2025 ACC Firmware Documentation

Nov 2025
Rev. 1
Author & Project Head: Franco Heraud

Signed off by: ____________________

1 Introduction

1.1 Overview

1.2 Requirements

The main functional requirements for the onboard MCU are as follows:

- **Drivers**
  - Must be dual-CAN compatible (one for data-logging and the other for receiving temperature data from the AMS)
  - Preferred to use DMA ADC reads
- **Sensors**
  - Must perform ADC reads on the 4x temperature sensors and 4x pressure sensors and perform the necessary sensor input conversions
  - Must receive additional segment temperature data from the CAN bus
  - Must accept a tachometer input from one of the fans
  - Must perform another ADC read from the power input pins connected to the switch circuit to measure the current power consumption of the synchronous buck controller
- **Control System + Other Considerations**
  - The fan speed must be regulated based off the sensor and tachometer inputs as inputs to cool the accumulator
  - Preferred to use a PID controller on top of basic threshold logic for controlling the fan speed
  - Sensor and power consumption data must be transmitted over the CAN bus for data-logging purposes

Any feedback + additional notes would be greatly appreciated.

2 Sensors

2.1 Fan RPM reading logic

The main fan RPM calculation logic will utilize a GPIO pin configured as an external interrupt and an internal hardware timer that uses a counter to calculate the pulse frequency in Hz and, hence, the fan RPM. Firstly, we alter the clock configuration settings to ensure that the input clock frequency used by the timer (TIM2 in this case) is 1 MHz. Where the clock is being divided from the input 8 MHz clock.

Then, if we are using a 32-bit counter resolution, the counter period is $= 2^{32}$ ticks. This is the number of ticks before the timer overflows, which occurs at:

$$T_{overflow} = \frac{2^{32}}{10^6} \approx 71.58 \text{ minutes}$$

Essentially, there is $\Delta t = 1/1\text{MHz} = 1\mu s$ per tick. So we just need to count the number of ticks per digital pulse supplied by the tachometer input and find the pulse period using:

$$T_{pulse} = \Delta t \times (\text{Count Between Pulses})$$

Thus, we derive the pulse frequency in Hz using:

$$f_{pulse} = \frac{1}{T_{pulse}} = \frac{1}{1\mu s \times \text{Count Between Pulses}} = \frac{10^6 \text{ Hz}}{\text{Count Between Pulses}}$$

The core logic implemented through an external interrupt and an internal hardware timer is as follows:

```c
const uint8_t PULSES_PER_REVOLUTION = 1;

static volatile uint32_t tach_last_ticks  = 0;
static volatile uint32_t tach_delta_ticks = 0;
static volatile uint8_t  tach_new_period  = 0;

void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin) {
    if (GPIO_Pin == TACH_IN_Pin) {
        uint32_t now = __HAL_TIM_GET_COUNTER(&htim2);

        uint32_t delta = now - tach_last_ticks;
        tach_last_ticks = now;

        if (delta > 0) {
            tach_delta_ticks = delta;
            tach_new_period  = 1;
        }
    }
}

void Update_Fan_Speed(SensorInputs_t *si) {
  if (!tach_new_period || tach_delta_ticks == 0) si->fan_rpm = 0.0f;
  const float tick_freq_hz = (float)pow(10, 6);
  float pulse_freq_hz = tick_freq_hz / (float)tach_delta_ticks;
  si->fan_rpm = (pulse_freq_hz * 60) / (float)PULSES_PER_REVOLUTION;
}
```

2.2 Temperature sensor reading logic

The ACC internal/external temperature inputs come from a temperature sensor with an in-built thermistor. Given the non-linear relationship between the thermistor temperature and resistance, we must use the provided constants and calibration values from the data sheet [] to accurately convert the input voltage from the ADC inputs to a temperature value in degrees Celsius.

...

(WIP)