

Micras Firmware

Generated by Doxygen 1.9.1

1 Micras	1
2 Todo List	3
3 Namespace Index	5
3.1 Namespace List	5
4 Class Index	7
4.1 Class List	7
5 File Index	11
5.1 File List	11
6 Namespace Documentation	15
6.1 hal Namespace Reference	15
6.2 proxy Namespace Reference	15
6.2.1 Variable Documentation	16
6.2.1.1 Fundamental	16
7 Class Documentation	17
7.1 hal::AdcDma Class Reference	17
7.1.1 Detailed Description	18
7.1.2 Constructor & Destructor Documentation	18
7.1.2.1 AdcDma()	18
7.1.3 Member Function Documentation	18
7.1.3.1 start_dma()	18
7.1.3.2 stop_dma()	19
7.1.4 Member Data Documentation	19
7.1.4.1 max_reading	19
7.1.4.2 reference_voltage	19
7.2 proxy::Argb< num_of_leds > Class Template Reference	20
7.2.1 Detailed Description	21
7.2.2 Constructor & Destructor Documentation	21
7.2.2.1 Argb()	21
7.2.3 Member Function Documentation	21
7.2.3.1 set_color() [1/2]	22
7.2.3.2 set_color() [2/2]	22
7.2.3.3 turn_off() [1/2]	22
7.2.3.4 turn_off() [2/2]	22
7.3 proxy::Battery Class Reference	23
7.3.1 Detailed Description	23
7.3.2 Constructor & Destructor Documentation	24
7.3.2.1 Battery()	24
7.3.3 Member Function Documentation	25

7.3.3.1 get_voltage()	25
7.3.3.2 get_voltage_raw()	25
7.4 proxy::Button Class Reference	25
7.4.1 Detailed Description	26
7.4.2 Member Enumeration Documentation	26
7.4.2.1 PullResistor	26
7.4.2.2 Status	26
7.4.3 Constructor & Destructor Documentation	27
7.4.3.1 Button()	27
7.4.4 Member Function Documentation	27
7.4.4.1 get_status()	27
7.4.4.2 is_pressed()	27
7.5 proxy::Buzzer Class Reference	28
7.5.1 Detailed Description	28
7.5.2 Constructor & Destructor Documentation	28
7.5.2.1 Buzzer()	28
7.5.3 Member Function Documentation	29
7.5.3.1 play()	29
7.5.3.2 stop()	29
7.5.3.3 update()	29
7.6 proxy::Argb< num_of_leds >::Color Struct Reference	30
7.6.1 Detailed Description	30
7.6.2 Member Data Documentation	30
7.6.2.1 blue	31
7.6.2.2 green	31
7.6.2.3 red	31
7.7 hal::AdcDma::Config Struct Reference	31
7.7.1 Detailed Description	32
7.7.2 Member Data Documentation	32
7.7.2.1 handle	32
7.7.2.2 init_function	32
7.7.2.3 max_reading	32
7.7.2.4 reference_voltage	32
7.8 hal::Crc::Config Struct Reference	33
7.8.1 Detailed Description	33
7.8.2 Member Data Documentation	33
7.8.2.1 handle	33
7.9 hal::Encoder::Config Struct Reference	34
7.9.1 Detailed Description	34
7.9.2 Member Data Documentation	34
7.9.2.1 handle	34
7.9.2.2 init_function	35

7.9.2.3 timer_channel	35
7.10 hal::Gpio::Config Struct Reference	35
7.10.1 Detailed Description	36
7.10.2 Member Data Documentation	36
7.10.2.1 pin	36
7.10.2.2 port	36
7.11 hal::Pwm::Config Struct Reference	36
7.11.1 Detailed Description	37
7.11.2 Member Data Documentation	37
7.11.2.1 handle	37
7.11.2.2 init_function	37
7.11.2.3 timer_channel	37
7.12 hal::PwmDma::Config Struct Reference	38
7.12.1 Detailed Description	38
7.12.2 Member Data Documentation	38
7.12.2.1 handle	38
7.12.2.2 init_function	39
7.12.2.3 timer_channel	39
7.13 hal::Spi::Config Struct Reference	39
7.13.1 Detailed Description	40
7.13.2 Member Data Documentation	40
7.13.2.1 gpio	40
7.13.2.2 handle	40
7.13.2.3 init_function	40
7.13.2.4 timeout	40
7.14 hal::Timer::Config Struct Reference	41
7.14.1 Detailed Description	41
7.14.2 Member Data Documentation	41
7.14.2.1 handle	41
7.14.2.2 init_function	42
7.15 proxy::Argb< num_of_leds >::Config Struct Reference	42
7.15.1 Detailed Description	43
7.15.2 Member Data Documentation	43
7.15.2.1 pwm	43
7.16 proxy::Battery::Config Struct Reference	43
7.16.1 Detailed Description	44
7.16.2 Member Data Documentation	44
7.16.2.1 adc	44
7.16.2.2 voltage_divider	44
7.17 proxy::Button::Config Struct Reference	44
7.17.1 Detailed Description	45
7.17.2 Member Data Documentation	45

7.17.2.1 debounce_delay	45
7.17.2.2 extra_long_press_delay	46
7.17.2.3 gpio	46
7.17.2.4 long_press_delay	46
7.17.2.5 pull_resistor	46
7.18 proxy::Buzzer::Config Struct Reference	46
7.18.1 Detailed Description	47
7.18.2 Member Data Documentation	47
7.18.2.1 pwm	47
7.19 proxy::CurrentSensors< num_of_sensors >::Config Struct Reference	48
7.19.1 Detailed Description	48
7.19.2 Member Data Documentation	48
7.19.2.1 adc	48
7.19.2.2 shunt_resistor	49
7.20 proxy::DipSwitch< num_of_switches >::Config Struct Reference	49
7.20.1 Detailed Description	49
7.20.2 Member Data Documentation	50
7.20.2.1 gpio_array	50
7.21 proxy::DistanceSensors< num_of_sensors >::Config Struct Reference	50
7.21.1 Detailed Description	50
7.21.2 Member Data Documentation	51
7.21.2.1 adc	51
7.21.2.2 led_pwm	51
7.21.2.3 max_distance	51
7.22 proxy::Fan::Config Struct Reference	51
7.22.1 Detailed Description	52
7.22.2 Member Data Documentation	52
7.22.2.1 direction_gpio	52
7.22.2.2 enable_gpio	52
7.22.2.3 pwm	52
7.23 proxy::Imu::Config Struct Reference	52
7.23.1 Detailed Description	53
7.23.2 Member Data Documentation	53
7.23.2.1 accelerometer_data_rate	53
7.23.2.2 accelerometer_filter	53
7.23.2.3 accelerometer_scale	53
7.23.2.4 gyroscope_data_rate	53
7.23.2.5 gyroscope_filter	54
7.23.2.6 gyroscope_scale	54
7.23.2.7 orientation_data_rate	54
7.23.2.8 spi	54
7.24 proxy::Led::Config Struct Reference	54

7.24.1 Detailed Description	55
7.24.2 Member Data Documentation	55
7.24.2.1 gpio	55
7.25 proxy::Locomotion::Config Struct Reference	56
7.25.1 Detailed Description	56
7.25.2 Member Data Documentation	56
7.25.2.1 enable_gpio	56
7.25.2.2 pwm_left_bwd	57
7.25.2.3 pwm_left_fwd	57
7.25.2.4 pwm_right_bwd	57
7.25.2.5 pwm_right_fwd	57
7.26 proxy::RotarySensor::Config Struct Reference	57
7.26.1 Detailed Description	58
7.26.2 Member Data Documentation	58
7.26.2.1 crc	58
7.26.2.2 encoder	58
7.26.2.3 registers	58
7.26.2.4 resolution	58
7.26.2.5 spi	58
7.27 proxy::Storage::Config Struct Reference	59
7.27.1 Detailed Description	59
7.27.2 Member Data Documentation	59
7.27.2.1 number_of_pages	59
7.27.2.2 start_page	60
7.28 proxy::TorqueSensors< num_of_sensors >::Config Struct Reference	60
7.28.1 Detailed Description	60
7.28.2 Member Data Documentation	61
7.28.2.1 current_sensors	61
7.28.2.2 max_torque	61
7.29 hal::Crc Class Reference	61
7.29.1 Detailed Description	62
7.29.2 Constructor & Destructor Documentation	62
7.29.2.1 Crc()	62
7.29.3 Member Function Documentation	62
7.29.3.1 calculate()	62
7.30 proxy::CurrentSensors< num_of_sensors > Class Template Reference	63
7.30.1 Detailed Description	64
7.30.2 Constructor & Destructor Documentation	64
7.30.2.1 CurrentSensors()	64
7.30.3 Member Function Documentation	64
7.30.3.1 get_current()	64
7.30.3.2 get_current_raw()	65

7.31 proxy::DipSwitch< num_of_switches > Class Template Reference	65
7.31.1 Detailed Description	66
7.31.2 Constructor & Destructor Documentation	66
7.31.2.1 DipSwitch()	66
7.31.3 Member Function Documentation	67
7.31.3.1 get_switch_state()	67
7.31.3.2 get_switches_value()	67
7.32 Registers::Disable Union Reference	68
7.32.1 Detailed Description	68
7.32.2 Member Data Documentation	69
7.32.2.1 fields	69
7.32.2.2 raw	69
7.33 proxy::DistanceSensors< num_of_sensors > Class Template Reference	69
7.33.1 Detailed Description	70
7.33.2 Constructor & Destructor Documentation	70
7.33.2.1 DistanceSensors()	70
7.33.3 Member Function Documentation	71
7.33.3.1 get_distance()	71
7.33.3.2 get_distance_raw()	71
7.33.3.3 set_led_intensity()	72
7.34 Registers::Ecc Union Reference	72
7.34.1 Member Data Documentation	73
7.34.1.1 fields	73
7.34.1.2 raw	73
7.35 hal::Encoder Class Reference	73
7.35.1 Detailed Description	74
7.35.2 Constructor & Destructor Documentation	74
7.35.2.1 Encoder()	74
7.35.3 Member Function Documentation	74
7.35.3.1 get_counter()	74
7.36 proxy::Fan Class Reference	75
7.36.1 Detailed Description	75
7.36.2 Member Enumeration Documentation	76
7.36.2.1 RotationDirection	76
7.36.3 Constructor & Destructor Documentation	76
7.36.3.1 Fan()	76
7.36.4 Member Function Documentation	76
7.36.4.1 disable()	76
7.36.4.2 enable()	77
7.36.4.3 set_speed()	77
7.36.4.4 stop()	77
7.37 proxy::RotarySensor::CommandFrame::Fields Struct Reference	77

7.37.1 Member Data Documentation	78
7.37.1.1 address	78
7.37.1.2 crc	78
7.37.1.3 do_not_care	79
7.37.1.4 rw	79
7.38 proxy::RotarySensor::DataFrame::Fields Struct Reference	79
7.38.1 Member Data Documentation	80
7.38.1.1 crc	80
7.38.1.2 data	80
7.38.1.3 error	80
7.38.1.4 warning	80
7.39 Registers::Disable::Fields Struct Reference	80
7.39.1 Member Data Documentation	81
7.39.1.1 ABI_off	81
7.39.1.2 FILTER_disable	81
7.39.1.3 na	82
7.39.1.4 UVW_off	82
7.40 Registers::Ecc::Fields Struct Reference	82
7.40.1 Member Data Documentation	82
7.40.1.1 ECC_chsum	83
7.40.1.2 ECC_en	83
7.41 Registers::Settings1::Fields Struct Reference	83
7.41.1 Member Data Documentation	84
7.41.1.1 Dia3_en	84
7.41.1.2 Dia4_en	84
7.41.1.3 K_max	84
7.41.1.4 K_min	84
7.42 Registers::Settings2::Fields Struct Reference	84
7.42.1 Member Data Documentation	85
7.42.1.1 ABI_DEC	85
7.42.1.2 DAECDIS	86
7.42.1.3 Data_select	86
7.42.1.4 DIR	86
7.42.1.5 IWIDTH	86
7.42.1.6 NOISESET	86
7.42.1.7 PWMon	86
7.42.1.8 UVW_ABI	86
7.43 Registers::Settings3::Fields Struct Reference	87
7.43.1 Member Data Documentation	87
7.43.1.1 ABIRES	87
7.43.1.2 HYS	87
7.43.1.3 UVWPP	88

7.44 Registers::Zposl::Fields Struct Reference	88
7.44.1 Member Data Documentation	88
7.44.1.1 Dia1_en	89
7.44.1.2 Dia2_en	89
7.44.1.3 ZPOSL	89
7.45 Registers::Zposm::Fields Struct Reference	89
7.45.1 Member Data Documentation	90
7.45.1.1 ZPOSM	90
7.46 hal::Flash Class Reference	90
7.46.1 Detailed Description	91
7.46.2 Constructor & Destructor Documentation	91
7.46.2.1 Flash()	91
7.46.3 Member Function Documentation	91
7.46.3.1 erase_pages()	91
7.46.3.2 read() [1/2]	91
7.46.3.3 read() [2/2]	92
7.46.3.4 write() [1/2]	92
7.46.3.5 write() [2/2]	92
7.47 hal::Gpio Class Reference	93
7.47.1 Detailed Description	94
7.47.2 Constructor & Destructor Documentation	94
7.47.2.1 Gpio()	94
7.47.3 Member Function Documentation	94
7.47.3.1 read()	94
7.47.3.2 toggle()	94
7.47.3.3 write()	94
7.48 proxy::Imu Class Reference	95
7.48.1 Detailed Description	96
7.48.2 Member Enumeration Documentation	96
7.48.2.1 Axis	96
7.48.3 Constructor & Destructor Documentation	96
7.48.3.1 Imu()	96
7.48.4 Member Function Documentation	96
7.48.4.1 get_angular_velocity()	96
7.48.4.2 get_linear_acceleration()	97
7.48.4.3 get_orientation()	97
7.48.4.4 update_data()	98
7.49 ISerializable Class Reference	98
7.49.1 Detailed Description	99
7.49.2 Constructor & Destructor Documentation	99
7.49.2.1 ~ISerializable()	99
7.49.2.2 ISerializable() [1/2]	99

7.49.2.3 ISerializable() [2/2]	99
7.49.3 Member Function Documentation	99
7.49.3.1 deserialize()	99
7.49.3.2 operator=() [1/2]	100
7.49.3.3 operator=() [2/2]	100
7.49.3.4 serialize()	100
7.50 proxy::Led Class Reference	100
7.50.1 Detailed Description	101
7.50.2 Constructor & Destructor Documentation	101
7.50.2.1 Led()	101
7.50.3 Member Function Documentation	101
7.50.3.1 toggle()	101
7.50.3.2 turn_off()	102
7.50.3.3 turn_on()	102
7.51 proxy::Locomotion Class Reference	102
7.51.1 Detailed Description	103
7.51.2 Constructor & Destructor Documentation	103
7.51.2.1 Locomotion()	103
7.51.3 Member Function Documentation	103
7.51.3.1 disable()	103
7.51.3.2 enable()	104
7.51.3.3 set_speed()	104
7.51.3.4 set_wheel_speed()	104
7.51.3.5 stop()	104
7.51.3.6 stop_left()	105
7.51.3.7 stop_right()	105
7.52 hal::Mcu Class Reference	105
7.52.1 Detailed Description	106
7.52.2 Constructor & Destructor Documentation	106
7.52.2.1 Mcu()	106
7.52.3 Member Function Documentation	106
7.52.3.1 init()	106
7.53 MicrasController Class Reference	106
7.53.1 Detailed Description	107
7.53.2 Constructor & Destructor Documentation	107
7.53.2.1 MicrasController()	107
7.53.3 Member Function Documentation	107
7.53.3.1 run()	107
7.54 hal::Pwm Class Reference	108
7.54.1 Detailed Description	108
7.54.2 Constructor & Destructor Documentation	108
7.54.2.1 Pwm()	108

7.54.3 Member Function Documentation	109
7.54.3.1 set_duty_cycle()	109
7.54.3.2 set_frequency()	109
7.55 hal::PwmDma Class Reference	109
7.55.1 Detailed Description	110
7.55.2 Constructor & Destructor Documentation	110
7.55.2.1 PwmDma()	110
7.55.3 Member Function Documentation	111
7.55.3.1 get_compare()	111
7.55.3.2 start_dma()	111
7.55.3.3 stop_dma()	111
7.56 Registers Struct Reference	112
7.56.1 Detailed Description	113
7.56.2 Member Data Documentation	113
7.56.2.1 disable	113
7.56.2.2 disable_addr	113
7.56.2.3 ecc	113
7.56.2.4 ecc_addr	113
7.56.2.5 settings1	114
7.56.2.6 settings1_addr	114
7.56.2.7 settings2	114
7.56.2.8 settings2_addr	114
7.56.2.9 settings3	114
7.56.2.10 settings3_addr	114
7.56.2.11 zposl	114
7.56.2.12 zposl_addr	114
7.56.2.13 zposm	115
7.56.2.14 zposm_addr	115
7.57 proxy::RotarySensor Class Reference	115
7.57.1 Detailed Description	116
7.57.2 Constructor & Destructor Documentation	116
7.57.2.1 RotarySensor()	116
7.57.3 Member Function Documentation	116
7.57.3.1 get_position()	116
7.58 Registers::Settings1 Union Reference	117
7.58.1 Member Data Documentation	117
7.58.1.1 fields	118
7.58.1.2 raw	118
7.59 Registers::Settings2 Union Reference	118
7.59.1 Member Data Documentation	119
7.59.1.1 fields	119
7.59.1.2 raw	119

7.60 Registers::Settings3 Union Reference	119
7.60.1 Member Data Documentation	120
7.60.1.1 fields	120
7.60.1.2 raw	121
7.61 hal::Spi Class Reference	121
7.61.1 Detailed Description	122
7.61.2 Constructor & Destructor Documentation	122
7.61.2.1 Spi()	122
7.61.3 Member Function Documentation	122
7.61.3.1 receive()	122
7.61.3.2 select_device()	122
7.61.3.3 transmit()	123
7.61.3.4 unselect_device()	123
7.62 proxy::Storage Class Reference	123
7.62.1 Detailed Description	124
7.62.2 Constructor & Destructor Documentation	124
7.62.2.1 Storage()	124
7.62.3 Member Function Documentation	124
7.62.3.1 create() [1/2]	125
7.62.3.2 create() [2/2]	125
7.62.3.3 save()	125
7.62.3.4 sync() [1/2]	125
7.62.3.5 sync() [2/2]	126
7.63 hal::Timer Class Reference	126
7.63.1 Detailed Description	127
7.63.2 Constructor & Destructor Documentation	128
7.63.2.1 Timer() [1/2]	128
7.63.2.2 Timer() [2/2]	128
7.63.3 Member Function Documentation	128
7.63.3.1 elapsed_time_ms()	128
7.63.3.2 elapsed_time_us()	128
7.63.3.3 reset_ms()	129
7.63.3.4 reset_us()	129
7.63.3.5 sleep_ms()	129
7.63.3.6 sleep_us()	129
7.64 proxy::TorqueSensors< num_of_sensors > Class Template Reference	130
7.64.1 Detailed Description	131
7.64.2 Constructor & Destructor Documentation	131
7.64.2.1 TorqueSensors()	131
7.64.3 Member Function Documentation	131
7.64.3.1 get_current()	131
7.64.3.2 get_torque()	132

7.65 Registers::Zposl Union Reference	132
7.65.1 Member Data Documentation	133
7.65.1.1 Fields	133
7.65.1.2 raw	134
7.66 Registers::Zposm Union Reference	134
7.66.1 Member Data Documentation	135
7.66.1.1 fields	135
7.66.1.2 raw	135
8 File Documentation	137
8.1 cfg/target.cpp File Reference	137
8.1.1 Detailed Description	137
8.1.2 Variable Documentation	138
8.1.2.1 argb_config	138
8.1.2.2 battery_config	138
8.1.2.3 button_config	138
8.1.2.4 buzzer_config	138
8.1.2.5 dip_switch_config	139
8.1.2.6 distance_sensors_config	139
8.1.2.7 fan_config	139
8.1.2.8 imu_config	140
8.1.2.9 led_config	140
8.1.2.10 locomotion_config	140
8.1.2.11 rotary_sensor_left_config	141
8.1.2.12 rotary_sensor_reg_config	141
8.1.2.13 rotary_sensor_right_config	141
8.1.2.14 torque_sensors_config	142
8.2 cfg/target.hpp File Reference	142
8.2.1 Detailed Description	143
8.2.2 Variable Documentation	143
8.2.2.1 argb_config	143
8.2.2.2 battery_config	143
8.2.2.3 button_config	143
8.2.2.4 buzzer_config	144
8.2.2.5 dip_switch_config	144
8.2.2.6 distance_sensors_config	144
8.2.2.7 fan_config	144
8.2.2.8 imu_config	144
8.2.2.9 led_config	144
8.2.2.10 locomotion_config	144
8.2.2.11 rotary_sensor_left_config	144
8.2.2.12 rotary_sensor_right_config	145

8.2.2.13 torque_sensors_config	145
8.3 inc/controller/micras_controller.hpp File Reference	145
8.4 inc/hal/adc_dma.hpp File Reference	146
8.4.1 Detailed Description	146
8.5 inc/hal/crc.hpp File Reference	147
8.5.1 Detailed Description	147
8.6 inc/hal/encoder.hpp File Reference	148
8.6.1 Detailed Description	148
8.7 inc/hal/flash.hpp File Reference	149
8.7.1 Detailed Description	149
8.8 inc/hal/gpio.hpp File Reference	149
8.8.1 Detailed Description	150
8.9 inc/hal/mcu.hpp File Reference	150
8.9.1 Detailed Description	151
8.10 inc/hal/pwm.hpp File Reference	151
8.10.1 Detailed Description	151
8.11 inc/hal/pwm_dma.hpp File Reference	152
8.11.1 Detailed Description	152
8.12 inc/hal/spi.hpp File Reference	153
8.12.1 Detailed Description	154
8.13 inc/hal/timer.hpp File Reference	154
8.13.1 Detailed Description	154
8.14 inc/proxy/argb.hpp File Reference	155
8.14.1 Detailed Description	156
8.15 inc/proxy/battery.hpp File Reference	156
8.15.1 Detailed Description	157
8.16 inc/proxy/button.hpp File Reference	157
8.16.1 Detailed Description	158
8.17 inc/proxy/buzzer.hpp File Reference	158
8.17.1 Detailed Description	159
8.18 inc/proxy/current_sensors.hpp File Reference	159
8.18.1 Detailed Description	160
8.19 inc/proxy/dip_switch.hpp File Reference	161
8.19.1 Detailed Description	162
8.20 inc/proxy/distance_sensors.hpp File Reference	162
8.20.1 Detailed Description	163
8.21 inc/proxy/fan.hpp File Reference	163
8.21.1 Detailed Description	164
8.22 inc/proxy/imu.hpp File Reference	164
8.22.1 Detailed Description	165
8.23 inc/proxy/led.hpp File Reference	165
8.23.1 Detailed Description	166

8.24 inc/proxy/locomotion.hpp File Reference	167
8.24.1 Detailed Description	168
8.25 inc/proxy/rotary_sensor.hpp File Reference	168
8.25.1 Detailed Description	169
8.26 inc/proxy/rotary_sensor_reg.hpp File Reference	169
8.26.1 Detailed Description	170
8.27 inc/proxy/serializable_interface.hpp File Reference	170
8.27.1 Detailed Description	171
8.28 inc/proxy/storage.hpp File Reference	171
8.28.1 Detailed Description	172
8.29 inc/proxy/torque_sensors.hpp File Reference	172
8.29.1 Detailed Description	173
8.30 README.md File Reference	173
8.31 src/controller/micras_controller.cpp File Reference	173
8.31.1 Detailed Description	174
8.32 src/hal/adc_dma.cpp File Reference	174
8.32.1 Detailed Description	174
8.33 src/hal/crc.cpp File Reference	175
8.33.1 Detailed Description	175
8.34 src/hal/encoder.cpp File Reference	175
8.34.1 Detailed Description	176
8.35 src/hal/flash.cpp File Reference	176
8.35.1 Detailed Description	176
8.36 src/hal/gpio.cpp File Reference	177
8.36.1 Detailed Description	177
8.37 src/hal/mcu.cpp File Reference	177
8.37.1 Detailed Description	178
8.37.2 Function Documentation	178
8.37.2.1 SystemClock_Config()	178
8.38 src/hal/pwm.cpp File Reference	178
8.38.1 Detailed Description	179
8.39 src/hal/pwm_dma.cpp File Reference	179
8.39.1 Detailed Description	179
8.40 src/hal/spi.cpp File Reference	180
8.40.1 Detailed Description	180
8.41 src/hal/timer.cpp File Reference	180
8.41.1 Detailed Description	181
8.42 src/main.cpp File Reference	181
8.42.1 Detailed Description	182
8.42.2 Function Documentation	182
8.42.2.1 main()	182
8.43 src/proxy/argb.cpp File Reference	182

8.43.1 Detailed Description	183
8.43.2 Macro Definition Documentation	183
8.43.2.1 MICRAS_PROXY_ARGB_CPP	183
8.44 src/proxy/battery.cpp File Reference	183
8.44.1 Detailed Description	184
8.45 src/proxy/button.cpp File Reference	184
8.45.1 Detailed Description	184
8.46 src/proxy/buzzer.cpp File Reference	185
8.46.1 Detailed Description	185
8.47 src/proxy/current_sensors.cpp File Reference	185
8.47.1 Detailed Description	186
8.47.2 Macro Definition Documentation	186
8.47.2.1 MICRAS_PROXY_CURRENT_SENSORS_CPP	186
8.48 src/proxy/dip_switch.cpp File Reference	187
8.48.1 Detailed Description	187
8.48.2 Macro Definition Documentation	187
8.48.2.1 MICRAS_PROXY_DIP_SWITCH_CPP	187
8.49 src/proxy/distance_sensors.cpp File Reference	188
8.49.1 Macro Definition Documentation	188
8.49.1.1 MICRAS_PROXY_DISTANCE_SENSORS_CPP	188
8.50 src/proxy/fan.cpp File Reference	189
8.50.1 Detailed Description	189
8.51 src/proxy/imu.cpp File Reference	189
8.51.1 Detailed Description	190
8.52 src/proxy/led.cpp File Reference	190
8.52.1 Detailed Description	191
8.53 src/proxy/locomotion.cpp File Reference	191
8.53.1 Detailed Description	192
8.54 src/proxy/rotary_sensor.cpp File Reference	192
8.54.1 Detailed Description	193
8.55 src/proxy/storage.cpp File Reference	193
8.55.1 Detailed Description	194
8.56 src/proxy/torque_sensors.cpp File Reference	194
8.56.1 Detailed Description	195
8.56.2 Macro Definition Documentation	195
8.56.2.1 MICRAS_PROXY_TORQUE_SENSORS_CPP	195

Chapter 1

Micras

Chapter 2

Todo List

Member [proxy::Imu::get_orientation](#) (Axis axis) const
implement function using sensor fusion

Chapter 3

Namespace Index

3.1 Namespace List

Here is a list of all namespaces with brief descriptions:

hal	15
proxy	15

Chapter 4

Class Index

4.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

hal::AdcDma	17
Class to handle ADC peripheral on STM32 microcontrollers using DMA	
proxy::Argb< num_of_leds >	20
Class for controlling an addressable RGB LED	
proxy::Battery	23
Class for getting the battery voltage	
proxy::Button	25
Class for controlling a button	
proxy::Buzzer	28
Class for controlling a buzzer	
proxy::Argb< num_of_leds >::Color	30
Structure for storing color information	
hal::AdcDma::Config	31
Configuration structure for ADC DMA	
hal::Crc::Config	33
CRC configuration struct	
hal::Encoder::Config	34
Encoder configuration struct	
hal::Gpio::Config	35
Configuration structure for GPIO pin	
hal::Pwm::Config	36
PWM configuration struct	
hal::PwmDma::Config	38
PWM configuration struct	
hal::Spi::Config	39
SPI configuration struct	
hal::Timer::Config	41
Timer configuration struct	
proxy::Argb< num_of_leds >::Config	42
Configuration structure for the addressable RGB LED	
proxy::Battery::Config	43
Configuration structure for the battery	
proxy::Button::Config	44
Configuration structure for button	
proxy::Buzzer::Config	46
Configuration structure for the buzzer	

proxy::CurrentSensors< num_of_sensors >::Config	
Configuration structure for current sensors	48
proxy::DipSwitch< num_of_switches >::Config	
Configuration struct for DipSwitch	49
proxy::DistanceSensors< num_of_sensors >::Config	
Configuration structure for distance sensors	50
proxy::Fan::Config	
Configuration structure for the fan	51
proxy::Imu::Config	
IMU configuration struct	52
proxy::Led::Config	
Configuration structure for LED	54
proxy::Locomotion::Config	
Configuration structure for the locomotion	56
proxy::RotarySensor::Config	
Rotary sensor configuration struct	57
proxy::Storage::Config	
Configuration structure for the storage	59
proxy::TorqueSensors< num_of_sensors >::Config	
Configuration structure for torque sensors	60
hal::Crc	
Class to handle the cyclic redundancy check peripheral on STM32 microcontrollers	61
proxy::CurrentSensors< num_of_sensors >	
Class for controlling CurrentSensors	63
proxy::DipSwitch< num_of_switches >	
Class for controlling a dip switch	65
Registers::Disable	
Registers union types definition	68
proxy::DistanceSensors< num_of_sensors >	
Class for controlling DistanceSensors	69
Registers::Ecc	72
hal::Encoder	
Class to handle encoder peripheral on STM32 microcontrollers	73
proxy::Fan	
Class for controlling the fan driver	75
proxy::RotarySensor::CommandFrame::Fields	77
proxy::RotarySensor::DataFrame::Fields	79
Registers::Disable::Fields	80
Registers::Ecc::Fields	82
Registers::Settings1::Fields	83
Registers::Settings2::Fields	84
Registers::Settings3::Fields	87
Registers::Zposl::Fields	88
Registers::Zposm::Fields	89
hal::Flash	
Class to handle flash memory on STM32 microcontrollers	90
hal::Gpio	
Class for controlling GPIO pins on STM32 microcontrollers	93
proxy::Imu	
Class to handle IMU peripheral on STM32 microcontrollers	95
ISerializable	
Interface class for serializable classes	98
proxy::Led	
Class for controlling an LED	100
proxy::Locomotion	
Class for controlling the locomotion driver	102
hal::Mcu	
Microcontroller unit class	105

MicrasController	
Class for controlling the Micras robot	106
hal::Pwm	
Class to handle PWM peripheral on STM32 microcontrollers	108
hal::PwmDma	
Class to handle PWM peripheral on STM32 microcontrollers using DMA	109
Registers	
Registers class for the rotary sensor configuration	112
proxy::RotarySensor	
Class to handle rotary sensor peripheral on STM32 microcontrollers	115
Registers::Settings1	117
Registers::Settings2	118
Registers::Settings3	119
hal::Spi	
Class to handle SPI peripheral on STM32 microcontrollers	121
proxy::Storage	
Class for controlling the storage	123
hal::Timer	
Class to handle timer peripheral on STM32 microcontrollers	126
proxy::TorqueSensors< num_of_sensors >	
Class for controlling TorqueSensors	130
Registers::Zposl	132
Registers::Zposm	134

Chapter 5

File Index

5.1 File List

Here is a list of all files with brief descriptions:

cfg/target.cpp		
Target specific configuration	137	
cfg/target.hpp		
Target specific configuration	142	
inc/controller/micras_controller.hpp		145
inc/hal/adc_dma.hpp		
ADC DMA HAL header	146	
inc/hal/crc.hpp		
STM32 CRC HAL wrapper	147	
inc/hal/encoder.hpp		
STM32 encoder HAL wrapper	148	
inc/hal/flash.hpp		
STM32 flash HAL wrapper	149	
inc/hal/gpio.hpp		
HAL GPIO class header	149	
inc/hal/mcu.hpp		
MCU related	150	
inc/hal/pwm.hpp		
STM32 PWM HAL wrapper	151	
inc/hal/pwm_dma.hpp		
STM32 PWM DMA HAL wrapper	152	
inc/hal/spi.hpp		
STM32 SPI HAL wrapper	153	
inc/hal/timer.hpp		
STM32 Timer HAL wrapper	154	
inc/proxy/argb.hpp		
Proxy Argb class declaration	155	
inc/proxy/battery.hpp		
Proxy Battery class declaration	156	
inc/proxy/button.hpp		
Proxy Button class header	157	
inc/proxy/buzzer.hpp		
Proxy Buzzer class declaration	158	
inc/proxy/current_sensors.hpp		
Proxy CurrentSensors class header	159	

inc/proxy/dip_switch.hpp	
Proxy Dip Switch class header	161
inc/proxy/distance_sensors.hpp	
Proxy DistanceSensors class header	162
inc/proxy/fan.hpp	
Proxy Fan class declaration	163
inc/proxy/imu.hpp	
STM32 IMU HAL wrapper	164
inc/proxy/led.hpp	
Proxy Led class header	165
inc/proxy/locomotion.hpp	
Proxy Locomotion class declaration	167
inc/proxy/rotary_sensor.hpp	
STM32 rotary sensor HAL wrapper	168
inc/proxy/rotary_sensor_reg.hpp	
AS5047U rotary sensor registers definition	169
inc/proxy/serializable_interface.hpp	
Serializable interface for all classes that need to be serialized	170
inc/proxy/storage.hpp	
Proxy Storage class declaration	171
inc/proxy/torque_sensors.hpp	
Proxy TorqueSensors class header	172
src/main.cpp	
Main function	181
src/controller/micras_controller.cpp	
Micras Controller class implementation	173
src/hal/adc_dma.cpp	
STM32 ADC DMA HAL wrapper	174
src/hal/crc.cpp	
STM32 CRC HAL wrapper	175
src/hal/encoder.cpp	
STM32 encoder HAL wrapper	175
src/hal/flash.cpp	
STM32 flash HAL wrapper	176
src/hal/gpio.cpp	
HAL GPIO class source	177
src/hal/mcu.cpp	
MCU related	177
src/hal/pwm.cpp	
STM32 PWM HAL wrapper	178
src/hal/pwm_dma.cpp	
STM32 PWM DMA HAL wrapper	179
src/hal/spi.cpp	
Proxy SPI Switch class source	180
src/hal/timer.cpp	
STM32 TIM HAL wrapper	180
src/proxy/argb.cpp	
Proxy Argb class implementation	182
src/proxy/battery.cpp	
Proxy Battery class implementation	183
src/proxy/button.cpp	
Proxy Button class source	184
src/proxy/buzzer.cpp	
Proxy Buzzer class implementation	185
src/proxy/current_sensors.cpp	
Proxy CurrentSensors class implementation	185
src/proxy/dip_switch.cpp	
Proxy DIP Switch class source	187

src/proxy/ distance_sensors.cpp	188
src/proxy/ fan.cpp	
Proxy Fan class source	189
src/proxy/ imu.cpp	
Proxy Imu class source	189
src/proxy/ led.cpp	
Proxy Led class source	190
src/proxy/ locomotion.cpp	
Proxy Locomotion class source	191
src/proxy/ rotary_sensor.cpp	
Proxy RotarySensor class source	192
src/proxy/ storage.cpp	
Proxy Storage class source	193
src/proxy/ torque_sensors.cpp	
Proxy TorqueSensors class implementation	194

Chapter 6

Namespace Documentation

6.1 hal Namespace Reference

Classes

- class [AdcDma](#)
Class to handle ADC peripheral on STM32 microcontrollers using DMA.
- class [Crc](#)
Class to handle the cyclic redundancy check peripheral on STM32 microcontrollers.
- class [Encoder](#)
Class to handle encoder peripheral on STM32 microcontrollers.
- class [Flash](#)
Class to handle flash memory on STM32 microcontrollers.
- class [Gpio](#)
Class for controlling GPIO pins on STM32 microcontrollers.
- class [Mcu](#)
Microcontroller unit class.
- class [Pwm](#)
Class to handle PWM peripheral on STM32 microcontrollers.
- class [PwmDma](#)
Class to handle PWM peripheral on STM32 microcontrollers using DMA.
- class [Spi](#)
Class to handle SPI peripheral on STM32 microcontrollers.
- class [Timer](#)
Class to handle timer peripheral on STM32 microcontrollers.

6.2 proxy Namespace Reference

Classes

- class [Argb](#)
Class for controlling an addressable RGB LED.
- class [Battery](#)
Class for getting the battery voltage.

- class [Button](#)
Class for controlling a button.
- class [Buzzer](#)
Class for controlling a buzzer.
- class [CurrentSensors](#)
Class for controlling [CurrentSensors](#).
- class [DipSwitch](#)
Class for controlling a dip switch.
- class [DistanceSensors](#)
Class for controlling [DistanceSensors](#).
- class [Fan](#)
Class for controlling the fan driver.
- class [Imu](#)
Class to handle IMU peripheral on STM32 microcontrollers.
- class [Led](#)
Class for controlling an LED.
- class [Locomotion](#)
Class for controlling the locomotion driver.
- class [RotarySensor](#)
Class to handle rotary sensor peripheral on STM32 microcontrollers.
- class [Storage](#)
Class for controlling the storage.
- class [TorqueSensors](#)
Class for controlling [TorqueSensors](#).

Variables

- `template<typename T >`
`concept Fundamental = std::is_fundamental<T>::value`

6.2.1 Variable Documentation

6.2.1.1 Fundamental

```
template<typename T >
concept proxy::Fundamental = std::is_fundamental<T>::value
```

Chapter 7

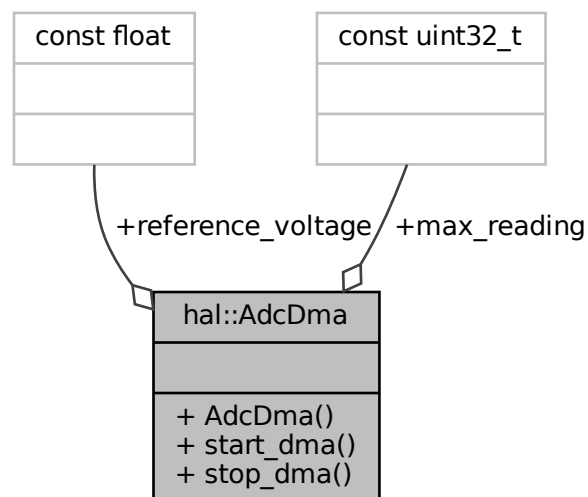
Class Documentation

7.1 hal::AdcDma Class Reference

Class to handle ADC peripheral on STM32 microcontrollers using DMA.

```
#include <adc_dma.hpp>
```

Collaboration diagram for hal::AdcDma:



Classes

- struct [Config](#)

Configuration structure for ADC DMA.

Public Member Functions

- [AdcDma](#) (const [Config](#) &config)
Construct a new [AdcDma](#) object.
- void [start_dma](#) (uint32_t buffer[], uint32_t size)
Enable ADC, start conversion of regular group and transfer result through DMA.
- void [stop_dma](#) ()
Stop ADC conversion of regular group (and injected group in case of auto_injection mode)

Public Attributes

- const uint32_t [max_reading](#)
Maximum ADC reading.
- const float [reference_voltage](#)
Reference voltage for the ADC measurement.

7.1.1 Detailed Description

Class to handle ADC peripheral on STM32 microcontrollers using DMA.

7.1.2 Constructor & Destructor Documentation

7.1.2.1 AdcDma()

```
hal::AdcDma::AdcDma (
    const Config & config ) [explicit]
```

Construct a new [AdcDma](#) object.

Parameters

<i>config</i>	ADC DMA configuration struct
---------------	------------------------------

7.1.3 Member Function Documentation

7.1.3.1 start_dma()

```
void hal::AdcDma::start_dma (
    uint32_t buffer[],
    uint32_t size )
```

Enable ADC, start conversion of regular group and transfer result through DMA.

Parameters

<i>buffer</i>	Destination Buffer address
<i>size</i>	Number of data to be transferred from ADC DMA peripheral to memory

7.1.3.2 stop_dma()

```
void hal::AdcDma::stop_dma ( )
```

Stop ADC conversion of regular group (and injected group in case of auto_injection mode)

7.1.4 Member Data Documentation

7.1.4.1 max_reading

```
const uint32_t hal::AdcDma::max_reading
```

Maximum ADC reading.

7.1.4.2 reference_voltage

```
const float hal::AdcDma::reference_voltage
```

Reference voltage for the ADC measurement.

The documentation for this class was generated from the following files:

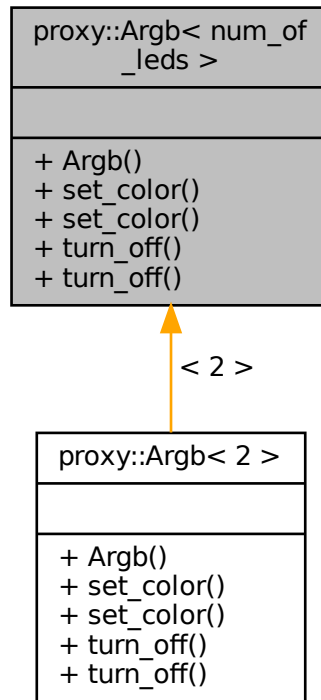
- [inc/hal/adc_dma.hpp](#)
- [src/hal/adc_dma.cpp](#)

7.2 proxy::Argb< num_of_leds > Class Template Reference

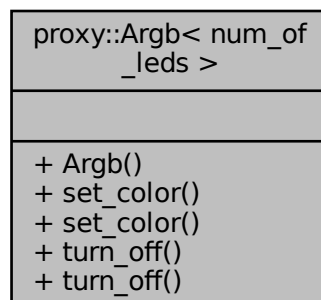
Class for controlling an addressable RGB LED.

```
#include <argb.hpp>
```

Inheritance diagram for proxy::Argb< num_of_leds >:



Collaboration diagram for proxy::Argb< num_of_leds >:



Classes

- struct [Color](#)
Structure for storing color information.
- struct [Config](#)
Configuration structure for the addressable RGB LED.

Public Member Functions

- [Argb](#) (const [Config](#) &config)
Constructor for the [Argb](#) class.
- void [set_color](#) (const [Color](#) &color, uint8_t index)
Set the color of the ARGB at the specified index.
- void [set_color](#) (const [Color](#) &color)
Set the color of all ARGBs.
- void [turn_off](#) (uint8_t index)
Turn off the ARGB at the specified index.
- void [turn_off](#) ()
Turn off all ARGBs.

7.2.1 Detailed Description

```
template<uint8_t num_of_leds>
class proxy::Argb< num_of_leds >
```

Class for controlling an addressable RGB LED.

7.2.2 Constructor & Destructor Documentation

7.2.2.1 Argb()

```
template<uint8_t num_of_leds>
proxy::Argb< num_of_leds >::Argb (
    const Config & config ) [explicit]
```

Constructor for the [Argb](#) class.

Parameters

<i>config</i>	Configuration for the addressable RGB LED
---------------	---

7.2.3 Member Function Documentation

7.2.3.1 set_color() [1/2]

```
template<uint8_t num_of_leds>
void proxy::Argb< num_of_leds >::set_color (
    const Color & color )
```

Set the color of all ARGBs.

Parameters

<i>color</i>	The color to set all ARGBs to
--------------	-------------------------------

7.2.3.2 set_color() [2/2]

```
template<uint8_t num_of_leds>
void proxy::Argb< num_of_leds >::set_color (
    const Color & color,
    uint8_t index )
```

Set the color of the ARGB at the specified index.

Parameters

<i>index</i>	The index of the ARGB to set the color of
<i>color</i>	The color to set the ARGB to

7.2.3.3 turn_off() [1/2]

```
template<uint8_t num_of_leds>
void proxy::Argb< num_of_leds >::turn_off
```

Turn off all ARGBs.

7.2.3.4 turn_off() [2/2]

```
template<uint8_t num_of_leds>
void proxy::Argb< num_of_leds >::turn_off (
    uint8_t index )
```

Turn off the ARGB at the specified index.

Parameters

<i>index</i>	The index of the ARGB to turn off
--------------	-----------------------------------

The documentation for this class was generated from the following files:

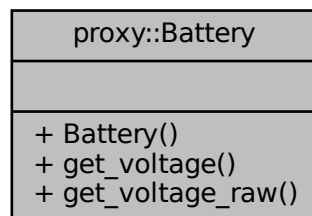
- inc/proxy/[argb.hpp](#)
- src/proxy/[argb.cpp](#)

7.3 proxy::Battery Class Reference

Class for getting the battery voltage.

```
#include <battery.hpp>
```

Collaboration diagram for proxy::Battery:



Classes

- struct [Config](#)
Configuration structure for the battery.

Public Member Functions

- [Battery](#) (const [Config](#) &config)
Constructor for the [Battery](#) class.
- float [get_voltage](#) () const
Get the battery voltage.
- uint32_t [get_voltage_raw](#) () const
Get the raw reading from the battery.

7.3.1 Detailed Description

Class for getting the battery voltage.

7.3.2 Constructor & Destructor Documentation

7.3.2.1 Battery()

```
proxy::Battery::Battery (  
    const Config & config ) [explicit]
```

Constructor for the [Battery](#) class.

Parameters

<i>config</i>	Configuration for the battery
---------------	-------------------------------

7.3.3 Member Function Documentation

7.3.3.1 get_voltage()

```
float proxy::Battery::get_voltage ( ) const
```

Get the battery voltage.

Returns

float [Battery](#) voltage in volts

7.3.3.2 get_voltage_raw()

```
uint32_t proxy::Battery::get_voltage_raw ( ) const
```

Get the raw reading from the battery.

Returns

uint32_t Raw reading from the battery

The documentation for this class was generated from the following files:

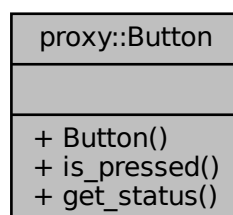
- inc/proxy/[battery.hpp](#)
- src/proxy/[battery.cpp](#)

7.4 proxy::Button Class Reference

Class for controlling a button.

```
#include <button.hpp>
```

Collaboration diagram for proxy::Button:



Classes

- struct [Config](#)
Configuration structure for button.

Public Types

- enum [Status](#) { [NO_PRESS](#) , [SHORT_PRESS](#) , [LONG_PRESS](#) , [EXTRA_LONG_PRESS](#) }
Enum for button status.
- enum [PullResistor](#) { [PULL_UP](#) , [PULL_DOWN](#) }
Enum for button pull resistor.

Public Member Functions

- [Button](#) (const [Config](#) &config)
Constructor for [Button](#) class.
- bool [is_pressed](#) ()
Check if button is pressed.
- [Status](#) [get_status](#) ()
Get button status.

7.4.1 Detailed Description

Class for controlling a button.

7.4.2 Member Enumeration Documentation

7.4.2.1 PullResistor

```
enum proxy::Button::PullResistor
```

Enum for button pull resistor.

Enumerator

PULL_UP	
PULL_DOWN	

7.4.2.2 Status

```
enum proxy::Button::Status
```

Enum for button status.

Enumerator

NO_PRESS	
SHORT_PRESS	
LONG_PRESS	
EXTRA_LONG_PRESS	

7.4.3 Constructor & Destructor Documentation

7.4.3.1 Button()

```
proxy::Button::Button (
    const Config & config ) [explicit]
```

Constructor for [Button](#) class.

Parameters

<i>config</i>	Button configuration
---------------	--------------------------------------

7.4.4 Member Function Documentation

7.4.4.1 get_status()

```
Button::Status proxy::Button::get_status ( )
```

Get button status.

Returns

Status [Button](#) status

7.4.4.2 is_pressed()

```
bool proxy::Button::is_pressed ( )
```

Check if button is pressed.

Returns

bool True if button is pressed, false otherwise

The documentation for this class was generated from the following files:

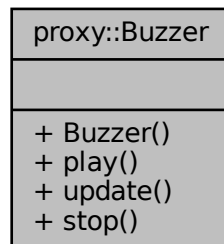
- inc/proxy/[button.hpp](#)
- src/proxy/[button.cpp](#)

7.5 proxy::Buzzer Class Reference

Class for controlling a buzzer.

```
#include <buzzer.hpp>
```

Collaboration diagram for proxy::Buzzer:



Classes

- struct [Config](#)
Configuration structure for the buzzer.

Public Member Functions

- [Buzzer](#) (const [Config](#) &config)
Constructor for the [Buzzer](#) class.
- void [play](#) (uint32_t frequency, uint32_t duration=0)
Play a tone for a duration.
- void [update](#) ()
Update the buzzer state.
- void [stop](#) ()
Stop the buzzer sound.

7.5.1 Detailed Description

Class for controlling a buzzer.

7.5.2 Constructor & Destructor Documentation

7.5.2.1 Buzzer()

```
proxy::Buzzer::Buzzer (  
    const Config & config ) [explicit]
```

Constructor for the [Buzzer](#) class.

Parameters

<i>config</i>	Configuration for the buzzer
---------------	------------------------------

7.5.3 Member Function Documentation

7.5.3.1 play()

```
void proxy::Buzzer::play (
    uint32_t frequency,
    uint32_t duration = 0 )
```

Play a tone for a duration.

Parameters

<i>frequency</i>	Buzzer sound frequency in Hz
<i>duration</i>	Duration of the sound in ms

7.5.3.2 stop()

```
void proxy::Buzzer::stop ( )
```

Stop the buzzer sound.

7.5.3.3 update()

```
void proxy::Buzzer::update ( )
```

Update the buzzer state.

The documentation for this class was generated from the following files:

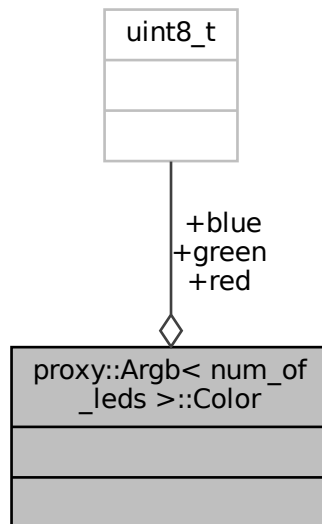
- [inc/proxy/buzzer.hpp](#)
- [src/proxy/buzzer.cpp](#)

7.6 proxy::Argb< num_of_leds >::Color Struct Reference

Structure for storing color information.

```
#include <argb.hpp>
```

Collaboration diagram for proxy::Argb< num_of_leds >::Color:



Public Attributes

- `uint8_t` [red](#)
- `uint8_t` [green](#)
- `uint8_t` [blue](#)

7.6.1 Detailed Description

```
template<uint8_t num_of_leds>
struct proxy::Argb< num_of_leds >::Color
```

Structure for storing color information.

7.6.2 Member Data Documentation

7.6.2.1 blue

```
template<uint8_t num_of_leds>
uint8_t proxy::Argb< num_of_leds >::Color::blue
```

7.6.2.2 green

```
template<uint8_t num_of_leds>
uint8_t proxy::Argb< num_of_leds >::Color::green
```

7.6.2.3 red

```
template<uint8_t num_of_leds>
uint8_t proxy::Argb< num_of_leds >::Color::red
```

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

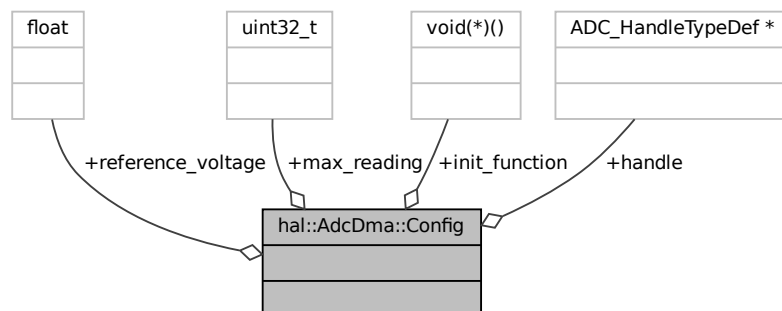
- [inc/proxy/argb.hpp](#)

7.7 hal::AdcDma::Config Struct Reference

Configuration structure for ADC DMA.

```
#include <adc_dma.hpp>
```

Collaboration diagram for hal::AdcDma::Config:



Public Attributes

- ADC_HandleTypeDef * [handle](#)
- void(* [init_function](#))()
- uint32_t [max_reading](#)
- float [reference_voltage](#)

7.7.1 Detailed Description

Configuration structure for ADC DMA.

7.7.2 Member Data Documentation

7.7.2.1 handle

```
ADC_HandleTypeDef* hal::AdcDma::Config::handle
```

7.7.2.2 init_function

```
void(* hal::AdcDma::Config::init_function) ()
```

7.7.2.3 max_reading

```
uint32_t hal::AdcDma::Config::max_reading
```

7.7.2.4 reference_voltage

```
float hal::AdcDma::Config::reference_voltage
```

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

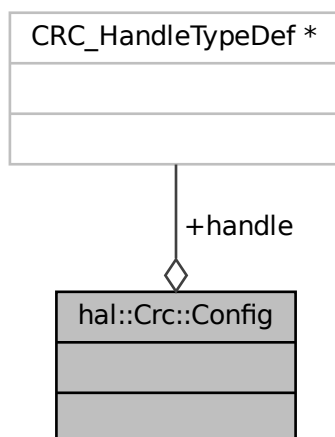
- inc/hal/[adc_dma.hpp](#)

7.8 hal::Crc::Config Struct Reference

CRC configuration struct.

```
#include <crc.hpp>
```

Collaboration diagram for hal::Crc::Config:



Public Attributes

- `CRC_HandleTypeDef *` [handle](#)

7.8.1 Detailed Description

CRC configuration struct.

7.8.2 Member Data Documentation

7.8.2.1 handle

```
CRC_HandleTypeDef* hal::Crc::Config::handle
```

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

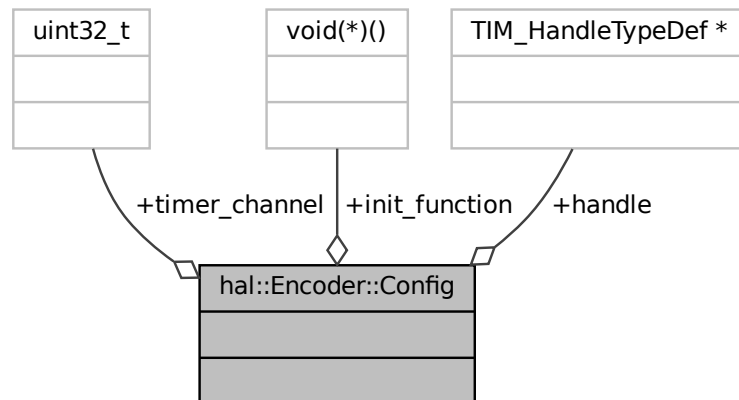
- `inc/hal/crc.hpp`

7.9 hal::Encoder::Config Struct Reference

[Encoder](#) configuration struct.

```
#include <encoder.hpp>
```

Collaboration diagram for hal::Encoder::Config:



Public Attributes

- TIM_HandleTypeDef * [handle](#)
- void(* [init_function](#))()
- uint32_t [timer_channel](#)

7.9.1 Detailed Description

[Encoder](#) configuration struct.

7.9.2 Member Data Documentation

7.9.2.1 handle

```
TIM_HandleTypeDef* hal::Encoder::Config::handle
```

7.9.2.2 init_function

```
void(* hal::Encoder::Config::init_function) ()
```

7.9.2.3 timer_channel

```
uint32_t hal::Encoder::Config::timer_channel
```

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

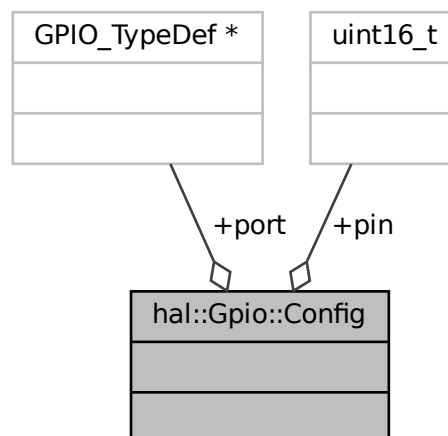
- [inc/hal/encoder.hpp](#)

7.10 hal::Gpio::Config Struct Reference

Configuration structure for GPIO pin.

```
#include <gpio.hpp>
```

Collaboration diagram for hal::Gpio::Config:



Public Attributes

- `GPIO_TypeDef *` [port](#)
- `uint16_t` [pin](#)

7.10.1 Detailed Description

Configuration structure for GPIO pin.

7.10.2 Member Data Documentation

7.10.2.1 pin

```
uint16_t hal::Gpio::Config::pin
```

7.10.2.2 port

```
GPIO_TypeDef* hal::Gpio::Config::port
```

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

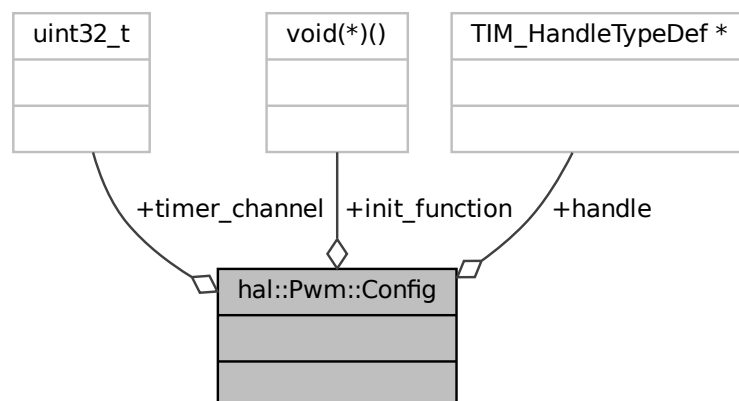
- [inc/hal/gpio.hpp](#)

7.11 hal::Pwm::Config Struct Reference

PWM configuration struct.

```
#include <pwm.hpp>
```

Collaboration diagram for hal::Pwm::Config:



Public Attributes

- TIM_HandleTypeDef * [handle](#)
- void(* [init_function](#))()
- uint32_t [timer_channel](#)

7.11.1 Detailed Description

PWM configuration struct.

7.11.2 Member Data Documentation

7.11.2.1 handle

```
TIM_HandleTypeDef* hal::Pwm::Config::handle
```

7.11.2.2 init_function

```
void(* hal::Pwm::Config::init_function) ()
```

7.11.2.3 timer_channel

```
uint32_t hal::Pwm::Config::timer_channel
```

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

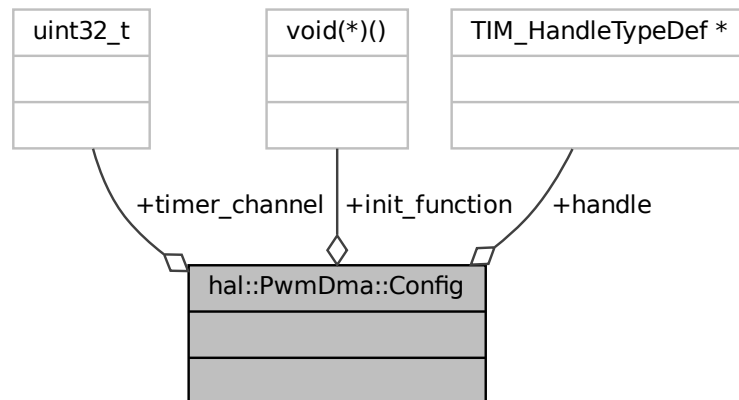
- [inc/hal/pwm.hpp](#)

7.12 hal::PwmDma::Config Struct Reference

PWM configuration struct.

```
#include <pwm_dma.hpp>
```

Collaboration diagram for hal::PwmDma::Config:



Public Attributes

- TIM_HandleTypeDef * [handle](#)
- void(* [init_function](#))()
- uint32_t [timer_channel](#)

7.12.1 Detailed Description

PWM configuration struct.

7.12.2 Member Data Documentation

7.12.2.1 handle

```
TIM_HandleTypeDef* hal::PwmDma::Config::handle
```


7.12.2.2 init_function

```
void(* hal::PwmDma::Config::init_function) ()
```

7.12.2.3 timer_channel

```
uint32_t hal::PwmDma::Config::timer_channel
```

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

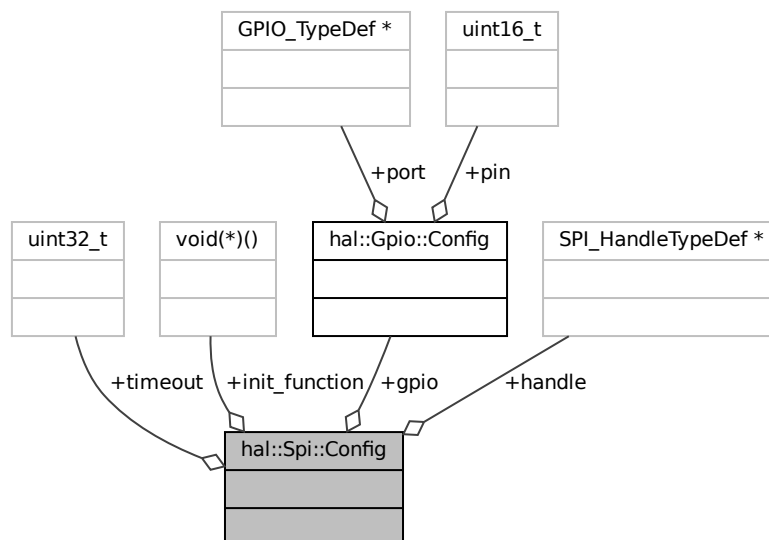
- [inc/hal/pwm_dma.hpp](#)

7.13 hal::Spi::Config Struct Reference

SPI configuration struct.

```
#include <spi.hpp>
```

Collaboration diagram for hal::Spi::Config:



Public Attributes

- SPI_HandleTypeDef * [handle](#)
- void(* [init_function](#))()
- [hal::Gpio::Config](#) [gpio](#)
- uint32_t [timeout](#)

7.13.1 Detailed Description

SPI configuration struct.

7.13.2 Member Data Documentation

7.13.2.1 gpio

```
hal::Gpio::Config hal::Spi::Config::gpio
```

7.13.2.2 handle

```
SPI_HandleTypeDef* hal::Spi::Config::handle
```

7.13.2.3 init_function

```
void(* hal::Spi::Config::init_function) ()
```

7.13.2.4 timeout

```
uint32_t hal::Spi::Config::timeout
```

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

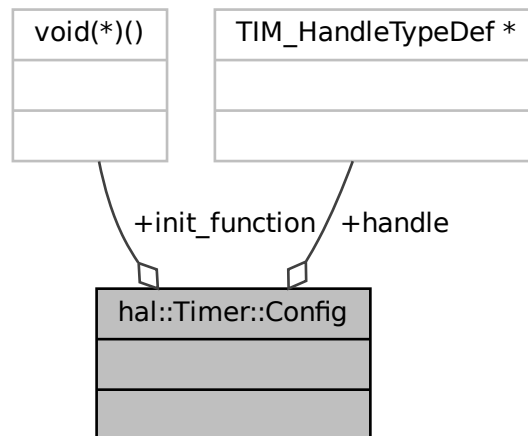
- [inc/hal/spi.hpp](#)

7.14 hal::Timer::Config Struct Reference

Timer configuration struct.

```
#include <timer.hpp>
```

Collaboration diagram for hal::Timer::Config:



Public Attributes

- TIM_HandleTypeDef * [handle](#)
- void(* [init_function](#))()

7.14.1 Detailed Description

Timer configuration struct.

7.14.2 Member Data Documentation

7.14.2.1 handle

```
TIM_HandleTypeDef* hal::Timer::Config::handle
```

7.14.2.2 init_function

```
void(* hal::Timer::Config::init_function) ()
```

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

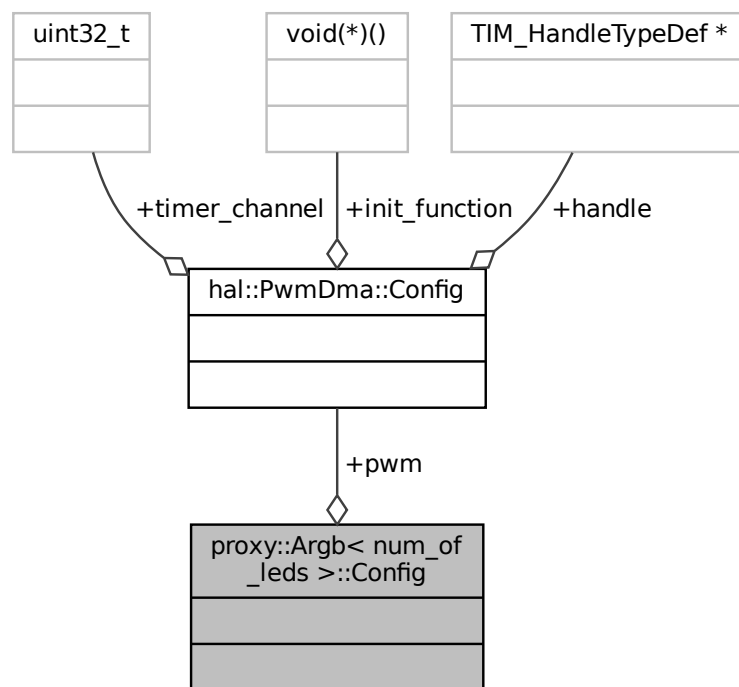
- [inc/hal/timer.hpp](#)

7.15 proxy::Argb< num_of_leds >::Config Struct Reference

Configuration structure for the addressable RGB LED.

```
#include <argb.hpp>
```

Collaboration diagram for proxy::Argb< num_of_leds >::Config:



Public Attributes

- [hal::PwmDma::Config pwm](#)

7.15.1 Detailed Description

```
template<uint8_t num_of_leds>
struct proxy::Argb< num_of_leds >::Config
```

Configuration structure for the addressable RGB LED.

7.15.2 Member Data Documentation

7.15.2.1 pwm

```
template<uint8_t num_of_leds>
hal::PwmDma::Config proxy::Argb< num_of_leds >::Config::pwm
```

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

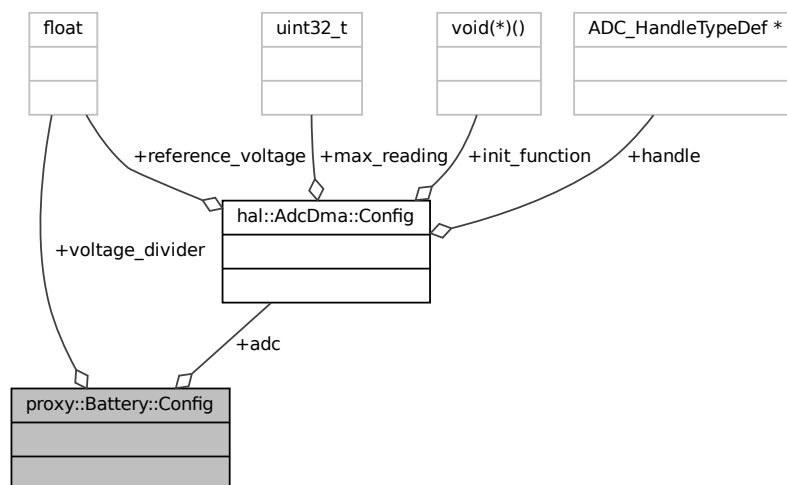
- [inc/proxy/argb.hpp](#)

7.16 proxy::Battery::Config Struct Reference

Configuration structure for the battery.

```
#include <battery.hpp>
```

Collaboration diagram for proxy::Battery::Config:



Public Attributes

- [hal::AdcDma::Config](#) `adc`
- float `voltage_divider`

7.16.1 Detailed Description

Configuration structure for the battery.

7.16.2 Member Data Documentation

7.16.2.1 `adc`

```
hal::AdcDma::Config proxy::Battery::Config::adc
```

7.16.2.2 `voltage_divider`

```
float proxy::Battery::Config::voltage_divider
```

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

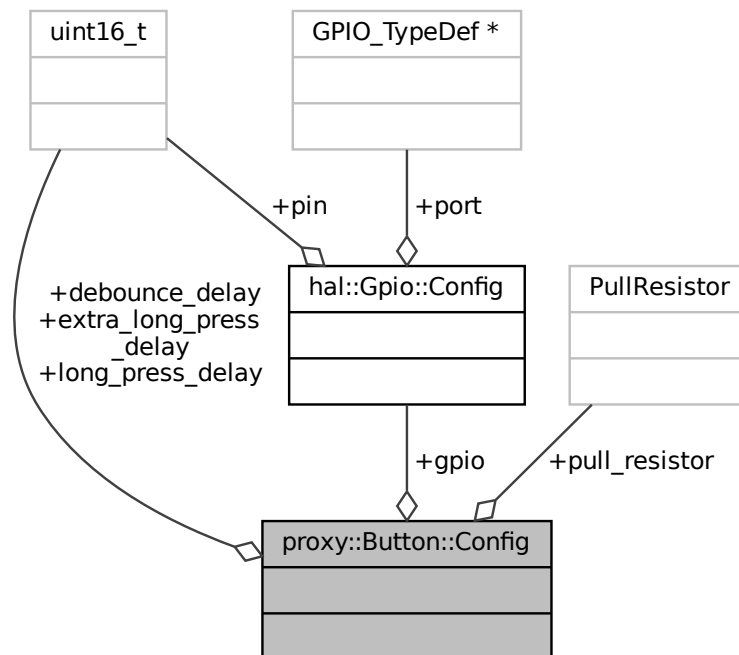
- inc/proxy/[battery.hpp](#)

7.17 proxy::Button::Config Struct Reference

Configuration structure for button.

```
#include <button.hpp>
```

Collaboration diagram for proxy::Button::Config:



Public Attributes

- [hal::Gpio::Config gpio](#) { }
- [PullResistor pull_resistor](#) { }
- [uint16_t debounce_delay](#) = 10
- [uint16_t long_press_delay](#) = 1000
- [uint16_t extra_long_press_delay](#) = 5000

7.17.1 Detailed Description

Configuration structure for button.

7.17.2 Member Data Documentation

7.17.2.1 debounce_delay

```
uint16_t proxy::Button::Config::debounce_delay = 10
```

7.17.2.2 extra_long_press_delay

```
uint16_t proxy::Button::Config::extra_long_press_delay = 5000
```

7.17.2.3 gpio

```
hal::Gpio::Config proxy::Button::Config::gpio { }
```

7.17.2.4 long_press_delay

```
uint16_t proxy::Button::Config::long_press_delay = 1000
```

7.17.2.5 pull_resistor

```
PullResistor proxy::Button::Config::pull_resistor { }
```

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

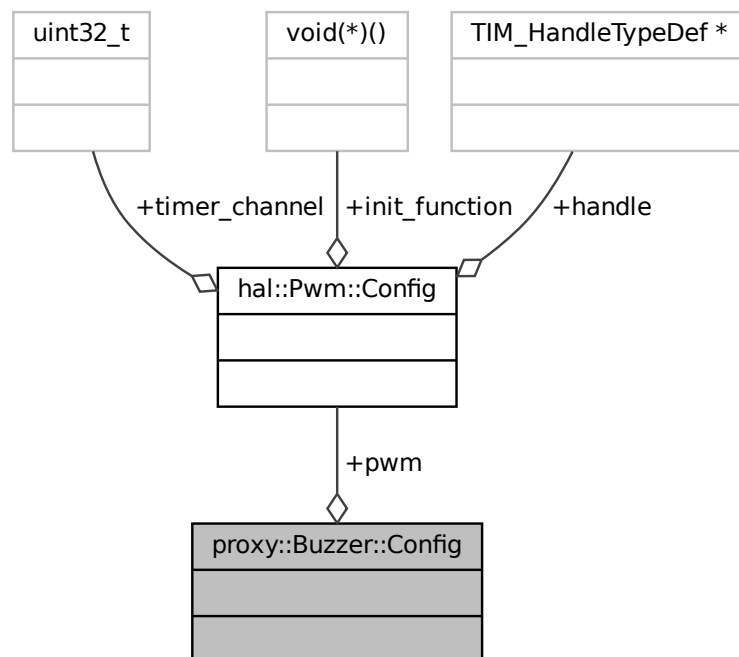
- [inc/proxy/button.hpp](#)

7.18 proxy::Buzzer::Config Struct Reference

Configuration structure for the buzzer.

```
#include <buzzer.hpp>
```


Collaboration diagram for proxy::Buzzer::Config:



Public Attributes

- [hal::Pwm::Config pwm](#)

7.18.1 Detailed Description

Configuration structure for the buzzer.

7.18.2 Member Data Documentation

7.18.2.1 pwm

[hal::Pwm::Config](#) proxy::Buzzer::Config::pwm

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

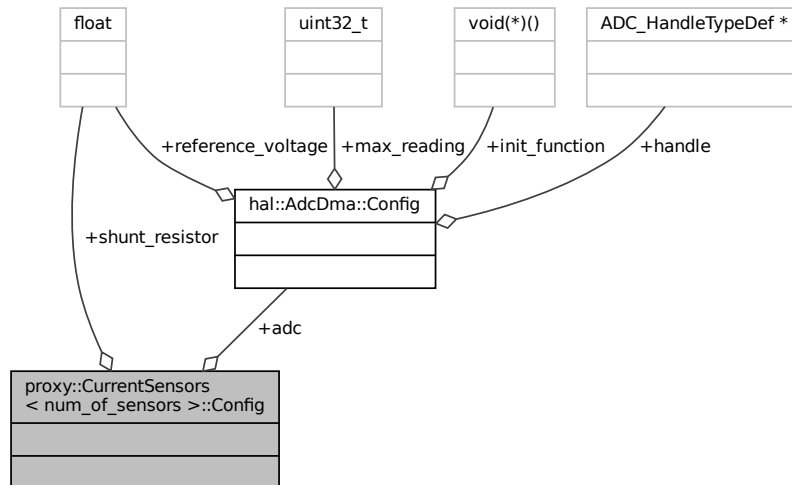
- [inc/proxy/buzzer.hpp](#)

7.19 proxy::CurrentSensors< num_of_sensors >::Config Struct Reference

Configuration structure for current sensors.

```
#include <current_sensors.hpp>
```

Collaboration diagram for proxy::CurrentSensors< num_of_sensors >::Config:



Public Attributes

- [hal::AdcDma::Config](#) `adc`
- float `shunt_resistor`

7.19.1 Detailed Description

```
template<uint8_t num_of_sensors>
struct proxy::CurrentSensors< num_of_sensors >::Config
```

Configuration structure for current sensors.

7.19.2 Member Data Documentation

7.19.2.1 adc

```
template<uint8_t num_of_sensors>
hal::AdcDma::Config proxy::CurrentSensors< num_of_sensors >::Config::adc
```

7.19.2.2 shunt_resistor

```
template<uint8_t num_of_sensors>
float proxy::CurrentSensors< num_of_sensors >::Config::shunt_resistor
```

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

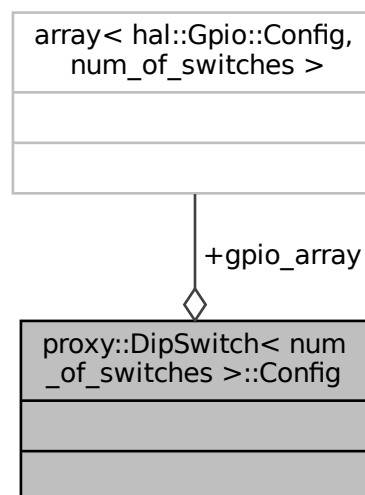
- inc/proxy/current_sensors.hpp

7.20 proxy::DipSwitch< num_of_switches >::Config Struct Reference

Configuration struct for [DipSwitch](#).

```
#include <dip_switch.hpp>
```

Collaboration diagram for proxy::DipSwitch< num_of_switches >::Config:



Public Attributes

- std::array< [hal::Gpio::Config](#), num_of_switches > [gpio_array](#)

7.20.1 Detailed Description

```
template<uint8_t num_of_switches>
struct proxy::DipSwitch< num_of_switches >::Config
```

Configuration struct for [DipSwitch](#).

7.20.2 Member Data Documentation

7.20.2.1 gpio_array

```
template<uint8_t num_of_switches>
std::array<hal::Gpio::Config, num_of_switches> proxy::DipSwitch< num_of_switches >::Config<
::gpio_array
```

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

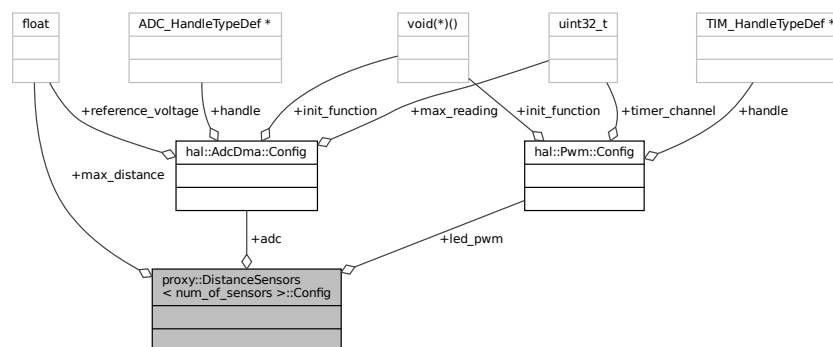
- [inc/proxy/dip_switch.hpp](#)

7.21 proxy::DistanceSensors< num_of_sensors >::Config Struct Reference

Configuration structure for distance sensors.

```
#include <distance_sensors.hpp>
```

Collaboration diagram for proxy::DistanceSensors< num_of_sensors >::Config:



Public Attributes

- [hal::AdcDma::Config](#) adc
- [hal::Pwm::Config](#) led_pwm
- float [max_distance](#)

7.21.1 Detailed Description

```
template<uint8_t num_of_sensors>
struct proxy::DistanceSensors< num_of_sensors >::Config
```

Configuration structure for distance sensors.

7.21.2 Member Data Documentation

7.21.2.1 adc

```
template<uint8_t num_of_sensors>
hal::AdcDma::Config proxy::DistanceSensors< num_of_sensors >::Config::adc
```

7.21.2.2 led_pwm

```
template<uint8_t num_of_sensors>
hal::Pwm::Config proxy::DistanceSensors< num_of_sensors >::Config::led_pwm
```

7.21.2.3 max_distance

```
template<uint8_t num_of_sensors>
float proxy::DistanceSensors< num_of_sensors >::Config::max_distance
```

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

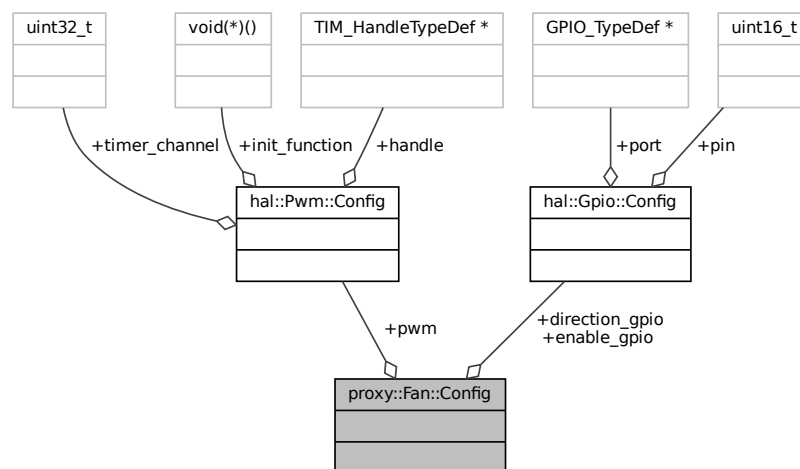
- [inc/proxy/distance_sensors.hpp](#)

7.22 proxy::Fan::Config Struct Reference

Configuration structure for the fan.

```
#include <fan.hpp>
```

Collaboration diagram for proxy::Fan::Config:



Public Attributes

- [hal::Pwm::Config](#) pwm
- [hal::Gpio::Config](#) direction_gpio
- [hal::Gpio::Config](#) enable_gpio

7.22.1 Detailed Description

Configuration structure for the fan.

7.22.2 Member Data Documentation

7.22.2.1 direction_gpio

[hal::Gpio::Config](#) proxy::Fan::Config::direction_gpio

7.22.2.2 enable_gpio

[hal::Gpio::Config](#) proxy::Fan::Config::enable_gpio

7.22.2.3 pwm

[hal::Pwm::Config](#) proxy::Fan::Config::pwm

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

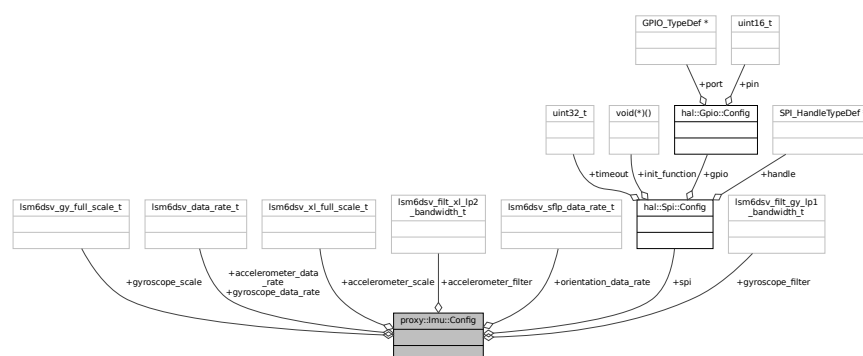
- [inc/proxy/fan.hpp](#)

7.23 proxy::Imu::Config Struct Reference

IMU configuration struct.

```
#include <imu.hpp>
```

Collaboration diagram for proxy::Imu::Config:



Public Attributes

- [hal::Spi::Config spi](#)
- lsm6dsv_data_rate_t [gyroscope_data_rate](#)
- lsm6dsv_data_rate_t [accelerometer_data_rate](#)
- lsm6dsv_sflp_data_rate_t [orientation_data_rate](#)
- lsm6dsv_gy_full_scale_t [gyroscope_scale](#)
- lsm6dsv_xl_full_scale_t [accelerometer_scale](#)
- lsm6dsv_filt_gy_lp1_bandwidth_t [gyroscope_filter](#)
- lsm6dsv_filt_xl_lp2_bandwidth_t [accelerometer_filter](#)

7.23.1 Detailed Description

IMU configuration struct.

7.23.2 Member Data Documentation

7.23.2.1 accelerometer_data_rate

lsm6dsv_data_rate_t proxy::Imu::Config::accelerometer_data_rate

7.23.2.2 accelerometer_filter

lsm6dsv_filt_xl_lp2_bandwidth_t proxy::Imu::Config::accelerometer_filter

7.23.2.3 accelerometer_scale

lsm6dsv_xl_full_scale_t proxy::Imu::Config::accelerometer_scale

7.23.2.4 gyroscope_data_rate

lsm6dsv_data_rate_t proxy::Imu::Config::gyroscope_data_rate

7.23.2.5 gyroscope_filter

```
lsm6dsv_filt_gy_lpl_bandwidth_t proxy::Imu::Config::gyroscope_filter
```

7.23.2.6 gyroscope_scale

```
lsm6dsv_gy_full_scale_t proxy::Imu::Config::gyroscope_scale
```

7.23.2.7 orientation_data_rate

```
lsm6dsv_sflp_data_rate_t proxy::Imu::Config::orientation_data_rate
```

7.23.2.8 spi

```
hal::Spi::Config proxy::Imu::Config::spi
```

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

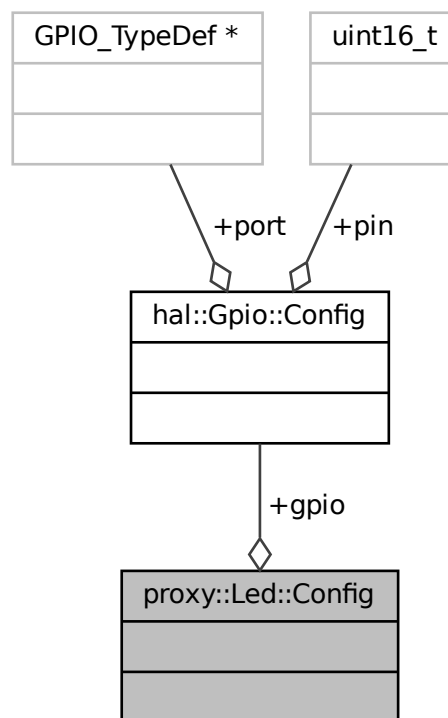
- [inc/proxy/imu.hpp](#)

7.24 proxy::Led::Config Struct Reference

Configuration structure for LED.

```
#include <led.hpp>
```


Collaboration diagram for proxy::Led::Config:



Public Attributes

- [hal::Gpio::Config gpio](#)

7.24.1 Detailed Description

Configuration structure for LED.

7.24.2 Member Data Documentation

7.24.2.1 gpio

[hal::Gpio::Config](#) `proxy::Led::Config::gpio`

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

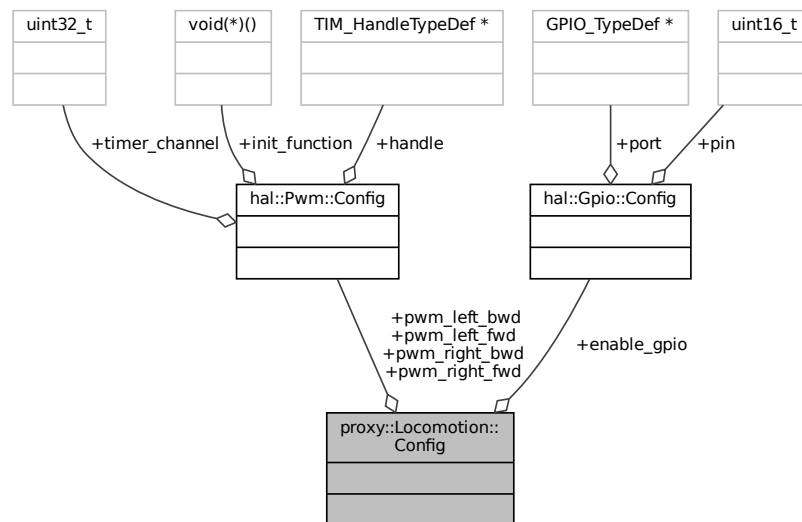
- `inc/proxy/led.hpp`

7.25 proxy::Locomotion::Config Struct Reference

Configuration structure for the locomotion.

```
#include <locomotion.hpp>
```

Collaboration diagram for proxy::Locomotion::Config:



Public Attributes

- [hal::Pwm::Config pwm_left_fwd](#)
- [hal::Pwm::Config pwm_left_bwd](#)
- [hal::Pwm::Config pwm_right_fwd](#)
- [hal::Pwm::Config pwm_right_bwd](#)
- [hal::Gpio::Config enable_gpio](#)

7.25.1 Detailed Description

Configuration structure for the locomotion.

7.25.2 Member Data Documentation

7.25.2.1 enable_gpio

```
hal::Gpio::Config proxy::Locomotion::Config::enable_gpio
```

```
hal::Pwm::Config proxy::Locomotion::Config::pwm_left_bwd
```

```
hal::Pwm::Config proxy::Locomotion::Config::pwm_left_fwd
```

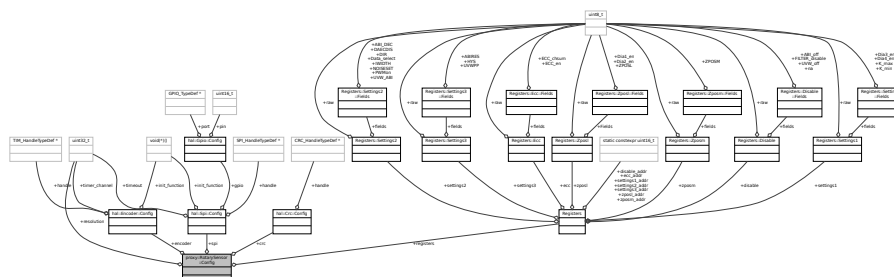
```
hal::Pwm::Config proxy::Locomotion::Config::pwm_right_bwd
```

```
hal::Pwm::Config proxy::Locomotion::Config::pwm_right_fwd
```

- inc/proxy/locomotion.hpp

Rotary sensor configuration struct.

Collaboration diagram for proxy::RotarySensor::Config:



Public Attributes

- [hal::Spi::Config](#) spi
- [hal::Encoder::Config](#) encoder
- [hal::Crc::Config](#) crc
- [uint32_t](#) resolution
- [Registers](#) registers

7.26.1 Detailed Description

Rotary sensor configuration struct.

7.26.2 Member Data Documentation

7.26.2.1 crc

[hal::Crc::Config](#) proxy::RotarySensor::Config::crc

7.26.2.2 encoder

[hal::Encoder::Config](#) proxy::RotarySensor::Config::encoder

7.26.2.3 registers

[Registers](#) proxy::RotarySensor::Config::registers

7.26.2.4 resolution

[uint32_t](#) proxy::RotarySensor::Config::resolution

7.26.2.5 spi

[hal::Spi::Config](#) proxy::RotarySensor::Config::spi

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

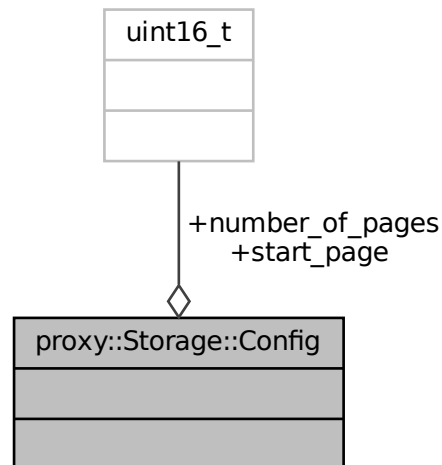
- [inc/proxy/rotary_sensor.hpp](#)

7.27 proxy::Storage::Config Struct Reference

Configuration structure for the storage.

```
#include <storage.hpp>
```

Collaboration diagram for proxy::Storage::Config:



Public Attributes

- `uint16_t` [start_page](#)
- `uint16_t` [number_of_pages](#)

7.27.1 Detailed Description

Configuration structure for the storage.

7.27.2 Member Data Documentation

7.27.2.1 number_of_pages

```
uint16_t proxy::Storage::Config::number_of_pages
```

7.27.2.2 start_page

```
uint16_t proxy::Storage::Config::start_page
```

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

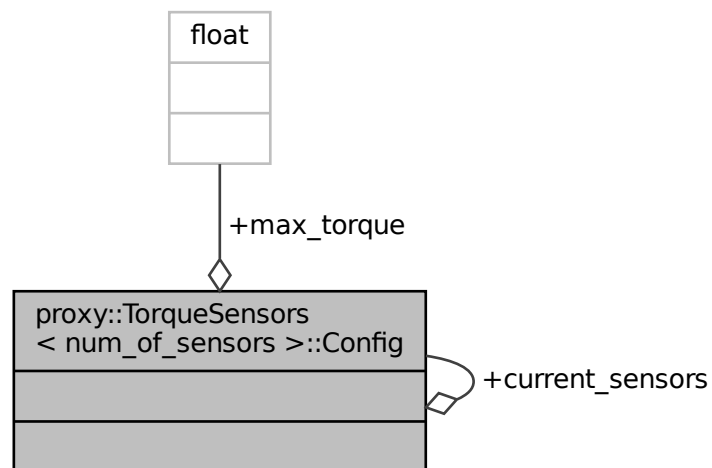
- [inc/proxy/storage.hpp](#)

7.28 proxy::TorqueSensors< num_of_sensors >::Config Struct Reference

Configuration structure for torque sensors.

```
#include <torque_sensors.hpp>
```

Collaboration diagram for proxy::TorqueSensors< num_of_sensors >::Config:



Public Attributes

- [CurrentSensors< num_of_sensors >::Config current_sensors](#)
- float [max_torque](#)

7.28.1 Detailed Description

```
template<uint8_t num_of_sensors>
struct proxy::TorqueSensors< num_of_sensors >::Config
```

Configuration structure for torque sensors.

7.28.2 Member Data Documentation

7.28.2.1 current_sensors

```
template<uint8_t num_of_sensors>
CurrentSensors<num_of_sensors>::Config proxy::TorqueSensors< num_of_sensors >::Config::current←
_sensors
```

7.28.2.2 max_torque

```
template<uint8_t num_of_sensors>
float proxy::TorqueSensors< num_of_sensors >::Config::max_torque
```

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

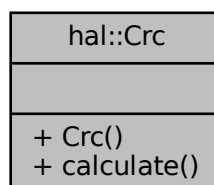
- inc/proxy/torque_sensors.hpp

7.29 hal::Crc Class Reference

Class to handle the cyclic redundancy check peripheral on STM32 microcontrollers.

```
#include <crc.hpp>
```

Collaboration diagram for hal::Crc:



Classes

- struct [Config](#)
CRC configuration struct.

Public Member Functions

- [Crc](#) (const [Config](#) &config)
Construct a new [Crc](#) object.
- uint32_t [calculate](#) (uint32_t data[], uint32_t size)
Calculate the CRC value.

7.29.1 Detailed Description

Class to handle the cyclic redundancy check peripheral on STM32 microcontrollers.

7.29.2 Constructor & Destructor Documentation

7.29.2.1 Crc()

```
hal::Crc::Crc (
    const Config & config ) [explicit]
```

Construct a new [Crc](#) object.

Parameters

<i>config</i>	Configuration for the CRC
---------------	---------------------------

7.29.3 Member Function Documentation

7.29.3.1 calculate()

```
uint32_t hal::Crc::calculate (
    uint32_t data[],
    uint32_t size )
```

Calculate the CRC value.

Parameters

<i>data</i>	Data to calculate the CRC
<i>size</i>	Size of the buffer

Returns

uint32_t CRC value

The documentation for this class was generated from the following files:

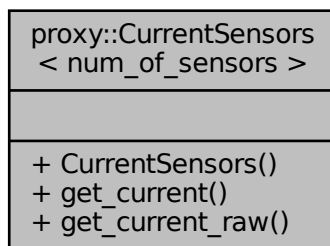
- inc/hal/crc.hpp
- src/hal/crc.cpp

7.30 proxy::CurrentSensors< num_of_sensors > Class Template Reference

Class for controlling [CurrentSensors](#).

```
#include <current_sensors.hpp>
```

Collaboration diagram for proxy::CurrentSensors< num_of_sensors >:

**Classes**

- struct [Config](#)
Configuration structure for current sensors.

Public Member Functions

- [CurrentSensors](#) (const [Config](#) &config)
Constructor for the [CurrentSensors](#) class.
- float [get_current](#) (uint8_t sensor_index) const
Get the current from the sensor.
- uint32_t [get_current_raw](#) (uint8_t sensor_index) const
Get the raw reading from the current sensor.

7.30.1 Detailed Description

```
template<uint8_t num_of_sensors>
class proxy::CurrentSensors< num_of_sensors >
```

Class for controlling [CurrentSensors](#).

7.30.2 Constructor & Destructor Documentation

7.30.2.1 CurrentSensors()

```
template<uint8_t num_of_sensors>
proxy::CurrentSensors< num_of_sensors >::CurrentSensors (
    const Config & config ) [explicit]
```

Constructor for the [CurrentSensors](#) class.

Parameters

<i>config</i>	Configuration for the current sensors
---------------	---------------------------------------

7.30.3 Member Function Documentation

7.30.3.1 get_current()

```
template<uint8_t num_of_sensors>
float proxy::CurrentSensors< num_of_sensors >::get_current (
    uint8_t sensor_index ) const
```

Get the current from the sensor.

Parameters

<i>sensor_index</i>	Index of the sensor
---------------------	---------------------

Returns

float Current reading from the sensor in amps

7.30.3.2 get_current_raw()

```
template<uint8_t num_of_sensors>
uint32_t proxy::CurrentSensors< num_of_sensors >::get_current_raw (
    uint8_t sensor_index ) const
```

Get the raw reading from the current sensor.

Parameters

<i>sensor_index</i>	Index of the sensor
---------------------	---------------------

Returns

uint16_t Current reading from the sensor

The documentation for this class was generated from the following files:

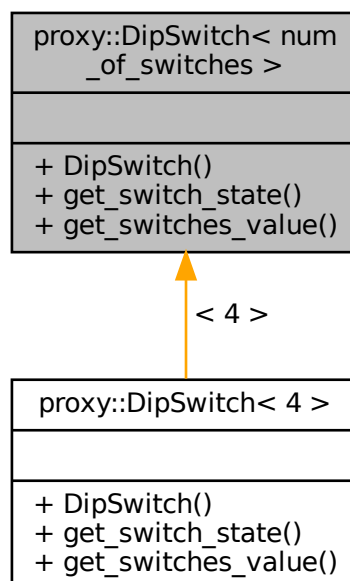
- [inc/proxy/current_sensors.hpp](#)
- [src/proxy/current_sensors.cpp](#)

7.31 proxy::DipSwitch< num_of_switches > Class Template Reference

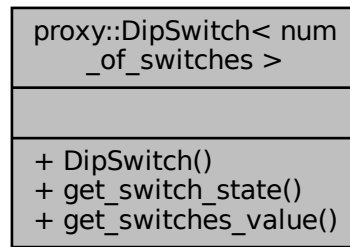
Class for controlling a dip switch.

```
#include <dip_switch.hpp>
```

Inheritance diagram for proxy::DipSwitch< num_of_switches >:



Collaboration diagram for proxy::DipSwitch< num_of_switches >:



Classes

- struct [Config](#)
Configuration struct for [DipSwitch](#).

Public Member Functions

- [DipSwitch](#) (const [Config](#) &config)
Construct a new Dip Switch object.
- bool [get_switch_state](#) (uint8_t switch_index) const
Get the state of a switch.
- uint8_t [get_switches_value](#) () const
Get the value of all switches.

7.31.1 Detailed Description

```
template<uint8_t num_of_switches>
class proxy::DipSwitch< num_of_switches >
```

Class for controlling a dip switch.

7.31.2 Constructor & Destructor Documentation

7.31.2.1 DipSwitch()

```
template<uint8_t num_of_sensors>
proxy::DipSwitch< num_of_sensors >::DipSwitch (
    const Config & config ) [explicit]
```

Construct a new Dip Switch object.

Parameters

<i>config</i>	Configuration struct for DipSwitch
---------------	--

7.31.3 Member Function Documentation

7.31.3.1 get_switch_state()

```
template<uint8_t num_of_sensors>
bool proxy::DipSwitch< num_of_sensors >::get_switch_state (
    uint8_t switch_index ) const
```

Get the state of a switch.

Parameters

<i>switch_index</i>	Index of the switch
---------------------	---------------------

Returns

bool True if the switch is on, false otherwise

7.31.3.2 get_switches_value()

```
template<uint8_t num_of_sensors>
uint8_t proxy::DipSwitch< num_of_sensors >::get_switches_value
```

Get the value of all switches.

Returns

uint8_t Value of all switches

The documentation for this class was generated from the following files:

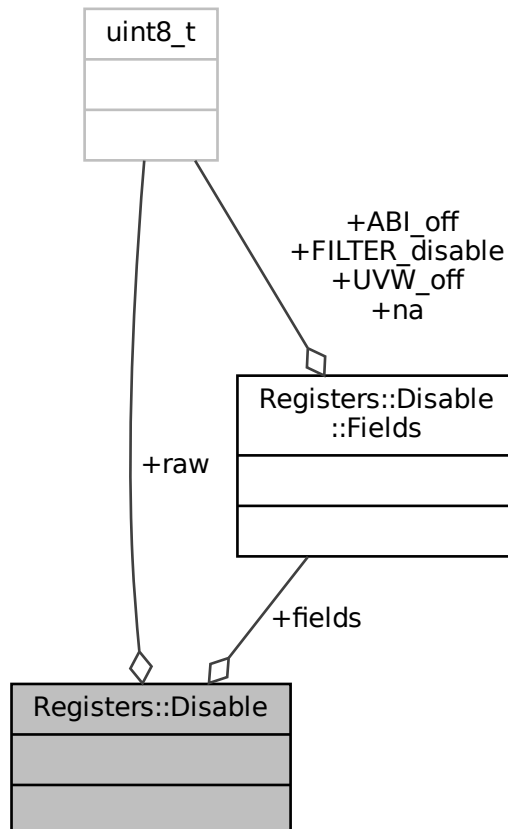
- [inc/proxy/dip_switch.hpp](#)
- [src/proxy/dip_switch.cpp](#)

7.32 Registers::Disable Union Reference

[Registers](#) union types definition.

```
#include <rotary_sensor_reg.hpp>
```

Collaboration diagram for Registers::Disable:



Classes

- struct [Fields](#)

Public Attributes

- [Fields](#) `fields`
- `uint8_t` `raw`

7.32.1 Detailed Description

[Registers](#) union types definition.

7.32.2 Member Data Documentation

7.32.2.1 fields

`Fields` Registers::Disable::fields

7.32.2.2 raw

`uint8_t` Registers::Disable::raw

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

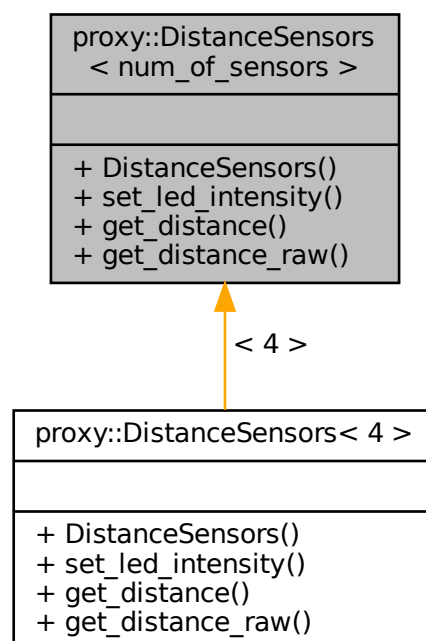
- `inc/proxy/rotary_sensor_reg.hpp`

7.33 proxy::DistanceSensors< num_of_sensors > Class Template Reference

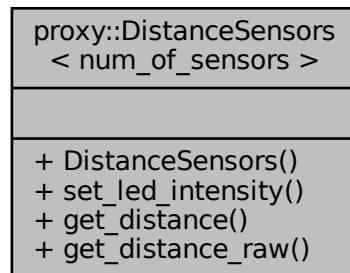
Class for controlling [DistanceSensors](#).

```
#include <distance_sensors.hpp>
```

Inheritance diagram for proxy::DistanceSensors< num_of_sensors >:



Collaboration diagram for proxy::DistanceSensors< num_of_sensors >:



Classes

- struct [Config](#)
Configuration structure for distance sensors.

Public Member Functions

- [DistanceSensors](#) (const [Config](#) &config)
Constructor for the [DistanceSensors](#) class.
- void [set_led_intensity](#) (float intensity)
Set the distance sensors led intensity.
- float [get_distance](#) (uint8_t sensor_index) const
Get the distance from a sensor.
- uint32_t [get_distance_raw](#) (uint8_t sensor_index) const
Get the distance from a sensor.

7.33.1 Detailed Description

```
template<uint8_t num_of_sensors>
class proxy::DistanceSensors< num_of_sensors >
```

Class for controlling [DistanceSensors](#).

7.33.2 Constructor & Destructor Documentation

7.33.2.1 DistanceSensors()

```
template<uint8_t num_of_sensors>
proxy::DistanceSensors< num_of_sensors >::DistanceSensors (
    const Config & config ) [explicit]
```

Constructor for the [DistanceSensors](#) class.

Parameters

<i>config</i>	Configuration for the distance sensors
---------------	--

7.33.3 Member Function Documentation

7.33.3.1 get_distance()

```
template<uint8_t num_of_sensors>
float proxy::DistanceSensors< num_of_sensors >::get_distance (
    uint8_t sensor_index ) const
```

Get the distance from a sensor.

Parameters

<i>sensor_index</i>	Index of the sensor
---------------------	---------------------

Returns

float Distance reading from the sensors

7.33.3.2 get_distance_raw()

```
template<uint8_t num_of_sensors>
uint32_t proxy::DistanceSensors< num_of_sensors >::get_distance_raw (
    uint8_t sensor_index ) const
```

Get the distance from a sensor.

Parameters

<i>sensor_index</i>	Index of the sensor
---------------------	---------------------

Returns

uint32_t Raw reading from the distance sensor

7.33.3.3 set_led_intensity()

```
template<uint8_t num_of_sensors>
void proxy::DistanceSensors< num_of_sensors >::set_led_intensity (
    float intensity )
```

Set the distance sensors led intensity.

Parameters

<i>intensity</i>	Intensity percentage of the infrared LED
------------------	--

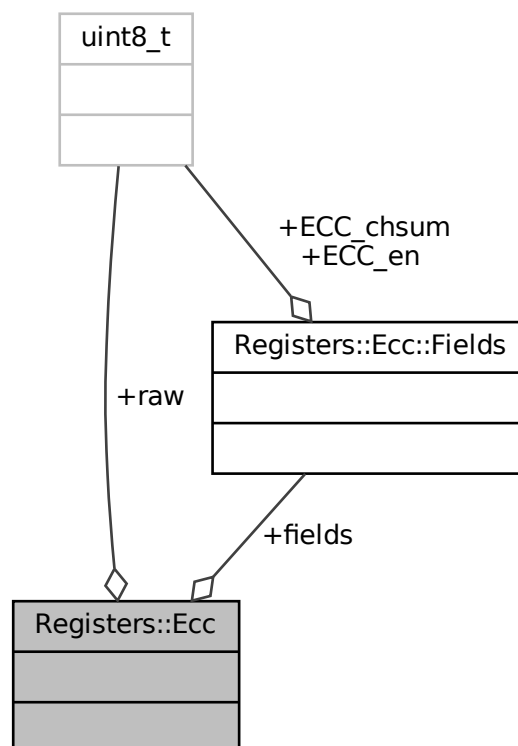
The documentation for this class was generated from the following files:

- [inc/proxy/distance_sensors.hpp](#)
- [src/proxy/distance_sensors.cpp](#)

7.34 Registers::Ecc Union Reference

```
#include <rotary_sensor_reg.hpp>
```

Collaboration diagram for Registers::Ecc:



Classes

- struct [Fields](#)

Public Attributes

- [Fields](#) [fields](#)
- [uint8_t](#) [raw](#)

7.34.1 Member Data Documentation

7.34.1.1 fields

[Fields](#) `Registers::Ecc::fields`

7.34.1.2 raw

`uint8_t` `Registers::Ecc::raw`

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

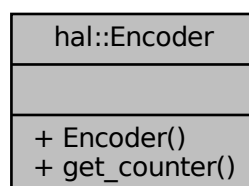
- `inc/proxy/rotary_sensor_reg.hpp`

7.35 hal::Encoder Class Reference

Class to handle encoder peripheral on STM32 microcontrollers.

```
#include <encoder.hpp>
```

Collaboration diagram for `hal::Encoder`:



Classes

- struct [Config](#)
Encoder configuration struct.

Public Member Functions

- [Encoder](#) (const [Config](#) &config)
Construct a new [Encoder](#) object.
- int32_t [get_counter](#) () const
Get the counter value.

7.35.1 Detailed Description

Class to handle encoder peripheral on STM32 microcontrollers.

7.35.2 Constructor & Destructor Documentation

7.35.2.1 Encoder()

```
hal::Encoder::Encoder (
    const Config & config ) [explicit]
```

Construct a new [Encoder](#) object.

Parameters

<i>config</i>	Configuration for the encoder
---------------	-------------------------------

7.35.3 Member Function Documentation

7.35.3.1 get_counter()

```
int32_t hal::Encoder::get_counter ( ) const
```

Get the counter value.

Returns

int32_t Current value of the counter

The documentation for this class was generated from the following files:

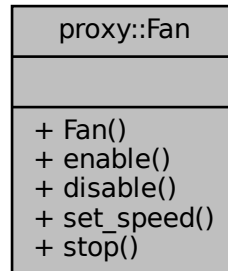
- inc/hal/[encoder.hpp](#)
- src/hal/[encoder.cpp](#)

7.36 proxy::Fan Class Reference

Class for controlling the fan driver.

```
#include <fan.hpp>
```

Collaboration diagram for proxy::Fan:



Classes

- struct [Config](#)
Configuration structure for the fan.

Public Types

- enum [RotationDirection](#) { [FORWARD](#) , [BACKWARD](#) }
Enum for rotation direction.

Public Member Functions

- [Fan](#) (const [Config](#) &config)
Construct a new fan object.
- void [enable](#) ()
Enable the fan.
- void [disable](#) ()
Disable the fan.
- void [set_speed](#) (float speed)
Set the speed of the fans.
- void [stop](#) ()
Stop the fan.

7.36.1 Detailed Description

Class for controlling the fan driver.

7.36.2 Member Enumeration Documentation

7.36.2.1 RotationDirection

```
enum proxy::Fan::RotationDirection
```

Enum for rotation direction.

Enumerator

FORWARD	
BACKWARD	

7.36.3 Constructor & Destructor Documentation

7.36.3.1 Fan()

```
proxy::Fan::Fan (  
    const Config & config ) [explicit]
```

Construct a new fan object.

Parameters

<i>config</i>	Configuration for the fan driver
---------------	----------------------------------

7.36.4 Member Function Documentation

7.36.4.1 disable()

```
void proxy::Fan::disable ( )
```

Disable the fan.

7.36.4.2 enable()

```
void proxy::Fan::enable ( )
```

Enable the fan.

7.36.4.3 set_speed()

```
void proxy::Fan::set_speed (
    float speed )
```

Set the speed of the fans.

Parameters

<i>speed</i>	Speed percentage of the fan
--------------	-----------------------------

7.36.4.4 stop()

```
void proxy::Fan::stop ( )
```

Stop the fan.

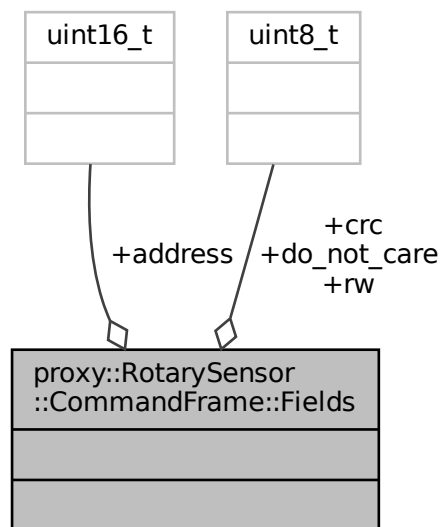
The documentation for this class was generated from the following files:

- [inc/proxy/fan.hpp](#)
- [src/proxy/fan.cpp](#)

7.37 proxy::RotarySensor::CommandFrame::Fields Struct Reference

```
#include <rotary_sensor.hpp>
```

Collaboration diagram for proxy::RotarySensor::CommandFrame::Fields:



Public Attributes

- uint8_t [do_not_care](#): 1
- uint8_t [rw](#): 1
- uint16_t [address](#): 14
- uint8_t [crc](#): 8

7.37.1 Member Data Documentation

7.37.1.1 address

```
uint16_t proxy::RotarySensor::CommandFrame::Fields::address
```

7.37.1.2 crc

```
uint8_t proxy::RotarySensor::CommandFrame::Fields::crc
```


7.37.1.3 do_not_care

```
uint8_t proxy::RotarySensor::CommandFrame::Fields::do_not_care
```

7.37.1.4 rw

```
uint8_t proxy::RotarySensor::CommandFrame::Fields::rw
```

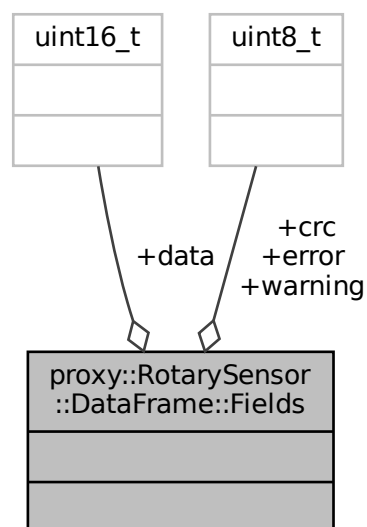
The documentation for this struct was generated from the following file:

- [inc/proxy/rotary_sensor.hpp](#)

7.38 proxy::RotarySensor::DataFrame::Fields Struct Reference

```
#include <rotary_sensor.hpp>
```

Collaboration diagram for proxy::RotarySensor::DataFrame::Fields:



Public Attributes

- [uint8_t warning](#): 1
- [uint8_t error](#): 1
- [uint16_t data](#): 14
- [uint8_t crc](#): 8

7.38.1 Member Data Documentation

7.38.1.1 crc

```
uint8_t proxy::RotarySensor::DataFrame::Fields::crc
```

7.38.1.2 data

```
uint16_t proxy::RotarySensor::DataFrame::Fields::data
```

7.38.1.3 error

```
uint8_t proxy::RotarySensor::DataFrame::Fields::error
```

7.38.1.4 warning

```
uint8_t proxy::RotarySensor::DataFrame::Fields::warning
```

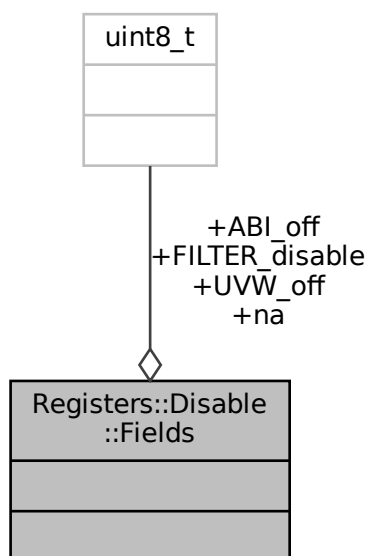
The documentation for this struct was generated from the following file:

- [inc/proxy/rotary_sensor.hpp](#)

7.39 Registers::Disable::Fields Struct Reference

```
#include <rotary_sensor_reg.hpp>
```

Collaboration diagram for Registers::Disable::Fields:



Public Attributes

- uint8_t [UVW_off](#): 1
- uint8_t [ABI_off](#): 1
- uint8_t [na](#): 4
- uint8_t [FILTER_disable](#): 1

7.39.1 Member Data Documentation

7.39.1.1 ABI_off

```
uint8_t Registers::Disable::Fields::ABI_off
```

7.39.1.2 FILTER_disable

```
uint8_t Registers::Disable::Fields::FILTER_disable
```

7.39.1.3 na

```
uint8_t Registers::Disable::Fields::na
```

7.39.1.4 UVW_off

```
uint8_t Registers::Disable::Fields::UVW_off
```

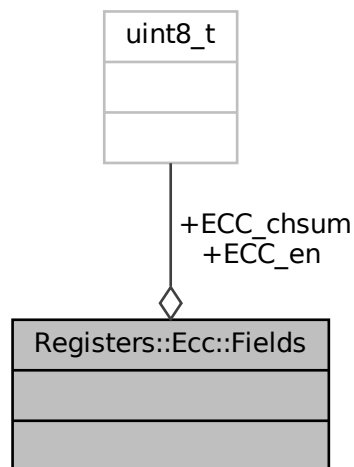
The documentation for this struct was generated from the following file:

- [inc/proxy/rotary_sensor_reg.hpp](#)

7.40 Registers::Ecc::Fields Struct Reference

```
#include <rotary_sensor_reg.hpp>
```

Collaboration diagram for Registers::Ecc::Fields:



Public Attributes

- `uint8_t` [ECC_chsum](#): 7
- `uint8_t` [ECC_en](#): 1

7.40.1 Member Data Documentation

7.40.1.1 ECC_chsum

```
uint8_t Registers::Ecc::Fields::ECC_chsum
```

7.40.1.2 ECC_en

```
uint8_t Registers::Ecc::Fields::ECC_en
```

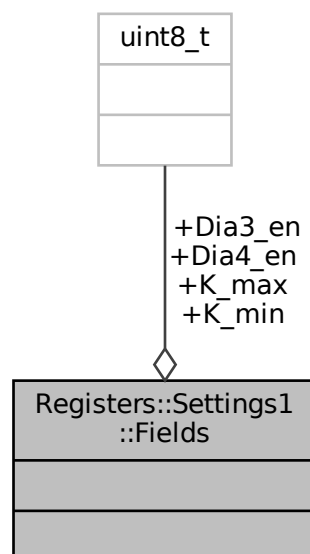
The documentation for this struct was generated from the following file:

- [inc/proxy/rotary_sensor_reg.hpp](#)

7.41 Registers::Settings1::Fields Struct Reference

```
#include <rotary_sensor_reg.hpp>
```

Collaboration diagram for Registers::Settings1::Fields:



Public Attributes

- `uint8_t` [K_max](#): 3
- `uint8_t` [K_min](#): 3
- `uint8_t` [Dia3_en](#): 1
- `uint8_t` [Dia4_en](#): 1

7.41.1 Member Data Documentation

7.41.1.1 Dia3_en

```
uint8_t Registers::Settings1::Fields::Dia3_en
```

7.41.1.2 Dia4_en

```
uint8_t Registers::Settings1::Fields::Dia4_en
```

7.41.1.3 K_max

```
uint8_t Registers::Settings1::Fields::K_max
```

7.41.1.4 K_min

```
uint8_t Registers::Settings1::Fields::K_min
```

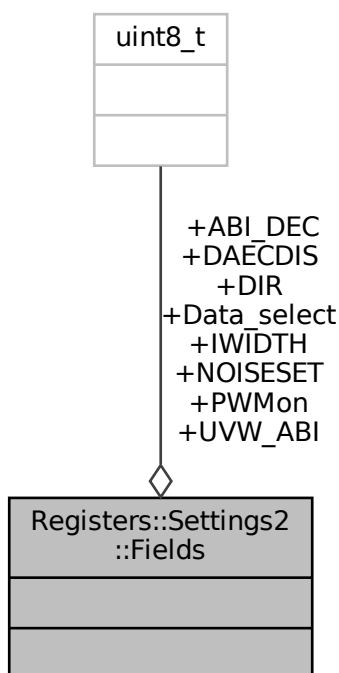
The documentation for this struct was generated from the following file:

- [inc/proxy/rotary_sensor_reg.hpp](#)

7.42 Registers::Settings2::Fields Struct Reference

```
#include <rotary_sensor_reg.hpp>
```

Collaboration diagram for Registers::Settings2::Fields:



Public Attributes

- uint8_t [IWIDTH](#): 1
- uint8_t [NOISESET](#): 1
- uint8_t [DIR](#): 1
- uint8_t [UVW_ABI](#): 1
- uint8_t [DAECDIS](#): 1
- uint8_t [ABI_DEC](#): 1
- uint8_t [Data_select](#): 1
- uint8_t [PWMon](#): 1

7.42.1 Member Data Documentation

7.42.1.1 ABI_DEC

uint8_t Registers::Settings2::Fields::ABI_DEC

7.42.1.2 DAECDIS

```
uint8_t Registers::Settings2::Fields::DAECDIS
```

7.42.1.3 Data_select

```
uint8_t Registers::Settings2::Fields::Data_select
```

7.42.1.4 DIR

```
uint8_t Registers::Settings2::Fields::DIR
```

7.42.1.5 IWIDTH

```
uint8_t Registers::Settings2::Fields::IWIDTH
```

7.42.1.6 NOISESET

```
uint8_t Registers::Settings2::Fields::NOISESET
```

7.42.1.7 PWMon

```
uint8_t Registers::Settings2::Fields::PWMon
```

7.42.1.8 UVW_ABI

```
uint8_t Registers::Settings2::Fields::UVW_ABI
```

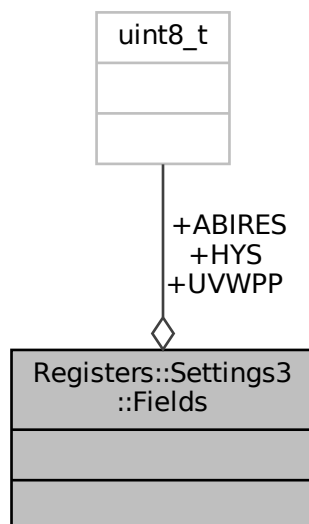
The documentation for this struct was generated from the following file:

- [inc/proxy/rotary_sensor_reg.hpp](#)

7.43 Registers::Settings3::Fields Struct Reference

```
#include <rotary_sensor_reg.hpp>
```

Collaboration diagram for Registers::Settings3::Fields:



Public Attributes

- `uint8_t UVWPP`: 3
- `uint8_t HYS`: 2
- `uint8_t ABIRES`: 3

7.43.1 Member Data Documentation

7.43.1.1 ABIRES

```
uint8_t Registers::Settings3::Fields::ABIRES
```

7.43.1.2 HYS

```
uint8_t Registers::Settings3::Fields::HYS
```

7.43.1.3 UVWPP

```
uint8_t Registers::Settings3::Fields::UVWPP
```

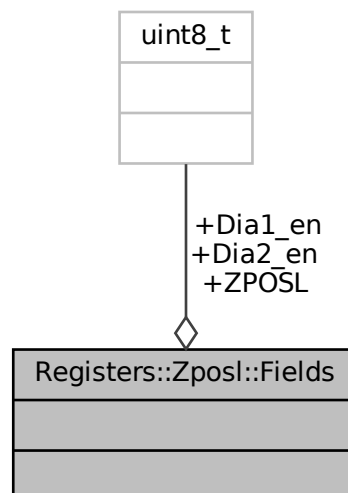
The documentation for this struct was generated from the following file:

- [inc/proxy/rotary_sensor_reg.hpp](#)

7.44 Registers::Zposl::Fields Struct Reference

```
#include <rotary_sensor_reg.hpp>
```

Collaboration diagram for Registers::Zposl::Fields:



Public Attributes

- [uint8_t ZPOS_L](#): 6
- [uint8_t Dia1_en](#): 1
- [uint8_t Dia2_en](#): 1

7.44.1 Member Data Documentation

7.44.1.1 Dia1_en

```
uint8_t Registers::Zposl::Fields::Dia1_en
```

7.44.1.2 Dia2_en

```
uint8_t Registers::Zposl::Fields::Dia2_en
```

7.44.1.3 ZPSL

```
uint8_t Registers::Zposl::Fields::ZPSL
```

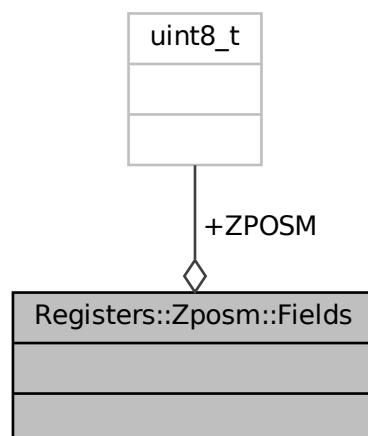
The documentation for this struct was generated from the following file:

- [inc/proxy/rotary_sensor_reg.hpp](#)

7.45 Registers::Zposm::Fields Struct Reference

```
#include <rotary_sensor_reg.hpp>
```

Collaboration diagram for Registers::Zposm::Fields:



Public Attributes

- `uint8_t` [ZPSM](#): 8

7.45.1 Member Data Documentation

7.45.1.1 ZPOSM

```
uint8_t Registers::Zposm::Fields::ZPOSM
```

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

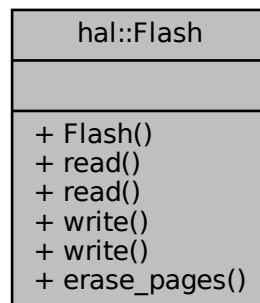
- [inc/proxy/rotary_sensor_reg.hpp](#)

7.46 hal::Flash Class Reference

Class to handle flash memory on STM32 microcontrollers.

```
#include <flash.hpp>
```

Collaboration diagram for hal::Flash:



Public Member Functions

- [Flash](#) ()=delete
Deleted constructor for static class.

Static Public Member Functions

- static void [read](#) (uint32_t address, uint64_t data[], uint32_t size=1)
Read data from flash memory.
- static void [read](#) (uint16_t page, uint16_t page_address, uint64_t data[], uint32_t size=1)
Read data from flash memory.
- static void [write](#) (uint32_t address, const uint64_t data[], uint32_t size=1)
Write data to flash memory.
- static void [write](#) (uint16_t page, uint16_t page_address, const uint64_t data[], uint32_t size=1)
Write data to flash memory.
- static void [erase_pages](#) (uint16_t page, uint16_t number_of_pages=1)
Erase flash memory pages.

7.46.1 Detailed Description

Class to handle flash memory on STM32 microcontrollers.

7.46.2 Constructor & Destructor Documentation

7.46.2.1 Flash()

```
hal::Flash::Flash ( ) [delete]
```

Deleted constructor for static class.

7.46.3 Member Function Documentation

7.46.3.1 erase_pages()

```
void hal::Flash::erase_pages (
    uint16_t page,
    uint16_t number_of_pages = 1 ) [static]
```

Erase flash memory pages.

Parameters

<i>page</i>	first page to erase (counting from the last to the first)
<i>number_of_pages</i>	number of pages to erase

7.46.3.2 read() [1/2]

```
void hal::Flash::read (
    uint16_t page,
    uint16_t page_address,
    uint64_t data[],
    uint32_t size = 1 ) [static]
```

Read data from flash memory.

Parameters

<i>page</i>	page to read from (counting from the last to the first)
-------------	---

Parameters

<i>page_address</i>	address inside the page to read from (indexed by double words)
<i>data</i>	pointer to store the data read
<i>size</i>	size in double words of the data to read

7.46.3.3 read() [2/2]

```
void hal::Flash::read (
    uint32_t address,
    uint64_t data[],
    uint32_t size = 1 ) [static]
```

Read data from flash memory.

Parameters

<i>address</i>	address to read from (indexed by double words)
<i>data</i>	pointer to store the data read
<i>size</i>	size in double words of the data to read

7.46.3.4 write() [1/2]

```
void hal::Flash::write (
    uint16_t page,
    uint16_t page_address,
    const uint64_t data[],
    uint32_t size = 1 ) [static]
```

Write data to flash memory.

Parameters

<i>page</i>	page to write to (counting from the last to the first)
<i>page_address</i>	address inside the page to write to (indexed by double words)
<i>data</i>	pointer to the data to write
<i>size</i>	size in double words of the data to write

7.46.3.5 write() [2/2]

```
void hal::Flash::write (
    uint32_t address,
```

```
const uint64_t data[],
uint32_t size = 1 ) [static]
```

Write data to flash memory.

Parameters

<i>address</i>	address to write to (indexed by double words)
<i>data</i>	pointer to the data to write
<i>size</i>	size in double words of the data to write

The documentation for this class was generated from the following files:

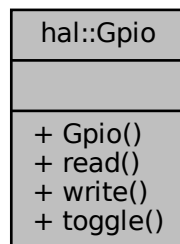
- [inc/hal/flash.hpp](#)
- [src/hal/flash.cpp](#)

7.47 hal::Gpio Class Reference

Class for controlling GPIO pins on STM32 microcontrollers.

```
#include <gpio.hpp>
```

Collaboration diagram for hal::Gpio:



Classes

- struct [Config](#)
Configuration structure for GPIO pin.

Public Member Functions

- [Gpio](#) (const [Config](#) &config)
Constructor for the [Gpio](#) class.
- bool [read](#) () const
Read the current state of the GPIO pin.
- void [write](#) (bool state)
Write a new state to the GPIO pin.
- void [toggle](#) ()
Toggle the state of the GPIO pin.

7.47.1 Detailed Description

Class for controlling GPIO pins on STM32 microcontrollers.

7.47.2 Constructor & Destructor Documentation

7.47.2.1 Gpio()

```
hal::Gpio::Gpio (
    const Config & config ) [explicit]
```

Constructor for the [Gpio](#) class.

Parameters

<i>config</i>	Configuration for the GPIO pin
---------------	--------------------------------

7.47.3 Member Function Documentation

7.47.3.1 read()

```
bool hal::Gpio::read ( ) const
```

Read the current state of the GPIO pin.

Returns

bool The current state of the GPIO pin (true for high, false for low)

7.47.3.2 toggle()

```
void hal::Gpio::toggle ( )
```

Toggle the state of the GPIO pin.

7.47.3.3 write()

```
void hal::Gpio::write (
    bool state )
```

Write a new state to the GPIO pin.

Parameters

<code>pin_state</code>	The state to be written (true for high, false for low)
------------------------	--

The documentation for this class was generated from the following files:

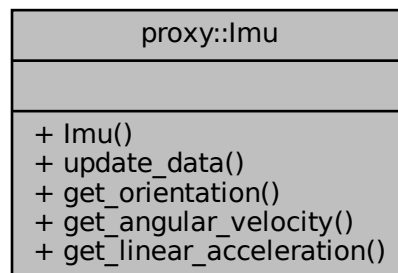
- [inc/hal/gpio.hpp](#)
- [src/hal/gpio.cpp](#)

7.48 proxy::Imu Class Reference

Class to handle IMU peripheral on STM32 microcontrollers.

```
#include <imu.hpp>
```

Collaboration diagram for proxy::Imu:



Classes

- struct [Config](#)
IMU configuration struct.

Public Types

- enum [Axis](#) { [W](#) , [X](#) , [Y](#) , [Z](#) }

Public Member Functions

- [Imu](#) (const [Config](#) &config)
Construct a new [Imu](#) object.
- void [update_data](#) ()
Update the IMU data.
- float [get_orientation](#) ([Axis](#) axis) const
Get the IMU orientation over an axis.
- float [get_angular_velocity](#) ([Axis](#) axis) const
Get the IMU angular velocity over an axis.
- float [get_linear_acceleration](#) ([Axis](#) axis) const
Get the IMU linear acceleration over an axis.

7.48.1 Detailed Description

Class to handle IMU peripheral on STM32 microcontrollers.

7.48.2 Member Enumeration Documentation

7.48.2.1 Axis

```
enum proxy::Imu::Axis
```

Enumerator

W	
X	
Y	
Z	

7.48.3 Constructor & Destructor Documentation

7.48.3.1 Imu()

```
proxy::Imu::Imu (
    const Config & config ) [explicit]
```

Construct a new [Imu](#) object.

Parameters

<i>config</i>	Configuration for the IMU
---------------	---------------------------

7.48.4 Member Function Documentation

7.48.4.1 get_angular_velocity()

```
float proxy::Imu::get_angular_velocity (
    Axis axis ) const
```

Get the IMU angular velocity over an axis.

Parameters

<i>axis</i>	Axis to get the angular velocity from
-------------	---------------------------------------

Returns

float Angular velocity over the desired axis in rad/s

7.48.4.2 get_linear_acceleration()

```
float proxy::Imu::get_linear_acceleration (
    Axis axis ) const
```

Get the IMU linear acceleration over an axis.

Parameters

<i>axis</i>	Axis to get the linear acceleration from
-------------	--

Returns

float Linear acceleration over the desired axis in m/s²

7.48.4.3 get_orientation()

```
float proxy::Imu::get_orientation (
    Axis axis ) const
```

Get the IMU orientation over an axis.

Todo implement function using sensor fusion

Parameters

<i>axis</i>	Axis to get the orientation from
-------------	----------------------------------

Returns

float Orientation over the desired axis using quaternions

7.48.4.4 update_data()

```
void proxy::Imu::update_data ( )
```

Update the IMU data.

The documentation for this class was generated from the following files:

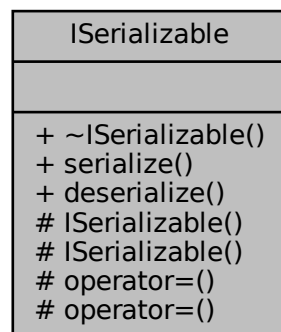
- [inc/proxy/imu.hpp](#)
- [src/proxy/imu.cpp](#)

7.49 ISerializable Class Reference

Interface class for serializable classes.

```
#include <serializable_interface.hpp>
```

Collaboration diagram for ISerializable:



Public Member Functions

- virtual [~ISerializable](#) ()=default
Virtual destructor for the [ISerializable](#) class.
- virtual std::vector< uint8_t > [serialize](#) () const =0
Serialize the class instance.
- virtual void [deserialize](#) (uint8_t *serial_data, uint16_t size)=0
Deserialize the class instance.

Protected Member Functions

- [ISerializable](#) (const [ISerializable](#) &)=default
Special member functions declared as default.
- [ISerializable](#) ([ISerializable](#) &&)=default
- [ISerializable](#) & [operator=](#) (const [ISerializable](#) &)=default
- [ISerializable](#) & [operator=](#) ([ISerializable](#) &&)=default

7.49.1 Detailed Description

Interface class for serializable classes.

7.49.2 Constructor & Destructor Documentation

7.49.2.1 ~ISerializable()

```
virtual ISerializable::~~ISerializable ( ) [virtual], [default]
```

Virtual destructor for the [ISerializable](#) class.

7.49.2.2 ISerializable() [1/2]

```
ISerializable::ISerializable (
    const ISerializable & ) [protected], [default]
```

Special member functions declared as default.

7.49.2.3 ISerializable() [2/2]

```
ISerializable::ISerializable (
    ISerializable && ) [protected], [default]
```

7.49.3 Member Function Documentation

7.49.3.1 deserialize()

```
virtual void ISerializable::deserialize (
    uint8_t * serial_data,
    uint16_t size ) [pure virtual]
```

Deserialize the class instance.

Parameters

<i>serial_data</i>	Serialized data
<i>size</i>	Size of the serialized data

7.49.3.2 operator=() [1/2]

```
ISerializable& ISerializable::operator= (
    const ISerializable & ) [protected], [default]
```

7.49.3.3 operator=() [2/2]

```
ISerializable& ISerializable::operator= (
    ISerializable && ) [protected], [default]
```

7.49.3.4 serialize()

```
virtual std::vector<uint8_t> ISerializable::serialize ( ) const [pure virtual]
```

Serialize the class instance.

Returns

std::vector<uint8_t> Serialized data

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

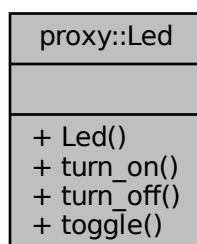
- inc/proxy/[serializable_interface.hpp](#)

7.50 proxy::Led Class Reference

Class for controlling an LED.

```
#include <led.hpp>
```

Collaboration diagram for proxy::Led:



Classes

- struct [Config](#)
Configuration structure for LED.

Public Member Functions

- [Led](#) (const [Config](#) &config)
Constructor for the [Led](#) class.
- void [turn_on](#) ()
Turn the LED on.
- void [turn_off](#) ()
Turn the LED off.
- void [toggle](#) ()
Toggle the LED.

7.50.1 Detailed Description

Class for controlling an LED.

7.50.2 Constructor & Destructor Documentation

7.50.2.1 Led()

```
proxy::Led::Led (  
    const Config & config ) [explicit]
```

Constructor for the [Led](#) class.

Parameters

<i>config</i>	Configuration for the LED
---------------	---------------------------

7.50.3 Member Function Documentation

7.50.3.1 toggle()

```
void proxy::Led::toggle ( )
```

Toggle the LED.

7.50.3.2 turn_off()

```
void proxy::Led::turn_off ( )
```

Turn the LED off.

7.50.3.3 turn_on()

```
void proxy::Led::turn_on ( )
```

Turn the LED on.

The documentation for this class was generated from the following files:

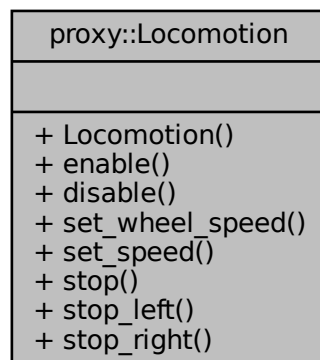
- [inc/proxy/led.hpp](#)
- [src/proxy/led.cpp](#)

7.51 proxy::Locomotion Class Reference

Class for controlling the locomotion driver.

```
#include <locomotion.hpp>
```

Collaboration diagram for proxy::Locomotion:



Classes

- struct [Config](#)

Configuration structure for the locomotion.

Public Member Functions

- [Locomotion](#) (const [Config](#) &config)
Construct a new locomotion object.
- void [enable](#) ()
Enable the locomotion driver.
- void [disable](#) ()
Disable the locomotion driver.
- void [set_wheel_speed](#) (float left_speed, float right_speed)
Set the speed of the wheels.
- void [set_speed](#) (float linear, float angular)
Set the linear and angular speeds of the robot.
- void [stop](#) ()
Stop the motors.
- void [stop_left](#) ()
Stop the left motor.
- void [stop_right](#) ()
Stop the right motor.

7.51.1 Detailed Description

Class for controlling the locomotion driver.

7.51.2 Constructor & Destructor Documentation

7.51.2.1 Locomotion()

```
proxy::Locomotion::Locomotion (
    const Config & config ) [explicit]
```

Construct a new locomotion object.

Parameters

config	Configuration for the locomotion driver
------------------------	---

7.51.3 Member Function Documentation

7.51.3.1 disable()

```
void proxy::Locomotion::disable ( )
```

Disable the locomotion driver.

7.51.3.2 enable()

```
void proxy::Locomotion::enable ( )
```

Enable the locomotion driver.

7.51.3.3 set_speed()

```
void proxy::Locomotion::set_speed (
    float linear,
    float angular )
```

Set the linear and angular speeds of the robot.

Parameters

<i>linear</i>	Linear speed of the robot
<i>angular</i>	Angular speed of the robot

7.51.3.4 set_wheel_speed()

```
void proxy::Locomotion::set_wheel_speed (
    float left_speed,
    float right_speed )
```

Set the speed of the wheels.

Parameters

<i>left_speed</i>	Speed of the left wheels
<i>right_speed</i>	Speed of the right wheels

7.51.3.5 stop()

```
void proxy::Locomotion::stop ( )
```

Stop the motors.

7.51.3.6 stop_left()

```
void proxy::Locomotion::stop_left ( )
```

Stop the left motor.

7.51.3.7 stop_right()

```
void proxy::Locomotion::stop_right ( )
```

Stop the right motor.

The documentation for this class was generated from the following files:

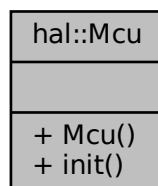
- inc/proxy/[locomotion.hpp](#)
- src/proxy/[locomotion.cpp](#)

7.52 hal::Mcu Class Reference

Microcontroller unit class.

```
#include <mcu.hpp>
```

Collaboration diagram for hal::Mcu:



Public Member Functions

- [Mcu](#) ()=delete
Deleted constructor for static class.

Static Public Member Functions

- static void [init](#) ()
Initializes MCU and some peripherals.

7.52.1 Detailed Description

Microcontroller unit class.

7.52.2 Constructor & Destructor Documentation

7.52.2.1 Mcu()

```
hal::Mcu::Mcu ( ) [delete]
```

Deleted constructor for static class.

7.52.3 Member Function Documentation

7.52.3.1 init()

```
void hal::Mcu::init ( ) [static]
```

Initializes MCU and some peripherals.

The documentation for this class was generated from the following files:

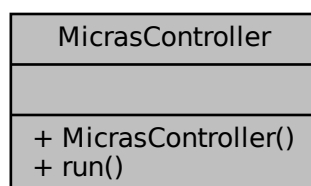
- [inc/hal/mcu.hpp](#)
- [src/hal/mcu.cpp](#)

7.53 MicrasController Class Reference

Class for controlling the Micras robot.

```
#include <micras_controller.hpp>
```

Collaboration diagram for MicrasController:



Public Member Functions

- [MicrasController](#) ()
Constructor for the [MicrasController](#) class.
- void [run](#) ()
Runs the controller loop once.

7.53.1 Detailed Description

Class for controlling the Micras robot.

7.53.2 Constructor & Destructor Documentation

7.53.2.1 MicrasController()

```
MicrasController::MicrasController ( )
```

Constructor for the [MicrasController](#) class.

7.53.3 Member Function Documentation

7.53.3.1 run()

```
void MicrasController::run ( )
```

Runs the controller loop once.

The documentation for this class was generated from the following files:

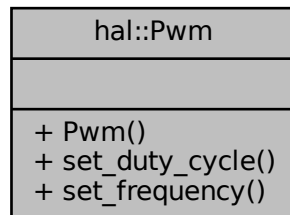
- [inc/controller/micras_controller.hpp](#)
- [src/controller/micras_controller.cpp](#)

7.54 hal::Pwm Class Reference

Class to handle PWM peripheral on STM32 microcontrollers.

```
#include <pwm.hpp>
```

Collaboration diagram for hal::Pwm:



Classes

- struct [Config](#)
PWM configuration struct.

Public Member Functions

- [Pwm](#) (const [Config](#) &config)
Construct a new [Pwm](#) object.
- void [set_duty_cycle](#) (float duty_cycle)
Set the PWM duty cycle.
- void [set_frequency](#) (uint32_t frequency)
Set the PWM frequency.

7.54.1 Detailed Description

Class to handle PWM peripheral on STM32 microcontrollers.

7.54.2 Constructor & Destructor Documentation

7.54.2.1 Pwm()

```
hal::Pwm::Pwm (
    const Config & config ) [explicit]
```

Construct a new [Pwm](#) object.

Parameters

<i>config</i>	Configuration for the PWM
---------------	---------------------------

7.54.3 Member Function Documentation

7.54.3.1 set_duty_cycle()

```
void hal::Pwm::set_duty_cycle (
    float duty_cycle )
```

Set the PWM duty cycle.

Parameters

<i>duty_cycle</i>	Duty cycle value
-------------------	------------------

7.54.3.2 set_frequency()

```
void hal::Pwm::set_frequency (
    uint32_t frequency )
```

Set the PWM frequency.

Parameters

<i>frequency</i>	Frequency value in Hz
------------------	-----------------------

The documentation for this class was generated from the following files:

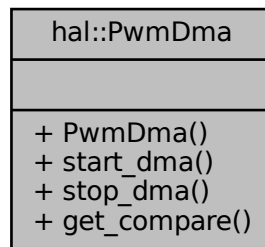
- [inc/hal/pwm.hpp](#)
- [src/hal/pwm.cpp](#)

7.55 hal::PwmDma Class Reference

Class to handle PWM peripheral on STM32 microcontrollers using DMA.

```
#include <pwm_dma.hpp>
```

Collaboration diagram for hal::PwmDma:



Classes

- struct [Config](#)
PWM configuration struct.

Public Member Functions

- [PwmDma](#) (const [Config](#) &config)
Construct a new [PwmDma](#) object.
- void [start_dma](#) (uint32_t buffer[], uint32_t size)
Start PWM and DMA transfer.
- void [stop_dma](#) ()
Stop PWM and DMA transfer.
- uint32_t [get_compare](#) (float duty_cycle) const
Get the compare value for ad duty cycle.

7.55.1 Detailed Description

Class to handle PWM peripheral on STM32 microcontrollers using DMA.

7.55.2 Constructor & Destructor Documentation

7.55.2.1 PwmDma()

```
hal::PwmDma::PwmDma (
    const Config & config ) [explicit]
```

Construct a new [PwmDma](#) object.

Parameters

<i>config</i>	Configuration for the PWM
---------------	---------------------------

7.55.3 Member Function Documentation

7.55.3.1 get_compare()

```
uint32_t hal::PwmDma::get_compare (
    float duty_cycle ) const
```

Get the compare value for ad duty cycle.

Parameters

<i>duty_cycle</i>	Duty cycle to get the compare value for
-------------------	---

Returns

uint32_t Compare value for the duty cycle

7.55.3.2 start_dma()

```
void hal::PwmDma::start_dma (
    uint32_t buffer[],
    uint32_t size )
```

Start PWM and DMA transfer.

Parameters

<i>buffer</i>	Buffer to transfer
<i>size</i>	Size of the buffer

7.55.3.3 stop_dma()

```
void hal::PwmDma::stop_dma ( )
```

Stop PWM and DMA transfer.

The documentation for this class was generated from the following files:

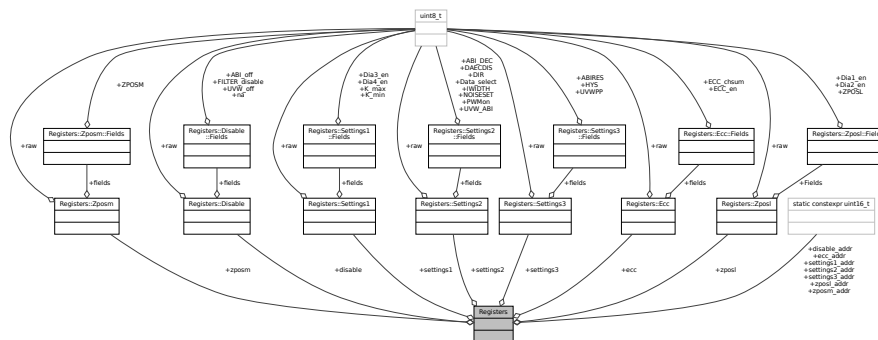
- [inc/hal/pwm_dma.hpp](#)
- [src/hal/pwm_dma.cpp](#)

7.56 Registers Struct Reference

[Registers](#) class for the rotary sensor configuration.

```
#include <rotary_sensor_reg.hpp>
```

Collaboration diagram for Registers:



Classes

- union [Disable](#)
Registers union types definition.
- union [Ecc](#)
- union [Settings1](#)
- union [Settings2](#)
- union [Settings3](#)
- union [Zposl](#)
- union [Zposm](#)

Public Attributes

- [Disable](#) `disable`
Member variables to be configured in the rotary sensor.
- [Zposm](#) `zposm`
- [Zposl](#) `zposl`
- [Settings1](#) `settings1`
- [Settings2](#) `settings2`
- [Settings3](#) `settings3`
- [Ecc](#) `ecc`

Static Public Attributes

- static constexpr uint16_t [disable_addr](#) {0x0015}
Register addresses in the rotary sensor memory.
- static constexpr uint16_t [zposm_addr](#) {0x0016}
- static constexpr uint16_t [zposl_addr](#) {0x0017}
- static constexpr uint16_t [settings1_addr](#) {0x0018}
- static constexpr uint16_t [settings2_addr](#) {0x0019}
- static constexpr uint16_t [settings3_addr](#) {0x001A}
- static constexpr uint16_t [ecc_addr](#) {0x001B}

7.56.1 Detailed Description

[Registers](#) class for the rotary sensor configuration.

7.56.2 Member Data Documentation

7.56.2.1 disable

[Disable](#) `Registers::disable`

Member variables to be configured in the rotary sensor.

7.56.2.2 disable_addr

```
constexpr uint16_t Registers::disable_addr {0x0015} [static], [constexpr]
```

Register addresses in the rotary sensor memory.

7.56.2.3 ecc

[Ecc](#) `Registers::ecc`

7.56.2.4 ecc_addr

```
constexpr uint16_t Registers::ecc_addr {0x001B} [static], [constexpr]
```

7.56.2.5 settings1

`Settings1` Registers::settings1

7.56.2.6 settings1_addr

`constexpr uint16_t` Registers::settings1_addr {0x0018} [static], [constexpr]

7.56.2.7 settings2

`Settings2` Registers::settings2

7.56.2.8 settings2_addr

`constexpr uint16_t` Registers::settings2_addr {0x0019} [static], [constexpr]

7.56.2.9 settings3

`Settings3` Registers::settings3

7.56.2.10 settings3_addr

`constexpr uint16_t` Registers::settings3_addr {0x001A} [static], [constexpr]

7.56.2.11 zposl

`Zposl` Registers::zposl

7.56.2.12 zposl_addr

`constexpr uint16_t` Registers::zposl_addr {0x0017} [static], [constexpr]

7.56.2.13 zposm

`Zposm` `Registers::zposm`

7.56.2.14 zposm_addr

```
constexpr uint16_t Registers::zposm_addr {0x0016} [static], [constexpr]
```

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

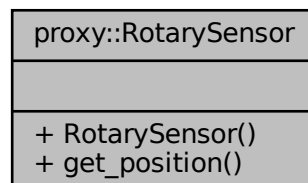
- `inc/proxy/rotary_sensor_reg.hpp`

7.57 proxy::RotarySensor Class Reference

Class to handle rotary sensor peripheral on STM32 microcontrollers.

```
#include <rotary_sensor.hpp>
```

Collaboration diagram for proxy::RotarySensor:



Classes

- struct `Config`
Rotary sensor configuration struct.

Public Member Functions

- `RotarySensor` (const `Config` &config)
Construct a new `RotarySensor` object.
- float `get_position` () const
Get the rotary sensor position over an axis.

7.57.1 Detailed Description

Class to handle rotary sensor peripheral on STM32 microcontrollers.

7.57.2 Constructor & Destructor Documentation

7.57.2.1 RotarySensor()

```
proxy::RotarySensor::RotarySensor (
    const Config & config ) [explicit]
```

Construct a new [RotarySensor](#) object.

Parameters

<i>config</i>	Configuration for the rotary sensor
---------------	-------------------------------------

7.57.3 Member Function Documentation

7.57.3.1 get_position()

```
float proxy::RotarySensor::get_position ( ) const
```

Get the rotary sensor position over an axis.

Returns

float Current angular position of the sensor in radians

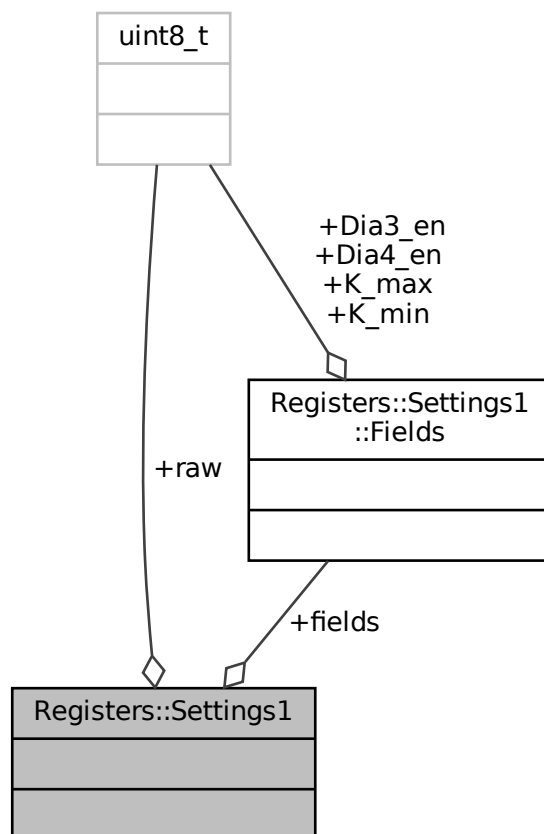
The documentation for this class was generated from the following files:

- [inc/proxy/rotary_sensor.hpp](#)
- [src/proxy/rotary_sensor.cpp](#)

7.58 Registers::Settings1 Union Reference

```
#include <rotary_sensor_reg.hpp>
```

Collaboration diagram for Registers::Settings1:



Classes

- struct [Fields](#)

Public Attributes

- [Fields](#) `fields`
- `uint8_t` `raw`

7.58.1 Member Data Documentation

7.58.1.1 fields

`Fields` `Registers::Settings1::fields`

7.58.1.2 raw

`uint8_t` `Registers::Settings1::raw`

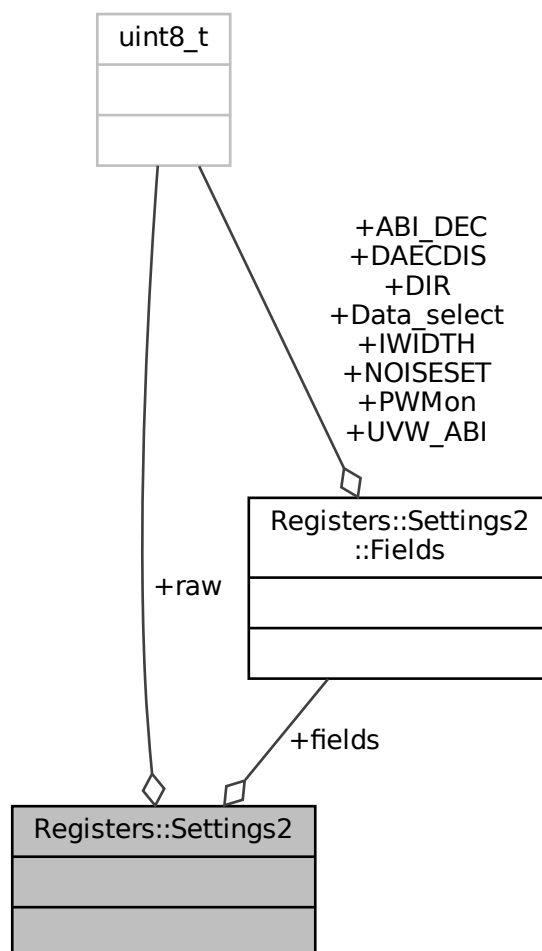
The documentation for this union was generated from the following file:

- [inc/proxy/rotary_sensor_reg.hpp](#)

7.59 Registers::Settings2 Union Reference

```
#include <rotary_sensor_reg.hpp>
```

Collaboration diagram for `Registers::Settings2`:



Classes

- struct [Fields](#)

Public Attributes

- [Fields](#) `fields`
- `uint8_t` `raw`

7.59.1 Member Data Documentation

7.59.1.1 `fields`

[Fields](#) `Registers::Settings2::fields`

7.59.1.2 `raw`

`uint8_t` `Registers::Settings2::raw`

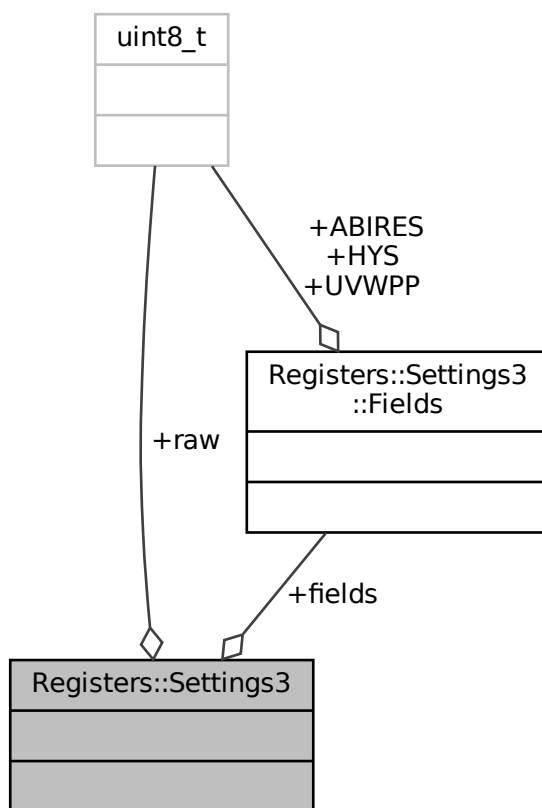
The documentation for this union was generated from the following file:

- `inc/proxy/rotary_sensor_reg.hpp`

7.60 Registers::Settings3 Union Reference

```
#include <rotary_sensor_reg.hpp>
```

Collaboration diagram for Registers::Settings3:



Classes

- struct [Fields](#)

Public Attributes

- [Fields](#) `fields`
- `uint8_t` `raw`

7.60.1 Member Data Documentation

7.60.1.1 fields

[Fields](#) `Registers::Settings3::fields`

7.60.1.2 raw

```
uint8_t Registers::Settings3::raw
```

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

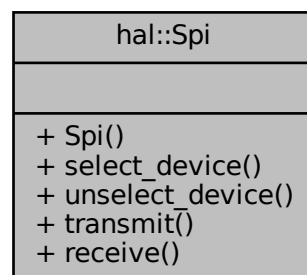
- [inc/proxy/rotary_sensor_reg.hpp](#)

7.61 hal::Spi Class Reference

Class to handle SPI peripheral on STM32 microcontrollers.

```
#include <spi.hpp>
```

Collaboration diagram for hal::Spi:



Classes

- struct [Config](#)
SPI configuration struct.

Public Member Functions

- [Spi](#) (const [Config](#) &config)
Construct a new Spi object.
- bool [select_device](#) ()
Activate the chip select.
- void [unselect_device](#) ()
Deactivate the chip select.
- void [transmit](#) (uint8_t data[], uint32_t size)
Transmit data over SPI.
- void [receive](#) (uint8_t data[], uint32_t size)
Receive data over SPI.

7.61.1 Detailed Description

Class to handle SPI peripheral on STM32 microcontrollers.

7.61.2 Constructor & Destructor Documentation

7.61.2.1 Spi()

```
hal::Spi::Spi (
    const Config & config ) [explicit]
```

Construct a new Spi object.

Parameters

<i>config</i>	Configuration for the SPI
---------------	---------------------------

7.61.3 Member Function Documentation

7.61.3.1 receive()

```
void hal::Spi::receive (
    uint8_t data[],
    uint32_t size )
```

Receive data over SPI.

Parameters

<i>data</i>	Data to receive data
<i>size</i>	Size of the data

7.61.3.2 select_device()

```
bool hal::Spi::select_device ( )
```

Activate the chip select.

Returns

bool True if the device was successfully selected, false otherwise

7.61.3.3 transmit()

```
void hal::Spi::transmit (
    uint8_t data[],
    uint32_t size )
```

Transmit data over SPI.

Parameters

<i>data</i>	Data to transmit
<i>size</i>	Size of the buffer

7.61.3.4 unselect_device()

```
void hal::Spi::unselect_device ( )
```

Deactivate the chip select.

The documentation for this class was generated from the following files:

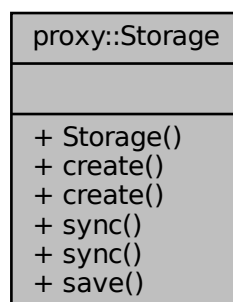
- [inc/hal/spi.hpp](#)
- [src/hal/spi.cpp](#)

7.62 proxy::Storage Class Reference

Class for controlling the storage.

```
#include <storage.hpp>
```

Collaboration diagram for proxy::Storage:



Classes

- struct [Config](#)

Configuration structure for the storage.

Public Member Functions

- [Storage](#) (const [Config](#) &config)
Constructor for the [Storage](#) class.
- template<Fundamental T>
void [create](#) (const std::string &name, const T &data)
Create a new primitive variable in the storage.
- void [create](#) (const std::string &name, const [ISerializable](#) &data)
Create a new serializable variable in the storage.
- template<Fundamental T>
void [sync](#) (const std::string &name, T &data)
Sync a primitive variable with the storage.
- void [sync](#) (const std::string &name, [ISerializable](#) &data)
Sync a serializable variable with the storage.
- void [save](#) ()
Save the storage to the flash.

7.62.1 Detailed Description

Class for controlling the storage.

7.62.2 Constructor & Destructor Documentation

7.62.2.1 [Storage](#)()

```
proxy::Storage::Storage (  
    const Config & config ) [explicit]
```

Constructor for the [Storage](#) class.

Parameters

<i>config</i>	Configuration for the storage
---------------	-------------------------------

7.62.3 Member Function Documentation

7.62.3.1 create() [1/2]

```
void proxy::Storage::create (
    const std::string & name,
    const ISerializable & data )
```

Create a new serializable variable in the storage.

Parameters

<i>name</i>	Name of the variable
<i>data</i>	Reference to the variable

7.62.3.2 create() [2/2]

```
template<Fundamental T>
void proxy::Storage::create (
    const std::string & name,
    const T & data )
```

Create a new primitive variable in the storage.

Template Parameters

<i>T</i>	Type of the variable
----------	----------------------

Parameters

<i>name</i>	Name of the variable
<i>data</i>	Reference to the variable

7.62.3.3 save()

```
void proxy::Storage::save ( )
```

Save the storage to the flash.

7.62.3.4 sync() [1/2]

```
void proxy::Storage::sync (
    const std::string & name,
    ISerializable & data )
```

Sync a serializable variable with the storage.

Parameters

<i>name</i>	Name of the variable
<i>data</i>	Reference to the variable

7.62.3.5 sync() [2/2]

```
template<Fundamental T>
void proxy::Storage::sync (
    const std::string & name,
    T & data )
```

Sync a primitive variable with the storage.

Template Parameters

<i>T</i>	Type of the variable
----------	----------------------

Parameters

<i>name</i>	Name of the variable
<i>data</i>	Reference to the variable

The documentation for this class was generated from the following files:

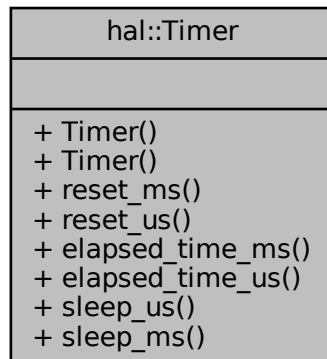
- inc/proxy/[storage.hpp](#)
- src/proxy/[storage.cpp](#)

7.63 hal::Timer Class Reference

Class to handle timer peripheral on STM32 microcontrollers.

```
#include <timer.hpp>
```


Collaboration diagram for hal::Timer:



Classes

- struct [Config](#)
Timer configuration struct.

Public Member Functions

- [Timer](#) ()=default
Construct a new [Timer](#) object.
- [Timer](#) (const [Config](#) &config)
Construct a new [Timer](#) object.
- void [reset_ms](#) ()
Reset the timer counter in milliseconds.
- void [reset_us](#) ()
Reset the timer counter in microseconds.
- uint32_t [elapsed_time_ms](#) () const
Get the time elapsed since the last reset.
- uint32_t [elapsed_time_us](#) () const
Get the time elapsed since the last reset.
- void [sleep_us](#) (uint32_t time) const
Sleep for a given amount of time.

Static Public Member Functions

- static void [sleep_ms](#) (uint32_t time)
Sleep for a given amount of time.

7.63.1 Detailed Description

Class to handle timer peripheral on STM32 microcontrollers.

7.63.2 Constructor & Destructor Documentation

7.63.2.1 Timer() [1/2]

```
hal::Timer::Timer ( ) [default]
```

Construct a new [Timer](#) object.

7.63.2.2 Timer() [2/2]

```
hal::Timer::Timer (
    const Config & config ) [explicit]
```

Construct a new [Timer](#) object.

Parameters

<i>config</i>	Configuration for the timer
---------------	-----------------------------

7.63.3 Member Function Documentation

7.63.3.1 elapsed_time_ms()

```
uint32_t hal::Timer::elapsed_time_ms ( ) const
```

Get the time elapsed since the last reset.

Returns

uint32_t Time elapsed in milliseconds

7.63.3.2 elapsed_time_us()

```
uint32_t hal::Timer::elapsed_time_us ( ) const
```

Get the time elapsed since the last reset.

Returns

uint32_t Time elapsed in microseconds

7.63.3.3 reset_ms()

```
void hal::Timer::reset_ms ( )
```

Reset the timer counter in milliseconds.

7.63.3.4 reset_us()

```
void hal::Timer::reset_us ( )
```

Reset the timer counter in microseconds.

7.63.3.5 sleep_ms()

```
void hal::Timer::sleep_ms (
    uint32_t time ) [static]
```

Sleep for a given amount of time.

Parameters

<i>time</i>	Time to sleep in milliseconds
-------------	-------------------------------

7.63.3.6 sleep_us()

```
void hal::Timer::sleep_us (
    uint32_t time ) const
```

Sleep for a given amount of time.

Parameters

<i>time</i>	Time to sleep in microseconds
-------------	-------------------------------

The documentation for this class was generated from the following files:

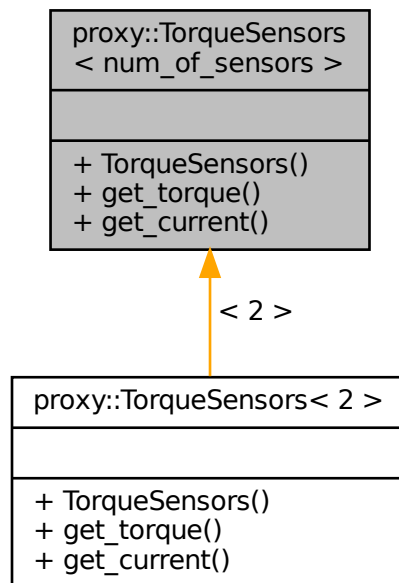
- [inc/hal/timer.hpp](#)
- [src/hal/timer.cpp](#)

7.64 proxy::TorqueSensors< num_of_sensors > Class Template Reference

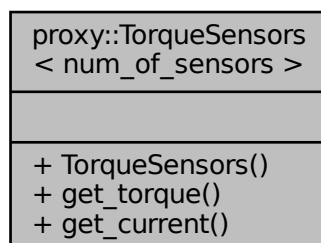
Class for controlling [TorqueSensors](#).

```
#include <torque_sensors.hpp>
```

Inheritance diagram for proxy::TorqueSensors< num_of_sensors >:



Collaboration diagram for proxy::TorqueSensors< num_of_sensors >:



Classes

- struct [Config](#)

Configuration structure for torque sensors.

Public Member Functions

- [TorqueSensors](#) (const [Config](#) &config)
Constructor for the [TorqueSensors](#) class.
- float [get_torque](#) (uint8_t sensor_index) const
Get the torque from the sensor.
- float [get_current](#) (uint8_t sensor_index) const
Get the current from the sensor.

7.64.1 Detailed Description

```
template<uint8_t num_of_sensors>
class proxy::TorqueSensors< num_of_sensors >
```

Class for controlling [TorqueSensors](#).

7.64.2 Constructor & Destructor Documentation

7.64.2.1 TorqueSensors()

```
template<uint8_t num_of_sensors>
proxy::TorqueSensors< num_of_sensors >::TorqueSensors (
    const Config & config ) [explicit]
```

Constructor for the [TorqueSensors](#) class.

Parameters

<i>config</i>	Configuration for the torque sensors
---------------	--------------------------------------

7.64.3 Member Function Documentation

7.64.3.1 get_current()

```
template<uint8_t num_of_sensors>
float proxy::TorqueSensors< num_of_sensors >::get_current (
    uint8_t sensor_index ) const
```

Get the current from the sensor.

Parameters

<i>sensor_index</i>	Index of the sensor
---------------------	---------------------

Returns

float Current reading from the sensor in amps

7.64.3.2 get_torque()

```
template<uint8_t num_of_sensors>
float proxy::TorqueSensors< num_of_sensors >::get_torque (
    uint8_t sensor_index ) const
```

Get the torque from the sensor.

Parameters

<i>sensor_index</i>	Index of the sensor
---------------------	---------------------

Returns

float Torque reading from the sensor in N*m

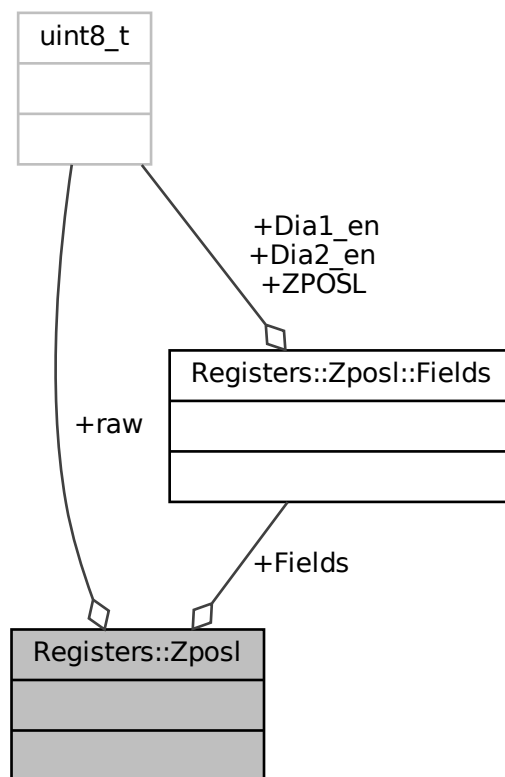
The documentation for this class was generated from the following files:

- [inc/proxy/torque_sensors.hpp](#)
- [src/proxy/torque_sensors.cpp](#)

7.65 Registers::Zposl Union Reference

```
#include <rotary_sensor_reg.hpp>
```

Collaboration diagram for Registers::Zposl:



Classes

- struct [Fields](#)

Public Attributes

- [Fields](#) `Fields`
- `uint8_t` `raw`

7.65.1 Member Data Documentation

7.65.1.1 Fields

`Fields` `Registers::Zposl::Fields`

7.65.1.2 raw

```
uint8_t Registers::Zposl::raw
```

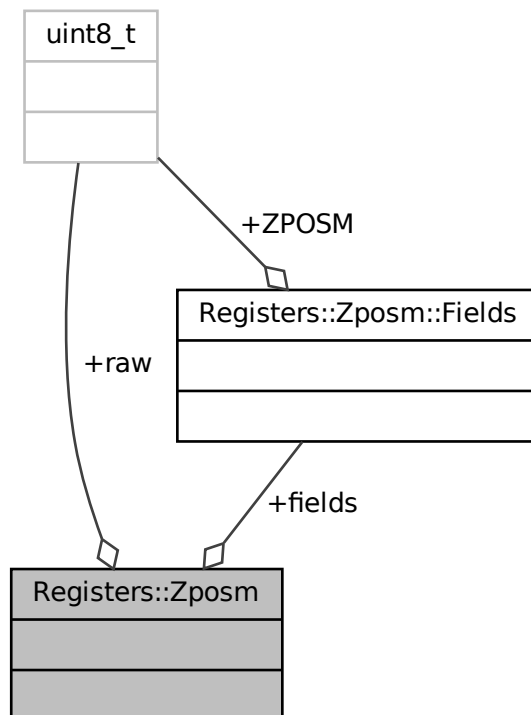
The documentation for this union was generated from the following file:

- inc/proxy/[rotary_sensor_reg.hpp](#)

7.66 Registers::Zposm Union Reference

```
#include <rotary_sensor_reg.hpp>
```

Collaboration diagram for Registers::Zposm:



Classes

- struct [Fields](#)

Public Attributes

- [Fields](#) `fields`
- `uint8_t` `raw`

7.66.1 Member Data Documentation

7.66.1.1 fields

[Fields](#) Registers::Zposm::fields

7.66.1.2 raw

uint8_t Registers::Zposm::raw

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

- [inc/proxy/rotary_sensor_reg.hpp](#)

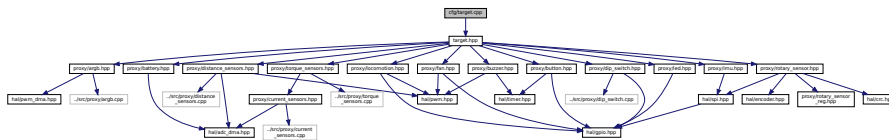
Chapter 8

File Documentation

8.1 cfg/target.cpp File Reference

Target specific configuration.

```
#include "target.hpp"  
Include dependency graph for target.cpp:
```



Variables

- const [proxy::Argb< 2 >::Config](#) [argb_config](#)
- const [proxy::Battery::Config](#) [battery_config](#)
- const [proxy::Button::Config](#) [button_config](#)
- const [proxy::Buzzer::Config](#) [buzzer_config](#)
- const [proxy::DipSwitch< 4 >::Config](#) [dip_switch_config](#)
- const [proxy::DistanceSensors< 4 >::Config](#) [distance_sensors_config](#)
- const [proxy::Fan::Config](#) [fan_config](#)
- const [proxy::Imu::Config](#) [imu_config](#)
- const [proxy::Led::Config](#) [led_config](#)
- const [proxy::Locomotion::Config](#) [locomotion_config](#)
- const [proxy::RotarySensor::Registers](#) [rotary_sensor_reg_config](#)
- const [proxy::RotarySensor::Config](#) [rotary_sensor_left_config](#)
- const [proxy::RotarySensor::Config](#) [rotary_sensor_right_config](#)
- const [proxy::TorqueSensors< 2 >::Config](#) [torque_sensors_config](#)

8.1.1 Detailed Description

Target specific configuration.

Date

03/2024

8.1.2 Variable Documentation

8.1.2.1 argb_config

```
const proxy::Argb<2>::Config argb_config
```

Initial value:

```
= {  
    .pwm = {  
        .handle = &htim8,  
        .init_function = MX_TIM8_Init,  
        .timer_channel = TIM_CHANNEL_1  
    }  
}
```

8.1.2.2 battery_config

```
const proxy::Battery::Config battery_config
```

Initial value:

```
= {  
    .adc = {  
        .handle = &hadc3,  
        .init_function = MX_ADC3_Init,  
        .max_reading = 4095,  
        .reference_voltage = 3.0F  
    },  
    .voltage_divider = 3.0F  
}
```

8.1.2.3 button_config

```
const proxy::Button::Config button_config
```

Initial value:

```
= {  
    .gpio = {  
        .port = GPIOA,  
        .pin = GPIO_PIN_12  
    },  
    .pull_resistor = proxy::Button::PullResistor::PULL_UP  
}
```

8.1.2.4 buzzer_config

```
const proxy::Buzzer::Config buzzer_config
```

Initial value:

```
= {  
    .pwm = {  
        .handle = &htim4,  
        .init_function = MX_TIM4_Init,  
        .timer_channel = TIM_CHANNEL_1  
    }  
}
```

8.1.2.5 dip_switch_config

```
const proxy::DipSwitch<4>::Config dip_switch_config
```

Initial value:

```
= {  
    .gpio_array = {{  
        {  
            .port = GPIOC,  
            .pin = GPIO_PIN_7  
        },  
        {  
            .port = GPIOB,  
            .pin = GPIO_PIN_15  
        },  
        {  
            .port = GPIOB,  
            .pin = GPIO_PIN_14  
        },  
        {  
            .port = GPIOB,  
            .pin = GPIO_PIN_13  
        }  
    }}  
}
```

8.1.2.6 distance_sensors_config

```
const proxy::DistanceSensors<4>::Config distance_sensors_config
```

Initial value:

```
= {  
    .adc = {  
        .handle = &hadc1,  
        .init_function = MX_ADC1_Init,  
        .max_reading = 4095,  
        .reference_voltage = 3.3F  
    },  
    .led_pwm = {  
        .handle = &htim15,  
        .init_function = MX_TIM15_Init,  
        .timer_channel = TIM_CHANNEL_1  
    },  
    .max_distance = 0.3F  
}
```

8.1.2.7 fan_config

```
const proxy::Fan::Config fan_config
```

Initial value:

```
= {  
    .pwm = {  
        .handle = &htim17,  
        .init_function = MX_TIM17_Init,  
        .timer_channel = TIM_CHANNEL_1  
    },  
    .direction_gpio = {  
        .port = GPIOC,  
        .pin = GPIO_PIN_13  
    },  
    .enable_gpio = {  
        .port = GPIOB,  
        .pin = GPIO_PIN_7  
    }  
}
```

8.1.2.8 imu_config

```
const proxy::Imu::Config imu_config
```

Initial value:

```
= {  
    .spi = {  
        .handle = &hspi1,  
        .init_function = MX_SPI1_Init,  
        .gpio = {  
            .port = GPIOB,  
            .pin = GPIO_PIN_6  
        },  
        .timeout = 2  
    },  
    .gyroscope_data_rate = LSM6DSV_ODR_AT_480Hz,  
    .accelerometer_data_rate = LSM6DSV_ODR_AT_480Hz,  
    .orientation_data_rate = LSM6DSV_SFLP_480Hz,  
    .gyroscope_scale = LSM6DSV_4000dps,  
    .accelerometer_scale = LSM6DSV_8g,  
    .gyroscope_filter = LSM6DSV_GY_ULTRA_LIGHT,  
    .accelerometer_filter = LSM6DSV_XL_ULTRA_LIGHT  
}
```

8.1.2.9 led_config

```
const proxy::Led::Config led_config
```

Initial value:

```
= {  
    .gpio = {  
        .port = GPIOA,  
        .pin = GPIO_PIN_15  
    }  
}
```

8.1.2.10 locomotion_config

```
const proxy::Locomotion::Config locomotion_config
```

Initial value:

```
= {  
    .pwm_left_fwd = {  
        .handle = &htim4,  
        .init_function = MX_TIM4_Init,  
        .timer_channel = TIM_CHANNEL_4  
    },  
    .pwm_left_bwd = {  
        .handle = &htim4,  
        .init_function = MX_TIM4_Init,  
        .timer_channel = TIM_CHANNEL_3  
    },  
    .pwm_right_fwd = {  
        .handle = &htim1,  
        .init_function = MX_TIM1_Init,  
        .timer_channel = TIM_CHANNEL_2  
    },  
    .pwm_right_bwd = {  
        .handle = &htim1,  
        .init_function = MX_TIM1_Init,  
        .timer_channel = TIM_CHANNEL_1  
    },  
    .enable_gpio = {  
        .port = GPIOA,  
        .pin = GPIO_PIN_10  
    }  
}
```

8.1.2.11 rotary_sensor_left_config

```
const proxy::RotarySensor::Config rotary_sensor_left_config
```

Initial value:

```
= {  
    .spi = {  
        .handle = &hspi1,  
        .init_function = MX_SPI1_Init,  
        .gpio = {  
            .port = GPIOA,  
            .pin = GPIO_PIN_6  
        },  
        .timeout = 2  
    },  
    .encoder = {  
        .handle = &htim2,  
        .init_function = MX_TIM2_Init,  
        .timer_channel = TIM_CHANNEL_ALL  
    },  
    .crc = {  
        .handle = &hcrc  
    },  
    .resolution = 4096,  
    .registers = rotary_sensor_reg_config  
}
```

8.1.2.12 rotary_sensor_reg_config

```
const proxy::RotarySensor::Registers rotary_sensor_reg_config
```

8.1.2.13 rotary_sensor_right_config

```
const proxy::RotarySensor::Config rotary_sensor_right_config
```

Initial value:

```
= {  
    .spi = {  
        .handle = &hspi1,  
        .init_function = MX_SPI1_Init,  
        .gpio = {  
            .port = GPIOB,  
            .pin = GPIO_PIN_1  
        },  
        .timeout = 2  
    },  
    .encoder = {  
        .handle = &htim5,  
        .init_function = MX_TIM5_Init,  
        .timer_channel = TIM_CHANNEL_ALL  
    },  
    .crc = {  
        .handle = &hcrc  
    },  
    .resolution = 4096,  
    .registers = rotary_sensor_reg_config  
}
```

8.1.2.14 torque_sensors_config

```
const proxy::TorqueSensors<2>::Config torque_sensors_config
```

Initial value:

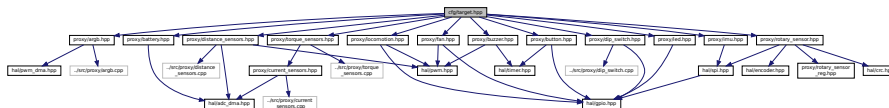
```
= {
    .current_sensors = {
        .adc = {
            .handle = &hadc2,
            .init_function = MX_ADC2_Init,
            .max_reading = 4095,
            .reference_voltage = 3.3F
        },
        .shunt_resistor = 0.04F
    },
    .max_torque = 0.5F
}
```

8.2 cfg/target.hpp File Reference

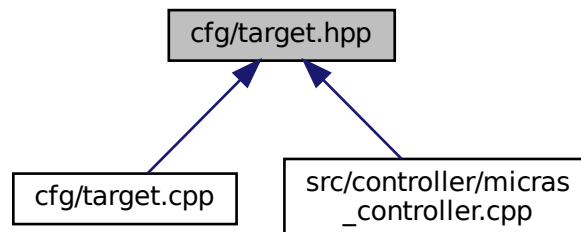
Target specific configuration.

```
#include "proxy/argb.hpp"
#include "proxy/battery.hpp"
#include "proxy/button.hpp"
#include "proxy/buzzer.hpp"
#include "proxy/dip_switch.hpp"
#include "proxy/distance_sensors.hpp"
#include "proxy/fan.hpp"
#include "proxy/imu.hpp"
#include "proxy/led.hpp"
#include "proxy/locomotion.hpp"
#include "proxy/rotary_sensor.hpp"
#include "proxy/torque_sensors.hpp"
```

Include dependency graph for target.hpp:



This graph shows which files directly or indirectly include this file:



Variables

- const [proxy::Argb< 2 >::Config](#) [argb_config](#)
- const [proxy::Battery::Config](#) [battery_config](#)
- const [proxy::Button::Config](#) [button_config](#)
- const [proxy::Buzzer::Config](#) [buzzer_config](#)
- const [proxy::DipSwitch< 4 >::Config](#) [dip_switch_config](#)
- const [proxy::DistanceSensors< 4 >::Config](#) [distance_sensors_config](#)
- const [proxy::Fan::Config](#) [fan_config](#)
- const [proxy::Imu::Config](#) [imu_config](#)
- const [proxy::Led::Config](#) [led_config](#)
- const [proxy::Locomotion::Config](#) [locomotion_config](#)
- const [proxy::RotarySensor::Config](#) [rotary_sensor_left_config](#)
- const [proxy::RotarySensor::Config](#) [rotary_sensor_right_config](#)
- const [proxy::TorqueSensors< 2 >::Config](#) [torque_sensors_config](#)

8.2.1 Detailed Description

Target specific configuration.

Date

03/2024

8.2.2 Variable Documentation

8.2.2.1 [argb_config](#)

```
const proxy::Argb<2>::Config argb\_config [extern]
```

8.2.2.2 [battery_config](#)

```
const proxy::Battery::Config battery\_config [extern]
```

8.2.2.3 [button_config](#)

```
const proxy::Button::Config button\_config [extern]
```

8.2.2.4 buzzer_config

```
const proxy::Buzzer::Config buzzer_config [extern]
```

8.2.2.5 dip_switch_config

```
const proxy::DipSwitch<4>::Config dip_switch_config [extern]
```

8.2.2.6 distance_sensors_config

```
const proxy::DistanceSensors<4>::Config distance_sensors_config [extern]
```

8.2.2.7 fan_config

```
const proxy::Fan::Config fan_config [extern]
```

8.2.2.8 imu_config

```
const proxy::Imu::Config imu_config [extern]
```

8.2.2.9 led_config

```
const proxy::Led::Config led_config [extern]
```

8.2.2.10 locomotion_config

```
const proxy::Locomotion::Config locomotion_config [extern]
```

8.2.2.11 rotary_sensor_left_config

```
const proxy::RotarySensor::Config rotary_sensor_left_config [extern]
```

8.2.2.12 rotary_sensor_right_config

```
const proxy::RotarySensor::Config rotary_sensor_right_config [extern]
```

8.2.2.13 torque_sensors_config

```
const proxy::TorqueSensors<2>::Config torque_sensors_config [extern]
```

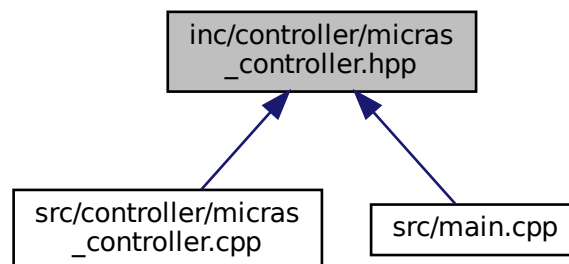
8.3 inc/controller/micras_controller.hpp File Reference

```
#include "proxy/argb.hpp"
#include "proxy/battery.hpp"
#include "proxy/button.hpp"
#include "proxy/buzzer.hpp"
#include "proxy/dip_switch.hpp"
#include "proxy/distance_sensors.hpp"
#include "proxy/fan.hpp"
#include "proxy/imu.hpp"
#include "proxy/led.hpp"
#include "proxy/locomotion.hpp"
#include "proxy/rotary_sensor.hpp"
#include "proxy/torque_sensors.hpp"
```

Include dependency graph for micras_controller.hpp:



This graph shows which files directly or indirectly include this file:



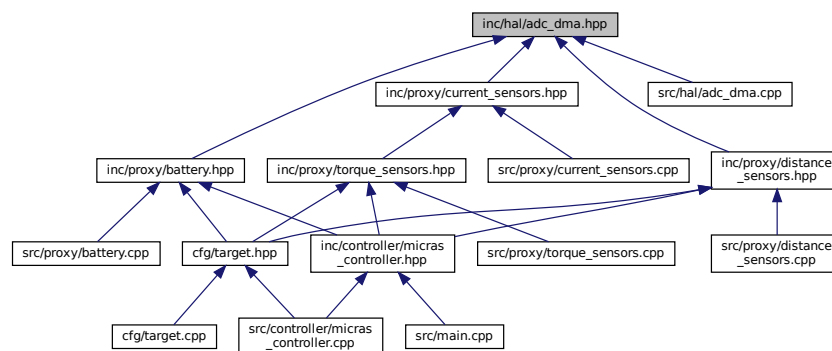
Classes

- class [MicrasController](#)
Class for controlling the Micras robot.

8.4 inc/hal/adc_dma.hpp File Reference

ADC DMA HAL header.

This graph shows which files directly or indirectly include this file:



Classes

- class [hal::AdcDma](#)
Class to handle ADC peripheral on STM32 microcontrollers using DMA.
- struct [hal::AdcDma::Config](#)
Configuration structure for ADC DMA.

Namespaces

- [hal](#)

8.4.1 Detailed Description

ADC DMA HAL header.

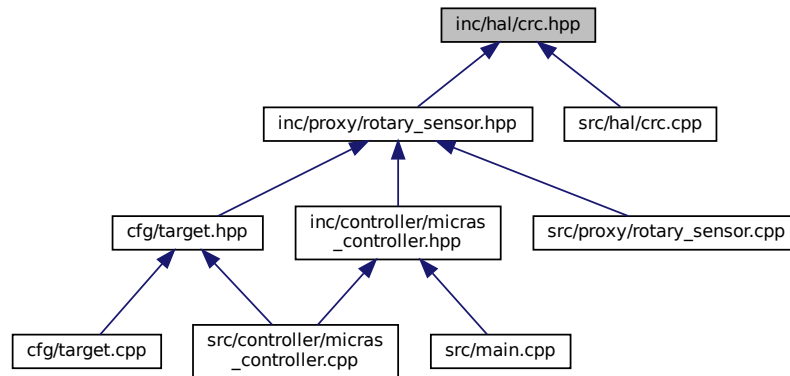
Date

03/2024

8.5 inc/hal/crc.hpp File Reference

STM32 CRC HAL wrapper.

This graph shows which files directly or indirectly include this file:



Classes

- class [hal::Crc](#)
Class to handle the cyclic redundancy check peripheral on STM32 microcontrollers.
- struct [hal::Crc::Config](#)
CRC configuration struct.

Namespaces

- [hal](#)

8.5.1 Detailed Description

STM32 CRC HAL wrapper.

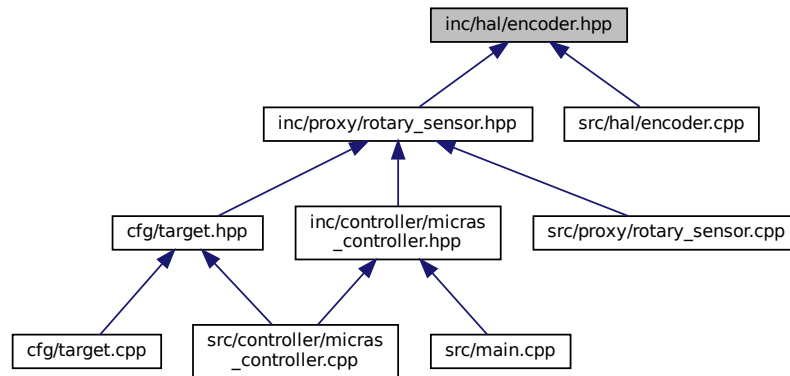
Date

03/2024

8.6 inc/hal/encoder.hpp File Reference

STM32 encoder HAL wrapper.

This graph shows which files directly or indirectly include this file:



Classes

- class [hal::Encoder](#)
Class to handle encoder peripheral on STM32 microcontrollers.
- struct [hal::Encoder::Config](#)
Encoder configuration struct.

Namespaces

- [hal](#)

8.6.1 Detailed Description

STM32 encoder HAL wrapper.

Date

03/2024

Classes

- class [hal::Gpio](#)
Class for controlling GPIO pins on STM32 microcontrollers.
- struct [hal::Gpio::Config](#)
Configuration structure for GPIO pin.

Namespaces

- [hal](#)

8.8.1 Detailed Description

HAL GPIO class header.

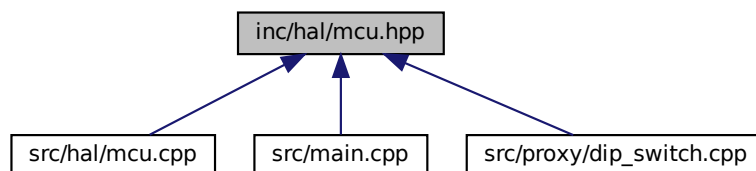
Date

03/2024

8.9 inc/hal/mcu.hpp File Reference

MCU related.

This graph shows which files directly or indirectly include this file:



Classes

- class [hal::Mcu](#)
Microcontroller unit class.

Namespaces

- [hal](#)

8.9.1 Detailed Description

MCU related.

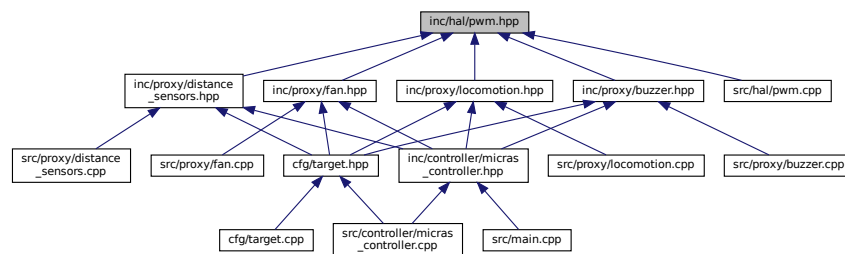
Date

03/2024

8.10 inc/hal/pwm.hpp File Reference

STM32 PWM HAL wrapper.

This graph shows which files directly or indirectly include this file:



Classes

- class [hal::Pwm](#)
Class to handle PWM peripheral on STM32 microcontrollers.
- struct [hal::Pwm::Config](#)
PWM configuration struct.

Namespaces

- [hal](#)

8.10.1 Detailed Description

STM32 PWM HAL wrapper.

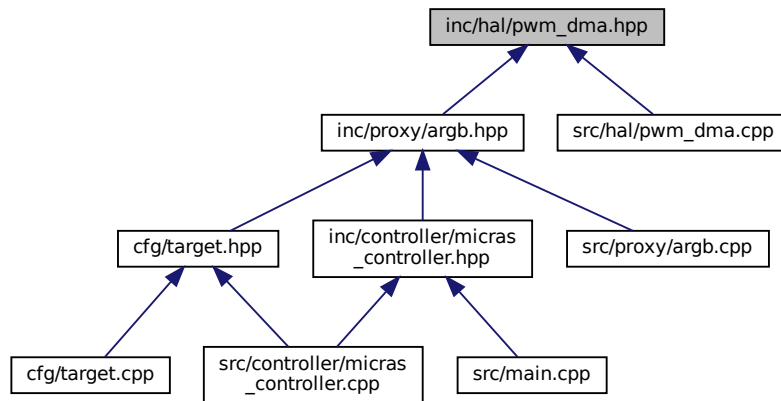
Date

03/2024

8.11 inc/hal/pwm_dma.hpp File Reference

STM32 PWM DMA HAL wrapper.

This graph shows which files directly or indirectly include this file:



Classes

- class [hal::PwmDma](#)
Class to handle PWM peripheral on STM32 microcontrollers using DMA.
- struct [hal::PwmDma::Config](#)
PWM configuration struct.

Namespaces

- [hal](#)

8.11.1 Detailed Description

STM32 PWM DMA HAL wrapper.

Date

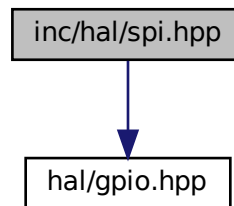
03/2024

8.12 inc/hal/spi.hpp File Reference

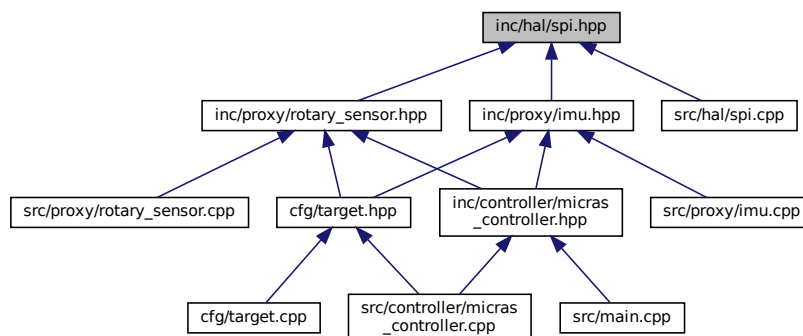
STM32 SPI HAL wrapper.

```
#include "hal/gpio.hpp"
```

Include dependency graph for spi.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class [hal::Spi](#)
Class to handle SPI peripheral on STM32 microcontrollers.
- struct [hal::Spi::Config](#)
SPI configuration struct.

Namespaces

- [hal](#)

8.12.1 Detailed Description

STM32 SPI HAL wrapper.

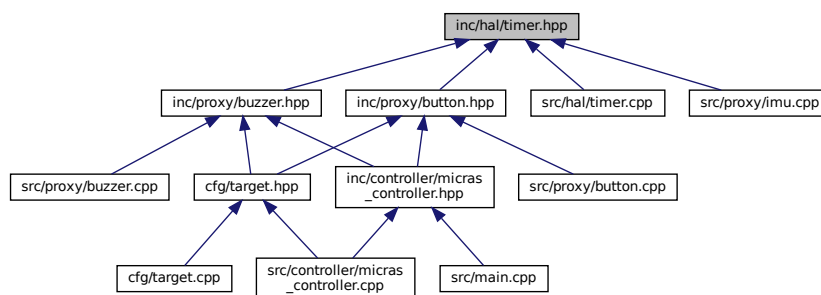
Date

03/2024

8.13 inc/hal/timer.hpp File Reference

STM32 Timer HAL wrapper.

This graph shows which files directly or indirectly include this file:



Classes

- class [hal::Timer](#)
Class to handle timer peripheral on STM32 microcontrollers.
- struct [hal::Timer::Config](#)
Timer configuration struct.

Namespaces

- [hal](#)

8.13.1 Detailed Description

STM32 Timer HAL wrapper.

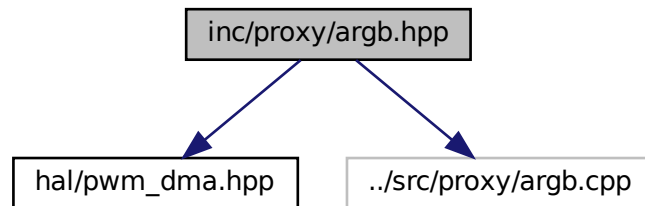
Date

03/2024

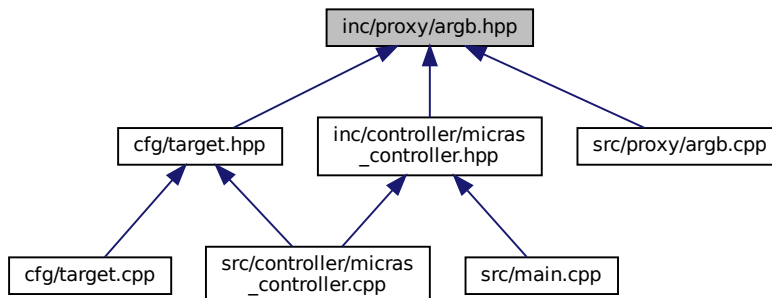
8.14 inc/proxy/argb.hpp File Reference

Proxy Argb class declaration.

```
#include "hal/pwm_dma.hpp"
#include "../src/proxy/argb.cpp"
Include dependency graph for argb.hpp:
```



This graph shows which files directly or indirectly include this file:



Classes

- class `proxy::Argb< num_of_leds >`
Class for controlling an addressable RGB LED.
- struct `proxy::Argb< num_of_leds >::Config`
Configuration structure for the addressable RGB LED.
- struct `proxy::Argb< num_of_leds >::Color`
Structure for storing color information.

Namespaces

- `proxy`

8.14.1 Detailed Description

Proxy Argb class declaration.

Date

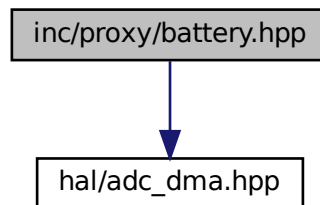
03/2024

8.15 inc/proxy/battery.hpp File Reference

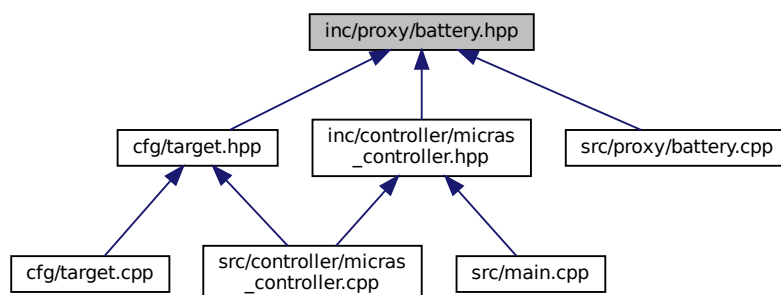
Proxy Battery class declaration.

```
#include "hal/adc_dma.hpp"
```

Include dependency graph for battery.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class [proxy::Battery](#)
Class for getting the battery voltage.
- struct [proxy::Battery::Config](#)
Configuration structure for the battery.

Namespaces

- [proxy](#)

8.15.1 Detailed Description

Proxy Battery class declaration.

Date

03/2024

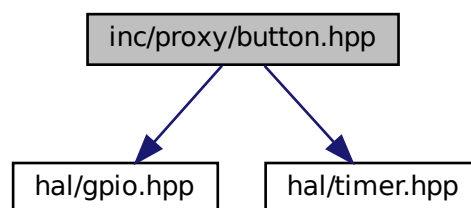
8.16 inc/proxy/button.hpp File Reference

Proxy Button class header.

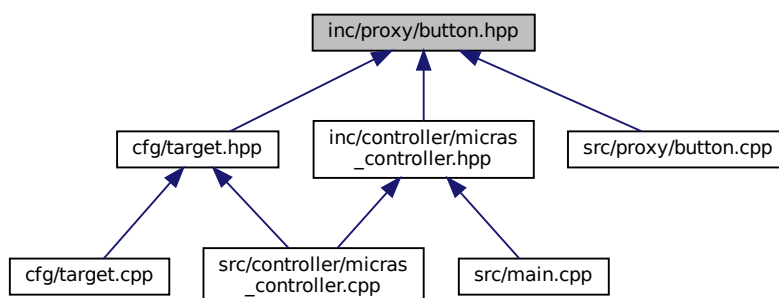
```
#include "hal/gpio.hpp"
```

```
#include "hal/timer.hpp"
```

Include dependency graph for button.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class `proxy::Button`
Class for controlling a button.
- struct `proxy::Button::Config`
Configuration structure for button.

Namespaces

- `proxy`

8.16.1 Detailed Description

Proxy Button class header.

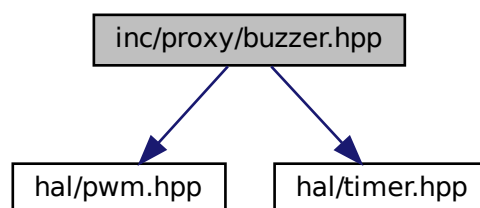
Date

03/2024

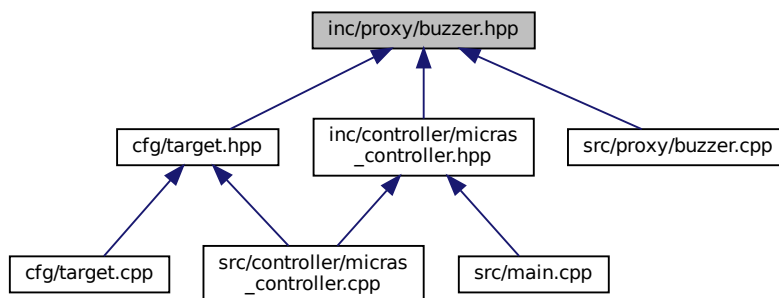
8.17 inc/proxy/buzzer.hpp File Reference

Proxy Buzzer class declaration.

```
#include "hal/pwm.hpp"  
#include "hal/timer.hpp"  
Include dependency graph for buzzer.hpp:
```



This graph shows which files directly or indirectly include this file:



Classes

- class `proxy::Buzzer`
Class for controlling a buzzer.
- struct `proxy::Buzzer::Config`
Configuration structure for the buzzer.

Namespaces

- `proxy`

8.17.1 Detailed Description

Proxy Buzzer class declaration.

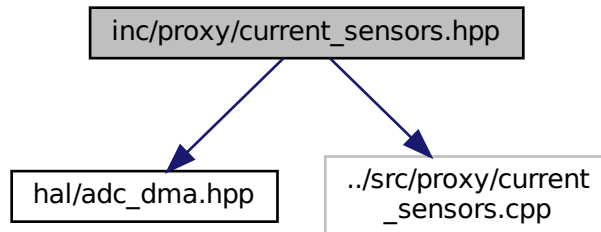
Date

03/2024

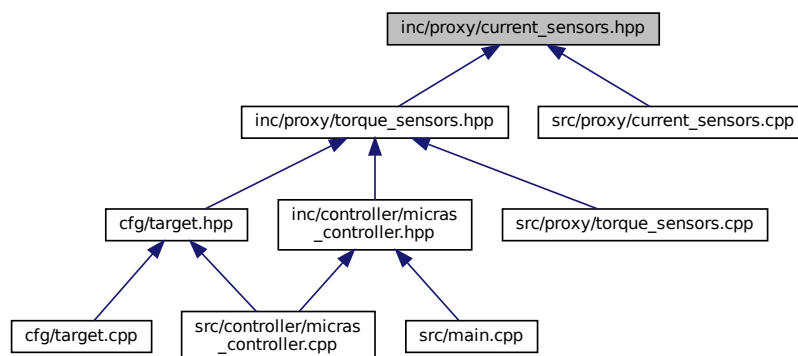
8.18 inc/proxy/current_sensors.hpp File Reference

Proxy CurrentSensors class header.

```
#include "hal/adc_dma.hpp"
#include "../src/proxy/current_sensors.cpp"
Include dependency graph for current_sensors.hpp:
```



This graph shows which files directly or indirectly include this file:



Classes

- class `proxy::CurrentSensors< num_of_sensors >`
Class for controlling `CurrentSensors`.
- struct `proxy::CurrentSensors< num_of_sensors >::Config`
Configuration structure for current sensors.

Namespaces

- `proxy`

8.18.1 Detailed Description

Proxy `CurrentSensors` class header.

Date

03/2024

8.19.1 Detailed Description

Proxy Dip Switch class header.

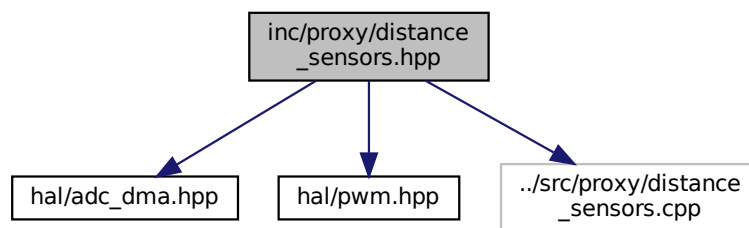
Date

03/2024

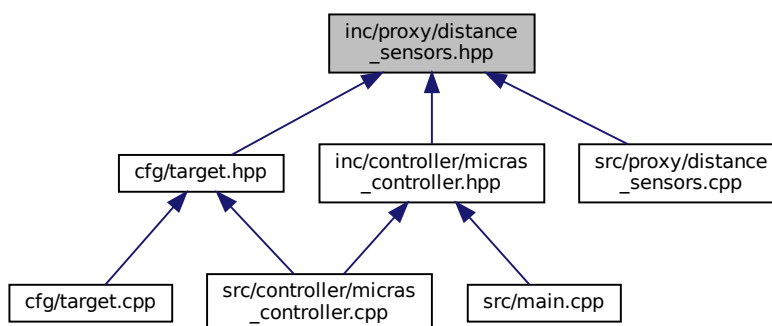
8.20 inc/proxy/distance_sensors.hpp File Reference

Proxy DistanceSensors class header.

```
#include "hal/adc_dma.hpp"
#include "hal/pwm.hpp"
#include "../src/proxy/distance_sensors.cpp"
Include dependency graph for distance_sensors.hpp:
```



This graph shows which files directly or indirectly include this file:



Classes

- class `proxy::DistanceSensors< num_of_sensors >`
Class for controlling *DistanceSensors*.
- struct `proxy::DistanceSensors< num_of_sensors >::Config`
Configuration structure for distance sensors.

Namespaces

- [proxy](#)

8.20.1 Detailed Description

Proxy DistanceSensors class header.

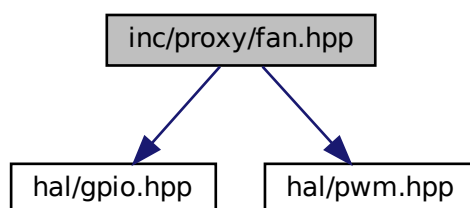
Date

03/2024

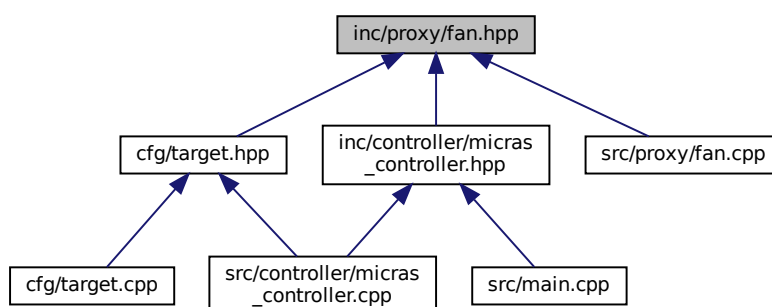
8.21 inc/proxy/fan.hpp File Reference

Proxy Fan class declaration.

```
#include "hal/gpio.hpp"
#include "hal/pwm.hpp"
Include dependency graph for fan.hpp:
```



This graph shows which files directly or indirectly include this file:



Classes

- class [proxy::Fan](#)
Class for controlling the fan driver.
- struct [proxy::Fan::Config](#)
Configuration structure for the fan.

Namespaces

- [proxy](#)

8.21.1 Detailed Description

Proxy Fan class declaration.

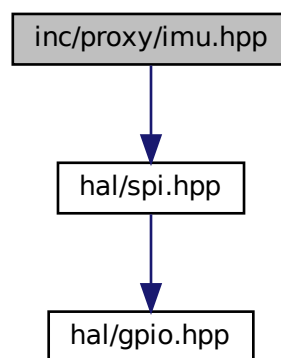
Date

03/2024

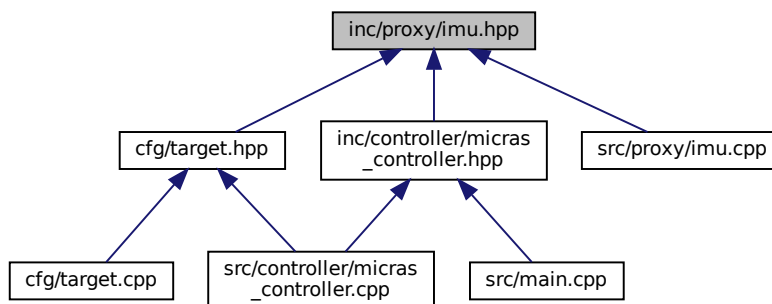
8.22 inc/proxy/imu.hpp File Reference

STM32 IMU HAL wrapper.

```
#include "hal/spi.hpp"
Include dependency graph for imu.hpp:
```



This graph shows which files directly or indirectly include this file:



Classes

- class `proxy::Imu`
Class to handle IMU peripheral on STM32 microcontrollers.
- struct `proxy::Imu::Config`
IMU configuration struct.

Namespaces

- `proxy`

8.22.1 Detailed Description

STM32 IMU HAL wrapper.

Date

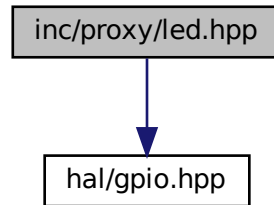
03/2024

8.23 inc/proxy/led.hpp File Reference

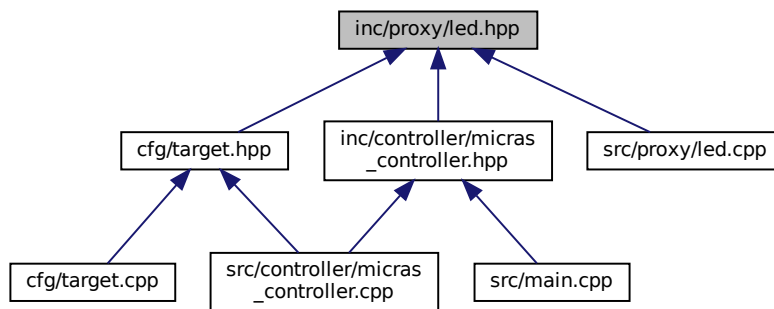
Proxy Led class header.

```
#include "hal/gpio.hpp"
```

Include dependency graph for led.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class [proxy::Led](#)
Class for controlling an LED.
- struct [proxy::Led::Config](#)
Configuration structure for LED.

Namespaces

- [proxy](#)

8.23.1 Detailed Description

Proxy Led class header.

Date

03/2024

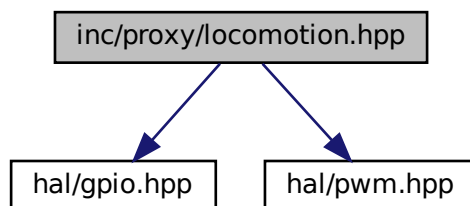
8.24 inc/proxy/locomotion.hpp File Reference

Proxy Locomotion class declaration.

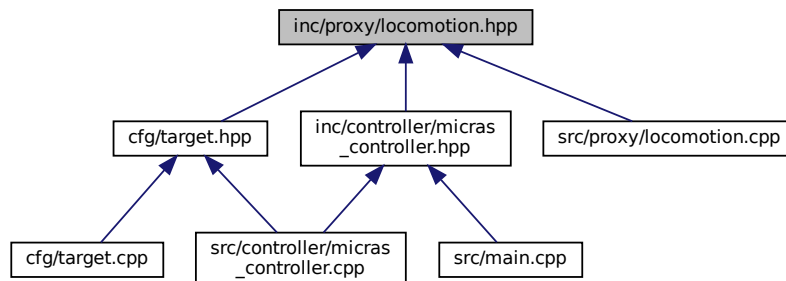
```
#include "hal/gpio.hpp"
```

```
#include "hal/pwm.hpp"
```

Include dependency graph for locomotion.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class [proxy::Locomotion](#)
Class for controlling the locomotion driver.
- struct [proxy::Locomotion::Config](#)
Configuration structure for the locomotion.

Namespaces

- [proxy](#)

8.24.1 Detailed Description

Proxy Locomotion class declaration.

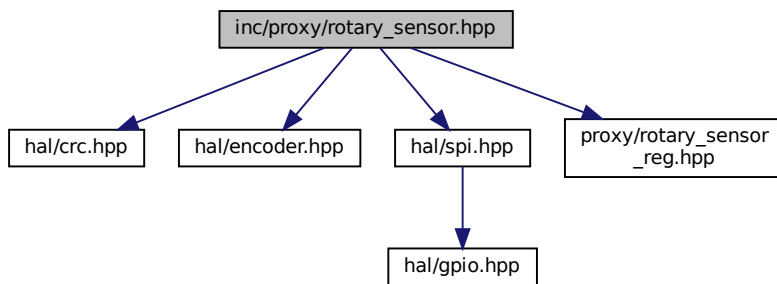
Date

03/2024

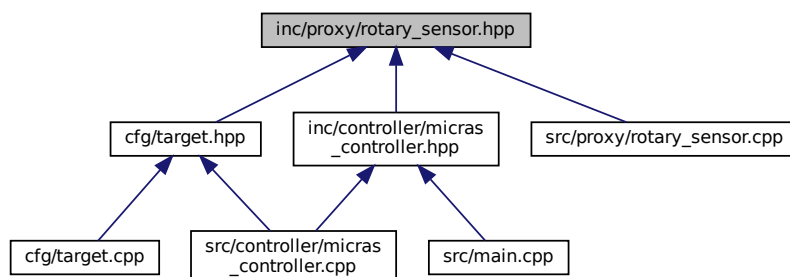
8.25 inc/proxy/rotary_sensor.hpp File Reference

STM32 rotary sensor HAL wrapper.

```
#include "hal/crc.hpp"
#include "hal/encoder.hpp"
#include "hal/spi.hpp"
#include "proxy/rotary_sensor_reg.hpp"
Include dependency graph for rotary_sensor.hpp:
```



This graph shows which files directly or indirectly include this file:



Classes

- class [proxy::RotarySensor](#)
Class to handle rotary sensor peripheral on STM32 microcontrollers.
- struct [proxy::RotarySensor::Config](#)
Rotary sensor configuration struct.
- struct [proxy::RotarySensor::CommandFrame::Fields](#)
- struct [proxy::RotarySensor::DataFrame::Fields](#)

Namespaces

- [proxy](#)

8.25.1 Detailed Description

STM32 rotary sensor HAL wrapper.

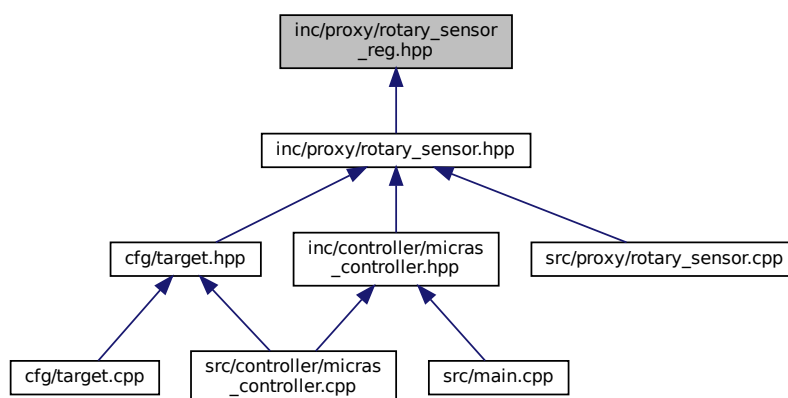
Date

03/2024

8.26 inc/proxy/rotary_sensor_reg.hpp File Reference

AS5047U rotary sensor registers definition.

This graph shows which files directly or indirectly include this file:



Classes

- struct [Registers](#)
[Registers](#) class for the rotary sensor configuration.
- union [Registers::Disable](#)
[Registers](#) union types definition.
- struct [Registers::Disable::Fields](#)
- union [Registers::Zposm](#)
- struct [Registers::Zposm::Fields](#)
- union [Registers::Zposl](#)
- struct [Registers::Zposl::Fields](#)
- union [Registers::Settings1](#)
- struct [Registers::Settings1::Fields](#)
- union [Registers::Settings2](#)
- struct [Registers::Settings2::Fields](#)
- union [Registers::Settings3](#)
- struct [Registers::Settings3::Fields](#)
- union [Registers::Ecc](#)
- struct [Registers::Ecc::Fields](#)

8.26.1 Detailed Description

AS5047U rotary sensor registers definition.

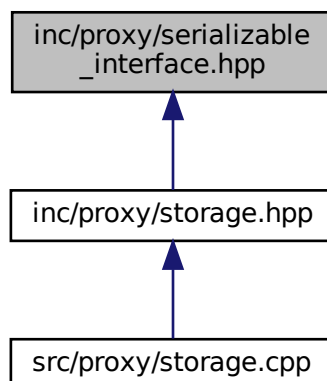
Date

03/2024

8.27 inc/proxy/serializable_interface.hpp File Reference

Serializable interface for all classes that need to be serialized.

This graph shows which files directly or indirectly include this file:



Classes

- class [ISerializable](#)
Interface class for serializable classes.

8.27.1 Detailed Description

Serializable interface for all classes that need to be serialized.

Date

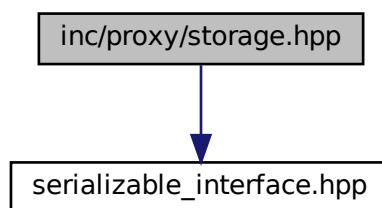
03/2024

8.28 inc/proxy/storage.hpp File Reference

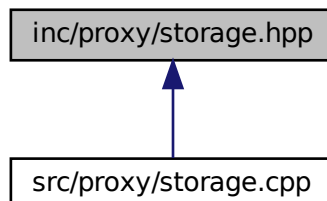
Proxy Storage class declaration.

```
#include "serializable_interface.hpp"
```

Include dependency graph for storage.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class `proxy::Storage`
Class for controlling the storage.
- struct `proxy::Storage::Config`
Configuration structure for the storage.

Namespaces

- `proxy`

Variables

- `template<typename T>`
`concept proxy::Fundamental = std::is_fundamental<T>::value`

8.28.1 Detailed Description

Proxy Storage class declaration.

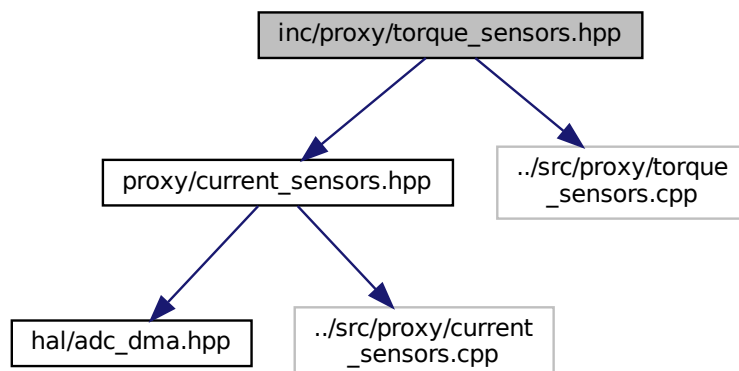
Date

03/2024

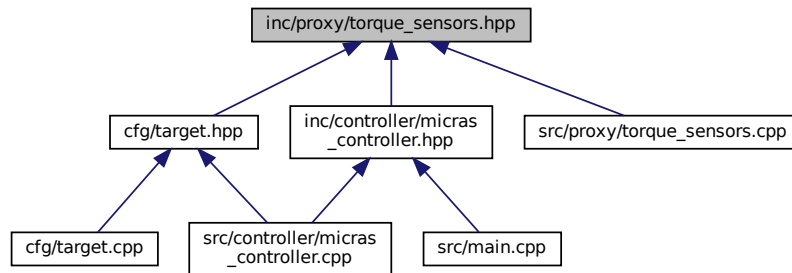
8.29 inc/proxy/torque_sensors.hpp File Reference

Proxy TorqueSensors class header.

```
#include "proxy/current_sensors.hpp"  
#include "../src/proxy/torque_sensors.cpp"  
Include dependency graph for torque_sensors.hpp:
```



This graph shows which files directly or indirectly include this file:



Classes

- class `proxy::TorqueSensors< num_of_sensors >`
Class for controlling `TorqueSensors`.
- struct `proxy::TorqueSensors< num_of_sensors >::Config`
Configuration structure for torque sensors.

Namespaces

- proxy

8.29.1 Detailed Description

Proxy TorqueSensors class header.

Date

03/2024

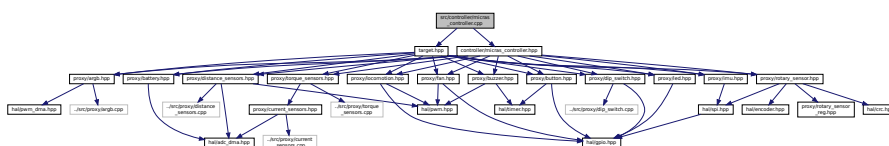
8.30 README.md File Reference

8.31 src/controller/micras_controller.cpp File Reference

Micras Controller class implementation.

```
#include "controller/micras_controller.hpp"
#include "target.hpp"
```

Include dependency graph for micras_controller.cpp:



8.31.1 Detailed Description

Micras Controller class implementation.

Date

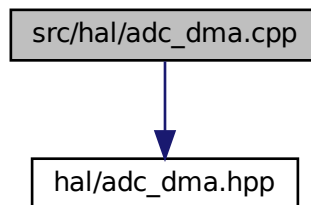
03/2024

8.32 src/hal/adc_dma.cpp File Reference

STM32 ADC DMA HAL wrapper.

```
#include "hal/adc_dma.hpp"
```

Include dependency graph for adc_dma.cpp:



Namespaces

- [hal](#)

8.32.1 Detailed Description

STM32 ADC DMA HAL wrapper.

Date

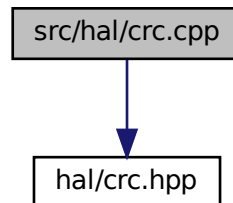
03/2024

8.33 src/hal/crc.cpp File Reference

STM32 CRC HAL wrapper.

```
#include "hal/crc.hpp"
```

Include dependency graph for crc.cpp:



Namespaces

- [hal](#)

8.33.1 Detailed Description

STM32 CRC HAL wrapper.

Date

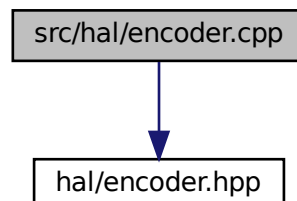
03/2024

8.34 src/hal/encoder.cpp File Reference

STM32 encoder HAL wrapper.

```
#include "hal/encoder.hpp"
```

Include dependency graph for encoder.cpp:



Namespaces

- [hal](#)

8.34.1 Detailed Description

STM32 encoder HAL wrapper.

Date

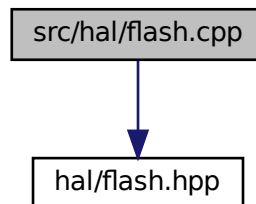
03/2024

8.35 src/hal/flash.cpp File Reference

STM32 flash HAL wrapper.

```
#include "hal/flash.hpp"
```

Include dependency graph for flash.cpp:



Namespaces

- [hal](#)

8.35.1 Detailed Description

STM32 flash HAL wrapper.

Date

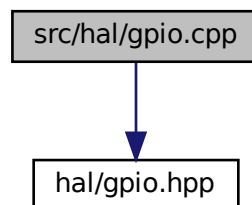
03/2024

8.36 src/hal/gpio.cpp File Reference

HAL GPIO class source.

```
#include "hal/gpio.hpp"
```

Include dependency graph for gpio.cpp:



Namespaces

- [hal](#)

8.36.1 Detailed Description

HAL GPIO class source.

Date

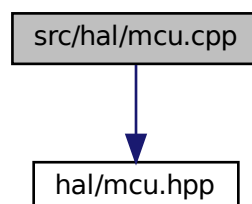
03/2024

8.37 src/hal/mcu.cpp File Reference

MCU related.

```
#include "hal/mcu.hpp"
```

Include dependency graph for mcu.cpp:



Namespaces

- [hal](#)

Functions

- void [SystemClock_Config](#) ()
Initializes System Clock.

8.37.1 Detailed Description

MCU related.

8.37.2 Function Documentation

8.37.2.1 SystemClock_Config()

```
void SystemClock_Config ( )
```

Initializes System Clock.

Note

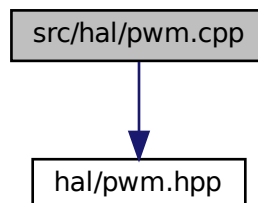
Defined by cube

8.38 src/hal/pwm.cpp File Reference

STM32 PWM HAL wrapper.

```
#include "hal/pwm.hpp"
```

Include dependency graph for pwm.cpp:



Namespaces

- [hal](#)

8.38.1 Detailed Description

STM32 PWM HAL wrapper.

Date

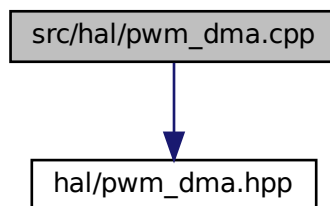
03/2024

8.39 src/hal/pwm_dma.cpp File Reference

STM32 PWM DMA HAL wrapper.

```
#include "hal/pwm_dma.hpp"
```

Include dependency graph for pwm_dma.cpp:



Namespaces

- [hal](#)

8.39.1 Detailed Description

STM32 PWM DMA HAL wrapper.

Date

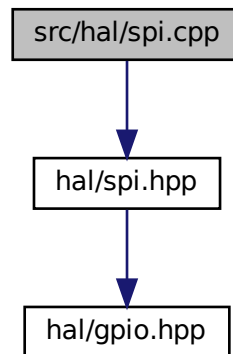
03/2024

8.40 src/hal/spi.cpp File Reference

Proxy SPI Switch class source.

```
#include "hal/spi.hpp"
```

Include dependency graph for spi.cpp:



Namespaces

- [hal](#)

8.40.1 Detailed Description

Proxy SPI Switch class source.

Date

03/2024

8.41 src/hal/timer.cpp File Reference

STM32 TIM HAL wrapper.

8.42.1 Detailed Description

Main function.

Date

03/2024

8.42.2 Function Documentation

8.42.2.1 main()

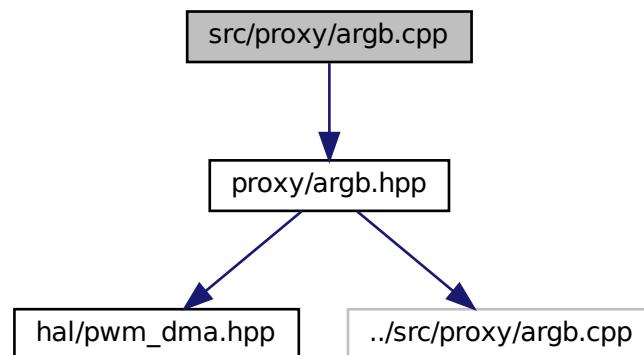
```
int main ( )
```

8.43 src/proxy/argb.cpp File Reference

Proxy Argb class implementation.

```
#include "proxy/argb.hpp"
```

Include dependency graph for argb.cpp:



Namespaces

- [proxy](#)

Macros

- `#define` [MICRAS_PROXY_ARGB_CPP](#)

8.43.1 Detailed Description

Proxy Argb class implementation.

Date

03/2024

8.43.2 Macro Definition Documentation

8.43.2.1 MICRAS_PROXY_ARGB_CPP

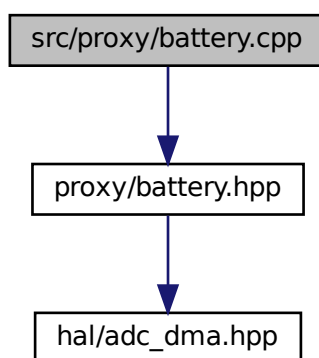
```
#define MICRAS_PROXY_ARGB_CPP
```

8.44 src/proxy/battery.cpp File Reference

Proxy Battery class implementation.

```
#include "proxy/battery.hpp"
```

Include dependency graph for battery.cpp:



Namespaces

- [proxy](#)

8.44.1 Detailed Description

Proxy Battery class implementation.

Date

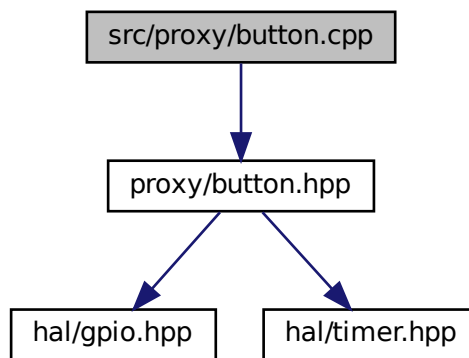
03/2024

8.45 src/proxy/button.cpp File Reference

Proxy Button class source.

```
#include "proxy/button.hpp"
```

Include dependency graph for button.cpp:



Namespaces

- [proxy](#)

8.45.1 Detailed Description

Proxy Button class source.

Date

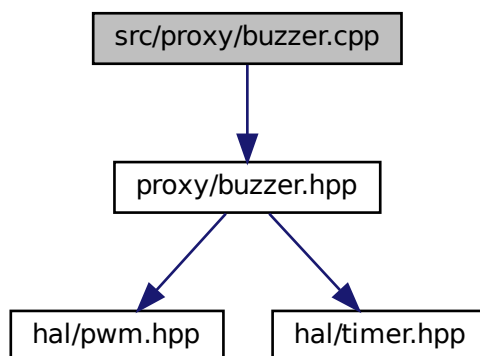
03/2024

8.46 src/proxy/buzzer.cpp File Reference

Proxy Buzzer class implementation.

```
#include "proxy/buzzer.hpp"
```

Include dependency graph for buzzer.cpp:



Namespaces

- [proxy](#)

8.46.1 Detailed Description

Proxy Buzzer class implementation.

Date

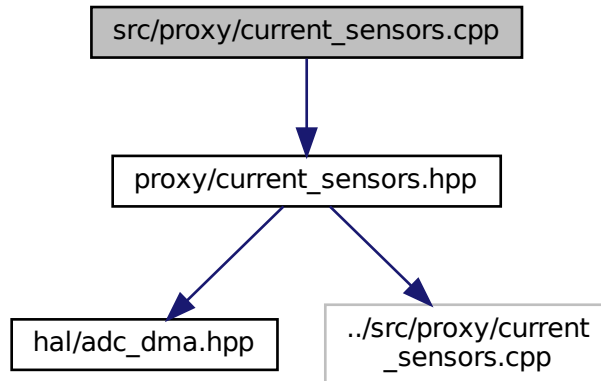
03/2024

8.47 src/proxy/current_sensors.cpp File Reference

Proxy CurrentSensors class implementation.

```
#include "proxy/current_sensors.hpp"
```

Include dependency graph for current_sensors.cpp:



Namespaces

- [proxy](#)

Macros

- `#define MICRAS_PROXY_CURRENT_SENSORS_CPP`

8.47.1 Detailed Description

Proxy CurrentSensors class implementation.

Date

03/2024

8.47.2 Macro Definition Documentation

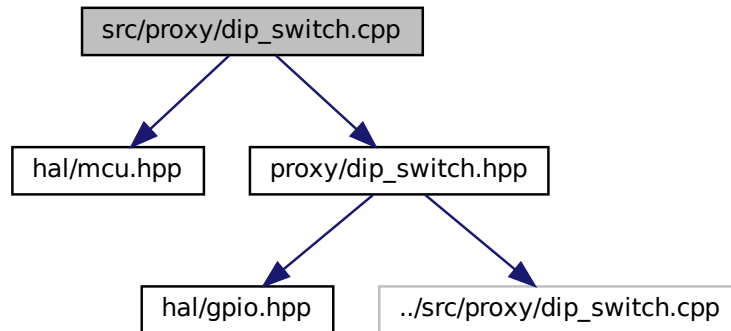
8.47.2.1 MICRAS_PROXY_CURRENT_SENSORS_CPP

```
#define MICRAS_PROXY_CURRENT_SENSORS_CPP
```

8.48 src/proxy/dip_switch.cpp File Reference

Proxy DIP Switch class source.

```
#include "hal/mcu.hpp"  
#include "proxy/dip_switch.hpp"  
Include dependency graph for dip_switch.cpp:
```



Namespaces

- [proxy](#)

Macros

- `#define` [MICRAS_PROXY_DIP_SWITCH_CPP](#)

8.48.1 Detailed Description

Proxy DIP Switch class source.

Date

03/2024

8.48.2 Macro Definition Documentation

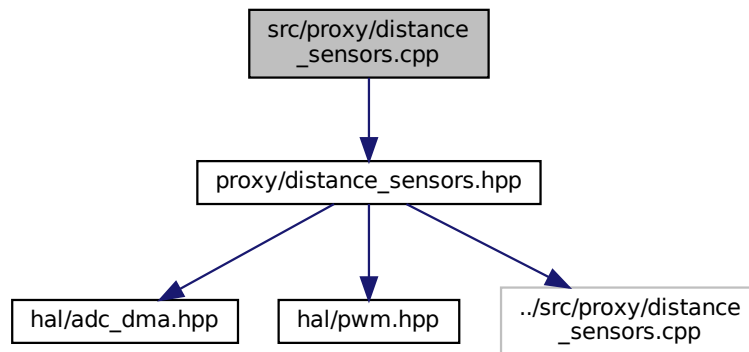
8.48.2.1 MICRAS_PROXY_DIP_SWITCH_CPP

```
#define MICRAS_PROXY_DIP_SWITCH_CPP
```

8.49 src/proxy/distance_sensors.cpp File Reference

```
#include "proxy/distance_sensors.hpp"
```

Include dependency graph for distance_sensors.cpp:



Namespaces

- [proxy](#)

Macros

- `#define MICRAS_PROXY_DISTANCE_SENSORS_CPP`

8.49.1 Macro Definition Documentation

8.49.1.1 MICRAS_PROXY_DISTANCE_SENSORS_CPP

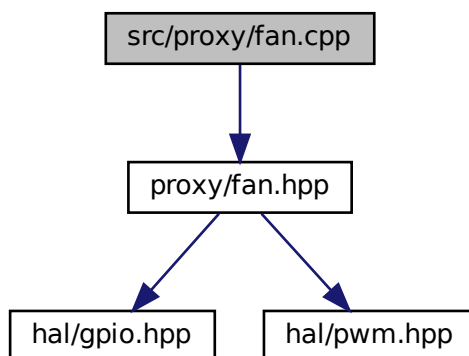
```
#define MICRAS_PROXY_DISTANCE_SENSORS_CPP
```

8.50 src/proxy/fan.cpp File Reference

Proxy Fan class source.

```
#include "proxy/fan.hpp"
```

Include dependency graph for fan.cpp:



Namespaces

- [proxy](#)

8.50.1 Detailed Description

Proxy Fan class source.

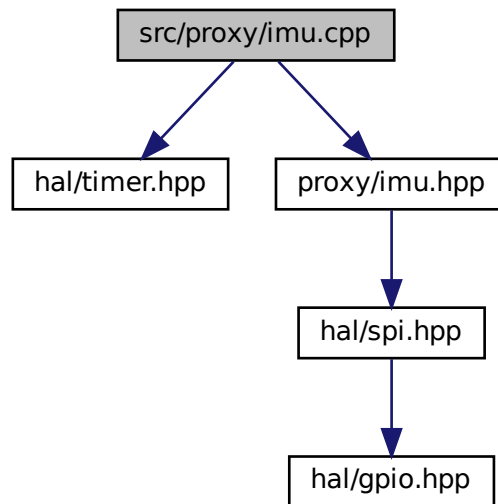
Date

03/2024

8.51 src/proxy/imu.cpp File Reference

Proxy Imu class source.

```
#include "hal/timer.hpp"  
#include "proxy/imu.hpp"  
Include dependency graph for imu.cpp:
```



Namespaces

- [proxy](#)

8.51.1 Detailed Description

Proxy Imu class source.

Date

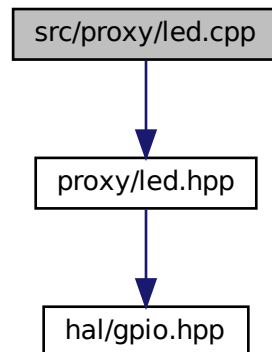
03/2024

8.52 `src/proxy/led.cpp` File Reference

Proxy Led class source.


```
#include "proxy/led.hpp"
```

Include dependency graph for led.cpp:



Namespaces

- [proxy](#)

8.52.1 Detailed Description

Proxy Led class source.

Date

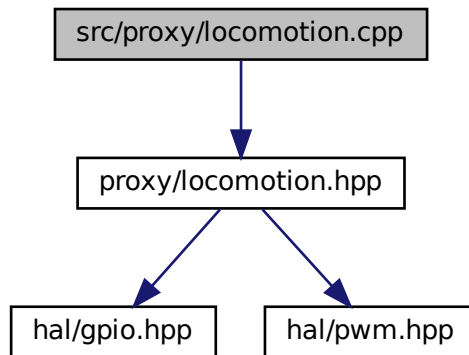
03/2024

8.53 src/proxy/locomotion.cpp File Reference

Proxy Locomotion class source.

```
#include "proxy/locomotion.hpp"
```

Include dependency graph for locomotion.cpp:



Namespaces

- [proxy](#)

8.53.1 Detailed Description

Proxy Locomotion class source.

Date

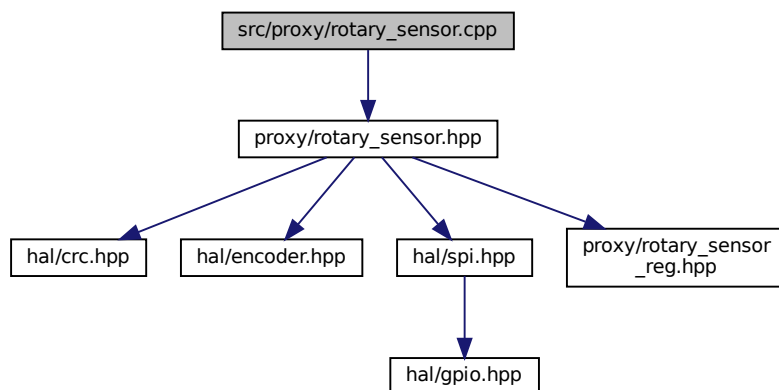
03/2024

8.54 `src/proxy/rotary_sensor.cpp` File Reference

Proxy RotarySensor class source.

```
#include "proxy/rotary_sensor.hpp"
```

Include dependency graph for rotary_sensor.cpp:



Namespaces

- [proxy](#)

8.54.1 Detailed Description

Proxy RotarySensor class source.

Date

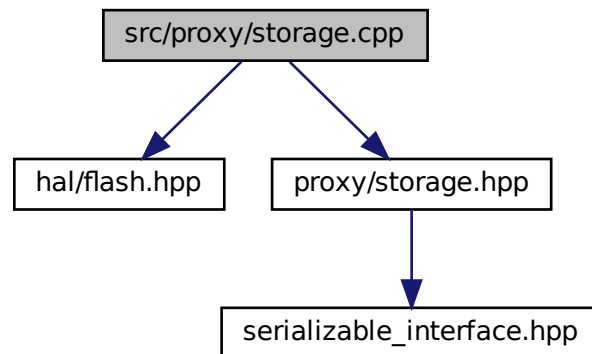
03/2024

8.55 src/proxy/storage.cpp File Reference

Proxy Storage class source.

```
#include "hal/flash.hpp"
#include "proxy/storage.hpp"
```

Include dependency graph for storage.cpp:



Namespaces

- [proxy](#)

8.55.1 Detailed Description

Proxy Storage class source.

Date

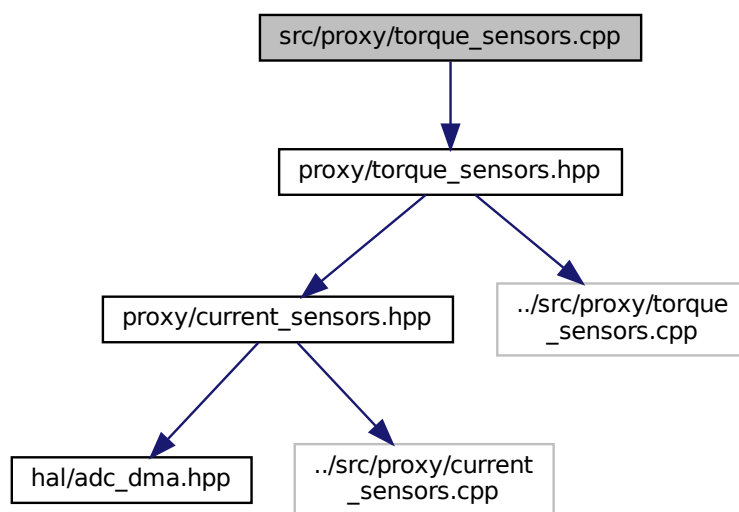
03/2024

8.56 `src/proxy/torque_sensors.cpp` File Reference

Proxy TorqueSensors class implementation.

```
#include "proxy/torque_sensors.hpp"
```

Include dependency graph for torque_sensors.cpp:



Namespaces

- [proxy](#)

Macros

- `#define` [MICRAS_PROXY_TORQUE_SENSORS_CPP](#)

8.56.1 Detailed Description

Proxy TorqueSensors class implementation.

Date

03/2024

8.56.2 Macro Definition Documentation

8.56.2.1 MICRAS_PROXY_TORQUE_SENSORS_CPP

```
#define MICRAS_PROXY_TORQUE_SENSORS_CPP
```


Index

- ~ISerializable
 - ISerializable, [99](#)
- ABI_DEC
 - Registers::Settings2::Fields, [85](#)
- ABI_off
 - Registers::Disable::Fields, [81](#)
- ABIRES
 - Registers::Settings3::Fields, [87](#)
- accelerometer_data_rate
 - proxy::Imu::Config, [53](#)
- accelerometer_filter
 - proxy::Imu::Config, [53](#)
- accelerometer_scale
 - proxy::Imu::Config, [53](#)
- adc
 - proxy::Battery::Config, [44](#)
 - proxy::CurrentSensors< num_of_sensors >::Config, [48](#)
 - proxy::DistanceSensors< num_of_sensors >::Config, [51](#)
- AdcDma
 - hal::AdcDma, [18](#)
- address
 - proxy::RotarySensor::CommandFrame::Fields, [78](#)
- Argb
 - proxy::Argb< num_of_leds >, [21](#)
- argb.cpp
 - MICRAS_PROXY_ARGB_CPP, [183](#)
- argb_config
 - target.cpp, [138](#)
 - target.hpp, [143](#)
- Axis
 - proxy::Imu, [96](#)
- BACKWARD
 - proxy::Fan, [76](#)
- Battery
 - proxy::Battery, [24](#)
- battery_config
 - target.cpp, [138](#)
 - target.hpp, [143](#)
- blue
 - proxy::Argb< num_of_leds >::Color, [30](#)
- Button
 - proxy::Button, [27](#)
- button_config
 - target.cpp, [138](#)
 - target.hpp, [143](#)
- Buzzer
 - proxy::Buzzer, [28](#)
- buzzer_config
 - target.cpp, [138](#)
 - target.hpp, [143](#)
- calculate
 - hal::Crc, [62](#)
- cfg/target.cpp, [137](#)
- cfg/target.hpp, [142](#)
- Crc
 - hal::Crc, [62](#)
- crc
 - proxy::RotarySensor::CommandFrame::Fields, [78](#)
 - proxy::RotarySensor::Config, [58](#)
 - proxy::RotarySensor::DataFrame::Fields, [80](#)
- create
 - proxy::Storage, [124](#), [125](#)
- current_sensors
 - proxy::TorqueSensors< num_of_sensors >::Config, [61](#)
- current_sensors.cpp
 - MICRAS_PROXY_CURRENT_SENSORS_CPP, [186](#)
- CurrentSensors
 - proxy::CurrentSensors< num_of_sensors >, [64](#)
- DAECDIS
 - Registers::Settings2::Fields, [85](#)
- data
 - proxy::RotarySensor::DataFrame::Fields, [80](#)
- Data_select
 - Registers::Settings2::Fields, [86](#)
- debounce_delay
 - proxy::Button::Config, [45](#)
- deserialize
 - ISerializable, [99](#)
- Dia1_en
 - Registers::Zposl::Fields, [88](#)
- Dia2_en
 - Registers::Zposl::Fields, [89](#)
- Dia3_en
 - Registers::Settings1::Fields, [84](#)
- Dia4_en
 - Registers::Settings1::Fields, [84](#)
- dip_switch.cpp
 - MICRAS_PROXY_DIP_SWITCH_CPP, [187](#)
- dip_switch_config
 - target.cpp, [138](#)
 - target.hpp, [144](#)
- DipSwitch

- proxy::DipSwitch< num_of_switches >, 66
- DIR
 - Registers::Settings2::Fields, 86
- direction_gpio
 - proxy::Fan::Config, 52
- disable
 - proxy::Fan, 76
 - proxy::Locomotion, 103
 - Registers, 113
- disable_addr
 - Registers, 113
- distance_sensors.cpp
 - MICRAS_PROXY_DISTANCE_SENSORS_CPP, 188
- distance_sensors_config
 - target.cpp, 139
 - target.hpp, 144
- DistanceSensors
 - proxy::DistanceSensors< num_of_sensors >, 70
- do_not_care
 - proxy::RotarySensor::CommandFrame::Fields, 78
- ecc
 - Registers, 113
- ecc_addr
 - Registers, 113
- ECC_chsum
 - Registers::Ecc::Fields, 82
- ECC_en
 - Registers::Ecc::Fields, 83
- elapsed_time_ms
 - hal::Timer, 128
- elapsed_time_us
 - hal::Timer, 128
- enable
 - proxy::Fan, 76
 - proxy::Locomotion, 104
- enable_gpio
 - proxy::Fan::Config, 52
 - proxy::Locomotion::Config, 56
- Encoder
 - hal::Encoder, 74
- encoder
 - proxy::RotarySensor::Config, 58
- erase_pages
 - hal::Flash, 91
- error
 - proxy::RotarySensor::DataFrame::Fields, 80
- EXTRA_LONG_PRESS
 - proxy::Button, 27
- extra_long_press_delay
 - proxy::Button::Config, 45
- Fan
 - proxy::Fan, 76
- fan_config
 - target.cpp, 139
 - target.hpp, 144
- Fields
 - Registers::Zposl, 133
- fields
 - Registers::Disable, 69
 - Registers::Ecc, 73
 - Registers::Settings1, 117
 - Registers::Settings2, 119
 - Registers::Settings3, 120
 - Registers::Zposm, 135
- FILTER_disable
 - Registers::Disable::Fields, 81
- Flash
 - hal::Flash, 91
- FORWARD
 - proxy::Fan, 76
- Fundamental
 - proxy, 16
- get_angular_velocity
 - proxy::Imu, 96
- get_compare
 - hal::PwmDma, 111
- get_counter
 - hal::Encoder, 74
- get_current
 - proxy::CurrentSensors< num_of_sensors >, 64
 - proxy::TorqueSensors< num_of_sensors >, 131
- get_current_raw
 - proxy::CurrentSensors< num_of_sensors >, 64
- get_distance
 - proxy::DistanceSensors< num_of_sensors >, 71
- get_distance_raw
 - proxy::DistanceSensors< num_of_sensors >, 71
- get_linear_acceleration
 - proxy::Imu, 97
- get_orientation
 - proxy::Imu, 97
- get_position
 - proxy::RotarySensor, 116
- get_status
 - proxy::Button, 27
- get_switch_state
 - proxy::DipSwitch< num_of_switches >, 67
- get_switches_value
 - proxy::DipSwitch< num_of_switches >, 67
- get_torque
 - proxy::TorqueSensors< num_of_sensors >, 132
- get_voltage
 - proxy::Battery, 25
- get_voltage_raw
 - proxy::Battery, 25
- Gpio
 - hal::Gpio, 94
- gpio
 - hal::Spi::Config, 40
 - proxy::Button::Config, 46
 - proxy::Led::Config, 55
- gpio_array
 - proxy::DipSwitch< num_of_switches >::Config, 50
- green

- proxy::Argb< num_of_leds >::Color, 31
- gyroscope_data_rate
 - proxy::Imu::Config, 53
- gyroscope_filter
 - proxy::Imu::Config, 53
- gyroscope_scale
 - proxy::Imu::Config, 54
- hal, 15
- hal::AdcDma, 17
 - AdcDma, 18
 - max_reading, 19
 - reference_voltage, 19
 - start_dma, 18
 - stop_dma, 19
- hal::AdcDma::Config, 31
 - handle, 32
 - init_function, 32
 - max_reading, 32
 - reference_voltage, 32
- hal::Crc, 61
 - calculate, 62
 - Crc, 62
- hal::Crc::Config, 33
 - handle, 33
- hal::Encoder, 73
 - Encoder, 74
 - get_counter, 74
- hal::Encoder::Config, 34
 - handle, 34
 - init_function, 34
 - timer_channel, 35
- hal::Flash, 90
 - erase_pages, 91
 - Flash, 91
 - read, 91, 92
 - write, 92
- hal::Gpio, 93
 - Gpio, 94
 - read, 94
 - toggle, 94
 - write, 94
- hal::Gpio::Config, 35
 - pin, 36
 - port, 36
- hal::Mcu, 105
 - init, 106
 - Mcu, 106
- hal::Pwm, 108
 - Pwm, 108
 - set_duty_cycle, 109
 - set_frequency, 109
- hal::Pwm::Config, 36
 - handle, 37
 - init_function, 37
 - timer_channel, 37
- hal::PwmDma, 109
 - get_compare, 111
 - PwmDma, 110
 - start_dma, 111
 - stop_dma, 111
- hal::PwmDma::Config, 38
 - handle, 38
 - init_function, 38
 - timer_channel, 39
- hal::Spi, 121
 - receive, 122
 - select_device, 122
 - Spi, 122
 - transmit, 122
 - unselect_device, 123
- hal::Spi::Config, 39
 - gpio, 40
 - handle, 40
 - init_function, 40
 - timeout, 40
- hal::Timer, 126
 - elapsed_time_ms, 128
 - elapsed_time_us, 128
 - reset_ms, 128
 - reset_us, 129
 - sleep_ms, 129
 - sleep_us, 129
 - Timer, 128
- hal::Timer::Config, 41
 - handle, 41
 - init_function, 41
- handle
 - hal::AdcDma::Config, 32
 - hal::Crc::Config, 33
 - hal::Encoder::Config, 34
 - hal::Pwm::Config, 37
 - hal::PwmDma::Config, 38
 - hal::Spi::Config, 40
 - hal::Timer::Config, 41
- HYS
 - Registers::Settings3::Fields, 87
- Imu
 - proxy::Imu, 96
- imu_config
 - target.cpp, 139
 - target.hpp, 144
- inc/controller/micras_controller.hpp, 145
- inc/hal/adc_dma.hpp, 146
- inc/hal/crc.hpp, 147
- inc/hal/encoder.hpp, 148
- inc/hal/flash.hpp, 149
- inc/hal/gpio.hpp, 149
- inc/hal/mcu.hpp, 150
- inc/hal/pwm.hpp, 151
- inc/hal/pwm_dma.hpp, 152
- inc/hal/spi.hpp, 153
- inc/hal/timer.hpp, 154
- inc/proxy/argb.hpp, 155
- inc/proxy/battery.hpp, 156
- inc/proxy/button.hpp, 157
- inc/proxy/buzzer.hpp, 158

- inc/proxy/current_sensors.hpp, 159
- inc/proxy/dip_switch.hpp, 161
- inc/proxy/distance_sensors.hpp, 162
- inc/proxy/fan.hpp, 163
- inc/proxy/imu.hpp, 164
- inc/proxy/led.hpp, 165
- inc/proxy/locomotion.hpp, 167
- inc/proxy/rotary_sensor.hpp, 168
- inc/proxy/rotary_sensor_reg.hpp, 169
- inc/proxy/serializable_interface.hpp, 170
- inc/proxy/storage.hpp, 171
- inc/proxy/torque_sensors.hpp, 172
- init
 - hal::Mcu, 106
- init_function
 - hal::AdcDma::Config, 32
 - hal::Encoder::Config, 34
 - hal::Pwm::Config, 37
 - hal::PwmDma::Config, 38
 - hal::Spi::Config, 40
 - hal::Timer::Config, 41
- is_pressed
 - proxy::Button, 27
- ISerializable, 98
 - ~ISerializable, 99
 - deserialize, 99
 - ISerializable, 99
 - operator=, 100
 - serialize, 100
- IWIDTH
 - Registers::Settings2::Fields, 86
- K_max
 - Registers::Settings1::Fields, 84
- K_min
 - Registers::Settings1::Fields, 84
- Led
 - proxy::Led, 101
- led_config
 - target.cpp, 140
 - target.hpp, 144
- led_pwm
 - proxy::DistanceSensors< num_of_sensors >::Config, 51
- Locomotion
 - proxy::Locomotion, 103
- locomotion_config
 - target.cpp, 140
 - target.hpp, 144
- LONG_PRESS
 - proxy::Button, 27
- long_press_delay
 - proxy::Button::Config, 46
- main
 - main.cpp, 182
- main.cpp
 - main, 182
- max_distance
 - proxy::DistanceSensors< num_of_sensors >::Config, 51
- max_reading
 - hal::AdcDma, 19
 - hal::AdcDma::Config, 32
- max_torque
 - proxy::TorqueSensors< num_of_sensors >::Config, 61
- Mcu
 - hal::Mcu, 106
- mcu.cpp
 - SystemClock_Config, 178
- MICRAS_PROXY_ARGB_CPP
 - argb.cpp, 183
- MICRAS_PROXY_CURRENT_SENSORS_CPP
 - current_sensors.cpp, 186
- MICRAS_PROXY_DIP_SWITCH_CPP
 - dip_switch.cpp, 187
- MICRAS_PROXY_DISTANCE_SENSORS_CPP
 - distance_sensors.cpp, 188
- MICRAS_PROXY_TORQUE_SENSORS_CPP
 - torque_sensors.cpp, 195
- MicrasController, 106
 - MicrasController, 107
 - run, 107
- na
 - Registers::Disable::Fields, 81
- NO_PRESS
 - proxy::Button, 27
- NOISESET
 - Registers::Settings2::Fields, 86
- number_of_pages
 - proxy::Storage::Config, 59
- operator=
 - ISerializable, 100
- orientation_data_rate
 - proxy::Imu::Config, 54
- pin
 - hal::Gpio::Config, 36
- play
 - proxy::Buzzer, 29
- port
 - hal::Gpio::Config, 36
- proxy, 15
 - Fundamental, 16
- proxy::Argb< num_of_leds >, 20
 - Argb, 21
 - set_color, 21, 22
 - turn_off, 22
- proxy::Argb< num_of_leds >::Color, 30
 - blue, 30
 - green, 31
 - red, 31
- proxy::Argb< num_of_leds >::Config, 42
 - pwm, 43

- proxy::Battery, 23
 - Battery, 24
 - get_voltage, 25
 - get_voltage_raw, 25
- proxy::Battery::Config, 43
 - adc, 44
 - voltage_divider, 44
- proxy::Button, 25
 - Button, 27
 - EXTRA_LONG_PRESS, 27
 - get_status, 27
 - is_pressed, 27
 - LONG_PRESS, 27
 - NO_PRESS, 27
 - PULL_DOWN, 26
 - PULL_UP, 26
 - PullResistor, 26
 - SHORT_PRESS, 27
 - Status, 26
- proxy::Button::Config, 44
 - debounce_delay, 45
 - extra_long_press_delay, 45
 - gpio, 46
 - long_press_delay, 46
 - pull_resistor, 46
- proxy::Buzzer, 28
 - Buzzer, 28
 - play, 29
 - stop, 29
 - update, 29
- proxy::Buzzer::Config, 46
 - pwm, 47
- proxy::CurrentSensors< num_of_sensors >, 63
 - CurrentSensors, 64
 - get_current, 64
 - get_current_raw, 64
- proxy::CurrentSensors< num_of_sensors >::Config, 48
 - adc, 48
 - shunt_resistor, 48
- proxy::DipSwitch< num_of_switches >, 65
 - DipSwitch, 66
 - get_switch_state, 67
 - get_switches_value, 67
- proxy::DipSwitch< num_of_switches >::Config, 49
 - gpio_array, 50
- proxy::DistanceSensors< num_of_sensors >, 69
 - DistanceSensors, 70
 - get_distance, 71
 - get_distance_raw, 71
 - set_led_intensity, 71
- proxy::DistanceSensors< num_of_sensors >::Config, 50
 - adc, 51
 - led_pwm, 51
 - max_distance, 51
- proxy::Fan, 75
 - BACKWARD, 76
 - disable, 76
 - enable, 76
 - Fan, 76
 - FORWARD, 76
 - RotationDirection, 76
 - set_speed, 77
 - stop, 77
- proxy::Fan::Config, 51
 - direction_gpio, 52
 - enable_gpio, 52
 - pwm, 52
- proxy::Imu, 95
 - Axis, 96
 - get_angular_velocity, 96
 - get_linear_acceleration, 97
 - get_orientation, 97
 - Imu, 96
 - update_data, 97
 - W, 96
 - X, 96
 - Y, 96
 - Z, 96
- proxy::Imu::Config, 52
 - accelerometer_data_rate, 53
 - accelerometer_filter, 53
 - accelerometer_scale, 53
 - gyroscope_data_rate, 53
 - gyroscope_filter, 53
 - gyroscope_scale, 54
 - orientation_data_rate, 54
 - spi, 54
- proxy::Led, 100
 - Led, 101
 - toggle, 101
 - turn_off, 101
 - turn_on, 102
- proxy::Led::Config, 54
 - gpio, 55
- proxy::Locomotion, 102
 - disable, 103
 - enable, 104
 - Locomotion, 103
 - set_speed, 104
 - set_wheel_speed, 104
 - stop, 104
 - stop_left, 104
 - stop_right, 105
- proxy::Locomotion::Config, 56
 - enable_gpio, 56
 - pwm_left_bwd, 56
 - pwm_left_fwd, 57
 - pwm_right_bwd, 57
 - pwm_right_fwd, 57
- proxy::RotarySensor, 115
 - get_position, 116
 - RotarySensor, 116
- proxy::RotarySensor::CommandFrame::Fields, 77
 - address, 78
 - crc, 78

- do_not_care, [78](#)
- rw, [79](#)
- proxy::RotarySensor::Config, [57](#)
 - crc, [58](#)
 - encoder, [58](#)
 - registers, [58](#)
 - resolution, [58](#)
 - spi, [58](#)
- proxy::RotarySensor::DataFrame::Fields, [79](#)
 - crc, [80](#)
 - data, [80](#)
 - error, [80](#)
 - warning, [80](#)
- proxy::Storage, [123](#)
 - create, [124](#), [125](#)
 - save, [125](#)
 - Storage, [124](#)
 - sync, [125](#), [126](#)
- proxy::Storage::Config, [59](#)
 - number_of_pages, [59](#)
 - start_page, [59](#)
- proxy::TorqueSensors< num_of_sensors >, [130](#)
 - get_current, [131](#)
 - get_torque, [132](#)
 - TorqueSensors, [131](#)
- proxy::TorqueSensors< num_of_sensors >::Config, [60](#)
 - current_sensors, [61](#)
 - max_torque, [61](#)
- PULL_DOWN
 - proxy::Button, [26](#)
- pull_resistor
 - proxy::Button::Config, [46](#)
- PULL_UP
 - proxy::Button, [26](#)
- PullResistor
 - proxy::Button, [26](#)
- Pwm
 - hal::Pwm, [108](#)
- pwm
 - proxy::Argb< num_of_leds >::Config, [43](#)
 - proxy::Buzzer::Config, [47](#)
 - proxy::Fan::Config, [52](#)
- pwm_left_bwd
 - proxy::Locomotion::Config, [56](#)
- pwm_left_fwd
 - proxy::Locomotion::Config, [57](#)
- pwm_right_bwd
 - proxy::Locomotion::Config, [57](#)
- pwm_right_fwd
 - proxy::Locomotion::Config, [57](#)
- PwmDma
 - hal::PwmDma, [110](#)
- PWMon
 - Registers::Settings2::Fields, [86](#)
- raw
 - Registers::Disable, [69](#)
 - Registers::Ecc, [73](#)
 - Registers::Settings1, [118](#)
 - Registers::Settings2, [119](#)
 - Registers::Settings3, [120](#)
 - Registers::Zposl, [133](#)
 - Registers::Zposm, [135](#)
- read
 - hal::Flash, [91](#), [92](#)
 - hal::Gpio, [94](#)
- README.md, [173](#)
- receive
 - hal::Spi, [122](#)
- red
 - proxy::Argb< num_of_leds >::Color, [31](#)
- reference_voltage
 - hal::AdcDma, [19](#)
 - hal::AdcDma::Config, [32](#)
- Registers, [112](#)
 - disable, [113](#)
 - disable_addr, [113](#)
 - ecc, [113](#)
 - ecc_addr, [113](#)
 - settings1, [113](#)
 - settings1_addr, [114](#)
 - settings2, [114](#)
 - settings2_addr, [114](#)
 - settings3, [114](#)
 - settings3_addr, [114](#)
 - zposl, [114](#)
 - zposl_addr, [114](#)
 - zposm, [114](#)
 - zposm_addr, [115](#)
- registers
 - proxy::RotarySensor::Config, [58](#)
- Registers::Disable, [68](#)
 - fields, [69](#)
 - raw, [69](#)
- Registers::Disable::Fields, [80](#)
 - ABI_off, [81](#)
 - FILTER_disable, [81](#)
 - na, [81](#)
 - UVW_off, [82](#)
- Registers::Ecc, [72](#)
 - fields, [73](#)
 - raw, [73](#)
- Registers::Ecc::Fields, [82](#)
 - ECC_chsum, [82](#)
 - ECC_en, [83](#)
- Registers::Settings1, [117](#)
 - fields, [117](#)
 - raw, [118](#)
- Registers::Settings1::Fields, [83](#)
 - Dia3_en, [84](#)
 - Dia4_en, [84](#)
 - K_max, [84](#)
 - K_min, [84](#)
- Registers::Settings2, [118](#)
 - fields, [119](#)
 - raw, [119](#)
- Registers::Settings2::Fields, [84](#)

- ABI_DEC, [85](#)
- DAECDIS, [85](#)
- Data_select, [86](#)
- DIR, [86](#)
- IWIDTH, [86](#)
- NOISESET, [86](#)
- PWMon, [86](#)
- UVW_ABI, [86](#)
- Registers::Settings3, [119](#)
 - fields, [120](#)
 - raw, [120](#)
- Registers::Settings3::Fields, [87](#)
 - ABIRES, [87](#)
 - HYS, [87](#)
 - UVWPP, [87](#)
- Registers::Zposl, [132](#)
 - Fields, [133](#)
 - raw, [133](#)
- Registers::Zposl::Fields, [88](#)
 - Dia1_en, [88](#)
 - Dia2_en, [89](#)
 - ZPOSL, [89](#)
- Registers::Zposm, [134](#)
 - fields, [135](#)
 - raw, [135](#)
- Registers::Zposm::Fields, [89](#)
 - ZPOSM, [90](#)
- reset_ms
 - hal::Timer, [128](#)
- reset_us
 - hal::Timer, [129](#)
- resolution
 - proxy::RotarySensor::Config, [58](#)
- rotary_sensor_left_config
 - target.cpp, [140](#)
 - target.hpp, [144](#)
- rotary_sensor_reg_config
 - target.cpp, [141](#)
- rotary_sensor_right_config
 - target.cpp, [141](#)
 - target.hpp, [144](#)
- RotarySensor
 - proxy::RotarySensor, [116](#)
- RotationDirection
 - proxy::Fan, [76](#)
- run
 - MicrasController, [107](#)
- rw
 - proxy::RotarySensor::CommandFrame::Fields, [79](#)
- save
 - proxy::Storage, [125](#)
- select_device
 - hal::Spi, [122](#)
- serialize
 - ISerializable, [100](#)
- set_color
 - proxy::Argb< num_of_leds >, [21](#), [22](#)
- set_duty_cycle
 - hal::Pwm, [109](#)
- set_frequency
 - hal::Pwm, [109](#)
- set_led_intensity
 - proxy::DistanceSensors< num_of_sensors >, [71](#)
- set_speed
 - proxy::Fan, [77](#)
 - proxy::Locomotion, [104](#)
- set_wheel_speed
 - proxy::Locomotion, [104](#)
- settings1
 - Registers, [113](#)
- settings1_addr
 - Registers, [114](#)
- settings2
 - Registers, [114](#)
- settings2_addr
 - Registers, [114](#)
- settings3
 - Registers, [114](#)
- settings3_addr
 - Registers, [114](#)
- SHORT_PRESS
 - proxy::Button, [27](#)
- shunt_resistor
 - proxy::CurrentSensors< num_of_sensors >::Config, [48](#)
- sleep_ms
 - hal::Timer, [129](#)
- sleep_us
 - hal::Timer, [129](#)
- Spi
 - hal::Spi, [122](#)
- spi
 - proxy::Imu::Config, [54](#)
 - proxy::RotarySensor::Config, [58](#)
- src/controller/micras_controller.cpp, [173](#)
- src/hal/adc_dma.cpp, [174](#)
- src/hal/crc.cpp, [175](#)
- src/hal/encoder.cpp, [175](#)
- src/hal/flash.cpp, [176](#)
- src/hal/gpio.cpp, [177](#)
- src/hal/mcu.cpp, [177](#)
- src/hal/pwm.cpp, [178](#)
- src/hal/pwm_dma.cpp, [179](#)
- src/hal/spi.cpp, [180](#)
- src/hal/timer.cpp, [180](#)
- src/main.cpp, [181](#)
- src/proxy/argb.cpp, [182](#)
- src/proxy/battery.cpp, [183](#)
- src/proxy/button.cpp, [184](#)
- src/proxy/buzzer.cpp, [185](#)
- src/proxy/current_sensors.cpp, [185](#)
- src/proxy/dip_switch.cpp, [187](#)
- src/proxy/distance_sensors.cpp, [188](#)
- src/proxy/fan.cpp, [189](#)
- src/proxy/imu.cpp, [189](#)
- src/proxy/led.cpp, [190](#)

- src/proxy/locomotion.cpp, 191
- src/proxy/rotary_sensor.cpp, 192
- src/proxy/storage.cpp, 193
- src/proxy/torque_sensors.cpp, 194
- start_dma
 - hal::AdcDma, 18
 - hal::PwmDma, 111
- start_page
 - proxy::Storage::Config, 59
- Status
 - proxy::Button, 26
- stop
 - proxy::Buzzer, 29
 - proxy::Fan, 77
 - proxy::Locomotion, 104
- stop_dma
 - hal::AdcDma, 19
 - hal::PwmDma, 111
- stop_left
 - proxy::Locomotion, 104
- stop_right
 - proxy::Locomotion, 105
- Storage
 - proxy::Storage, 124
- sync
 - proxy::Storage, 125, 126
- SystemClock_Config
 - mcu.cpp, 178
- target.cpp
 - argb_config, 138
 - battery_config, 138
 - button_config, 138
 - buzzer_config, 138
 - dip_switch_config, 138
 - distance_sensors_config, 139
 - fan_config, 139
 - imu_config, 139
 - led_config, 140
 - locomotion_config, 140
 - rotary_sensor_left_config, 140
 - rotary_sensor_reg_config, 141
 - rotary_sensor_right_config, 141
 - torque_sensors_config, 141
- target.hpp
 - argb_config, 143
 - battery_config, 143
 - button_config, 143
 - buzzer_config, 143
 - dip_switch_config, 144
 - distance_sensors_config, 144
 - fan_config, 144
 - imu_config, 144
 - led_config, 144
 - locomotion_config, 144
 - rotary_sensor_left_config, 144
 - rotary_sensor_right_config, 144
 - torque_sensors_config, 145
- timeout
 - hal::Spi::Config, 40
- Timer
 - hal::Timer, 128
- timer_channel
 - hal::Encoder::Config, 35
 - hal::Pwm::Config, 37
 - hal::PwmDma::Config, 39
- toggle
 - hal::Gpio, 94
 - proxy::Led, 101
- torque_sensors.cpp
 - MICRAS_PROXY_TORQUE_SENSORS_CPP, 195
- torque_sensors_config
 - target.cpp, 141
 - target.hpp, 145
- TorqueSensors
 - proxy::TorqueSensors< num_of_sensors >, 131
- transmit
 - hal::Spi, 122
- turn_off
 - proxy::Argb< num_of_leds >, 22
 - proxy::Led, 101
- turn_on
 - proxy::Led, 102
- unselect_device
 - hal::Spi, 123
- update
 - proxy::Buzzer, 29
- update_data
 - proxy::Imu, 97
- UVW_ABI
 - Registers::Settings2::Fields, 86
- UVW_off
 - Registers::Disable::Fields, 82
- UVWPP
 - Registers::Settings3::Fields, 87
- voltage_divider
 - proxy::Battery::Config, 44
- W
 - proxy::Imu, 96
- warning
 - proxy::RotarySensor::DataFrame::Fields, 80
- write
 - hal::Flash, 92
 - hal::Gpio, 94
- X
 - proxy::Imu, 96
- Y
 - proxy::Imu, 96
- Z
 - proxy::Imu, 96
- ZPSL
 - Registers::Zpsl::Fields, 89

zposl
 Registers, [114](#)
zposl_addr
 Registers, [114](#)
ZPOSM
 Registers::Zposm::Fields, [90](#)
zposm
 Registers, [114](#)
zposm_addr
 Registers, [115](#)