Phase A

Command & Data Management System

Prepared by:

Tapparel Pierre-André

Checked by:

Approved by:

•
HEVs
Sion
Switzerland
•
16/06/2006
•

HEVs

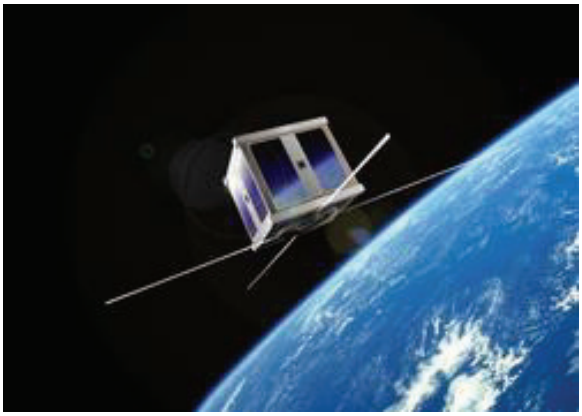
RECORD OF REVISIONS

ISS/REV	Date	Modifications	Created/modified by
1/0	14.06.2006	First edition	Pierre-André Tapparel

INTRODUCTION

This paper is about the construction of a student satellite, named Swisscube. This satellite is categorised as a picosatellite, which refers to the small size of the satellite. Commercial satellites are although varying in size, usually quite big and weighing in the range of 1000 kg's. Our satellite is restricted to 1 kg and a size of 10x10x10 cm - and is therefore referred to as a cubesat. Because of its small size and low weight, many cubesats can be placed in the same launch as secondary payloads. Because of our secondary status, we have no influence on the orbit our satellite will have, we have to do with whichever orbit the companies of the primary payloads select.

Because of the relatively humble conditions mentioned above, the price of getting the satellite in space is very low compared to the prices of commercial satellite launches.



Picture of a cubesat

As building a satellite is a very big project, the various tasks of designing the different parts have been assigned to different groups.

Our group consists of three people working in the University of Applied Science (HEVs) based in Sion, Switzerland. The first one is a second year student who is interested in the project, the second, a third year student working on the project during his summer semester project, and finally a supervisor, who is professor at the HEVs. Our tasks consist in designing the control and data-management system, and provide hardware expertise to other groups.

At it is imperative that the different parts function together, communication between the groups is of outmost importance. Budgets have to be made, interfaces have to be defined and the knowledge contained in the groups has to be shared in the best possible way in order to hope for a successful mission. The different groups meet once a week to discuss various issues, progresses and setbacks.

1 REFERENCES

1.1 Normative references

[N1] Preliminary Functional Specification: E3-AB-SEIC-Functional_Spec

1.2 Informative references

[R1] MAXIM MAX3051 CAN transceiver datasheet

[R2] MAXIM MAX7375 Integrated Silicon Oscillator datasheet

[R3] Microchip 24FC1025 Serial EEPROM datasheet

[R4] Maxwell 79LV0408 Parallel EEPROM datasheet

[R5] ATMEL AT91SAM7A1 32-bit microcontroller datasheet

[R6] Microchip PIC18LF4680 8-bit microcontroller datasheet

[R7] Microchip PIC18LF8680 8-bit microcontroller datasheet

[R8] "The CAN bus: An introduction" course, Dominique Gabioud, HEVs

2 TERMS, DEFINITIONS AND ABBREVIATED TERMS

2.1 Abbreviated terms

EDAC	:	Error Detection And Correction
TID	:	Total Ionizing Dose
PL	:	Payload
CDMS	:	Control and Data-Management System
COM	:	COMmunication Subsystem
EPS	:	Electrical Power Subsystem
ADCS	:	Attitude Determination and Control System
ROM	:	Read Only Memory
RAM	:	Random Access Memory
RTC	:	Real Time Clock
CMOS	:	Complementary Metal-Oxide Semiconductor
OSI	:	Open System Interconnection
CAN	:	Controller Area Network
I^2C	:	Inter-Integrated Circuit
UART	:	Universal Asynchronous Receiver-Transmitter
ISI	:	Inter-Symbol Interference
SNR	:	Signal to Noise Ratio
COTS	:	Commercial Off-The-Shelf
NRZ	:	Non-Return to Zero
SCL	:	Serial Clock Line
SDA	:	Serial Data Line
MSB	:	Most Significant Bit
SMBus	:	System Management Bus
TTL	:	Transistor-Transistor Logic
SOF	:	Start Of Frame
EOF	:	End Of Frame
CRC	:	Cyclic Redundancy Code
EMI	:	ElectroMagnetic Interference
DOD	:	Department Of Defence

3 DESIGN DRIVERS

3.1 Radiation effects on electronics

Radiation in space is generated by particles emitted from a variety of sources both within and beyond our solar system. The nature of the radiation differs in place and time. In some places our satellite might cross the so-called 'solar wind', which is a burst of particles (mostly protons and electrons) emitted from the sun, and here radiation will be substantial. In other places radiation will be very light. The single particles also vary in the amount of energy they possess. High-energy particles are more dangerous to our circuitry than low-energy particles (often referred to as background radiation), as they penetrate deeper into the components and cause greater damage in case of collision. Much research is carried out in making maps of intensities and different kinds of radiation in space. Such a map is show in the figure 1. The figure shows that the radiation is not only depending on the sun but also very much on the magnetic field of earth (magnetosphere).The orbit of our project is so low that one will not touch the Van-Allen belts (turquoise zone).As further details of this area are quite complex and out of the scope of this project, we will not discuss it further.

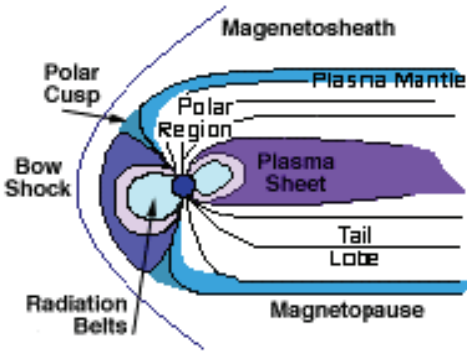


Figure 1: Radiation near the earth

3.1.1 Single event effects

A single-event effect results from, as the term suggests, a single, energetic particle. If such a particle hits one of our components, we might encounter three different errors:

- Bit-flips (soft error)

This phenomenon causes the value of a specific bit to change. This will lead to software malfunction. In order to avoid this malfunction we plan to implement error detection and correction circuitry (EDAC) between the processor and the memory chips. This will be able to correct most errors. In case of substantial damage to the software, it might be necessary to reboot or even upload new software from earth. The ROM has to be radiation-hard so that it does not suffer from these bit-flips.

- Latch-up (hard error)

The Figure 2 shows a part of a CMOS component being hit by a high-energy particle. This particle can either short-circuit the source or drain to ground, which is located in the bottom of the figure. The short circuit arises because the particle creates a conductive 'tunnel'. A short circuit will draw a lot of current and the component is very likely to take permanent damage. The issue is to include a latch-up protection circuitry near the sensible component to turn off the power immediately.

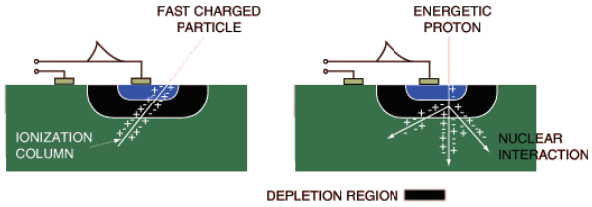


Figure 2: Particle colliding with a CMOS Component

- Burn-out (hard error)

In case of the power not being turned off at a latch-up, a burn-out can occur. A burn-out can sometimes be seen by the human eye, as a burn-out means that the chips is simply melted in the area of the latch-up. After this, the chip is useless.

3.1.2 Total ionizing dose (TID)

The total ionizing dose, mostly due to electrons and protons, can result in device failure. TID is measured in terms of the absorbed dose. The TID is calculated from the trapped protons and electrons, secondary Bremsstrahlung photons, and solar flare protons. As TID increase, material degradation increase. Long-term exposure can cause device threshold shifts, increased device leakage and power consumption, timing changes, decreased functionality, etc.

In order to have an idea of how our components will react to these long-term effects, a sample of component should be tested. The amount of radiations is measured in kRad. 5 to 10 kRad is the average amount of radiation we can expect in one year on the orbit used by our cubesat. The duration of the spatial mission is set as three month. The components should so be able to work wit a TID of 3 kRad. This factor should not play a major role for our short mission; nevertheless the component must be tested.

The only thing that can be done to limit these long-term effects is to place shielding material around the components. This could for example be aluminium, which is both light and well shielding. This kind of shielding will not stop any of the high-energy radiation that causes latch-ups and bit-flips.

3.2 General hardware block diagram

The fact that the computer has to operate in space also implies that the system has to be designed in a special way. In this section we will describe how these demands affect the design of microcontroller based architecture. We will describe which blocks the computer must be build from, but not which chips we choose to use, as this will be covered in a later section. The system described in this section is depicted in the following figure. This block diagram provides an overview of a processor based architecture and not the final architecture that will be used.

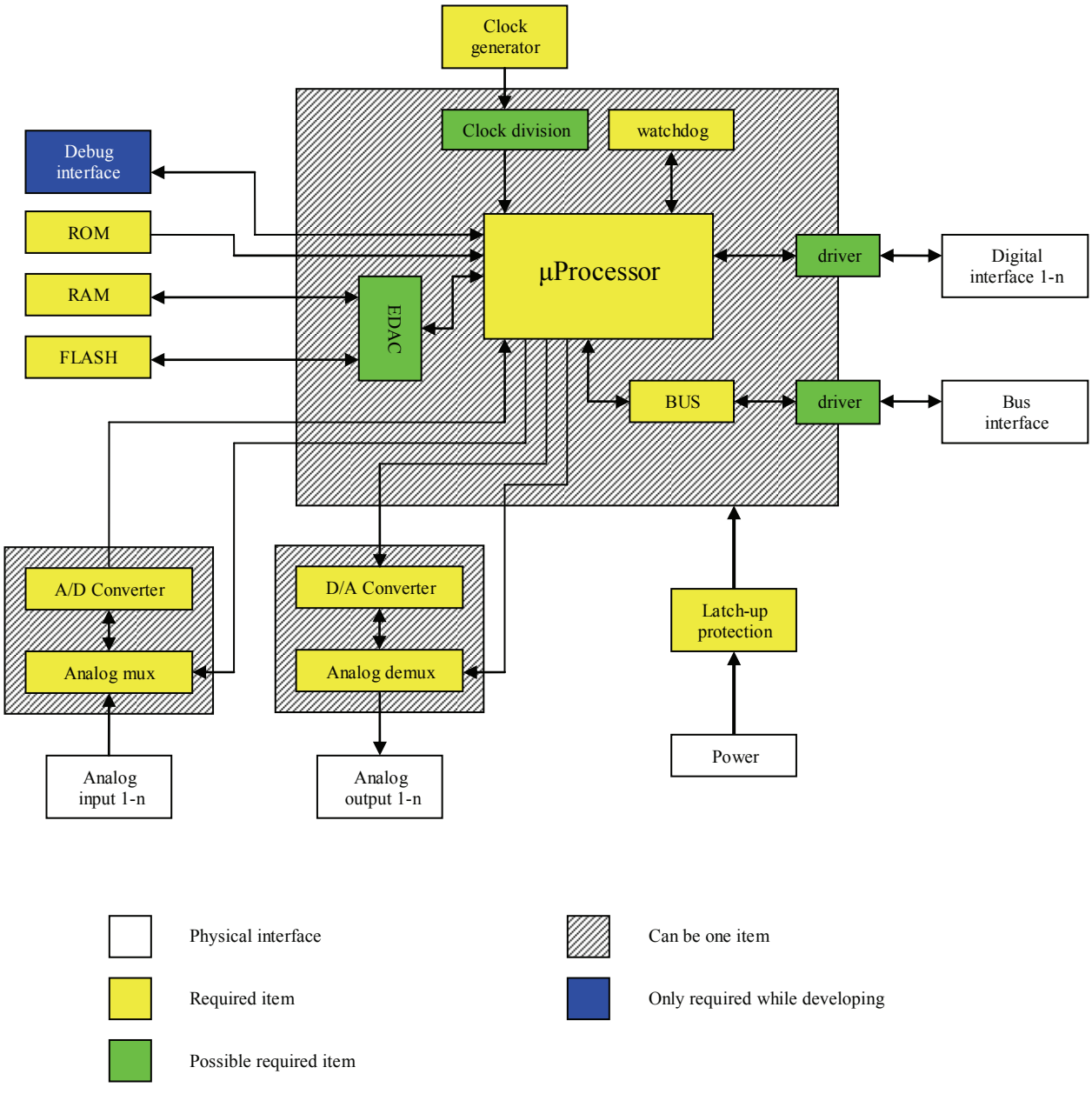


Figure 3: General block diagram

The one thing probably that limits the design of the architecture the most, is the space-operating demand. As no one will be able to reset the processor manually in case of a software malfunction, the system must be able to do this by itself. The way is done is to use a watchdog timer. The timer is increased on a regular basis (typically it is clocked by the master clock) and when it reaches a certain value, it resets the processor. Of course this is not what we want when the computer is working correctly. Therefore a part of the software must reset the watchdog before it reaches the reset value. This id done when the software is working correctly, but if it gets in a deadlock the watchdog is not reset, and therefore it resets the system. The watchdog can either be build-in in the processor or it can be a separate unit as shown in the figure 3.

When studying the figure you will find three different types of memory: ROM, flash, and RAM. These three types are also chosen because of the space-operating nature of this computer. In space, the radiation may lead to bit-flips in memory after which the software may do unpredictable things. The watchdog will obviously reset the processor if this happens, and the processor starts loading the boot-software. It is essential that this software always work correctly, and therefore it must be stored in a memory, where bit-flips do not occur. Some types of ROM (read only memory) have this feature.

The flash will be used to store the operating system and other software. Some of this software could be store in ROM, but as it might be necessary to change parts of it, when the Swisscube is in space, it will be stores in flash. On the issue of flash memory ESA experts tend to give the advice from do not use to no problem at all. However XI-V (a cubesat developed by the university of Tokyo) team has used it on their satellite (doing prior radiation testing) with success.

The RAM (random access memory) will be used as a temporary memory when the programs run, and as a place where measured values can be stored. As both flash and RAM suffer from bit-flips it might be desired to have some error detection and correction circuitry (EDAC) between these memories and the processor.

In order to communicate with other elements, the computer must provide different interface, a bus interface to exchange data with other subsystem, the bus selection will be covered in a later section, both analogue and digital interfaces. Since some processors may not be able to supply very much current it might be necessary to insert drivers between the digital interface and the processor.

Depending on which processor is chosen we may have to add a real time clock (RTC) to the board, but if the processor has enough timers, this can also be implemented in software. The purpose of the real time clock is to make it possible to schedule tasks.

As we described above, the CMOS component are sensible to latch-up. To avoid burn-out, a latch-up protection circuitry must be implemented between power bus and sensible component. When a latch-up occurs the power must be turn off immediately.

One last thing of special interest is shown on the figure: The debug interface. The purpose of this is to make it possible to find and correct errors in the hard- and software design. It also makes it possible to upload new software to the flash. It consists of measuring points for important signals and some sort of interface to the processor and the flash.

4 BUS TOPOLOGY OVERVIEW

As we decided to build a distributed architecture, distributed architecture means that all system have their own microcontroller and do their own job, all subsystem must be able to communicate with each-other. In order to transmit data or other information, a bus must be implemented between the subsystems. The following topology has been decided.

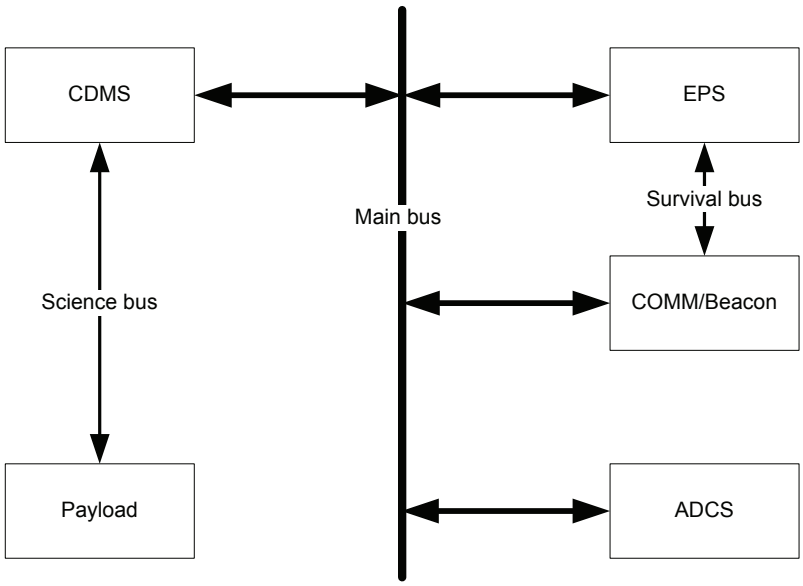


Figure 5: Bus topology overview

4.1.1 Main bus

The main bus will provide the communication between CDMS, EPS, COMM and ADCS.

As command and important measure will be transferred on the main bus, the assumption was made that the main bus will be reliable and provides an important data rate. Moreover as four subsystems will be connected on the main bus, the protocol will provide multipoint topology.

4.1.2 Science bus

The science bus will only connect the payload with the CDMS, and transfer image captured with the payload to the CDMS in the way to save data between to communications with ground station.

If some science data will be corrupted or loosed, the satellite will still be able to work properly, in that way the reliability is not as important as for the main bus. The size of images transferred via the science bus is 5[kbit], and one image will be transferred every 30[s], a low data rate bus should suits our needs. . More of that, as only two subsystems will be connected, the topology should be point-to-point.

4.1.3 Survival bus

In the case of the main bus will not work properly, EPS should ever be able to transmit some information directly to the COMM. Those informations are only used to have an idea of what happen wrong during the mission. The data rate on the bus is limited by the data rate of the transceiver of the COMM subsystem which is about 1.2[kbps].

The survival and science bus have appreciatively the same constraints, and in order to simplify the architecture, the same bus type will be used either for the survival and science bus.

4.2 Type of bus

In computer architecture, a bus is a subsystem that transfers data or power between computer components inside a computer or between computers.

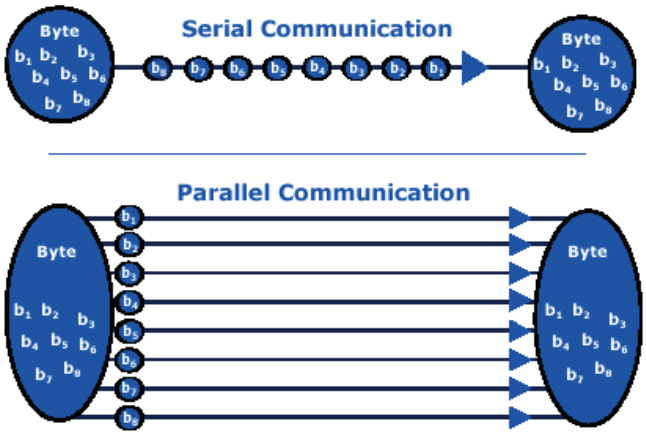


Figure 6: Serial vs. parallel communication

In a digital communications system, there are two methods for data transfer: parallel and serial. Parallel connections have multiple wires running parallel to each other (hence the name), and can transmit data on all the wires simultaneously. Serial, on the other hand, uses a single wire to transmit the data bits one at a time.

It is a natural question to ask which one of the two transmission methods is better. At first glance, it would seem that parallel ports should be able to send data much faster then serial ports. Let's say we have a parallel connection with 8 data wires, and a serial connection with a single data wire. Simple arithmetic seems to show that the parallel system can transmit 8 times as fast as the serial system.

However, parallel ports suffer extremely from inter-symbol interference (ISI) and noise, and therefore the data can be corrupted over long distances. Also, because the wires in a parallel system have small amounts of capacitance and mutual inductance, the bandwidth of parallel wires is much lowers than the bandwidth of serial wires. An increased bandwidth leads to a better bit rate. We also know that less noise in the channel means we can successfully transmit data reliably with a lower signal to noise ratio (SNR). In addition, because of the increased potential for noise and interference, parallel wires need to be far shorter than serial wires. Moreover, as the bus will connect the different subsystems, it will be preferable to use less wire than possible.

In conclusion, for our use, it is preferable to use a serial link between the subsystems.

4.3 Bus candidates

As shown before, the serial link will provides the best choice for our needs. All the following bus use serial link. The descriptions following are only summaries which present the main features of the bus. The OSI layer model is presented on appendix 8.1

- UART (universal Asynchronous Receiver Transmitter)

The UART is a piece of computer hardware that translates between parallel bits of data and serial bits.

The four following bus are based on the UART component, but used with different drivers.

  - driver RS232

    The RS232 specification defines Mechanical, Electrical, and Functional characteristics. RS232 is an Unbalanced (Single Ended), unidirectional (point-to-point) interface. Signal is referenced to ground. RS232 drivers feature a controlled slew rate. Normal output levels are $\pm$5 [V]. RS232 use asynchronous framing with a known data width of 8bits, and NRZ (non-return to zero) encoding.

  - driver RS422

    RS422 define a balanced (differential) interface; specifying a single, unidirectional driver with multiple receivers (up to 32). RS422 will support Point-to-Point, Multi-Drop circuits, but not Multi-point.

  - driver RS485

    RS485 define a balanced (differential) interface; defines the Physical layer (OSI Layer 1), signalling protocol is not defined. RS485 specifies bidirectional, half-duplex data transmission. Up to 32 transmitters and 32 receivers may be interconnected in any combination, including one driver and multiple receivers (multi-drop), or one receiver and multiple drivers. RS485 is the Multi-Point version of RS422.

  - driver LVDS EIA-644

    LVDS as Low Voltage Differential Signalling defines the electrical layer only. The EIA-644 provides a point to point topology with differential interface.

- I^2C

The I^2C bus uses a bi-directional Serial Clock Line (SCL) and Serial Data Lines (SDA) and due to its two-wire nature can only communicate half-duplex. The interface uses 8 bit long bytes, most significant bit (MSB) first, with each device having a unique address. Any device may be a transmitter or receiver, and a master or slave. Data and clock are sent from the master.

- SMBus

The SMBus (system management bus) is a two wire interface which is based on the I^2C bus. The two SMBus lines are called SMBCLK and SMBDAT and operate at a frequency of 100[KHz]. Both SMBCLK and SMBDAT are bidirectional, and pulled high via a resistor. The SMBus link may have multiple masters and slaves on the bus, but only one master may be active at any one time. Slaves may receive or transmit data to the master.

- SPI (Serial Peripheral Interface)

The SPI Bus is a four wires serial communications interface used by many microprocessor peripheral chips. The SPI circuit is a synchronous, full duplex serial data link setup as a Master/Slave interface. The SPI bus specifies two control lines. Chip Select (CS) and Serial Clock (SCLK) and two data lines, Serial Data In (SDI) and Serial Data Out (SDO). One SPI device acts as the SPPI Master by controlling the data flow and asserting device select then receives or transmit data via the two data lines.

- CAN bus

The Controller Area Network (CAN) specification defines the Data Link Layer (OSI Layer 2) and the Physical Layer (OSI Layer 1) too.

The CAN bus is a balanced (differential) two-wire interface. This bus use NRZ encoding to ensure compact messages with a minimum number of transition and high resilience to external disturbance.

- Spacewire (IEEE 1355)

The Spacewire bus provides a bidirectional serial interconnect which builds a scalable parallel system using a pair of unidirectional lines. IEEE 1355 defines the Physical and Data Link Layer. The electrical interface is specified as standard Transistor-Transistor Logic (TTL), using either 3.3 or 5[V].

- MIL – STD – 1553

MIL-STD-1553 is a department of defence (DOD) Military (MIL) Standard (STD), which defines both the mechanical, electrical, and functional characteristics. MIL 1553 uses a balanced (differential) interface. The interface is dual redundant with between 2 and 32 interfaces devices on the bus. The multiplex data bus system shall function asynchronously in a command/response mode, and transmission shall occur in a half-duplex manner.

4.4 Analysis criteria

As shown further some criteria has different reference and weighting depending on which use of the bus will be made. The other criteria has the same weight and reference for the two different trade-offs.

- Reliability

The reliability provides a comparison between different buses; this comparison is made upon the immunity to electromagnetic interferences, the possibilities offered to check if errors occur and prioritization it several nodes will transmit at the same time.

As said further main bus and science/survival bus have not the same constraints, in that way the reliability factor will be adapted. It has a weighting of 5 in the case of the main bus and 2 for the other one.

- Power consumption

In accordance to the embedded system specification, we only have a few watts available, in that way the power consumption is a really important criterion.

Power consumption has the weight of 5.

- Availability on commercial market

As the name says, this criterion compares the provided component supporting the bus, in other words it indicates if the bus is commonly used or not.

The availability has the weight of 4.

- Data rate

The assumption was made that the main bus will provide a high data rate, for this case the reference is set as 1[Mbps] and the weight factor as 3. For the other bus, the reference is 100[kbps] and the weighting 2.

- Topology

This criterion indicates if the bus supports multi-point or multi master topology. The main bus requires a multi-point or multi-master topology in that case this factor has the weight of 4.

As only two devices will be connected on the science or survival bus, the requirements are different and a point to point topology will be sufficient. The weight factor will be adapted and has weight of 3 in that case.

- Implementation facilities

As implementation facilities, we say if the bus needs specific component like transceiver, buffer, or a specific protocol.

The availability has the weight of 3.

We have also compared the bus on a few others subjects like voltage, development tools and if this bus has already be used in space.

4.5 Bus selection

The two following tables present the bus comparison. The first one is for the main bus and the second one for the survival and science bus.

4.5.1 Main bus weight factor

	Items	Reliability	Power consumption	Availability on commercial market	Data Rate	Vcc 3.3V or 5V	Topology	Already used in space	Implementation facilities	Dvpt tool in school	Results
	Reference	CRC/diff	2Tx & 2Rx Active: 25 mW Stdby: 5 mW	Multiple sources	1Mbit	dig.std	Multipoint / Multi MASTER	Industry or cubesat	Without ext. Component / protocol	Facilities	
	weighting factor	5	5	4	3	1	4	4	3	3	
Bus	RS232 (UART)	-	++	++	--	++	--	+	++	++	113
Bus	UART (driver RS422)	o	-	o	++	++	-	+	+	o	102
Bus	UART (driver RS485)	o	-	o	++	++	-	o	+	o	98
Bus	UART (driver LVDS EIA-644)	o	--	o	++	++	-	+	+	o	97
Bus	I2C	-	++	++	-	++	++	o	o	++	122
Bus	SMBus	o	++	o	--	++	++	--	o	+	105
Bus	SPI	-	++	+	o	++	++	--	o	+	110
Bus	CAN bus	++	-	+	+	++	++	++	o	++	132
Bus	Spacewire (IEEE1355)	++	--	--	++	++	--	++	--	-	87
Bus	MIL-STD-1553	++	--	--	+	--	+	++	--	--	89

In accordance with the assumptions made, the CAN bus will provides the best choice for the main bus.

4.5.2 Science and Survival bus weight factor

Items		Reliability	Power consumption	Availability on commercial market	Data Rate	Vcc 3.3V or 5V	Topology	Already used in space	Implementation facilities	Dvpt tool in school	Results
Reference		CRC/diff	2Tx & 2Rx Active: 25 mW Stdby: 5 mW	Multiple sources	100kbit	dig.std	Multipoint / Multi MASTER	Industry or cubesat	Without ext. Component / protocol	Facilities	
weighting factor		2	5	4	2	1	3	4	3	3	
Bus	RS232 (UART)	-	++	++	+	++	o	+	++	++	117
	UART (driver RS422)	o	-	o	++	++	+	+	+	o	92
	UART (driver RS485)	o	-	o	++	++	+	o	+	o	88
	UART (driver LVDS EIA-644)	o	--	o	++	++	+	+	+	o	87
	I2C	-	++	++	+	++	++	o	o	++	113
	SMBus	o	++	o	+	++	++	--	o	+	96
	SPI	-	++	+	+	++	++	--	o	+	98
	CAN bus	++	-	+	++	++	++	++	o	++	110
	Spacewire (IEEE1355)	++	--	--	++	++	o	++	--	-	72
	MIL-STD-1553	++	--	--	++	--	++	++	--	--	71

At that time, either RS232 or I^2C should be used for the science and survival bus. The final choice is not yet done, and will depend on the different microcontroller we will use, but as the EPS should already provides an I^2C interface for the smart battery monitor and as no 8-bit microcontroller will offers two I^2C interface, the final choice will probably be the RS232 bus.

4.6 CAN bus, how does it work?

The CAN (Controller Area Network) is a serial bus for a network of (micro-) controllers. It was originally developed by R.BOSH for use in automobiles at the end of the 80's. Today it is an ISO (International Standard Organization) standards, cheap, robust, medium speed, and real-time.

As seen before, the CAN bus is a differential two-wire interface which use NRZ encoding and bit-stuffing. Bit-stuffing is to create artificially transition upon long series of 0's or 1's. The transmitter adds a dummy dominant (recessive) after 5 recessive (dominant) symbols. The receiver does not consider the dominant (recessive) following 5 recessive (dominant) symbols.

NRZ Encoding is used in slow speed synchronous and asynchronous transmission interfaces. With NRZ, a logic '1' bit is sent as a high value and a logic '0' bit is sent as a low value. The receiver may lose synchronisation when using NRZ to encode a synchronous link which may have long runs of consecutive bits with the same value (no changes in voltage). Other problems with NRZ include; Data sequence containing the same number of 1's and 0's will produce a DC level, and NRZ requires a large bandwidth, from 0[Hz] (for a sequence containing only 1's or only 0's) to half of the data rate (for a sequence of 10101010).

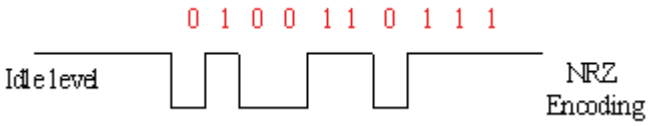


Figure 7: NRZ Encoding

While CAN uses differential wire, it has a great immunity against electromagnetic interference (EMI), because EMI generates the same changes on the potential of both wires. The receiver evaluates the differential voltage U_{diff} which remains stable.

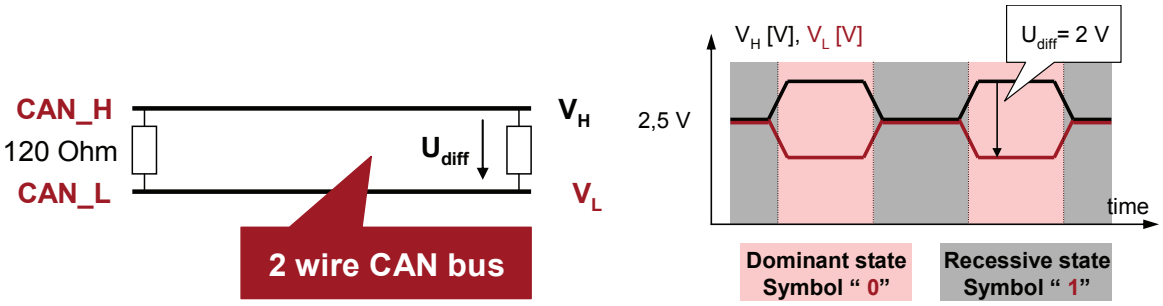


Figure 8: CAN bus electrical interface circuit

A number of different data rates are defined, with 1[Mbps] (Bits per second) being the top end, and 10[Kbps] the minimum rate. All modules must support 20[Kbps]. Cable length depends on the data rate used. Normally all the device in a system transfer uniform and fixed bit-rates. The maximum line length is 1[Km], 40 meters at 1[Mbps]. Termination resistors, avoiding reflection, are used at each end of the cable.

The CAN Bus interface uses an asynchronous transmission scheme controlled by start and stop bits at the beginning and end of each character. This interface is used, employing serial binary interchange. Information is passed from transmitter to receivers in a data frame.

As from a CAN point of view, all nodes are equivalent; several nodes can transmit at the same time. The message with the lowest Message Identifier gets higher priority. Transmission of the messages with the higher Message Identifier is stopped at some time during the transmission of the Message Identifier. The bus arbitration scheme is called CSMA/CA (Carrier Sense Multiple Access based on Collision Avoidance).

4.6.1 CAN bus message frame

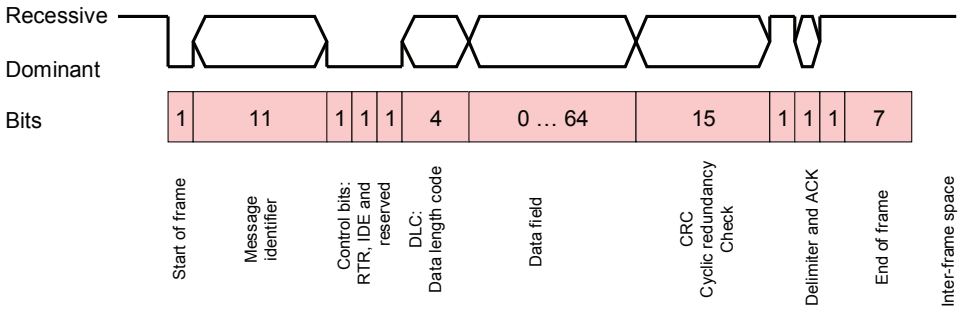


Figure 9: CAN frame

- Start of frame

The start of frame (SOF) is a single symbol at dominant state.

- Message identifier

The message identifier is an 11 bits parameter freely managed by the application designers. A message identifier is assigned to an input/output or to a group of inputs/outputs

- Control bits

There are two controls bits, the remote transmission request (RTR) which indicate the frame type and the identifier extension (IDE) which enable 29-bits message identifier instead of 11-bits.

- Data length code

Data length code (DLC) is a 4 bits parameter which specifies the length of the Data field. 4 bits allows 16 different values but only 9 permitted.

DLC	0	1	2	3	4	5	6	7	8
Length in bits	0	8	16	24	32	40	48	56	64

- Data field

The content of Data field can be freely managed by the developers but all actively receiving nodes must be capable to interpret its content.

- Cyclic redundancy check

The cyclic redundancy check (CRC) enables the receiver to detect transmission errors. The coefficients are generated modulo-2.

- First delimiter

One symbol time in order to let the receiver(s) check the CRC.

- Acknowledge

Acknowledge (ACK) is in dominant state if no error is detected otherwise in recessive state.

- End of frame

End of frame (EOF) is a 7 symbol field in recessive state.

- Inter-frame space

The time between two different frames is at least 2 symbol times which are in recessive state.

4.6.2 Power consumption

In order to work properly, an interface between the CAN protocol controller and the physical wires of the bus lines must be implemented. The CAN transceiver provides differential transmit capability to the bus and differential receive capability to the CAN controller.

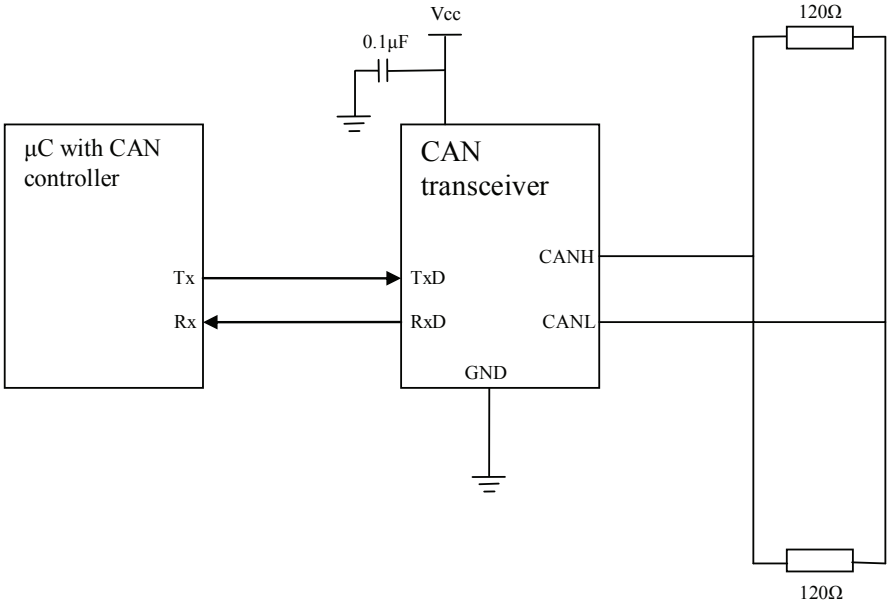


Figure 10: Typical operating circuit

The MAX3051 CAN transceiver from MAXIM provides the features required for our use. It works with a typical supply voltage of 3.3[V] (value of our power bus) and a low supply current. It typically use 35[mA] (max: 70) to send a dominant bit and 2[mA] (max: 5) to send a recessive bit. Moreover, it provides a standby mode which only uses 8[µA] (max: 15). The transceiver use current only while speaking, and only one device can speak at a time. While there is nothing to send or while listening, the transceiver is in standby mode. The power consumption of the transceiver is slightly depending on data rate.

(This can transceiver is just given as example and will not provide the final architecture description)

As the final duty cycle on the bus can not be yet defined and in order to have an idea on the power consumption of the bus, we make the assumption that, each second, the CDMS sends one frame to

all subsystems connected on the bus, and the subsystems send one frame to the CDMS. Moreover, the size of frame is set to a mid-range value of 40 'dominants' bits. As at that time four subsystems are connected, and to let the highest priority Message Identifier for the ground station, the highest value of the message identifier will be 5; this value will be used to calculate the approximated power consumption. With those assumptions, the size of one CAN frame is 98 bits long and contain 63 dominant bits and 35 recessive bit. Including the minimal inter-frame space of 2 recessive bits, the final value is 100 bits long with, 63 dominant and 37 recessive. The calculation is done with typical power consumption of the MAX3051 CAN transceiver.

Two different data rate value will be used, one at 100[kbps] and the other at 1[Mbps]

- Legend

D	: *data rate*
T_{bit}	: *time to transmit one bit*
I_R / I_D	: *current used to transmit one recessive / dom inant bit*
P_R / P_D	: *power used to transmit one recessive / dom inant bit*
E_R / E_D	: *enery used to transmit one recessive / dom inant bit*
E_{frame}	: *energy used to transmit one frame*
E_{comm}	: *energy used to make one communication cycle*
$E_{onOneHour}$	: *energy used on one hour*

- Data rate = 100[kbps]

$$D = 100[kbps] \Rightarrow T_{bit} = \frac{1}{D} = 10[\mu s]$$
$$I_R = 2[mA] \Rightarrow P_R = 2[mA] \cdot 3{,}3[V] = 6{,}6[mW] \Rightarrow E_R = 6{,}6[mW] \cdot 10[\mu s] = 66[\eta J]$$
$$I_D = 35[mA] \Rightarrow P_D = 35[mA] \cdot 3{,}3[V] = 115.5[mW] \Rightarrow E_D = 115.5[mW] \cdot 10[\mu s] = 1{,}155[\mu J]$$
$$E_{frame} = 37 \cdot 66[\eta J] + 63 \cdot 1{,}155[\mu J] = 75{,}2[\mu J]$$
$$E_{comm} = 10 \cdot 75{,}2[\mu J] = 752[\mu J]$$
$$E_{onOneHour} = 752[\mu J] \cdot 3600 = 2.7[Wh]$$

- Data rate = 1 [Mbps]

$$D = 1[Mbps] \Rightarrow T_{bit} = \frac{1}{D} = 1[\mu s]$$
$$I_R = 2[mA] \Rightarrow P_R = 2[mA] \cdot 3{,}3[V] = 6{,}6[mW] \Rightarrow E_R = 6{,}6[mW] \cdot 1[\mu s] = 6{,}6[\eta J]$$
$$I_D = 35[mA] \Rightarrow P_D = 35[mA] \cdot 3{,}3[V] = 115{,}5[mW] \Rightarrow E_D = 115{,}5[mW] \cdot 1[\mu s] = 115{,}5[nJ]$$
$$E_{frame} = 37 \cdot 6{,}6[\eta J] + 63 \cdot 115{,}5[nJ] = 7{,}5[\mu J]$$
$$E_{comm} = 10 \cdot 7{,}5[\mu J] = 75[\mu J]$$
$$E_{onOneHour} = 75[\mu J] \cdot 3600 = 270[mWh]$$

As we can see with those calculations, the highest data rate is the lowest power consumption.

According to use the highest data rate of 1[Mbps], the only way to reduce power consumption is to reduce the number of frame transmitted, or to find another transceiver which use less power.

I remember that this power consumption is not the final value, but only an approximated value.

5 SUB-SYSTEM REQUIREMENT

In order to have a complete and fully functional satellite, it is imperative that all groups comply with a set of specifications, so that the resource in terms of space and power are used in the best manner without exceeding the budgets. The different groups, who individually have made preliminary budgets for their own parts, have formed these specifications and budgets. These budgets have been summed up in system-engineering perspective, mainly by the power and the mechanics group, in order to have an idea of whether or not they are realistic. The final measures are still to be determined, but below the present guidelines are listed.

All the following subsystem must provides a number of A/D converters, digital I/O and sufficient memory to store software.

5.1 Electrical power subsystem (EPS)

The 8 bit microcontrollers take measures of solar array current; calculate the slope between to measures and chose the best duty cycle to command the step-up converter in order to have the maximum power. The step-up converter frequency will be about 100[kHz] .This microcontroller must execute the software of beacon subsystem too.

The microcontroller should provide the CAN interface in order to communicate with other subsystems and an I^2C to communicate with the smart battery monitor. Moreover, a third bus interface must be present to communicate directly with the COM subsystem.

5.2 Communications subsystem (COM)

In order to send and receive data from the ground station, the satellite must have a communication subsystem. The COM subsystem will be based on an 8 bit microcontroller too. This microcontroller should provide a CAN interface to be connected on the main bus, in the way to communicate with other subsystem, and another interface to be directly connected to the EPS, in case of malfunction of the main bus and to have some information, as temperature, current, voltage and state of the subsystems.

5.3 Control and data-management system (CDMS)

The CDMS must provide a 750[kbit] long memory in order to store the 150 compressed pictures took by the payload until the data can be transferred to the ground station. This subsystem should provide the ability to made housekeeping, scheduling and interprets different command send from the ground station. The communication, as shown further, is made with the CAN bus, but another interface with the payload will be used.

5.4 Attitude determination and control system (ADCS)

The ADCS needs processing abilities in order to calculate the attitude of the satellite and the algorithm which will command the motor. We make the assumption that a microcontroller which offer between 10 to 20 MIPS should provide sufficient power to do that job. A microcontroller with a 32 bits core will suits our needs.

5.5 Payload (PL)

The payload will take picture of nightglow effects; the size of one picture is about 40*40 pixels with 10 bit long pixel. This subsystem should provide a memory of 16[kbit] in order to temporary save one picture. Moreover the microcontroller must compress one image each 30[s] to 5[kbit] length. This compressed image will be sent to the CDMS via the science bus.

As the time to compress an image is quite long, we made the assumption that an 8 bit microcontroller should provide enough processing abilities.

5.6 Summary

With those requirements, we see that two different microcontroller must be used, one 32-bit and one 8-bit. The 8-bit microcontroller will be used by different subsystem which requirements should differ. In order to simplify the architecture and spatial test, only one type of 8-bit microcontroller for the four different subsystems.

6 CHOICE OF COMPONENTS

Satellites are normally extremely expensive and account for years of development. To improve the odds for success, it is custom to choose components of great reliability for satellites. To be absolutely sure of reliability, it is not uncommon to buy components that have actually been tested in space. This form of certainty is very expensive and lead times are a.

A cheaper alternative is to choose a Commercial off-the-shelf (COTS) component that has been used in space before. Another component just like it will probably work in space too. The only problem with this alternative is, that development of satellites is a long process, and when you find a component that has been successfully used in space, it is most likely more than two or three years old. In these days, where new and better technologies revolutionize the chip-marked every other day, it is very tempting to choose new and more powerful components for your design. We have agreed to yield to this temptation. Furthermore we will perform tests to verify that the components do not collapse with the first signs of space radiation.

6.1 The processor

We need a processor for the different subsystem that can execute the onboard programs in a reasonable way. The compulsory tasks of the satellite (power control, communication, attitude control, etc.) could be managed by a quite humble processor or microcontroller. Processors and microcontroller from a few years back would do the job we no problems and we could choose a very reliable one.

Nevertheless, we examined both new and old processors because of the above argumentation on why to take a chance on newer products.

We found that the newer processors have made drastic improvements on power consumption even though the speed of the processors has greatly increased.

6.1.1 Analysis criteria

In order to choose the perfect (or at least the best) processor, we have evaluated the different processors in different categories and summed up the information in the table below. The different categories have different weights, as they are not equally important. In some categories, the rating has been made from a reference value, which we have set from average specification of the component. The categories are:

- Peak power consumption (1 MIPS)

As we only have a few Watts available to all subsystems in the satellite, it is of outmost importance that the processor has low power consumption. The 1 MIPS value is not the final frequency of the processor, but the value which the power consumption of processors will be compared to.

Power consumption has the weight of 5.

- Standby power consumption

During the scheduling of the mission, some subsystems will not be used. In order to save power those subsystems will be in standby mode. The reference value has been set at 3[mW].

Standby consumption has the weight of 3.

- Temperature Range

The temperature will differ significantly depending on the location of the satellite. In the sunny side of earth, the surface facing the sun will be very warm and in the shadow of earth, everything will get very cold. The temperature range has been set to 0°C to 40°C for normal operation. But in special case the temperature might exceed this range in both directions, which is why we would like to have components with great temperature tolerances. Luckily, it is normal for military/industrial components to be operational from -40°C to 80°C, which we believe is sufficient.

Temperature Range has the weight of 3.

- Already used in space

As the name says, by 'already used in space', we mean whether or not the processor has been tried in space before, and if this processor has work properly.

Space rating has a weight of 4.

- Power voltage (core & I/O)

The power bus will provide 3.3[V]. Instead of having numerous supply voltages, which take place and weight, it is preferable to run a single supply voltage for the core and I/O of the processor. This standard supply voltage can also be used for other parts of the computer.

Those two categories have the weight of 4.

- Package

The place available on the satellite is very limited too. We look for having components where the pins are located along the edges, which offers greater mechanical characteristic than ball grid array components where the pins are distributed underneath the chip. Due to the nature of project, it should be interesting to have ceramic packaging than usual plastic.

Package has the weight of 3.

- I/O compatibility

As the sensors are not yet defined we made the assumption that 5[V] tolerant I/O which is a standard voltage will be appreciable.

The I/O compatibility has the weight of 2.

- MIPS

We made the assumption that a microcontroller which provides between 10 to 20 MIPS will be sufficient to make the processing of the ADCS. The reference value for 8 bit microcontroller is set to 1 to 10 MIPS.

This criterion has the weight of 4.

- Multiplier

As some processing will require multiplication, it should be interesting to have an integrated multiplier, but not required.

The multiplier has the weight of 1.

- Digital and Analogical I/O number

All sensors and actuators need access to the processor via the I/O pins of the processor, it is preferable to have sufficient I/O pin in order to connect all the peripherals.

We made the assumption that 12 digital I/O and 8 A/D converters will be sufficient.

Those criteria have the weight of 3.

- UART

The UART component will be used to program the chip on the earth during the debug and test phase, and probably for the survival and science bus during spatial mission.

UART has the weight of 2.

- Development environment

In order to have a fully functional microcontroller software must be developed and implemented. This criterion compares the facilities of development environment provided by the manufacturer. It is really important to have powerful tools to manage the hardware in order to debug software.

Development environment has the weight of 4.

The 8-bit microcontroller

The entire following microcontroller provides a CAN and I^2C interface.

Items		Peak Power consumption (1 MIPS)	Standby Power consumption	Temperature range	Package	Power voltage	I/O compatibility	multiplier	Already used in space	MIPS	Digital I/O number	Analog I/O number	UART	development environment	Results
Reference		20mW	3mW	-40°C to + 85°C	ceramic/flat < 5cm2/	3.3V	5V tolerant	1	minimum one success	1 to 10	12	8	1	Facilities	
weighting factor		5	3	3	3	4	2	1	4	1	3	3	2	4	
Items	PIC18LF4680	+	++	o	++	o	-	+	-	o	++	+	o	++	149
	PIC18F4680	+	++	o	++	--	-	+	-	o	++	+	o	++	141
	PIC18LF8680	o	++	+	++	o	o	+	-	o	++	++	o	++	152
	ATMEL AT90CAN128	o	++	o	+	o	-	o	-	+	++	o	o	++	135

The PIC18LF4680 and PIC18LF8680 from Microchip manufacturer will offer the best features for our needs. The first one is based on the new nanoWatt technology which can significantly reduce power consumption during operation. However the software is not yet developed, so it is impossible to estimate who many power will this technologies save. The second one, which is based on an older core, has not the new nanoWatt technologies but provides an external memory interface, more AD converters and 80 pins. The PIC18LF4680 has only 40 pins. At that time the choice is not done, further analysis should provides the answer on which microcontroller will be used.

6.1.2 The 32-bit microcontroller

The entire microcontrollers listed contain a CAN controller and an I^2C interface. This is not an exhaustive list, but a list that compare the main core family which would be interesting in term of low power consumption and features.

Items		Peak Power consumption (1 MIPS)	Standby Power consumption	Temperature range	Package	Power voltage core	Power voltage I/O	I/O compatibility	Already used in space	MIPS	Multiplier	Digital I/O number	Analog I/O number	UART	development environment	Results
Reference		80mW	3mW	-40°C to + 85°C	ceramic/flat < 5cm2/	3.3V	3.3V	5V tolerant	minimum one success	10 to 20	1	12	8	1	Facilities	
weighting factor		5	3	3	3	4	4	2	4	3	1	3	3	2	4	
Items	NEC UPD703286 (V850ES/SJ2)	++	o	o	-	o	o	o	-	+	o	++	+	++	--	143
	FUJITSU MB91F267N (FR60 lite)	--	-	+	-	--	--	o	-	+	-	++	+	++	--	106
	STM STR910FM32X (ARM966)	+	o	o	-	--	o	o	-	++	-	++	o	++	+	141
	ATMEL AT91SAM7A1 (ARM7TDMI)	+	++	o	-	o	o	o	-	+	-	++	o	++	++	156

As shown on this table the best microcontroller is one from ATMEL based on an ARM7TDMI core. The ARM7TDMI core is a 32-bit embedded RISC processor delivered as a hard macrocell optimized to provide the best combination of performance, power and area characteristics. The ARM7TDMI core enables system designers to build embedded devices requiring small size, low power and high performance.

6.1.3 Microcontroller used on other Cubesat

This section will speak about the microcontrollers used on other cubesat project; this is not an exhaustive list, but only some references.

The dtuSat, cubesat developed by the University of Denmark is based of a 32-bits ATMEL AT91M40800 microcontroller, this microcontroller will not enter in the selection because it does not provide a CAN interface. This microcontroller is based on the ARM7TDMI core, the same core as the AT91SAM7A1 selected before.

The XI-V cubesat developed by the University of Tokyo used PIC16C622 and PIC16F877 with success. Those two 8-bits microcontrollers have no CAN interface, for this reason they will not enter the comparative.

Texas A&M used 32-bits Dragonball VZ integrated processor. This processor provides 5.4MIPS at 33MHz, which will not provide low power solution compared to the other available microcontroller. Moreover, the assumption made that the 32-bit microcontroller should be able to have a 10 to 20 MIPS performance.

The cubesat Kit is based on a MSP430 16-bits microcontroller from Texas Instrument. As, by assumption, we will only used 8 and 32-bits microcontroller and as they do not provide a CAN interface, this microcontroller will not be used for the Swisscube.

The University of Illinois used a Hitachi SH7045 SH2. This microcontroller offers interesting features like low power consumption for quite high performance. The reason that this microcontroller will not enter the comparative is that it does not provides a CAN interface.

The specifications and requirements differ a lot from one cubesat to another. As we can see none microcontroller will suits our needs. However, the dtuSat and XI-V used Microchip and ATMEL microcontroller with success. The microcontroller we selected comes from the same microcontroller technologies, only features change. That means that there are strong chances that our microcontroller successes in the radiation test. However it is impossible to say, at that time, that the selected microcontroller for the Swisscube will work properly during the mission in space.

6.2 Oscillator

The majority of clock sources for microcontroller can be grouped into two types: those based on mechanical resonant devices, such as crystals and ceramic resonators, and those based on electrical phase-shift circuits such as RC (resistor, capacitor) oscillators. Silicon oscillators are typically a fully integrated version of the RC oscillator with the added benefits of current sources, matched resistors and capacitors, and temperature compensation circuits for increased stability. Two examples of clock source are illustrated in the following figure. Figure (a) shows a Pierce oscillator configuration suitable for use with mechanical resonant device like crystals and ceramic resonators, while figure (b) shows a simple RC feedback oscillator.

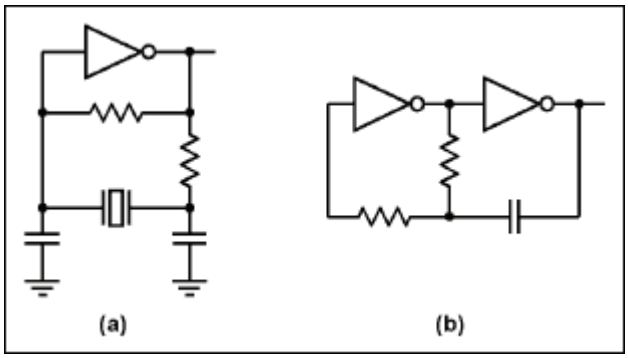


Figure 11: (a) pierce oscillator (b) RC oscillator

- Primary differences between mechanical resonator and RC oscillators

Crystals and ceramic resonator-based oscillators (mechanical) typically provide very high initial accuracy and a moderately low temperature coefficient. RC oscillators, in contrast, provide fast start-up and low cost, but generally suffer from poor accuracy over temperature and supply voltage, and show variation from 5% to 50% of nominal output frequency. While the circuits illustrated above can produce clean reliable clock signals, their performance will be heavily influenced by environmental conditions, circuit component choice, and the layout of the oscillator circuit. Ceramic resonators and their associated load capacitance values must be optimized for operation with particular logic families. Crystals are not as sensitive to amplifier selection but are susceptible to frequency shifts (and even damage) when overdriven. Environmental factors like electromagnetic interference (EMI), mechanical vibration and shock, humidity, and temperature affect oscillator operation. These environmental factors can cause output frequency changes, increased jitter, and in severe cases, can cause the oscillator to stop functioning.

- Oscillator modules

Many of the considerations described above can be avoided through use of oscillator modules. These modules contain all oscillator circuit components and provide a clock signal as a low-impedance square-wave output. Operation is guaranteed over a range of conditions. Crystal oscillator modules and fully integrated silicon oscillators are most common. Crystal oscillators are more precise than discrete component RC oscillator circuits, and many provide comparable accuracy to ceramic resonator-based oscillators.

- Power consumption

Power consumption is another important consideration of oscillator selection. The power consumption of discrete component crystal-oscillator circuits is primarily determined by the feedback-amplifier supply current and by the in-circuit capacitance values used. The power consumption of amplifiers fabricated in CMOS is largely proportional to the operating frequency and can be expressed as a power-dissipation capacitance value.

The different power consumptions below are given for example.

The power-dissipation capacitance value of an HC04 inverter gates used as an inverting amplifier is typically 90[pF]. For operating at 4[MHz] from a 5[V] supply, this equates to a supply current of 1.8[mA] The discrete component crystal oscillator circuit will typically include an additional load capacitance value of 20[pF], and the total supply current becomes 2.2[mA].

Ceramic resonator circuits typically specify larger load capacitance values than crystal circuits, and draw still more current than the crystal circuit using the same amplifier.

By comparison, crystal oscillator modules typically draw between 10[mA] and 60[mA] of supply current because o the temperature compensation and control functions included.

The supply current for silicon oscillators depends on type and function, and can range from a few [µA] for low-frequency (fixed) devices to ten of [mA] for programmable-frequency parts. A low-power silicon oscillator as the MAX7375 from Maxim draws less than 2[mA] when operating a 4[MHz].

6.2.1 Summary

Clock Source	Accuracy	Advantages	Disadvantages
Crystal	Medium to high	Low cost	Sensitive to EMI, vibration, and humidity. Complex circuit impedance matching.
Crystal Oscillator Module	Medium to high	Insensitive to EMI and humidity. No additional components or matching issues.	High cost; high power consumption; sensitive to vibration; large packaging.
Ceramic Resonator	Medium	Lower cost	Sensitive to EMI, vibration, and humidity.
Integrated Silicon Oscillator	Low to medium	Insensitive to EMI, vibration, and humidity. Fast startup, small size, and no additional components or matching issues.	Temperature sensitivity is generally worse than crystal and ceramic resonator types; high supply current with some types.
RC Oscillator	Very low	Lowest cost	Usually sensitive to EMI and humidity. Poor temperature and supply-voltage rejection performance.

As shown on the summary, an integrated silicon oscillator will provide the best choice for our need. This type of oscillator is insensitive to EMI and vibration, more of that it provides the ability to drive many microcontrollers. Typically, the MAX7375 is able to drive a load of about 100[pF] capacitance with an acceptable rise time, and a microcontroller typically has a 10[pF] fan-in capacitance. As all subsystems are connected on a motherboard it is possible, in order to save power, to distribute the signal clock to the different components with only one oscillator. The final design is not yet decided, but this solution should be an elegant one.

6.3 The CDMS memory

In order to save images captures by the payload, the CDMS board should provide a least 750[kbit] memory. 2[Mbit] would be very good, as it would allow the CDMS to store other housekeeping data and memory security margin. Different type of memory exists; they can be classed in the three following categories.

- *Volatile memory* requires constant power to maintain the stored information. The SRAM (static random access memory) is the most known example.
- *Dynamic memory* is volatile memory which also requires that stored information is periodically refreshed, or read and rewritten without modifications. One example is the DRAM (dynamic random access memory)
- *Non-volatile memory* will retain the stored information even if it is not constantly supplied with electrical power. FLASH and EEPROM (electrically erasable programmable read-only memory) are two different widely used non-volatile memory.

It could be interesting to keep data either if the subsystem encounters a dysfunction and must be shut-down or reboot, in that way non-volatile memory will be used. EEPROM has similar qualities than Flash but differs in the way that with EEPROM it is possible to rewrite a single byte at any address. It is a big advantage compared to a Flash where you have to erase an entire sector at a time. In that way the power consumption of a Flash memory chip will generally be higher than EEPROM. The sector size differs in function of the type of Flash selected. The Flash and EEPROM memory chip should be either access in serial or parallel. At first approximation it seems that Flash is less reliable than EEPROM for a spatial use. Moreover, typical endurance levels for parallel Flash are 10 000 cycles, serial Flash devices can range from 10 000 to 100 000 cycles and serial EEPROM can exceed 1 million cycles. Because of this we chose EEPROM memory.

As we say by assumption that the CDMS board will be based upon an 8-bit microcontroller. The microcontroller is not yet defines. However two microcontrollers are already selected. The PIC18LF4680 which does not provide a lot a pin and the PIC18LF8680 which has a specific external memory interface.

In the case we would choose the 40 pin microcontroller it is interesting to access the EEPROM memory chip with a serial link. Some different serial link should be used, but in the way that the microcontroller must have an I^2C interface, because EPS requirements, the memory chip will used an I^2C serial link. In the case we choose the second one, parallel memory chip will be used.

There is not much difference between parallel and serial memory chip. Power consumption over one typical write cycle will be substantially equal, the serial link will draws a few current but as the bit are coming serially it takes much longer to write the data than with parallel link which will consume much more current. The advantage of serial devices is they typically need only 8 pins to be connected. The space used by the memory chip is smaller than with parallel devices. In the other hand parallel devices requires about 16 address lines and 8 data lines (depending on the internal organization of memory) in addition to the usual chip select, write/read pins. The advantage of the parallel devices is that memory density is higher than serial one and data transfer shorter.

To compare the possible choice of serial EEPROM we looked at the following categories:

- Size

  As say before a size of 2[Mbit] will suit our needs.

- Supply voltage

  A supply voltage of 3.3[V] is preferred as this is the supply voltage of the processor.

- Read/write/stand-by current

  Read and write current are the current drawn during active time, stand-by current is the current drawn when the component is passive.

Serial EEPROM	Item	Size (kbit)	Supply voltage	Max write current	Max read current	Standby current
	ATMEL AT24C1024	1024	2.7 to 5.5	5mA	2mA	3uA
	Microchip 24FC1025	1024	2.5 to 5.5	5mA	450uA	100nA
	Renesas HN58W241000I	1024	2.5 to 3.6	4mA	1mA	1uA

As shown on this table the maximum available size for serial EEPROM is 1024[kbit], but it is possible to connect two different devices on the same bus. In that way it is possible to use one memory chip in order to save science data and one other to save housekeeping data. The Microchip memory chip has an extremely low standby current, and as the memory will often be in standby mode, it will only consume 330[nW] at most of the time.

A space radiation tested parallel memory chip is the Maxwell 79LV0408. This memory chip has a size of 4[Mbit] and has a low power dissipation of 88[mW/MHz] in active mode and 440[µW] in stand-by mode. The supply voltage used is 3.3[V].

As the final design is not yet done, further calculations depending on the microcontroller frequency, the time between two memory access and other criteria must provide reliable power consumption results. Depending on this results and some other analysis as risk studies, final board size and weight information, the best solution will be hold.

7 FUTURE DEVELOPMENT

At this point of the project there is still much work to do, the first thing is to design a final architecture for the CDMS, and provide a test board to the software group in order to validate or modify this design. Many different hardware tests like radiation test, risk studies will be done on this board in the way to certify that the final board will work properly during space mission. In a more remote future, the other subsystem board will be analyzed according to the same method.

8 CONCLUSION

At the end of the phase A of the project (the first of the six definite phase), each subsystem groups have spent a lot of time in order to provide the main baselines of the Swisscube. The mechanical, electrical and logic configuration are yet define, but those informations are only preliminary and will not provide the final architecture when the satellite will be launch. A lot of thing will be changed during the next phase of the project in accordance with informations and other elements that we do not know at that time.

APPENDIX

8.1 OSI (Open Systems Interconnection) Model

The OSI model divides the function of a protocol into a series of layers. Each layer has the property that it only uses the function of the layer below, and only exports functionality to the layer above. A system that implements protocol behaviour consisting of a series of these layers is known as a 'protocol stack'. Protocol stacks can be implemented either in hardware or software, or a mixture of both. Typically, only the lower layers are implemented in hardware, with the higher layers being implemented in software.

The OSI reference model is a hierarchical structure of seven layers that defines the requirements for communications between two systems.

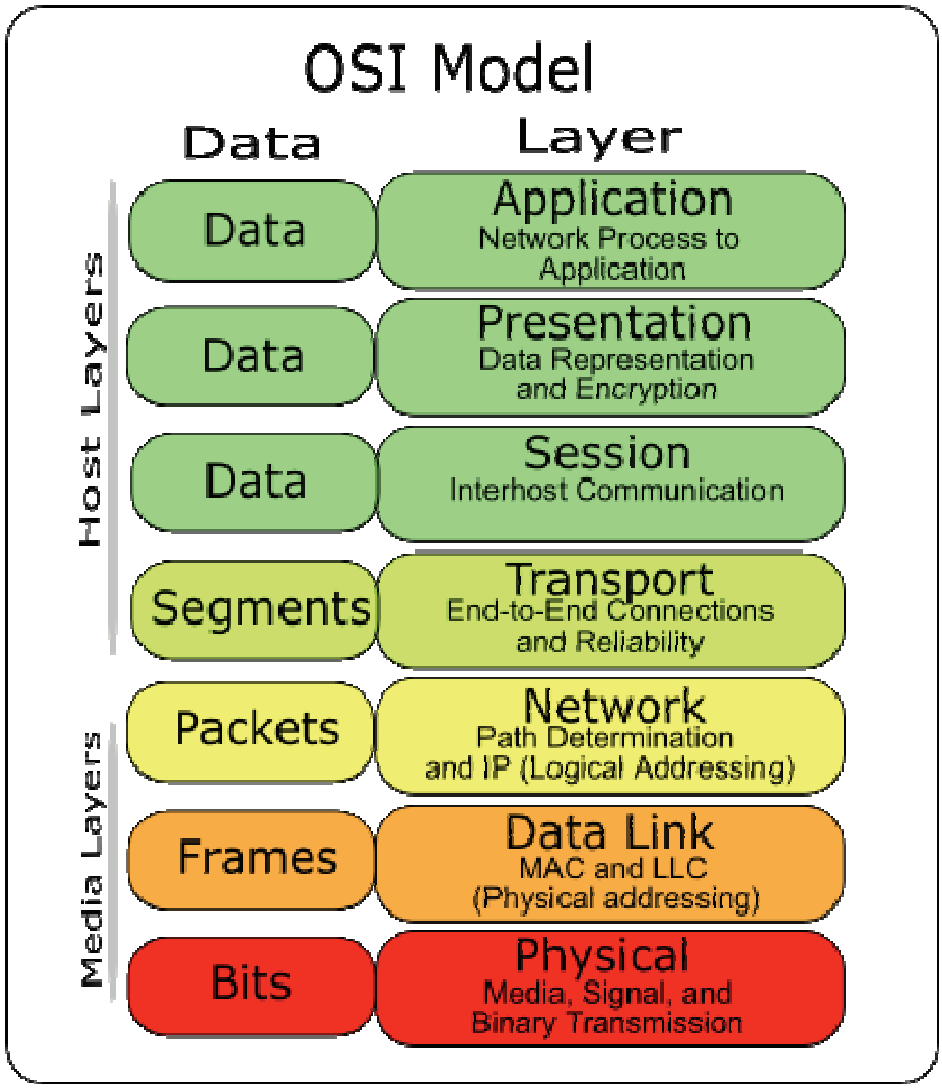


Figure 4: OSI layer model

8.1.1 Description of OSI layers

- Layer 7: Application Layer

The Application Layer is closest to the end user, provides a means for the user to access information on the network through an application. This layer is the main interface for the user to interact with the application and therefore the network. Some examples of application layer implementations include Telnet, File Transfer Protocol (FTP), and Simple Mail Transfer Protocol (SMTP).

- Layer 6: Presentation Layer

The Presentation Layer relieves the Application Layer of concern regarding syntactical differences in a message's data representation with the end-user systems. MIMI encoding, data compression, and similar manipulation of presentation is done at this layer to present the data as a service or protocol developers sees fit.

- Layer 5: Session Layer

The session layer provides the mechanism for managing the dialogue between end-user application processes. It provides for either duplex or half-duplex operation and establishes check pointing, adjournment termination, and restart procedures. The OSI model made this layer responsible for 'graceful close' of sessions, which is property of TCP (Transmission Control Protocol), and also for session check pointing and recovery, which is not usually used in the Internet protocol suite.

- Layer 4: Transport Layer

The Transport layer provides transparent transfer of data between en users, thus relieving the upper layers from any concern with providing reliable and cost- effective data transfer. The transport layer controls the reliability of a given link. Some protocols are state and connection orientated. This means that the transport layer can keep track of the packets and retransmit those that fail. The best known example of a layer 4 protocol is TCP. It is the layer that converts message into TCP (Transmission Control Protocol) or UDP (User Datagram Protocol) packets.

- Layer 3: Network Layer

The Network layer provides the functional and procedural means of transferring variable length data sequence from a source to a destination via one or more networks while maintaining the quality of service requested by the Transport layer. The Network layer performs routing, flow control, segmentation/documentation, and error control functions. The best known example of layer 3 protocol is the Internet Protocol (IP).

- Layer 2: Data Link Layer

The Data Link layer provides the functional and procedural means to transfer data between network entities and to detect and possibly correct errors that may occur in Physical Layer

- Layer 1: Physical Layer

The Physical layer defines all the electrical and physical specifications for devices. This includes the layout of pins, voltages, and cable specification. The major functions and services performed by the physical layer are.