SELINUS UNIVERSITY
OF SCIENCES AND LITERATURE

DEVELOPMENT OF A SECURE OPERATING SYSTEM FOR CUBESAT

By Mostafa Imam Salem Imam

Supervised by
Prof. Salvatore Fava Ph.D.

A DISSERTATION

Presented to the Department of
Computer Science
program at Selinus University

Faculty of Computer Science
in fulfillment of the requirements
for the degree of Doctor of Philosophy
in Software Engineering

2021

Declaration

I do hereby that I am the only author of this Thesis, and all the results and the researches on this Thesis done by me. The Thesis's title is "Development of a secure operating system for Cubesats" Submitted for the Award of Doctor of Philosophy in Information Security at the University of Selinus only. The borrowed material and sources in the Thesis have been acknowledged. The research is based on my work and not borrowed from another material.

April, 2021

Student Signature

Student ID: UNISE1273IT

Acknowledgment

My appreciation goes first to Almighty Allah for his blessings, guidance and accepting my prayers. I am also thankful for the support and love that I received from my parents, family and friends as well to the University of Selinus Faculty of Computer Sciences for the opportunity to pursue a Doctorate in Information Security.

April, 2021

Student Signature

Student ID: UNISE1273IT

Dedication

The Thesis is dedicated to my lovely parents, Mr. Imam Salem, Mrs. Maha Abdelmageed, my siblings (Mohamed, Hoda, Heba, Mona and Ahmed), my Egyptian friends in Sweden (Mostafa H, M. Fouad, M. Adel, Bassam G, Saher H, Luay S, Ramy N, M. Samy, M. Jawad, M. Diaa, Hady N, M. Nehad, Ahmed Rafaat, Hany F, Mina H, M. Moamen, M. Rabee and M. Alaa), My Swedish Friends (Stavros Constantinou, Sanna Axell, Amanda Jonsson, Ali Salem, Didrik Skiöldebrand) and Special thanks to prof. Ayman Bahaa, prof. Mohammed Sobh, prof. Hazem Abbas, prof. Your love, encourage and support was the source of inspiration in my life.

April, 2021

Table of Content

List of tables

List of figures

Glossary of terms

LEO	*Low Earth Orbit*
IoT	*Internet of things*
MarCO	*Mars Cube One*
NARSS	*National Authority for Remote Sensing and Space Sciences*
ADCS	*Attitude Determination and Control System*
MCU	*Microcontroller Unit*
OBC	*On Board Computer*
EPS	*Electrical Power SubSystem*
TT&C	*Telemetry Tracking and Command*
TLE	*Transient Launch Environment*
PCB	*Printed Circuit Board*
C&DH	*command and data handling*
VNIR	*Visible & Near-Infrared*
SWIR	*Short-Wave Infrared*
TIR	*Thermal Infrared*
SEL	*Single Event Latehup*
SEU	*Single Event Upset*

SEFI	*Single Event Functional Interrupt*
TID	*Total Ionization Dose*
TD	*Transient Does*
SEE	*Single event effect*
SMT	*Surface mount technology*
DIP	*dual in-line package*
SOIC	*Small-outline integrated circuit*
QFP	*quad flat package*
PGA	*pin grid array*
LGA	*land grid array*
IC	*Integrated circuits*
CPU	*Central processing unit*
CMSIS	*The Cortex Microcontroller Software Interface Standard*
RAM	*Random access memory*
ROM	*Read only memory*
SRAM	*Static random access memory*
EBI	*External Bus Interface*
DMA	*Direct Memory Access*
RTC	*Real-Time Clock*

ADC	*Analog to digital converter*
UART	*Universal Asynchronous Receiver/Transmitter*
SPI	*The Serial Peripheral Interface*
CMOS	*Complementary Metal Oxide Semiconductor*
FATFS	*File allocation table File system*
I2C	*Inter-Integrated Circuit*
RNG	*Random number generator*
MQTT	*Message Queuing Telemetry Transport*
Protobuf	*Protocol buffer*
RTOS	*Real-Time Operating System*
FCFS	*First Come First Serve*
SJF	*Shortest-Job-First*
ISR	*Interrupt Service Routine*
OTA	*Over the air update*
FCFS	*First Come First Serve*
SJF	*Shortest Job First*
RR	*Round Robin*
OASIS	*Organization for the Advancement of Structured Information Standards*
ISO	*International Organization for Standardization*
CSR	*Certificate signed request*

1 Introduction

1.1 Background

The National Authority for Remote Sensing and Space Sciences in Egypt (NARSS) has worked on a CubeSat project called EgyCubeSat1 and other Satellites for practical and educational purpose listed in [12]. EgyCubeSat1 was launched 2013 and its mission was to capture images for Egypt territories. A lot of issues found during the project in Attitude Determination and Control System (ADCS) of dominating the on board computer (OBC) resources and because it is important to have a strong ADCS in most satellite missions, It will be much better to have secure real-time operating system that will control all the actuators and sensors and also run all the algorithms and perform calculations and communicate securely with the ground station to reduce the power consumption and work efficiently.

1.2 Mission

In This Thesis, will found the design and development of the OBC needed for 3U CubeSat. The OBC will also be as an interface between the attitude controller unit of the satellite. The ADCS estimator of the satellite will be sent to the OBC via a tele-commands. Then OBC will compute the output/commands for the actuators and the sensors to achieve the desired attitude. This system must be secure and increasing the security will affect the power management which will be described later.

1.3 CubeSat Architecture

Every country nowadays is in a race to occupy the space, Every country wants to leave it is own footprint in this wide area, Every country wants to reach out for this unexplored treasures, The revenues from space industry expected to be 22 billions of dollars by 2024 and may go for 40 billions of dollars by 2029 [1]. Small Satellites made the race easier and much encouraging, the Space industry is now flourishing, especially those small Satellites are deployed in LEO, providing low latency communications [2], which offers a lot of new applications such as remote sensing, observation or communications [10]. That made huge companies like GOOGLE, FACEBOOK, SpaceX putting tons of investments to investigate how to use these small satellites to monitor the Earth and provide a connection to IoT devices in remote places. Besides the growing business, the academic enthusiast scientists have started to join the race to develop those small satellites, mostly those satellites can be classified by their weight, Femto which are less than 0.1 kg, Pico which are between 0.1 and 1 kg and also called **CubeSats** and those are the most popular small satellites, Nano which are between 1 and 10 kg, Micro which are between 10 and 100 kg, and Finally Mini which are between 100 and 1000 kg [2][11].

CubeSats development was started at Stanford University in 1999 in order to build a low-cost and low-weight satellite. After that, the CubeSat standards were defined to build these satellites in a cubic structure, this is why they called it CubeSat, with a small mass of 1.33 kg per each unit (U), a cheap cost less than $1000, available components and low power consumption. The shape for this satellite was chosen to be a cube because it provides enough surface area to generate solar power and provide better space-thermal stability. The one unit (1U) of the CubeSat satellite has dimensions of 10cm × 10cm × 10cm and a mass of 1.33 kg and this is called basic unit. CubeSats sizes vary from one unit (1U) to sixteen unit (16U) as in figure 1 [3].

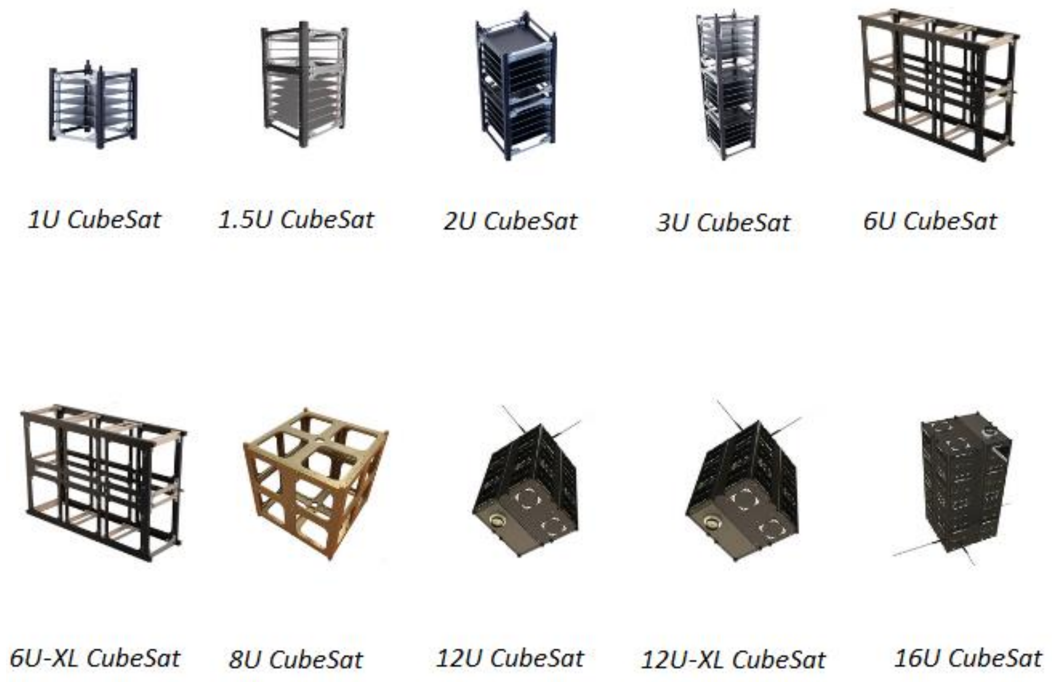


Figure 1: ISISPACE Standard CubeSat supported sizes

To take full advantage of the cubic shape, each face of the CubeSat covered solar cells or efficient solar panel wings. Those solar panel wings generates more power than what is needed which is between 20W and 60W in full sunlight compared to eight mounted solar cells, which generates power between 1W and 7W [4].

CubeSats usually used in communications and also missions dedicated to the scientist to understand and predict the weather, climate, earth's environment and disaster monitoring. CubeSats also used for space-science missions, which aim to expand the scientific knowledge in astronomy, heliophysics (space weather), and planetary science [5][6][7]. One of the most famous applications for small satellites is providing ubiquitous Internet connectivity for consumers and IoT devices. The Starlink project developed by SpaceX and deployed around 30,000 satellites to fill the consumers' need for high-speed Internet around the world, especially if the other solution are too expensive or in the regions without connectivity at all [8]. One other example of space-science CubeSat mission was MarCO, launched by NASA in May 2018 consisting of 6U CubeSats. MarCO was the first of this kind of spacecraft to fly to deep space and it succeeded in a flyby of Mars, relaying data to Earth from Insight as it landed on Mars [9].

Based on the increasing number of space-related applications, missions, research in the space industry is becoming more attractive as mentioned previously. The needs of making a suitable general purpose secure operating systems becomes more important for those kind of satellites to ensure the protection, the efficiency and usability.

The CubeSat have different modules and subsystems as following Figure:

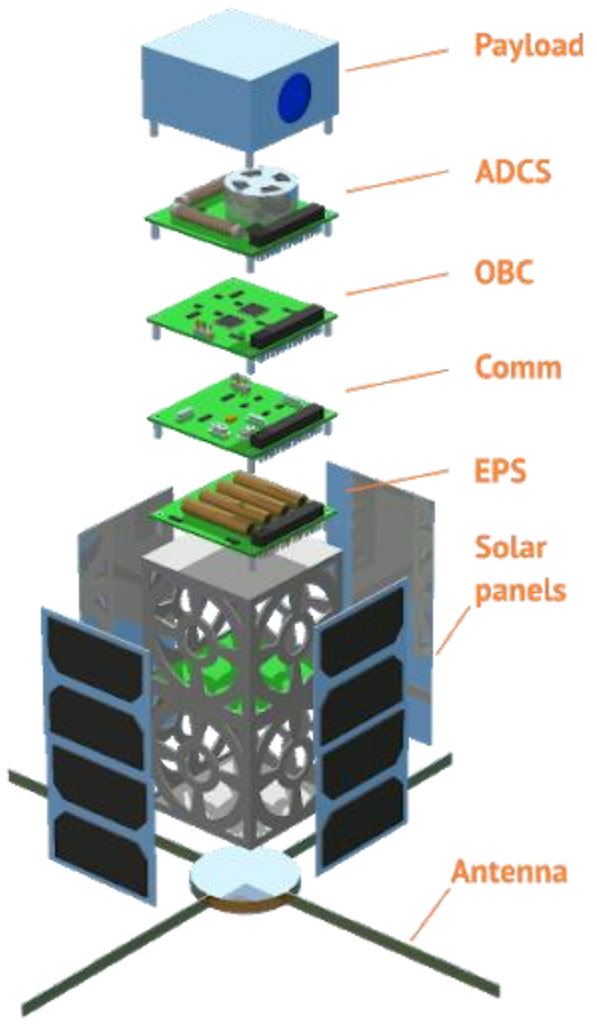


Figure 2: CubeSat structure

1.3.1 ADCS Module

ADCS is responsible for stabilization of the CubeSat on its orbit through a series of actuators and sensors connected in a loop, it ensures the stability, orientation and steadiness of the satellite [13]. The sensors might include earth, sun and star sensors, gyroscopes, magnetometers and directional antennas [14]. The actuators could include magnetic thrusters and wheels which in turn can be reaction wheels, momentum wheels or control moment gyros [14]. Reading the data from all sensors and applying a certain algorithm helps determine the attitude of the satellite and based on that the actuator can perform maneuvers to put the satellite in the right orbit. A feedback control loop is applied between the sensors and the actuators is in play until the satellite reaches the desired state. In order for the CubeSats to produce usable images it needs to be always pointing down to earth, it's quite common to use reaction wheels to achieve 3-axis stabilization [15].

1.3.2 OBC Module

The OBC is main brain of the satellite responsible of the data transfer and link all the modules and subsystems together. The OBC is an embedded system computer focuses on command and data handling (C&DH) of the Cubesat. Also responsible to handle the difference data transfer rates between the subsystems and ensure the security [21]. Temperature and power generation and consumption from the other modules are stored constantly OBC's memory module to be send to the ground station. The ground station can upload new firmware to the OBC through the uplink channel and the OBC will execute it. The OBC handles as well the telecommands and executes them from and to the other subsystems [22]. The OBC as well takes the data from ADCS sensors to determine the satellite orbit [13] as shown in following Figure 3.

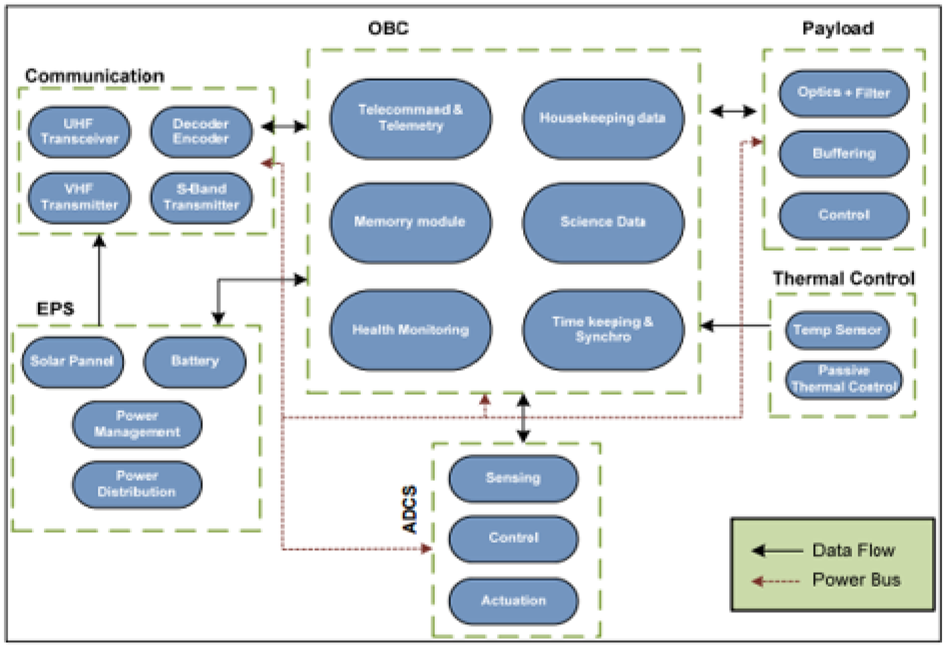


Figure 3: OBC architecture

1.3.3 Payload Module

Types of payloads include telecommunication, imagery and scientific measurements, In order to increase the cost and integration efficiency of the satellite it's common to combine two or more types of payloads [16].

Communication satellite can send and receive three types of payload, voice, video and data, the ground station sends the signal to the satellite and the satellite retransmit it either to ground or another satellite as a relay through a channel at a certain frequency band, commonly used in communication with airplanes, ships, trains and trucks [16].

1.3.3.1 Imaging satellites

The payload onboard imaging satellites consists of a camera which is classified according to the resolution of the image that can be reproduced. The imager is required to be capable of capturing images of Earth or other universal bodies over different spectral bands according to the application. Filtering techniques are then applied to present the image in the usable frequency range. Applications of imaging satellites can range from mapping (cartography), disaster management (fire detection), meteorology or universal observation [17].

1.3.4 Power Module

The EPS perform the following operations:

- Generates power to supply satellite loads with electricity needed during the mission.
- Administer electrical power distribution to the satellite.
- Provide average and peak power demands from loads.
- Generate EPS health and status telemetry data.
- Provides control by ground station or other systems.
- Protect against bus faults by suppressing transient voltages.

The electrical system consists of a photovoltaic solar panels that converts the solar radiation to electricity to be used for the different workloads in the satellite. An eclipse happens when the earth is positioned between the orbiting satellite and the sun that might last between few minutes to few hours, during this period the solar panel doesn't generate electrical power [18]. During the eclipse a secondary source of power is required to support the satellite operations, rechargeable batteries can be an excellent choice, in a study conducted by Clark and Simon (2007) Lithium Ion, Lithium Polymer and Nickel Cadmium batteries are excellent choices for small satellite application.

1.3.5 Communication Module

TT&C (Telemetry Tracking and Command) is a subsystem responsible for the communication between the satellite and the ground systems through an interface or link, RF link is a two one-war channels in air or free space, the uplink channel where the communication from Earth to the satellite to upload telecommands, and the downlink channel where the communication from the satellite to earth to download telemetry. A high gain RF transmitter and receiver antenna onboard the satellite and on the ground station, every time the satellite appears and become visible to the ground station antennas, telecomm and from the ground station is sent to the satellite via the uplink channel.

Similarly the same happens for the communication sent from the satellite to the ground station via the downlink channel to download the satellite telemetry, that happens in a very short amount of time when the satellite pass above the ground station and the receiving antenna have a clear field of view.

1.4 Space Environment Effects

There is a lot of aspects that need to be taken in consideration when designing the system due to environmental perspective, this chapter is discussing these aspects of design considerations, although there are some aspects are not considered relevant that we shall not discuss such as aerodynamic drag and microgravity.

1.4.1 Transient Launch Environment (TLE)

This environment consists of mostly mechanical components that consists of:

- Quasi-static launcher acceleration load required to get to the orbit.
- Dynamic loads launcher caused by solid-booster ignition, travels through a lot of high wind zones and termination of engine thrust for both solid-booster and liquid stage.
- The shocks caused by the separation of the launcher.
- Engine noise acoustic pressure reflecting from the ground.

In order to reduce the effect of the TLE and on the vehicle payloads during launch and enclosure around the payload area need to be introduced, referred to as fairing typically consists of damped material to absorb the shocks and acoustic pressure, it also helps reduce the temperature of the components when it reach higher atmospheric friction. The maximum acceleration during launch usually subjected to 10g or 20g [19] accelerations in can of SUNSAT that contain reaction wheels, IMAGER components that can withstand such accelerations. Sensitivity of the electronic components are not that high expect the connection on the PCB if it’s not soldered properly thus it really important for the electronic components to be soldered by a qualified personnel, shock test and vibration test should be conducted. It’s important for the components to be able to operate in vacuum also that it can withstand working in high temperatures also for the electronic components to be protected from any corrosive agent from the satellite structure. While the satellite is orbiting it might be subjected to micro-meteoroids and orbital debris impact thus the satellite structure design must provide enough protection against such objects [20].

1.5 Radiation Effects on Semiconductors

Radiation can have major impact on the operation of the satellite that can categorized into non-ionizing and Ionizing radiation, the effect of radiation can be either "hard" (permanent damage) or "soft" (temporary damage, normally only results on loss of system state, i.e. information), there are different kind of radiation such as X-ray, Gamma, Alpha, Beta, Proton, Neutron and Cosmic radiation [20] which will be explained in the following table.

Region	Wavelength	Characteristics		
Gamma ray	Less than 0.03nm	Absorbed completely by the upper atmosphere	Not used in remote sensing	
X-ray	Between 0.03nm to 3.0nm	Absorbed completely by atmosphere.		
Ultraviolet	Between 0.03nm to 0.4nm	Absorbed completely by ozone in the upper atmosphere.		
Photographic UV Band	Between 0.3µm to 0.4µm	Transmitted through atmosphere but scattered and can be detected by photodetectors.		
Visible	Between 0.4µm to 0.7µm	Imaged with photodetectors. Including the reflected energy peak of earth at 0.5 µm.	Visible & Near-Infrared (VNIR) remote sensing	
Infrared	Between 0.7µm to 100µm	Atmospheric transmission windows are separated by absorption bands.		
Reflected IR	Between 0.7µm to 3.0µm	Reflected solar radiation that contains no band information about thermal properties of materials. The *photographic IR band* (0.7µm to 0.9µm) is detectable with film.	Short-Wave Infrared (SWIR) Remote Sensing	
Thermal IR band	Between 3µm to 5µm and between 8µm to 14µm	Principal atmospheric windows in the thermal region. Images at these wavelengths are acquired by optical-mechanical scanners and special vidicon systems but not by film.	Thermal Infrared (TIR) Remote Sensing	
Microwave	Between 0.1cm to 30cm	Can penetrate clouds, fog, and rain.	Active or Passive	Microwave Remote Sensing

Radar	Between 0.1cm to 30cm	Radar images are acquired at different wavelength ranges.	Active
Radio	wavelengths longer than 30cm	Longest wavelength portion of electromagnetic spectrum. Some classified radars with very long wavelength operate in this region.	

Table 1: Electromagnetic spectral regions

1.5.1 Radiation Regions

The are some regions that have a unique radiation characteristics such as low energy solar winds protons and medium to high energy solar flares and trapped electrons and proton in Van Allen belts, except chip capacitors electronic components are manufactured to avoid such radiations. The Following Figure describe (a) The Earth's magnetosphere showing the Van Allen radiation belt. (b) Outer and inner (proton) belt.

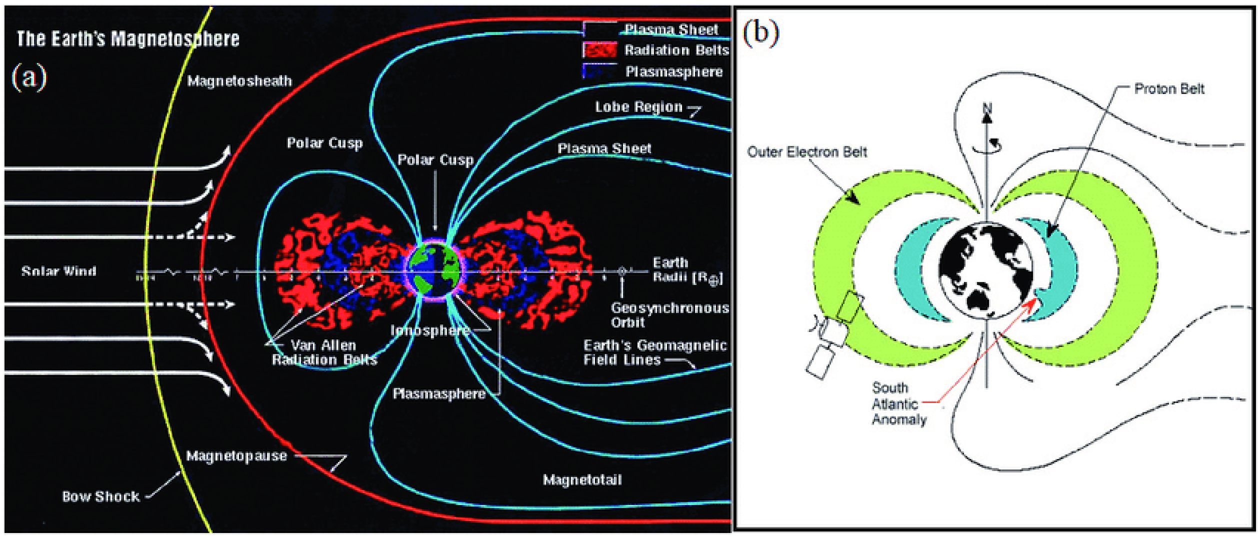


Figure 4: The Impact of Space Radiation on Satellites [23].

1.5.2 Macro Effects on Semiconductors

1.5.2.1 Single Event Latehup (SEL)

Semiconductor latchup happens when the device goes to an anomalous state where it stops responding to input signals, this latechup happens typically because of a parasitic transistor [20].

1.5.2.2 Single Event Upset (SEU)

A single event upset is an isolated logic errors where unwanted change of information stored in the memory happens due to spurious charge generated by the single ionizing particle, smaller devices with higher packaging density increase the SEU susceptibility of the service [20].

1.5.2.3 Single Event Functional Interrupt (SEFI)

This happens when an upset in the control circuit on the device occurs, normal operations will be halted subsequently to go to test mode, and the device would need power to rest to recover from such hated or undefined state [20].

1.5.3 Radiation Hardness Definitions

The radiation tolerance of the systems are given with respect to:

- Total Ionization Dose (TID) ability to withstand accumulated doses of radiation.
- Transient does (TD) high radiation dose rates.

- Single event effect (SEE) measure of the sensitivity of a device to radiation.

1.6 Thesis outline

- Chapter 1: introduction and background: describing the Cubesat history and submodule overview.
- Chapter 2: Hardware Selection: discusses the selection hardware based on the suitable microcontroller
- Chapter 3: Design and implementation: discusses the architecture and the software and hardware design
- Chapter 4: Future work

2 Hardware Selection

2.1 Introduction

There are different aspects that needs to be considered while choosing a microcontroller like Power consumption, Temperature, Operating voltage and I/O Serial bus compatibility. There are also different variants of microcontrollers supporting 8, 16 and 32-bit word length. In CubeSat 32-bit is not preferred as the real time applications are not very critical dependent on speed, power or memory and it sufficient enough to handle the data. The more the applications become complicated and require more processing power, it becomes very necessary to migrate to 32-bit microcontroller, although the complexity remains a major issue, that's why the CubeSat OBC will be developed with 16-bit microcontroller.

2.2 Selection criteria

The selection criteria depends on the requirements of the satellite mission, in this study the requirements is based on a fillable mission while developing the OBC architecture, however it's possible to find common attributes for the hardware for the next mission. For both hardware and software of the OBC is intended to do a particular function it's important to take in considerations some constraints [24][25]. Those constraints will be discuss in the following sections.

2.2.1 Power consumption

It’s really important for the CubeSat to operate on a low power considering the minimal external surface area covered by the solar panel [26], the CubeSat has total available power budget of 1W for 1U and 5W for 3U, in addition to that the microcontroller should be able to save power by disabling peripherals when are not in use.

2.2.2 Operating temperature

The CubeSat operating temperature can vary a lot based on the location of the satellite from the sun while orbiting, in the closet point to the sun temperature can reach up to 150°C and when it’s at the furthest point from the sun it can go as below as -150 °C [27]. These temperature extremes might cause damage to the OBC components thus the CubeSat OBC components should be rated for operating range between -40°C to 85°C which is the standard for industrial electronic equipment.

2.2.3 Operating voltage

The CubeSat EPS provides unregulated bus voltage ranges between 6v and 8.3v, the active components should be chosen to operate below or equal the value of 3.3v so we don’t have to add extra regulators that would increase the weight of the CubeSat.

2.2.4 Packaging

The different consideration of the chosen components and the way they are mounted will have an impact on the size and weight of the OBC. Surface mount technology (SMT) is preferred over dual in-line package (DIP) due to the volume impact, Small-outline integrated circuit (SOIC) and quad flat package (QFP) are preferred over pin grid array (PGA) and land grid array (LGA) for ease of integration during the soldering process.

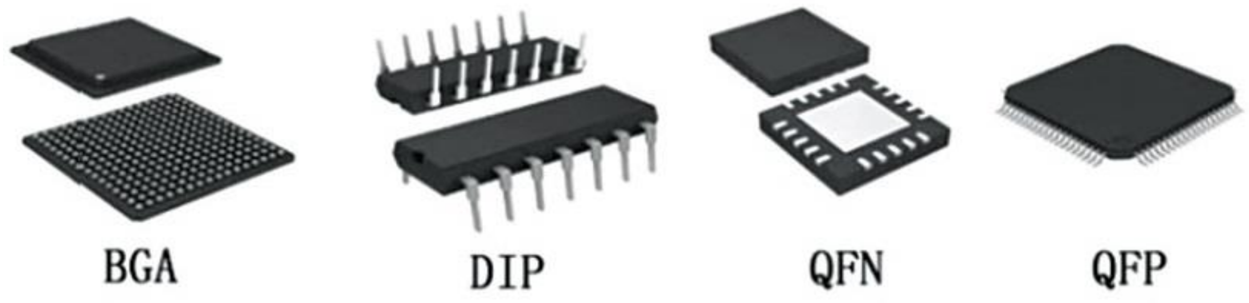


Figure 5: Some IC packaging types

2.2.5 I/O and serial bus compatibility

It’s necessary to have digital and analog I/Os available in the microcontroller to be able to interface with the final OBC prototype with all the other subsystems and is directly related to the microcontroller chosen. The OBC’s microcontroller’s integrated serial interfaces (UART, I2 C or UART) makes it possible to transfer data between subsystems, it’s highly recommended to choose a microcontroller which possesses one or more of each of the serial interfaces [24].

2.3 Selection process

The microcontroller is considered the core of the OBC hardware, it consists of CPU, Memories, Timers, Interfaces and Peripherals for interconnections.

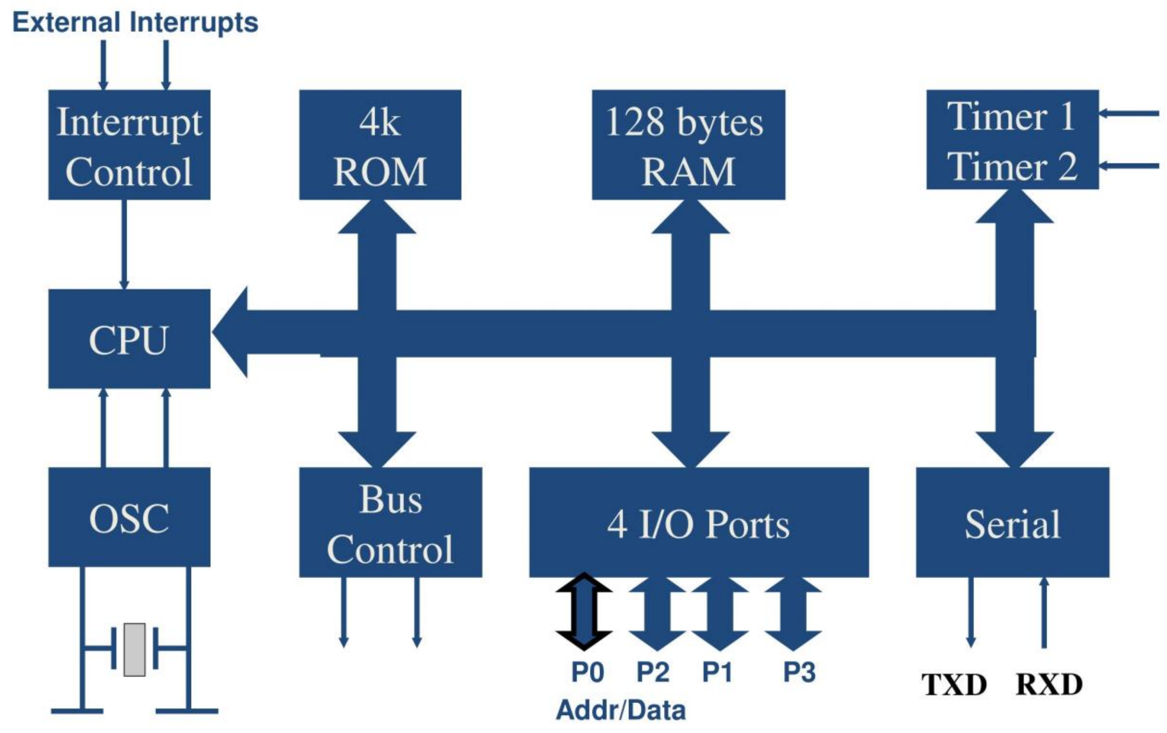


Figure 6: Example of a block diagram for microcontroller

Based on the mission of the CubeSat and how fast the operations need to be processed choosing between 8, 16 and 32-bit word length can be chosen.

2.3.1 Upgrading from 8-bit and 16-bit to 32-bit

The trend of upgrade the 32-bit is mainly driven by the need to deliver increased processing power and flexibility in code reuse across projects using high level languages. These lacked in the 8-bit and 16-bit architectures.

In the past, CubeSats have mostly used 8-bit and 16-bit microcontrollers. 32-bit core based microcontrollers were initially introduced in 1985 by Intel and because of the complexity they presented and because many of the embedded and real time applications at the time were not critically dependent on memory, power or speed and the amount of data to handle was sufficient.

Today, the 32-bit core architecture's complexity has been significantly reduced and has been made efficient and capable of handling 8-bit, 16-bit and 32-bit instructions and data. In addition to the reduction in complexity, multiple features have been added and customized by different core manufacturers.

2.3.2 History of 32-bit implementations in Cubsats

The 32-bit core architecture has been successfully tested on CubeSats for several missions, such as NCube2 and CanX-1. NCube2 for example is product of the Norwegian University of Science and Technology and its payload consisted of an automatic identification system for the marine industry where the AIS protocol which is a standard for tracking ships, is used to identify, position and exchange data messages between ships. Its C&DH OBC used a 32-bit ATmega32L from Atmel with an AVR32 core and 4kB of ROM.

Although contact was never established after launch, the flight-model operated properly on the ground [28]. CanX-1 from the University of Toronto was launched with three experimental payloads onboard: a low cost CMOS horizon sensor, a star tracker and a GPS receiver. Its C&DH OBC used the AT91SAM7 microcontroller which is a 32-bit ARM7 core, from Atmel with 512kB of static random access memory (SRAM), 32MB of Flash-RAM and with a maximum operating frequency of 40MHz [29]. Some examples of other Cubesats can be found in the Appendix.

2.4 EFM32™ Gecko 32-bit Microcontroller

The EFM32™ Gecko MCU was selected due to its efficiency and it also offer all the features needed to design the OBC for a CubeSat based on the selection criteria. The following Figure 7 shows the block diagram of the EFM32™ on system level. More features on EFM32™ in the Appendix.

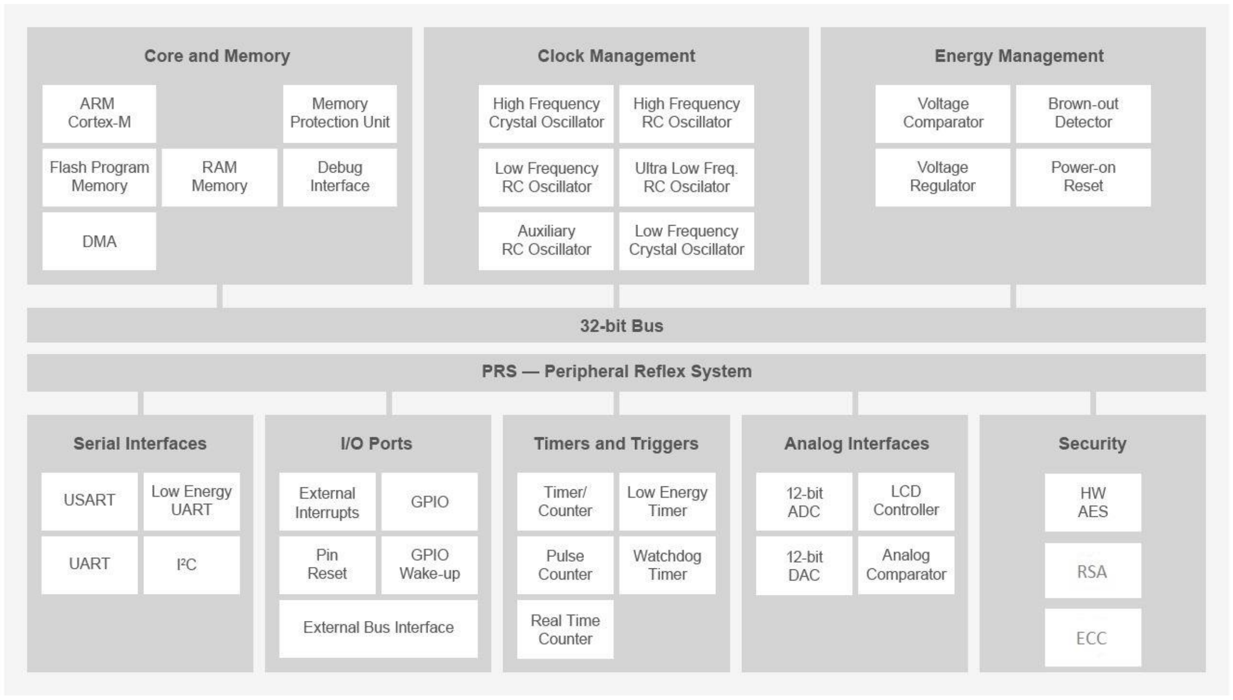
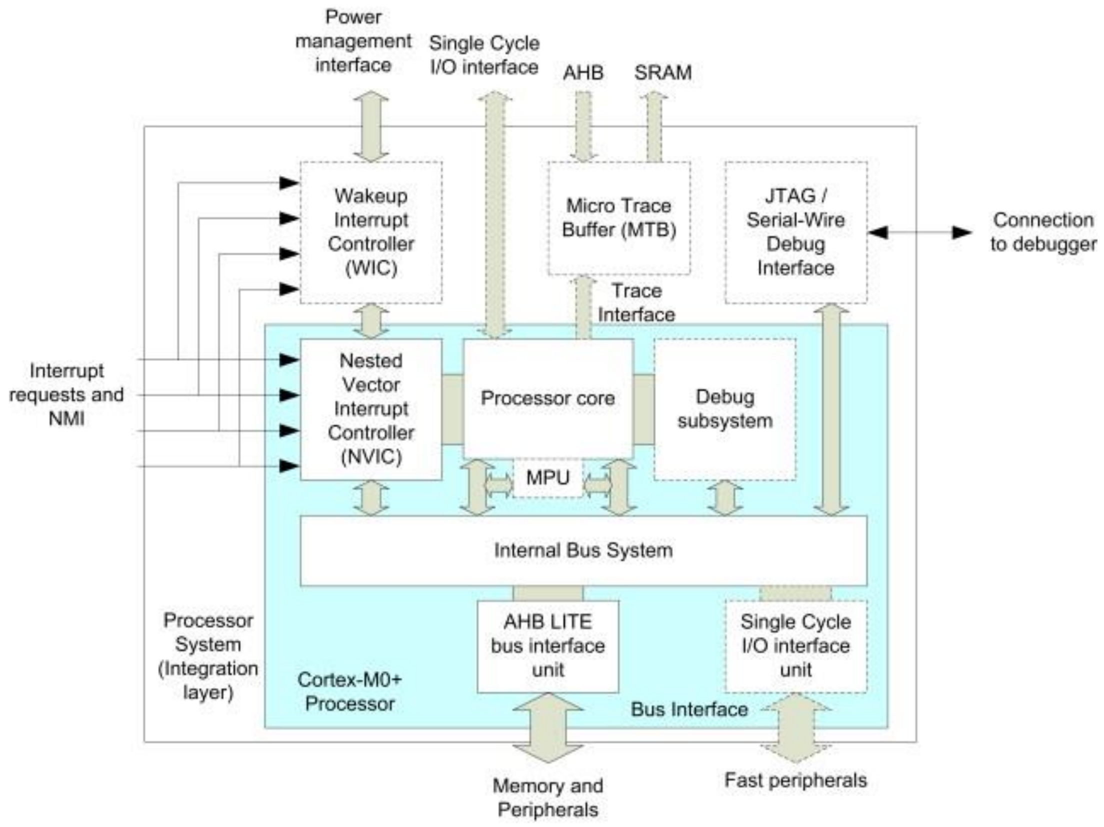


Figure 7: Block Diagram of EFM32™ Gecko

3 Software Design and Implementation

The software is responsible for giving the instructions to be executed by the hardware of an OBC thus it's as important as the hardware design if it's not more, these instruction is responsible for telling the hardware how and when to execute a certain operation, the instruction also called program are quite complex in nature and there are factors that need to be taken in considerations such as performance, reliability and efficiency and they apply on both hardware and software.



Power management interface
Single Cycle I/O interface
AHB
SRAM
Wakeup Interrupt Controller (WIC)
Micro Trace Buffer (MTB)
JTAG / Serial-Wire Debug Interface
Connection to debugger
Trace Interface
Interrupt requests and NMI
Nested Vector Interrupt Controller (NVIC)
Processor core
Debug subsystem
MPU
Internal Bus System
Processor System (Integration layer)
Cortex-M0+ Processor
AHB LITE bus interface unit
Single Cycle I/O interface unit
Bus Interface
Memory and Peripherals
Fast peripherals

Figure 8: ARM Cortex-M3 architecture

In this chapter discuss the low-level perspective of the software touching on the hardware abstraction of the ARM Cortex-M3 architecture as described in the Figure 8, a summary on the drivers that were developed for the hardware that was designed, after that discussing the error detection and correction algorithm implemented on the FPGA for the SRAM, finally a high-level software will be discussed.

The operating system consists of four layers all of them inside the Kernel. The number of layers are minimized to avoid extra usage of stack. The following figure 9 will describe the overall software architecture.

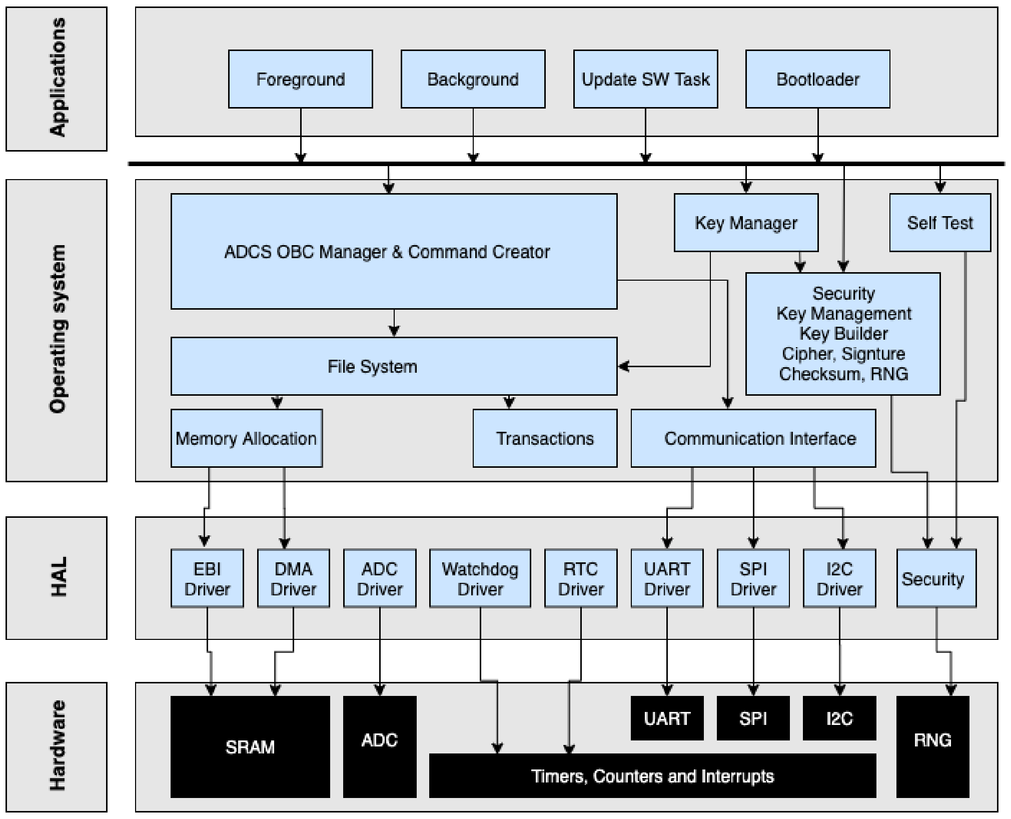


Figure 9: The Operation system Architecture

3.1 Hardware Abstraction Layer (HAL)

The Cortex-M3 Microcontroller Software Interface Standard (CMSIS)[30] is a Hardware Abstraction Layer (HAL) which is a low-level software that acts as an interface of the hardware from different manufacturers, the ARM standard is CMSIS, as shown in the Figure 9 the interface facilitate the access to the processor and peripherals on all microcontrollers. Having a software community behind the ARM-based microcontroller is quite advantageous and having Cortex-M community becoming vendor independent is a big plus, all that greatly improves the quality of code and makes a lot of reusable code available.

For this design Cortex-M3 based MCU is used and its software is based on ARM CMSIS, currently there is a large pool of application code available which can be found on their website [31].

In order for the operating system to be able to utilize the hardware without going to the low level details of the hardware implementation, a driver is needed witch it abstract the low-level implementation of the hardware and provide an interface where the operating system can utilize.

The EFM32™ has a lot drivers that can be found however these driver is modified or re-developed to be more suitable for the current project requirement, in the following sections the drivers developed will be listed to be used by the OBC.

3.1.1 External Bus Interface

The External Bus Interface (EBI) is the interface that deals with the external memory flash and SRAM on the OBC, this driver is responsible for reading and writing in these memories and on certain location. Data and address lines also can be multiplexed, to reducing the pins' number required to connect external devices.

3.1.2 Direct Memory Access

The Direct Memory Access (DMA) is the interface that deals with transferring data between peripherals and the memory of the MCU without intervention, thanks to the DMA the MCU can sustain lower energy for longer periods of time that lead to better efficiency.

Figure 10 shows the implementation of the DMA, a channel between peripheral/memory and memory/peripheral need to be sat up and a call had to be made to signal the start of the transfer. The transfers remaining numbers are determined by this value N_MINUS_1 and the looping functionality not exist on the EFM32™ Gecko so will consider the EN bit to be 0 always in this board. Some terminology:

- N: is the number of transfers needed. And can be Calculated by

  N = (N_MINUS_1 in the beginning of transfer) + 1
- Arbitration Rate: $2^{\mathrm{R\_POWER}}$

- REQ and SREQ: those are the request signals selected by DMA_CHx_CTRL

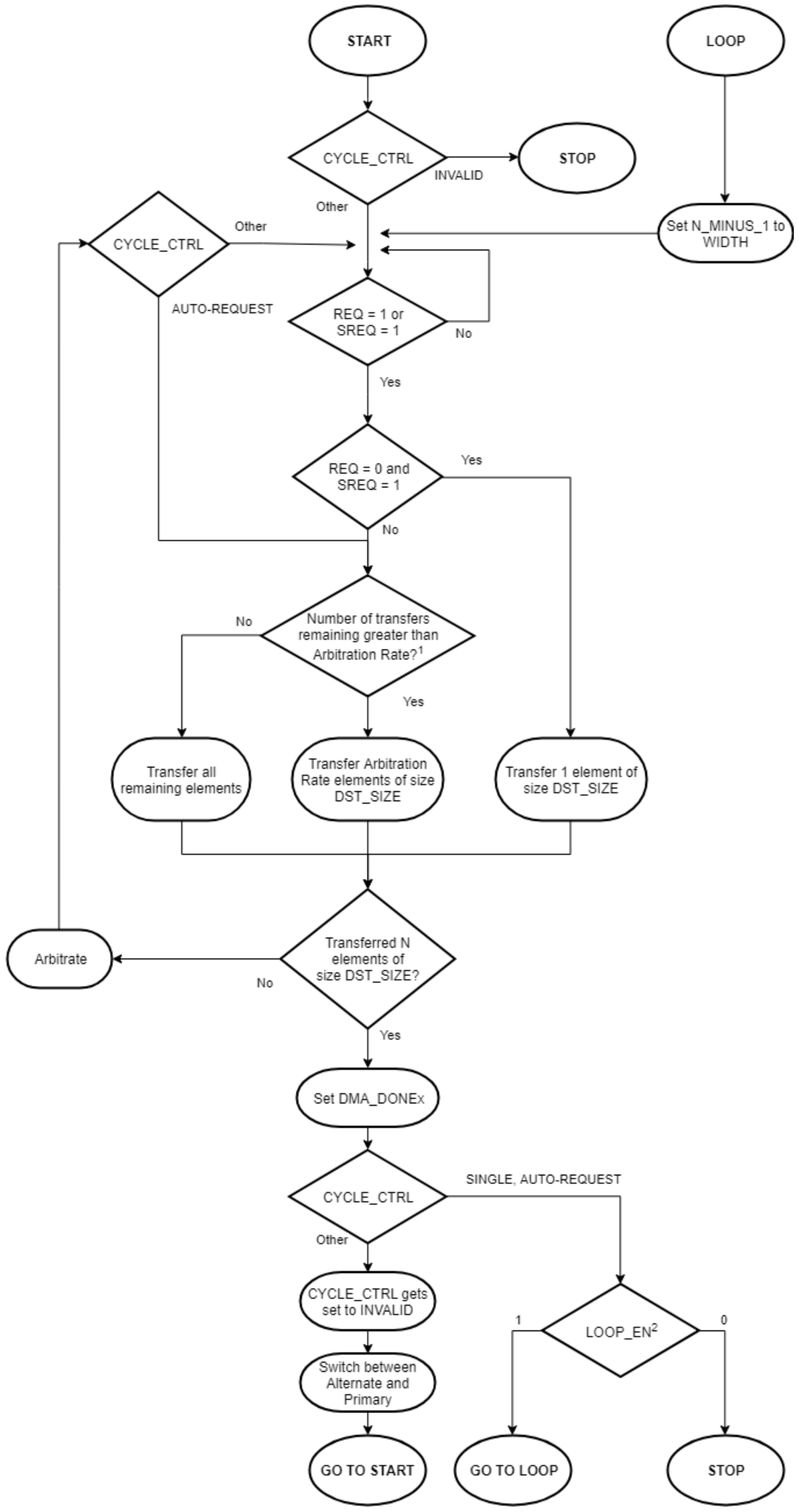

START
LOOP
CYCLE_CTRL
INVALID
STOP
Other
Set N_MINUS_1 to WIDTH
CYCLE_CTRL
Other
AUTO-REQUEST
REQ = 1 or SREQ = 1
No
Yes
REQ = 0 and SREQ = 1
Yes
No
Number of transfers remaining greater than Arbitration Rate?1
No
Yes
Transfer all remaining elements
Transfer Arbitration Rate elements of size DST_SIZE
Transfer 1 element of size DST_SIZE
Arbitrate
Transferred N elements of size DST_SIZE?
No
Yes
Set DMA_DONEx
CYCLE_CTRL
SINGLE, AUTO-REQUEST
Other
CYCLE_CTRL gets set to INVALID
Switch between Alternate and Primary
LOOP_EN2
1
0
GO TO START
GO TO LOOP
STOP

Figure 10: Flow chart of single channel transfers in the EFM32 Series 0 DMA

3.1.3 Real Time Clock

The Real-time clock (RTC) is responsible for the timekeeping system, more important when we have more than one OBC we need to avoid any conflicts, implementation and synchronization throughout the satellite requires subsystems, OBC and ground station.

RTC uses external crystal oscillator for stability over RC oscillators, it creates an interrupt every second to tick, and the RTC can use UTC timestamp which is popular in software programs.

3.1.4 Analog to Digital Converter

The Analog to digital converter (ADC) in the interface that deals with sampling the analog channels for the monitoring subsystem including the supply voltage, load currents and temperature, the implementation of the ADC on the OBC is either periodic or continuous.

- *Periodic*: The sampled channels are scanned and stored periodically with maximum frequency.
- *Continuous*: the sampled channels are monitored by analog comparator compared to a threshold an interrupt is generated and the MCU react accordingly.

Table 2 shows the different functions supplied by the ADC drive.

Function	**Description**
BSP_ADC_Init	Initializes the clock, DMA channel and settings for the ADC.
BSP_ADC_Scan	Starts the ADC and scan of all channels.
BSP_ADC_IsScanComplete	Returns true if a scan is still in progress, otherwise returns false.
BSP_ADC_GetAllData	Returns all the values from the latest ADC scan.
BSP_ADC_GetData	Returns the latest value of the specified ADC channel

Table 2: ADC Drive Functions

3.1.5 Watchdog

The Watchdog safeguards the MCU against a software lockup that will lead to the MCU and the OBC to become unresponsive, its designed with two watchdogs, internal and external where both have to be periodically toggled so that the MCU doesn't have to reset. The two watch dogs are toggled in separate areas in the operating system foreground and background to prevent the MCU from entering a logic loop.

3.1.6 UART

The UART is the interface for communication debugging purposes, it’s ideal for testing inter-subsystem communications, protocols and synchronization, it is simple to interface with a personal computer and emulate any other subsystems which is very important for debugging without having the full satellite system available, Communicating with the PC signals should be converted to RS232 voltage levels used by the serial port as in the Figure 11.

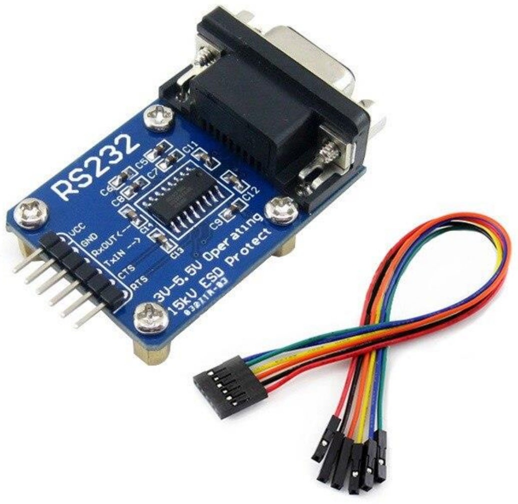


Figure 11: RS232 to UART converter

3.1.7 SPI

The Serial Peripheral Interface (SPI) is the interface that deals with the microSD card, it offers very fast synchronized link between the MCU and the storage, the MCU has two SPI interface that can be used for large point to point data transfer for example transfer images between the ADCS and the CubeSense (CMOS-based Sun and Earth sensor) for debugging purposes. The file format which the microSD is using is FATFS, data accessed through disk I/O driver that is especially modified driver only intended for FATFS.

3.1.8 I2C

The Inter-Integrated Circuit (I2C) is the interface that deals with the main communication bus that is used mainly by the OBC to communicate with all the other major subsystems.

As the OBC interacts with its subsystems, the main I2C bus must allow for multi master usage, it requires arbitration and synchronization to simultaneously communicate with main OBC, it would be much easier if each OBC is allowed a time slice within witch it communicate with it subsystems while acting as slave for the main I2C bus tele-commands as described in Figure 12.

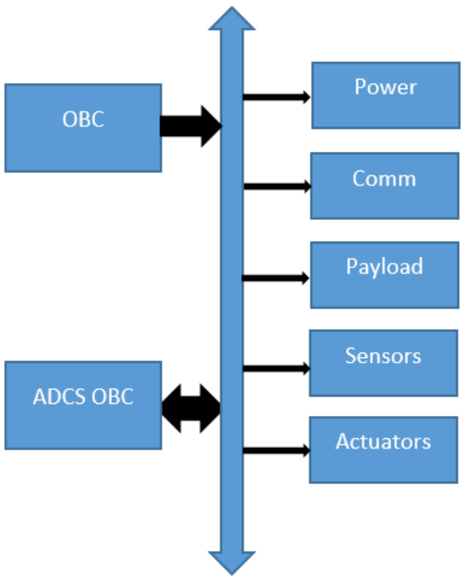


Figure 12: Multi Master Bus

3.1.9 Security Driver

To handle the hardware for the random number generator (RNG) and other security sub-microcontrollers.

3.2 Operating system

The designed OS is a Real-Time Operating System (RTOS) [32], as described by [41], is developed to handle and execute multiple tasks in the real time. It uses a different scheduling algorithm first comes first Serve (FCFS), shortest job first (SJF), priority scheduling Round Robin scheduling to ensure each task receives an equal amount of processing time from the MCU core. This scheduling ensures the tasks processed closer to “real time”. The RTOS is suitable the main OBC. The RTOS have different responsibilities as described previously in Figure 9.

3.2.1 File System Module

File system is the way of storing the data on physical storage devices like disk, magnetic tapes, compact disk, flash drives etc. so, the operating system organizes the data and manage the file system in order to retrieve the stored data with a fast, reliable, secure, efficient, scalable and fault tolerant way [33].

The need for data exchange has increased and to have efficient transmission and fast storage of data the first step is to select appropriate memory and it’s necessary to transplant open FATFS file management module to single-chip embedded systems. It’s completely free and open source FAT file system and it doesn’t depend on I/O layer and implemented in C language so it has a lot of characteristics such as independency and portability of the hardware, and a separate buffers that can be used for multiple read-write texts [34]. FATFS also is designed to read and write in a very special way, which makes it high reading and writing efficiency.

3.2.2 Memory Allocation Module:

It's a hardware operation and managed by the operating system, programs and services are assigned to memory based on requirements when they are executed, and once the program has finished its operation or Idle the memory is released and allocated to another program. Memory allocation is using the following functions to operate its work:

3.2.2.1 Memory Read

Read from memory address and transfers it to the requester/process.

3.2.2.2 Memory write

Store the data bits in memory from data input lines. Then it activates the write control line to be written in the memory.

3.2.2.3 Memory interleaving

Processor requests Data from the memory. Data chunks is read by the cache from memory then processor. Therefor any missed chunks can be fetched again from the memory to the cache. But the memory is usually slower than cache so to recover and improve the data access time interleaving is used.

3.2.3 Security Module

The Security Module has many Responsibilities:

- Off-Sat entity authorization.
- Initiation, open and close the secure channels.
- Unwrapping the incoming commands.
- Wrapping the outgoing responses.
- Encrypt & decrypt data.
- Set the security level for incoming command or outgoing response.
- Generate Keys, sign and verify data.

3.2.4 Selftests and Error Detection Module

The Error Detection and Correction (EDAC) is implemented to test the data bus between the MCU and the SRAM and MCU in real-time without affecting any of the reading or writing commands of the MCU. This will add no extra memory access overhead to the MCU.

3.2.5 Communication Interface

This module provides an implementation the some of the communication protocols like Message Queuing Telemetry Transport (MQTT) and protocol buffer (protobuf). It also provides a high-level interface for the communication process and isolates the application layer from the details of the communication protocol.

3.2.6 Monitor Service Module

The monitor service is used to monitor set of tasks that might not be essential for the goal of the applications but it's essential for the operation of the OBC, Tasks for housekeeping, emergency responses and interrupts. Those tasks are managed by the command Manager.

Interrupt Service Routine (ISR) is the interrupt executed by the monitor service, each ISR is developed to execute rapidly as this helps the main ADCS loop to execute deterministically, and quick ISR also avoid the probability of data loss. The monitor service for this design are as follows:

3.2.6.1 RTC Counter

Every second the RTC counter generates an interrupt. The RTC ISR increments the timestamp of the system, which indicates when some tasks, such as the control loop, are scheduled to start.

3.2.6.2 I2C Communication

When the I2C receives a data packet it generates an interrupt, the data could be dropped if the transmission is not dealt with immediately, the tele-command from the main OBC could contain settings such as orientation schedules and so , it's would be quite undesirable if these data is dropped and should update the OBC as soon as possible.

3.2.6.3 SRAM SEU Detection

When the board send a signal indicating and error the SRAM SEU detection it will generate an interrupt, in case of correction the ISR logs the error and schedules a memory clean up, if multiple uncorrectable errors are detected the ISR logs the error and performs a variable integrity check, If a critical variable is corrupt and cannot be corrected, the MCU is reset.

3.2.6.4 SRAM SEL Detection

When the SRAM power supply exceeds a certain threshold (latchup) value, The SRAM SEL detection generates an interrupt, in attempt to fix the SEL the ISR immediately power cycles the SRAM module, the OBC will switch to the backup SRAM module for normal operation, If the SEL persists.

3.2.7 Process Management Module

The process management is the process where the scheduler prioritize the list of jobs received by the ground station, there are several way to go about the prioritization and that depends on the mission requirements of the CubeSat, the different ways to decide on the priority can be based on when the job is queued, how long does it take for the job to execute, a set priority sent alongside with the job or distributed in a round robin fashion.

3.2.7.1 First Come First Serve (FCFS)

In this method the priority of the job is set based on when the job is received, it is quite useful when the order of the jobs are critical, it is also the simplest way of managing the process, but that comes at the cost of the overall performance, The CPU handles only one job at a time regardless of the long does it take to complete, when the CPU becomes free, it gets assigned to the process at the beginning of the queue, this method supports non-preemptive and preemptive scheduling algorithm.

3.2.7.2 Shortest Job First (SJF)

This method improves the overall performance over the last method where the shortest job to complete is picked up first, it significantly decreases the awaited time on average for the process waiting in queue, this scheduling method can be preemptive or non-preemptive.

3.2.7.3 Round Robin (RR)

This method's name comes from the round-robin principle, it's one of the oldest and simplest algorithms to implement, the biggest advantage to this method is that it allows all the process in the queue to take turns in the process but that happens on the cost of spending time context switching, this is a preemptive algorithm.

3.2.7.4 Priority scheduling

In this scheduling algorithm the process is scheduled based on the priority of the task, the process with higher priority get carried out first, where the jabs that have equal priority get carried in a First come first serve (FCFS) or Round Robin (RR), the disadvantage of this method is that if process with high priority take a long time to execute, the average waiting time for the rest of jobs increase, this method can be preemptive where the tasks are mostly assigned with their priorities or non-preemptive where the CPU has been allocated to a specific process.

3.2.8 MQTT Module

Message Queuing Telemetry Transport (MQTT) is a standard lightweight messaging protocol that is widely used in so many industries that requires communication with minimal network bandwidth, it is efficient, reliable and bi-directional, MQTT standard is v5.0 and v3.1.1 are now OASIS standards (v3.1.1 has also been ratified by ISO) [35]. The Mqtt module that the operating system have are responsible to create the commands in Mqttt format.

3.2.9 Protobuf Module

Protocol buffers (protobuf) is a way for serializing structured data, it's developed by google and widely supported by so many programming language, it's smaller, faster and simpler than other format such as xml or json. The Protobuff module responsible for serialize and unserialize the data to be send to the MQTT module.

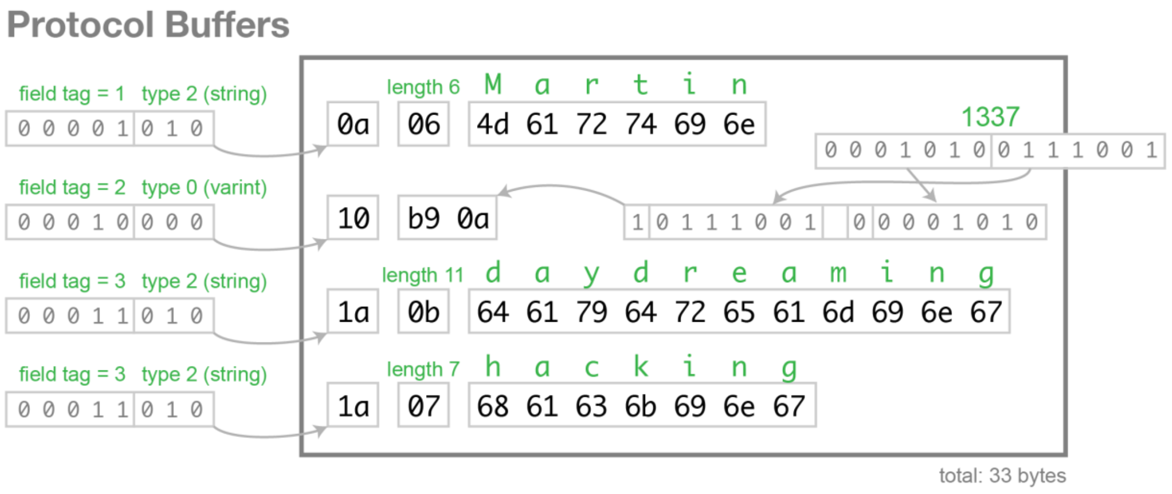


Figure 14: Protobuf example

3.2.10 Command Manager Module

This module responsible for creating the different commands and send it to the right module or to the ground station though the communication module and the focus will be on the commands that's will be sending between the ground station and the CubeSat. The CubeSat will always be the client and ground station will be always the broker in the MQTT messages.

3.2.10.1 OnBoarding Process

To start the communication between the ground station and the cube satellite, an on boarding process has to be initiated first, this process is necessary for the ground station to register the cube satellite and be able to communicate with in a secure fashion, the communication between the ground station and the cube satellite happens over

MQTT protocol where the ground station sends MQTT messages/commands and receives response, during this phase an initial MQTT message is sent to cube satellite with the manufacturer certificate and receives a CSR (certificate signed request). In Figure 15 will describe how the process works.

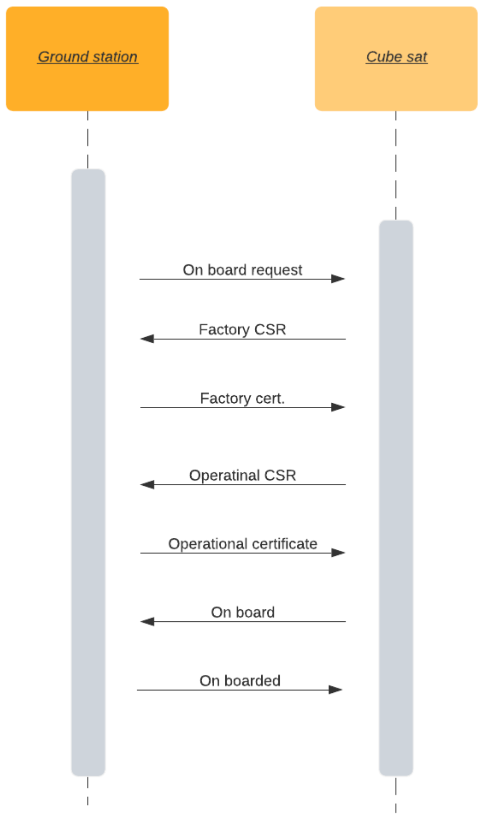


Figure 15: OnBoarding process

The onboarding process will be in 7 steps as described in the following table

Step1: Onboard Request From ground station to CubeSat

Protobuf	`message OnboardRequest {` `required string CubeSatID = 1;` `required string GroundSatationID = 2;` `}`
MQTT Topic	/GroundStation/OnboardRequest/RandomNumber
Step2: Factory CSR	From CubeSat to ground station
Protobuf	`message CSRRequest {` `required string CSR = 1;` `required enum type = 2;` `}`
MQTT Topic	/CubeSat/FactoryCSRRequest/RandomNumber
Step3: Factory Cert	From ground station to CubeSat
Protobuf	`message Certificate {` `required string certificate = 1;` `}`
MQTT Topic	/GroundStation/FactoryCertificate/RandomNumber
Step4: Operational CSR	From CubeSat to ground station
Protobuf	`message CSRRequest {` `required string CSR = 1;` `required enum type = 2;` `}`
MQTT Topic	/CubeSat/OperationalCSRRequest/RandomNumber
Step5: Operational Cert	From ground station to CubeSat
Protobuf	`message Certificate {` `required string certificate = 1;` `}`
MQTT Topic	/GroundStation/OperationalCertificate/RandomNumber
Step6: Onboard	From CubeSat to ground station

Protobuf	`message OnboardResp {`
	`  required enum SCUESS = 1;`
	`}`

MQTT Topic	/CubeSat/OnboardResp/RandomNumber
Step7: Onboarded	From ground station to CubeSat

Protobuf	`message OnboardResp {`
	`  required enum SCUESS = 1;`
	`}`

MQTT Topic	/GroundStation/OnboardResp/RandomNumber

Table 3: OnBoarding Process

3.2.10.2 OffBoarding Process

The offboarding process is the opposite of the on boarding, the target of the offboarding is to end the communication between the satellite and the ground station.

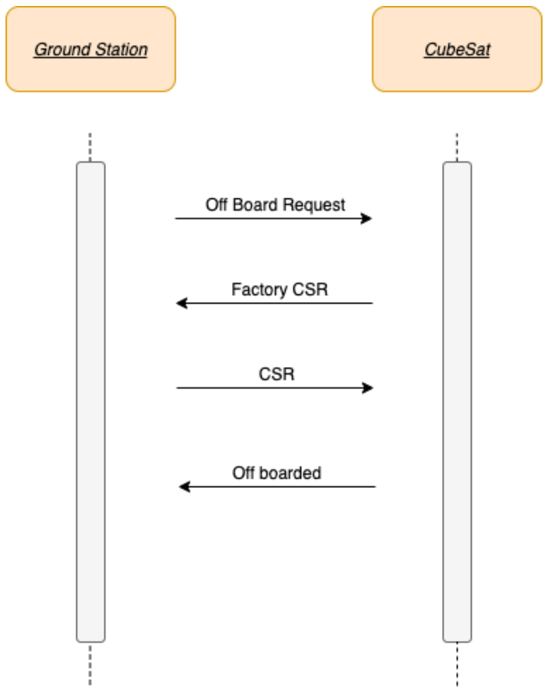


Figure 16: OffBoarding Process

The offboarding process will be in 4 steps as described in the following table

Step1: Onboard Request	From ground station to CubeSat

Protobuf	`message OffboardRequest {`
	`  required string CubeSatID = 1;`
	`  required string GroundSatationID = 2;`
	`}`

MQTT Topic	/GroundStation/OnboardRequest/RandomNumber
Step2: Factory CSR	From CubeSat to ground station

Protobuf	`message CSRRequest {`
	`  required string CSR = 1;`
	`  required enum type = 2;`
	`}`

MQTT Topic	/CubeSat/FactoryCSRRequest/RandomNumber
Step3: Factory Cert	From ground station to CubeSat

Protobuf	`message Certificate { required string certificate = 1; }`
MQTT Topic	/GroundStation/FactoryCertificate/RandomNumber
Step4: Offboard	From CubeSat to ground station
Protobuf	`message OffboardResp { required enum SCUESS = 1; }`
MQTT Topic	/CubeSat/OffboardResp/RandomNumber

Table 4: OffBoarding Process

3.2.10.3 Sending Data Process

This process can’t be done unless the CubeSat is onboarded the data can be photos, configurations, sensors’ data or errors data. The ground station is always responsible to request those data from the CubeSat as following table.

Step1: Data Request	From ground station to CubeSat
Protobuf	`message DataRequest { required string CubeSatID = 1; required string GroundSatationID = 2; }`
MQTT Topic	/GroundStation/DataRequest/RandomNumber
Step2: Data Response	From CubeSat to ground station
Protobuf	`message DataResponse { required string data = 1; required int size = 2; required enum type = 3; }`

MQTT Topic	/CubeSat/DataResponse/RandomNumber

Table 5: Sending Data Process

3.2.10.4 Getting Data Process

This process can’t be done unless the CubeSat is onboarded the data in this process will be send from the ground station to the CubeSat which is the opposite of sending data process, the data configurations or software update. The ground station is always responsible to send those data to the CubeSat.

Step1:Getting Data Request	From ground station to CubeSat

Protobuf	`message GettingDataRequest {`
	`  required string CubeSatID = 1;`
	`  required string GroundSatationID = 2;`
	`  required string data = 1;`
	`  required int size = 2;`
	`  required enum type = 3;`
	`}`

MQTT Topic	/GroundStation/GettingDataRequest/RandomNumber
Step2: Getting Data Response	From CubeSat to ground station

Protobuf	`message DataResponse {`
	`  required enum SUCESS = 1;`
	`}`

MQTT Topic	/CubeSat/GettingDataResponse/RandomNumber

Table 6: Getting Data Process

3.3 Applications

The Application layer is the layer that deals with different tasks that is executed by the OBC, tasks fall into three different categories, Executer, Updater and bootloader, in Figure 13 the executer and monitor service are shown:

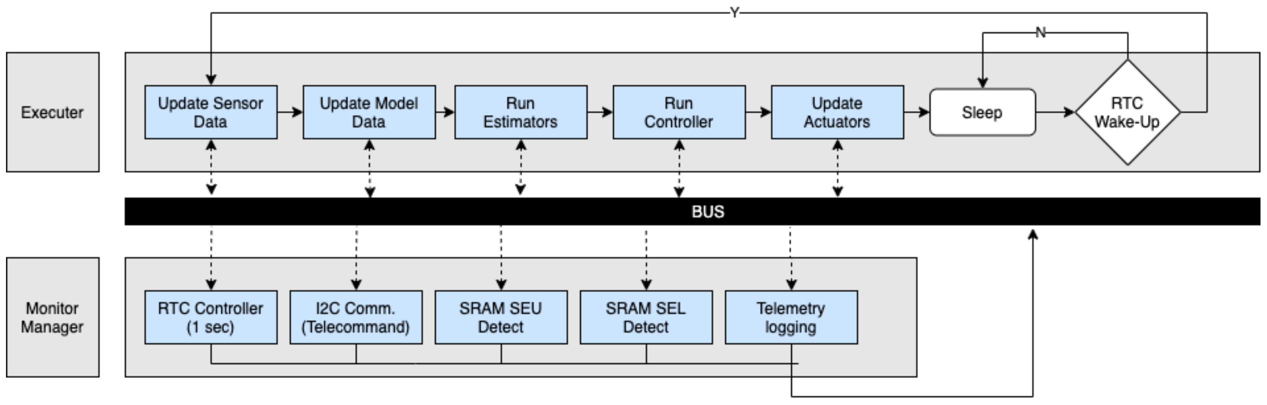


Figure 13: Excuter-Monitor Manager Flow Diagram

3.3.1 Executer

To achieve a single goal a collection of executer tasks aim to archive that goal, by utilizing different subsystems/components of the OBC such as memory, communications, core etc. The executer periodically executes the ADCS control loop for this design it starts reading the most recent sensor values and based on that the control algorithm start executing actuators commands the orient the satellite accordingly, then the foreground application is put to sleep mode until its next schedule time.

3.3.2 Updater

There are two types of software update remote update and local update. The local update means updating the system locally before launching the CubeSat for testing purposes. The remote update called over the air update (OTA). The updater responsible for the update the firmware process and handle all the issues of speed of transfer and the errors during the transfer and send the firmware to the transaction module and after receiving all the data the transaction module will verify the system and send it to the memory for replacement the old version.

3.3.3 Bootloader

A bootloader is a small application, designed to download as well software to the board but the differences between the bootloader and the updater is that the updater updates the whole of the firmware at once but the bootloader updates part for the system only in case of a bug fixing or adding feature. The bootloader also uses the transaction module during the updates.

4 Future Work

The next version of the CubeSat will focuses on separate the OBC into two modules, one of them responsible of handling the communications securely between the ground station and the CubeSat and the other will focus on controlling the CubeSat and the current implementation will be sent to be reviewed and hopefully published as an educational satellite at the end of 2021. In the next phase as well, will manufacture group of CubeSat and manage the communication between all of them.

5 References

[1] “Why Facebook, SpaceX and dozens of others are battling over Internet access from space”, http://fortune.com/2019/01/25/facebook-spacex-internet-access-space, seen: 2021-02-14

[2] K. Woellert, P. Ehrenfreund, A. J. Ricco, and H. Hertzfeld, “Cubesats: Cost-effective science and technology platforms for emerging and developing nations”, vol. 47, 2011.

[3] “CubeSats in brief”, https://www.isispace.nl/cubesats, seen: 2021-02-15.

[4] F. Davoli, C. Kourogiorgas, M. Marchese, A. Panagopoulos and F. Patrone, “Small satellites and CubeSats: Survey of structures, ar-chitectures and protocols”, vol. 1, 2018.

[5] E. Kulu, “Nanosatellite and CubeSat database”, 2019.

[6] S. Bandyopadhyay, G. P. Subramanian, R. Foust, D. Morgan, S.-J, Chung and F. Hadaegh, “A review of impending small satellite formation flying missions”, 2015.

[7] H. Polat, J. Virgili-Llop, and M. Romano, “Survey, statistical analysis and classification of launched cubesat missions with emphasis on the attitude control method”, vol. 5, Jan. 2016.

[8] “SpaceX submits paper work for 30,000 more Starlink satellites”, https://spacenews.com/spacex-submits-paperwork-for-30000-more-starlink-satellites/, seen: 2021-02-14.

[9] “MarsCubeOne (MarCO)”, https://www.jpl.nasa.gov/cubesat/missions/marco.php, seen: 2019-02-14.

[10] Chantal Cappelletti, Simone Battistini, Benjamin Malphrus “CubeSat Handbook”, 1st Edition, 2020.

[11] Nasir Saeed, Ahmed Elzanaty, Heba Almorad, Hayssam Dahrouj, Tareq Y. Al-Naffouri,Senior, Mohamed-SlimAlouini, “CubeSat Communications: Recent Advances and Future Challenges”, 2019

[12] “List of the Egyptian satellites from NARSS”, https://space.skyrocket.de/directories/sat_sci-tech_egypt.htm, seen: 2021-02-14.

[13] Jan Hales, Martin Pedersen, Klaus Krogsgaard, “Attitude Control and Determination System for DTUsat - a CubeSat contribution”, 2002

[14] James R. Wertz, Wiley Larson, Space Mission Analysis and Design, 3rd Edition, 2008

[15] Rune Schlanbusch, Espen Oland, Per Johan Nicklasson, “Modeling and simulation of a cubesat using nonlinear control in an elliptic orbit”, June 2009

[16] Denier, W., “Satellite Payload Systems: Satellite Operations and 3D modeling”, 2010

[17] Ranganath R. Navalgund, V. Jayaraman and P. S., “Remote sensing applications”, 2007

[18] Koskienen, H., Eliasson, L., Holback, B., Anderson, L., Eriksson, A., Malkki, A., Norberg, O., Oulkkinen, T., Viljanen, A., Wahlund, J. E. & Wu, J. G., ” Space Weather and Interactions with Spacecraft”, 1999

[19] “M. D. A. S. T. SUNSAT Integration Meeting”, May 1994.

[20] H. Grobler, “Aspects affecting the design of a low earth orbit satellite on-board computer”, 2000.

[21] Hidayat, “Development of Flight Software for the Nanosatellite Onboard Computer Based on FM430 and Real-Time Operating System”, 2010.

[22] M. Polaschegg, “Study of a CubeSat Mission”, 2005.

[23] Victor U. J. Nwankwo, N. N. Jibiri, Michael Kio, “The Impact of Space Radiation Environment on Satellites Operation in Near-Earth Space”, 2020

[24] John Catsoulis, “Designing Embedded Hardware”, 2005

[25] Pieter Johannes Botma, “The Design and Development of an ADCS OBC for a CubeSat”, December 2011

[26] California Polytechnic State University, “CubeSat Design Specification”, 2009

[27] H. Koskinen, L. Eliasson, B. Holback, L. Andersson, A. Eriksson, A. Mälkki, O. Norberg, T. Pulkkinen, A. Viljanen, J.-E. Wahlund, J.-G. Wu, ”Weather and Interactions with Spacecraft”, 1999

[28] Egil Eide, Jørgen Ilstad, “NCUBE-1, the first Norwegian CUBESAT student satellite”, August 2003

[29] Luke Stras, G. James Wells, Daniel D. Kekez, Tiger Jeans, Daniel Foisy, Robert E. Zee, Freddy Pranajaya, “The Design and Operation of The Canadian Advanced Nanospace experiment (CanX-1)”, 2003

[30] ARM,” https://www.arm.com/products/silicon-ip-cpu/cortex-m/cortex-m3”, seen: 2021-04-01

[31] Energy Micro, “http://www.energymicro.com”, seen: 2021-4-01.

[32] Wikipedia Website, “http://en.wikipedia.org/wiki/Real-time_operating_system”, seen: 2021-04-01

[33] Isma Irum, Mudassar Raza, Muhammad Sharif, ” File Systems for Various Operating Systems: A Review”, August 2012

[34] Zerui Chen, “Design and implementation of FATFS data exchange with multiple storage media based on single-chip microcomputer”, 2020

[35] MQTT, “https://mqtt.org/faq/”, seen: 2020-03-26

6 Appendix

6.1 List of launched CubeSats

Name	Organization	Type	Mission status	Launch date
ACRUX-1	Melbourne Space Program	1U	Success	29-Jun-19
AAU CubeSat	Aalborg University	1U	Failed	30-Jun-03
STARS-Me	Shizuoka University, Japan	2U	CW receiving	22-Sep-18
CanX-1	UTIAS	1U	Failed	30-Jun-03
Cubesat XI-IV (Oscar 57)	University of Tokyo	1U	Active	30-Jun-03
CUTE-I (Oscar 55)	Tokyo Institute of Technology	1U	Active	30-Jun-03
DTUsat	Technical University of Denmark	1U	Failed	30-Jun-03
QuakeSat	Stanford University	3U	Active	30-Jun-03
TUSat1	Taylor University	1U		30-Jun-03
STARS	Kagawa University	2U	Active	23-Jan-09
Cubesat XI-V (Oscar 58)	University of Tokyo	1U	Active	27-Oct-05
nCube-2	ARR/NSC	1U	Failed	27-Oct-05
UWE-1	University of Würzburg	1U	Failed	27-Oct-05
SACRED	University of Arizona	1U	Destroyed	26-Jul-06
ION	University of Illinois	2U	Destroyed	26-Jul-06
Rincon 1	University of Arizona	1U	Destroyed	26-Jul-06
ICE Cube 1	Cornell University	1U	Destroyed	26-Jul-06
KUTESat	University of Kansas	1U	Destroyed	26-Jul-06
NCUBE-1	ARR/NSC	1U	Destroyed	26-Jul-06
HAUSAT-1	Hankuk Aviation University	1U	Destroyed	26-Jul-06
SEEDS-1	Nihon University	1U	Destroyed	26-Jul-06
CP-2	California Polytechnic University	1U	Destroyed	26-Jul-06
AeroCube 1	Aerospace Corporation	1U	Destroyed	26-Jul-06

MEROPE	Montana State University	1U	Destroyed	26-Jul-06
Mea Huaka`i (Voyager)	University of Hawaii	1U	Destroyed	26-Jul-06
ICE Cube 2	Cornell University	1U	Destroyed	26-Jul-06
CP-1	California Polytechnic University	1U	Destroyed	26-Jul-06
GeneSat-1	NASA/Santa Clara University	3U	Completed	16-Dec-06
CSTB1	Boeing	1U	Active	17-Apr-07
AeroCube 2	Aerospace Corporation	1U	Failed	17-Apr-07
CP-4	California Polytechnic University	1U	Active	17-Apr-07
Libertad-1	Sergio Arboleda University	1U	Successful	17-Apr-07
CAPE-1	University of Louisiana at Lafayette	1U	Active	17-Apr-07
CP-3	California Polytechnic University	1U	Active	17-Apr-07
MAST	Tethers Unlimited	1U		17-Apr-07
Cute-1.7 + APD II	Tokyo Institute of Technology	2U	Active	28-Apr-08
COMPASS-1	FH Aachen	1U	Active	28-Apr-08
AAUSAT-II	Aalborg University, Denmark	1U	Active	28-Apr-08
Delfi-C3	Delft University of Technology, The Netherlands	3U	Active	28-Apr-08
CanX-2	University of Toronto, Canada	3U	Active	28-Apr-08
SEEDS-2	Nihon University, Japan	1U	Active	28-Apr-08
PREsat	NASA	3U	Destroyed	3-Aug-08
NanoSail-D	NASA	3U	Destroyed	3-Aug-08
PharmaSat	NASA Ames Research Center,	3U	Completed	19-May-09

	Santa Clara University, UTMB			
CP6	California Polytechnic University	1U		19-May-09
HawkSat I	HISS	1U		19-May-09
AeroCube 3	Aerospace Corporation	1U		19-May-09
SwissCube-1	Ecole Polytechnique Fédérale de Lausanne	1U	Active	23-Sep-09
BeeSat-1	Berlin Institute of Technology	1U	Active	23-Sep-09
UWE-2	Universität Würzburg	1U	Active	23-Sep-09
ITUpSAT1	Istanbul Technical University	1U	Active	23-Sep-09
Hayato	Kagoshima University	1U	Failed[citation needed]	20-May-10
Waseda-SAT2	Waseda University	1U	Unclear[citation needed]	20-May-10
Negai	Soka University	1U	Successful[citation needed]	20-May-10
TIsat-1	University of Applied Scieces of Southern Switzerland (SUPSI)	1U	Active[citation needed]	12-Jul-10
StudSat	StudSat	1U	Inactive[citation needed]	12-Jul-10
RAX-1	University of Michigan	3U	Premature End	20-Nov-10
O/OREOS	NASA SMD	3U		20-Nov-10
NanoSail-D2	NASA Ames Research Center	3U	Completed	20-Nov-10
Perseus 000	Los Alamos National Laboratory	1.5U	Completed	8-Dec-10
Perseus 001	Los Alamos National Laboratory	1.5U	Completed	8-Dec-10
Perseus 002	Los Alamos National Laboratory	1.5U	Completed	8-Dec-10
Perseus 003	Los Alamos National Laboratory	1.5U	Completed	8-Dec-10

QbX1	NRL	3U	Successful	8-Dec-10
QbX2	NRL	3U	Successful	8-Dec-10
SMDC-ONE	US Army SMDC	3U	Completed	8-Dec-10
Mayflower-Caerus	Northrop Grumman (Mayflower); University of Southern California (Caerus)	3U	Completed	8-Dec-10
KySat-1	Kentucky Space	1U	Destroyed	4-Mar-11
Hermes	University of Colorado at Boulder	1U	Destroyed	4-Mar-11
Explorer-1 [Prime]	Montana State University	1U	Destroyed	4-Mar-11
Jugnu	IIT Kanpur	3U	Active	12-Oct-11
M-Cubed	University of Michigan	1U		28-Oct-11
DICE-1	Space Dynamics Laboratory	1.5U	Active	28-Oct-11
DICE-2	Space Dynamics Laboratory	1.5U	Active	28-Oct-11
Explorer-1 [Prime] Unit 2	Montana Space Grant Consortium	1U	Active	28-Oct-11
RAX-2	University of Michigan	3U	Active	28-Oct-11
AubieSat-1	Auburn University	1U	Active	28-Oct-11
ROBUSTA	Université Montpellier 2	1U	Failed	13-Feb-12
e-st@r	Politecnico di Torino	1U	Tumbling	13-Feb-12
MaSat-1	BME	1U	Active	13-Feb-12
Xatcobeo	University of Vigo	1U	Active	13-Feb-12
Goliat	University of Bucharest Romania	1U		13-Feb-12
PW-Sat	Warsaw University of Technology Poland	1U		13-Feb-12
UniCubeSat-GG	GAUSS team-Sapienza University of Rome, Italy	1U		13-Feb-12
F-1[98]	FPT University	1U	Failed; No signal received	21-Jul-12
TechEdSat-1	San Jose State University	1U		21-Jul-12

Raiko	Tohoku University / Wakayama University	2U	Successful	21-Jul-12
We-Wish	Meisei Electric	1U	Failed, no signal received	21-Jul-12
CSSWE	University of Colorado Boulder / LASP	3U	Active	13-Sep-12
VELOX-P1	Satellite Research Center, Nanyang Technological University	1U		1-Oct-12
FITSAT-1 (NIWAKA)	Fukuoka Institute of Technology	1U	Active	4-Oct-12
AAUSAT3	Aalborg University, Denmark	1U	Active	25-Feb-13
STRaND-1	Surrey Space Centre (SSC), University of Surrey, & Surrey Satellite Technology Ltd (SSTL)	3U	Operational	25-Feb-13
BeeSat-2	Berlin Institute of Technology	1U	Active	19-Apr-13
BeeSat-3	Berlin Institute of Technology	1U	Active	19-Apr-13
SOMP	Dresden University of Technology, Germany	1U		19-Apr-13
Dove-2	Planet Labs	3U	Active	19-Apr-13
OSSI-1		1U	Complete	19-Apr-13
PhoneSat 1.0 (Graham)	NASA Ames Research Center	1U		21-Apr-13
PhoneSat 1.0 (Bell)	NASA Ames Research Center	1U		21-Apr-13
PhoneSat 2.0.beta (Alexander)	NASA Ames Research Center	1U		21-Apr-13
NEE-01 Pegasus	Ecuadorian Space Agency	1U	Active	26-Apr-13
TurkSat-3USat	Istanbul Technical University	3U	Active	26-Apr-13

ESTCube-1[114]	University of Tartu	1U	Inactive since 17 February 2015. Partial failure.	7-May-13
ArduSat1	Nanosatisfi LLC	1U		3-Aug-13
ArduSatX	Nanosatisfi LLC	1U		3-Aug-13
Firefly	Taylor University	3U	Operational	20-Nov-13
ChargerSat-1	University of Alabama in Huntsville	1U	Launched. No contact established. Maximum time before passive return and destruction: 24 months	20-Nov-13
Vermont Lunar	Vermont Technical College	1U	Active	20-Nov-13
iCUBE-1	Institute of Space Technology Islamabad Pakistan	1U	Active	21-Nov-13
FUNcube-1	AMSAT-UK/ National Radio Centre	1U	Active	21-Nov-13
Delfi-n3Xt	Delft University of Technology, The Netherlands	3U	Active	21-Nov-13
ZACUBE-1 (TshepisoSat)	French South African Institute of Technology at the Cape Peninsula University of Technology	1U	Active	21-Nov-13
NEE-02 Krysaor	Ecuadorian Space Agency	1U		21-Nov-13
PUCPSAT-1	Universidad Católica del Perú	1U		21-Nov-13
VELOX-P2	Satellite Research Center, Nanyang Technological University	1U		21-Nov-13
ArduSat2	Nanosatisfi LLC	2U		9-Jan-14
CHASQUI - I	UNI	1U	Unknown	9-Jan-14
SkyCube	Southern Stars LLC	1U	Partial failure, satellite reentered	9-Jan-14
UAPSat-1	University Alas Peruanas	1U		9-Jan-14

OPUSAT	Osaka Prefecture University	1U		27-Feb-14
ITF-1 (Yui)	Tsukuba University	1U		27-Feb-14
INVADER (Artsat-1)	Tama Art University	1U		27-Feb-14
KSAT2 (Hayato-2)	Kagoshima University	1U		27-Feb-14
KickSat		3U	Failed	18-Apr-14
TSat4	Taylor University			18-Apr-14
AeroCube-6	Aerospace Corporation	1U		19-Jun-14
UniSat 6	GAUSS	In-orbit CubeSat launcher		19-Jun-14
Perseus-M	Canopus Systems US (Operated by Aquila Space, Inc.)	6U	Active	19-Jun-14
LEMUR-1	Nanosatisfi	3U		19-Jun-14
Antelsat	FING-IIE (Facultad de Ingeniería de la Universidad de la República, Instituto de Ingeniería Eléctrica), Antel (Administración Nacional de Telecomunicaciones)	2U	Active	19-Jun-14
Flock-1c x 11	Planet Labs, US	3U	Active	19-Jun-14
NanoSatC-Br 1	UFSM, INPE, Brazil	1U	Active	19-Jun-14
POPSAT-HIP 1	Microspace Rapid	3U	Active	19-Jun-14
ATHENOXAT-1	Microspace Rapid	3U	Active	16-Dec-15
QB50P1, QB50P2	Von Karman Institute, Belgium	2U	Active	19-Jun-14
VELOX-1-NSAT	Satellite Research Center, Nanyang Technological University	1U		30-Jun-14
VELOX-1-PSAT	Satellite Research Center, Nanyang Technological University	1U		30-Jun-14
UKube-1	UK Space Agency	3U	Complete, but still operational, awaiting	8-Jul-14

			further possible use by Amsat.	
AESP-14	ITA	1U	Failed	10-Jan-15
ExoCube	Cal Poly & PolySat	3U		31-Jan-15
FIREBIRD II	Montana State University & University of New Hampshire& Los Alamos National Laboratory& Aerospace Corp	1.5U ×2		31-Jan-15
GRIFEX	University of Michigan NASA JPL	3U		31-Jan-15
OptiCube 3	Cal Poly, SLO	3U	Active	20-May-15
AeroCube 8B	Aerospace Corp.	1.5U	Active	20-May-15
AeroCube 8A	Aerospace Corp.	1.5U	Active	20-May-15
OptiCube 2	Cal Poly, SLO	3U	Active	20-May-15
GEARRS-2	NearSpace Launch Inc	3U	Active	20-May-15
OptiCube 1	Cal Poly, SLO	3U	Active	20-May-15
BRICSat-P	U.S. Naval Academy	1.5U	Active	20-May-15
ParkinsonSAT	U.S. Naval Academy	1.5U	Active	20-May-15
USS Langley	U.S. Naval Academy	3U	Active	20-May-15
LightSail-1	The Planetary Society	3U	Completed (Reentered)	20-May-15
MinXSS	University of Colorado Boulder & Laboratory for Atmospheric and Space Physics	3U	Completed	6-Dec-15
Swayam	College of Engineering, Pune	1U	Active	22-Jun-16
Aalto-2	Aalto University, Finland	2U	Inactive	18-Apr-17

BRAC Onnesha	BRAC University	1U	Active	3-Jun-17
GhanaSat-1	All Nations University	1U	Active	3-Jun-17
Mazaalai	National University of Mongolia	1U	Active	3-Jun-17
Nigeria EduSat-1	Federal University of Technology Akure	1U	Active	3-Jun-17
TOKI	Kyushu Institute of Technology	1U	Active	3-Jun-17
Kalam SAT	NASA	Femto Satellite	Active	22-Jun-17
Aalto-1	Aalto University and Finnish Meteorological Institute, Finland	3U	Active	23-Jun-17
InflateSail	von Karman Institute for Fluid Dynamics	3U	Complete	23-Jun-17
LituanicaSAT-2	Vilnius University	3U		23-Jun-17
ASTERIA	NASA (JPL)	6U	Completed	14-Aug-17
Asgardia-1	Space Kingdom of Asgardia	2U	Active	12-Nov-17
EcAMSat	Santa Clara University	6U	Active	12-Nov-17
PicSat	Paris Observatory & CNRS	3U	Active	12-Jan-18
UBAKUSAT	Istanbul Technical University	3U	Active	2-Apr-18
1KUNS-PF	University of Nairobi	1U	Active	2-Apr-18
Irazú	Costa Rica Institute of Technology	1U	Active	2-Apr-18
DebrisSat 1 & 2	Surrey Satellite Technology	2U each	Active	2-Apr-18
Mars Cube One (MarCO)	NASA	6U each	Completed Mars flyby and successful relay	5-May-18
BHUTAN-1	Kyushu Institute of Technology	1U	Active	29-Jun-18
Maya-1	Kyushu Institute of Technology	1U	Active	29-Jun-18
UiTMSAT-1	Universiti Teknologi MARA	1U	Active	29-Jun-18

DAVE (CP-7)	PolySat & California Polytechnic State University	1U	Active	15-Sep-18
UNITE	University of Southern Indiana	3U	Awaiting deployment	5-Dec-18
RSat-P	United States Naval Academy	3U	Active	16-Dec-18
Lume-1	Alén Space & University of Vigo	2U	Active	27-Dec-18
Delphini-1	Aarhus University	1U	Active	5-Dec-18
ZA-AeroSat	Stellenbosch University	2U	active	
nSight-1	SCS-Space	2U	Successful	18-Apr-17
Światowid	SatRevolution S.A.	2U	Successful	17-Apr-19
Kraksat	SatRevolution S.A.	1U	Part successful	17-Apr-19
Bobcat-1	Ohio University		Launched	5-Nov-20
NEUTRON-1				5-Nov-20
SPectral Ocean Color (SPOC)	University of Georgia	3U	Launched	5-Nov-20
CACTUS-1	Capitol Technology University	3U	Launched	17-Jan-21
CAPE-3	University of Louisiana at Lafayette	3U	Launched	17-Jan-21
EXOCUBE-2	California Polytechnic University	3U	Launched	17-Jan-21
MiTEE	University of Michigan	3U	Launched	17-Jan-21
PICS-1 & PICS-2	Brigham Young University	1U	Launched	17-Jan-21
PolarCube	University of Colorado at Boulder	3U	Launched	17-Jan-21
Q-PACE	University of Central Florida	3U	Launched	17-Jan-21
RadFXSat-2	Vanderbilt University	3U	Launched	17-Jan-21
TechEdSat-7	NASA Ames Research Center	1U	Launched	17-Jan-21

6.2 ARM Cortex-M3

6.2.1 About the processor

6.2.1.1 Processor core Features

Low latency interrupt processor with Nested Vectored Interrupt Controller (NVIC) that features:

- Hardware divide instructions, SDIV and UDIV.
- Thumb and Debug states.
- Handler and Thread modes.
- Banked Stack Pointer (SP).
- Automatic processor saving and restoration state.
- ARM architecture v6 style BE8/LE support.
- Bits of priority of 3 to 8 configurable size.
- ARMv6 unaligned accesses.
- External interrupts of 1 to 240 configurable size.
- Dynamic reprioritization of interrupts.
- Priority grouping.

6.2.1.2 Memory Protection Unit (MPU)

An optional MPU used for memory protection with:

- Sub Region Disable (SRD)

- Eight memory regions.

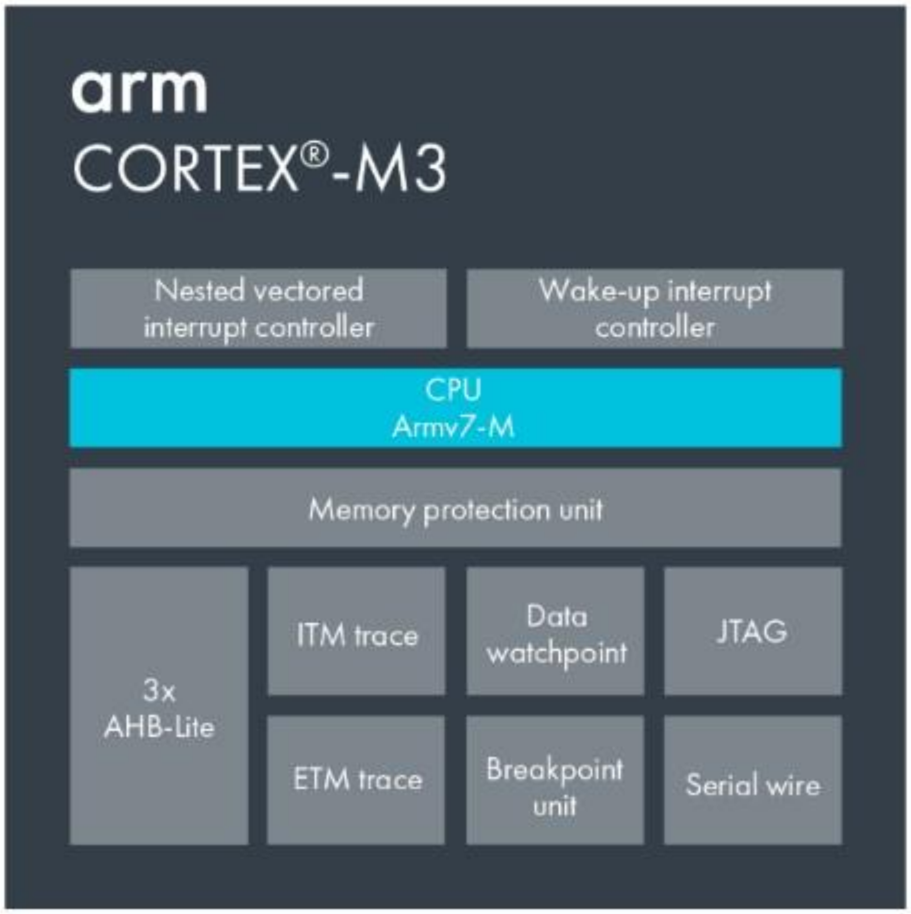
6.2.1.3 Bus interfaces

- Advanced High-performance Bus-Lite (AHB-Lite) ICode, DCode and System bus interfaces.
- Bit band support that includes atomic bit band write and read operations.
- Write buffer for buffering of write data.
- Advanced Peripheral Bus (APB) and Private Peripheral Bus (PPB) Interface.
- Memory access alignment.

6.2.1.4 Low-cost debug solution that features

- Debug all memory and registers access as well as the Cortex-M3 register bank when the core is running, halted, or held in reset.
- Serial Wire JTAG Debug Port (SWJ-DP) or Serial Wire Debug Port (SW-DP) debug access, or both.
- Flash Patch and Breakpoint (FPB) unit to be able to implement code patches and breakpoints.
- Data Watchpoint and Trace (DWT) unit to be able to implement data tracing, watchpoints, and system profiling
- Optional Embedded Trace Macrocell (ETM) for instruction trace.

- Instrumentation Trace Macrocell (ITM) to be able to printf style debugging.
- Trace Port Interface Unit (TPIU) for bridging to a Trace Port Analyzer (TPA).

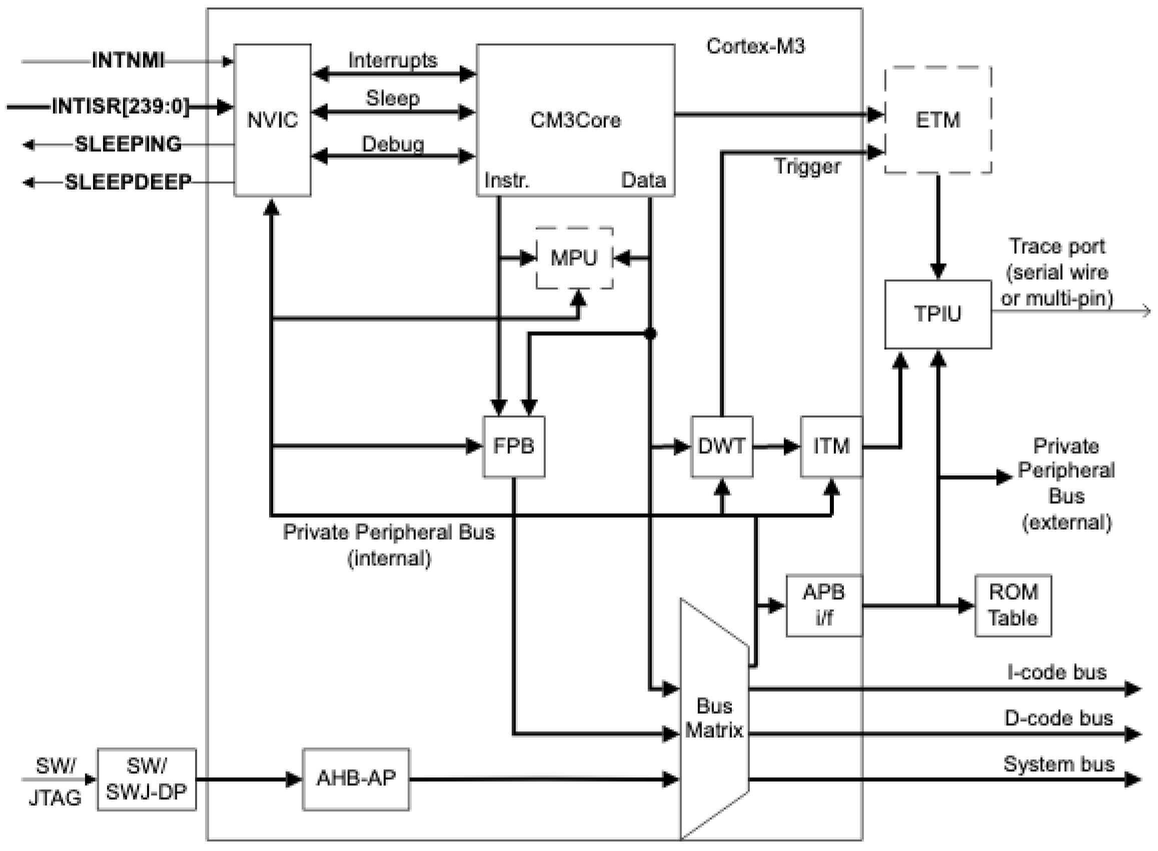


6.2.2 Components, hierarchy, and implementation

The main blocks of this processor are:

- Processor core
- Bus Matrix
- NVIC

- ETM
- DWT
- ITM
- MPU
- FPB
- TPIU.



6.2.3 Configurable options

- Interrupts

- Memory Protection Unit (MPU).
- Data Watchpoint and Trace (DWT)
- Embedded Trace Macrocell (ETM)
- AHB Trace Macrocell interface