Specifications for the Attitude Control and Determination System-XCubeSat

Dhruv Sharma, Damien Seux, Quentin Lisack, Kevin Garanger 12 April 2015

Contents

1 Introduction **4**

2 Architecture of the ADCS **5**

3 External interfaces **10**

4 Software Architecture **11**
4.1 Division of the software 11
4.2 Structure of classes 12
4.3 Definition of ports 13
4.4 API Firmware 14
4.4.1 PWM 14
4.4.2 ADC 14
4.4.3 USART 14
4.5 Driver 14
4.6 Specialized Functions 14

5 Data formats **15**

6 Testing **16**

List of Figures

4.1 *Division of the software in three layers each independent of the other* 11
4.2 *Class structure for ADCS Software* 12

List of Tables

4.1 List of peripherals with alternate functions and other configurable parameters . . 14

Chapter 1

Introduction

The objective of this document is to summarize in one place the specifications related to the software of the Attitude Control and Determination System, henceforth ADCS of the nano-satellite X-CubeSat developed by the students of cole Polytechnique, Paris. We have divided the document into the following sections

- Definition of the architecture of the ADCS
- External Interfaces available on the ADCS
- Sensors and peripherals on the ADCS
- Data formats used for computing
- Protocols for testing

This document shall be useful for all people currently concerned with the conception of the ADCS and eventually for students who will take over the project in the months to come. This document is however, far from being complete and additions, corrections and deletions will be made till the day of delivery.

Chapter 2

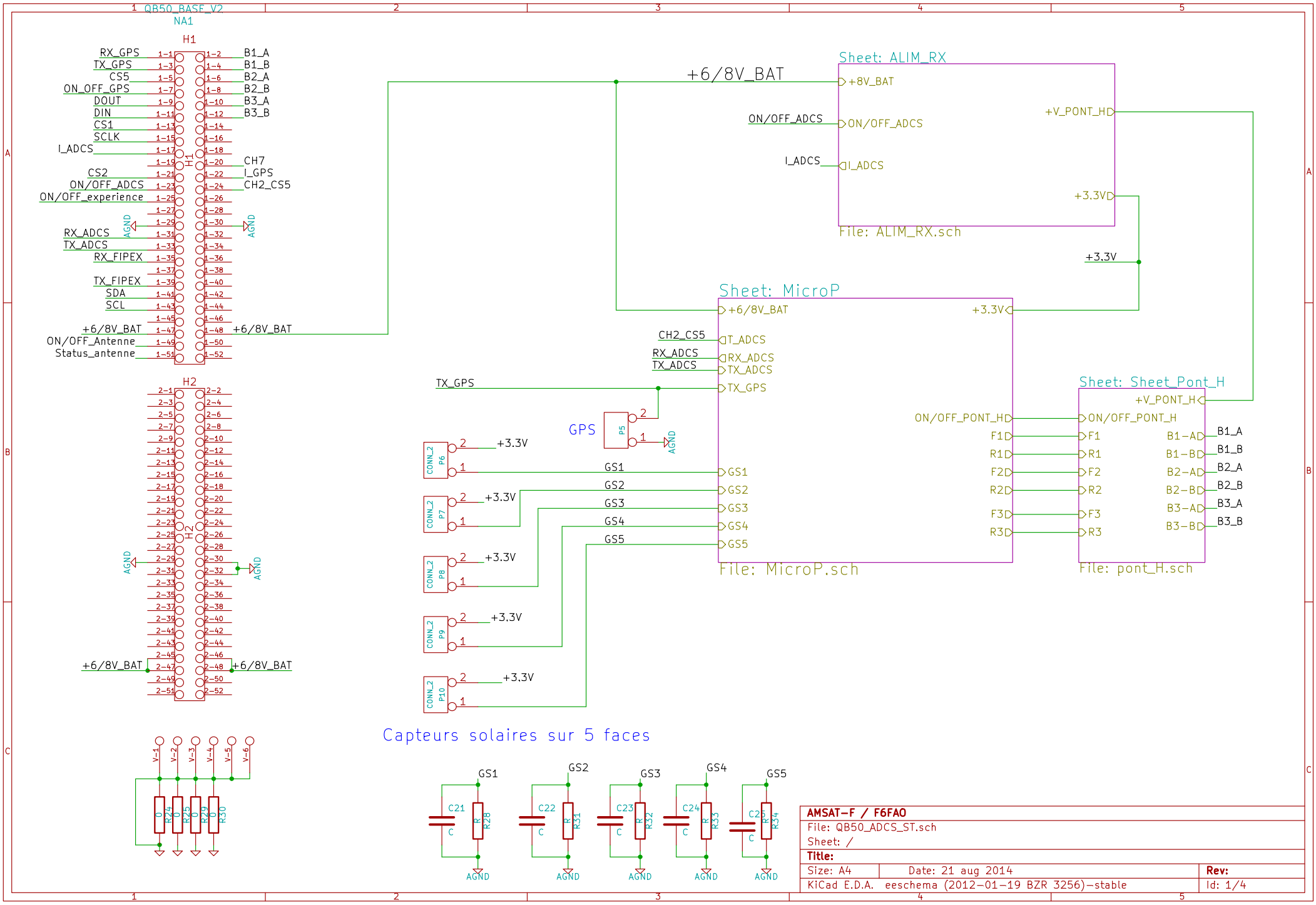
Architecture of the ADCS

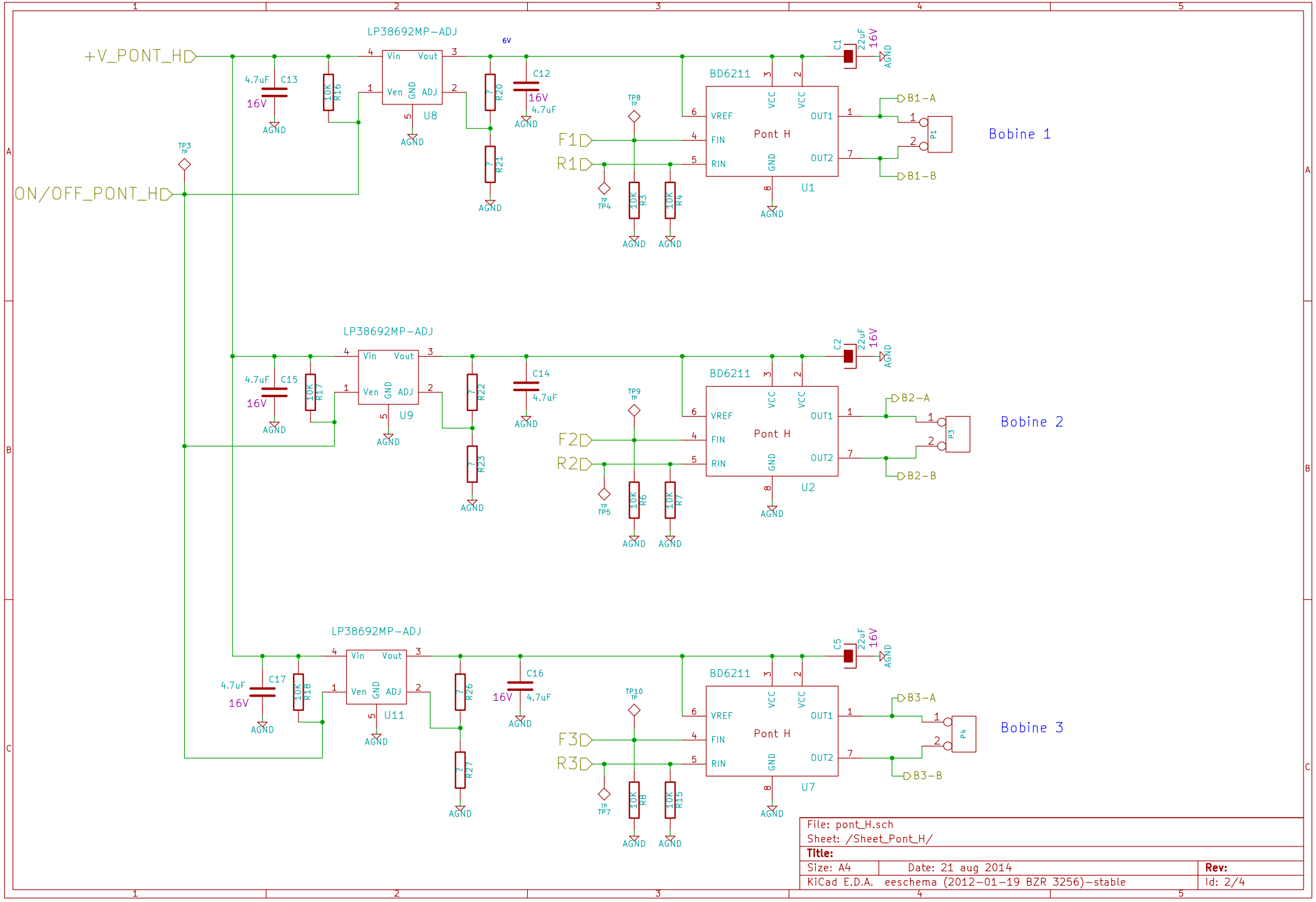
The ADCS uses a microprocessor manufactured by ST Microelectronics based on the ARM Cortex-M4 Architecture. The model used is STM 32F405 RGT6

The available interfaces, sensors and other peripherals available on the ADCS are:

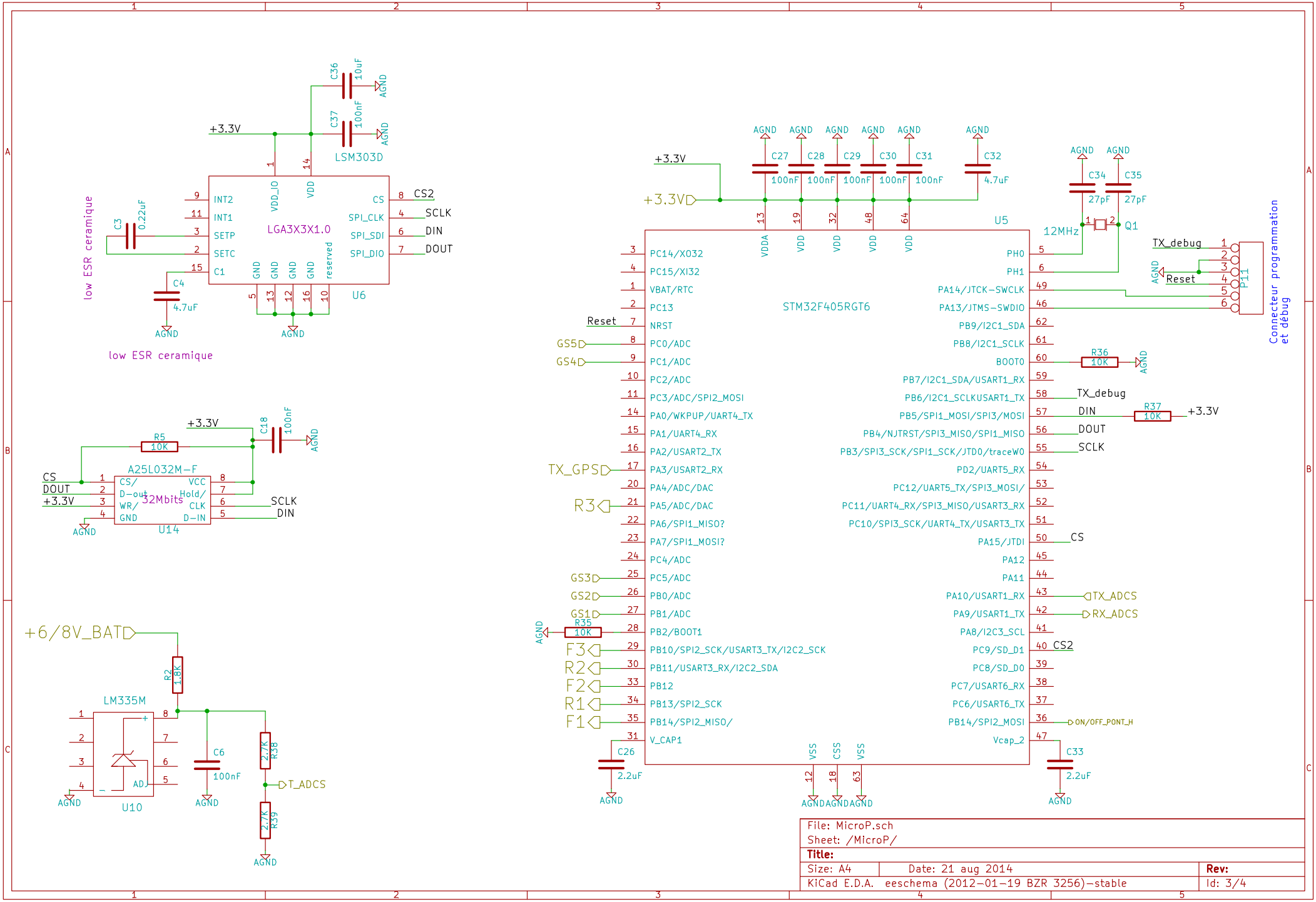
- 32 MB External Memory
- 7(or 9) sun sensors
- Magnetometer
- Gyrometer
- H-Bridges to modulate power to the magneto-torquers
- A Serial interface with the On-Board Computer (*Ordinateur de Bord* henceforth, ODB)

Following is the complete plan for the ADCS:

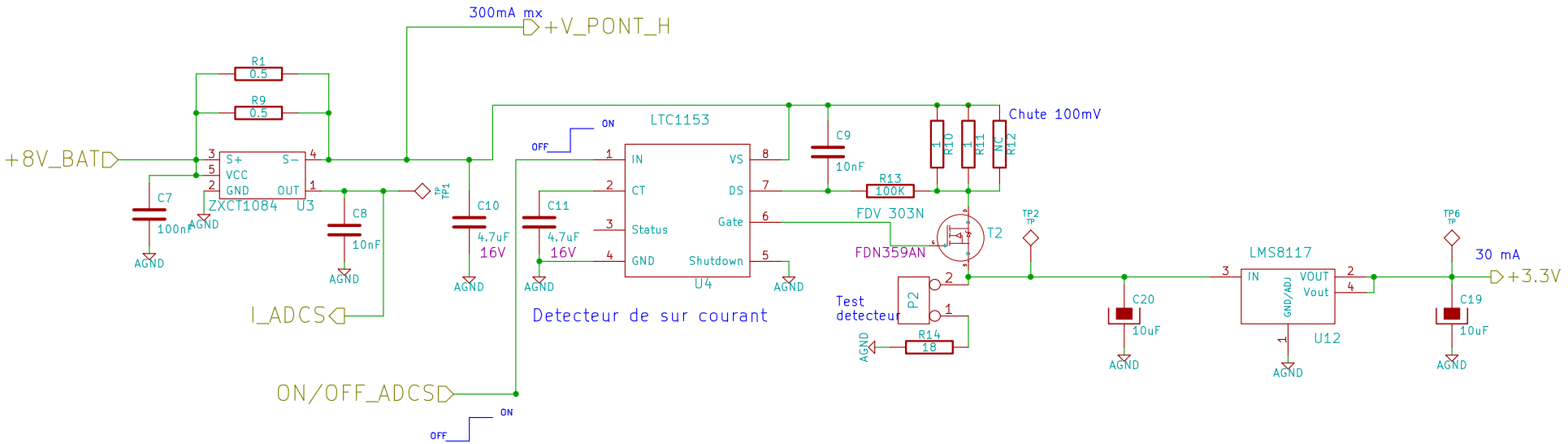

QB50_BASE_V2
NA1
H1
RX_GPS
TX_GPS
CS5
ON_OFF_GPS
DOUT
DIN
CS1
SCLK
I_ADCS
CS2
ON/OFF_ADCS
ON/OFF_experience
RX_ADCS
TX_ADCS
RX_FIPEX
TX_FIPEX
SDA
SCL
+6/8V_BAT
ON/OFF_Antenne
Status_antenne
B1_A
B1_B
B2_A
B2_B
B3_A
B3_B
CH7
I_GPS
CH2_CS5
AGND
H2
+6/8V_BAT
Sheet: ALIM_RX
+8V_BAT
ON/OFF_ADCS
I_ADCS
+V_PONT_H
+3.3V
File: ALIM_RX.sch
Sheet: MicroP
+6/8V_BAT
+3.3V
CH2_CS5
T_ADCS
RX_ADCS
TX_ADCS
TX_GPS
GPS
P5
CONN_2
P6
P7
P8
P9
P10
GS1
GS2
GS3
GS4
GS5
File: MicroP.sch
Sheet: Sheet_Pont_H
+V_PONT_H
ON/OFF_PONT_H
F1
R1
F2
R2
F3
R3
B1-A
B1-B
B2-A
B2-B
B3-A
B3-B
File: pont_H.sch
Capteurs solaires sur 5 faces
V-1
V-2
V-3
V-4
V-5
V-6
R24
R25
R29
R30
C21
C22
C23
C24
C25
R28
R31
R32
R33
R34
AMSAT-F / F6FAO
File: QB50_ADCS_ST.sch
Sheet: /
Title:
Size: A4
Date: 21 aug 2014
Rev:
KiCad E.D.A. eeschema (2012-01-19 BZR 3256)-stable
Id: 1/4



+V_PONT_H
ON/OFF_PONT_H
LP38692MP-ADJ
Vin
Vout
Ven
GND
ADJ
U8
U9
U11
6V
4.7uF
16V
C13
C15
C17
C12
C14
C16
R16
R17
R18
R20
R21
R22
R23
R26
R27
TP3
TP4
TP5
TP7
TP8
TP9
TP10
F1
R1
F2
R2
F3
R3
R3
R4
R6
R7
R8
R15
10K
BD6211
VREF
FIN
RIN
VCC
GND
OUT1
OUT2
Pont H
U1
U2
U7
C1
C2
C5
22uF
AGND
B1-A
B1-B
B2-A
B2-B
B3-A
B3-B
P1
P3
P4
Bobine 1
Bobine 2
Bobine 3
File: pont_H.sch
Sheet: /Sheet_Pont_H/
Title:
Size: A4
Date: 21 aug 2014
Rev:
KiCad E.D.A. eeschema (2012-01-19 BZR 3256)-stable
Id: 2/4



LSM303D
LGA3X3X1.0
U6
+3.3V
C36 10uF
C37 100nF
AGND
INT2
INT1
SETP
SETC
C1
VDD_IO
VDD
CS
SPI_CLK
SPI_SDI
SPI_DIO
GND
reserved
CS2
SCLK
DIN
DOUT
C3 0.22uF
C4 4.7uF
low ESR ceramique
low ESR ceramique
A25L032M-F
32Mbits
U14
R5 10K
C18 100nF
CS
DOUT
+3.3V
CS/
D-out
WR/
GND
VCC
Hold/
CLK
D-IN
SCLK
DIN
+6/8V_BAT
LM335M
U10
R2 1.8K
C6 100nF
R38 2.7K
R39 2.7K
T_ADCS
STM32F405RGT6
U5
C27 100nF
C28 100nF
C29 100nF
C30 100nF
C31 100nF
C32 4.7uF
C34 27pF
C35 27pF
12MHz
Q1
VDDA
VDD
PC14/XO32
PC15/XI32
VBAT/RTC
PC13
NRST
PC0/ADC
PC1/ADC
PC2/ADC
PC3/ADC/SPI2_MOSI
PA0/WKPUP/UART4_TX
PA1/UART4_RX
PA2/USART2_TX
PA3/USART2_RX
PA4/ADC/DAC
PA5/ADC/DAC
PA6/SPI1_MISO?
PA7/SPI1_MOSI?
PC4/ADC
PC5/ADC
PB0/ADC
PB1/ADC
PB2/BOOT1
PB10/SPI2_SCK/USART3_TX/I2C2_SCK
PB11/USART3_RX/I2C2_SDA
PB12
PB13/SPI2_SCK
PB14/SPI2_MISO/
V_CAP1
PH0
PH1
PA14/JTCK-SWCLK
PA13/JTMS-SWDIO
PB9/I2C1_SDA
PB8/I2C1_SCLK
BOOT0
PB7/I2C1_SDA/USART1_RX
PB6/I2C1_SCLKUSART1_TX
PB5/SPI1_MOSI/SPI3/MOSI
PB4/NJTRST/SPI3_MISO/SPI1_MISO
PB3/SPI3_SCK/SPI1_SCK/JTD0/traceW0
PD2/UART5_RX
PC12/UART5_TX/SPI3_MOSI/
PC11/UART4_RX/SPI3_MISO/USART3_RX
PC10/SPI3_SCK/UART4_TX/USART3_TX
PA15/JTDI
PA12
PA11
PA10/USART1_RX
PA9/USART1_TX
PA8/I2C3_SCL
PC9/SD_D1
PC8/SD_D0
PC7/USART6_RX
PC6/USART6_TX
PB14/SPI2_MOSI
Vcap_2
VSS
CSS
Reset
GS5
GS4
TX_GPS
R3
GS3
GS2
GS1
R35 10K
F3
R2
F2
R1
F1
C26 2.2uF
C33 2.2uF
R36 10K
R37 10K
TX_debug
DIN
DOUT
SCLK
CS
TX_ADCS
RX_ADCS
CS2
ON/OFF_PONT_H
TX_debug
Reset
P11
Connecteur programmation et débug
File: MicroP.sch
Sheet: /MicroP/
Title:
Size: A4
Date: 21 aug 2014
Rev:
KiCad E.D.A. eeschema (2012-01-19 BZR 3256)-stable
Id: 3/4



File: ALIM_RX.sch		
Sheet: /ALIM_RX/		
Title:		
Size: A4	Date: 21 aug 2014	**Rev:**
KiCad E.D.A. eeschema (2012-01-19 BZR 3256)-stable		Id: 4/4

Chapter 3

External interfaces

Chapter 4

Software Architecture

This is the heart of this document. We discuss here the architecture of the software, the various peripherals and the sensors mounted on the microprocessors, the various functions associated with them detailing every time the input, output and the configuration parameters for each of the peripherals.

We first begin by detailing how the software has been divided. Next we define the various classes that we have divided the software into. While the first division is an abstract division that we made to facilitate the development of the software, the structure of classes and their interdependance correspond to how the software will actually be implemented.

Once the abstract division and the structure of classes are defined we detail the correspondence of the various ports/ pins available on the microprocessor and the peripherals connected. Next, we define the functions corresponding to each of the divisions. Irrespective of their eventual use, all functions shall find themselves in one of three categories that we shall define in the section **Division of the software**.

4.1 Division of the software

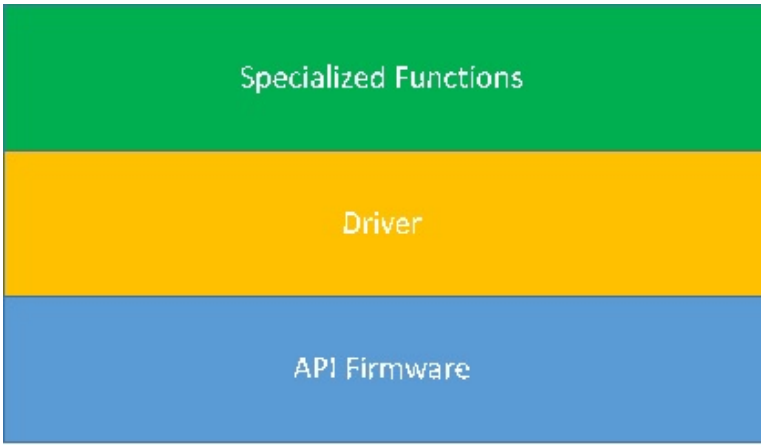


Figure 4.1: *Division of the software in three layers each independent of the other*

The ADCS software is divided into the following layers

1. **API Firmware**: In this layer, all the low-level functionalities are collected. By low-level functionalities, we mean all those functionalities which are not directly relevant to the functioning of the algorithms. This layer doesn't provide any data to the algorithms. Rather, this layer is concerned only with initializing and configuring the various peripherals that we have on the ADCS.

2. **Driver**: jhis layer acts as a bridge between the lower level API firmware and the higher-level specialized functions that we require to run our algorithms. Ideally, in this layer we find the functions which get the raw data from the sensors, convert or calibrate them as necessary and send them to the algorithms for further use.

3. **Specialized Functions**: The specialized functions are the functions which implement the algorithms for determining and controlling the attitude of the satellite. These functions are high-level functions and require the data provided to them by the *Driver* layer above to function.

4.2 Structure of classes

Once we have abstracted the software as being divided into layers depending on the kind of functions we will write for the software, we consider next the division of the software into dedicated classes. The class structure as envisaged for the ADCS is summarized in the graph below. The arrow head signifies that the a particular class gets a particular piece of data from the class where the arrow originates.

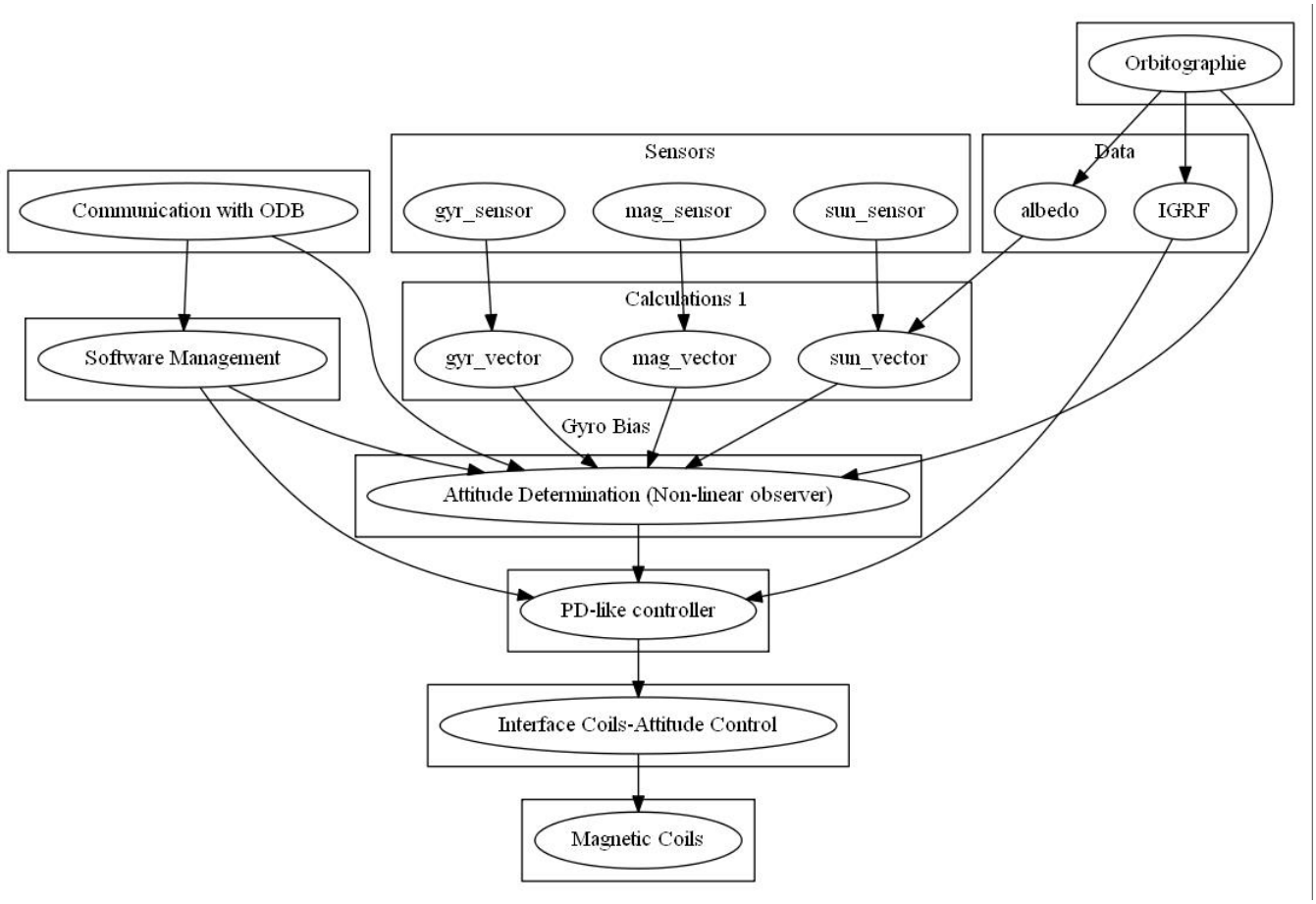


Figure 4.2: *Class structure for ADCS Software*

The various layers as shown in the class structure above are detailed below

1. **Sensors**: This layer corresponds to the API Firmware layer that we presented earlier. We have three types of sensors: Gyrometer, a magnetometer and 9 sun-sensors. This part of the software corresponds to the initialization and configuring the sensors (and the ports

to which they are associated on the microprocessor) so that they can start generating the data.

2. **Calculations 1**: This layer consists of a first layer of calculations effectuated once raw data is received from the sensors. In this layer, we receive the raw data from the sun-sensors, magnetometer and the gyrometer.

3. **Calculations 2**:This corresponds to a supplementary layer of calculations which is effectuated once the raw data has been received (not shown in the class structure). This step might take different forms for different sensors. For instance, once we have received the sun sensor raw data, the supplementary steps include converting the raw data into a voltage and further combining the data from the 9 sun-sensors into one sun-vector with three components. For the magnetometer however, we can eventually consider calibrating the data with the temperature. Identically for the gyrometer as well.

   Finally, this step also relates to reading the terrestrial magnetic field map and the albedo map from the external memory. All this data once collected, calibrated and treated will be given to the algorithms, which we explain below.

4. **Attitude Determination and Control Phase**: Once the data has been generated (after eventual calibrations), we have the layer where the actual control of the satellite takes place. The details of the algorithms will be presented in an ulterior section.

5. **PD-like controller**: This is what controls the satellite by providing us as output the magnetic moment to be generated in the magnetic coils. The controller then gives its output to the Coils-Attitude Control Interface.

6. **Interface Coils-Attitude Control**: This layer corresponds to the calculations to be done to configure the magneto-torquers. The magneto-torquers are controlled by Pulse-Width Modulation(PWM). For PWM, we require the frequency of modulation (in KHz) and the duty cycle (in percent). The result of the algorithms, however, is the magnetic moment to be generated by the magnetic torquers. Thus we require precise data on the behaviour of the magnetic coils used to convert this magnetic moment into a duty-cycle. (The frequency of modulation shall be fixed for all the coils)

7. **Magnetic Coils**: This is the layer which corresponds to the actual control that will be applied via the magnetic coils. The coils are powered via PWM for a fixed frequency usually aroun 15-20KHz.

4.3 Definition of ports

In this section we define the various ports, what their functionalities are and to which of the peripherals they are connected.

We also specify the alternate functions that need to be configured for each of the ports. These alternate functions could be Serial Communication (USART) for communicating with the ODB or an Analog to Digital Converter (ADC) for the sun-sensors.

Peripheral	Name of the port	Port on MPU	Logical State	Alternate Function(AF)	AF Details 1	AF Details 2
Sun Sensor	GS5	PC0	N/A	ADC	ADC_Resolution_8b	Channel10
	GS4	PC1	N/A	ADC	ADC_Resolution_8b	Channel 11
	GS3	PC5	N/A	ADC	ADC_Resolution_8b	Channel 15
	GS2	PB0	N/A	ADC	ADC_Resolution_8b	Channel 8
	GS1	PB1	N/A	ADC	ADC_Resolution_8b	Channel 9
PWM	F1	PB14	N/A	TIM1	Output Compare	
	R1	PB13	0 or 1	N/A		
	F2	PB12	0 or 1	N/A		
	R2	PB11	N/A	TIM2	Output Compare	
	F3	PB10	N/A	TIM2	Output Compare	
	R3	PA5	0 or 1	N/A		
Comm ODB	TX_ADCS	PA10	N/A	USART1_RX	Baud Rate	Bit size
	RX_ADCS	PA9	N/A	USART1_TX	Baud Rate	Bit size
MISC	On/Off Pont H	PB14	0 or 1	N/A		

Table 4.1: List of peripherals with alternate functions and other configurable parameters

4.4 API Firmware

This section details all the functions concerning the initialization and configuration of the peripherals.

4.4.1 PWM

4.4.2 ADC

4.4.3 USART

4.5 Driver

4.6 Specialized Functions

Chapter 5

Data formats

Chapter 6

Testing