

ACC Firmware

Generated by Doxygen 1.15.0

1 Class Index	1
1.1 Class List	1
2 File Index	3
2.1 File List	3
3 Class Documentation	5
3.1 ACC_t Struct Reference	5
3.2 CAN_Driver_t Struct Reference	5
3.2.1 Detailed Description	6
3.3 SensorInputs_t Struct Reference	6
4 File Documentation	7
4.1 acc_config.h	7
4.2 can_driver.h	8
4.3 control_system.h	8
4.4 sensor_inputs.h	9
Index	11

Chapter 1

Class Index

1.1 Class List

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

ACC_t	5
CAN_Driver_t	5
SensorInputs_t	6

Chapter 2

File Index

2.1 File List

Here is a list of all documented files with brief descriptions:

C:/Users/Franc/OneDrive/Desktop/UWAM_Projects/Github/UWAM_ACC/ACC_Firmware/App/Inc/ acc_config.h	
7	
C:/Users/Franc/OneDrive/Desktop/UWAM_Projects/Github/UWAM_ACC/ACC_Firmware/App/Inc/↔ Drivers/ can_driver.h	8
C:/Users/Franc/OneDrive/Desktop/UWAM_Projects/Github/UWAM_ACC/ACC_Firmware/App/Inc/↔ Drivers/ control_system.h	8
C:/Users/Franc/OneDrive/Desktop/UWAM_Projects/Github/UWAM_ACC/ACC_Firmware/App/Inc/↔ Drivers/ sensor_inputs.h	9

Chapter 3

Class Documentation

3.1 ACC_t Struct Reference

Public Attributes

- float **setpoint_temperature**
- float **setpoint_pressure**
- float **actual_temperature**
- float **actual_pressure**

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

- C:/Users/Franc/OneDrive/Desktop/UWAM_Projects/Github/UWAM_ACC/ACC_Firmware/App/Inc/Drivers/control↵_system.h

3.2 CAN_Driver_t Struct Reference

```
#include <can_driver.h>
```

Public Attributes

- CAN_HandleTypeDef * **hcan1**
- CAN_HandleTypeDef * **hcan2**
- CAN_TxHeaderTypeDef **tx1**
- CAN_TxHeaderTypeDef **tx2**
- CAN_RxHeaderTypeDef **rx1**
- uint8_t **tx_data** [8]
- uint8_t **rx_data** [8]
- uint32_t **id**
- uint8_t **len**

3.2.1 Detailed Description

Dual CAN driver for the ACC

- Franco H

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

- C:/Users/Franc/OneDrive/Desktop/UWAM_Projects/Github/UWAM_ACC/ACC_Firmware/App/Inc/Drivers/can↵_driver.h

3.3 SensorInputs_t Struct Reference

Public Attributes

- ADC_HandleTypeDef * **adc**
- TIM_HandleTypeDef * **tim**
- volatile uint16_t **adc_raw** [ADC_CH_COUNT]
- volatile float **adc_mV** [ADC_CH_COUNT]
- float **temp_c** [4]
- float **pressure_psi** [4]
- float **seg_temp_c** [AMS_SEGMENT_COUNT]
- float **ch1_duty_cycle**
- float **ch2_duty_cycle**
- float **ch3_duty_cycle**
- float **fan_rpm**

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

- C:/Users/Franc/OneDrive/Desktop/UWAM_Projects/Github/UWAM_ACC/ACC_Firmware/App/Inc/Drivers/sensor↵_inputs.h

Chapter 4

File Documentation

4.1 acc_config.h

```
00001 // AUTO-GENERATED FILE. DO NOT EDIT.
00002 // Generated from acc_config.yaml
00003
00004 #pragma once
00005
00006 // Project: UWA Motorsports Accumulator Cooling Control
00007 // Version: 1.0.0
00008 // Author: Franco Heraud
00009 // Contributors: ...
00010
00011 #define ACC_ADC_RES_BITS 12
00012 #define ACC_ADC_VREF_VOLTS 3.3f
00013 #define ACC_TIMER_TICK_HZ 1000000u
00014
00015 #define ACC_FAN_COUNT 2
00016 #define ACC_FAN_PULSES_PER_REV 1
00017 #define ACC_FAN_PWM_MIN_DUTY 0.2f
00018 #define ACC_FAN_PWM_MAX_DUTY 0.95f
00019
00020 // CAN IDs
00021 #define ACC_CAN_ID_ACC_BASE 0x440u
00022 #define ACC_CAN_ID_ACC_TEMP 0x441u
00023 #define ACC_CAN_ID_ACC_PRESSURE 0x442u
00024 #define ACC_CAN_ID_ACC_SEG_TEMP 0x443u
00025 #define ACC_CAN_ID_ACC_TACH 0x444u
00026 #define ACC_CAN_ID_ACC_POWER 0x445u
00027
00028 // Pressure sensors
00029 #define ACC_PRESSURE_SENSOR_COUNT 4u
00030
00031 #define ACC_PRESS_P1_ADC_CH 1u
00032 #define ACC_PRESS_P1_MIN_PSI 0f
00033 #define ACC_PRESS_P1_MAX_PSI 100f
00034
00035 #define ACC_PRESS_P2_ADC_CH 2u
00036 #define ACC_PRESS_P2_MIN_PSI 0f
00037 #define ACC_PRESS_P2_MAX_PSI 100f
00038
00039 #define ACC_PRESS_P3_ADC_CH 3u
00040 #define ACC_PRESS_P3_MIN_PSI 0f
00041 #define ACC_PRESS_P3_MAX_PSI 100f
00042
00043 #define ACC_PRESS_P4_ADC_CH 4u
00044 #define ACC_PRESS_P4_MIN_PSI 0f
00045 #define ACC_PRESS_P4_MAX_PSI 100f
00046
00047 // Thermistor (Steinhart-Hart) coefficients (common)
00048 #define ACC_TH_A 0.003354016f
00049 #define ACC_TH_B 0.000250275f
00050 #define ACC_TH_C 2.42945e-06f
00051 #define ACC_TH_D -7.3121e-08f
00052
00053 #define ACC_TEMP_SENSOR_COUNT 4u
00054
00055 #define ACC_TEMP_T1_ADC_CH 5u
00056 #define ACC_TEMP_T2_ADC_CH 6u
00057 #define ACC_TEMP_T3_ADC_CH 7u
```

```

00058 #define ACC_TEMP_T4_ADC_CH      8u
00059
00060 // Control thresholds
00061 #define ACC_TEMP_THRESHOLD1_DEGC  40f
00062 #define ACC_TEMP_THRESHOLD2_DEGC  60f
00063 #define ACC_TEMP_HYSTERESIS_DEGC  3f

```

4.2 can_driver.h

```

00001
00005
00006
00007 #ifndef INC_CAN_DRIVER_H_
00008 #define INC_CAN_DRIVER_H_
00009
00010 #ifdef __cplusplus
00011 extern "C" {
00012 #endif
00013
00014 #include "main.h"
00015 #include <stdint.h>
00016 #include "acc_config.h"
00017
00018
00019 typedef struct {
00020     CAN_HandleTypeDef *hcan1;
00021     CAN_HandleTypeDef *hcan2;
00022     CAN_TxHeaderTypeDef tx1;
00023     CAN_TxHeaderTypeDef tx2;
00024     CAN_RxHeaderTypeDef rx1;
00025     uint8_t tx_data[8];
00026     uint8_t rx_data[8];
00027     uint32_t id;
00028     uint8_t len;
00029 } CAN_Driver_t;
00030
00031 HAL_StatusTypeDef CAN_InitDriver(CAN_Driver_t *drv);
00032 HAL_StatusTypeDef CAN_Transmit1 (CAN_Driver_t *drv);
00033 HAL_StatusTypeDef CAN_Transmit2 (CAN_Driver_t *drv);
00034 HAL_StatusTypeDef CAN_Receive1 (CAN_Driver_t *drv);
00035
00036 void CAN_Test(void);
00037
00038 #ifdef __cplusplus
00039 }
00040 #endif
00041 #endif /* INC_CAN_DRIVER_H_ */

```

4.3 control_system.h

```

00001 /*
00002  * control_system.h
00003  *
00004  * Created on: Nov 3, 2025
00005  * Author: Franc
00006  */
00007
00008 #ifndef INC_CONTROL_SYSTEM_H_
00009 #define INC_CONTROL_SYSTEM_H_
00010
00011 #include "main.h"
00012 #include "can_driver.h"
00013 #include "sensor_inputs.h"
00014 #include <stdbool.h>
00015 #include <string.h>
00016 #include "acc_config.h"
00017
00018
00019 #define TEMP_THRESHOLD1  40.0f
00020 #define TEMP_THRESHOLD2  50.0f
00021 #define HYSTERESIS_WIDTH  3.0f
00022 #define ACC_DOUT_CAN_ID1  (uint16_t)0x4A2
00023 #define ACC_DOUT_CAN_ID2  (uint16_t)0x4A3
00024
00025
00026
00027
00028 typedef struct {

```

```

00029     float setpoint_temperature;    // deg C
00030     float setpoint_pressure;        // Pa
00031
00032     float actual_temperature;
00033     float actual_pressure;
00034
00035 } ACC_t;
00036
00037
00038
00039
00040
00041 #endif /* INC_CONTROL_SYSTEM_H_ */

```

4.4 sensor_inputs.h

```

00001 /*
00002  * sensor_inputs.h
00003  *
00004  * Created on: Nov 1, 2025
00005  * Author: Franc
00006  *
00007  */
00008
00009
00010
00011 #ifndef INC_SENSOR_INPUTS_H_
00012 #define INC_SENSOR_INPUTS_H_
00013
00014
00015 #ifdef __cplusplus
00016 extern "C" {
00017 #endif
00018
00019 #include "main.h"
00020 #include "can_driver.h"
00021 #include <stdint.h>
00022 #include <stdbool.h>
00023 #include <limits.h>
00024 #include <string.h>
00025 #include <math.h>
00026 #include <stdio.h>
00027 #include <math.h>
00028 #include "acc_config.h"
00029
00030
00031
00032 // pc8 adc no worky >:(
00033 #define ADC_CH_COUNT      8
00034 #define ADC_RESOLUTION    4095
00035 #define PWM_RESOLUTION    255
00036
00037 #define TEMP_ACC1_IN0      0
00038 #define TEMP_ACC2_IN1      1
00039 #define TEMP_DUCT1_IN2     2
00040 #define TEMP_DUCT2_IN3     3
00041
00042 #define PRES_ACC1_IN7      4
00043 #define PRES_ACC2_IN8      5
00044 #define PRES_DUCT1_IN9     6
00045 #define PRES_DUCT2_INX    99    // placeholder for now...
00046
00047 #define STATUS_LED1        GPIO_PIN_6
00048 #define STATUS_LED2        GPIO_PIN_7
00049 #define STATUS_LED3        GPIO_PIN_11
00050 #define STATUS_LED4        GPIO_PIN_12
00051 #define STATUS_LED5        GPIO_PIN_13
00052
00053 #define P_SENSOR_VSUP1      4.5f    // V
00054 #define P_SENSOR_PMAX1      50.0f   // psi
00055 #define P_SENSOR_PMIN1      0.0f    // psi
00056
00057 #define AMS_SEGMENT_COUNT   4
00058 #define AMS_SEG_TEMP_CAN_ID (uint16_t)0x00000000
00059
00060 #define TACH_TIMER_CLK_HZ    16000000 // 16 MHz
00061
00062
00063 typedef struct {
00064     ADC_HandleTypeDef *adc;
00065     TIM_HandleTypeDef *tim;
00066
00067     // adc dma buffers

```

```
00068     volatile uint16_t adc_raw[ADC_CH_COUNT];
00069     volatile float adc_mV[ADC_CH_COUNT];
00070
00071     // data buffers for temperature and pressure values
00072     float temp_c[4];
00073     float pressure_psi[4]; // (what unit??)
00074     float seg_temp_c[AMS_SEGMENT_COUNT];
00075
00076     // pwm
00077     float ch1_duty_cycle;
00078     float ch2_duty_cycle;
00079     float ch3_duty_cycle;
00080
00081     // other
00082     float fan_rpm;
00083
00084
00085 } SensorInputs_t;
00086
00087 HAL_StatusTypeDef Sensors_Init(SensorInputs_t *si, ADC_HandleTypeDef *adc, TIM_HandleTypeDef *tim);
00088 HAL_StatusTypeDef Sensors_Start(SensorInputs_t *si);
00089
00090 void Store_Temperature_Readings(SensorInputs_t *si);
00091 void Store_Pressure_Readings(SensorInputs_t *si);
00092 void PWM_SetAll(SensorInputs_t *si);
00093 void Update_Fan_Speed(SensorInputs_t *si);
00094 void Update_ADC_Buffers(SensorInputs_t *si);
00095
00096
00097 #endif /* INC_SENSOR_INPUTS_H_ */
```

Index

ACC_t, [5](#)

C:/Users/Franc/OneDrive/Desktop/UWAM_Projects/Github/UWAM_ACC/ACC_Firmware/App/Inc/acc_config.h,
[7](#)

C:/Users/Franc/OneDrive/Desktop/UWAM_Projects/Github/UWAM_ACC/ACC_Firmware/App/Inc/Drivers/can_driver.h,
[8](#)

C:/Users/Franc/OneDrive/Desktop/UWAM_Projects/Github/UWAM_ACC/ACC_Firmware/App/Inc/Drivers/control_system.h,
[8](#)

C:/Users/Franc/OneDrive/Desktop/UWAM_Projects/Github/UWAM_ACC/ACC_Firmware/App/Inc/Drivers/sensor_inputs.h,
[9](#)

CAN_Driver_t, [5](#)

SensorInputs_t, [6](#)