\_ENABLED
  - stm32f4xx\_hal\_conf.h, 82
  - stm32f4xx\_hal\_conf\_C.h, 41
- HAL\_I2C\_MODULE\_ENABLED
  - stm32f4xx\_hal\_conf\_C.h, 41
- HAL\_I2C\_MspDeInit
  - stm32f4xx\_hal\_msp\_C.c, 60
- HAL\_I2C\_MspInit
  - stm32f4xx\_hal\_msp\_C.c, 60
- HAL\_MODULE\_ENABLED
  - stm32f4xx\_hal\_conf.h, 82
  - stm32f4xx\_hal\_conf\_C.h, 42
- HAL\_MspInit
  - stm32f4xx\_hal\_msp.c, 111
  - stm32f4xx\_hal\_msp\_C.c, 61
- HAL\_PWR\_MODULE\_ENABLED
  - stm32f4xx\_hal\_conf.h, 82
  - stm32f4xx\_hal\_conf\_C.h, 42
- HAL\_RCC\_MODULE\_ENABLED
  - stm32f4xx\_hal\_conf.h, 83
  - stm32f4xx\_hal\_conf\_C.h, 42
- HAL\_TIM\_IC\_CaptureCallback
  - main.c, 101
- HAL\_TIM\_IC\_MspDeInit
  - stm32f4xx\_hal\_msp.c, 111
- HAL\_TIM\_IC\_MspInit
  - stm32f4xx\_hal\_msp.c, 112
- HAL\_TIM\_MODULE\_ENABLED
  - stm32f4xx\_hal\_conf.h, 83
- HAL\_TIM\_MspPostInit
  - main.h, 74
  - stm32f4xx\_hal\_msp.c, 112
- HAL\_TIM\_PWM\_MspDeInit
  - stm32f4xx\_hal\_msp.c, 112
- HAL\_TIM\_PWM\_MspInit
  - stm32f4xx\_hal\_msp.c, 113
- HAL\_UART\_MODULE\_ENABLED
  - stm32f4xx\_hal\_conf.h, 83
  - stm32f4xx\_hal\_conf\_C.h, 42
- HAL\_UART\_MspDeInit
  - stm32f4xx\_hal\_msp.c, 113
  - stm32f4xx\_hal\_msp\_C.c, 61
- HAL\_UART\_MspInit
  - stm32f4xx\_hal\_msp.c, 113
  - stm32f4xx\_hal\_msp\_C.c, 61
- HAL\_UART\_RxCpltCallback
  - main.c, 101
- HardFault\_Handler
  - stm32f4xx\_it.c, 115
  - stm32f4xx\_it.h, 93
  - stm32f4xx\_it\_C.c, 63
  - stm32f4xx\_it\_C.h, 52
- hi2c
  - MPU6050, 28
- hi2c1
  - main\_C.c, 55
- HOME
  - main.c, 103
- HSE\_STARTUP\_TIMEOUT
  - stm32f4xx\_hal\_conf.h, 83
  - stm32f4xx\_hal\_conf\_C.h, 42
- HSE\_VALUE
  - stm32f4xx\_hal\_conf.h, 83
  - stm32f4xx\_hal\_conf\_C.h, 42
  - STM32F4xx\_System\_Private\_Includes, 20
- HSI\_VALUE
  - stm32f4xx\_hal\_conf.h, 83
  - stm32f4xx\_hal\_conf\_C.h, 42
  - STM32F4xx\_System\_Private\_Includes, 20
- htim2
  - main.c, 103
- htim4
  - main.c, 103
  - stm32f4xx\_it.c, 116
- huart1
  - main.c, 103
  - main\_C.c, 55
  - stm32f4xx\_it.c, 116
- huart2
  - main\_C.c, 56
- huart6
  - main.c, 103
- idx
  - main.c, 103
- initialise\_monitor\_handles
  - syscalls.c, 121
  - syscalls\_C.c, 67
- INSTRUCTION\_CACHE\_ENABLE
  - stm32f4xx\_hal\_conf.h, 83
  - stm32f4xx\_hal\_conf\_C.h, 42
- LAUNCHER, 7
- led.c
  - LED\_init, 98
  - LED\_off, 98

- LED\_on, 98
- LED\_toggle, 99
- led.h
  - LED\_init, 72
  - LED\_off, 72
  - LED\_on, 72
  - LED\_toggle, 72
- LED\_init
  - led.c, 98
  - led.h, 72
- LED\_off
  - led.c, 98
  - led.h, 72
- LED\_on
  - led.c, 98
  - led.h, 72
- LED\_toggle
  - led.c, 99
  - led.h, 72
- LEDX, 26
  - GPIOx, 26
  - PIN, 26
- LSE\_STARTUP\_TIMEOUT
  - stm32f4xx\_hal\_conf.h, 83
  - stm32f4xx\_hal\_conf\_C.h, 43
- LSE\_VALUE
  - stm32f4xx\_hal\_conf.h, 84
  - stm32f4xx\_hal\_conf\_C.h, 43
- LSI\_VALUE
  - stm32f4xx\_hal\_conf.h, 84
  - stm32f4xx\_hal\_conf\_C.h, 43
- MAC\_ADDR0
  - stm32f4xx\_hal\_conf.h, 84
  - stm32f4xx\_hal\_conf\_C.h, 43
- MAC\_ADDR1
  - stm32f4xx\_hal\_conf.h, 84
  - stm32f4xx\_hal\_conf\_C.h, 43
- MAC\_ADDR2
  - stm32f4xx\_hal\_conf.h, 84
  - stm32f4xx\_hal\_conf\_C.h, 43
- MAC\_ADDR3
  - stm32f4xx\_hal\_conf.h, 84
  - stm32f4xx\_hal\_conf\_C.h, 43
- MAC\_ADDR4
  - stm32f4xx\_hal\_conf.h, 84
  - stm32f4xx\_hal\_conf\_C.h, 43
- MAC\_ADDR5
  - stm32f4xx\_hal\_conf.h, 84
  - stm32f4xx\_hal\_conf\_C.h, 44
- main
  - main.c, 101
  - main\_C.c, 54
- MAIN PAGE, 1
- main.c
  - angleTarget, 102
  - CAL, 102
  - charIn, 102
  - data, 102
  - dataProcess, 101
  - dt, 102
  - Error\_Handler, 101
  - ESTOP, 103
  - fall, 103
  - HAL\_TIM\_IC\_CaptureCallback, 101
  - HAL\_UART\_RxCpltCallback, 101
  - HOME, 103
  - htim2, 103
  - htim4, 103
  - huart1, 103
  - huart6, 103
  - idx, 103
  - main, 101
  - map, 101
  - MOVE, 103
  - pitch, 103
  - pitchDirection, 104
  - pitchDirectionSW, 104
  - rise, 104
  - SHOT, 104
  - speedCalculate, 102
  - STATE, 104
  - STATE\_0\_INIT, 104
  - STATE\_1\_HUB, 104
  - STATE\_2\_ESTOP, 104
  - STATE\_3\_STEPPER, 104
  - STATE\_4\_BLDC, 104
  - SW, 105
  - SW1, 105
  - SW2, 105
  - SW3, 105
  - SystemClock\_Config, 102
  - tf, 105
  - ti, 105
  - total, 105
  - yaw, 105
  - yawDirection, 105
  - yawDirectionSW, 105
- main.h
  - Error\_Handler, 73
  - HAL\_TIM\_MspPostInit, 74
- main\_C.c
  - buffer, 55
  - Error\_Handler, 54
  - gX, 55
  - gY, 55
  - gZ, 55
  - hi2c1, 55
  - huart1, 55
  - huart2, 56
  - main, 54
  - move, 56
  - shot, 56
  - stat, 56
  - STATE, 56
  - STATE\_0\_INIT, 56
  - STATE\_1\_ERROR, 56

- STATE\_2\_IMU, 56
- STATE\_3\_BUTTON\_LED, 56
- STATE\_4\_TRANSFER, 56
- SystemClock\_Config, 55
- main\_C.h
  - Error\_Handler, 33
- map
  - main.c, 101
- MemManage\_Handler
  - stm32f4xx\_it.c, 115
  - stm32f4xx\_it.h, 93
  - stm32f4xx\_it\_C.c, 63
  - stm32f4xx\_it\_C.h, 52
- MOVE
  - main.c, 103
- move
  - main\_C.c, 56
- MPU6050, 27
  - addr, 27
  - dt, 27
  - gX, 27
  - gX\_offset, 27
  - gY, 27
  - gY\_offset, 28
  - gZ, 28
  - gZ\_offset, 28
  - hi2c, 28
  - status, 28
- mpu6050.c
  - mpu6050\_calibrate, 57
  - mpu6050\_get\_gX, 57
  - mpu6050\_get\_gY, 58
  - mpu6050\_get\_gZ, 58
  - mpu6050\_init, 58
  - mpu6050\_update, 59
- mpu6050.h
  - mpu6050\_calibrate, 34
  - mpu6050\_get\_gX, 35
  - mpu6050\_get\_gY, 35
  - mpu6050\_get\_gZ, 35
  - mpu6050\_get\_X, 36
  - mpu6050\_get\_Y, 36
  - mpu6050\_get\_Z, 36
  - mpu6050\_init, 37
  - mpu6050\_update, 37
- mpu6050\_calibrate
  - mpu6050.c, 57
  - mpu6050.h, 34
- mpu6050\_get\_gX
  - mpu6050.c, 57
  - mpu6050.h, 35
- mpu6050\_get\_gY
  - mpu6050.c, 58
  - mpu6050.h, 35
- mpu6050\_get\_gZ
  - mpu6050.c, 58
  - mpu6050.h, 35
- mpu6050\_get\_X
  - mpu6050.h, 36
- mpu6050\_get\_Y
  - mpu6050.h, 36
- mpu6050\_get\_Z
  - mpu6050.h, 36
- mpu6050\_init
  - mpu6050.c, 58
  - mpu6050.h, 37
- mpu6050\_update
  - mpu6050.c, 59
  - mpu6050.h, 37
- NMI\_Handler
  - stm32f4xx\_it.c, 115
  - stm32f4xx\_it.h, 93
  - stm32f4xx\_it\_C.c, 63
  - stm32f4xx\_it\_C.h, 52
- pages/controller.c, 124
- pages/launcher.c, 124
- pages/mainpage.c, 125
- pages/report.c, 125
- PendSV\_Handler
  - stm32f4xx\_it.c, 115
  - stm32f4xx\_it.h, 93
  - stm32f4xx\_it\_C.c, 63
  - stm32f4xx\_it\_C.h, 52
- PHY\_AUTONEGO\_COMPLETE
  - stm32f4xx\_hal\_conf.h, 85
  - stm32f4xx\_hal\_conf\_C.h, 44
- PHY\_AUTONEGOTIATION
  - stm32f4xx\_hal\_conf.h, 85
  - stm32f4xx\_hal\_conf\_C.h, 44
- PHY\_BCR
  - stm32f4xx\_hal\_conf.h, 85
  - stm32f4xx\_hal\_conf\_C.h, 44
- PHY\_BSR
  - stm32f4xx\_hal\_conf.h, 85
  - stm32f4xx\_hal\_conf\_C.h, 44
- PHY\_CONFIG\_DELAY
  - stm32f4xx\_hal\_conf.h, 85
  - stm32f4xx\_hal\_conf\_C.h, 44
- PHY\_DUPLEX\_STATUS
  - stm32f4xx\_hal\_conf.h, 85
  - stm32f4xx\_hal\_conf\_C.h, 44
- PHY\_FULLDUPLEX\_100M
  - stm32f4xx\_hal\_conf.h, 85
  - stm32f4xx\_hal\_conf\_C.h, 44
- PHY\_FULLDUPLEX\_10M
  - stm32f4xx\_hal\_conf.h, 86
  - stm32f4xx\_hal\_conf\_C.h, 45
- PHY\_HALFDUPLEX\_100M
  - stm32f4xx\_hal\_conf.h, 86
  - stm32f4xx\_hal\_conf\_C.h, 45
- PHY\_HALFDUPLEX\_10M
  - stm32f4xx\_hal\_conf.h, 86
  - stm32f4xx\_hal\_conf\_C.h, 45
- PHY\_ISOLATE
  - stm32f4xx\_hal\_conf.h, 86

- stm32f4xx\_hal\_conf\_C.h, 45
- PHY\_JABBER\_DETECTION
  - stm32f4xx\_hal\_conf.h, 86
  - stm32f4xx\_hal\_conf\_C.h, 45
- PHY\_LINKED\_STATUS
  - stm32f4xx\_hal\_conf.h, 86
  - stm32f4xx\_hal\_conf\_C.h, 45
- PHY\_LOOPBACK
  - stm32f4xx\_hal\_conf.h, 86
  - stm32f4xx\_hal\_conf\_C.h, 45
- PHY\_POWERDOWN
  - stm32f4xx\_hal\_conf.h, 87
  - stm32f4xx\_hal\_conf\_C.h, 46
- PHY\_READ\_TO
  - stm32f4xx\_hal\_conf.h, 87
  - stm32f4xx\_hal\_conf\_C.h, 46
- PHY\_RESET
  - stm32f4xx\_hal\_conf.h, 87
  - stm32f4xx\_hal\_conf\_C.h, 46
- PHY\_RESET\_DELAY
  - stm32f4xx\_hal\_conf.h, 87
  - stm32f4xx\_hal\_conf\_C.h, 46
- PHY\_RESTART\_AUTONEGOTIATION
  - stm32f4xx\_hal\_conf.h, 87
  - stm32f4xx\_hal\_conf\_C.h, 46
- PHY\_SPEED\_STATUS
  - stm32f4xx\_hal\_conf.h, 87
  - stm32f4xx\_hal\_conf\_C.h, 46
- PHY\_SR
  - stm32f4xx\_hal\_conf.h, 87
  - stm32f4xx\_hal\_conf\_C.h, 46
- PHY\_WRITE\_TO
  - stm32f4xx\_hal\_conf.h, 87
  - stm32f4xx\_hal\_conf\_C.h, 46
- PIN
  - LEDX, 26
  - SwitchX, 31
- pitch
  - main.c, 103
- pitchDirection
  - main.c, 104
- pitchDirectionSW
  - main.c, 104
- PREFETCH\_ENABLE
  - stm32f4xx\_hal\_conf.h, 88
  - stm32f4xx\_hal\_conf\_C.h, 47
- pulseWidth
  - RadioX, 29
- radio\_driver.c
  - capturePulseWidth, 106
  - Radio\_getPulseWidth, 107
  - Radio\_init, 107
  - update, 107
- radio\_driver.h
  - capturePulseWidth, 74
  - Radio\_getPulseWidth, 75
  - Radio\_init, 75
  - update, 76
- Radio\_getPulseWidth
  - radio\_driver.c, 107
  - radio\_driver.h, 75
- Radio\_init
  - radio\_driver.c, 107
  - radio\_driver.h, 75
- RadioX, 28
  - channel1, 29
  - channel2, 29
  - counter1, 29
  - counter2, 29
  - pulseWidth, 29
  - sum1, 29
  - sum2, 30
  - timer, 30
- REPORTS, 11
- rise
  - main.c, 104
- SHOT
  - main.c, 104
- shot
  - main\_C.c, 56
- speed
  - D4215X, 25
- speedCalculate
  - main.c, 102
- stat
  - main\_C.c, 56
- STATE
  - main.c, 104
  - main\_C.c, 56
- STATE\_0\_INIT
  - main.c, 104
  - main\_C.c, 56
- STATE\_1\_ERROR
  - main\_C.c, 56
- STATE\_1\_HUB
  - main.c, 104
- STATE\_2\_ESTOP
  - main.c, 104
- STATE\_2\_IMU
  - main\_C.c, 56
- STATE\_3\_BUTTON\_LED
  - main\_C.c, 56
- STATE\_3\_STEPPER
  - main.c, 104
- STATE\_4\_BLDC
  - main.c, 104
- STATE\_4\_TRANSFER
  - main\_C.c, 56
- status
  - MPU6050, 28
  - SwitchX, 32
- Stepper\_disable
  - stepper\_driver.c, 109
  - stepper\_driver.h, 77
- stepper\_driver.c
  - Delay\_us, 108

- Stepper\_disable, 109
- Stepper\_enable, 109
- Stepper\_init, 109
- Stepper\_setspeed, 109
- SysTick\_Init, 110
- stepper\_driver.h
  - Delay\_us, 77
  - Stepper\_disable, 77
  - Stepper\_enable, 77
  - Stepper\_init, 77
  - Stepper\_setspeed, 78
  - SysTick\_Init, 78
- Stepper\_enable
  - stepper\_driver.c, 109
  - stepper\_driver.h, 77
- Stepper\_init
  - stepper\_driver.c, 109
  - stepper\_driver.h, 77
- Stepper\_setspeed
  - stepper\_driver.c, 109
  - stepper\_driver.h, 78
- StepperX, 30
  - DIR\_PIN, 30
  - EN\_PIN, 30
  - GPIOx, 31
  - STP\_PIN, 31
- stm32f4xx\_hal\_conf.h
  - assert\_param, 81
  - DATA\_CACHE\_ENABLE, 81
  - DP83848\_PHY\_ADDRESS, 81
  - ETH\_RX\_BUF\_SIZE, 81
  - ETH\_RXBUFNB, 81
  - ETH\_TX\_BUF\_SIZE, 81
  - ETH\_TXBUFNB, 82
  - EXTERNAL\_CLOCK\_VALUE, 82
  - HAL\_CORTEX\_MODULE\_ENABLED, 82
  - HAL\_DMA\_MODULE\_ENABLED, 82
  - HAL\_EXTI\_MODULE\_ENABLED, 82
  - HAL\_FLASH\_MODULE\_ENABLED, 82
  - HAL\_GPIO\_MODULE\_ENABLED, 82
  - HAL\_MODULE\_ENABLED, 82
  - HAL\_PWR\_MODULE\_ENABLED, 82
  - HAL\_RCC\_MODULE\_ENABLED, 83
  - HAL\_TIM\_MODULE\_ENABLED, 83
  - HAL\_UART\_MODULE\_ENABLED, 83
  - HSE\_STARTUP\_TIMEOUT, 83
  - HSE\_VALUE, 83
  - HSI\_VALUE, 83
  - INSTRUCTION\_CACHE\_ENABLE, 83
  - LSE\_STARTUP\_TIMEOUT, 83
  - LSE\_VALUE, 84
  - LSI\_VALUE, 84
  - MAC\_ADDR0, 84
  - MAC\_ADDR1, 84
  - MAC\_ADDR2, 84
  - MAC\_ADDR3, 84
  - MAC\_ADDR4, 84
  - MAC\_ADDR5, 84
  - PHY\_AUTONEGO\_COMPLETE, 85
  - PHY\_AUTONEGOTIATION, 85
  - PHY\_BCR, 85
  - PHY\_BSR, 85
  - PHY\_CONFIG\_DELAY, 85
  - PHY\_DUPLEX\_STATUS, 85
  - PHY\_FULLDUPLEX\_100M, 85
  - PHY\_FULLDUPLEX\_10M, 86
  - PHY\_HALFDUPLEX\_100M, 86
  - PHY\_HALFDUPLEX\_10M, 86
  - PHY\_ISOLATE, 86
  - PHY\_JABBER\_DETECTION, 86
  - PHY\_LINKED\_STATUS, 86
  - PHY\_LOOPBACK, 86
  - PHY\_POWERDOWN, 87
  - PHY\_READ\_TO, 87
  - PHY\_RESET, 87
  - PHY\_RESET\_DELAY, 87
  - PHY\_RESTART\_AUTONEGOTIATION, 87
  - PHY\_SPEED\_STATUS, 87
  - PHY\_SR, 87
  - PHY\_WRITE\_TO, 87
  - PREFETCH\_ENABLE, 88
  - TICK\_INT\_PRIORITY, 88
  - USE\_HAL\_ADC\_REGISTER\_CALLBACKS, 88
  - USE\_HAL\_CAN\_REGISTER\_CALLBACKS, 88
  - USE\_HAL\_CEC\_REGISTER\_CALLBACKS, 88
  - USE\_HAL\_Cryp\_REGISTER\_CALLBACKS, 88
  - USE\_HAL\_DAC\_REGISTER\_CALLBACKS, 88
  - USE\_HAL\_DCMI\_REGISTER\_CALLBACKS, 88
  - USE\_HAL\_DFSDM\_REGISTER\_CALLBACKS, 88
  - USE\_HAL\_DMA2D\_REGISTER\_CALLBACKS, 88
  - USE\_HAL\_DSI\_REGISTER\_CALLBACKS, 89
  - USE\_HAL\_ETH\_REGISTER\_CALLBACKS, 89
  - USE\_HAL\_FMPI2C\_REGISTER\_CALLBACKS, 89
  - USE\_HAL\_FMPSMBUS\_REGISTER\_CALLBACKS, 89
  - USE\_HAL\_HASH\_REGISTER\_CALLBACKS, 89
  - USE\_HAL\_HCD\_REGISTER\_CALLBACKS, 89
  - USE\_HAL\_I2C\_REGISTER\_CALLBACKS, 89
  - USE\_HAL\_I2S\_REGISTER\_CALLBACKS, 89
  - USE\_HAL\_IRDA\_REGISTER\_CALLBACKS, 89
  - USE\_HAL\_LPTIM\_REGISTER\_CALLBACKS, 89
  - USE\_HAL\_LTDC\_REGISTER\_CALLBACKS, 90
  - USE\_HAL\_MMC\_REGISTER\_CALLBACKS, 90
  - USE\_HAL\_NAND\_REGISTER\_CALLBACKS, 90
  - USE\_HAL\_NOR\_REGISTER\_CALLBACKS, 90
  - USE\_HAL\_PCCARD\_REGISTER\_CALLBACKS, 90
  - USE\_HAL\_PCD\_REGISTER\_CALLBACKS, 90
  - USE\_HAL\_QSPI\_REGISTER\_CALLBACKS, 90
  - USE\_HAL\_RNG\_REGISTER\_CALLBACKS, 90
  - USE\_HAL\_RTC\_REGISTER\_CALLBACKS, 90
  - USE\_HAL\_SAI\_REGISTER\_CALLBACKS, 90
  - USE\_HAL\_SD\_REGISTER\_CALLBACKS, 91
  - USE\_HAL\_SDRAM\_REGISTER\_CALLBACKS, 91
  - USE\_HAL\_SMARTCARD\_REGISTER\_CALLBACKS, 91

- USE\_HAL\_SMBUS\_REGISTER\_CALLBACKS, 91
- USE\_HAL\_SPDIFRX\_REGISTER\_CALLBACKS, 91
- USE\_HAL\_SPI\_REGISTER\_CALLBACKS, 91
- USE\_HAL\_SRAM\_REGISTER\_CALLBACKS, 91
- USE\_HAL\_TIM\_REGISTER\_CALLBACKS, 91
- USE\_HAL\_UART\_REGISTER\_CALLBACKS, 91
- USE\_HAL\_USART\_REGISTER\_CALLBACKS, 91
- USE\_HAL\_WWDG\_REGISTER\_CALLBACKS, 92
- USE\_RTOS, 92
- USE\_SPI\_CRC, 92
- VDD\_VALUE, 92
- stm32f4xx\_hal\_conf\_C.h
  - assert\_param, 40
  - DATA\_CACHE\_ENABLE, 40
  - DP83848\_PHY\_ADDRESS, 40
  - ETH\_RX\_BUF\_SIZE, 40
  - ETH\_RXBUFNB, 40
  - ETH\_TX\_BUF\_SIZE, 41
  - ETH\_TXBUFNB, 41
  - EXTERNAL\_CLOCK\_VALUE, 41
  - HAL\_CORTEX\_MODULE\_ENABLED, 41
  - HAL\_DMA\_MODULE\_ENABLED, 41
  - HAL\_EXTI\_MODULE\_ENABLED, 41
  - HAL\_FLASH\_MODULE\_ENABLED, 41
  - HAL\_GPIO\_MODULE\_ENABLED, 41
  - HAL\_I2C\_MODULE\_ENABLED, 41
  - HAL\_MODULE\_ENABLED, 42
  - HAL\_PWR\_MODULE\_ENABLED, 42
  - HAL\_RCC\_MODULE\_ENABLED, 42
  - HAL\_UART\_MODULE\_ENABLED, 42
  - HSE\_STARTUP\_TIMEOUT, 42
  - HSE\_VALUE, 42
  - HSI\_VALUE, 42
  - INSTRUCTION\_CACHE\_ENABLE, 42
  - LSE\_STARTUP\_TIMEOUT, 43
  - LSE\_VALUE, 43
  - LSI\_VALUE, 43
  - MAC\_ADDR0, 43
  - MAC\_ADDR1, 43
  - MAC\_ADDR2, 43
  - MAC\_ADDR3, 43
  - MAC\_ADDR4, 43
  - MAC\_ADDR5, 44
  - PHY\_AUTONEGO\_COMPLETE, 44
  - PHY\_AUTONEGOTIATION, 44
  - PHY\_BCR, 44
  - PHY\_BSR, 44
  - PHY\_CONFIG\_DELAY, 44
  - PHY\_DUPLEX\_STATUS, 44
  - PHY\_FULLDUPLEX\_100M, 44
  - PHY\_FULLDUPLEX\_10M, 45
  - PHY\_HALFDUPLEX\_100M, 45
  - PHY\_HALFDUPLEX\_10M, 45
  - PHY\_ISOLATE, 45
  - PHY\_JABBER\_DETECTION, 45
  - PHY\_LINKED\_STATUS, 45
  - PHY\_LOOPBACK, 45
  - PHY\_POWERDOWN, 46
  - PHY\_READ\_TO, 46
  - PHY\_RESET, 46
  - PHY\_RESET\_DELAY, 46
  - PHY\_RESTART\_AUTONEGOTIATION, 46
  - PHY\_SPEED\_STATUS, 46
  - PHY\_SR, 46
  - PHY\_WRITE\_TO, 46
  - PREFETCH\_ENABLE, 47
  - TICK\_INT\_PRIORITY, 47
  - USE\_HAL\_ADC\_REGISTER\_CALLBACKS, 47
  - USE\_HAL\_CAN\_REGISTER\_CALLBACKS, 47
  - USE\_HAL\_CEC\_REGISTER\_CALLBACKS, 47
  - USE\_HAL\_Cryp\_REGISTER\_CALLBACKS, 47
  - USE\_HAL\_DAC\_REGISTER\_CALLBACKS, 47
  - USE\_HAL\_DCMI\_REGISTER\_CALLBACKS, 47
  - USE\_HAL\_DFSDM\_REGISTER\_CALLBACKS, 47
  - USE\_HAL\_DMA2D\_REGISTER\_CALLBACKS, 47
  - USE\_HAL\_DSI\_REGISTER\_CALLBACKS, 48
  - USE\_HAL\_ETH\_REGISTER\_CALLBACKS, 48
  - USE\_HAL\_FMPI2C\_REGISTER\_CALLBACKS, 48
  - USE\_HAL\_FMPsMBUS\_REGISTER\_CALLBACKS, 48
  - USE\_HAL\_HASH\_REGISTER\_CALLBACKS, 48
  - USE\_HAL\_HCD\_REGISTER\_CALLBACKS, 48
  - USE\_HAL\_I2C\_REGISTER\_CALLBACKS, 48
  - USE\_HAL\_I2S\_REGISTER\_CALLBACKS, 48
  - USE\_HAL\_IRDA\_REGISTER\_CALLBACKS, 48
  - USE\_HAL\_LPTIM\_REGISTER\_CALLBACKS, 48
  - USE\_HAL\_LTDC\_REGISTER\_CALLBACKS, 49
  - USE\_HAL\_MMC\_REGISTER\_CALLBACKS, 49
  - USE\_HAL\_NAND\_REGISTER\_CALLBACKS, 49
  - USE\_HAL\_NOR\_REGISTER\_CALLBACKS, 49
  - USE\_HAL\_PCCARD\_REGISTER\_CALLBACKS, 49
  - USE\_HAL\_PCD\_REGISTER\_CALLBACKS, 49
  - USE\_HAL\_QSPI\_REGISTER\_CALLBACKS, 49
  - USE\_HAL\_RNG\_REGISTER\_CALLBACKS, 49
  - USE\_HAL\_RTC\_REGISTER\_CALLBACKS, 49
  - USE\_HAL\_SAI\_REGISTER\_CALLBACKS, 49
  - USE\_HAL\_SD\_REGISTER\_CALLBACKS, 50
  - USE\_HAL\_SDRAM\_REGISTER\_CALLBACKS, 50
  - USE\_HAL\_SMARTCARD\_REGISTER\_CALLBACKS, 50
  - USE\_HAL\_SMBUS\_REGISTER\_CALLBACKS, 50
  - USE\_HAL\_SPDIFRX\_REGISTER\_CALLBACKS, 50
  - USE\_HAL\_SPI\_REGISTER\_CALLBACKS, 50
  - USE\_HAL\_SRAM\_REGISTER\_CALLBACKS, 50
  - USE\_HAL\_TIM\_REGISTER\_CALLBACKS, 50
  - USE\_HAL\_UART\_REGISTER\_CALLBACKS, 50
  - USE\_HAL\_USART\_REGISTER\_CALLBACKS, 50
  - USE\_HAL\_WWDG\_REGISTER\_CALLBACKS, 51
  - USE\_RTOS, 51
  - USE\_SPI\_CRC, 51
  - VDD\_VALUE, 51
- stm32f4xx\_hal\_msp.c
  - HAL\_MspInit, 111



- HAL\_TIM\_IC\_MspDeInit, 111
- HAL\_TIM\_IC\_MspInit, 112
- HAL\_TIM\_MspPostInit, 112
- HAL\_TIM\_PWM\_MspDeInit, 112
- HAL\_TIM\_PWM\_MspInit, 113
- HAL\_UART\_MspDeInit, 113
- HAL\_UART\_MspInit, 113
- stm32f4xx\_hal\_msp.C.c
  - HAL\_I2C\_MspDeInit, 60
  - HAL\_I2C\_MspInit, 60
  - HAL\_MspInit, 61
  - HAL\_UART\_MspDeInit, 61
  - HAL\_UART\_MspInit, 61
- stm32f4xx\_it.c
  - BusFault\_Handler, 115
  - DebugMon\_Handler, 115
  - HardFault\_Handler, 115
  - htim4, 116
  - huart1, 116
  - MemManage\_Handler, 115
  - NMI\_Handler, 115
  - PendSV\_Handler, 115
  - SVC\_Handler, 116
  - SysTick\_Handler, 116
  - TIM4\_IRQHandler, 116
  - UsageFault\_Handler, 116
  - USART1\_IRQHandler, 116
- stm32f4xx\_it.h
  - BusFault\_Handler, 93
  - DebugMon\_Handler, 93
  - HardFault\_Handler, 93
  - MemManage\_Handler, 93
  - NMI\_Handler, 93
  - PendSV\_Handler, 93
  - SVC\_Handler, 94
  - SysTick\_Handler, 94
  - TIM4\_IRQHandler, 94
  - UsageFault\_Handler, 94
  - USART1\_IRQHandler, 94
- stm32f4xx\_it\_C.c
  - BusFault\_Handler, 62
  - DebugMon\_Handler, 62
  - HardFault\_Handler, 63
  - MemManage\_Handler, 63
  - NMI\_Handler, 63
  - PendSV\_Handler, 63
  - SVC\_Handler, 63
  - SysTick\_Handler, 63
  - UsageFault\_Handler, 63
- stm32f4xx\_it\_C.h
  - BusFault\_Handler, 52
  - DebugMon\_Handler, 52
  - HardFault\_Handler, 52
  - MemManage\_Handler, 52
  - NMI\_Handler, 52
  - PendSV\_Handler, 52
  - SVC\_Handler, 53
  - SysTick\_Handler, 53
  - UsageFault\_Handler, 53
- Stm32f4xx\_system, 19
- STM32F4xx\_System\_Private\_Defines, 20
- STM32F4xx\_System\_Private\_FunctionPrototypes, 21
- STM32F4xx\_System\_Private\_Functions, 21
  - SystemCoreClockUpdate, 22
  - SystemInit, 22
- STM32F4xx\_System\_Private\_Includes, 19
  - HSE\_VALUE, 20
  - HSI\_VALUE, 20
- STM32F4xx\_System\_Private\_Macros, 20
- STM32F4xx\_System\_Private\_TypesDefinitions, 20
- STM32F4xx\_System\_Private\_Variables, 20
  - AHBPrescTable, 21
  - APBPrescTable, 21
  - SystemCoreClock, 21
- STP\_PIN
  - StepperX, 31
- sum1
  - RadioX, 29
- sum2
  - RadioX, 30
- SVC\_Handler
  - stm32f4xx\_it.c, 116
  - stm32f4xx\_it.h, 94
  - stm32f4xx\_it\_C.c, 63
  - stm32f4xx\_it\_C.h, 53
- SW
  - main.c, 105
- SW1
  - main.c, 105
- SW2
  - main.c, 105
- SW3
  - main.c, 105
- switch.c
  - Switch\_getStatus, 117
  - Switch\_init, 118
- switch.h
  - Switch\_getStatus, 95
  - Switch\_init, 95
- Switch\_getStatus
  - switch.c, 117
  - switch.h, 95
- Switch\_init
  - switch.c, 118
  - switch.h, 95
- SwitchX, 31
  - GPIOx, 31
  - PIN, 31
  - status, 32
- syscalls.c
  - \_\_attribute\_\_, 119
  - \_\_io\_getchar, 119
  - \_\_io\_putchar, 119
  - \_close, 120
  - \_execve, 120
  - \_exit, 120



- stm32f4xx\_hal\_conf.h, 89
- stm32f4xx\_hal\_conf\_C.h, 48
- USE\_HAL\_HASH\_REGISTER\_CALLBACKS
  - stm32f4xx\_hal\_conf.h, 89
  - stm32f4xx\_hal\_conf\_C.h, 48
- USE\_HAL\_HCD\_REGISTER\_CALLBACKS
  - stm32f4xx\_hal\_conf.h, 89
  - stm32f4xx\_hal\_conf\_C.h, 48
- USE\_HAL\_I2C\_REGISTER\_CALLBACKS
  - stm32f4xx\_hal\_conf.h, 89
  - stm32f4xx\_hal\_conf\_C.h, 48
- USE\_HAL\_I2S\_REGISTER\_CALLBACKS
  - stm32f4xx\_hal\_conf.h, 89
  - stm32f4xx\_hal\_conf\_C.h, 48
- USE\_HAL\_IRDA\_REGISTER\_CALLBACKS
  - stm32f4xx\_hal\_conf.h, 89
  - stm32f4xx\_hal\_conf\_C.h, 48
- USE\_HAL\_LPTIM\_REGISTER\_CALLBACKS
  - stm32f4xx\_hal\_conf.h, 89
  - stm32f4xx\_hal\_conf\_C.h, 48
- USE\_HAL\_LTDC\_REGISTER\_CALLBACKS
  - stm32f4xx\_hal\_conf.h, 90
  - stm32f4xx\_hal\_conf\_C.h, 49
- USE\_HAL\_MMC\_REGISTER\_CALLBACKS
  - stm32f4xx\_hal\_conf.h, 90
  - stm32f4xx\_hal\_conf\_C.h, 49
- USE\_HAL\_NAND\_REGISTER\_CALLBACKS
  - stm32f4xx\_hal\_conf.h, 90
  - stm32f4xx\_hal\_conf\_C.h, 49
- USE\_HAL\_NOR\_REGISTER\_CALLBACKS
  - stm32f4xx\_hal\_conf.h, 90
  - stm32f4xx\_hal\_conf\_C.h, 49
- USE\_HAL\_PCCARD\_REGISTER\_CALLBACKS
  - stm32f4xx\_hal\_conf.h, 90
  - stm32f4xx\_hal\_conf\_C.h, 49
- USE\_HAL\_PCD\_REGISTER\_CALLBACKS
  - stm32f4xx\_hal\_conf.h, 90
  - stm32f4xx\_hal\_conf\_C.h, 49
- USE\_HAL\_QSPI\_REGISTER\_CALLBACKS
  - stm32f4xx\_hal\_conf.h, 90
  - stm32f4xx\_hal\_conf\_C.h, 49
- USE\_HAL\_RNG\_REGISTER\_CALLBACKS
  - stm32f4xx\_hal\_conf.h, 90
  - stm32f4xx\_hal\_conf\_C.h, 49
- USE\_HAL\_RTC\_REGISTER\_CALLBACKS
  - stm32f4xx\_hal\_conf.h, 90
  - stm32f4xx\_hal\_conf\_C.h, 49
- USE\_HAL\_SAI\_REGISTER\_CALLBACKS
  - stm32f4xx\_hal\_conf.h, 90
  - stm32f4xx\_hal\_conf\_C.h, 49
- USE\_HAL\_SD\_REGISTER\_CALLBACKS
  - stm32f4xx\_hal\_conf.h, 91
  - stm32f4xx\_hal\_conf\_C.h, 50
- USE\_HAL\_SDRAM\_REGISTER\_CALLBACKS
  - stm32f4xx\_hal\_conf.h, 91
  - stm32f4xx\_hal\_conf\_C.h, 50
- USE\_HAL\_SMARTCARD\_REGISTER\_CALLBACKS
  - stm32f4xx\_hal\_conf.h, 91
- stm32f4xx\_hal\_conf\_C.h, 50
- USE\_HAL\_SMBUS\_REGISTER\_CALLBACKS
  - stm32f4xx\_hal\_conf.h, 91
  - stm32f4xx\_hal\_conf\_C.h, 50
- USE\_HAL\_SPDIFRX\_REGISTER\_CALLBACKS
  - stm32f4xx\_hal\_conf.h, 91
  - stm32f4xx\_hal\_conf\_C.h, 50
- USE\_HAL\_SPI\_REGISTER\_CALLBACKS
  - stm32f4xx\_hal\_conf.h, 91
  - stm32f4xx\_hal\_conf\_C.h, 50
- USE\_HAL\_SRAM\_REGISTER\_CALLBACKS
  - stm32f4xx\_hal\_conf.h, 91
  - stm32f4xx\_hal\_conf\_C.h, 50
- USE\_HAL\_TIM\_REGISTER\_CALLBACKS
  - stm32f4xx\_hal\_conf.h, 91
  - stm32f4xx\_hal\_conf\_C.h, 50
- USE\_HAL\_UART\_REGISTER\_CALLBACKS
  - stm32f4xx\_hal\_conf.h, 91
  - stm32f4xx\_hal\_conf\_C.h, 50
- USE\_HAL\_USART\_REGISTER\_CALLBACKS
  - stm32f4xx\_hal\_conf.h, 91
  - stm32f4xx\_hal\_conf\_C.h, 50
- USE\_HAL\_WWDG\_REGISTER\_CALLBACKS
  - stm32f4xx\_hal\_conf.h, 92
  - stm32f4xx\_hal\_conf\_C.h, 51
- USE\_RTOS
  - stm32f4xx\_hal\_conf.h, 92
  - stm32f4xx\_hal\_conf\_C.h, 51
- USE\_SPI\_CRC
  - stm32f4xx\_hal\_conf.h, 92
  - stm32f4xx\_hal\_conf\_C.h, 51
- VDD\_VALUE
  - stm32f4xx\_hal\_conf.h, 92
  - stm32f4xx\_hal\_conf\_C.h, 51
- yaw
  - main.c, 105
- yawDirection
  - main.c, 105
- yawDirectionSW
  - main.c, 105