

MECHA05 Term Project

Generated by Doxygen 1.10.0

Conor Schott
Vinayak Sharath
ME 507

1 Topic Index	1
1.1 Topics	1
2 Data Structure Index	3
2.1 Data Structures	3
3 File Index	5
3.1 File List	5
4 Topic Documentation	7
4.1 CMSIS	7
4.1.1 Detailed Description	7
4.1.2 Stm32f4xx_system	7
4.1.2.1 Detailed Description	7
4.1.2.2 STM32F4xx_System_Private_Includes	7
4.1.2.3 STM32F4xx_System_Private_TypesDefinitions	8
4.1.2.4 STM32F4xx_System_Private_Defines	8
4.1.2.5 STM32F4xx_System_Private_Macros	8
4.1.2.6 STM32F4xx_System_Private_Variables	8
4.1.2.7 STM32F4xx_System_Private_FunctionPrototypes	9
4.1.2.8 STM32F4xx_System_Private_Functions	9
5 Data Structure Documentation	11
5.1 bno055_axis_map_t Struct Reference	11
5.1.1 Field Documentation	11
5.1.1.1 x	11
5.1.1.2 x_sign	11
5.1.1.3 y	11
5.1.1.4 y_sign	11
5.1.1.5 z	12
5.1.1.6 z_sign	12
5.2 bno055_calibration_data_t Struct Reference	12
5.2.1 Field Documentation	12
5.2.1.1 offset	12
5.2.1.2 radius	12
5.3 bno055_calibration_offset_t Struct Reference	12
5.3.1 Field Documentation	13
5.3.1.1 accel	13
5.3.1.2 gyro	13
5.3.1.3 mag	13
5.4 bno055_calibration_radius_t Struct Reference	13
5.4.1 Field Documentation	13
5.4.1.1 accel	13
5.4.1.2 mag	13

5.5 bno055_calibration_state_t Struct Reference	14
5.5.1 Field Documentation	14
5.5.1.1 accel	14
5.5.1.2 gyro	14
5.5.1.3 mag	14
5.5.1.4 sys	14
5.6 bno055_self_test_result_t Struct Reference	14
5.6.1 Field Documentation	15
5.6.1.1 accState	15
5.6.1.2 gyrState	15
5.6.1.3 magState	15
5.6.1.4 mcuState	15
5.7 bno055_vector_t Struct Reference	15
5.7.1 Field Documentation	15
5.7.1.1 w	15
5.7.1.2 x	15
5.7.1.3 y	16
5.7.1.4 z	16
5.8 bno055_vector_xyz_int16_t Struct Reference	16
5.8.1 Field Documentation	16
5.8.1.1 x	16
5.8.1.2 y	16
5.8.1.3 z	16
5.9 MotorDriver Struct Reference	16
5.9.1 Field Documentation	17
5.9.1.1 backward_channel	17
5.9.1.2 forward_channel	17
5.9.1.3 htim	17
5.10 RCReceiver Struct Reference	17
5.10.1 Detailed Description	18
5.10.2 Field Documentation	18
5.10.2.1 highPWM	18
5.10.2.2 htim	18
5.10.2.3 lowPWM	18
5.10.2.4 message	18
5.10.2.5 nominalPWM	18
5.10.2.6 Trigger_Channel1_Fall	18
5.10.2.7 Trigger_Channel1_Rise	19
5.10.2.8 Trigger_Channel2_Fall	19
5.10.2.9 Trigger_Channel2_Rise	19

6 File Documentation

21

6.1 bno055.c File Reference	21
6.1.1 Function Documentation	22
6.1.1.1 bno055_disableExternalCrystal()	22
6.1.1.2 bno055_enableExternalCrystal()	22
6.1.1.3 bno055_getBootloaderRevision()	22
6.1.1.4 bno055_getCalibrationData()	22
6.1.1.5 bno055_getCalibrationState()	22
6.1.1.6 bno055_getOperationMode()	22
6.1.1.7 bno055_getSelfTestResult()	22
6.1.1.8 bno055_getSWRevision()	22
6.1.1.9 bno055_getSystemError()	23
6.1.1.10 bno055_getSystemStatus()	23
6.1.1.11 bno055_getTemp()	23
6.1.1.12 bno055_getVector()	23
6.1.1.13 bno055_getVectorAccelerometer()	23
6.1.1.14 bno055_getVectorEuler()	23
6.1.1.15 bno055_getVectorGravity()	23
6.1.1.16 bno055_getVectorGyroscope()	23
6.1.1.17 bno055_getVectorLinearAccel()	23
6.1.1.18 bno055_getVectorMagnetometer()	23
6.1.1.19 bno055_getVectorQuaternion()	24
6.1.1.20 bno055_reset()	24
6.1.1.21 bno055_setAxisMap()	24
6.1.1.22 bno055_setCalibrationData()	24
6.1.1.23 bno055_setExternalCrystalUse()	24
6.1.1.24 bno055_setOperationMode()	24
6.1.1.25 bno055_setOperationModeConfig()	24
6.1.1.26 bno055_setOperationModeNDOF()	24
6.1.1.27 bno055_setPage()	24
6.1.1.28 bno055_setup()	25
6.1.2 Variable Documentation	25
6.1.2.1 accelScale	25
6.1.2.2 angularRateScale	25
6.1.2.3 eulerScale	25
6.1.2.4 magScale	25
6.1.2.5 quaScale	25
6.1.2.6 tempScale	25
6.2 bno055.h File Reference	25
6.2.1 Macro Definition Documentation	29
6.2.1.1 BNO055_ACC_AM_THRES	29
6.2.1.2 BNO055_ACC_CONFIG	30
6.2.1.3 BNO055_ACC_DATA_X_LSB	30

6.2.1.4 BNO055_ACC_DATA_X_MSB	30
6.2.1.5 BNO055_ACC_DATA_Y_LSB	30
6.2.1.6 BNO055_ACC_DATA_Y_MSB	30
6.2.1.7 BNO055_ACC_DATA_Z_LSB	30
6.2.1.8 BNO055_ACC_DATA_Z_MSB	30
6.2.1.9 BNO055_ACC_HG_DURATION	30
6.2.1.10 BNO055_ACC_HG_THRESH	30
6.2.1.11 BNO055_ACC_ID	30
6.2.1.12 BNO055_ACC_INT_SETTINGS	31
6.2.1.13 BNO055_ACC_NM_SET	31
6.2.1.14 BNO055_ACC_NM_THRESH	31
6.2.1.15 BNO055_ACC_OFFSET_X_LSB	31
6.2.1.16 BNO055_ACC_OFFSET_X_MSB	31
6.2.1.17 BNO055_ACC_OFFSET_Y_LSB	31
6.2.1.18 BNO055_ACC_OFFSET_Y_MSB	31
6.2.1.19 BNO055_ACC_OFFSET_Z_LSB	31
6.2.1.20 BNO055_ACC_OFFSET_Z_MSB	31
6.2.1.21 BNO055_ACC_RADIUS_LSB	31
6.2.1.22 BNO055_ACC_RADIUS_MSB	32
6.2.1.23 BNO055_ACC_SLEEP_CONFIG	32
6.2.1.24 BNO055_AXIS_MAP_CONFIG	32
6.2.1.25 BNO055_AXIS_MAP_SIGN	32
6.2.1.26 BNO055_BL_REV_ID	32
6.2.1.27 BNO055_CALIB_STAT	32
6.2.1.28 BNO055_CHIP_ID	32
6.2.1.29 BNO055_EUL_HEADING_LSB	32
6.2.1.30 BNO055_EUL_HEADING_MSB	32
6.2.1.31 BNO055_EUL_PITCH_LSB	32
6.2.1.32 BNO055_EUL_PITCH_MSB	33
6.2.1.33 BNO055_EUL_ROLL_LSB	33
6.2.1.34 BNO055_EUL_ROLL_MSB	33
6.2.1.35 BNO055_GRV_DATA_X_LSB	33
6.2.1.36 BNO055_GRV_DATA_X_MSB	33
6.2.1.37 BNO055_GRV_DATA_Y_LSB	33
6.2.1.38 BNO055_GRV_DATA_Y_MSB	33
6.2.1.39 BNO055_GRV_DATA_Z_LSB	33
6.2.1.40 BNO055_GRV_DATA_Z_MSB	33
6.2.1.41 BNO055_GYR_AM_SET	33
6.2.1.42 BNO055_GYR_AM_THRESH	34
6.2.1.43 BNO055_GYR_DATA_X_LSB	34
6.2.1.44 BNO055_GYR_DATA_X_MSB	34
6.2.1.45 BNO055_GYR_DATA_Y_LSB	34

6.2.1.46 BNO055_GYR_DATA_Y_MSB	34
6.2.1.47 BNO055_GYR_DATA_Z_LSB	34
6.2.1.48 BNO055_GYR_DATA_Z_MSB	34
6.2.1.49 BNO055_GYR_DUR_X	34
6.2.1.50 BNO055_GYR_DUR_Y	34
6.2.1.51 BNO055_GYR_DUR_Z	34
6.2.1.52 BNO055_GYR_HR_X_SET	35
6.2.1.53 BNO055_GYR_HR_Y_SET	35
6.2.1.54 BNO055_GYR_HR_Z_SET	35
6.2.1.55 BNO055_GYR_INT_SETTINGS	35
6.2.1.56 BNO055_GYR_OFFSET_X_LSB	35
6.2.1.57 BNO055_GYR_OFFSET_X_MSB	35
6.2.1.58 BNO055_GYR_OFFSET_Y_LSB	35
6.2.1.59 BNO055_GYR_OFFSET_Y_MSB	35
6.2.1.60 BNO055_GYR_OFFSET_Z_LSB	35
6.2.1.61 BNO055_GYR_OFFSET_Z_MSB	35
6.2.1.62 BNO055_GYR_SLEEP_CONFIG	36
6.2.1.63 BNO055_GYRO_CONFIG_0	36
6.2.1.64 BNO055_GYRO_CONFIG_1	36
6.2.1.65 BNO055_GYRO_ID	36
6.2.1.66 BNO055_I2C_ADDR	36
6.2.1.67 BNO055_I2C_ADDR_HI	36
6.2.1.68 BNO055_I2C_ADDR_LO	36
6.2.1.69 BNO055_ID	36
6.2.1.70 BNO055_INT_EN	36
6.2.1.71 BNO055_INT_MSK	36
6.2.1.72 BNO055_INT_STATUS	37
6.2.1.73 BNO055_LIA_DATA_X_LSB	37
6.2.1.74 BNO055_LIA_DATA_X_MSB	37
6.2.1.75 BNO055_LIA_DATA_Y_LSB	37
6.2.1.76 BNO055_LIA_DATA_Y_MSB	37
6.2.1.77 BNO055_LIA_DATA_Z_LSB	37
6.2.1.78 BNO055_LIA_DATA_Z_MSB	37
6.2.1.79 BNO055_MAG_CONFIG	37
6.2.1.80 BNO055_MAG_DATA_X_LSB	37
6.2.1.81 BNO055_MAG_DATA_X_MSB	37
6.2.1.82 BNO055_MAG_DATA_Y_LSB	38
6.2.1.83 BNO055_MAG_DATA_Y_MSB	38
6.2.1.84 BNO055_MAG_DATA_Z_LSB	38
6.2.1.85 BNO055_MAG_DATA_Z_MSB	38
6.2.1.86 BNO055_MAG_ID	38
6.2.1.87 BNO055_MAG_OFFSET_X_LSB	38

6.2.1.88 BNO055_MAG_OFFSET_X_MSB	38
6.2.1.89 BNO055_MAG_OFFSET_Y_LSB	38
6.2.1.90 BNO055_MAG_OFFSET_Y_MSB	38
6.2.1.91 BNO055_MAG_OFFSET_Z_LSB	38
6.2.1.92 BNO055_MAG_OFFSET_Z_MSB	39
6.2.1.93 BNO055_MAG_RADIUS_LSB	39
6.2.1.94 BNO055_MAG_RADIUS_MSB	39
6.2.1.95 BNO055_OPR_MODE	39
6.2.1.96 BNO055_PAGE_ID [1/2]	39
6.2.1.97 BNO055_PAGE_ID [2/2]	39
6.2.1.98 BNO055_PWR_MODE	39
6.2.1.99 BNO055_QUA_DATA_W_LSB	39
6.2.1.100 BNO055_QUA_DATA_W_MSB	39
6.2.1.101 BNO055_QUA_DATA_X_LSB	39
6.2.1.102 BNO055_QUA_DATA_X_MSB	40
6.2.1.103 BNO055_QUA_DATA_Y_LSB	40
6.2.1.104 BNO055_QUA_DATA_Y_MSB	40
6.2.1.105 BNO055_QUA_DATA_Z_LSB	40
6.2.1.106 BNO055_QUA_DATA_Z_MSB	40
6.2.1.107 BNO055_READ_TIMEOUT	40
6.2.1.108 BNO055_ST_RESULT	40
6.2.1.109 BNO055_SW_REV_ID_LSB	40
6.2.1.110 BNO055_SW_REV_ID_MSB	40
6.2.1.111 BNO055_SYS_CLK_STATUS	40
6.2.1.112 BNO055_SYS_ERR	41
6.2.1.113 BNO055_SYS_STATUS	41
6.2.1.114 BNO055_SYS_TRIGGER	41
6.2.1.115 BNO055_TEMP	41
6.2.1.116 BNO055_TEMP_SOURCE	41
6.2.1.117 BNO055_UNIT_SEL	41
6.2.1.118 BNO055_WRITE_TIMEOUT	41
6.2.1.119 ERROR_BUS_OVERRUN_ERR	41
6.2.1.120 ERROR_BYTE	41
6.2.1.121 ERROR_MAX_LEN_ERR	41
6.2.1.122 ERROR_MIN_LEN_ERR	42
6.2.1.123 ERROR_RECV_CHAR_TIMEOUT	42
6.2.1.124 ERROR_REGMAP_INV_ADDR	42
6.2.1.125 ERROR_REGMAP_WRITE_DIS	42
6.2.1.126 ERROR_WRITE_FAIL	42
6.2.1.127 ERROR_WRITE_SUCCESS	42
6.2.1.128 ERROR_WRONG_START_BYTE	42
6.2.1.129 REG_READ	42

6.2.1.130 REG_WRITE	42
6.2.1.131 RESPONSE_BYTE	42
6.2.1.132 START_BYTE	42
6.2.2 Enumeration Type Documentation	42
6.2.2.1 bno055_axis_map_representation_t	42
6.2.2.2 bno055_axis_map_sign_t	43
6.2.2.3 bno055_opmode_t	43
6.2.2.4 bno055_system_error_t	43
6.2.2.5 bno055_system_status_t	44
6.2.2.6 bno055_vector_type_t	44
6.2.3 Function Documentation	44
6.2.3.1 bno055_delay()	44
6.2.3.2 bno055_disableExternalCrystal()	45
6.2.3.3 bno055_enableExternalCrystal()	45
6.2.3.4 bno055_getBootloaderRevision()	45
6.2.3.5 bno055_getCalibrationData()	45
6.2.3.6 bno055_getCalibrationState()	45
6.2.3.7 bno055_getOperationMode()	45
6.2.3.8 bno055_getSelfTestResult()	45
6.2.3.9 bno055_getSWRevision()	45
6.2.3.10 bno055_getSystemError()	45
6.2.3.11 bno055_getSystemStatus()	45
6.2.3.12 bno055_getTemp()	46
6.2.3.13 bno055_getVectorAccelerometer()	46
6.2.3.14 bno055_getVectorEuler()	46
6.2.3.15 bno055_getVectorGravity()	46
6.2.3.16 bno055_getVectorGyroscope()	46
6.2.3.17 bno055_getVectorLinearAccel()	46
6.2.3.18 bno055_getVectorMagnetometer()	46
6.2.3.19 bno055_getVectorQuaternion()	46
6.2.3.20 bno055_readData()	46
6.2.3.21 bno055_reset()	46
6.2.3.22 bno055_setAxisMap()	47
6.2.3.23 bno055_setCalibrationData()	47
6.2.3.24 bno055_setOperationMode()	47
6.2.3.25 bno055_setOperationModeConfig()	47
6.2.3.26 bno055_setOperationModeNDOF()	47
6.2.3.27 bno055_setup()	47
6.2.3.28 bno055_writeData()	47
6.3 bno055.h	48
6.4 bno055_stm32.h File Reference	51
6.4.1 Function Documentation	52

6.4.1.1 bno055_assignI2C()	52
6.4.1.2 bno055_delay()	52
6.4.1.3 bno055_readData()	52
6.4.1.4 bno055_writeData()	52
6.4.2 Variable Documentation	52
6.4.2.1 _bno055_i2c_port	52
6.5 bno055_stm32.h	53
6.6 fsr_controller.c File Reference	54
6.6.1 Function Documentation	54
6.6.1.1 apply_low_pass_filter()	54
6.6.1.2 get_averaged_adc_value()	55
6.6.1.3 map()	55
6.7 fsr_controller.h File Reference	56
6.7.1 Function Documentation	56
6.7.1.1 apply_low_pass_filter()	56
6.7.1.2 get_averaged_adc_value()	57
6.7.1.3 map()	57
6.8 fsr_controller.h	59
6.9 main.c File Reference	59
6.9.1 Detailed Description	60
6.9.2 Macro Definition Documentation	61
6.9.2.1 FILTER_SIZE	61
6.9.3 Function Documentation	61
6.9.3.1 isRadioOn()	61
6.9.3.2 main()	61
6.9.3.3 SystemClock_Config()	63
6.9.4 Variable Documentation	63
6.9.4.1 adc_value	63
6.9.4.2 euler_index	63
6.9.4.3 euler_y	63
6.9.4.4 euler_z	63
6.9.4.5 hadc1	63
6.9.4.6 hdma_i2c1_rx	63
6.9.4.7 hdma_i2c1_tx	64
6.9.4.8 hi2c1	64
6.9.4.9 htim1	64
6.9.4.10 htim2	64
6.9.4.11 htim3	64
6.9.4.12 hysteresis	64
6.9.4.13 message	64
6.9.4.14 motor1	64
6.9.4.15 motor2	64

6.9.4.16 pointer_rc	64
6.9.4.17 press_detected	65
6.9.4.18 receiver	65
6.9.4.19 servo_speed	65
6.9.4.20 state	65
6.9.4.21 sum_y	65
6.9.4.22 sum_z	65
6.9.4.23 threshold	65
6.9.4.24 x_position	65
6.9.4.25 y_position	65
6.10 Motor_Driver.c File Reference	66
6.10.1 Macro Definition Documentation	66
6.10.1.1 MAX_DUTY_CYCLE	66
6.10.2 Function Documentation	66
6.10.2.1 motor_disable()	66
6.10.2.2 motor_enable()	66
6.10.2.3 motor_set_duty_cycle()	67
6.11 Motor_Driver.h File Reference	67
6.11.1 Function Documentation	67
6.11.1.1 motor_disable()	67
6.11.1.2 motor_enable()	68
6.11.1.3 motor_set_duty_cycle()	68
6.12 Motor_Driver.h	68
6.13 Receiver.c File Reference	69
6.13.1 Macro Definition Documentation	69
6.13.1.1 MESSAGE_LENGTH	69
6.13.2 Function Documentation	69
6.13.2.1 calculateTriggerValueCallback()	69
6.13.2.2 calculateWheelValueCallback()	70
6.13.2.3 initializeRCReceiver()	70
6.13.2.4 startRCReceiverCapture()	70
6.13.3 Variable Documentation	71
6.13.3.1 g_trigger_val	71
6.13.3.2 g_wheel_val	71
6.14 Receiver.h File Reference	71
6.14.1 Macro Definition Documentation	71
6.14.1.1 MESSAGE_LENGTH	71
6.14.2 Function Documentation	71
6.14.2.1 calculateTriggerValueCallback()	71
6.14.2.2 calculateWheelValueCallback()	72
6.14.2.3 initializeRCReceiver()	72
6.14.2.4 startRCReceiverCapture()	72

6.15 Receiver.h	73
6.16 stm32f4xx_hal_msp.c File Reference	73
6.16.1 Detailed Description	74
6.16.2 Function Documentation	74
6.16.2.1 HAL_ADC_MspDeInit()	74
6.16.2.2 HAL_ADC_MspInit()	75
6.16.2.3 HAL_I2C_MspDeInit()	75
6.16.2.4 HAL_I2C_MspInit()	75
6.16.2.5 HAL_MspInit()	76
6.16.2.6 HAL_TIM_Base_MspDeInit()	76
6.16.2.7 HAL_TIM_Base_MspInit()	76
6.16.2.8 HAL_TIM_IC_MspDeInit()	76
6.16.2.9 HAL_TIM_IC_MspInit()	77
6.16.2.10 HAL_TIM_MspPostInit()	77
6.16.2.11 HAL_TIM_PWM_MspDeInit()	77
6.16.2.12 HAL_TIM_PWM_MspInit()	78
6.16.3 Variable Documentation	78
6.16.3.1 hdma_i2c1_rx	78
6.16.3.2 hdma_i2c1_tx	78
6.17 stm32f4xx_it.c File Reference	78
6.17.1 Detailed Description	79
6.17.2 Function Documentation	79
6.17.2.1 BusFault_Handler()	79
6.17.2.2 DebugMon_Handler()	80
6.17.2.3 DMA1_Stream0_IRQHandler()	80
6.17.2.4 DMA1_Stream1_IRQHandler()	80
6.17.2.5 HardFault_Handler()	80
6.17.2.6 MemManage_Handler()	80
6.17.2.7 NMI_Handler()	80
6.17.2.8 PendSV_Handler()	80
6.17.2.9 SVC_Handler()	81
6.17.2.10 SysTick_Handler()	81
6.17.2.11 UsageFault_Handler()	81
6.17.3 Variable Documentation	81
6.17.3.1 hdma_i2c1_rx	81
6.17.3.2 hdma_i2c1_tx	81
6.18 syscalls.c File Reference	81
6.18.1 Detailed Description	82
6.18.2 Function Documentation	82
6.18.2.1 __attribute__	82
6.18.2.2 __io_getchar()	83
6.18.2.3 __io_putchar()	83

6.18.2.4 _close()	83
6.18.2.5 _execve()	83
6.18.2.6 _exit()	83
6.18.2.7 _fork()	83
6.18.2.8 _fstat()	83
6.18.2.9 _getpid()	83
6.18.2.10 _isatty()	84
6.18.2.11 _kill()	84
6.18.2.12 _link()	84
6.18.2.13 _lseek()	84
6.18.2.14 _open()	84
6.18.2.15 _stat()	84
6.18.2.16 _times()	84
6.18.2.17 _unlink()	84
6.18.2.18 _wait()	85
6.18.2.19 initialise_monitor_handles()	85
6.18.3 Variable Documentation	85
6.18.3.1 environ	85
6.19 systemem.c File Reference	85
6.19.1 Detailed Description	85
6.19.2 Function Documentation	86
6.19.2.1 _sbrk()	86
6.20 system_stm32f4xx.c File Reference	86
6.20.1 Detailed Description	87
Index	89

Chapter 1

Topic Index

1.1 Topics

Here is a list of all topics with brief descriptions:

CMSIS	7
Stm32f4xx_system	7
STM32F4xx_System_Private_Includes	7
STM32F4xx_System_Private_TypesDefinitions	8
STM32F4xx_System_Private_Defines	8
STM32F4xx_System_Private_Macros	8
STM32F4xx_System_Private_Variables	8
STM32F4xx_System_Private_FunctionPrototypes	9
STM32F4xx_System_Private_Functions	9

Chapter 2

Data Structure Index

2.1 Data Structures

Here are the data structures with brief descriptions:

bno055_axis_map_t	11
bno055_calibration_data_t	12
bno055_calibration_offset_t	12
bno055_calibration_radius_t	13
bno055_calibration_state_t	14
bno055_self_test_result_t	14
bno055_vector_t	15
bno055_vector_xyz_int16_t	16
MotorDriver	16
RCReceiver	
Structure to hold receiver configuration and state	17

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

bno055.c	21
bno055.h	25
bno055_stm32.h	51
fsr_controller.c	54
fsr_controller.h	56
main.c	
: Main program body	59
Motor_Driver.c	66
Motor_Driver.h	67
Receiver.c	69
Receiver.h	71
stm32f4xx_hal_msp.c	
This file provides code for the MSP Initialization and de-Initialization codes	73
stm32f4xx_it.c	
Interrupt Service Routines	78
syscalls.c	
STM32CubeIDE Minimal System calls file	81
systemem.c	
STM32CubeIDE System Memory calls file	85
system_stm32f4xx.c	
CMSIS Cortex-M4 Device Peripheral Access Layer System Source File	86

Chapter 4

Topic Documentation

4.1 CMSIS

Topics

- [Stm32f4xx_system](#)

4.1.1 Detailed Description

4.1.2 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](#)

4.1.2.1 Detailed Description

4.1.2.2 STM32F4xx_System_Private_Includes

Macros

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

4.1.2.2.1 Detailed Description**4.1.2.2.2 Macro Definition Documentation****4.1.2.2.2.1 HSE_VALUE**

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

Default value of the External oscillator in Hz

4.1.2.2.2.2 HSI_VALUE

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

Value of the Internal oscillator in Hz

4.1.2.3 STM32F4xx_System_Private_Definitions**4.1.2.4 STM32F4xx_System_Private_Defines****4.1.2.5 STM32F4xx_System_Private_Macros****4.1.2.6 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}

4.1.2.6.1 Detailed Description**4.1.2.6.2 Variable Documentation****4.1.2.6.2.1 AHBPrescTable**

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

4.1.2.6.2.2 APBPrescTable

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

4.1.2.6.2.3 SystemCoreClock

```
uint32_t SystemCoreClock = 16000000
```

4.1.2.7 STM32F4xx_System_Private_FunctionPrototypes

4.1.2.8 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.

4.1.2.8.1 Detailed Description

4.1.2.8.2 Function Documentation

4.1.2.8.2.1 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\(*\)](#)
- If SYSCLK source is HSE, SystemCoreClock will contain the [HSE_VALUE\(**\)](#)
- If SYSCLK source is PLL, SystemCoreClock will contain the [HSE_VALUE\(**\)](#) or [HSI_VALUE\(*\)](#) multiplied/divided by the PLL factors.

(*) HSI_VALUE is a constant defined in stm32f4xx_hal_conf.h 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 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

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

Return values

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

4.1.2.8.2.2 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 5

Data Structure Documentation

5.1 bno055_axis_map_t Struct Reference

```
#include <bno055.h>
```

Data Fields

- [uint8_t x](#)
- [uint8_t x_sign](#)
- [uint8_t y](#)
- [uint8_t y_sign](#)
- [uint8_t z](#)
- [uint8_t z_sign](#)

5.1.1 Field Documentation

5.1.1.1 x

```
uint8_t x
```

5.1.1.2 x_sign

```
uint8_t x_sign
```

5.1.1.3 y

```
uint8_t y
```

5.1.1.4 y_sign

```
uint8_t y_sign
```

5.1.1.5 z

```
uint8_t z
```

5.1.1.6 z_sign

```
uint8_t z_sign
```

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

- [bno055.h](#)

5.2 bno055_calibration_data_t Struct Reference

```
#include <bno055.h>
```

Data Fields

- [bno055_calibration_offset_t](#) offset
- [bno055_calibration_radius_t](#) radius

5.2.1 Field Documentation

5.2.1.1 offset

```
bno055_calibration_offset_t offset
```

5.2.1.2 radius

```
bno055_calibration_radius_t radius
```

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

- [bno055.h](#)

5.3 bno055_calibration_offset_t Struct Reference

```
#include <bno055.h>
```

Data Fields

- [bno055_vector_xyz_int16_t](#) gyro
- [bno055_vector_xyz_int16_t](#) mag
- [bno055_vector_xyz_int16_t](#) accel

5.3.1 Field Documentation

5.3.1.1 accel

`bno055_vector_xyz_int16_t` accel

5.3.1.2 gyro

`bno055_vector_xyz_int16_t` gyro

5.3.1.3 mag

`bno055_vector_xyz_int16_t` mag

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

- [bno055.h](#)

5.4 bno055_calibration_radius_t Struct Reference

```
#include <bno055.h>
```

Data Fields

- `uint16_t` [mag](#)
- `uint16_t` [accel](#)

5.4.1 Field Documentation

5.4.1.1 accel

`uint16_t` accel

5.4.1.2 mag

`uint16_t` mag

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

- [bno055.h](#)

5.5 bno055_calibration_state_t Struct Reference

```
#include <bno055.h>
```

Data Fields

- [uint8_t sys](#)
- [uint8_t gyro](#)
- [uint8_t mag](#)
- [uint8_t accel](#)

5.5.1 Field Documentation

5.5.1.1 accel

```
uint8_t accel
```

5.5.1.2 gyro

```
uint8_t gyro
```

5.5.1.3 mag

```
uint8_t mag
```

5.5.1.4 sys

```
uint8_t sys
```

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

- [bno055.h](#)

5.6 bno055_self_test_result_t Struct Reference

```
#include <bno055.h>
```

Data Fields

- [uint8_t mcuState](#)
- [uint8_t gyrState](#)
- [uint8_t magState](#)
- [uint8_t accState](#)

5.6.1 Field Documentation

5.6.1.1 accState

`uint8_t accState`

5.6.1.2 gyrState

`uint8_t gyrState`

5.6.1.3 magState

`uint8_t magState`

5.6.1.4 mcuState

`uint8_t mcuState`

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

- [bno055.h](#)

5.7 bno055_vector_t Struct Reference

```
#include <bno055.h>
```

Data Fields

- double [w](#)
- double [x](#)
- double [y](#)
- double [z](#)

5.7.1 Field Documentation

5.7.1.1 w

`double w`

5.7.1.2 x

`double x`

5.7.1.3 y

double y

5.7.1.4 z

double z

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

- [bno055.h](#)

5.8 bno055_vector_xyz_int16_t Struct Reference

```
#include <bno055.h>
```

Data Fields

- [int16_t x](#)
- [int16_t y](#)
- [int16_t z](#)

5.8.1 Field Documentation

5.8.1.1 x

int16_t x

5.8.1.2 y

int16_t y

5.8.1.3 z

int16_t z

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

- [bno055.h](#)

5.9 MotorDriver Struct Reference

```
#include <Motor_Driver.h>
```

Data Fields

- uint32_t [forward_channel](#)
- uint32_t [backward_channel](#)
- TIM_HandleTypeDef * [htim](#)

5.9.1 Field Documentation**5.9.1.1 backward_channel**

```
uint32_t backward_channel
```

5.9.1.2 forward_channel

```
uint32_t forward_channel
```

5.9.1.3 htim

```
TIM_HandleTypeDef* htim
```

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

- [Motor_Driver.h](#)

5.10 RCReceiver Struct Reference

Structure to hold receiver configuration and state.

```
#include <Receiver.h>
```

Data Fields

- TIM_HandleTypeDef * [htim](#)
Pointer to the Timer handle.
- uint32_t [nominalPWM](#)
Nominal PWM value.
- uint32_t [highPWM](#)
High PWM threshold.
- uint32_t [lowPWM](#)
Low PWM threshold.
- uint16_t [Trigger_Channel1_Rise](#)
Timer channel for Trigger - Rising edge.
- uint16_t [Trigger_Channel1_Fall](#)
Timer channel for Trigger - Falling edge.
- uint16_t [Trigger_Channel2_Rise](#)
Timer channel for Trigger - Rising edge (second channel).
- uint16_t [Trigger_Channel2_Fall](#)
Timer channel for Trigger - Falling edge (second channel).
- uint16_t [message](#) [[MESSAGE_LENGTH](#)]
Buffer to hold received messages.

5.10.1 Detailed Description

Structure to hold receiver configuration and state.

5.10.2 Field Documentation

5.10.2.1 highPWM

```
uint32_t highPWM
```

High PWM threshold.

5.10.2.2 htim

```
TIM_HandleTypeDef* htim
```

Pointer to the Timer handle.

5.10.2.3 lowPWM

```
uint32_t lowPWM
```

Low PWM threshold.

5.10.2.4 message

```
uint16_t message[MESSAGE_LENGTH]
```

Buffer to hold received messages.

5.10.2.5 nominalPWM

```
uint32_t nominalPWM
```

Nominal PWM value.

5.10.2.6 Trigger_Channel1_Fall

```
uint16_t Trigger_Channel1_Fall
```

Timer channel for Trigger - Falling edge.

5.10.2.7 Trigger_Channel1_Rise

```
uint16_t Trigger_Channel1_Rise
```

Timer channel for Trigger - Rising edge.

5.10.2.8 Trigger_Channel2_Fall

```
uint16_t Trigger_Channel2_Fall
```

Timer channel for Trigger - Falling edge (second channel).

5.10.2.9 Trigger_Channel2_Rise

```
uint16_t Trigger_Channel2_Rise
```

Timer channel for Trigger - Rising edge (second channel).

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

- [Receiver.h](#)

Chapter 6

File Documentation

6.1 bno055.c File Reference

```
#include "bno055.h"  
#include <string.h>
```

Functions

- void [bno055_setPage](#) (uint8_t page)
- [bno055_opmode_t](#) [bno055_getOperationMode](#) ()
- void [bno055_setOperationMode](#) ([bno055_opmode_t](#) mode)
- void [bno055_setOperationModeConfig](#) ()
- void [bno055_setOperationModeNDOF](#) ()
- void [bno055_setExternalCrystalUse](#) (bool state)
- void [bno055_enableExternalCrystal](#) ()
- void [bno055_disableExternalCrystal](#) ()
- void [bno055_reset](#) ()
- int8_t [bno055_getTemp](#) ()
- void [bno055_setup](#) ()
- int16_t [bno055_getSWRevision](#) ()
- uint8_t [bno055_getBootloaderRevision](#) ()
- uint8_t [bno055_getSystemStatus](#) ()
- [bno055_self_test_result_t](#) [bno055_getSelfTestResult](#) ()
- uint8_t [bno055_getSystemError](#) ()
- [bno055_calibration_state_t](#) [bno055_getCalibrationState](#) ()
- [bno055_calibration_data_t](#) [bno055_getCalibrationData](#) ()
- void [bno055_setCalibrationData](#) ([bno055_calibration_data_t](#) calData)
- [bno055_vector_t](#) [bno055_getVector](#) (uint8_t vec)
- [bno055_vector_t](#) [bno055_getVectorAccelerometer](#) ()
- [bno055_vector_t](#) [bno055_getVectorMagnetometer](#) ()
- [bno055_vector_t](#) [bno055_getVectorGyroscope](#) ()
- [bno055_vector_t](#) [bno055_getVectorEuler](#) ()
- [bno055_vector_t](#) [bno055_getVectorLinearAccel](#) ()
- [bno055_vector_t](#) [bno055_getVectorGravity](#) ()
- [bno055_vector_t](#) [bno055_getVectorQuaternion](#) ()
- void [bno055_setAxisMap](#) ([bno055_axis_map_t](#) axis)

Variables

- uint16_t `accelScale` = 100
- uint16_t `tempScale` = 1
- uint16_t `angularRateScale` = 16
- uint16_t `eulerScale` = 16
- uint16_t `magScale` = 16
- uint16_t `quaScale` = (1<<14)

6.1.1 Function Documentation

6.1.1.1 bno055_disableExternalCrystal()

```
void bno055_disableExternalCrystal ( )
```

6.1.1.2 bno055_enableExternalCrystal()

```
void bno055_enableExternalCrystal ( )
```

6.1.1.3 bno055_getBootloaderRevision()

```
uint8_t bno055_getBootloaderRevision ( )
```

6.1.1.4 bno055_getCalibrationData()

```
bno055_calibration_data_t bno055_getCalibrationData ( )
```

6.1.1.5 bno055_getCalibrationState()

```
bno055_calibration_state_t bno055_getCalibrationState ( )
```

6.1.1.6 bno055_getOperationMode()

```
bno055_opmode_t bno055_getOperationMode ( )
```

6.1.1.7 bno055_getSelfTestResult()

```
bno055_self_test_result_t bno055_getSelfTestResult ( )
```

6.1.1.8 bno055_getSWRevision()

```
int16_t bno055_getSWRevision ( )
```

6.1.1.9 bno055_getSystemError()

```
uint8_t bno055_getSystemError ( )
```

6.1.1.10 bno055_getSystemStatus()

```
uint8_t bno055_getSystemStatus ( )
```

6.1.1.11 bno055_getTemp()

```
int8_t bno055_getTemp ( )
```

6.1.1.12 bno055_getVector()

```
bno055_vector_t bno055_getVector (
    uint8_t vec )
```

6.1.1.13 bno055_getVectorAccelerometer()

```
bno055_vector_t bno055_getVectorAccelerometer ( )
```

6.1.1.14 bno055_getVectorEuler()

```
bno055_vector_t bno055_getVectorEuler ( )
```

6.1.1.15 bno055_getVectorGravity()

```
bno055_vector_t bno055_getVectorGravity ( )
```

6.1.1.16 bno055_getVectorGyroscope()

```
bno055_vector_t bno055_getVectorGyroscope ( )
```

6.1.1.17 bno055_getVectorLinearAccel()

```
bno055_vector_t bno055_getVectorLinearAccel ( )
```

6.1.1.18 bno055_getVectorMagnetometer()

```
bno055_vector_t bno055_getVectorMagnetometer ( )
```

6.1.1.19 bno055_getVectorQuaternion()

```
bno055_vector_t bno055_getVectorQuaternion ( )
```

6.1.1.20 bno055_reset()

```
void bno055_reset ( )
```

6.1.1.21 bno055_setAxisMap()

```
void bno055_setAxisMap (
    bno055_axis_map_t axis )
```

6.1.1.22 bno055_setCalibrationData()

```
void bno055_setCalibrationData (
    bno055_calibration_data_t calData )
```

6.1.1.23 bno055_setExternalCrystalUse()

```
void bno055_setExternalCrystalUse (
    bool state )
```

6.1.1.24 bno055_setOperationMode()

```
void bno055_setOperationMode (
    bno055_opmode_t mode )
```

6.1.1.25 bno055_setOperationModeConfig()

```
void bno055_setOperationModeConfig ( )
```

6.1.1.26 bno055_setOperationModeNDOF()

```
void bno055_setOperationModeNDOF ( )
```

6.1.1.27 bno055_setPage()

```
void bno055_setPage (
    uint8_t page )
```

6.1.1.28 bno055_setup()

```
void bno055_setup ( )
```

6.1.2 Variable Documentation

6.1.2.1 accelScale

```
uint16_t accelScale = 100
```

6.1.2.2 angularRateScale

```
uint16_t angularRateScale = 16
```

6.1.2.3 eulerScale

```
uint16_t eulerScale = 16
```

6.1.2.4 magScale

```
uint16_t magScale = 16
```

6.1.2.5 quaScale

```
uint16_t quaScale = (1<<14)
```

6.1.2.6 tempScale

```
uint16_t tempScale = 1
```

6.2 bno055.h File Reference

```
#include <stdbool.h>
#include <stdint.h>
#include <stdio.h>
```

Data Structures

- struct [bno055_self_test_result_t](#)
- struct [bno055_calibration_state_t](#)
- struct [bno055_vector_xyz_int16_t](#)
- struct [bno055_calibration_offset_t](#)
- struct [bno055_calibration_radius_t](#)
- struct [bno055_calibration_data_t](#)
- struct [bno055_vector_t](#)
- struct [bno055_axis_map_t](#)

Macros

- #define [START_BYTE](#) 0xAA
- #define [RESPONSE_BYTE](#) 0xBB
- #define [ERROR_BYTE](#) 0xEE
- #define [BNO055_I2C_ADDR_HI](#) 0x29
- #define [BNO055_I2C_ADDR_LO](#) 0x28
- #define [BNO055_I2C_ADDR](#) [BNO055_I2C_ADDR_LO](#)
- #define [BNO055_READ_TIMEOUT](#) 100
- #define [BNO055_WRITE_TIMEOUT](#) 10
- #define [ERROR_WRITE_SUCCESS](#) 0x01
- #define [ERROR_WRITE_FAIL](#) 0x03
- #define [ERROR_REGMAP_INV_ADDR](#) 0x04
- #define [ERROR_REGMAP_WRITE_DIS](#) 0x05
- #define [ERROR_WRONG_START_BYTE](#) 0x06
- #define [ERROR_BUS_OVERRUN_ERR](#) 0x07
- #define [ERROR_MAX_LEN_ERR](#) 0x08
- #define [ERROR_MIN_LEN_ERR](#) 0x09
- #define [ERROR_RECV_CHAR_TIMEOUT](#) 0x0A
- #define [REG_WRITE](#) 0x00
- #define [REG_READ](#) 0x01
- #define [BNO055_ID](#) (0xA0)
- #define [BNO055_CHIP_ID](#) 0x00
- #define [BNO055_ACC_ID](#) 0x01
- #define [BNO055_MAG_ID](#) 0x02
- #define [BNO055_GYRO_ID](#) 0x03
- #define [BNO055_SW_REV_ID_LSB](#) 0x04
- #define [BNO055_SW_REV_ID_MSB](#) 0x05
- #define [BNO055_BL_REV_ID](#) 0x06
- #define [BNO055_PAGE_ID](#) 0x07
- #define [BNO055_ACC_DATA_X_LSB](#) 0x08
- #define [BNO055_ACC_DATA_X_MSB](#) 0x09
- #define [BNO055_ACC_DATA_Y_LSB](#) 0x0A
- #define [BNO055_ACC_DATA_Y_MSB](#) 0x0B
- #define [BNO055_ACC_DATA_Z_LSB](#) 0x0C
- #define [BNO055_ACC_DATA_Z_MSB](#) 0x0D
- #define [BNO055_MAG_DATA_X_LSB](#) 0x0E
- #define [BNO055_MAG_DATA_X_MSB](#) 0x0F
- #define [BNO055_MAG_DATA_Y_LSB](#) 0x10
- #define [BNO055_MAG_DATA_Y_MSB](#) 0x11
- #define [BNO055_MAG_DATA_Z_LSB](#) 0x12
- #define [BNO055_MAG_DATA_Z_MSB](#) 0x13
- #define [BNO055_GYR_DATA_X_LSB](#) 0x14

- #define [BNO055_GYR_DATA_X_MSB](#) 0x15
- #define [BNO055_GYR_DATA_Y_LSB](#) 0x16
- #define [BNO055_GYR_DATA_Y_MSB](#) 0x17
- #define [BNO055_GYR_DATA_Z_LSB](#) 0x18
- #define [BNO055_GYR_DATA_Z_MSB](#) 0x19
- #define [BNO055_EUL_HEADING_LSB](#) 0x1A
- #define [BNO055_EUL_HEADING_MSB](#) 0x1B
- #define [BNO055_EUL_ROLL_LSB](#) 0x1C
- #define [BNO055_EUL_ROLL_MSB](#) 0x1D
- #define [BNO055_EUL_PITCH_LSB](#) 0x1E
- #define [BNO055_EUL_PITCH_MSB](#) 0x1F
- #define [BNO055_QUA_DATA_W_LSB](#) 0x20
- #define [BNO055_QUA_DATA_W_MSB](#) 0x21
- #define [BNO055_QUA_DATA_X_LSB](#) 0x22
- #define [BNO055_QUA_DATA_X_MSB](#) 0x23
- #define [BNO055_QUA_DATA_Y_LSB](#) 0x24
- #define [BNO055_QUA_DATA_Y_MSB](#) 0x25
- #define [BNO055_QUA_DATA_Z_LSB](#) 0x26
- #define [BNO055_QUA_DATA_Z_MSB](#) 0x27
- #define [BNO055_LIA_DATA_X_LSB](#) 0x28
- #define [BNO055_LIA_DATA_X_MSB](#) 0x29
- #define [BNO055_LIA_DATA_Y_LSB](#) 0x2A
- #define [BNO055_LIA_DATA_Y_MSB](#) 0x2B
- #define [BNO055_LIA_DATA_Z_LSB](#) 0x2C
- #define [BNO055_LIA_DATA_Z_MSB](#) 0x2D
- #define [BNO055_GRV_DATA_X_LSB](#) 0x2E
- #define [BNO055_GRV_DATA_X_MSB](#) 0x2F
- #define [BNO055_GRV_DATA_Y_LSB](#) 0x30
- #define [BNO055_GRV_DATA_Y_MSB](#) 0x31
- #define [BNO055_GRV_DATA_Z_LSB](#) 0x32
- #define [BNO055_GRV_DATA_Z_MSB](#) 0x33
- #define [BNO055_TEMP](#) 0x34
- #define [BNO055_CALIB_STAT](#) 0x35
- #define [BNO055_ST_RESULT](#) 0x36
- #define [BNO055_INT_STATUS](#) 0x37
- #define [BNO055_SYS_CLK_STATUS](#) 0x38
- #define [BNO055_SYS_STATUS](#) 0x39
- #define [BNO055_SYS_ERR](#) 0x3A
- #define [BNO055_UNIT_SEL](#) 0x3B
- #define [BNO055_OPR_MODE](#) 0x3D
- #define [BNO055_PWR_MODE](#) 0x3E
- #define [BNO055_SYS_TRIGGER](#) 0x3F
- #define [BNO055_TEMP_SOURCE](#) 0x40
- #define [BNO055_AXIS_MAP_CONFIG](#) 0x41
- #define [BNO055_AXIS_MAP_SIGN](#) 0x42
- #define [BNO055_ACC_OFFSET_X_LSB](#) 0x55
- #define [BNO055_ACC_OFFSET_X_MSB](#) 0x56
- #define [BNO055_ACC_OFFSET_Y_LSB](#) 0x57
- #define [BNO055_ACC_OFFSET_Y_MSB](#) 0x58
- #define [BNO055_ACC_OFFSET_Z_LSB](#) 0x59
- #define [BNO055_ACC_OFFSET_Z_MSB](#) 0x5A
- #define [BNO055_MAG_OFFSET_X_LSB](#) 0x5B
- #define [BNO055_MAG_OFFSET_X_MSB](#) 0x5C
- #define [BNO055_MAG_OFFSET_Y_LSB](#) 0x5D
- #define [BNO055_MAG_OFFSET_Y_MSB](#) 0x5E

- #define [BNO055_MAG_OFFSET_Z_LSB](#) 0x5F
- #define [BNO055_MAG_OFFSET_Z_MSB](#) 0x60
- #define [BNO055_GYR_OFFSET_X_LSB](#) 0x61
- #define [BNO055_GYR_OFFSET_X_MSB](#) 0x62
- #define [BNO055_GYR_OFFSET_Y_LSB](#) 0x63
- #define [BNO055_GYR_OFFSET_Y_MSB](#) 0x64
- #define [BNO055_GYR_OFFSET_Z_LSB](#) 0x65
- #define [BNO055_GYR_OFFSET_Z_MSB](#) 0x66
- #define [BNO055_ACC_RADIUS_LSB](#) 0x67
- #define [BNO055_ACC_RADIUS_MSB](#) 0x68
- #define [BNO055_MAG_RADIUS_LSB](#) 0x69
- #define [BNO055_MAG_RADIUS_MSB](#) 0x6A
- #define [BNO055_PAGE_ID](#) 0x07
- #define [BNO055_ACC_CONFIG](#) 0x08
- #define [BNO055_MAG_CONFIG](#) 0x09
- #define [BNO055_GYRO_CONFIG_0](#) 0x0A
- #define [BNO055_GYRO_CONFIG_1](#) 0x0B
- #define [BNO055_ACC_SLEEP_CONFIG](#) 0x0C
- #define [BNO055_GYR_SLEEP_CONFIG](#) 0x0D
- #define [BNO055_INT_MSK](#) 0x0F
- #define [BNO055_INT_EN](#) 0x10
- #define [BNO055_ACC_AM_THRES](#) 0x11
- #define [BNO055_ACC_INT_SETTINGS](#) 0x12
- #define [BNO055_ACC_HG_DURATION](#) 0x13
- #define [BNO055_ACC_HG_THRESH](#) 0x14
- #define [BNO055_ACC_NM_THRESH](#) 0x15
- #define [BNO055_ACC_NM_SET](#) 0x16
- #define [BNO055_GYR_INT_SETTINGS](#) 0x17
- #define [BNO055_GYR_HR_X_SET](#) 0x18
- #define [BNO055_GYR_DUR_X](#) 0x19
- #define [BNO055_GYR_HR_Y_SET](#) 0x1A
- #define [BNO055_GYR_DUR_Y](#) 0x1B
- #define [BNO055_GYR_HR_Z_SET](#) 0x1C
- #define [BNO055_GYR_DUR_Z](#) 0x1D
- #define [BNO055_GYR_AM_THRESH](#) 0x1E
- #define [BNO055_GYR_AM_SET](#) 0x1F

Enumerations

- enum [bno055_system_status_t](#) {
[BNO055_SYSTEM_STATUS_IDLE](#) = 0x00 , [BNO055_SYSTEM_STATUS_SYSTEM_ERROR](#) = 0x01 ,
[BNO055_SYSTEM_STATUS_INITIALIZING_PERIPHERALS](#) = 0x02 , [BNO055_SYSTEM_STATUS_SYSTEM_INITIALIZATION](#)
= 0x03 ,
[BNO055_SYSTEM_STATUS_EXECUTING_SELF_TEST](#) = 0x04 , [BNO055_SYSTEM_STATUS_FUSION_ALGO_RUNNING](#)
= 0x05 , [BNO055_SYSTEM_STATUS_FUSION ALOG_NOT_RUNNING](#) = 0x06 }
- enum [bno055_opmode_t](#) {
[BNO055_OPERATION_MODE_CONFIG](#) = 0x00 , [BNO055_OPERATION_MODE_ACCONLY](#) , [BNO055_OPERATION_MODE_MAGONLY](#) ,
[BNO055_OPERATION_MODE_GYRONLY](#) ,
[BNO055_OPERATION_MODE_ACCMAG](#) , [BNO055_OPERATION_MODE_ACCGYRO](#) , [BNO055_OPERATION_MODE_MAGACC](#) ,
[BNO055_OPERATION_MODE_AMG](#) ,
[BNO055_OPERATION_MODE_IMU](#) , [BNO055_OPERATION_MODE_COMPASS](#) , [BNO055_OPERATION_MODE_M4G](#) ,
[BNO055_OPERATION_MODE_NDOF_FMC_OFF](#) ,
[BNO055_OPERATION_MODE_NDOF](#) }

- enum `bno055_vector_type_t` {
`BNO055_VECTOR_ACCELEROMETER` = 0x08 , `BNO055_VECTOR_MAGNETOMETER` = 0x0E ,
`BNO055_VECTOR_GYROSCOPE` = 0x14 , `BNO055_VECTOR_EULER` = 0x1A ,
`BNO055_VECTOR_QUATERNION` = 0x20 , `BNO055_VECTOR_LINEARACCEL` = 0x28 , `BNO055_VECTOR_GRAVITY`
= 0x2E }
- enum `bno055_system_error_t` {
`BNO055_SYSTEM_ERROR_NO_ERROR` = 0x00 , `BNO055_SYSTEM_ERROR_PERIPHERAL_INITIALIZATION_ERROR`
= 0x01 , `BNO055_SYSTEM_ERROR_SYSTEM_INITIALIZATION_ERROR` = 0x02 , `BNO055_SYSTEM_ERROR_SELF_TEST`
= 0x03 ,
`BNO055_SYSTEM_ERROR_REG_MAP_VAL_OUT_OF_RANGE` = 0x04 , `BNO055_SYSTEM_ERROR_REG_MAP_ADDR_C`
= 0x05 , `BNO055_SYSTEM_ERROR_REG_MAP_WRITE_ERROR` = 0x06 , `BNO055_SYSTEM_ERROR_LOW_PWR_MODE`
= 0x07 ,
`BNO055_SYSTEM_ERROR_ACCEL_PWR_MODE_NOT_AVAILABLE` = 0x08 , `BNO055_SYSTEM_ERROR_FUSION_ALGO`
= 0x09 , `BNO055_SYSTEM_ERROR_SENSOR_CONF_ERROR` = 0x0A }
- enum `bno055_axis_map_representation_t` { `BNO055_AXIS_X` = 0x00 , `BNO055_AXIS_Y` = 0x01 ,
`BNO055_AXIS_Z` = 0x02 }
- enum `bno055_axis_map_sign_t` { `BNO055_AXIS_SIGN_POSITIVE` = 0x00 , `BNO055_AXIS_SIGN_NEGATIVE`
= 0x01 }

Functions

- void `bno055_writeData` (uint8_t reg, uint8_t data)
- void `bno055_readData` (uint8_t reg, uint8_t *data, uint8_t len)
- void `bno055_delay` (int time)
- void `bno055_reset` ()
- `bno055_opmode_t` `bno055_getOperationMode` ()
- void `bno055_setOperationMode` (`bno055_opmode_t` mode)
- void `bno055_setOperationModeConfig` ()
- void `bno055_setOperationModeNDOF` ()
- void `bno055_enableExternalCrystal` ()
- void `bno055_disableExternalCrystal` ()
- void `bno055_setup` ()
- int8_t `bno055_getTemp` ()
- uint8_t `bno055_getBootloaderRevision` ()
- uint8_t `bno055_getSystemStatus` ()
- uint8_t `bno055_getSystemError` ()
- int16_t `bno055_getSWRevision` ()
- `bno055_self_test_result_t` `bno055_getSelfTestResult` ()
- `bno055_calibration_state_t` `bno055_getCalibrationState` ()
- `bno055_calibration_data_t` `bno055_getCalibrationData` ()
- void `bno055_setCalibrationData` (`bno055_calibration_data_t` calData)
- `bno055_vector_t` `bno055_getVectorAccelerometer` ()
- `bno055_vector_t` `bno055_getVectorMagnetometer` ()
- `bno055_vector_t` `bno055_getVectorGyroscope` ()
- `bno055_vector_t` `bno055_getVectorEuler` ()
- `bno055_vector_t` `bno055_getVectorLinearAccel` ()
- `bno055_vector_t` `bno055_getVectorGravity` ()
- `bno055_vector_t` `bno055_getVectorQuaternion` ()
- void `bno055_setAxisMap` (`bno055_axis_map_t` axis)

6.2.1 Macro Definition Documentation

6.2.1.1 BNO055_ACC_AM_THRES

```
#define BNO055_ACC_AM_THRES 0x11
```

6.2.1.2 BNO055_ACC_CONFIG

```
#define BNO055_ACC_CONFIG 0x08
```

6.2.1.3 BNO055_ACC_DATA_X_LSB

```
#define BNO055_ACC_DATA_X_LSB 0x08
```

6.2.1.4 BNO055_ACC_DATA_X_MSB

```
#define BNO055_ACC_DATA_X_MSB 0x09
```

6.2.1.5 BNO055_ACC_DATA_Y_LSB

```
#define BNO055_ACC_DATA_Y_LSB 0x0A
```

6.2.1.6 BNO055_ACC_DATA_Y_MSB

```
#define BNO055_ACC_DATA_Y_MSB 0x0B
```

6.2.1.7 BNO055_ACC_DATA_Z_LSB

```
#define BNO055_ACC_DATA_Z_LSB 0x0C
```

6.2.1.8 BNO055_ACC_DATA_Z_MSB

```
#define BNO055_ACC_DATA_Z_MSB 0x0D
```

6.2.1.9 BNO055_ACC_HG_DURATION

```
#define BNO055_ACC_HG_DURATION 0x13
```

6.2.1.10 BNO055_ACC_HG_THRESH

```
#define BNO055_ACC_HG_THRESH 0x14
```

6.2.1.11 BNO055_ACC_ID

```
#define BNO055_ACC_ID 0x01
```

6.2.1.12 BNO055_ACC_INT_SETTINGS

```
#define BNO055_ACC_INT_SETTINGS 0x12
```

6.2.1.13 BNO055_ACC_NM_SET

```
#define BNO055_ACC_NM_SET 0x16
```

6.2.1.14 BNO055_ACC_NM_THRESH

```
#define BNO055_ACC_NM_THRESH 0x15
```

6.2.1.15 BNO055_ACC_OFFSET_X_LSB

```
#define BNO055_ACC_OFFSET_X_LSB 0x55
```

6.2.1.16 BNO055_ACC_OFFSET_X_MSB

```
#define BNO055_ACC_OFFSET_X_MSB 0x56
```

6.2.1.17 BNO055_ACC_OFFSET_Y_LSB

```
#define BNO055_ACC_OFFSET_Y_LSB 0x57
```

6.2.1.18 BNO055_ACC_OFFSET_Y_MSB

```
#define BNO055_ACC_OFFSET_Y_MSB 0x58
```

6.2.1.19 BNO055_ACC_OFFSET_Z_LSB

```
#define BNO055_ACC_OFFSET_Z_LSB 0x59
```

6.2.1.20 BNO055_ACC_OFFSET_Z_MSB

```
#define BNO055_ACC_OFFSET_Z_MSB 0x5A
```

6.2.1.21 BNO055_ACC_RADIUS_LSB

```
#define BNO055_ACC_RADIUS_LSB 0x67
```

6.2.1.22 BNO055_ACC_RADIUS_MSB

```
#define BNO055_ACC_RADIUS_MSB 0x68
```

6.2.1.23 BNO055_ACC_SLEEP_CONFIG

```
#define BNO055_ACC_SLEEP_CONFIG 0x0C
```

6.2.1.24 BNO055_AXIS_MAP_CONFIG

```
#define BNO055_AXIS_MAP_CONFIG 0x41
```

6.2.1.25 BNO055_AXIS_MAP_SIGN

```
#define BNO055_AXIS_MAP_SIGN 0x42
```

6.2.1.26 BNO055_BL_REV_ID

```
#define BNO055_BL_REV_ID 0x06
```

6.2.1.27 BNO055_CALIB_STAT

```
#define BNO055_CALIB_STAT 0x35
```

6.2.1.28 BNO055_CHIP_ID

```
#define BNO055_CHIP_ID 0x00
```

6.2.1.29 BNO055_EUL_HEADING_LSB

```
#define BNO055_EUL_HEADING_LSB 0x1A
```

6.2.1.30 BNO055_EUL_HEADING_MSB

```
#define BNO055_EUL_HEADING_MSB 0x1B
```

6.2.1.31 BNO055_EUL_PITCH_LSB

```
#define BNO055_EUL_PITCH_LSB 0x1E
```

6.2.1.32 BNO055_EUL_PITCH_MSB

```
#define BNO055_EUL_PITCH_MSB 0x1F
```

6.2.1.33 BNO055_EUL_ROLL_LSB

```
#define BNO055_EUL_ROLL_LSB 0x1C
```

6.2.1.34 BNO055_EUL_ROLL_MSB

```
#define BNO055_EUL_ROLL_MSB 0x1D
```

6.2.1.35 BNO055_GRV_DATA_X_LSB

```
#define BNO055_GRV_DATA_X_LSB 0x2E
```

6.2.1.36 BNO055_GRV_DATA_X_MSB

```
#define BNO055_GRV_DATA_X_MSB 0x2F
```

6.2.1.37 BNO055_GRV_DATA_Y_LSB

```
#define BNO055_GRV_DATA_Y_LSB 0x30
```

6.2.1.38 BNO055_GRV_DATA_Y_MSB

```
#define BNO055_GRV_DATA_Y_MSB 0x31
```

6.2.1.39 BNO055_GRV_DATA_Z_LSB

```
#define BNO055_GRV_DATA_Z_LSB 0x32
```

6.2.1.40 BNO055_GRV_DATA_Z_MSB

```
#define BNO055_GRV_DATA_Z_MSB 0x33
```

6.2.1.41 BNO055_GYR_AM_SET

```
#define BNO055_GYR_AM_SET 0x1F
```

6.2.1.42 BNO055_GYR_AM_THRESH

```
#define BNO055_GYR_AM_THRESH 0x1E
```

6.2.1.43 BNO055_GYR_DATA_X_LSB

```
#define BNO055_GYR_DATA_X_LSB 0x14
```

6.2.1.44 BNO055_GYR_DATA_X_MSB

```
#define BNO055_GYR_DATA_X_MSB 0x15
```

6.2.1.45 BNO055_GYR_DATA_Y_LSB

```
#define BNO055_GYR_DATA_Y_LSB 0x16
```

6.2.1.46 BNO055_GYR_DATA_Y_MSB

```
#define BNO055_GYR_DATA_Y_MSB 0x17
```

6.2.1.47 BNO055_GYR_DATA_Z_LSB

```
#define BNO055_GYR_DATA_Z_LSB 0x18
```

6.2.1.48 BNO055_GYR_DATA_Z_MSB

```
#define BNO055_GYR_DATA_Z_MSB 0x19
```

6.2.1.49 BNO055_GYR_DUR_X

```
#define BNO055_GYR_DUR_X 0x19
```

6.2.1.50 BNO055_GYR_DUR_Y

```
#define BNO055_GYR_DUR_Y 0x1B
```

6.2.1.51 BNO055_GYR_DUR_Z

```
#define BNO055_GYR_DUR_Z 0x1D
```


6.2.1.52 BNO055_GYR_HR_X_SET

```
#define BNO055_GYR_HR_X_SET 0x18
```

6.2.1.53 BNO055_GYR_HR_Y_SET

```
#define BNO055_GYR_HR_Y_SET 0x1A
```

6.2.1.54 BNO055_GYR_HR_Z_SET

```
#define BNO055_GYR_HR_Z_SET 0x1C
```

6.2.1.55 BNO055_GYR_INT_SETTINGS

```
#define BNO055_GYR_INT_SETTINGS 0x17
```

6.2.1.56 BNO055_GYR_OFFSET_X_LSB

```
#define BNO055_GYR_OFFSET_X_LSB 0x61
```

6.2.1.57 BNO055_GYR_OFFSET_X_MSB

```
#define BNO055_GYR_OFFSET_X_MSB 0x62
```

6.2.1.58 BNO055_GYR_OFFSET_Y_LSB

```
#define BNO055_GYR_OFFSET_Y_LSB 0x63
```

6.2.1.59 BNO055_GYR_OFFSET_Y_MSB

```
#define BNO055_GYR_OFFSET_Y_MSB 0x64
```

6.2.1.60 BNO055_GYR_OFFSET_Z_LSB

```
#define BNO055_GYR_OFFSET_Z_LSB 0x65
```

6.2.1.61 BNO055_GYR_OFFSET_Z_MSB

```
#define BNO055_GYR_OFFSET_Z_MSB 0x66
```

6.2.1.62 BNO055_GYR_SLEEP_CONFIG

```
#define BNO055_GYR_SLEEP_CONFIG 0x0D
```

6.2.1.63 BNO055_GYRO_CONFIG_0

```
#define BNO055_GYRO_CONFIG_0 0x0A
```

6.2.1.64 BNO055_GYRO_CONFIG_1

```
#define BNO055_GYRO_CONFIG_1 0x0B
```

6.2.1.65 BNO055_GYRO_ID

```
#define BNO055_GYRO_ID 0x03
```

6.2.1.66 BNO055_I2C_ADDR

```
#define BNO055_I2C_ADDR BNO055\_I2C\_ADDR\_LO
```

6.2.1.67 BNO055_I2C_ADDR_HI

```
#define BNO055_I2C_ADDR_HI 0x29
```

6.2.1.68 BNO055_I2C_ADDR_LO

```
#define BNO055_I2C_ADDR_LO 0x28
```

6.2.1.69 BNO055_ID

```
#define BNO055_ID (0xA0)
```

6.2.1.70 BNO055_INT_EN

```
#define BNO055_INT_EN 0x10
```

6.2.1.71 BNO055_INT_MSK

```
#define BNO055_INT_MSK 0x0F
```

6.2.1.72 BNO055_INT_STATUS

```
#define BNO055_INT_STATUS 0x37
```

6.2.1.73 BNO055_LIA_DATA_X_LSB

```
#define BNO055_LIA_DATA_X_LSB 0x28
```

6.2.1.74 BNO055_LIA_DATA_X_MSB

```
#define BNO055_LIA_DATA_X_MSB 0x29
```

6.2.1.75 BNO055_LIA_DATA_Y_LSB

```
#define BNO055_LIA_DATA_Y_LSB 0x2A
```

6.2.1.76 BNO055_LIA_DATA_Y_MSB

```
#define BNO055_LIA_DATA_Y_MSB 0x2B
```

6.2.1.77 BNO055_LIA_DATA_Z_LSB

```
#define BNO055_LIA_DATA_Z_LSB 0x2C
```

6.2.1.78 BNO055_LIA_DATA_Z_MSB

```
#define BNO055_LIA_DATA_Z_MSB 0x2D
```

6.2.1.79 BNO055_MAG_CONFIG

```
#define BNO055_MAG_CONFIG 0x09
```

6.2.1.80 BNO055_MAG_DATA_X_LSB

```
#define BNO055_MAG_DATA_X_LSB 0x0E
```

6.2.1.81 BNO055_MAG_DATA_X_MSB

```
#define BNO055_MAG_DATA_X_MSB 0x0F
```

6.2.1.82 BNO055_MAG_DATA_Y_LSB

```
#define BNO055_MAG_DATA_Y_LSB 0x10
```

6.2.1.83 BNO055_MAG_DATA_Y_MSB

```
#define BNO055_MAG_DATA_Y_MSB 0x11
```

6.2.1.84 BNO055_MAG_DATA_Z_LSB

```
#define BNO055_MAG_DATA_Z_LSB 0x12
```

6.2.1.85 BNO055_MAG_DATA_Z_MSB

```
#define BNO055_MAG_DATA_Z_MSB 0x13
```

6.2.1.86 BNO055_MAG_ID

```
#define BNO055_MAG_ID 0x02
```

6.2.1.87 BNO055_MAG_OFFSET_X_LSB

```
#define BNO055_MAG_OFFSET_X_LSB 0x5B
```

6.2.1.88 BNO055_MAG_OFFSET_X_MSB

```
#define BNO055_MAG_OFFSET_X_MSB 0x5C
```

6.2.1.89 BNO055_MAG_OFFSET_Y_LSB

```
#define BNO055_MAG_OFFSET_Y_LSB 0x5D
```

6.2.1.90 BNO055_MAG_OFFSET_Y_MSB

```
#define BNO055_MAG_OFFSET_Y_MSB 0x5E
```

6.2.1.91 BNO055_MAG_OFFSET_Z_LSB

```
#define BNO055_MAG_OFFSET_Z_LSB 0x5F
```

6.2.1.92 BNO055_MAG_OFFSET_Z_MSB

```
#define BNO055_MAG_OFFSET_Z_MSB 0x60
```

6.2.1.93 BNO055_MAG_RADIUS_LSB

```
#define BNO055_MAG_RADIUS_LSB 0x69
```

6.2.1.94 BNO055_MAG_RADIUS_MSB

```
#define BNO055_MAG_RADIUS_MSB 0x6A
```

6.2.1.95 BNO055_OPR_MODE

```
#define BNO055_OPR_MODE 0x3D
```

6.2.1.96 BNO055_PAGE_ID [1/2]

```
#define BNO055_PAGE_ID 0x07
```

6.2.1.97 BNO055_PAGE_ID [2/2]

```
#define BNO055_PAGE_ID 0x07
```

6.2.1.98 BNO055_PWR_MODE

```
#define BNO055_PWR_MODE 0x3E
```

6.2.1.99 BNO055_QUA_DATA_W_LSB

```
#define BNO055_QUA_DATA_W_LSB 0x20
```

6.2.1.100 BNO055_QUA_DATA_W_MSB

```
#define BNO055_QUA_DATA_W_MSB 0x21
```

6.2.1.101 BNO055_QUA_DATA_X_LSB

```
#define BNO055_QUA_DATA_X_LSB 0x22
```

6.2.1.102 BNO055_QUA_DATA_X_MSB

```
#define BNO055_QUA_DATA_X_MSB 0x23
```

6.2.1.103 BNO055_QUA_DATA_Y_LSB

```
#define BNO055_QUA_DATA_Y_LSB 0x24
```

6.2.1.104 BNO055_QUA_DATA_Y_MSB

```
#define BNO055_QUA_DATA_Y_MSB 0x25
```

6.2.1.105 BNO055_QUA_DATA_Z_LSB

```
#define BNO055_QUA_DATA_Z_LSB 0x26
```

6.2.1.106 BNO055_QUA_DATA_Z_MSB

```
#define BNO055_QUA_DATA_Z_MSB 0x27
```

6.2.1.107 BNO055_READ_TIMEOUT

```
#define BNO055_READ_TIMEOUT 100
```

6.2.1.108 BNO055_ST_RESULT

```
#define BNO055_ST_RESULT 0x36
```

6.2.1.109 BNO055_SW_REV_ID_LSB

```
#define BNO055_SW_REV_ID_LSB 0x04
```

6.2.1.110 BNO055_SW_REV_ID_MSB

```
#define BNO055_SW_REV_ID_MSB 0x05
```

6.2.1.111 BNO055_SYS_CLK_STATUS

```
#define BNO055_SYS_CLK_STATUS 0x38
```

6.2.1.112 BNO055_SYS_ERR

```
#define BNO055_SYS_ERR 0x3A
```

6.2.1.113 BNO055_SYS_STATUS

```
#define BNO055_SYS_STATUS 0x39
```

6.2.1.114 BNO055_SYS_TRIGGER

```
#define BNO055_SYS_TRIGGER 0x3F
```

6.2.1.115 BNO055_TEMP

```
#define BNO055_TEMP 0x34
```

6.2.1.116 BNO055_TEMP_SOURCE

```
#define BNO055_TEMP_SOURCE 0x40
```

6.2.1.117 BNO055_UNIT_SEL

```
#define BNO055_UNIT_SEL 0x3B
```

6.2.1.118 BNO055_WRITE_TIMEOUT

```
#define BNO055_WRITE_TIMEOUT 10
```

6.2.1.119 ERROR_BUS_OVERRUN_ERR

```
#define ERROR_BUS_OVERRUN_ERR 0x07
```

6.2.1.120 ERROR_BYTE

```
#define ERROR_BYTE 0xEE
```

6.2.1.121 ERROR_MAX_LEN_ERR

```
#define ERROR_MAX_LEN_ERR 0x08
```

6.2.1.122 ERROR_MIN_LEN_ERR

```
#define ERROR_MIN_LEN_ERR 0x09
```

6.2.1.123 ERROR_RECV_CHAR_TIMEOUT

```
#define ERROR_RECV_CHAR_TIMEOUT 0x0A
```

6.2.1.124 ERROR_REGMAP_INV_ADDR

```
#define ERROR_REGMAP_INV_ADDR 0x04
```

6.2.1.125 ERROR_REGMAP_WRITE_DIS

```
#define ERROR_REGMAP_WRITE_DIS 0x05
```

6.2.1.126 ERROR_WRITE_FAIL

```
#define ERROR_WRITE_FAIL 0x03
```

6.2.1.127 ERROR_WRITE_SUCCESS

```
#define ERROR_WRITE_SUCCESS 0x01
```

6.2.1.128 ERROR_WRONG_START_BYTE

```
#define ERROR_WRONG_START_BYTE 0x06
```

6.2.1.129 REG_READ

```
#define REG_READ 0x01
```

6.2.1.130 REG_WRITE

```
#define REG_WRITE 0x00
```

6.2.1.131 RESPONSE_BYTE

```
#define RESPONSE_BYTE 0xBB
```

6.2.1.132 START_BYTE

```
#define START_BYTE 0xAA
```

6.2.2 Enumeration Type Documentation

6.2.2.1 bno055_axis_map_representation_t

```
enum bno055\_axis\_map\_representation\_t
```


Enumerator

BNO055_AXIS↔ _X	
BNO055_AXIS↔ _Y	
BNO055_AXIS↔ _Z	

6.2.2.2 bno055_axis_map_sign_t

```
enum bno055_axis_map_sign_t
```

Enumerator

BNO055_AXIS_SIGN_POSITIVE	
BNO055_AXIS_SIGN_NEGATIVE	

6.2.2.3 bno055_opmode_t

```
enum bno055_opmode_t
```

Enumerator

BNO055_OPERATION_MODE_CONFIG	
BNO055_OPERATION_MODE_ACONLY	
BNO055_OPERATION_MODE_MAGONLY	
BNO055_OPERATION_MODE_GYRONLY	
BNO055_OPERATION_MODE_ACCMAG	
BNO055_OPERATION_MODE_ACCGYRO	
BNO055_OPERATION_MODE_MAGGYRO	
BNO055_OPERATION_MODE_AMG	
BNO055_OPERATION_MODE_IMU	
BNO055_OPERATION_MODE_COMPASS	
BNO055_OPERATION_MODE_M4G	
BNO055_OPERATION_MODE_NDOF_FMC_OFF	
BNO055_OPERATION_MODE_NDOF	

6.2.2.4 bno055_system_error_t

```
enum bno055_system_error_t
```

Enumerator

BNO055_SYSTEM_ERROR_NO_ERROR	
BNO055_SYSTEM_ERROR_PERIPHERAL_INITIALIZATION_ERROR	
BNO055_SYSTEM_ERROR_SYSTEM_INITIALIZATION_ERROR	

Enumerator

BNO055_SYSTEM_ERROR_SELF_TEST_FAILED	
BNO055_SYSTEM_ERROR_REG_MAP_VAL_OUT_OF_RANGE	
BNO055_SYSTEM_ERROR_REG_MAP_ADDR_OUT_OF_RANGE	
BNO055_SYSTEM_ERROR_REG_MAP_WRITE_ERROR	
BNO055_SYSTEM_ERROR_LOW_PWR_MODE_NOT_AVAILABLE_FOR_SELECTED_OPR_MODE	
BNO055_SYSTEM_ERROR_ACCEL_PWR_MODE_NOT_AVAILABLE	
BNO055_SYSTEM_ERROR_FUSION_ALGO_CONF_ERROR	
BNO055_SYSTEM_ERROR_SENSOR_CONF_ERROR	

6.2.2.5 bno055_system_status_t

```
enum bno055_system_status_t
```

Enumerator

BNO055_SYSTEM_STATUS_IDLE	
BNO055_SYSTEM_STATUS_SYSTEM_ERROR	
BNO055_SYSTEM_STATUS_INITIALIZING_PERIPHERALS	
BNO055_SYSTEM_STATUS_SYSTEM_INITIALIZATION	
BNO055_SYSTEM_STATUS_EXECUTING_SELF_TEST	
BNO055_SYSTEM_STATUS_FUSION_ALGO_RUNNING	
BNO055_SYSTEM_STATUS_FUSION_ALOG_NOT_RUNNING	

6.2.2.6 bno055_vector_type_t

```
enum bno055_vector_type_t
```

Enumerator

BNO055_VECTOR_ACCELEROMETER	
BNO055_VECTOR_MAGNETOMETER	
BNO055_VECTOR_GYROSCOPE	
BNO055_VECTOR_EULER	
BNO055_VECTOR_QUATERNION	
BNO055_VECTOR_LINEARACCEL	
BNO055_VECTOR_GRAVITY	

6.2.3 Function Documentation

6.2.3.1 bno055_delay()

```
void bno055_delay (
    int time )
```

6.2.3.2 bno055_disableExternalCrystal()

```
void bno055_disableExternalCrystal ( )
```

6.2.3.3 bno055_enableExternalCrystal()

```
void bno055_enableExternalCrystal ( )
```

6.2.3.4 bno055_getBootloaderRevision()

```
uint8_t bno055_getBootloaderRevision ( )
```

6.2.3.5 bno055_getCalibrationData()

```
bno055_calibration_data_t bno055_getCalibrationData ( )
```

6.2.3.6 bno055_getCalibrationState()

```
bno055_calibration_state_t bno055_getCalibrationState ( )
```

6.2.3.7 bno055_getOperationMode()

```
bno055_opmode_t bno055_getOperationMode ( )
```

6.2.3.8 bno055_getSelfTestResult()

```
bno055_self_test_result_t bno055_getSelfTestResult ( )
```

6.2.3.9 bno055_getSWRevision()

```
int16_t bno055_getSWRevision ( )
```

6.2.3.10 bno055_getSystemError()

```
uint8_t bno055_getSystemError ( )
```

6.2.3.11 bno055_getSystemStatus()

```
uint8_t bno055_getSystemStatus ( )
```

6.2.3.12 bno055_getTemp()

```
int8_t bno055_getTemp ( )
```

6.2.3.13 bno055_getVectorAccelerometer()

```
bno055_vector_t bno055_getVectorAccelerometer ( )
```

6.2.3.14 bno055_getVectorEuler()

```
bno055_vector_t bno055_getVectorEuler ( )
```

6.2.3.15 bno055_getVectorGravity()

```
bno055_vector_t bno055_getVectorGravity ( )
```

6.2.3.16 bno055_getVectorGyroscope()

```
bno055_vector_t bno055_getVectorGyroscope ( )
```

6.2.3.17 bno055_getVectorLinearAccel()

```
bno055_vector_t bno055_getVectorLinearAccel ( )
```

6.2.3.18 bno055_getVectorMagnetometer()

```
bno055_vector_t bno055_getVectorMagnetometer ( )
```

6.2.3.19 bno055_getVectorQuaternion()

```
bno055_vector_t bno055_getVectorQuaternion ( )
```

6.2.3.20 bno055_readData()

```
void bno055_readData (
    uint8_t reg,
    uint8_t * data,
    uint8_t len )
```

6.2.3.21 bno055_reset()

```
void bno055_reset ( )
```

6.2.3.22 bno055_setAxisMap()

```
void bno055_setAxisMap (
    bno055_axis_map_t axis )
```

6.2.3.23 bno055_setCalibrationData()

```
void bno055_setCalibrationData (
    bno055_calibration_data_t calData )
```

6.2.3.24 bno055_setOperationMode()

```
void bno055_setOperationMode (
    bno055_opmode_t mode )
```

6.2.3.25 bno055_setOperationModeConfig()

```
void bno055_setOperationModeConfig ( )
```

6.2.3.26 bno055_setOperationModeNDOF()

```
void bno055_setOperationModeNDOF ( )
```

6.2.3.27 bno055_setup()

```
void bno055_setup ( )
```

6.2.3.28 bno055_writeData()

```
void bno055_writeData (
    uint8_t reg,
    uint8_t data )
```

6.3 bno055.h

[Go to the documentation of this file.](#)

```

00001 /*
00002  * bno055.c
00003  *
00004  * Created on: Jun 5, 2024
00005  * Modifier: Conor Schott
00006  *
00007  * Description:
00008  * This file provides functions to interface with the BNO055 IMU sensor. It includes functions
00009  * to set the operation mode, read sensor data, manage calibration, and configure sensor settings.
00010  *
00011  * The original code was authored by Ivyknob and was modified to integrate with a different
00012  * main application that outputs sensor data differently.
00013  *
00014  * Original code source:
00015  * https://github.com/ivyknob/bno055_stm32
00016  */
00017
00018
00019 #ifndef BNO055_H_
00020 #define BNO055_H_
00021
00022 #ifdef __cplusplus
00023     extern "C" {
00024 #endif
00025 // #define FREERTOS_ENABLED true
00026
00027 #include <stdbool.h>
00028 #include <stdint.h>
00029 #include <stdio.h>
00030
00031 #define START_BYTE 0xAA
00032 #define RESPONSE_BYTE 0xBB
00033 #define ERROR_BYTE 0xEE
00034
00035 #define BNO055_I2C_ADDR_HI 0x29 //43
00036 #define BNO055_I2C_ADDR_LO 0x28 //42
00037 #define BNO055_I2C_ADDR BNO055_I2C_ADDR_LO
00038
00039 #define BNO055_READ_TIMEOUT 100
00040 #define BNO055_WRITE_TIMEOUT 10
00041
00042 #define ERROR_WRITE_SUCCESS 0x01 // Everything working as expected
00043 #define ERROR_WRITE_FAIL
00044     0x03 // Check connection, protocol settings and operation mode of BNO055 \
00045 #define ERROR_REGMAP_INV_ADDR 0x04 // Invalid register address
00046 #define ERROR_REGMAP_WRITE_DIS 0x05 // Register is read-only
00047 #define ERROR_WRONG_START_BYTE 0x06 // Check if the first byte
00048 #define ERROR_BUS_OVERRUN_ERR \
00049     0x07 // Resend the command, BNO055 was not able to clear the receive buffer \
00050 #define ERROR_MAX_LEN_ERR \
00051     0x08 // Split the command, max frame size can be up to 128 bytes \
00052 #define ERROR_MIN_LEN_ERR 0x09 // Min length of data is less than 1 \
00053 #define ERROR_RECV_CHAR_TIMEOUT \
00054     0x0A // Decrease the waiting time between sending of two bytes of one frame \
00055
00056 #define REG_WRITE 0x00
00057 #define REG_READ 0x01
00058
00059 // Page 0
00060 #define BNO055_ID (0xA0)
00061 #define BNO055_CHIP_ID 0x00 // value: 0xA0
00062 #define BNO055_ACC_ID 0x01 // value: 0xFB
00063 #define BNO055_MAG_ID 0x02 // value: 0x32
00064 #define BNO055_GYRO_ID 0x03 // value: 0x0F
00065 #define BNO055_SW_REV_ID_LSB 0x04 // value: 0x08
00066 #define BNO055_SW_REV_ID_MSB 0x05 // value: 0x03
00067 #define BNO055_BL_REV_ID 0x06 // N/A
00068 #define BNO055_PAGE_ID 0x07
00069 #define BNO055_ACC_DATA_X_LSB 0x08
00070 #define BNO055_ACC_DATA_X_MSB 0x09
00071 #define BNO055_ACC_DATA_Y_LSB 0x0A
00072 #define BNO055_ACC_DATA_Y_MSB 0x0B
00073 #define BNO055_ACC_DATA_Z_LSB 0x0C
00074 #define BNO055_ACC_DATA_Z_MSB 0x0D
00075 #define BNO055_MAG_DATA_X_LSB 0x0E
00076 #define BNO055_MAG_DATA_X_MSB 0x0F
00077 #define BNO055_MAG_DATA_Y_LSB 0x10
00078 #define BNO055_MAG_DATA_Y_MSB 0x11
00079 #define BNO055_MAG_DATA_Z_LSB 0x12
00080 #define BNO055_MAG_DATA_Z_MSB 0x13
00081 #define BNO055_GYR_DATA_X_LSB 0x14
00082 #define BNO055_GYR_DATA_X_MSB 0x15

```

```
00083 #define BNO055_GYR_DATA_Y_LSB 0x16
00084 #define BNO055_GYR_DATA_Y_MSB 0x17
00085 #define BNO055_GYR_DATA_Z_LSB 0x18
00086 #define BNO055_GYR_DATA_Z_MSB 0x19
00087 #define BNO055_EUL_HEADING_LSB 0x1A
00088 #define BNO055_EUL_HEADING_MSB 0x1B
00089 #define BNO055_EUL_ROLL_LSB 0x1C
00090 #define BNO055_EUL_ROLL_MSB 0x1D
00091 #define BNO055_EUL_PITCH_LSB 0x1E
00092 #define BNO055_EUL_PITCH_MSB 0x1F
00093 #define BNO055_QUA_DATA_W_LSB 0x20
00094 #define BNO055_QUA_DATA_W_MSB 0x21
00095 #define BNO055_QUA_DATA_X_LSB 0x22
00096 #define BNO055_QUA_DATA_X_MSB 0x23
00097 #define BNO055_QUA_DATA_Y_LSB 0x24
00098 #define BNO055_QUA_DATA_Y_MSB 0x25
00099 #define BNO055_QUA_DATA_Z_LSB 0x26
00100 #define BNO055_QUA_DATA_Z_MSB 0x27
00101 #define BNO055_LIA_DATA_X_LSB 0x28
00102 #define BNO055_LIA_DATA_X_MSB 0x29
00103 #define BNO055_LIA_DATA_Y_LSB 0x2A
00104 #define BNO055_LIA_DATA_Y_MSB 0x2B
00105 #define BNO055_LIA_DATA_Z_LSB 0x2C
00106 #define BNO055_LIA_DATA_Z_MSB 0x2D
00107 #define BNO055_GRV_DATA_X_LSB 0x2E
00108 #define BNO055_GRV_DATA_X_MSB 0x2F
00109 #define BNO055_GRV_DATA_Y_LSB 0x30
00110 #define BNO055_GRV_DATA_Y_MSB 0x31
00111 #define BNO055_GRV_DATA_Z_LSB 0x32
00112 #define BNO055_GRV_DATA_Z_MSB 0x33
00113 #define BNO055_TEMP 0x34
00114 #define BNO055_CALIB_STAT 0x35
00115 #define BNO055_ST_RESULT 0x36
00116 #define BNO055_INT_STATUS 0x37
00117 #define BNO055_SYS_CLK_STATUS 0x38
00118 #define BNO055_SYS_STATUS 0x39
00119 #define BNO055_SYS_ERR 0x3A
00120 #define BNO055_UNIT_SEL 0x3B
00121 #define BNO055_OPR_MODE 0x3D
00122 #define BNO055_PWR_MODE 0x3E
00123 #define BNO055_SYS_TRIGGER 0x3F
00124 #define BNO055_TEMP_SOURCE 0x40
00125 #define BNO055_AXIS_MAP_CONFIG 0x41
00126 #define BNO055_AXIS_MAP_SIGN 0x42
00127 #define BNO055_ACC_OFFSET_X_LSB 0x55
00128 #define BNO055_ACC_OFFSET_X_MSB 0x56
00129 #define BNO055_ACC_OFFSET_Y_LSB 0x57
00130 #define BNO055_ACC_OFFSET_Y_MSB 0x58
00131 #define BNO055_ACC_OFFSET_Z_LSB 0x59
00132 #define BNO055_ACC_OFFSET_Z_MSB 0x5A
00133 #define BNO055_MAG_OFFSET_X_LSB 0x5B
00134 #define BNO055_MAG_OFFSET_X_MSB 0x5C
00135 #define BNO055_MAG_OFFSET_Y_LSB 0x5D
00136 #define BNO055_MAG_OFFSET_Y_MSB 0x5E
00137 #define BNO055_MAG_OFFSET_Z_LSB 0x5F
00138 #define BNO055_MAG_OFFSET_Z_MSB 0x60
00139 #define BNO055_GYR_OFFSET_X_LSB 0x61
00140 #define BNO055_GYR_OFFSET_X_MSB 0x62
00141 #define BNO055_GYR_OFFSET_Y_LSB 0x63
00142 #define BNO055_GYR_OFFSET_Y_MSB 0x64
00143 #define BNO055_GYR_OFFSET_Z_LSB 0x65
00144 #define BNO055_GYR_OFFSET_Z_MSB 0x66
00145 #define BNO055_ACC_RADIUS_LSB 0x67
00146 #define BNO055_ACC_RADIUS_MSB 0x68
00147 #define BNO055_MAG_RADIUS_LSB 0x69
00148 #define BNO055_MAG_RADIUS_MSB 0x6A
00149 //
00150 // BNO055 Page 1
00151 #define BNO055_PAGE_ID 0x07
00152 #define BNO055_ACC_CONFIG 0x08
00153 #define BNO055_MAG_CONFIG 0x09
00154 #define BNO055_GYRO_CONFIG_0 0x0A
00155 #define BNO055_GYRO_CONFIG_1 0x0B
00156 #define BNO055_ACC_SLEEP_CONFIG 0x0C
00157 #define BNO055_GYR_SLEEP_CONFIG 0x0D
00158 #define BNO055_INT_MSK 0x0F
00159 #define BNO055_INT_EN 0x10
00160 #define BNO055_ACC_AM_THRES 0x11
00161 #define BNO055_ACC_INT_SETTINGS 0x12
00162 #define BNO055_ACC_HG_DURATION 0x13
00163 #define BNO055_ACC_HG_THRESH 0x14
00164 #define BNO055_ACC_NM_THRESH 0x15
00165 #define BNO055_ACC_NM_SET 0x16
00166 #define BNO055_GYR_INT_SETTINGS 0x17
00167 #define BNO055_GYR_HR_X_SET 0x18
00168 #define BNO055_GYR_DUR_X 0x19
00169 #define BNO055_GYR_HR_Y_SET 0x1A
```

```

00170 #define BNO055_GYR_DUR_Y 0x1B
00171 #define BNO055_GYR_HR_Z_SET 0x1C
00172 #define BNO055_GYR_DUR_Z 0x1D
00173 #define BNO055_GYR_AM_THRESH 0x1E
00174 #define BNO055_GYR_AM_SET 0x1F
00175
00176 enum bno055_system_status_t {
00177     BNO055_SYSTEM_STATUS_IDLE = 0x00,
00178     BNO055_SYSTEM_STATUS_SYSTEM_ERROR = 0x01,
00179     BNO055_SYSTEM_STATUS_INITIALIZING_PERIPHERALS = 0x02,
00180     BNO055_SYSTEM_STATUS_SYSTEM_INITIALIZATION = 0x03,
00181     BNO055_SYSTEM_STATUS_EXECUTING_SELF_TEST = 0x04,
00182     BNO055_SYSTEM_STATUS_FUSION_ALGO_RUNNING = 0x05,
00183     BNO055_SYSTEM_STATUS_FUSION_ALGO_NOT_RUNNING = 0x06
00184 };
00185
00186 typedef enum { // BNO-55 operation modes
00187     BNO055_OPERATION_MODE_CONFIG = 0x00,
00188     // Sensor Mode
00189     BNO055_OPERATION_MODE_ACCONLY,
00190     BNO055_OPERATION_MODE_MAGONLY,
00191     BNO055_OPERATION_MODE_GYRONLY,
00192     BNO055_OPERATION_MODE_ACCMAG,
00193     BNO055_OPERATION_MODE_ACCGYRO,
00194     BNO055_OPERATION_MODE_MAGGYRO,
00195     BNO055_OPERATION_MODE_AMG, // 0x07
00196     // Fusion Mode
00197     BNO055_OPERATION_MODE_IMU,
00198     BNO055_OPERATION_MODE_COMPASS,
00199     BNO055_OPERATION_MODE_M4G,
00200     BNO055_OPERATION_MODE_NDOF_FMC_OFF,
00201     BNO055_OPERATION_MODE_NDOF // 0x0C
00202 } bno055_opmode_t;
00203
00204 typedef struct {
00205     uint8_t mcuState;
00206     uint8_t gyrState;
00207     uint8_t magState;
00208     uint8_t accState;
00209 } bno055_self_test_result_t;
00210
00211 typedef struct {
00212     uint8_t sys;
00213     uint8_t gyro;
00214     uint8_t mag;
00215     uint8_t accel;
00216 } bno055_calibration_state_t;
00217
00218 typedef struct {
00219     int16_t x;
00220     int16_t y;
00221     int16_t z;
00222 } bno055_vector_xyz_int16_t;
00223
00224 typedef struct {
00225     bno055_vector_xyz_int16_t gyro;
00226     bno055_vector_xyz_int16_t mag;
00227     bno055_vector_xyz_int16_t accel;
00228 } bno055_calibration_offset_t;
00229
00230 typedef struct {
00231     uint16_t mag;
00232     uint16_t accel;
00233 } bno055_calibration_radius_t;
00234
00235 typedef struct {
00236     bno055_calibration_offset_t offset;
00237     bno055_calibration_radius_t radius;
00238 } bno055_calibration_data_t;
00239
00240 typedef struct {
00241     double w;
00242     double x;
00243     double y;
00244     double z;
00245 } bno055_vector_t;
00246
00247 typedef struct {
00248     uint8_t x;
00249     uint8_t x_sign;
00250     uint8_t y;
00251     uint8_t y_sign;
00252     uint8_t z;
00253     uint8_t z_sign;
00254 } bno055_axis_map_t;
00255
00256 typedef enum {

```



```

00257     BNO055_VECTOR_ACCELEROMETER = 0x08,    // Default: m/s2
00258     BNO055_VECTOR_MAGNETOMETER = 0x0E,    // Default: uT
00259     BNO055_VECTOR_GYROSCOPE = 0x14,       // Default: rad/s
00260     BNO055_VECTOR_EULER = 0x1A,          // Default: degrees
00261     BNO055_VECTOR_QUATERNION = 0x20,      // No units
00262     BNO055_VECTOR_LINEARACCEL = 0x28,     // Default: m/s2
00263     BNO055_VECTOR_GRAVITY = 0x2E         // Default: m/s2
00264 } bno055_vector_type_t;
00265
00266 enum bno055_system_error_t {
00267     BNO055_SYSTEM_ERROR_NO_ERROR = 0x00,
00268     BNO055_SYSTEM_ERROR_PERIPHERAL_INITIALIZATION_ERROR = 0x01,
00269     BNO055_SYSTEM_ERROR_SYSTEM_INITIALIZATION_ERROR = 0x02,
00270     BNO055_SYSTEM_ERROR_SELF_TEST_FAILED = 0x03,
00271     BNO055_SYSTEM_ERROR_REG_MAP_VAL_OUT_OF_RANGE = 0x04,
00272     BNO055_SYSTEM_ERROR_REG_MAP_ADDR_OUT_OF_RANGE = 0x05,
00273     BNO055_SYSTEM_ERROR_REG_MAP_WRITE_ERROR = 0x06,
00274     BNO055_SYSTEM_ERROR_LOW_PWR_MODE_NOT_AVAILABLE_FOR_SELECTED_OPR_MODE = 0x07,
00275     BNO055_SYSTEM_ERROR_ACCEL_PWR_MODE_NOT_AVAILABLE = 0x08,
00276     BNO055_SYSTEM_ERROR_FUSION_ALGO_CONF_ERROR = 0x09,
00277     BNO055_SYSTEM_ERROR_SENSOR_CONF_ERROR = 0x0A
00278 };
00279
00280 enum bno055_axis_map_representation_t {
00281     BNO055_AXIS_X = 0x00,
00282     BNO055_AXIS_Y = 0x01,
00283     BNO055_AXIS_Z = 0x02
00284 };
00285
00286 enum bno055_axis_map_sign_t {
00287     BNO055_AXIS_SIGN_POSITIVE = 0x00,
00288     BNO055_AXIS_SIGN_NEGATIVE = 0x01
00289 };
00290
00291 void bno055_writeData(uint8_t reg, uint8_t data);
00292 void bno055_readData(uint8_t reg, uint8_t *data, uint8_t len);
00293 void bno055_delay(int time);
00294
00295 void bno055_reset();
00296 bno055_opmode_t bno055_getOperationMode();
00297 void bno055_setOperationMode(bno055_opmode_t mode);
00298 void bno055_setOperationModeConfig();
00299 void bno055_setOperationModeNDOF();
00300 void bno055_enableExternalCrystal();
00301 void bno055_disableExternalCrystal();
00302 void bno055_setup();
00303
00304 int8_t bno055_getTemp();
00305
00306 uint8_t bno055_getBootloaderRevision();
00307 uint8_t bno055_getSystemStatus();
00308 uint8_t bno055_getSystemError();
00309 int16_t bno055_getSWRevision();
00310
00311 bno055_self_test_result_t bno055_getSelfTestResult();
00312 bno055_calibration_state_t bno055_getCalibrationState();
00313 bno055_calibration_data_t bno055_getCalibrationData();
00314 void bno055_setCalibrationData(bno055_calibration_data_t calData);
00315 bno055_vector_t bno055_getVectorAccelerometer();
00316 bno055_vector_t bno055_getVectorMagnetometer();
00317 bno055_vector_t bno055_getVectorGyroscope();
00318 bno055_vector_t bno055_getVectorEuler();
00319 bno055_vector_t bno055_getVectorLinearAccel();
00320 bno055_vector_t bno055_getVectorGravity();
00321 bno055_vector_t bno055_getVectorQuaternion();
00322 void bno055_setAxisMap(bno055_axis_map_t axis);
00323
00324 #ifdef __cplusplus
00325 }
00326 #endif
00327 #endif // BNO055_H_

```

6.4 bno055_stm32.h File Reference

```
#include "bno055.h"
```

Functions

- void [bno055_assignI2C](#) (I2C_HandleTypeDef *hi2c_device)

- void [bno055_delay](#) (int time)
- void [bno055_writeData](#) (uint8_t reg, uint8_t data)
- void [bno055_readData](#) (uint8_t reg, uint8_t *data, uint8_t len)

Variables

- I2C_HandleTypeDef * [_bno055_i2c_port](#)

6.4.1 Function Documentation

6.4.1.1 bno055_assignI2C()

```
void bno055_assignI2C (
    I2C_HandleTypeDef * hi2c_device )
```

6.4.1.2 bno055_delay()

```
void bno055_delay (
    int time )
```

6.4.1.3 bno055_readData()

```
void bno055_readData (
    uint8_t reg,
    uint8_t * data,
    uint8_t len )
```

6.4.1.4 bno055_writeData()

```
void bno055_writeData (
    uint8_t reg,
    uint8_t data )
```

6.4.2 Variable Documentation

6.4.2.1 _bno055_i2c_port

```
I2C_HandleTypeDef* _bno055_i2c_port
```

6.5 bno055_stm32.h

[Go to the documentation of this file.](#)

```

00001  /*
00002   * bno055.c
00003   *
00004   * Created on: Jun 5, 2024
00005   * Modifier: Conor Schott
00006   *
00007   * Description:
00008   * This file provides functions to interface with the BNO055 IMU sensor. It includes functions
00009   * to set the operation mode, read sensor data, manage calibration, and configure sensor settings.
00010   *
00011   * The original code was authored by Ivyknob and was modified to integrate with a different
00012   * main application that outputs sensor data differently.
00013   *
00014   * Original code source:
00015   * https://github.com/ivyknob/bno055_stm32
00016   */
00017
00018 #ifndef BNO055_STM32_H_
00019 #define BNO055_STM32_H_
00020
00021 #ifdef __cplusplus
00022     extern "C" {
00023 #endif
00024
00025 // #include "i2c.h"
00026
00027 #ifdef FREERTOS_ENABLED
00028     #include "FreeRTOS.h"
00029     #include "task.h"
00030     #include "cmsis_os.h"
00031 #endif
00032
00033 #include "bno055.h"
00034
00035 I2C_HandleTypeDef *_bno055_i2c_port;
00036
00037 void bno055_assignI2C(I2C_HandleTypeDef *hi2c_device) {
00038     _bno055_i2c_port = hi2c_device;
00039 }
00040
00041 void bno055_delay(int time) {
00042     #ifdef FREERTOS_ENABLED
00043         osDelay(time);
00044     #else
00045         HAL_Delay(time);
00046     #endif
00047 }
00048
00049 void bno055_writeData(uint8_t reg, uint8_t data) {
00050     uint8_t txdata[2] = {reg, data};
00051     uint8_t status;
00052     status = HAL_I2C_Master_Transmit(_bno055_i2c_port, BNO055_I2C_ADDR < 1,
00053                                     txdata, sizeof(txdata), 10);
00054     if (status == HAL_OK) {
00055         return;
00056     }
00057
00058     if (status == HAL_ERROR) {
00059         printf("HAL_I2C_Master_Transmit HAL_ERROR\r\n");
00060     } else if (status == HAL_TIMEOUT) {
00061         printf("HAL_I2C_Master_Transmit HAL_TIMEOUT\r\n");
00062     } else if (status == HAL_BUSY) {
00063         printf("HAL_I2C_Master_Transmit HAL_BUSY\r\n");
00064     } else {
00065         printf("Unknown status data %d", status);
00066     }
00067
00068     uint32_t error = HAL_I2C_GetError(_bno055_i2c_port);
00069     if (error == HAL_I2C_ERROR_NONE) {
00070         return;
00071     } else if (error == HAL_I2C_ERROR_BERR) {
00072         printf("HAL_I2C_ERROR_BERR\r\n");
00073     } else if (error == HAL_I2C_ERROR_ARLO) {
00074         printf("HAL_I2C_ERROR_ARLO\r\n");
00075     } else if (error == HAL_I2C_ERROR_AF) {
00076         printf("HAL_I2C_ERROR_AF\r\n");
00077     } else if (error == HAL_I2C_ERROR_OVR) {
00078         printf("HAL_I2C_ERROR_OVR\r\n");
00079     } else if (error == HAL_I2C_ERROR_DMA) {
00080         printf("HAL_I2C_ERROR_DMA\r\n");
00081     } else if (error == HAL_I2C_ERROR_TIMEOUT) {
00082         printf("HAL_I2C_ERROR_TIMEOUT\r\n");

```

```

00083     }
00084
00085     HAL_I2C_StateTypeDef state = HAL_I2C_GetState(_bno055_i2c_port);
00086     if (state == HAL_I2C_STATE_RESET) {
00087         printf("HAL_I2C_STATE_RESET\r\n");
00088     } else if (state == HAL_I2C_STATE_READY) {
00089         printf("HAL_I2C_STATE_RESET\r\n");
00090     } else if (state == HAL_I2C_STATE_BUSY) {
00091         printf("HAL_I2C_STATE_BUSY\r\n");
00092     } else if (state == HAL_I2C_STATE_BUSY_TX) {
00093         printf("HAL_I2C_STATE_BUSY_TX\r\n");
00094     } else if (state == HAL_I2C_STATE_BUSY_RX) {
00095         printf("HAL_I2C_STATE_BUSY_RX\r\n");
00096     } else if (state == HAL_I2C_STATE_LISTEN) {
00097         printf("HAL_I2C_STATE_LISTEN\r\n");
00098     } else if (state == HAL_I2C_STATE_BUSY_TX_LISTEN) {
00099         printf("HAL_I2C_STATE_BUSY_TX_LISTEN\r\n");
00100     } else if (state == HAL_I2C_STATE_BUSY_RX_LISTEN) {
00101         printf("HAL_I2C_STATE_BUSY_RX_LISTEN\r\n");
00102     } else if (state == HAL_I2C_STATE_ABORT) {
00103         printf("HAL_I2C_STATE_ABORT\r\n");
00104     } else if (state == HAL_I2C_STATE_TIMEOUT) {
00105         printf("HAL_I2C_STATE_TIMEOUT\r\n");
00106     } else if (state == HAL_I2C_STATE_ERROR) {
00107         printf("HAL_I2C_STATE_ERROR\r\n");
00108     }
00109     // while (HAL_I2C_GetState(_bno055_i2c_port) != HAL_I2C_STATE_READY) {}
00110     // return;
00111 }
00112
00113 void bno055_readData(uint8_t reg, uint8_t *data, uint8_t len) {
00114     HAL_I2C_Master_Transmit(_bno055_i2c_port, BNO055_I2C_ADDR < 1, &reg, 1,
00115                             100);
00116     HAL_I2C_Master_Receive(_bno055_i2c_port, BNO055_I2C_ADDR < 1, data, len,
00117                             100);
00118     // HAL_I2C_Mem_Read(_bno055_i2c_port, BNO055_I2C_ADDR_LO<1, reg,
00119     // I2C_MEMADD_SIZE_8BIT, data, len, 100);
00120 }
00121
00122 #ifdef __cplusplus
00123 }
00124 #endif
00125
00126 #endif // BNO055_STM32_H_

```

6.6 fsr_controller.c File Reference

```

#include "fsr_controller.h"
#include <stdio.h>
#include <string.h>

```

Functions

- uint32_t [map](#) (uint32_t x, uint32_t in_min, uint32_t in_max, uint32_t out_min, uint32_t out_max)
Maps a value from one range to another.
- uint32_t [get_averaged_adc_value](#) (ADC_HandleTypeDef *hadc)
Gets the average ADC value over a specified number of samples.
- uint32_t [apply_low_pass_filter](#) (uint32_t new_value)
Applies a low-pass filter to an input value.

6.6.1 Function Documentation

6.6.1.1 [apply_low_pass_filter\(\)](#)

```

uint32_t apply_low_pass_filter (
    uint32_t new_value )

```

Applies a low-pass filter to an input value.

Applies a low-pass filter to the input value.

Parameters

<i>new_value</i>	New input value to filter.
------------------	----------------------------

Return values

<i>Filtered</i>	value.
-----------------	--------

6.6.1.2 get_averaged_adc_value()

```
uint32_t get_averaged_adc_value (
    ADC_HandleTypeDef * hadc )
```

Gets the average ADC value over a specified number of samples.

Reads and averages the ADC value over a number of samples.

Parameters

<i>hadc</i>	Pointer to the ADC handle.
-------------	----------------------------

Return values

<i>Average</i>	ADC value.
----------------	------------

6.6.1.3 map()

```
uint32_t map (
    uint32_t x,
    uint32_t in_min,
    uint32_t in_max,
    uint32_t out_min,
    uint32_t out_max )
```

Maps a value from one range to another.

Maps a value from one range to another linearly.

File Name : [fsr_controller.c](#) Description : This file provides functions for FSR (Force-Sensitive Resistor) control.

Parameters

<i>x</i>	Input value to be mapped.
<i>in_min</i>	Minimum value of the input range.
<i>in_max</i>	Maximum value of the input range.
<i>out_min</i>	Minimum value of the output range.
<i>out_max</i>	Maximum value of the output range.

Return values

<i>Mapped</i>	value in the output range.
---------------	----------------------------

6.7 fsr_controller.h File Reference

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

Functions

- uint32_t [map](#) (uint32_t x, uint32_t in_min, uint32_t in_max, uint32_t out_min, uint32_t out_max)
Maps a value from one range to another linearly.
- uint32_t [get_averaged_adc_value](#) (ADC_HandleTypeDef *hadc)
Reads and averages the ADC value over a number of samples.
- uint32_t [apply_low_pass_filter](#) (uint32_t new_value)
Applies a low-pass filter to the input value.

6.7.1 Function Documentation

6.7.1.1 apply_low_pass_filter()

```
uint32_t apply_low_pass_filter (
    uint32_t new_value )
```

Applies a low-pass filter to the input value.

A simple exponential moving average filter is applied with a fixed smoothing factor.

Parameters

<i>new_value</i>	New input value to filter.
------------------	----------------------------

Returns

Filtered output value.

Applies a low-pass filter to the input value.

Parameters

<i>new_value</i>	New input value to filter.
------------------	----------------------------

Return values

<i>Filtered</i>	value.
-----------------	--------

6.7.1.2 get_averaged_adc_value()

```
uint32_t get_averaged_adc_value (
    ADC_HandleTypeDef * hadc )
```

Reads and averages the ADC value over a number of samples.

Parameters

<i>hadc</i>	ADC handle.
-------------	-------------

Returns

Averaged ADC value.

Reads and averages the ADC value over a number of samples.

Parameters

<i>hadc</i>	Pointer to the ADC handle.
-------------	----------------------------

Return values

<i>Average</i>	ADC value.
----------------	------------

6.7.1.3 map()

```
uint32_t map (
    uint32_t x,
    uint32_t in_min,
    uint32_t in_max,
    uint32_t out_min,
    uint32_t out_max )
```

Maps a value from one range to another linearly.

Parameters

<i>x</i>	Input value to be mapped.
<i>in_min</i>	Lower bound of the input range.
<i>in_max</i>	Upper bound of the input range.
<i>out_min</i>	Lower bound of the output range.
<i>out_max</i>	Upper bound of the output range.

Returns

Mapped value in the output range.

Maps a value from one range to another linearly.

File Name : [fsr_controller.c](#) Description : This file provides functions for FSR (Force-Sensitive Resistor) control.

Parameters

<i>x</i>	Input value to be mapped.
<i>in_min</i>	Minimum value of the input range.
<i>in_max</i>	Maximum value of the input range.
<i>out_min</i>	Minimum value of the output range.
<i>out_max</i>	Maximum value of the output range.

Return values

<i>Mapped</i>	value in the output range.
---------------	----------------------------

6.8 fsr_controller.h

[Go to the documentation of this file.](#)

```
00001 /*
00002  * fsr_controller.h
00003  *
00004  * Created on: June 5, 2024
00005  * Author: Conor Schott
00006  *
00007  * Description:
00008  * This file provides functions for controlling an FSR (Force-Sensitive Resistor) and
00009  * processing its analog input using an STM32 microcontroller. It includes functions
00010  * for mapping ADC values, averaging ADC readings, and applying a low-pass filter.
00011  *
00012  * The original code was authored by Conor Schott and was modified to integrate with a different
00013  * main application that outputs FSR controller differently.
00014  */
00015
00016 #ifndef FSR_CONTROLLER_H_
00017 #define FSR_CONTROLLER_H_
00018
00019 #include "main.h"
00020
00031 uint32_t map(uint32_t x, uint32_t in_min, uint32_t in_max, uint32_t out_min, uint32_t out_max);
00032
00039 uint32_t get_averaged_adc_value(ADC_HandleTypeDef *hadc);
00040
00049 uint32_t apply_low_pass_filter(uint32_t new_value);
00050
00051 #endif /* FSR_CONTROLLER_H_ */
```

6.9 main.c File Reference

: Main program body

```
#include "main.h"
#include "Motor_Driver.h"
#include "fsr_controller.h"
#include "Receiver.h"
#include "bno055_stm32.h"
#include <stdio.h>
#include <stdint.h>
#include <string.h>
#include <stdlib.h>
#include <ctype.h>
```

Macros

- `#define FILTER_SIZE 10`

Functions

- `int isRadioOn (RCReceiver *rc)`
- `void SystemClock_Config (void)`
- `int main (void)`

The application entry point.

Variables

- `ADC_HandleTypeDef hadc1`
- `I2C_HandleTypeDef hi2c1`
- `DMA_HandleTypeDef hdma_i2c1_rx`
- `DMA_HandleTypeDef hdma_i2c1_tx`
- `TIM_HandleTypeDef htim1`
- `TIM_HandleTypeDef htim2`
- `TIM_HandleTypeDef htim3`
- `char message [150]`
- `RCReceiver receiver`
- `RCReceiver * pointer_rc = &receiver`
- `MotorDriver motor1`
- `MotorDriver motor2`
- `uint32_t adc_value = 0`
- `uint32_t servo_speed = 0`
- `uint32_t threshold = 100`
- `uint32_t hysteresis = 50`
- `int state = 0`
- `int press_detected = 0`
- `float euler_y [FILTER_SIZE] = {0}`
- `float euler_z [FILTER_SIZE] = {0}`
- `uint8_t euler_index = 0`
- `float sum_y = 0`
- `float sum_z = 0`
- `int16_t x_position = 0`
- `int16_t y_position = 0`

6.9.1 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.

6.9.2 Macro Definition Documentation

6.9.2.1 FILTER_SIZE

```
#define FILTER_SIZE 10
```

6.9.3 Function Documentation

6.9.3.1 isRadioOn()

```
int isRadioOn (  
    RCTransmitter * rc )
```

6.9.3.2 main()

```
int main (  
    void )
```

The application entry point.

Return values

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

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

ADC1 Initialization Function

Parameters

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

Return values

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

Configure the global features of the ADC (Clock, Resolution, Data Alignment and number of conversion)

Configure for the selected ADC regular channel its corresponding rank in the sequencer and its sample time.

I2C1 Initialization Function

Parameters

None	
------	--

Return values

None	
------	--

TIM1 Initialization Function

Parameters

None	
------	--

Return values

None	
------	--

TIM2 Initialization Function

Parameters

None	
------	--

Return values

None	
------	--

TIM3 Initialization Function

Parameters

None	
------	--

Return values

None	
------	--

Enable DMA controller clock

GPIO Initialization Function

Parameters

None	
------	--

Return values

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

This function is executed in case of error occurrence.

Return values

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

6.9.3.3 SystemClock_Config()

```
void SystemClock_Config (
    void )
```

6.9.4 Variable Documentation

6.9.4.1 adc_value

```
uint32_t adc_value = 0
```

6.9.4.2 euler_index

```
uint8_t euler_index = 0
```

6.9.4.3 euler_y

```
float euler_y[FILTER_SIZE] = {0}
```

6.9.4.4 euler_z

```
float euler_z[FILTER_SIZE] = {0}
```

6.9.4.5 hadc1

```
ADC_HandleTypeDef hadc1
```

6.9.4.6 hdma_i2c1_rx

```
DMA_HandleTypeDef hdma_i2c1_rx
```

6.9.4.7 hdma_i2c1_tx

```
DMA_HandleTypeDef hdma_i2c1_tx
```

6.9.4.8 hi2c1

```
I2C_HandleTypeDef hi2c1
```

6.9.4.9 htim1

```
TIM_HandleTypeDef htim1
```

6.9.4.10 htim2

```
TIM_HandleTypeDef htim2
```

6.9.4.11 htim3

```
TIM_HandleTypeDef htim3
```

6.9.4.12 hysteresis

```
uint32_t hysteresis = 50
```

6.9.4.13 message

```
char message[150]
```

6.9.4.14 motor1

```
MotorDriver motor1
```

6.9.4.15 motor2

```
MotorDriver motor2
```

6.9.4.16 pointer_rc

```
RCReceiver* pointer_rc = &receiver
```

6.9.4.17 press_detected

```
int press_detected = 0
```

6.9.4.18 receiver

```
RCTReceiver receiver
```

Initial value:

```
= {  
    .Trigger_Channel1_Rise = TIM_CHANNEL_1,  
    .Trigger_Channel1_Fall = TIM_CHANNEL_2,  
    .Trigger_Channel2_Rise = TIM_CHANNEL_3,  
    .Trigger_Channel2_Fall = TIM_CHANNEL_4,  
    .htim = &htim3,  
    .nominalPWM = 1521,  
    .highPWM = 2042,  
    .lowPWM = 1019  
}
```

6.9.4.19 servo_speed

```
uint32_t servo_speed = 0
```

6.9.4.20 state

```
int state = 0
```

6.9.4.21 sum_y

```
float sum_y = 0
```

6.9.4.22 sum_z

```
float sum_z = 0
```

6.9.4.23 threshold

```
uint32_t threshold = 100
```

6.9.4.24 x_position

```
int16_t x_position = 0
```

6.9.4.25 y_position

```
int16_t y_position = 0
```

6.10 Motor_Driver.c File Reference

```
#include "Motor_Driver.h"
#include "stm32f4xx_hal_tim.h"
```

Macros

- #define [MAX_DUTY_CYCLE](#) 3999

Functions

- void [motor_enable](#) ([MotorDriver](#) *motor)
Enables PWM output for both forward and backward motion.
- void [motor_disable](#) ([MotorDriver](#) *motor)
Disables PWM output for both forward and backward motion.
- void [motor_set_duty_cycle](#) ([MotorDriver](#) *motor, [int16_t](#) duty_cycle)
Sets the duty cycle for the motor.

6.10.1 Macro Definition Documentation

6.10.1.1 MAX_DUTY_CYCLE

```
#define MAX_DUTY_CYCLE 3999
```

6.10.2 Function Documentation

6.10.2.1 motor_disable()

```
void motor_disable (  
    MotorDriver * motor )
```

Disables PWM output for both forward and backward motion.

Parameters

<i>motor</i>	Pointer to MotorDriver structure.
--------------	---

6.10.2.2 motor_enable()

```
void motor_enable (  
    MotorDriver * motor )
```

Enables PWM output for both forward and backward motion.

Parameters

<i>motor</i>	Pointer to MotorDriver structure.
--------------	---

6.10.2.3 motor_set_duty_cycle()

```
void motor_set_duty_cycle (
    MotorDriver * motor,
    int16_t duty_cycle )
```

Sets the duty cycle for the motor.

The duty cycle can be positive (forward motion) or negative (backward motion). If the duty cycle is outside the valid range (-MAX_DUTY_CYCLE to MAX_DUTY_CYCLE), it is capped to the maximum or minimum value.

Parameters

<i>motor</i>	Pointer to MotorDriver structure.
<i>duty_cycle</i>	Desired duty cycle, ranging from -MAX_DUTY_CYCLE to MAX_DUTY_CYCLE.

6.11 Motor_Driver.h File Reference

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

Data Structures

- struct [MotorDriver](#)

Functions

- void [motor_enable](#) ([MotorDriver](#) *motor)
Enables PWM output for both forward and backward motion.
- void [motor_disable](#) ([MotorDriver](#) *motor)
Disables PWM output for both forward and backward motion.
- void [motor_set_duty_cycle](#) ([MotorDriver](#) *motor, int16_t duty_cycle)
Sets the duty cycle for the motor.

6.11.1 Function Documentation

6.11.1.1 motor_disable()

```
void motor_disable (
    MotorDriver * motor )
```

Disables PWM output for both forward and backward motion.

Parameters

<i>motor</i>	Pointer to MotorDriver structure.
--------------	---

6.11.1.2 motor_enable()

```
void motor_enable (
    MotorDriver * motor )
```

Enables PWM output for both forward and backward motion.

Parameters

<i>motor</i>	Pointer to MotorDriver structure.
--------------	---

6.11.1.3 motor_set_duty_cycle()

```
void motor_set_duty_cycle (
    MotorDriver * motor,
    int16_t duty_cycle )
```

Sets the duty cycle for the motor.

The duty cycle can be positive (forward motion) or negative (backward motion). If the duty cycle is outside the valid range (-MAX_DUTY_CYCLE to MAX_DUTY_CYCLE), it is capped to the maximum or minimum value.

Parameters

<i>motor</i>	Pointer to MotorDriver structure.
<i>duty_cycle</i>	Desired duty cycle, ranging from -MAX_DUTY_CYCLE to MAX_DUTY_CYCLE.

6.12 Motor_Driver.h

[Go to the documentation of this file.](#)

```
00001 /*
00002  * motor_driver.h
00003  *
00004  * Created on: June 5, 2024
00005  * Author: Conor Schott
00006  *
00007  * Description:
00008  * This file provides an abstraction for controlling a DC motor using STM32 HAL.
00009  * It defines a MotorDriver structure and functions to enable/disable the motor
00010  * and set its duty cycle.
00011  *
00012  * The original code was authored by Conor Schott and was modified to integrate with a different
00013  * main application that outputs motor control differently.
00014  */
00015
00016 #ifndef MOTOR_DRIVER_H
00017 #define MOTOR_DRIVER_H
00018
00019 #include "stm32f4xx_hal.h"
00020
```

```

00021 typedef struct {
00022     uint32_t forward_channel; // Timer channel for forward motion
00023     uint32_t backward_channel; // Timer channel for backward motion
00024     TIM_HandleTypeDef *htim; // Pointer to the TIM_HandleTypeDef structure for PWM control
00025 } MotorDriver;
00026
00032 void motor_enable(MotorDriver *motor);
00033
00039 void motor_disable(MotorDriver *motor);
00040
00051 void motor_set_duty_cycle(MotorDriver *motor, int16_t duty_cycle);
00052
00053 #endif /* MOTOR_DRIVER_H */

```

6.13 Receiver.c File Reference

```

#include "Receiver.h"
#include <string.h>
#include <stdlib.h>

```

Macros

- #define `MESSAGE_LENGTH` 100

Functions

- `RCReceiver * initializeRCReceiver` (TIM_HandleTypeDef *htim)
Initializes the RC Receiver with the given Timer handle.
- void `startRCReceiverCapture` (RCReceiver *rc)
Starts the RC Receiver capture process.
- void `calculateTriggerValueCallback` (RCReceiver *rc, uint16_t previousValue)
Calculates the trigger value based on the previous value and thresholds.
- void `calculateWheelValueCallback` (RCReceiver *rc, uint16_t previousValue)
Calculates the wheel value based on the previous value and thresholds.

Variables

- uint16_t `g_trigger_val` = 0
- uint16_t `g_wheel_val` = 0

6.13.1 Macro Definition Documentation

6.13.1.1 MESSAGE_LENGTH

```
#define MESSAGE_LENGTH 100
```

6.13.2 Function Documentation

6.13.2.1 calculateTriggerValueCallback()

```

void calculateTriggerValueCallback (
    RCTReceiver * rc,
    uint16_t previousValue )

```

Calculates the trigger value based on the previous value and thresholds.

Parameters

<i>rc</i>	Pointer to the RCReceiver structure.
<i>previousValue</i>	Previous captured value.

6.13.2.2 calculateWheelValueCallback()

```
void calculateWheelValueCallback (
    RCReceiver * rc,
    uint16_t previousValue )
```

Calculates the wheel value based on the previous value and thresholds.

Parameters

<i>rc</i>	Pointer to the RCReceiver structure.
<i>previousValue</i>	Previous captured value.

6.13.2.3 initializeRCReceiver()

```
RCReceiver * initializeRCReceiver (
    TIM_HandleTypeDef * htim )
```

Initializes the RC Receiver with the given Timer handle.

Parameters

<i>htim</i>	Pointer to the Timer handle.
-------------	------------------------------

Returns

Pointer to the initialized [RCReceiver](#) structure, or NULL if initialization fails.

6.13.2.4 startRCReceiverCapture()

```
void startRCReceiverCapture (
    RCReceiver * rc )
```

Starts the RC Receiver capture process.

Starts interrupt-based capture on all configured channels.

Parameters

<i>rc</i>	Pointer to the RCReceiver structure.
-----------	--

6.13.3 Variable Documentation

6.13.3.1 g_trigger_val

```
uint16_t g_trigger_val = 0
```

6.13.3.2 g_wheel_val

```
uint16_t g_wheel_val = 0
```

6.14 Receiver.h File Reference

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

Data Structures

- struct [RCReceiver](#)
Structure to hold receiver configuration and state.

Macros

- #define [MESSAGE_LENGTH](#) 100

Functions

- [RCReceiver](#) * [initializeRCReceiver](#) (TIM_HandleTypeDef *htim)
Initializes the RC Receiver with the given Timer handle.
- void [startRCReceiverCapture](#) ([RCReceiver](#) *rc)
Starts the RC Receiver capture process.
- void [calculateTriggerValueCallback](#) ([RCReceiver](#) *rc, uint16_t previousValue)
Calculates the trigger value based on the previous value and thresholds.
- void [calculateWheelValueCallback](#) ([RCReceiver](#) *rc, uint16_t previousValue)
Calculates the wheel value based on the previous value and thresholds.

6.14.1 Macro Definition Documentation

6.14.1.1 MESSAGE_LENGTH

```
#define MESSAGE_LENGTH 100
```

6.14.2 Function Documentation

6.14.2.1 calculateTriggerValueCallback()

```
void calculateTriggerValueCallback (
    RCReceiver * rc,
    uint16_t previousValue )
```

Calculates the trigger value based on the previous value and thresholds.

Parameters

<i>rc</i>	Pointer to the RCReceiver structure.
<i>previousValue</i>	Previous captured value.

6.14.2.2 calculateWheelValueCallback()

```
void calculateWheelValueCallback (
    RCReceiver * rc,
    uint16_t previousValue )
```

Calculates the wheel value based on the previous value and thresholds.

Parameters

<i>rc</i>	Pointer to the RCReceiver structure.
<i>previousValue</i>	Previous captured value.

6.14.2.3 initializeRCReceiver()

```
RCReceiver * initializeRCReceiver (
    TIM_HandleTypeDef * htim )
```

Initializes the RC Receiver with the given Timer handle.

Parameters

<i>htim</i>	Pointer to the Timer handle.
-------------	------------------------------

Returns

Pointer to the initialized [RCReceiver](#) structure, or NULL if initialization fails.

6.14.2.4 startRCReceiverCapture()

```
void startRCReceiverCapture (
    RCReceiver * rc )
```

Starts the RC Receiver capture process.

Starts interrupt-based capture on all configured channels.

Parameters

<i>rc</i>	Pointer to the RCReceiver structure.
-----------	--

6.15 Receiver.h

[Go to the documentation of this file.](#)

```

00001 /*
00002  * Receiver.h
00003  *
00004  * Created on: June 13, 2024
00005  * Author: Conor Schott
00006  *
00007  * Description:
00008  * This file defines a receiver module for capturing PWM signals using an STM32 microcontroller.
00009  * It includes functions for initializing the receiver, starting the capture, and callbacks for
00010  * calculating trigger and wheel values from PWM signals.
00011  *
00012  * The original code was authored by Conor Schott and is part of a larger system.
00013  */
00014
00015 #ifndef RECEIVER_H_
00016 #define RECEIVER_H_
00017
00018 #include <stdint.h>
00019 #include "stm32f4xx_hal.h"
00020
00021 #define MESSAGE_LENGTH 100
00022
00023 typedef struct {
00024     TIM_HandleTypeDef* htim;
00025     uint32_t nominalPWM;
00026     uint32_t highPWM;
00027     uint32_t lowPWM;
00028     uint16_t Trigger_Channel1_Rise;
00029     uint16_t Trigger_Channel1_Fall;
00030     uint16_t Trigger_Channel2_Rise;
00031     uint16_t Trigger_Channel2_Fall;
00032     uint16_t message[MESSAGE_LENGTH];
00033 } RCReceiver;
00034
00035 RCReceiver* initializeRCReceiver(TIM_HandleTypeDef* htim);
00036
00037 void startRCReceiverCapture(RCReceiver* rc);
00038
00039 void calculateTriggerValueCallback(RCReceiver* rc, uint16_t previousValue);
00040
00041 void calculateWheelValueCallback(RCReceiver* rc, uint16_t previousValue);
00042
00043 #endif /* RECEIVER_H_ */

```

6.16 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_ADC_MspInit](#) (ADC_HandleTypeDef *hadc)
 - ADC MSP Initialization This function configures the hardware resources used in this example.*
- void [HAL_ADC_MspDeInit](#) (ADC_HandleTypeDef *hadc)
 - ADC MSP De-Initialization This function freeze the hardware resources used in this example.*
- 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_TIM_IC_MspInit](#) (TIM_HandleTypeDef *htim_ic)

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

- void [HAL_TIM_Base_MspInit](#) (TIM_HandleTypeDef *htim_base)

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

- 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_MspDeInit](#) (TIM_HandleTypeDef *htim_ic)

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

- void [HAL_TIM_Base_MspDeInit](#) (TIM_HandleTypeDef *htim_base)

TIM_Base MSP De-Initialization This function freeze 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.

Variables

- DMA_HandleTypeDef [hdma_i2c1_rx](#)
- DMA_HandleTypeDef [hdma_i2c1_tx](#)

6.16.1 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.

6.16.2 Function Documentation

6.16.2.1 HAL_ADC_MspDeInit()

```
void HAL_ADC_MspDeInit (
    ADC_HandleTypeDef * hadc )
```

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

Parameters

<i>hadc</i>	ADC handle pointer
-------------	--------------------

Return values

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

ADC1 GPIO Configuration PA0-WKUP -----> ADC1_IN0

6.16.2.2 HAL_ADC_MspInit()

```
void HAL_ADC_MspInit (
    ADC_HandleTypeDef * hadc )
```

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

Parameters

<i>hadc</i>	ADC handle pointer
-------------	--------------------

Return values

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

ADC1 GPIO Configuration PA0-WKUP -----> ADC1_IN0

6.16.2.3 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

6.16.2.4 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

6.16.2.5 HAL_MspInit()

```
void HAL_MspInit (
    void )
```

Initializes the Global MSP.

6.16.2.6 HAL_TIM_Base_MspDeInit()

```
void HAL_TIM_Base_MspDeInit (
    TIM_HandleTypeDef * htim_base )
```

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

Parameters

<i>htim_base</i>	TIM_Base handle pointer
------------------	-------------------------

Return values

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

6.16.2.7 HAL_TIM_Base_MspInit()

```
void HAL_TIM_Base_MspInit (
    TIM_HandleTypeDef * htim_base )
```

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

Parameters

<i>htim_base</i>	TIM_Base handle pointer
------------------	-------------------------

Return values

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

6.16.2.8 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> ↔ _ic	TIM_IC handle pointer
----------------------	-----------------------

Return values

None	
------	--

TIM1 GPIO Configuration PA8 -----> TIM1_CH1 PA9 -----> TIM1_CH2 PA10 -----> TIM1_CH3 PA11 -----> TIM1_CH4

6.16.2.9 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> ↔ _ic	TIM_IC handle pointer
----------------------	-----------------------

Return values

None	
------	--

TIM1 GPIO Configuration PA8 -----> TIM1_CH1 PA9 -----> TIM1_CH2 PA10 -----> TIM1_CH3 PA11 -----> TIM1_CH4

6.16.2.10 HAL_TIM_MspPostInit()

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

TIM2 GPIO Configuration PA1 -----> TIM2_CH2

TIM3 GPIO Configuration PA6 -----> TIM3_CH1 PA7 -----> TIM3_CH2 PB0 -----> TIM3_CH3 PB1 -----> TIM3↔_CH4

6.16.2.11 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>	
-------------	--

6.16.2.12 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>	
-------------	--

6.16.3 Variable Documentation**6.16.3.1 hdma_i2c1_rx**

```
DMA_HandleTypeDef hdma_i2c1_rx [extern]
```

6.16.3.2 hdma_i2c1_tx

```
DMA_HandleTypeDef hdma_i2c1_tx [extern]
```

6.17 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 [DMA1_Stream0_IRQHandler](#) (void)
This function handles DMA1 stream0 global interrupt.
- void [DMA1_Stream1_IRQHandler](#) (void)
This function handles DMA1 stream1 global interrupt.

Variables

- DMA_HandleTypeDef [hdma_i2c1_rx](#)
- DMA_HandleTypeDef [hdma_i2c1_tx](#)

6.17.1 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.

6.17.2 Function Documentation

6.17.2.1 BusFault_Handler()

```
void BusFault_Handler (  
    void )
```

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

6.17.2.2 DebugMon_Handler()

```
void DebugMon_Handler (  
    void )
```

This function handles Debug monitor.

6.17.2.3 DMA1_Stream0_IRQHandler()

```
void DMA1_Stream0_IRQHandler (  
    void )
```

This function handles DMA1 stream0 global interrupt.

6.17.2.4 DMA1_Stream1_IRQHandler()

```
void DMA1_Stream1_IRQHandler (  
    void )
```

This function handles DMA1 stream1 global interrupt.

6.17.2.5 HardFault_Handler()

```
void HardFault_Handler (  
    void )
```

This function handles Hard fault interrupt.

6.17.2.6 MemManage_Handler()

```
void MemManage_Handler (  
    void )
```

This function handles Memory management fault.

6.17.2.7 NMI_Handler()

```
void NMI_Handler (  
    void )
```

This function handles Non maskable interrupt.

6.17.2.8 PendSV_Handler()

```
void PendSV_Handler (  
    void )
```

This function handles Pendable request for system service.

6.17.2.9 SVC_Handler()

```
void SVC_Handler (
    void )
```

This function handles System service call via SWI instruction.

6.17.2.10 SysTick_Handler()

```
void SysTick_Handler (
    void )
```

This function handles System tick timer.

6.17.2.11 UsageFault_Handler()

```
void UsageFault_Handler (
    void )
```

This function handles Undefined instruction or illegal state.

6.17.3 Variable Documentation

6.17.3.1 hdma_i2c1_rx

```
DMA_HandleTypeDef hdma_i2c1_rx [extern]
```

6.17.3.2 hdma_i2c1_tx

```
DMA_HandleTypeDef hdma_i2c1_tx [extern]
```

6.18 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`

6.18.1 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.

6.18.2 Function Documentation

6.18.2.1 `__attribute__()`

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


6.18.2.2 `__io_getchar()`

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

6.18.2.3 `__io_putchar()`

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

6.18.2.4 `_close()`

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

6.18.2.5 `_execve()`

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

6.18.2.6 `_exit()`

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

6.18.2.7 `_fork()`

```
int _fork (
    void )
```

6.18.2.8 `_fstat()`

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

6.18.2.9 `_getpid()`

```
int _getpid (
    void )
```

6.18.2.10 _isatty()

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

6.18.2.11 _kill()

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

6.18.2.12 _link()

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

6.18.2.13 _lseek()

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

6.18.2.14 _open()

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

6.18.2.15 _stat()

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

6.18.2.16 _times()

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

6.18.2.17 _unlink()

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

6.18.2.18 `_wait()`

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

6.18.2.19 `initialise_monitor_handles()`

```
void initialise_monitor_handles ( )
```

6.18.3 Variable Documentation

6.18.3.1 `environ`

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

6.19 sysmem.c File Reference

STM32CubeIDE System Memory calls file.

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

Functions

- void * [_sbrk](#) (ptrdiff_t incr)
[_sbrk\(\)](#) allocates memory to the newlib heap and is used by malloc and others from the C library

6.19.1 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.

6.19.2 Function Documentation

6.19.2.1 `_sbrk()`

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

`_sbrk()` 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

<code>incr</code>	Memory size
-------------------	-------------

Returns

Pointer to allocated memory

6.20 `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}

6.20.1 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\(\)](#): 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\(\)](#): 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.

Index

- `__attribute__`
 - `syscalls.c`, [82](#)
 - `__io_getchar`
 - `syscalls.c`, [82](#)
 - `__io_putchar`
 - `syscalls.c`, [83](#)
 - `_bno055_i2c_port`
 - `bno055_stm32.h`, [52](#)
 - `_close`
 - `syscalls.c`, [83](#)
 - `_execve`
 - `syscalls.c`, [83](#)
 - `_exit`
 - `syscalls.c`, [83](#)
 - `_fork`
 - `syscalls.c`, [83](#)
 - `_fstat`
 - `syscalls.c`, [83](#)
 - `_getpid`
 - `syscalls.c`, [83](#)
 - `_isatty`
 - `syscalls.c`, [83](#)
 - `_kill`
 - `syscalls.c`, [84](#)
 - `_link`
 - `syscalls.c`, [84](#)
 - `_lseek`
 - `syscalls.c`, [84](#)
 - `_open`
 - `syscalls.c`, [84](#)
 - `_sbrk`
 - `sysmem.c`, [86](#)
 - `_stat`
 - `syscalls.c`, [84](#)
 - `_times`
 - `syscalls.c`, [84](#)
 - `_unlink`
 - `syscalls.c`, [84](#)
 - `_wait`
 - `syscalls.c`, [84](#)
- `accel`
 - `bno055_calibration_offset_t`, [13](#)
 - `bno055_calibration_radius_t`, [13](#)
 - `bno055_calibration_state_t`, [14](#)
- `accelScale`
 - `bno055.c`, [25](#)
- `accState`
 - `bno055_self_test_result_t`, [15](#)
- `adc_value`
 - `main.c`, [63](#)
- `AHBPrescTable`
 - `STM32F4xx_System_Private_Variables`, [8](#)
- `angularRateScale`
 - `bno055.c`, [25](#)
- `APBPrescTable`
 - `STM32F4xx_System_Private_Variables`, [8](#)
- `apply_low_pass_filter`
 - `fsr_controller.c`, [54](#)
 - `fsr_controller.h`, [56](#)
- `backward_channel`
 - `MotorDriver`, [17](#)
- `bno055.c`, [21](#)
 - `accelScale`, [25](#)
 - `angularRateScale`, [25](#)
 - `bno055_disableExternalCrystal`, [22](#)
 - `bno055_enableExternalCrystal`, [22](#)
 - `bno055_getBootloaderRevision`, [22](#)
 - `bno055_getCalibrationData`, [22](#)
 - `bno055_getCalibrationState`, [22](#)
 - `bno055_getOperationMode`, [22](#)
 - `bno055_getSelfTestResult`, [22](#)
 - `bno055_getSWRevision`, [22](#)
 - `bno055_getSystemError`, [22](#)
 - `bno055_getSystemStatus`, [23](#)
 - `bno055_getTemp`, [23](#)
 - `bno055_getVector`, [23](#)
 - `bno055_getVectorAccelerometer`, [23](#)
 - `bno055_getVectorEuler`, [23](#)
 - `bno055_getVectorGravity`, [23](#)
 - `bno055_getVectorGyroscope`, [23](#)
 - `bno055_getVectorLinearAccel`, [23](#)
 - `bno055_getVectorMagnetometer`, [23](#)
 - `bno055_getVectorQuaternion`, [23](#)
 - `bno055_reset`, [24](#)
 - `bno055_setAxisMap`, [24](#)
 - `bno055_setCalibrationData`, [24](#)
 - `bno055_setExternalCrystalUse`, [24](#)
 - `bno055_setOperationMode`, [24](#)
 - `bno055_setOperationModeConfig`, [24](#)
 - `bno055_setOperationModeNDOF`, [24](#)
 - `bno055_setPage`, [24](#)
 - `bno055_setup`, [24](#)
 - `eulerScale`, [25](#)
 - `magScale`, [25](#)
 - `quaScale`, [25](#)
 - `tempScale`, [25](#)
- `bno055.h`, [25](#)
 - `BNO055_ACC_AM_THRES`, [29](#)

BNO055_ACC_CONFIG, [29](#)
 BNO055_ACC_DATA_X_LSB, [30](#)
 BNO055_ACC_DATA_X_MSB, [30](#)
 BNO055_ACC_DATA_Y_LSB, [30](#)
 BNO055_ACC_DATA_Y_MSB, [30](#)
 BNO055_ACC_DATA_Z_LSB, [30](#)
 BNO055_ACC_DATA_Z_MSB, [30](#)
 BNO055_ACC_HG_DURATION, [30](#)
 BNO055_ACC_HG_THRESH, [30](#)
 BNO055_ACC_ID, [30](#)
 BNO055_ACC_INT_SETTINGS, [30](#)
 BNO055_ACC_NM_SET, [31](#)
 BNO055_ACC_NM_THRESH, [31](#)
 BNO055_ACC_OFFSET_X_LSB, [31](#)
 BNO055_ACC_OFFSET_X_MSB, [31](#)
 BNO055_ACC_OFFSET_Y_LSB, [31](#)
 BNO055_ACC_OFFSET_Y_MSB, [31](#)
 BNO055_ACC_OFFSET_Z_LSB, [31](#)
 BNO055_ACC_OFFSET_Z_MSB, [31](#)
 BNO055_ACC_RADIUS_LSB, [31](#)
 BNO055_ACC_RADIUS_MSB, [31](#)
 BNO055_ACC_SLEEP_CONFIG, [32](#)
 BNO055_AXIS_MAP_CONFIG, [32](#)
 bno055_axis_map_representation_t, [42](#)
 BNO055_AXIS_MAP_SIGN, [32](#)
 bno055_axis_map_sign_t, [43](#)
 BNO055_AXIS_SIGN_NEGATIVE, [43](#)
 BNO055_AXIS_SIGN_POSITIVE, [43](#)
 BNO055_AXIS_X, [43](#)
 BNO055_AXIS_Y, [43](#)
 BNO055_AXIS_Z, [43](#)
 BNO055_BL_REV_ID, [32](#)
 BNO055_CALIB_STAT, [32](#)
 BNO055_CHIP_ID, [32](#)
 bno055_delay, [44](#)
 bno055_disableExternalCrystal, [44](#)
 bno055_enableExternalCrystal, [45](#)
 BNO055_EUL_HEADING_LSB, [32](#)
 BNO055_EUL_HEADING_MSB, [32](#)
 BNO055_EUL_PITCH_LSB, [32](#)
 BNO055_EUL_PITCH_MSB, [32](#)
 BNO055_EUL_ROLL_LSB, [33](#)
 BNO055_EUL_ROLL_MSB, [33](#)
 bno055_getBootloaderRevision, [45](#)
 bno055_getCalibrationData, [45](#)
 bno055_getCalibrationState, [45](#)
 bno055_getOperationMode, [45](#)
 bno055_getSelfTestResult, [45](#)
 bno055_getSWRevision, [45](#)
 bno055_getSystemError, [45](#)
 bno055_getSystemStatus, [45](#)
 bno055_getTemp, [45](#)
 bno055_getVectorAccelerometer, [46](#)
 bno055_getVectorEuler, [46](#)
 bno055_getVectorGravity, [46](#)
 bno055_getVectorGyroscope, [46](#)
 bno055_getVectorLinearAccel, [46](#)
 bno055_getVectorMagnetometer, [46](#)
 bno055_getVectorQuaternion, [46](#)
 BNO055_GRV_DATA_X_LSB, [33](#)
 BNO055_GRV_DATA_X_MSB, [33](#)
 BNO055_GRV_DATA_Y_LSB, [33](#)
 BNO055_GRV_DATA_Y_MSB, [33](#)
 BNO055_GRV_DATA_Z_LSB, [33](#)
 BNO055_GRV_DATA_Z_MSB, [33](#)
 BNO055_GYR_AM_SET, [33](#)
 BNO055_GYR_AM_THRESH, [33](#)
 BNO055_GYR_DATA_X_LSB, [34](#)
 BNO055_GYR_DATA_X_MSB, [34](#)
 BNO055_GYR_DATA_Y_LSB, [34](#)
 BNO055_GYR_DATA_Y_MSB, [34](#)
 BNO055_GYR_DATA_Z_LSB, [34](#)
 BNO055_GYR_DATA_Z_MSB, [34](#)
 BNO055_GYR_DUR_X, [34](#)
 BNO055_GYR_DUR_Y, [34](#)
 BNO055_GYR_DUR_Z, [34](#)
 BNO055_GYR_HR_X_SET, [34](#)
 BNO055_GYR_HR_Y_SET, [35](#)
 BNO055_GYR_HR_Z_SET, [35](#)
 BNO055_GYR_INT_SETTINGS, [35](#)
 BNO055_GYR_OFFSET_X_LSB, [35](#)
 BNO055_GYR_OFFSET_X_MSB, [35](#)
 BNO055_GYR_OFFSET_Y_LSB, [35](#)
 BNO055_GYR_OFFSET_Y_MSB, [35](#)
 BNO055_GYR_OFFSET_Z_LSB, [35](#)
 BNO055_GYR_OFFSET_Z_MSB, [35](#)
 BNO055_GYR_SLEEP_CONFIG, [35](#)
 BNO055_GYRO_CONFIG_0, [36](#)
 BNO055_GYRO_CONFIG_1, [36](#)
 BNO055_GYRO_ID, [36](#)
 BNO055_I2C_ADDR, [36](#)
 BNO055_I2C_ADDR_HI, [36](#)
 BNO055_I2C_ADDR_LO, [36](#)
 BNO055_ID, [36](#)
 BNO055_INT_EN, [36](#)
 BNO055_INT_MSK, [36](#)
 BNO055_INT_STATUS, [36](#)
 BNO055_LIA_DATA_X_LSB, [37](#)
 BNO055_LIA_DATA_X_MSB, [37](#)
 BNO055_LIA_DATA_Y_LSB, [37](#)
 BNO055_LIA_DATA_Y_MSB, [37](#)
 BNO055_LIA_DATA_Z_LSB, [37](#)
 BNO055_LIA_DATA_Z_MSB, [37](#)
 BNO055_MAG_CONFIG, [37](#)
 BNO055_MAG_DATA_X_LSB, [37](#)
 BNO055_MAG_DATA_X_MSB, [37](#)
 BNO055_MAG_DATA_Y_LSB, [37](#)
 BNO055_MAG_DATA_Y_MSB, [38](#)
 BNO055_MAG_DATA_Z_LSB, [38](#)
 BNO055_MAG_DATA_Z_MSB, [38](#)
 BNO055_MAG_ID, [38](#)
 BNO055_MAG_OFFSET_X_LSB, [38](#)
 BNO055_MAG_OFFSET_X_MSB, [38](#)
 BNO055_MAG_OFFSET_Y_LSB, [38](#)
 BNO055_MAG_OFFSET_Y_MSB, [38](#)
 BNO055_MAG_OFFSET_Z_LSB, [38](#)

- BNO055_MAG_OFFSET_Z_MSB, [38](#)
- BNO055_MAG_RADIUS_LSB, [39](#)
- BNO055_MAG_RADIUS_MSB, [39](#)
- BNO055_OPERATION_MODE_ACCGYRO, [43](#)
- BNO055_OPERATION_MODE_ACCMAG, [43](#)
- BNO055_OPERATION_MODE_ACCONLY, [43](#)
- BNO055_OPERATION_MODE_AMG, [43](#)
- BNO055_OPERATION_MODE_COMPASS, [43](#)
- BNO055_OPERATION_MODE_CONFIG, [43](#)
- BNO055_OPERATION_MODE_GYRONLY, [43](#)
- BNO055_OPERATION_MODE_IMU, [43](#)
- BNO055_OPERATION_MODE_M4G, [43](#)
- BNO055_OPERATION_MODE_MAGGYRO, [43](#)
- BNO055_OPERATION_MODE_MAGONLY, [43](#)
- BNO055_OPERATION_MODE_NDOF, [43](#)
- BNO055_OPERATION_MODE_NDOF_FMC_OFF, [43](#)
- bn055_opmode_t, [43](#)
- BNO055_OPR_MODE, [39](#)
- BNO055_PAGE_ID, [39](#)
- BNO055_PWR_MODE, [39](#)
- BNO055_QUA_DATA_W_LSB, [39](#)
- BNO055_QUA_DATA_W_MSB, [39](#)
- BNO055_QUA_DATA_X_LSB, [39](#)
- BNO055_QUA_DATA_X_MSB, [39](#)
- BNO055_QUA_DATA_Y_LSB, [40](#)
- BNO055_QUA_DATA_Y_MSB, [40](#)
- BNO055_QUA_DATA_Z_LSB, [40](#)
- BNO055_QUA_DATA_Z_MSB, [40](#)
- BNO055_READ_TIMEOUT, [40](#)
- bn055_readData, [46](#)
- bn055_reset, [46](#)
- bn055_setAxisMap, [46](#)
- bn055_setCalibrationData, [47](#)
- bn055_setOperationMode, [47](#)
- bn055_setOperationModeConfig, [47](#)
- bn055_setOperationModeNDOF, [47](#)
- bn055_setup, [47](#)
- BNO055_ST_RESULT, [40](#)
- BNO055_SW_REV_ID_LSB, [40](#)
- BNO055_SW_REV_ID_MSB, [40](#)
- BNO055_SYS_CLK_STATUS, [40](#)
- BNO055_SYS_ERR, [40](#)
- BNO055_SYS_STATUS, [41](#)
- BNO055_SYS_TRIGGER, [41](#)
- BNO055_SYSTEM_ERROR_ACCEL_PWR_MODE_NOT_AVAILABLE, [44](#)
- BNO055_SYSTEM_ERROR_FUSION_ALGO_CONF_ERROR, [44](#)
- BNO055_SYSTEM_ERROR_LOW_PWR_MODE_NOT_AVAILABLE, [44](#)
- BNO055_SYSTEM_ERROR_NO_ERROR, [43](#)
- BNO055_SYSTEM_ERROR_PERIPHERAL_INITIALIZATION_ERROR, [43](#)
- BNO055_SYSTEM_ERROR_REG_MAP_ADDR_OUT_OF_RANGE, [44](#)
- BNO055_SYSTEM_ERROR_REG_MAP_VAL_OUT_OF_RANGE, [44](#)
- BNO055_SYSTEM_ERROR_REG_MAP_WRITE_ERROR, [44](#)
- BNO055_SYSTEM_ERROR_SELF_TEST_FAILED, [44](#)
- BNO055_SYSTEM_ERROR_SENSOR_CONF_ERROR, [44](#)
- BNO055_SYSTEM_ERROR_SYSTEM_INITIALIZATION_ERROR, [43](#)
- bn055_system_error_t, [43](#)
- BNO055_SYSTEM_STATUS_EXECUTING_SELF_TEST, [44](#)
- BNO055_SYSTEM_STATUS_FUSION_ALGO_RUNNING, [44](#)
- BNO055_SYSTEM_STATUS_FUSION ALOG NOT RUNNING, [44](#)
- BNO055_SYSTEM_STATUS_IDLE, [44](#)
- BNO055_SYSTEM_STATUS_INITIALIZING_PERIPHERALS, [44](#)
- BNO055_SYSTEM_STATUS_SYSTEM_ERROR, [44](#)
- BNO055_SYSTEM_STATUS_SYSTEM_INITIALIZATION, [44](#)
- bn055_system_status_t, [44](#)
- BNO055_TEMP, [41](#)
- BNO055_TEMP_SOURCE, [41](#)
- BNO055_UNIT_SEL, [41](#)
- BNO055_VECTOR_ACCELEROMETER, [44](#)
- BNO055_VECTOR_EULER, [44](#)
- BNO055_VECTOR_GRAVITY, [44](#)
- BNO055_VECTOR_GYROSCOPE, [44](#)
- BNO055_VECTOR_LINEARACCEL, [44](#)
- BNO055_VECTOR_MAGNETOMETER, [44](#)
- BNO055_VECTOR_QUATERNION, [44](#)
- bn055_vector_type_t, [44](#)
- BNO055_WRITE_TIMEOUT, [41](#)
- bn055_writeData, [47](#)
- ERROR_BUS_OVERRUN_ERR, [41](#)
- ERROR_BYTE, [41](#)
- ERROR_MAX_LEN_ERR, [41](#)
- ERROR_MIN_LEN_ERR, [41](#)
- ERROR_RECV_CHAR_TIMEOUT, [42](#)
- ERROR_REGMAP_INV_ADDR, [42](#)
- ERROR_REGMAP_WRITE_DIS, [42](#)
- ERROR_WRITE_FAIL, [42](#)
- ERROR_WRITE_SUCCESS, [42](#)
- ERROR_WRONG_START_BYTE, [42](#)
- REG_READ, [42](#)
- REG_WRITE, [42](#)
- RESPONSE_BYTE, [42](#)
- STATUS_BYTE_SELECTED_OPR_MODE, [42](#)
- BNO055_ACC_AM_THRES, [29](#)
- bn055.h, [29](#)
- BNO055_ACC_CONFIG, [29](#)
- bn055.h, [29](#)
- BNO055_ACC_DATA_X_LSB, [30](#)
- bn055.h, [30](#)
- BNO055_ACC_DATA_X_MSB, [30](#)
- bn055.h, [30](#)

BNO055_ACC_DATA_Y_LSB
 bno055.h, [30](#)
 BNO055_ACC_DATA_Y_MSB
 bno055.h, [30](#)
 BNO055_ACC_DATA_Z_LSB
 bno055.h, [30](#)
 BNO055_ACC_DATA_Z_MSB
 bno055.h, [30](#)
 BNO055_ACC_HG_DURATION
 bno055.h, [30](#)
 BNO055_ACC_HG_THRESH
 bno055.h, [30](#)
 BNO055_ACC_ID
 bno055.h, [30](#)
 BNO055_ACC_INT_SETTINGS
 bno055.h, [30](#)
 BNO055_ACC_NM_SET
 bno055.h, [31](#)
 BNO055_ACC_NM_THRESH
 bno055.h, [31](#)
 BNO055_ACC_OFFSET_X_LSB
 bno055.h, [31](#)
 BNO055_ACC_OFFSET_X_MSB
 bno055.h, [31](#)
 BNO055_ACC_OFFSET_Y_LSB
 bno055.h, [31](#)
 BNO055_ACC_OFFSET_Y_MSB
 bno055.h, [31](#)
 BNO055_ACC_OFFSET_Z_LSB
 bno055.h, [31](#)
 BNO055_ACC_OFFSET_Z_MSB
 bno055.h, [31](#)
 BNO055_ACC_RADIUS_LSB
 bno055.h, [31](#)
 BNO055_ACC_RADIUS_MSB
 bno055.h, [31](#)
 BNO055_ACC_SLEEP_CONFIG
 bno055.h, [32](#)
 bno055_assignI2C
 bno055_stm32.h, [52](#)
 BNO055_AXIS_MAP_CONFIG
 bno055.h, [32](#)
 bno055_axis_map_representation_t
 bno055.h, [42](#)
 BNO055_AXIS_MAP_SIGN
 bno055.h, [32](#)
 bno055_axis_map_sign_t
 bno055.h, [43](#)
 bno055_axis_map_t, [11](#)
 x, [11](#)
 x_sign, [11](#)
 y, [11](#)
 y_sign, [11](#)
 z, [11](#)
 z_sign, [12](#)
 BNO055_AXIS_SIGN_NEGATIVE
 bno055.h, [43](#)
 BNO055_AXIS_SIGN_POSITIVE
 bno055.h, [43](#)
 BNO055_AXIS_X
 bno055.h, [43](#)
 BNO055_AXIS_Y
 bno055.h, [43](#)
 BNO055_AXIS_Z
 bno055.h, [43](#)
 BNO055_BL_REV_ID
 bno055.h, [32](#)
 BNO055_CALIB_STAT
 bno055.h, [32](#)
 bno055_calibration_data_t, [12](#)
 offset, [12](#)
 radius, [12](#)
 bno055_calibration_offset_t, [12](#)
 accel, [13](#)
 gyro, [13](#)
 mag, [13](#)
 bno055_calibration_radius_t, [13](#)
 accel, [13](#)
 mag, [13](#)
 bno055_calibration_state_t, [14](#)
 accel, [14](#)
 gyro, [14](#)
 mag, [14](#)
 sys, [14](#)
 BNO055_CHIP_ID
 bno055.h, [32](#)
 bno055_delay
 bno055.h, [44](#)
 bno055_stm32.h, [52](#)
 bno055_disableExternalCrystal
 bno055.c, [22](#)
 bno055.h, [44](#)
 bno055_enableExternalCrystal
 bno055.c, [22](#)
 bno055.h, [45](#)
 BNO055_EUL_HEADING_LSB
 bno055.h, [32](#)
 BNO055_EUL_HEADING_MSB
 bno055.h, [32](#)
 BNO055_EUL_PITCH_LSB
 bno055.h, [32](#)
 BNO055_EUL_PITCH_MSB
 bno055.h, [32](#)
 BNO055_EUL_ROLL_LSB
 bno055.h, [33](#)
 BNO055_EUL_ROLL_MSB
 bno055.h, [33](#)
 bno055_getBootloaderRevision
 bno055.c, [22](#)
 bno055.h, [45](#)
 bno055_getCalibrationData
 bno055.c, [22](#)
 bno055.h, [45](#)
 bno055_getCalibrationState
 bno055.c, [22](#)
 bno055.h, [45](#)

bn055_getOperationMode
 bn055.c, [22](#)
 bn055.h, [45](#)

bn055_getSelfTestResult
 bn055.c, [22](#)
 bn055.h, [45](#)

bn055_getSWRevision
 bn055.c, [22](#)
 bn055.h, [45](#)

bn055_getSystemError
 bn055.c, [22](#)
 bn055.h, [45](#)

bn055_getSystemStatus
 bn055.c, [23](#)
 bn055.h, [45](#)

bn055_getTemp
 bn055.c, [23](#)
 bn055.h, [45](#)

bn055_getVector
 bn055.c, [23](#)

bn055_getVectorAccelerometer
 bn055.c, [23](#)
 bn055.h, [46](#)

bn055_getVectorEuler
 bn055.c, [23](#)
 bn055.h, [46](#)

bn055_getVectorGravity
 bn055.c, [23](#)
 bn055.h, [46](#)

bn055_getVectorGyroscope
 bn055.c, [23](#)
 bn055.h, [46](#)

bn055_getVectorLinearAccel
 bn055.c, [23](#)
 bn055.h, [46](#)

bn055_getVectorMagnetometer
 bn055.c, [23](#)
 bn055.h, [46](#)

bn055_getVectorQuaternion
 bn055.c, [23](#)
 bn055.h, [46](#)

BNO055_GRV_DATA_X_LSB
 bn055.h, [33](#)

BNO055_GRV_DATA_X_MSB
 bn055.h, [33](#)

BNO055_GRV_DATA_Y_LSB
 bn055.h, [33](#)

BNO055_GRV_DATA_Y_MSB
 bn055.h, [33](#)

BNO055_GRV_DATA_Z_LSB
 bn055.h, [33](#)

BNO055_GRV_DATA_Z_MSB
 bn055.h, [33](#)

BNO055_GYR_AM_SET
 bn055.h, [33](#)

BNO055_GYR_AM_THRESH
 bn055.h, [33](#)

BNO055_GYR_DATA_X_LSB
 bn055.h, [34](#)

BNO055_GYR_DATA_X_MSB
 bn055.h, [34](#)

BNO055_GYR_DATA_Y_LSB
 bn055.h, [34](#)

BNO055_GYR_DATA_Y_MSB
 bn055.h, [34](#)

BNO055_GYR_DATA_Z_LSB
 bn055.h, [34](#)

BNO055_GYR_DATA_Z_MSB
 bn055.h, [34](#)

BNO055_GYR_DUR_X
 bn055.h, [34](#)

BNO055_GYR_DUR_Y
 bn055.h, [34](#)

BNO055_GYR_DUR_Z
 bn055.h, [34](#)

BNO055_GYR_HR_X_SET
 bn055.h, [34](#)

BNO055_GYR_HR_Y_SET
 bn055.h, [35](#)

BNO055_GYR_HR_Z_SET
 bn055.h, [35](#)

BNO055_GYR_INT_SETTINGS
 bn055.h, [35](#)

BNO055_GYR_OFFSET_X_LSB
 bn055.h, [35](#)

BNO055_GYR_OFFSET_X_MSB
 bn055.h, [35](#)

BNO055_GYR_OFFSET_Y_LSB
 bn055.h, [35](#)

BNO055_GYR_OFFSET_Y_MSB
 bn055.h, [35](#)

BNO055_GYR_OFFSET_Z_LSB
 bn055.h, [35](#)

BNO055_GYR_OFFSET_Z_MSB
 bn055.h, [35](#)

BNO055_GYR_SLEEP_CONFIG
 bn055.h, [35](#)

BNO055_GYRO_CONFIG_0
 bn055.h, [36](#)

BNO055_GYRO_CONFIG_1
 bn055.h, [36](#)

BNO055_GYRO_ID
 bn055.h, [36](#)

BNO055_I2C_ADDR
 bn055.h, [36](#)

BNO055_I2C_ADDR_HI
 bn055.h, [36](#)

BNO055_I2C_ADDR_LO
 bn055.h, [36](#)

BNO055_ID
 bn055.h, [36](#)

BNO055_INT_EN
 bn055.h, [36](#)

BNO055_INT_MSK
 bn055.h, [36](#)

BNO055_INT_STATUS

bno055.h, [36](#)
 BNO055_LIA_DATA_X_LSB
 bno055.h, [37](#)
 BNO055_LIA_DATA_X_MSB
 bno055.h, [37](#)
 BNO055_LIA_DATA_Y_LSB
 bno055.h, [37](#)
 BNO055_LIA_DATA_Y_MSB
 bno055.h, [37](#)
 BNO055_LIA_DATA_Z_LSB
 bno055.h, [37](#)
 BNO055_LIA_DATA_Z_MSB
 bno055.h, [37](#)
 BNO055_MAG_CONFIG
 bno055.h, [37](#)
 BNO055_MAG_DATA_X_LSB
 bno055.h, [37](#)
 BNO055_MAG_DATA_X_MSB
 bno055.h, [37](#)
 BNO055_MAG_DATA_Y_LSB
 bno055.h, [37](#)
 BNO055_MAG_DATA_Y_MSB
 bno055.h, [38](#)
 BNO055_MAG_DATA_Z_LSB
 bno055.h, [38](#)
 BNO055_MAG_DATA_Z_MSB
 bno055.h, [38](#)
 BNO055_MAG_ID
 bno055.h, [38](#)
 BNO055_MAG_OFFSET_X_LSB
 bno055.h, [38](#)
 BNO055_MAG_OFFSET_X_MSB
 bno055.h, [38](#)
 BNO055_MAG_OFFSET_Y_LSB
 bno055.h, [38](#)
 BNO055_MAG_OFFSET_Y_MSB
 bno055.h, [38](#)
 BNO055_MAG_OFFSET_Z_LSB
 bno055.h, [38](#)
 BNO055_MAG_OFFSET_Z_MSB
 bno055.h, [38](#)
 BNO055_MAG_RADIUS_LSB
 bno055.h, [39](#)
 BNO055_MAG_RADIUS_MSB
 bno055.h, [39](#)
 BNO055_OPERATION_MODE_ACCGYRO
 bno055.h, [43](#)
 BNO055_OPERATION_MODE_ACCMAG
 bno055.h, [43](#)
 BNO055_OPERATION_MODE_ACCONLY
 bno055.h, [43](#)
 BNO055_OPERATION_MODE_AMG
 bno055.h, [43](#)
 BNO055_OPERATION_MODE_COMPASS
 bno055.h, [43](#)
 BNO055_OPERATION_MODE_CONFIG
 bno055.h, [43](#)
 BNO055_OPERATION_MODE_GYRONLY
 bno055.h, [43](#)
 BNO055_OPERATION_MODE_IMU
 bno055.h, [43](#)
 BNO055_OPERATION_MODE_M4G
 bno055.h, [43](#)
 BNO055_OPERATION_MODE_MAGGYRO
 bno055.h, [43](#)
 BNO055_OPERATION_MODE_MAGONLY
 bno055.h, [43](#)
 BNO055_OPERATION_MODE_NDOF
 bno055.h, [43](#)
 BNO055_OPERATION_MODE_NDOF_FMC_OFF
 bno055.h, [43](#)
 bno055_opmode_t
 bno055.h, [43](#)
 BNO055_OPR_MODE
 bno055.h, [39](#)
 BNO055_PAGE_ID
 bno055.h, [39](#)
 BNO055_PWR_MODE
 bno055.h, [39](#)
 BNO055_QUA_DATA_W_LSB
 bno055.h, [39](#)
 BNO055_QUA_DATA_W_MSB
 bno055.h, [39](#)
 BNO055_QUA_DATA_X_LSB
 bno055.h, [39](#)
 BNO055_QUA_DATA_X_MSB
 bno055.h, [39](#)
 BNO055_QUA_DATA_Y_LSB
 bno055.h, [40](#)
 BNO055_QUA_DATA_Y_MSB
 bno055.h, [40](#)
 BNO055_QUA_DATA_Z_LSB
 bno055.h, [40](#)
 BNO055_QUA_DATA_Z_MSB
 bno055.h, [40](#)
 BNO055_READ_TIMEOUT
 bno055.h, [40](#)
 bno055_readData
 bno055.h, [46](#)
 bno055_stm32.h, [52](#)
 bno055_reset
 bno055.c, [24](#)
 bno055.h, [46](#)
 bno055_self_test_result_t, [14](#)
 accState, [15](#)
 gyrState, [15](#)
 magState, [15](#)
 mcuState, [15](#)
 bno055_setAxisMap
 bno055.c, [24](#)
 bno055.h, [46](#)
 bno055_setCalibrationData
 bno055.c, [24](#)
 bno055.h, [47](#)
 bno055_setExternalCrystalUse
 bno055.c, [24](#)

bno055_setOperationMode
 bno055.c, [24](#)
 bno055.h, [47](#)
 bno055_setOperationModeConfig
 bno055.c, [24](#)
 bno055.h, [47](#)
 bno055_setOperationModeNDOF
 bno055.c, [24](#)
 bno055.h, [47](#)
 bno055_setPage
 bno055.c, [24](#)
 bno055_setup
 bno055.c, [24](#)
 bno055.h, [47](#)
 BNO055_ST_RESULT
 bno055.h, [40](#)
 bno055_stm32.h, [51](#)
 _bno055_i2c_port, [52](#)
 bno055_assignI2C, [52](#)
 bno055_delay, [52](#)
 bno055_readData, [52](#)
 bno055_writeData, [52](#)
 BNO055_SW_REV_ID_LSB
 bno055.h, [40](#)
 BNO055_SW_REV_ID_MSB
 bno055.h, [40](#)
 BNO055_SYS_CLK_STATUS
 bno055.h, [40](#)
 BNO055_SYS_ERR
 bno055.h, [40](#)
 BNO055_SYS_STATUS
 bno055.h, [41](#)
 BNO055_SYS_TRIGGER
 bno055.h, [41](#)
 BNO055_SYSTEM_ERROR_ACCEL_PWR_MODE_NOT_AVAILABLE
 bno055.h, [44](#)
 BNO055_SYSTEM_ERROR_FUSION_ALGO_CONF_ERROR
 bno055.h, [44](#)
 BNO055_SYSTEM_ERROR_LOW_PWR_MODE_NOT_AVAILABLE
 bno055.h, [44](#)
 BNO055_SYSTEM_ERROR_NO_ERROR
 bno055.h, [43](#)
 BNO055_SYSTEM_ERROR_PERIPHERAL_INITIALIZATION_ERROR
 bno055.h, [43](#)
 BNO055_SYSTEM_ERROR_REG_MAP_ADDR_OUT_OF_RANGE
 bno055.h, [44](#)
 BNO055_SYSTEM_ERROR_REG_MAP_VAL_OUT_OF_RANGE
 bno055.h, [44](#)
 BNO055_SYSTEM_ERROR_REG_MAP_WRITE_ERROR
 bno055.h, [44](#)
 BNO055_SYSTEM_ERROR_SELF_TEST_FAILED
 bno055.h, [44](#)
 BNO055_SYSTEM_ERROR_SENSOR_CONF_ERROR
 bno055.h, [44](#)
 BNO055_SYSTEM_ERROR_SYSTEM_INITIALIZATION_ERROR
 bno055.h, [43](#)
 bno055_system_error_t
 bno055.h, [43](#)
 BNO055_SYSTEM_STATUS_EXECUTING_SELF_TEST
 bno055.h, [44](#)
 BNO055_SYSTEM_STATUS_FUSION_ALGO_RUNNING
 bno055.h, [44](#)
 BNO055_SYSTEM_STATUS_FUSION ALOG_NOT_RUNNING
 bno055.h, [44](#)
 BNO055_SYSTEM_STATUS_IDLE
 bno055.h, [44](#)
 BNO055_SYSTEM_STATUS_INITIALIZING_PERIPHERALS
 bno055.h, [44](#)
 BNO055_SYSTEM_STATUS_SYSTEM_ERROR
 bno055.h, [44](#)
 BNO055_SYSTEM_STATUS_SYSTEM_INITIALIZATION
 bno055.h, [44](#)
 bno055_system_status_t
 bno055.h, [44](#)
 BNO055_TEMP
 bno055.h, [41](#)
 BNO055_TEMP_SOURCE
 bno055.h, [41](#)
 BNO055_UNIT_SEL
 bno055.h, [41](#)
 BNO055_VECTOR_ACCELEROMETER
 bno055.h, [44](#)
 BNO055_VECTOR_EULER
 bno055.h, [44](#)
 BNO055_VECTOR_GRAVITY
 bno055.h, [44](#)
 BNO055_VECTOR_GYROSCOPE
 bno055.h, [44](#)
 BNO055_VECTOR_LINEARACCEL
 bno055.h, [44](#)
 BNO055_VECTOR_MAGNETOMETER
 bno055.h, [44](#)
 BNO055_VECTOR_QUATERNION
 bno055.h, [44](#)
 bno055_vector_t, [15](#)
 w, [15](#)
 x, [15](#)
 y, [15](#)
 z, [16](#)
 bno055_vector_type_t
 bno055.h, [44](#)
 bno055_vector_xyz_int16_t, [16](#)
 y, [16](#)
 z, [16](#)
 BNO055_WRITE_TIMEOUT
 bno055.h, [41](#)
 bno055_writeData
 bno055.h, [47](#)
 bno055_stm32.h, [52](#)
 BusFault_Handler
 stm32f4xx_it.c, [79](#)
 calculateTriggerValueCallback
 Receiver.c, [69](#)
 Receiver.h, [71](#)
 calculateWheelValueCallback

- Receiver.c, 70
- Receiver.h, 72
- CMSIS, 7
- DebugMon_Handler
 - stm32f4xx_it.c, 79
- DMA1_Stream0_IRQHandler
 - stm32f4xx_it.c, 80
- DMA1_Stream1_IRQHandler
 - stm32f4xx_it.c, 80
- environ
 - syscalls.c, 85
- ERROR_BUS_OVERRUN_ERR
 - bno055.h, 41
- ERROR_BYTE
 - bno055.h, 41
- ERROR_MAX_LEN_ERR
 - bno055.h, 41
- ERROR_MIN_LEN_ERR
 - bno055.h, 41
- ERROR_RECV_CHAR_TIMEOUT
 - bno055.h, 42
- ERROR_REGMAP_INV_ADDR
 - bno055.h, 42
- ERROR_REGMAP_WRITE_DIS
 - bno055.h, 42
- ERROR_WRITE_FAIL
 - bno055.h, 42
- ERROR_WRITE_SUCCESS
 - bno055.h, 42
- ERROR_WRONG_START_BYTE
 - bno055.h, 42
- euler_index
 - main.c, 63
- euler_y
 - main.c, 63
- euler_z
 - main.c, 63
- eulerScale
 - bno055.c, 25
- FILTER_SIZE
 - main.c, 61
- forward_channel
 - MotorDriver, 17
- fsr_controller.c, 54
 - apply_low_pass_filter, 54
 - get_averaged_adc_value, 55
 - map, 55
- fsr_controller.h, 56
 - apply_low_pass_filter, 56
 - get_averaged_adc_value, 57
 - map, 57
- g_trigger_val
 - Receiver.c, 71
- g_wheel_val
 - Receiver.c, 71
- get_averaged_adc_value
 - fsr_controller.c, 55
 - fsr_controller.h, 57
- gyro
 - bno055_calibration_offset_t, 13
 - bno055_calibration_state_t, 14
- gyrState
 - bno055_self_test_result_t, 15
- hadc1
 - main.c, 63
- HAL_ADC_MspDeInit
 - stm32f4xx_hal_msp.c, 74
- HAL_ADC_MspInit
 - stm32f4xx_hal_msp.c, 75
- HAL_I2C_MspDeInit
 - stm32f4xx_hal_msp.c, 75
- HAL_I2C_MspInit
 - stm32f4xx_hal_msp.c, 75
- HAL_MspInit
 - stm32f4xx_hal_msp.c, 76
- HAL_TIM_Base_MspDeInit
 - stm32f4xx_hal_msp.c, 76
- HAL_TIM_Base_MspInit
 - stm32f4xx_hal_msp.c, 76
- HAL_TIM_IC_MspDeInit
 - stm32f4xx_hal_msp.c, 76
- HAL_TIM_IC_MspInit
 - stm32f4xx_hal_msp.c, 77
- HAL_TIM_MspPostInit
 - stm32f4xx_hal_msp.c, 77
- HAL_TIM_PWM_MspDeInit
 - stm32f4xx_hal_msp.c, 77
- HAL_TIM_PWM_MspInit
 - stm32f4xx_hal_msp.c, 78
- HardFault_Handler
 - stm32f4xx_it.c, 80
- hdma_i2c1_rx
 - main.c, 63
 - stm32f4xx_hal_msp.c, 78
 - stm32f4xx_it.c, 81
- hdma_i2c1_tx
 - main.c, 63
 - stm32f4xx_hal_msp.c, 78
 - stm32f4xx_it.c, 81
- hi2c1
 - main.c, 64
- highPWM
 - RcReceiver, 18
- HSE_VALUE
 - STM32F4xx_System_Private_Includes, 8
- HSI_VALUE
 - STM32F4xx_System_Private_Includes, 8
- htim
 - MotorDriver, 17
 - RcReceiver, 18
- htim1
 - main.c, 64
- htim2

- main.c, 64
- htim3
 - main.c, 64
- hysteresis
 - main.c, 64
- initialise_monitor_handles
 - syscalls.c, 85
- initializeRCReceiver
 - Receiver.c, 70
 - Receiver.h, 72
- isRadioOn
 - main.c, 61
- lowPWM
 - RCReceiver, 18
- mag
 - bno055_calibration_offset_t, 13
 - bno055_calibration_radius_t, 13
 - bno055_calibration_state_t, 14
- magScale
 - bno055.c, 25
- magState
 - bno055_self_test_result_t, 15
- main
 - main.c, 61
- main.c, 59
 - adc_value, 63
 - euler_index, 63
 - euler_y, 63
 - euler_z, 63
 - FILTER_SIZE, 61
 - hadc1, 63
 - hdma_i2c1_rx, 63
 - hdma_i2c1_tx, 63
 - hi2c1, 64
 - htim1, 64
 - htim2, 64
 - htim3, 64
 - hysteresis, 64
 - isRadioOn, 61
 - main, 61
 - message, 64
 - motor1, 64
 - motor2, 64
 - pointer_rc, 64
 - press_detected, 64
 - receiver, 65
 - servo_speed, 65
 - state, 65
 - sum_y, 65
 - sum_z, 65
 - SystemClock_Config, 63
 - threshold, 65
 - x_position, 65
 - y_position, 65
- map
 - fsr_controller.c, 55
 - fsr_controller.h, 57
- MAX_DUTY_CYCLE
 - Motor_Driver.c, 66
- mcuState
 - bno055_self_test_result_t, 15
- MemManage_Handler
 - stm32f4xx_it.c, 80
- message
 - main.c, 64
 - RCReceiver, 18
- MESSAGE_LENGTH
 - Receiver.c, 69
 - Receiver.h, 71
- motor1
 - main.c, 64
- motor2
 - main.c, 64
- motor_disable
 - Motor_Driver.c, 66
 - Motor_Driver.h, 67
- Motor_Driver.c, 66
 - MAX_DUTY_CYCLE, 66
 - motor_disable, 66
 - motor_enable, 66
 - motor_set_duty_cycle, 67
- Motor_Driver.h, 67
 - motor_disable, 67
 - motor_enable, 68
 - motor_set_duty_cycle, 68
- motor_enable
 - Motor_Driver.c, 66
 - Motor_Driver.h, 68
- motor_set_duty_cycle
 - Motor_Driver.c, 67
 - Motor_Driver.h, 68
- MotorDriver, 16
 - backward_channel, 17
 - forward_channel, 17
 - htim, 17
- NMI_Handler
 - stm32f4xx_it.c, 80
- nominalPWM
 - RCReceiver, 18
- offset
 - bno055_calibration_data_t, 12
- PendSV_Handler
 - stm32f4xx_it.c, 80
- pointer_rc
 - main.c, 64
- press_detected
 - main.c, 64
- quaScale
 - bno055.c, 25
- radius

- [_sbrk](#), [86](#)
- [system_stm32f4xx.c](#), [86](#)
- [SystemClock_Config](#)
 - [main.c](#), [63](#)
- [SystemCoreClock](#)
 - [STM32F4xx_System_Private_Variables](#), [8](#)
- [SystemCoreClockUpdate](#)
 - [STM32F4xx_System_Private_Functions](#), [9](#)
- [SystemInit](#)
 - [STM32F4xx_System_Private_Functions](#), [10](#)
- [SysTick_Handler](#)
 - [stm32f4xx_it.c](#), [81](#)
-
- [tempScale](#)
 - [bno055.c](#), [25](#)
- [threshold](#)
 - [main.c](#), [65](#)
- [Trigger_Channel1_Fall](#)
 - [RCReceiver](#), [18](#)
- [Trigger_Channel1_Rise](#)
 - [RCReceiver](#), [18](#)
- [Trigger_Channel2_Fall](#)
 - [RCReceiver](#), [19](#)
- [Trigger_Channel2_Rise](#)
 - [RCReceiver](#), [19](#)
-
- [UsageFault_Handler](#)
 - [stm32f4xx_it.c](#), [81](#)
-
- [w](#)
 - [bno055_vector_t](#), [15](#)
-
- [x](#)
 - [bno055_axis_map_t](#), [11](#)
 - [bno055_vector_t](#), [15](#)
 - [bno055_vector_xyz_int16_t](#), [16](#)
- [x_position](#)
 - [main.c](#), [65](#)
- [x_sign](#)
 - [bno055_axis_map_t](#), [11](#)
-
- [y](#)
 - [bno055_axis_map_t](#), [11](#)
 - [bno055_vector_t](#), [15](#)
 - [bno055_vector_xyz_int16_t](#), [16](#)
- [y_position](#)
 - [main.c](#), [65](#)
- [y_sign](#)
 - [bno055_axis_map_t](#), [11](#)
-
- [z](#)
 - [bno055_axis_map_t](#), [11](#)
 - [bno055_vector_t](#), [16](#)
 - [bno055_vector_xyz_int16_t](#), [16](#)
- [z_sign](#)
 - [bno055_axis_map_t](#), [12](#)