NTNU
Norwegian University of
Science and Technology

Internal Data Bus of a Small Student Satellite

Marius Lind Volstad

Master of Science in Electronics
Submission date: June 2011
Supervisor: Bjørn B. Larsen, IET

Norwegian University of Science and Technology
Department of Electronics and Telecommunications

Problem description

In modern satellites, containing many separate digital systems, it is essential for the functionality of the satellite to allow for reliable and power efficient communication between all systems. The most effective way of ensuring system-to-system communication is through a common data bus. The internal data bus will be the backbone of the satellite and has to remain operable through the lifespan of the satellite. It is also beneficial for the integrity of the satellite to have an On-Board Controller (OBC) to log useful data and monitor the digital systems.

The student shall design a reliable data bus and OBC for use in the NTNU Test Satellite (NUTS) project.

Assignment Given: 24.01.2011
Supervisor: Bjørn B. Larsen
Co-supervisor: Roger Birkeland

Abstract

This thesis presents a platform for a small student satellite. A backplane solution with slots for 8 individual modules is proposed. The backplane provides the modules with power and a common data bus, as well as mechanisms for isolating modules from the rest of the system and a possibility of restoring misbehaving modules. It has a power consumption of about 100mW.

An on-board controller has been designed and tested. It consists of a 32-bit MCU, a NAND flash memory for housekeeping logs, an SRAM memory for processing variables, an OTP memory for safe storage of module firmwares, USB interface and a wireless intra-satellite communication module. The on-board controller has the main responsibility for monitoring and preserving the satellite's health. The power consumption will be dependent of the final firmware, but a rough estimate is about 100mW.

Also, an experimental wireless intra-satellite communication system is proposed. The system consists of a small wireless module, that will be integrated onto any module that is a part of the wireless network.

Preface

My first thought when joining the NUTS project, was that it would be extremely satisfying building something that is destined to be launched into space. But I felt a bit confused after the first introductory speeches of what building a satellite was all about. There were a lot of new concepts and methodologies to become familiar with. After some time immersing myself in space and satellite related material, I again began to feel the excitement of being a part of a team of satellite builders.

The work with the backplane has been a joint effort by Dewald de Bruyn and myself. Dewald has had main responsibility for the power solution and done most of the hardware layout drawings, while I have made the digital systems. We have had many discussions about the system, and feel that we have ended up with a solid design.

It has been incredibly educational to work with the NUTS project. I have met many challenges and felt a lot of joy in overcoming the obstacles. Also, I have had the pleasure of being a part of a very competent team. And now, I'm looking forward to coming back to NTNU for the launch of NTNU's first successful satellite mission!

_____

Marius Lind Volstad

Acknowledgement

I would like to thank my supervisors Bjørn B. Larsen and Roger Birkeland for guidance on this thesis, and thanks to Dewald de Bruyn for cooperation during the backplane development. Thanks to Torbjørn Finnøy, Daniel Aasbø, Kjetil Kvalø and Carolina Fiorella Inche Velezmoro for overall help with this thesis and great company at the office. A double thanks to Daniel Aasbø for corrective reading of the thesis.

Contents

Acronyms

ADC	Analog-to-Digital Converter.
CAN	Controller Area Network.
CD	Compact Disc.
CMOS	Complementary Metal-Oxide-Semiconductor.
DMIPS	Dhrystone Million Instructions Per Second (MIPS).
DSP	Digital Signal Processing.
ECC	Error Correcting Code.
GPIO	General Purpose Input/Output.
GUI	Graphical User Interface.
I2C	Inter-Integrated Circuit.
IC	Integrated Circuit.
LEO	Low Earth Orbit.
MCU	Microcontroller Unit.
MeV	Megaelectronvolt.

MIPS	Million Instructions Per Second.
MOS	Metal Oxide Semiconductor.
NUTS	NTNU Test Satellite.
OBC	On-Board Controller.
OTP	One-Time Programmable.
PCB	Printed Circuit Board.
RAM	Random Access Memory.
SCL	Serial Clock Line.
SDA	Serial Data Line.
SEE	Single Event Effect.
SEU	Single Event Upset.
SOI	Silicon-On-Insulator.
SPI	Serial Peripheral Interface.
SRAM	Static RAM.
TID	Total Ionizing Dose.
UART	Universal Asynchronous Receiver/Transmitter.
USB	Universal Serial Bus.

List of Figures

List of Tables

Chapter 1

Introduction

This thesis is one of many concerning the student satellite program at NTNU, called NTNU Test Satellite (NUTS). A document outlining the mission objectives for NUTS is attached on the accompanying Compact Disc (CD). A short summary of the mission objectives is given below:

> *The NUTS (NTNU Test Satellite) project was started in September 2010. The project is part of the Norwegian student satellite program run by NAROM (Norwegian Centre for Space-related Education). The projects goal is to design, manufacture and launch a double CubeSat by 2014. The national student satellite program involves three educational establishments, namely the University of Oslo (UiO), Narvik University College (HiN) and NTNU.*
>
> *As main payload, the NUTS project will fly an IR-camera for atmospheric observations. In addition, a concept for a wireless short range data bus connecting different subsystems will be added. For communication, the satellite will use the common amateur radio bands and fly one transceiver for each frequency. During the first half of 2011, ten students from different departments and curriculums were involved in the project.*

It has previously been done research on building a student satellite at NTNU. This research stems from both previous projects, and a precursory

study for the NUTS project. Some previous experiences have been incorporated into the research.

The NUTS will be orbiting Earth way out of reach for us to do any physical maintenance after launch. It is also too far out to be protected from solar radiation by our planet's atmosphere. Because of this, it is extremely important that the satellite is designed to be self-reliant and robust. There are many aspects to consider to design such a system. This thesis focuses on three main points; the internal communication, providing a systems platform for maximum lifespan and an OBC.

1.1 Internal Communication

The internal communication of the satellite will consist of a regular wired bus for main internal communication, and an experimental wireless bus. Wireless intra-satellite communication has not yet been widely tested, and one of the mission objectives is therefore to test if this is a viable replacement.

1.2 Backplane

The basic platform of the satellite is refered to as the backplane. It will consist of all the necessary circuitry for providing power and communications to the payloads, and also for isolating and restoring misbehaving sub-systems. In short terms; it is the connection hub for all sub-systems.

1.3 On-Board Controller

An OBC is responsible for much of the housekeeping of the satellite. It has the main responsibility for monitoring the health of the system, and to take any necessary actions. It should keep a log of all events and vital mission data, and be able to offer processing capabilities for other systems.

The accompanying CD contains developed code, as well as source files for schematics and layout.

Chapter 2

Background Information and Theory

This chapter will present various theory and concepts useful for understanding the rest of this thesis. Some electronic devices used are also presented.

2.1 The Space Environment

The environment in space is very different from the normal operating conditions of electronic devices. This section will outline these differences, and reflect on the difficulties they pose.

2.1.1 Space Radiation

Space radiation is one of the biggest concerns when dealing with satellites in orbit. It consists of high energy particles radiated from several sources in space. [1] These particles comes from cosmic rays, as well as our own sun. Cosmic rays mainly consists of the nuclei of a variety of atoms, but also includes subatomic particles such as positrons and electrons.[2] Other stars, novas, supernovas and other objects are responsible for the cosmic rays. The radiation from the sun mostly stems from the fusion processes in the core, which results in a solar wind of electrons and protons. The strength of the

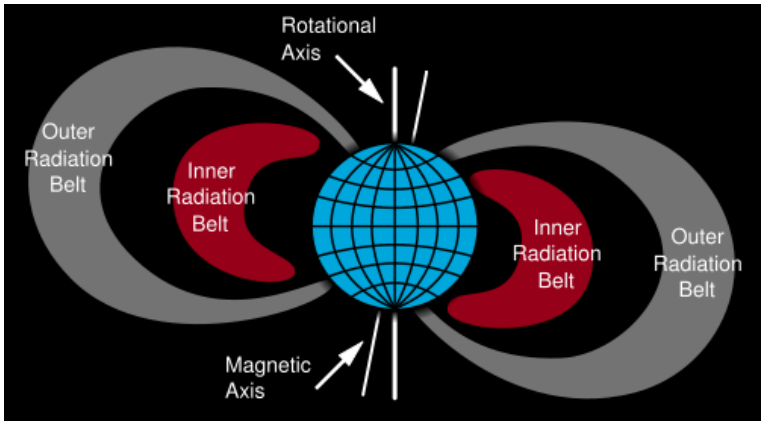

Figure 2.1: Illustration of the Van Allen belts. Courtesy of NASA.

solar radiation will vary with the sunspot activity, and will peak during solar flares.

When the radiation reaches Earth, it will be affected by Earth's electromagnetic field. This will cause some of the radiation to accumulate in the upper atmosphere in what is called the Van Allen Radiations belts. There are two belts, the inner and the outer belt. The inner belt mostly consist of protons with energy above 3 Megaelectronvolt (MeV). The outer belt contains less energetic protons, but has electrons that can have energies of several hundred MeV. [3] The inner Van Allen belt has its centre at approximately 2,000 km above the surface of the Earth. However, measurements have shown that the belt is descending closer to the poles, and that the radiation can have considerable concentrations at lower altitudes. An illustration of the Van Allen belts are shown in figure 2.1.

Effects of Radiation on Electronic Components

The bombardment of highly energetic space radiation can cause some troubling effects on electronic components. The total radiation dose will lead to a constant degradation of the components, and will eventually cause the device to fail. In addition, the radiation can cause spontaneous Single Event Effects (SEEs). Radiation in the form of protons or heavy ions will release energy by ionization when it penetrates the components. The ionization will give a charge at the node, and lead to a short current pulse.

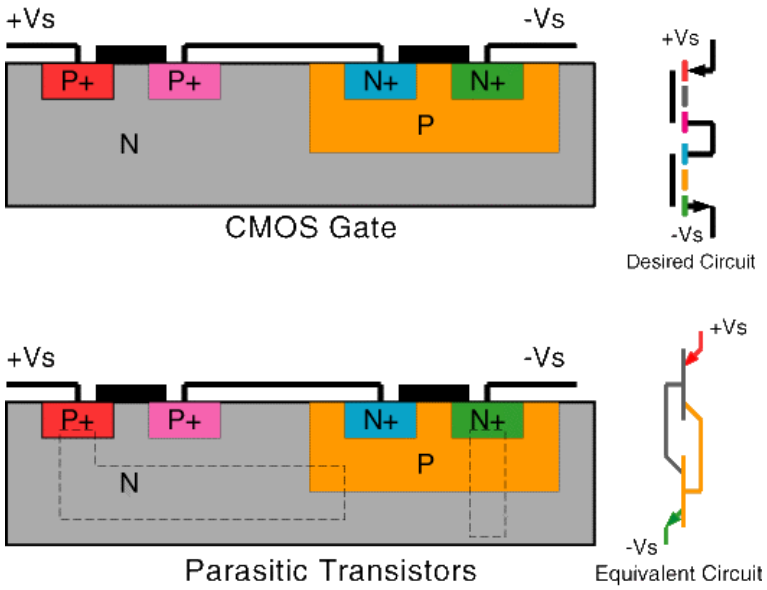

Figure 2.2: A standard CMOS inverter with parasitic transistors. Courtesy of Aerospace Corporation.

Latch-up is a state found in Complementary Metal-Oxide-Semiconductor (CMOS) devices where a high current moves from the component's source voltage to ground. In this state, the component will not respond properly to input stimuli. [4] Normal operation can be restored by removing and reapplying power, but heat dissiapation from the current flow might have destroyed the component.

The problem can be understood by considering the standard CMOS logic inverter seen in figure 2.2. The die has a PNPN-structure which results in the equivalent circuit of two connected transistors. When high energy radiation penetrates into this structure and releases some of it's energy as a current spike, it can activate one of the transistors. The current consumption will increase due to positive feedback in the circuit, and ultimately become a short-circuit. [5]

A Single Event Upset (SEU) is a less devastating effect, but can still cause mayhem on the circuit's operation. Consider a storage element such as a flip-flop, NAND flash or SRAM cell. A heavy ion can penetrate into the structure and leave a charge that flips one or more bits. The consequence

of this SEE will depend on the application's interpretation of the specific bits, but could possibly be mission-threatening if it affects e.g. navigation or power controls.

Total Ionizing Dose (TID) refers to the amount of ionizing radiation a component has accummulated over time. The ionizing radiation leads to a change in the physical parameters of the devices. More specifically it will result in a shift of the threshold voltage of Metal Oxide Semiconductor (MOS) transistors. [6] This could lead to a higher leakage current, and eventual failure of the electronic components.

Improving Radiation Tolerance

Physically shielding by encapsulating the sensitive components in a radiation absorbing or reflecting material is possible. Depending on the material characteristics and thickness it will reduce the amount of radiation reaching the components. However, the shielding will not be able to stop all the radiation, and especially not the most energetic particles that causes SEEs.

Fabrication processes exists that can be used to counteract latch-up. Latch-up is dependent on the existence of the PNPN-structure. For the most common CMOS fabrication process this structure will be present on numerous places. Luckily, there are design methods and alternate fabrication processes to counter the PNPN-structure. [7] One design method is to disrupt the current flow path by placing an n-well contact to the power supply for each pFET connected to the power supply. [4] An example for latch-up tolerant processes is the Silicon-On-Insulator (SOI) process. SOI builds it's transistors on an insulator instead of a silicon substrate. This won't lead to the formation of the PNPN-structure.

Redundancy is a way of increasing the reliability of a system by duplicating sub-systems or information. This is especially important in space applications, where the equipment is constantly subject to a bombardment of radiation, which results in SEUs and degradation. However, redundancy will add some extra space and complexity to both hardware and software.

Error Correcting Codes (ECCs) will assure data integrity by adding redundant bits to a bulk of data. [8] These bits are calculated using special algorithms to allow for detection and even correction of bit errors. There are

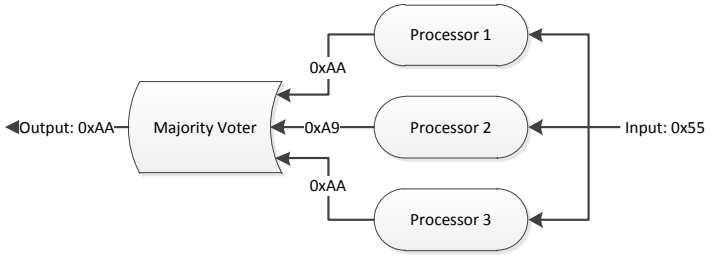

Figure 2.3: A majority voting system, processor 2 is defective.

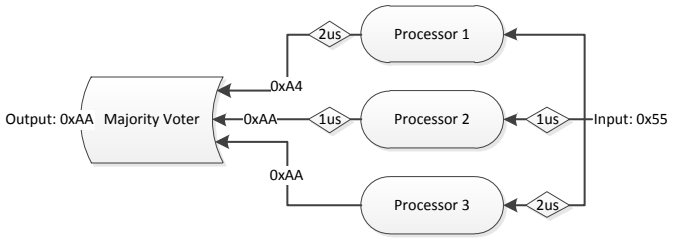

Figure 2.4: A majority voting system with time delay.

a number of different algorithms such as Hamming codes and Reed Solomon codes, which vary in calculation complexity and storage efficiency (i.e. how many bits are redundant). Some Microcontroller Units (MCUs) have integrated generators for some codes, which will heavily increase the throughput.

Majority voting is useful when adding redundant duplicated pieces of hardware for processing. It is used to decide which result is the correct one, and which sub-systems have been damaged and produces the faulty response. This can be done by majority voting where the result from all the redundant systems are compared, and the most common answer will be assumed to be correct. Figure 2.3 shows a system using majority voting.

The majority voting system can be improved by recognizing that some faults can stem from SEUs from space radiation. Then it is likely that many of the redundant systems are affected at the same time by a burst of radiation, e.g. from a solar storm. By adding different time delays for each system the calculations would take place at a different point in time, and decrease the possibility of a radiation burst changing the majority vote result. Figure 2.4

shows the majority voting system with time delay implemented.

2.1.2 Vacuum Conciderations

The concentration of particles of the atmosphere in Low Earth Orbit (LEO) is significantly smaller than on the surface of Earth. Since the atmosphere excert a pressure depending on the concentration of particles, the pressure is much higher on the surface. [9] Certain conciderations must be taken in order for the pressure difference of the design environment, the surface of Earth, and the flight environment, LEO, to not have a damaging effect. If bubbles of gas and liquid have been trapped in any component package or a Printed Circuit Board (PCB), it could lead to the pressure damaging the material and possibly the whole component or PCB.

Such bubbles can occur on a PCB during manufacturing if a via hole is buried between copper layers or solder mask. And even during soldering pockets of gas or liquids can be trapped in the solder. The components can also contain pockets from the packaging process. A danger also arises if any of these gases or liquids are acidic, which could result in damage to the whole satellite or even other satellites in the same rocket.

2.1.3 Space Debris

Space debris is an ever increasing concern for satellites and space vehicles. A vast number of old satellites, fragments of satellites, old rockets and similar is floating around Earth at different altitudes. [10] The debris count for various sizes are shown in table 2.1. In LEO, objects orbit at speeds above 7 km/s, which means that even small fragments can hold as much energy as a moving

Table 2.1: Estimated amount of space debris. Source: American Institute of Physics.

	0.1-1 cm	**1-10 cm**	**> 10 cm**
Total debris at all altitudes	150 million	650 000	22 000
Debris in low-Earth orbit	16 million	270 000	14 000

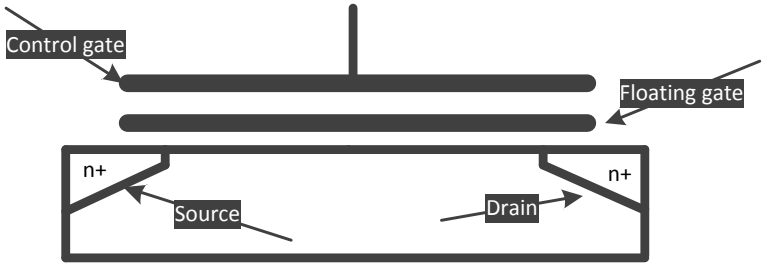

Figure 2.5: The floating gate transistor.

car on the freeway. All fragments between 1 mm and 1 cm are considered to pose a threat to satellites, while any fragment above 1 cm would definitely destroy it.

The only way to improve space debris resilience would be to shield the satellite with durable materials. This would make the satellite too heavy to fit into the CubeSat standards. Luckily, there is a vast amount of space and the chance of a collision is small, even though the possibility of a collision is real.

2.2 Memory

Memory devices are an indispensable part of all advanced systems, and are present in some form. This section will give a brief introduction to some types used in this thesis.

2.2.1 NAND Flash

NAND Flash storage devices are based on a floating gate MOS transistor. [11] A floating gate transistor can be thought of as a regular MOS transistor with an extra control gate on top of the normal gate, as illustrated in figure 2.5. A charge can be built up on the floating gate, which will hold the transistor in it's on or off state. The state of the transistor decides if a one or a zero is stored in the cell. The process of depositing charge on the floating gate varies among technologies and manufacturers. But one way of programming is to ground the P-substrate and N-well, and then applying voltage to the

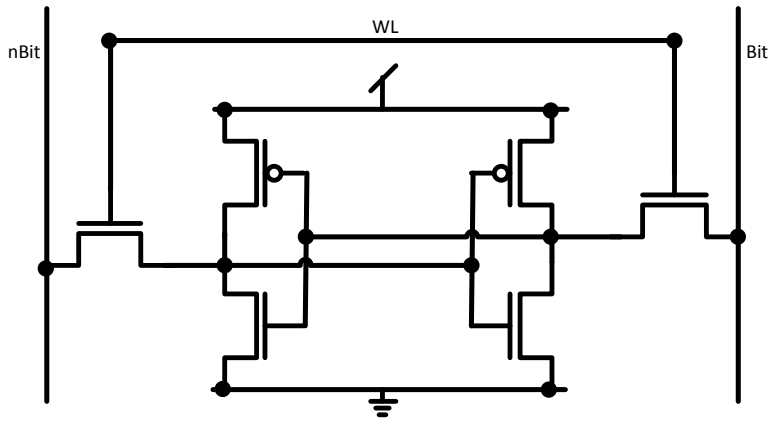

Figure 2.6: An SRAM memory cell.

control gate. This will create an electric field that pulls electrons onto the floating gate, making it negatively charged. The floating gate is charged positively by applying ground to the control gate and voltage to the source of the transistor.

2.2.2 Static Random Access Memory

A Static RAM (SRAM) memory cell is based on two connected transistor inverters, as shown in figure 2.6. [12] The inverters will keep the gate voltage of each other to retain the bit information. Reading is performed by activating the control transistors, which connects the inverters to the bit lines. Write is performed in the same way, but by applying the new bit on the bit line and thereby charging or discharging the gate.

2.2.3 One-Time-Programmable Memory

The One-Time Programmable (OTP) memory is a non-volatile memory that can only be programmed once. It is built up by an array of fuses that is permanently broken during programming. [13] There are many different configurations of OTP memory, but the basic principles are the same. Programming is performed by imposing a large enough current or voltage to damage the dielectric material. [14] Address lines are available to address

specific parts of the memory. Information is then stored by addressing the correct part and programming the fuses that should be programmed.

Since information is stored by structurally altering the component, it is considered to be more reliable than other memories in harsh radiation environments. In fact, certain technologies of OTP memories has proven to be specifically radiation tolerant. The resistance of a programmed OTP antifuse memory cell based on amorphous silicon is not affected by radiation, and the resistance of an unprogrammed cell will increase under radiation, i.e. improve. [15]

2.3 Digital Logic Series

Using digital logic is not as simple as just selecting the logic expression you want to realize and putting it together using the available Integrated Circuits (ICs). There are several key parameters to take into concideration, such as voltage levels, power consumption, speed and other internal aspects of the logic circuitry. For example a logic device that supports partial power down will not be partially powered through an I/O pin when it's power supply has been shut off. There are dozens of different families and technologies of digital logic to explore. [16]

2.4 AT32UC3A3

The AT32UC3A3 is a range of MCU devices built around the AVR32 core from Atmel. [17] It offers 32 bit processing at up to 92 Dhrystone MIPS (DMIPS), and a Digital Signal Processing (DSP) instruction set. It can handle many memory interfaces automatically, and supports multiple different memory devices connected to a common bus at the same time. The memory interface can be used together with an ECC generator to add redundancy. A Universal Serial Bus (USB) interface is available for connectivity to a computer.

2.5 nRF24LE1

The nRF24LE1 is a combination of a 2.4GHz radio transceiver and an 8051 MCU from Nordic. [18] It can transfer data at a maximum rate of 2Mbps, but can be lowered to acheive a better transmission reliability. The MCU offers several modules such as communication, encryption and Analog-to-Digital Converter (ADC). An external crystal can be connected to provide a low-power synchronization clock. Both the radio and the MCU has been optimized to be power efficient.

An OTP edition of nRF24LE1 is available. This version provides an OTP memory instead of the normal flash for firmware storage. In addition, some OTP memory is also available for storing data.

Chapter 3

Backplane

A system as complicated as a satellite will always be divided into several sub-systems. The designers working on those sub-systems will focus on completing their specific task. But eventually all the sepearate sub-systems has to be sown together into the completed satellite. So the question is; how will everything be tied together?

There are ultimately two main topologies for connecting the sub-systems together. A centralized approach, as shown in figure 3.1, will consist of a main computer functioning as the connection point between all sub-systems. The distributed system, shown in figure 3.2, consists of a common connection link shared by all sub-systems. Previous research on the subject has favored the distributed system because of reusability and scalability. [19] Scalability is important because the designer of the data bus-system does not necessarily know in what way the data bus will be used.

The use of a distributed system makes it much easier to define a common connector and pin-out for all sub-systems in the satellite. This introduces the thought of a backplane. A backplane is a board with several connectors that interfaces the sub-systems together. An illustration of the general thought of the backplane is shown in figure 3.3. The different sub-systems are then built as separate modules that fits directly onto the backplane connector. This solution makes it easy for designers to insert and remove single cards without having to dismantle the whole satellite every time a change must be made.

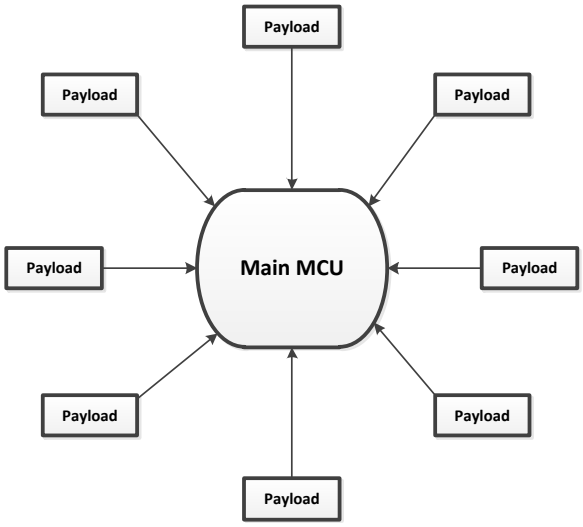

Figure 3.1: The centralized system solution.

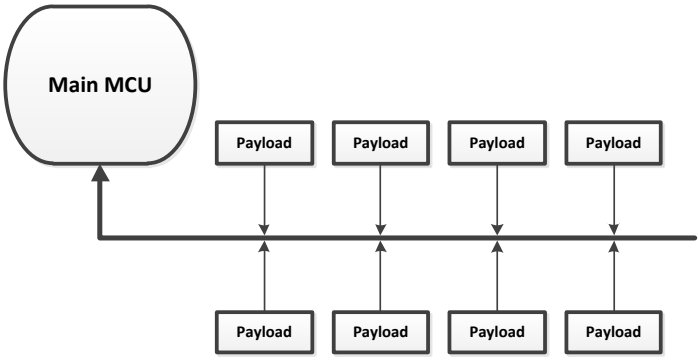

Figure 3.2: The distributed system solution.

Figure 3.3: An illustration of a backplane solution, from the NUTS-1 Mission Statement.

The backplane will have slots for 8 modules. Two of them will be reserved as extended functionality master module slots, one for a special purpose power module and the remaining five as regular payload slots. The two masters will be responsible for controlling most of the digital systems on the backplane, and do general housekeeping in the satellite. Both masters will have equal access to do everything, as a redundancy.

But there are still many considerations to be made. What if a module starts misbehaving and floods the data bus? Due to space radiation this is likely to occur eventually during the satellite's lifespan. There should be some mechanisms for detecting the misbehaving module, isolating it and hopefully correcting the error. This makes it necessary to add control logic to the backplane. The following section (3.1) presents a viable design solution.

3.1 Design

3.1.1 Data Bus

The data bus will be the main communication channel between all modules. It needs to be scalable and robust, provide a relatively high throughput rate and preferably have a low power consumption. In addition, it should be easy to use for the modules, and the best way to assure this is to select a commonly available standard for the data bus. There are many different standards available, both for serial and parallell communication. Parallell busses will require a much larger connector than the serial busses, therefore a serial interface is to prefer.

Many MCUs have integrated logic to automatically handle a range of communication protocols. Among the serial interfaces, the most found includes Inter-Integrated Circuit (I2C), Serial Peripheral Interface (SPI), Universal Asynchronous Receiver/Transmitter (UART) and Controller Area Network (CAN). The most important characteristics of each bus is summarized in table 3.1.

Table 3.1: Selected data bus characteristics based on common specifications.

Data Bus	Speed	Lines	Notes
SPI	20Mbps	4	One chip select for each device.
I2C	400kbps	2	Limited address and bus capacity.
CAN	1Mbps	2	Needs external circuitry. Noise resilient.
UART	1Mbps	2	Difficult to use between multiple nodes.

The most important aspects for the backplane is that the data bus should be equally available for all and be as simple as possible. This excludes SPI because it requires a master device to control the chip select signal, or some sort of complex logic to control the arbitration process. UART will also require som complex logic for arbitration since it is made for communication between just two modules. The CAN bus would have been a good choice because of it's noise resilience, but most MCUs do not have integrated support. It would therefore be necessary to add some extra circuitry, and increase the overall design complexity.

I2C

The I2C bus was selected as a common data bus. It is a very common data bus with integrated support in most MCUs, and many sensor chips uses this interface. The bus only requires two lines for communication; Serial Data Line (SDA) and Serial Clock Line (SCL). [20] The lines are controlled by open-collector logic, and pulled up to logic 1 by resistors in the idle state. This makes it possible to have multiple masters on the same bus, and implement collision detection and arbitration in software if there isn't already supported in hardware.

All devices are connected to these common lines, and addressed by a unique address for each device. The address space is limited to 7 bits giving 128 possible addresses, but a 10 bit addressing mode can be used if necessary. Some of the addresses are reserved for specific purposes, so the true number of available addresses is smaller depending on what devices are used. There will anyways be enough addresses for many modules and sensors connected to the bus. If more devices are needed there exists an abundance of I2C

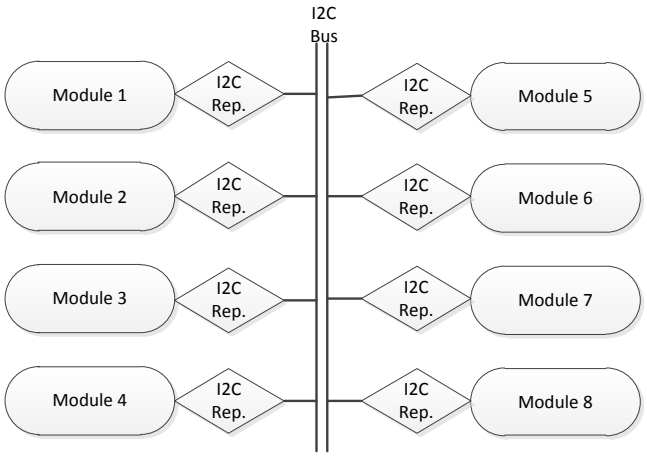

Figure 3.4: The proposed bus topology.

switches, multiplexers and hubs that can be used to create sub-domains of the I2C address space.

The total bus capacitance of the I2C bus will limit both the number of devices that can be connected to the bus and the total length of the bus. This can be solved by using I2C repeaters that replicates the bus conditions on both sides, acting like a buffer. These repeaters also have an enable/disable pin for connecting or disconnecting the two sides of the repeater, making them ideal for isolating malfunctioning devices from the bus.

Bus Topology

The bus has to be distributed in a reasonable way between all of the modules. The selected topology is presented in figure 3.4. The backplane contains a common I2C bus connected to an array of bus repeaters. Each of those bus repeaters are connected to a separate module slot, and one for a separate I2C bus on the backplane. The bus repeaters (PCA9517) has an active-high enable pin that can be used to isolate the module side of the bus from the backplane side. This bus topology makes it easy to isolate a separate module through the control logic.

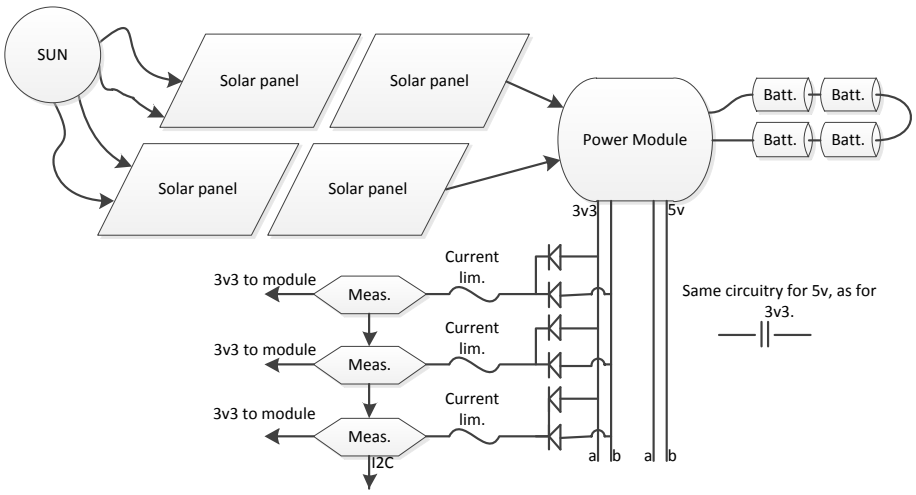

Figure 3.5: The satellite power system.

3.1.2 Power System

The power system, as illustrated in figure 3.5, consists of a set of solar panels, a power module, battery packs and a power distribution system. The power module has it's own special slot on the backplane because it is suppose provide the backplane with power, and not draw power as the other modules. It provides two regulated power levels of 3.3V and 5V, and two separately regulated lines of each for redundancy. On the backplane the redundant power lines are merged into one source for each of the modules. The merging is done through a special circuit that acts as an ideal diode, only leading one way. Following the ideal diodes is a current limiting circuit. It will cut the power whenever the module tries to draw too much current from either of the sources. There is also a circuit for reading the current consumption through the I2C bus.

It will also be possible to shut the power off for the separate modules. This is handled by the current limiting circuit which has an enable pin for the power. The current state can even be monitored by the master modules as a power flag, and proper steps taken to correct the behavior. All of this functionality will be handled by the digital logic on the backplane.

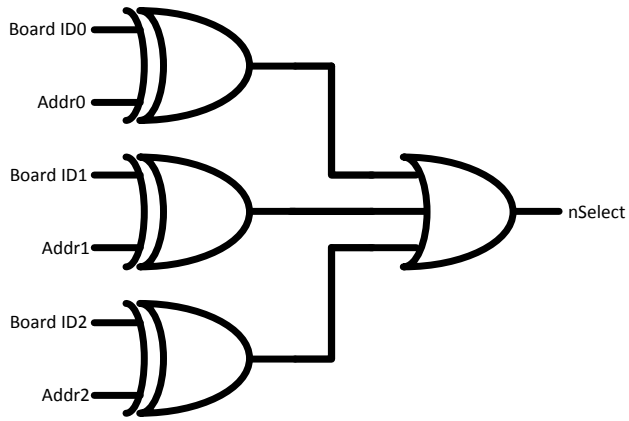

Figure 3.6: The address matching circuit for the module controls.

3.1.3 Digital Control

The digital control will handle shutting on and off power and data bus access for each module. In addition it will handle the reset signals and reprogrammability option for each of the modules. But if all the necessary signals were dedicated for each module, the master connectors would need way too many pins. It is therefore necessary to use a common control bus for all the modules, and add some kind of addressing to select the module. A more detailed explanation of the parts of the digital control system follows below.

Address Matching

Each of the modules will have their own address. When addressed, the module's select signal will go active. The simplest solution for doing this would be to have 8 select lines from the masters. This is still too many lines. A better solution is to use a 3-to-8 decoder circuit. The master modules can then use 3 lines to address each of the 8 modules. But the 3-to-8 decoder would have been a single point of failure, that could potentionally crippled the whole system.

The solution was to break the 3-to-8 decoder into individual address

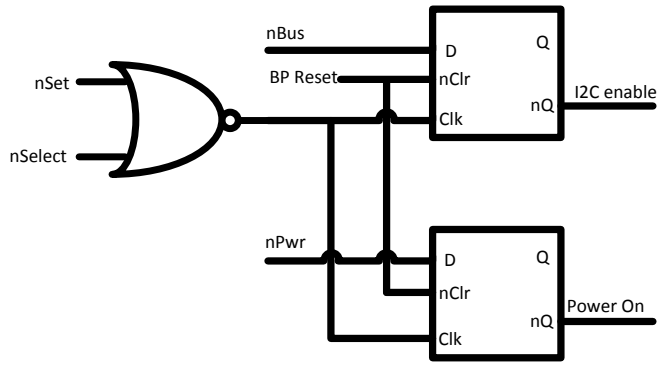

Figure 3.7: Power and data bus access control.

matching circuits, as shown in figure 3.6. The XOR-ports compares the value of each address bit and gives a low signal when it is a match. The OR-port will then check for a mismatch by propagating any high signals. This makes the select signal active-low.

The address for each module is hardwired onto the module itself. This makes it possible to address the specific module and not just the slot on the backplane. The modules can then be swapped without reprogramming the masters.

Data Bus and Power Control

The data bus access and power switch is simply controlled by enable pins. These two pins could have been controlled directly from the master modules, but because of the pin count a better solution had to be found. A D-type flip-flop is used for each signal to store the state, as shown in figure 3.7. The control bus for these flip-flops consists of a bus enable, power enable and clock signal. These signals are used together with the select signal to control the state of the flip-flop outputs. Figure 3.8 shows the correct waveforms for setting the states.

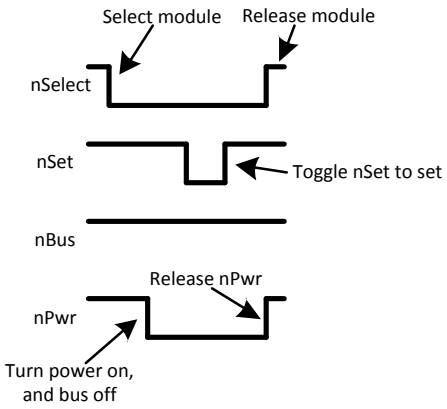

Figure 3.8: Control waveform for power and data bus access flip-flops.

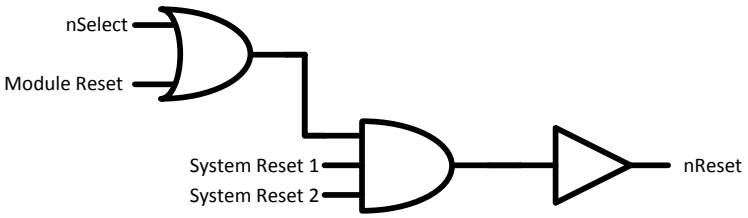

Figure 3.9: The backplane reset circuitry.

Reset System

When a module starts misbehaving it would be beneficial to be able to do a reset of the affected systems. This could be done by powering the module on and off, or by pulling on the reset line of all the digital circuits on the module. The reset circuitry is shown in figure 3.9. There are three separate reset lines on the backplane. The module reset is used to reset the individual module being addressed at the moment. The two system reset signals are used to reset all of the modules and the backplane as well. There is one system reset per master, so that the master does not reset itself when it pulls on the system reset line.

In the case of a lockup of both masters the backplane includes a watchdog timer. A lockup could occur if power for both masters have been shut down.

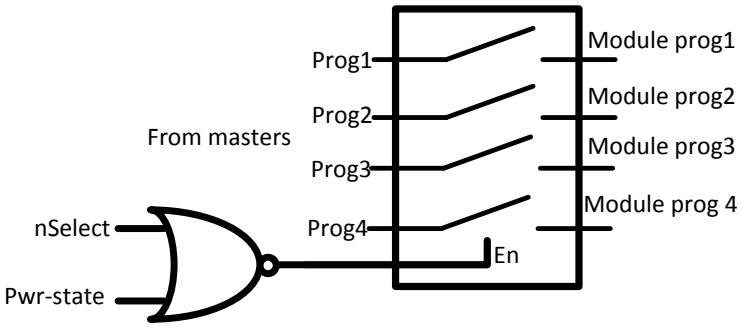

Figure 3.10: The backplane reprogramming circuitry.

Then there won't be any controlling modules on the backplane, and the power can't be turned back on. The watchdog timer will then trigger and reset the power and bus flip-flops and reset the entire system. To prevent the watchdog timer from triggering one of the masters must toggle the first addressing pin.

Reprogramming Modules

In a harsh radiation such as space, memory cells will suffer from degradation and bit flips. If a firmware flash on an MCU is affected it could possibly render the device useless. The only way to recover the device would then be to reprogram the device. Most devices uses an interface, e.g. JTAG, ISP, TPI etc., with 4 lines or less to program the device. Therefore four signals are routed from the masters to each module slot as a programming port. The programming port is as default disconnected, and is connected whenever a module is selected. The circuitry for handling this is shown in figure 3.10.

There doesn't have to be made any decisions regarding the actual programming protocols to be used. It will just be a matter of implementing the various necessary protocols on the master modules later. Then the master will know which protocol to use depending on which address is going to be reprogrammed.

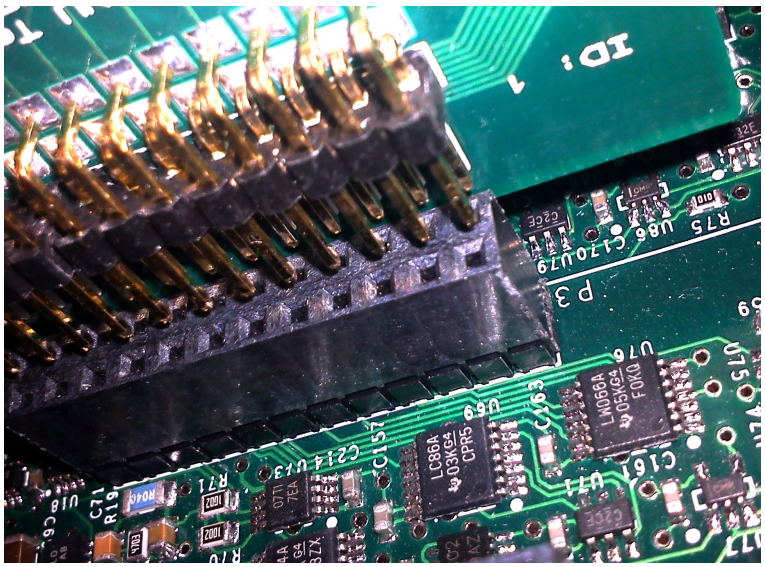

Figure 3.11: The module to backplane connector.

3.1.4 Mechanical Considerations

The satellite needs to be a mechanically reliable system. During launch the satellite must endure immense accelerations and vibrations. This means that there are special concerns to be handled regarding the connectors used for each of the modules, and also for how the backplane itself is secured in the satellite. It is even important to concider component packaging and placement to build a mechanical stable system.

The connector for the modules should be capable of providing the module with mechanical support in the satellite. It will act as an anchor point, and support the module together with the satellite's frame. The selected connector is a regular 2.54mm pitch two row connector, as shown in figure 3.11. It should provide sufficient support and be mechanical robust.

The circuit board of the backplane itself is shaped to be able to secure to the frame of the satellite. It is meant to fit into slots in the frame, and be secured by screws in the corner. The shape of the circuit board can be seen in figure 3.14.

3.1.5 System Overview

An overall overview of the backplane system is shown in figure 3.12. It consists of six 12x2 pin 2.54mm connectors, for regular payload and power, and two 18x2 pin 2.54mm connectors for the master modules. It is proposed that the master modules are the radio module and the OBC. The masters have the ability to control access to the common I2C bus, toggle power for each module, reprogram modules and reset specific modules. The connector pinouts for both the master modules, power module and payload modules are shown in figure 3.13.

3.2 Prototyping

The proposed solution has been realized and tested. The schematics are presented in appendix A. Figure 3.14 shows the PCB, both unassembled and assembled.

3.2.1 Testing

The basic functionality of the backplane has been tested by the steps shown in table 3.2. The first step was to do a visual inspection of the circuit board to check that all ICs were mounted correctly, and that there were no short circuits before powering the board. Power was applied to the power slot of the backplane, and the power levels throughout the board were verified using a multimeter. The redundancy of the power system was tested by removing power to one of the power contacts from the power slot, and then measuring that the power was still present. The current limiting circuits were also tested by short-circuiting the power terminals.

The digital systems were thoroughly tested in incremental steps. Proper I2C signalling, communication and full speed has been tested by checking that the I2C current monitor circuits could be read. Then, by hard-wiring addresses for each slot, an acknowledge signal was verified when the correct address was applied. The remaining tests were performed by attaching an MCU to the control ports. Bus access, power and reset signals have been

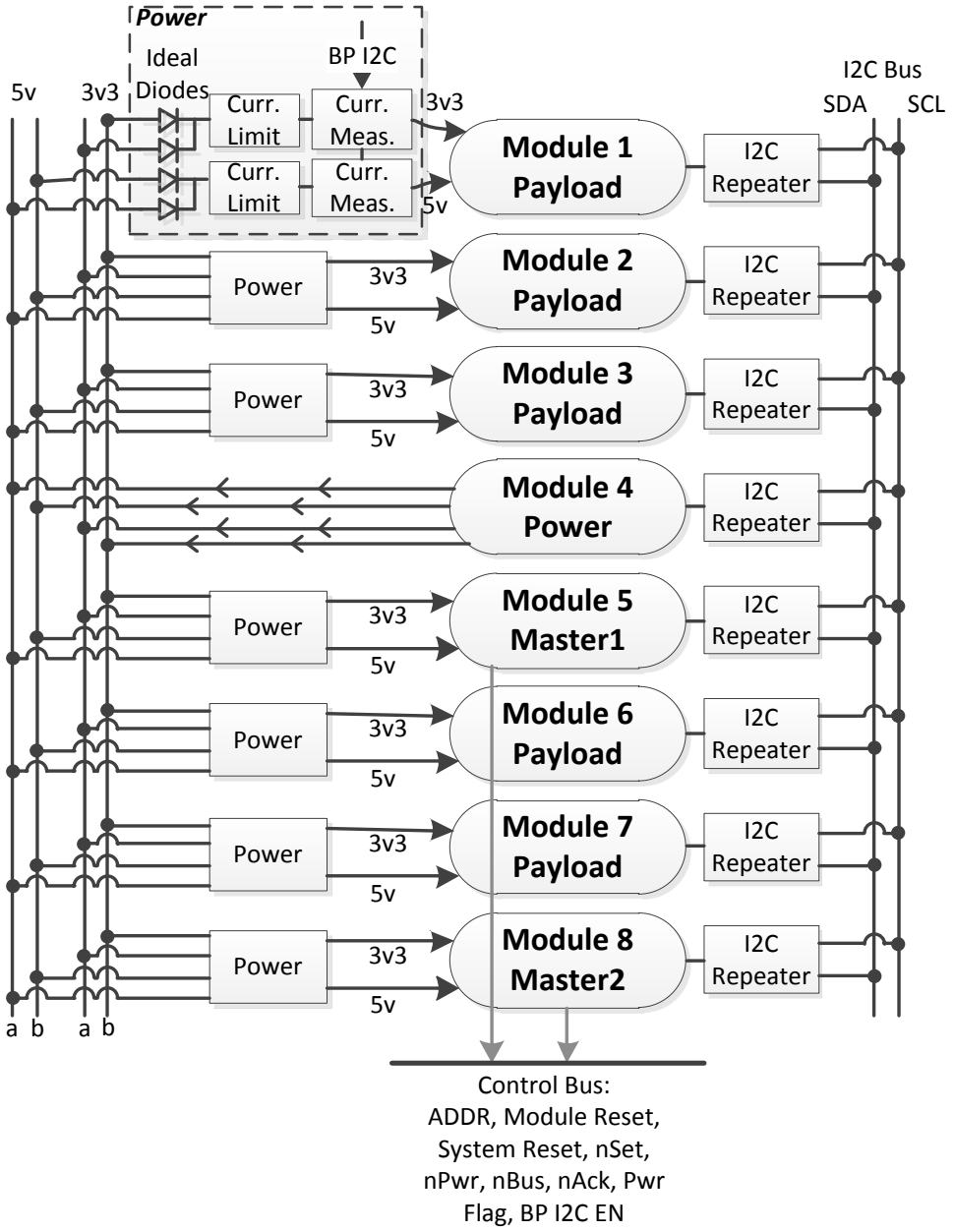

Figure 3.12: An overview of the backplane, excluding control logic.

I2C SCL	1	2	3V3
I2C SDA	3	4	Gnd
Gnd	5	6	5V
nReset	7	8	ID0
Module PROG1	9	10	ID1
Module PROG2	11	12	ID2
Module PROG3	13	14	nSet
Module PROG4	15	16	nAck
Gnd	17	18	PWR FLAG
I2C SCL	19	20	3V3
I2C SDA	21	22	BP I2C EN
Module Reset	23	24	5V
PROG1	25	26	nBus
PROG2	27	28	nPwr
PROG3	29	30	ADDR0
PROG4	31	32	ADDR1
System Reset	33	34	ADDR2
Gnd	35	36	Gnd

I2C SCL	1	2	3V3
I2C SDA	3	4	Gnd
Gnd	5	6	5V
nReset	7	8	ID0
Module PROG1	9	10	ID1
Module PROG2	11	12	ID2
Module PROG3	13	14	RESV0
Module PROG4	15	16	RESV1
nReset	17	18	RESV2
Gnd	19	20	3V3
I2C SCL	21	22	Gnd
I2C SDA	23	24	5V

I2C SCL	1	2	3V3_A
I2C SDA	3	4	Gnd
Gnd	5	6	5V_A
nReset	7	8	ID0
Module PROG1	9	10	ID1
Module PROG2	11	12	ID2
Module PROG3	13	14	RESV0
Module PROG4	15	16	RESV1
nReset	17	18	RESV2
Gnd	19	20	3V3_B
I2C SCL	21	22	Gnd
I2C SDA	23	24	5V_B

Figure 3.13: Pinouts for master slots (left), payload slots (middle) and power slot (right).

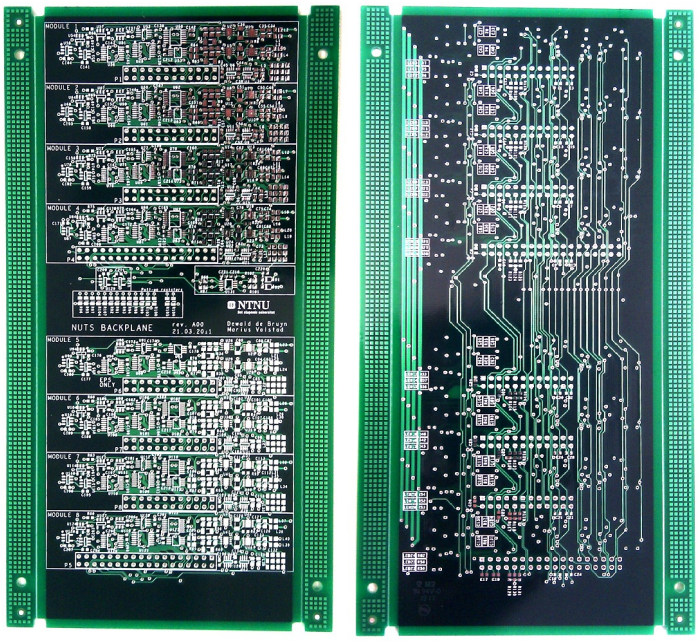

Figure 3.14: The circuit board for the backplane.

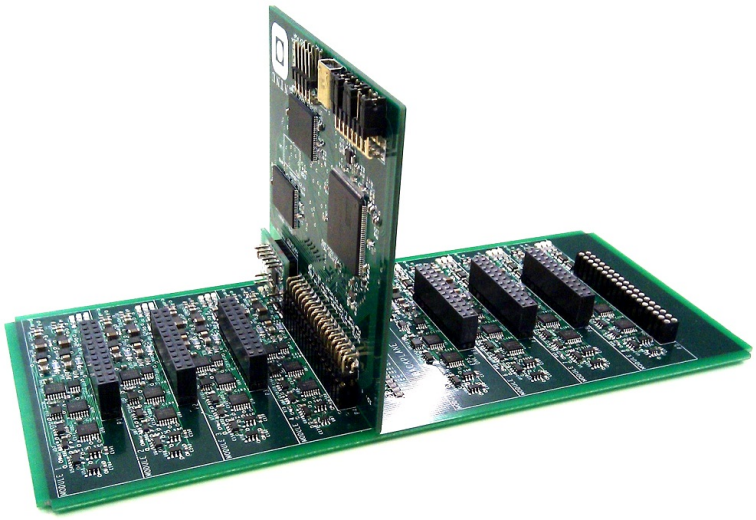

Figure 3.15: The finished backplane with OBC module.

toggled using the control bus. The programming interface have been tested using a JTAG interface connected to another MCU.

Table 3.2: Testing specification of the backplane.

Step	**Functionality**	**Specification**
1	Visual inspection	Check components and soldering
2	Basic power	Apply power to both
3	Redundant power	Remove power from one
4	Over-Current protection	Short-circuit power signals
5	I2C signalling	Pull and release I2C lines
6	I2C communication	Read current measurement ICs
7	I2C speed	Read at 400kHz
8	Module addressing	Addressed a slot
9	Power and bus control	Turn on/off controls
10	Ack and power flag	Address module and check flags
11	Reset system	System and module reset
12	Programming bus	Signal propagation to module
13	Programming bus	JTAG programming

3.2.2 Results

Overall, the backplane is functioning as expected. However, there have been a few problems regarding the pull-up resistors for the I2C bus and the first address bit. The bus repeaters needed a lower resistance than the default value of 10k ohms on the common backplane side, to be able to release the bus after a zero was transmitted. This is probably due to the load all the repeaters puts on the common bus, and the resistor is not sufficient to pull the line high again. The first address line also had troubles being pulled high. The watchdog timer required a larger current to pull the line high.

The maximum speed of the I2C bus exceeds the specified 400kHz, but communication speeds should still be limited to 400kHz for reliable communication. The programming bus was tested at speeds up to 500kHz, but it can probably work with higher speeds.

Power Consumption

The backplane's power consumption was measured to be ∼20mA. The backplane logics runs at 3.3V, and the total power consumption is thus 66mW. The power consumption will of course increase with control and I2C bus activity because of the pull-up resistors.

Chapter 4

On-Board Controller

The main tasks of the On-Board Controller are to keep track of the satellite's health, help to increase it's life span and provide support with storage and/or processing capabilities. It will serve as one of the two backplane masters, and will use it's heightened responsibility to monitor and act on all communication and behavior of the satellite.

Since the OBC will have a supervising responsibility, it is natural that it retains a log of events and house-keeping data. It could be interesting to be able to download this log, or at least a summary, and analyze the behavior of the satellite. Especially getting detailed logs of error condition would be helpful.

It is also desirable that the OBC can be used as a helpful tool before flight as well. The OBC can perform debug tasks as monitoring the messages on the data bus and power consumption, reprogram modules on request etc. It would therefore be advantageous to provide an option to interface the OBC to a computer.

4.1 Hardware Design

The hardware will mainly consist of the connector for the backplane, an MCU for processing and memory for different purposes. Because of space and power restrictions, hardware redundancy will not be prioritized. Although,

methods for increasing reliability are discussed and planned for. Methods for using the OBC for manual backplane control are also implemented.

4.1.1 Microcontroller

There were several issues to think about before selecting an MCU. It had to provide low-power modes or sleep modes for saving power, and also be able to process a sufficient amount of data. Since the board were to contain a lot of memory, it was also desirable to have integrated support for various memory busses. Another aspect was that the MCU should preferably be of a familiar architecture for both the author and the succeeding developers.

The final choice was the AT32UC3A3256 MCU from Atmel. It runs on the AVR32 core which is designed with power-saving in mind, and offers processing capabilities with up to 92 DMIPS. There is some DSP instructions available, which could be useful for image processing or other sensory equipment. There is also internal support for many different memory types, and an ECC generator with support for both Hamming codes and Reed Solomon codes.

4.1.2 Memory Hierarchy

The memory setup will be used for logging events in the satellite, temporary processing storage, module firmwares and possible more. It must therefore contain some different type of memory to accommodate all uses. It should also be taken into consideration that some memory types are volatile against radiation, and can suffer bit-flips. Also, all memories that are not radiation hardened are vulnerable against latch-up in the on-chip digital circuits, but those will hopefully be handled by the power system of the backplane.

First of all an SRAM chip is needed to expand on the limited amount of on-chip Random Access Memory (RAM) on the MCU. This type of memory has a low access time and will help speed up the calculations. Unfortunately, they are sensitive to radiation and bit-flips are a potential danger. [21] A radiation hardened SRAM would have solved this, but as the availability of those are limited a regular variant is still used. The specific chip used

is IS61WV102416BLL-10TLI, and expands the MCU RAM with 16Mbit of storage.

For a more long-term logging of data a non-volatile memory was needed. There is also a possibility of SEU for non-volatile memories, so the idea was to store redundant information to be able to detect any anomalies. By using a sophisticated ECC it is possible to use a relatively small amount of bits to detect a fair amount of bit errors, and even correct them. A NAND flash has been used because it comes in relatively large sizes, and there won't be any concerns for having too little space for adding redundancy. A 16Gbit flash, MT29F16G08DAAWP, gives the possibility of storing a lot of housekeeping data and even images.

The firmware for the different modules in the satellite are most commonly stored in a on-chip flash on the separate MCUs. This flash is also volatile to radiation, and can be struck by a SEU that flips a bit. It will then be necessary to have something to compare the firmware with to detect the error in the flash. When an error is detected the known correct firmware can be uploaded. A correct firmware version must then be available for the OBC. This is done by using an OTP memory which is known to be specially tolerant against radiation. The OTP memory will contain the firmware and perhaps other default settings for all of the modules, and can be used to compare and correct errors.

4.1.3 Backplane Connector

Since the OBC is a master, it will have a master connector following the pinout from figure 3.13. The I2C bus signals are connected to the General Purpose Input/Output (GPIO) pins with integrated I2C support. This means that the MCU can communicate over the I2C bus with much less processing power. The control signals are connected to normal GPIO, and will be configured to use open-drain signalling. This is done to assure that if the two masters fight over control of the backplane (i.e. one master want to transmit 0, and the other master tries to transmit 1) they won't harm the output stage of the GPIO pin.

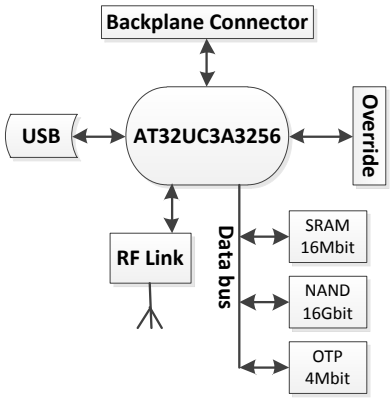

Figure 4.1: A system overview of the OBC.

4.1.4 USB Interface

An USB interface would be a simple way to communicate with the satellite. It could be used to debug and configure the satellite, both in the development process and just before launch. Since the selected MCU has direct support for USB it was not necessary to add any new chips to the design. The MCU is programmed to enumerate on the computer as a standard COM-port and the user can then decide to use a simple terminal or a specialized program to communicate with the satellite.

4.1.5 Manual Override

There are also possible to override the backplane control signals manually. By using some jumpers placed at the edge of the OBC most of the backplane control signals can be operated to e.g. turn off power for a module, select a module for programming and check the power state.

4.1.6 Hardware Overview

An overview of the hardware design for the OBC is shown in figure 4.1. The AT32UC3A3 MCU is connected to the memory hierarchy with a common

parallell bus. The backplane control signals are connected directly to general purpose input/output pins of the MCU, and some are also connected to the manual override headers. The USB connector is connected to the MCU as well. The wireless module shown in the figure will be discussed in chapter 5.

4.2 Software Design

The OBC will need a complex piece of software to function as a master for the satellite. It has been given many tasks, and all of them needs to be handled in an efficient and secure manner. Because of it's heightened status as a master on the backplane, it has a lot of responsibility to not abuse it's power and perform potentially devastating mistakes. This has to be kept in mind when developing the software.

A lot of the development work for the OBC software still remains, but an overall idea for the various tasks and abilities of the software is outlined. Some aspects have been tested, and hardware specific drivers have been developed and can be used in the future.

4.2.1 Drivers

The OBC contains 3 different types of memory devices which have been interfaced to the same bus interface, but in a slightly different way. This requires some configuration and setup in software in order to work as expected. The SRAM and OTP memory uses a standard random access addressed interface, but the NAND flash uses a command interface to read and store pages of data. This requires a little framework in order to function.

The USB interface also needs some drivers to work. Actually it would take several days, or even weeks, of work to get a usable USB stack up and running. Fortunately, Atmel provides a framework of drivers for most modules and the USB interface is one of them. This framework has been used for both the USB interface and for the memories, with the exception of the NAND flash command interface.

Also, some drivers for the backplane has to be developed in order to control all the functions properly. This includes turning the power and bus ac-

cess on/off, resetting modules, interpreting bus activity and reprogramming modules. Even the reprogramming must be taken care of by implementing a driver for each of the supported protocols.

4.2.2 Housekeeping

Tasks as supervising the power consumption, bus activity, module health etc. falls under housekeeping. This needs to be handled by the OBC by periodically reading the current measurement circuits on the I2C bus, checking that communication is valid and sending a ping to each of the modules. Housekeeping data should also be stored to the NAND flash and could be used to create a summary for download to the ground station. Then the ground station can request more detailed data of a duration in time that seem interesting.

4.2.3 Operating System

Since the OBC will perform many separate tasks, it would be easier, more efficient and more secure to use some sort of small operating system. There are many operating systems available to select from, and a suitable one should be implemented at a later time.

4.2.4 Support Software

The support software will consist of a python script with a Graphical User Interface (GUI). It will have the ability to connect to the OBC and monitor and control all of the OBC actions. E.g. power consumption, bus activity, module firmware etc. can be read and configured from the interface. All data can be sent and received using the USB interface with the COM-port emulation.

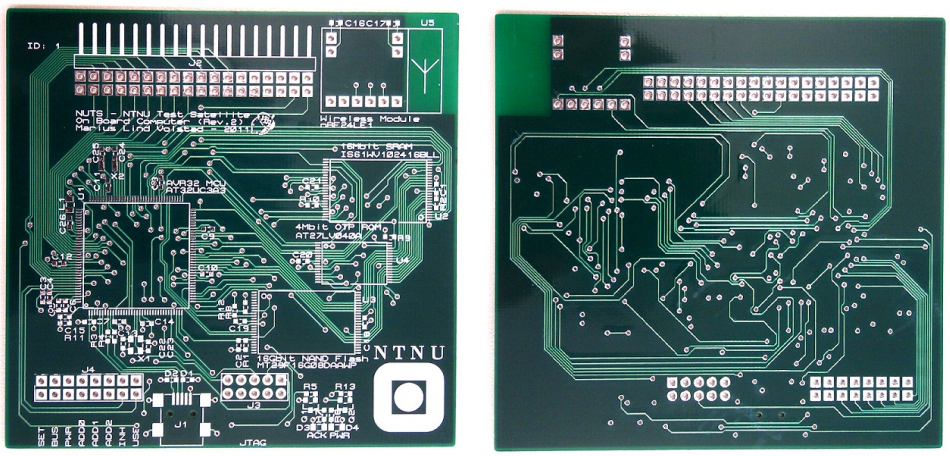

Figure 4.2: The circuit board for the OBC.

4.3 Prototyping

A prototype of the OBC has been developed and tested. The circuit diagram can be found in appendix B. The circuit board is shown in figure 4.2, and an assembled version is shown in figure 4.3. Note that the OTP memory is not mounted since it has to be programmed using a special programmer before mounting. The extra PCB on the top of the figure is the wireless internal communication module discussed in chapter 5.

4.3.1 Testing

The OBC module was tested by attaching it to the backplane through a master slot before applying power to the backplane. Then using the manual override connectors in front on the OBC module, the address was set to the OBC module itself. A JTAG programmer was attached to the JTAG connector, and the chip ID of the MCU was read. To be able to fully debug the rest of the system a firmware emulating a COM port through the USB interface was programmed onto the MCU

A simple command interface using single letters, controlled which test the MCU was going to run, and the result of the test was printed back on

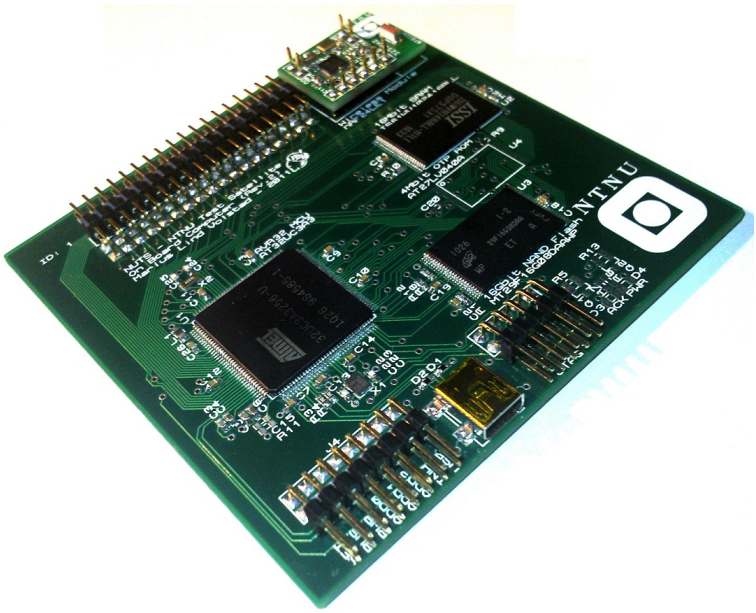

Figure 4.3: The finished OBC.

the terminal. A full write/read-test of the SRAM was performed, as well as a simple read test on the NAND flash. The OTP memory has unfortunately not been fully tested since the programming socket was not available.

The backplane was also controlled by the OBC to check that all the connector pins were correctly placed. The I2C bus was used to read current measurements from the backplane, and a simple JTAG programming driver was written to read the ID from modules.

4.3.2 Results

All functionality of the OBC has proven to work. The memory ICs are able to share a common bus, and operate at the timing specifications given in the datasheet. This means that the PCB routing is not a limiting factor for the throughput.

Power Consumption

The power consumption of the OBC module will rely heavily on the firmware for the MCU. In the worst case scenario with USB connected using no sleep modes, not shutting of clock domains in the MCU or other power optimizations the power consumption was ∼45mA, or ∼150mW. When putting the OBC into static sleep mode, which means all clocks are stopped, the power consumption fell to ∼4.5mA, or ∼15mW. The datasheet for the SRAM chip reports a nominal consumption of 4mA in retention mode, so below 0.5mA is used for the NAND flash.

It is much potential for saving power by carefully using sleep modes as much as possible. Even when the MCU can't sleep it can save a lot of power by using a low operating frequency when little processing power is required. A high frequency should be used when processor intensive tasks should be done, so that the MCU can go back to sleep as soon as possible.

Chapter 5

Internal Wireless Communication

There are certain disadvantages by using a standard electrically wired data bus for communication between the modules. A short circuit is the first problem, which would cripple the whole bus. The only way to protect against this would be to have an extra data bus for redundancy. Another problem would be if a module starts flooding the bus with garbage data.

One solution for all of this is to use a wireless data bus instead. There are many different wireless communication ICs available, and many are optimized for low-power and low-size applications. They also give the option to change the communication frequency, which would fix the problem of a module flooding the data bus. A wireless bus is also more flexible since it doesn't require any concern for how to connect the different modules together.

Some of the modules in the student satellite will be equipped with a wireless internal communication module, as described in this chapter. The goal is to test if a wireless internal data bus is a viable solution, and if it could replace the wired bus in the future. It should be as invisible as possible for the module MCUs, i.e. provide a seamless wireless link between the modules. Therefore the wireless module has to be smart and be able to transmit and receive data without help from the module MCU.

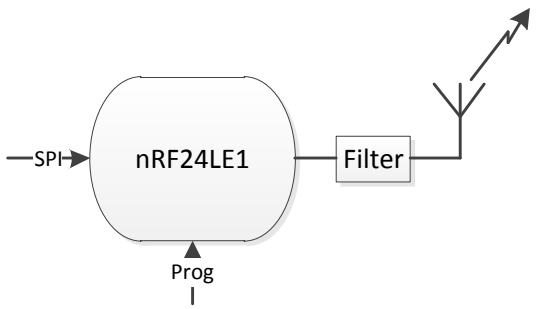

Figure 5.1: An overview of the wireless transceiver module.

5.1 Hardware Design

Since the wireless module should be able to process the wireless transmissions itself, it should contain some sort of a processing unit. There are a number of single-chip solutions with a wireless transceiver and a MCU on the same chip. One of these are the nRF24LE1 from Nordic Semiconductor. This chip has been selected because it can transmit at a rate of up to 2Mbps, and comes in a variety where the firmware for the MCU is stored in an on-chip OTP memory. This will help improve the radiation tolerance of the wireless modules. Figure 5.1 shows an overview of the wireless module hardware.

The wireless module contains two crystal oscillators connected to the nRF24LE1 chip. A 16MHz crystal is needed for the wireless transceiver and as a MCU clock, and a 32.768kHz crystal will be used by the chip to synchronize timings of the wireless protocols. By having a reliable synchronization it is possible to sleep between transceiving in order to save a substantial amount of power.

A chip antenna has been used instead of a trace antenna to eliminate the need to experiment on the optimal length and geometry of the trace. In addition, the chip antenna takes up a much smaller area than the trace antenna would have. A matching network of inductive coils and capacitors are still needed in order to match the output stage of the wireless transceiver to the chip antenna.

A SPI interface will be used between the module MCU and the wireless module. The SPI interface will provide a larger throughput than the I2C

bus, and is necessary in order to take full advantage of the 2Mbps transfer rate of the transceiver. Some extra connectors for reset and programming lines are added as well. These will be helpful during the development stage.

5.2 Firmware Design

Some firmware work is needed on the wireless module too. First of all a decent protocol must be found that will accommodate the various demands on throughput, latency and power consumption. A complicating piece of the puzzle is that the wireless network should be self-reliant and not depending on any form of master node. Some suggestive ideas for the protocol is given in section 5.2.1.

The firmware must also keep track of incoming and outgoing data, and where they come from and where they should be sent. Because of the limited amount of memory in the rf transceiver IC, the buffer will be too small to hold a signficant amount of packages. Therefore it will be easiest to limit the buffer to only be able to hold one incoming and one outgoing package at any given time.

The actual wireless packages from nRF24LE1 are limited to only 32 bytes per wireless transmission. If any more data is needed it must be split into several transmissions. The protocol will be able to handle the transmission of the entire buffer automatically, but if even larger amount of data needs to be transmitted, such as an image, it must be handled by the transmitting and receiving module MCUs.

5.2.1 Protocol

Addressed Transmission

The nRF24LE1 has the possibility to filter and buffer wireless packages based on a separate address. If wireless module A is the receiver, it can configure six individual addresses used for six independent links. Then wireless module B has been given one of those six addresses to use when transmitting to wireless module A. When wireless module B transmits a package with this

address, only wireless module A will receive this package, and will know that the message came from wireless module B because of the address that was used. Table 5.1 shows an example address table for transmission between 3 wireless modules.

Table 5.1: Example address table for wireless transmission.

Transmitter	**Receiver**	**Address**
Module A	Module B	0
Module A	Module C	1
Module B	Module A	2
Module B	Module C	3
Module C	Module A	4
Module C	Module B	5

The disadvantage with this approach is that all modules must constantly be in receive mode to check for messages. This will lead to a high power consumption. In addition, if two wireless modules start transmitting at the same time both messages will be corrupted.

Separate Channels

A slightly different approach is to use a separate channel for each, instead of the addresses. This will eliminate the danger of collisions, but will still lead to a high power consumption since the modules have to listen all the time. Therefore it could be beneficial to combine the use of addresses and channels. By assigning a separate channel for all communications going to module A, B, C etc. and making the transmitter listen for an invitation before transmitting it is possible to put the devices into sleep mode occasionally. Table 5.2 shows a step by step instruction of how a transmission would occur.

Table 5.2: A transmission from Module A to Module B.

Step	Module A action	Module B action
1	Change to B channel	Sleeping
2	Waiting for invitation	Invite C to start transmission
3	Waiting for invitation	Receiving
4	Waiting for invitation	Invite A to start transmission
5	Transmit data	Receiving
6	Receive ack	Send ack

Time-Split Channel

Further optimization of the power consumption can be done by synchronizing the transmissions. If all modules are synchronized and knows at what point in time it is their turn to transmit it will be easier to plan ahead, and decide when a sleep mode can be entered. If a time-consuming transmission needs to take place it is still possible for the two modules to negotiate a channel change to be able to complete their transmission without interruption from other nodes. An example time division table is shown in table 5.3, and shows how the modules can alternate the control of the wireless channel. Note that only the receiving nodes need to be awake at each point in time.

Table 5.3: Dividing the channel controls in time for each link.

Time	Direction
0 ms	A to B
1 ms	A to C
2 ms	B to A
3 ms	B to C
4 ms	C to A
5 ms	C to B
6 ms	A to B
...	...

This method will not suffer from collisions and will provide the modules

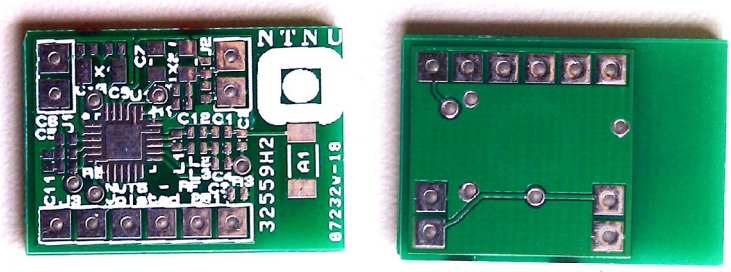

Figure 5.2: The wireless module circuit board. Top view on the left, and bottom view on the right.

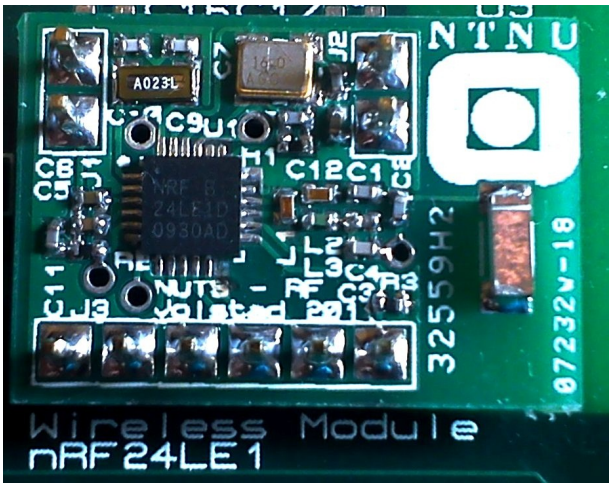

Figure 5.3: The wireless module.

a chance to sleep. However, the latency of a transmission will increase if the modules are allowed to sleep for too long. A good way to synchronize is also needed in order to make this work smoothly.

5.3 Prototyping

The hardware for the wireless module has been prototyped and tested, in addition to some protocol experimentation. The circuit board is shown in figure 5.2, and the completed wireless modules is shown in figure 5.3. Appendix C

shows the hardware schematics.

5.3.1 Testing

The wireless module was tested by programming it to constantly increment a variable, and transmit it to a receiving development board with a display. The counter was displayed on the display. Some testing has also been done on various ways to implement a communication protocol for the satellite.

5.3.2 Results

The wireless module is working as expected. Messages can be transmitted and received from the chip antenna. However, there is still some work left to do to find a suitable protocol for communicating efficiently between all of the modules. It could be easier to find a suitable protocol when the complete usage of the wireless link has been decided.

Power Consumption

The power consumption of the wireless module is ∼20mA during constant receive mode, and similar results for constant transmit mode. This is too much since there has to be one wireless module per backplane slot for a complete communication system. However, experimentation with different protocols has shown that the power consumption can probably reach as low as 2mA for each wireless module.

Chapter 6

Discussion

The satellite is suppose to be a self-reliant system. This means that it should be able to automatically handle any error conditions that might occur during it's lifespan. It is unfortunately impossible to predict every single problem that might occur. And not all of the ones that can be predicted will be possible to account for completely. A solution using redundant components will consume a lot of space and extra power, and the complexity of connecting these systems could lead to a more fragile system. Since NUTS will follow the CubeSat specification it has a heavily limited amount of space and weight. Because of this, the solutions presented in this thesis has emphasized simplicity over a fully redundant system.

The amount of space available for each module is quite limited. By doing a simple design for the backplane it was possible to put all the necessary control logic on it. This will ease the process for module designers, and make the backplane more reliable since all control logic is the same. One simple mistake that could have been done is to use a different logic family for the control logic. The logic families have been specially selected to account for voltage range, speed, partial power down and other aspects.

The backplane solution takes care of the most important design aspects, which has been to not allow a single fault to take down the whole satellite. This has been done by being able to isolate modules from the rest of the system, and power down modules drawing too much current. There is even a redundant set of regulated power lines in case one regulator fails.

Space radiation will eventually incapacitate the satellite due to increasing TID, and it is difficult to slow down the process. The satellite has mostly been protected from SEU to stop a sudden radiation burst from causing too much damage. The over-current protection will hopefully act quick enough to be able to stop a latch-up condition from damaging the die. And if the latch-up does not draw enough power to trigger the over-current protection, the OBC will notice the anomaly and do a power toggling. Latch-up can be eliminated by building the logic gates using separate transistors, but it would consume way too much area. SEU affecting memory cells will be detected by the ECC. This will protect against a lot of possible hazards.

The OBC will have the main responsibility as a master module on the backplane. It will keep polling the rest of the modules to make sure the system's health is good. In addition, a log of events and statuses will be kept in a NAND flash. The NAND flash is not concidered to be a very reliable memory, not even on the ground. Therefore much redundancy has been planned for the flash memory. By using ECC and storing data twice, data should be kept intact even if some bits are flipped by SEU. There is still the possibility of the NAND flash logic failing totally, but that risk is always present, no matter what memory type is used.

The backplane master responsibility is shared with the radio module, which can take control when the ground station sends a specific command. If the OBC suddenly starts misbehaving, the radio module can take over and try to restore the OBC. The radio can even hold a copy of the OBC firmware in order to be able to reprogram it. If the OBC can't be recovered, the radio module can shut off power to the OBC and continue acting as a master.

Power consumption for the backplane seems to be well below 100mW. From rough power calculations of the solar panel efficiency, the total power budget is about 4W. A consumption of 100mW is therefore a small piece of the total budget. For the OBC the power consumption will be dependent on the programming of the MCU, but it is probable that it's power consumption can end at 100mW also. The total power consumption would then be about 5% of the total power budget, which should not be too demanding.

Intra-satellite wireless communication will be an interesting addition to the regular wired bus. Since the total speed of the I2C bus has been limited to 400kb/s, the 2Mbit/s wireless link can be used to speed up transmissions of larger data sets. It could even prove to be more power efficient to

use this high speed wireless link over the wired I2C bus. Since the I2C bus signals will have to propagate to several separate sub-domains with separate pull-ups. And transmissions will require longer time to complete on the I2C bus, so power consumption could add up to be higher when using the wired bus. In addition, the point to point transmission using the wireless bus will not stop any other modules from communicating.

Chapter 7

Concluding Remarks

The overall system solution presented in this thesis seems to be a viable design for a reliable satellite. Although, radiation testing has not been done because of lack of equipment, the functionality of isolating separate modules and others have been verified. The design has emphasized simplicity over a complex redundant system, as this has several benefits such as lower area and power consumption, and since simple systems generally are more reliable.

Low power consumption has been a constant criteria for the backplane OBC and wireless communication module. With the OBC and backplane consuming 5% of the total power budget, there will be plenty of power left for other modules to perform their tasks. The wireless communication could even prove to improve on the overall power budget since transmissions can be done quicker, and devices can go into sleep modes faster.

The abrasions from radiation can only be countered by shielding or using different manufacturing processes for the ICs. These ICs are hard to find, and it is almost impossible to shield from the most energetic particles. Therefore eventual failure will occur. However, the most drastic effects of space radiation, such as latch-up and bit-flips, have been taken into account and compensated for in the design. The total lifespan of the satellite will be increased by the proposed solutions.

Chapter 8

Future Work

8.1 Wireless link

The exact usage of the wireless link must be assessed, and a suitable protocol must be implemented. For now the wireless link is considered an experiment to look at the viability of an internal wireless bus for communications.

8.2 OBC Completion

There are still a lot of work remaining on the OBC module. The behavior of the firmware has only been outlined, and some drivers developed. Still the implementation needs to be done and thoroughly tested. A file system should be implemented on the NAND flash device for logging, and other usage. The OTP memory also need some sort of structure.

8.3 Radiation Detector

There should also be done some research on adding a radiation detector circuit to the system. A photodiode can be used to detect charge injections due to ionizing radiation. This actually works much the same way as the

reason for latch-up and bit-flips. A charge is injected on the die, and flows as a current through the sensor. This minute current can be amplified and measured to attain an approximation of the amount of radiation.

The measurements can be logged and downloaded to the ground station. It could be interesting to see the changes in radiation at various locations in the Van Allen belt, and also compared to the solar cycle. In addition, the measurements could be used by the OBC to determine if certain fragile systems should be shut down during intense radiation showers.

Bibliography

[1] "Space radiation effects on electronic components in low-earth orbit," tech. rep., NASA, April 1996. Practice No. PD-ED-1258.

[2] R. A. Mewaldt, "Cosmic rays," 1996.

[3] "Van allen radiation belt." http://www.britannica.com/EBchecked/topic/622563/Van-Allen-radiation-belt.

[4] J. P. Uyemura, *Introduction to VLSI Circuits and Systems*. John Wiley & Sons, Inc., 2002.

[5] "Single event latchup." http://www.aero.org/capabilities/seet/SElatchup.html.

[6] S. Kayali, "Space radiation effects on microelectronics." http://parts.jpl.nasa.gov/docs/Radcrs_Final.pdf.

[7] "Understanding latch-up in advanced cmos logic." http://www.fairchildsemi.com/an/AN/AN-600.pdf, April 1999. AN-600.

[8] S. Chen, "What types of ecc should be used on flash memory?," tech. rep., Spansion, November 2007.

[9] N. Marquardt, "Introduction to the principles of vacuum physics." http://www.cientificosaficionados.com/libros/CERN/vacio1-CERN.pdf.

[10] D. Wright, "Space debris." http://www.ucsusa.org/assets/documents/nwgs/wright-space-debris-physics-today.pdf, October 2007.

[11] T. Miyahira and G. Swift, "Evaluation of radiation effects in flash memories." http://klabs.org/richcontent/MAPLDCon98/Papers/c4_miyahira.doc.

[12] A. S. Sedra and K. C. Smith, *Microelectronic Circuits*. Oxford University Press, 2004.

[13] "Evaluating embedded non-volatile memory for 65nm and beyond." http://www.sidense.com/images/stories/designcon_8_a_eval_embedded_nvm_65nm_and_beyond.pdf, 2008.

[14] H. S. R. S. V. L. G. L. Hans Reisinger, Martin Franosch and H. Wendt, "Patent no. us 6,215,140 b1 - electrically programmable non-volatile memory cell configuration," tech. rep., Siemens Aktiengesellschaft, April 2001.

[15] J. Benedetto and C. Hafer, "Ionizing radiation response of an amorphous silicon based antifuse," in *Radiation Effects Data Workshop, 1997 IEEE*, pp. 101 –104, jul 1997.

[16] "Logic guide." http://focus.ti.com/lit/sg/sdyu001z/sdyu001z.pdf, 2009.

[17] "At32uc3a3 datasheet," March 2010.

[18] "nrf24le1 - ultra-low power wireless system on-chip solution," August 2010.

[19] I. Helland, "Systems platform for nano and pico satellites," January 2009.

[20] "Um10204 - i2c-bus specification and user manual - rev.03," tech. rep., NXP, June 2007.

[21] G. S. L.Z. Scheick and S. Guertin, "Seu evaluation of sram memories for space applications," 2001.

Appendix A

Backplane

The source drawings for the hardware of the design is located in this section.

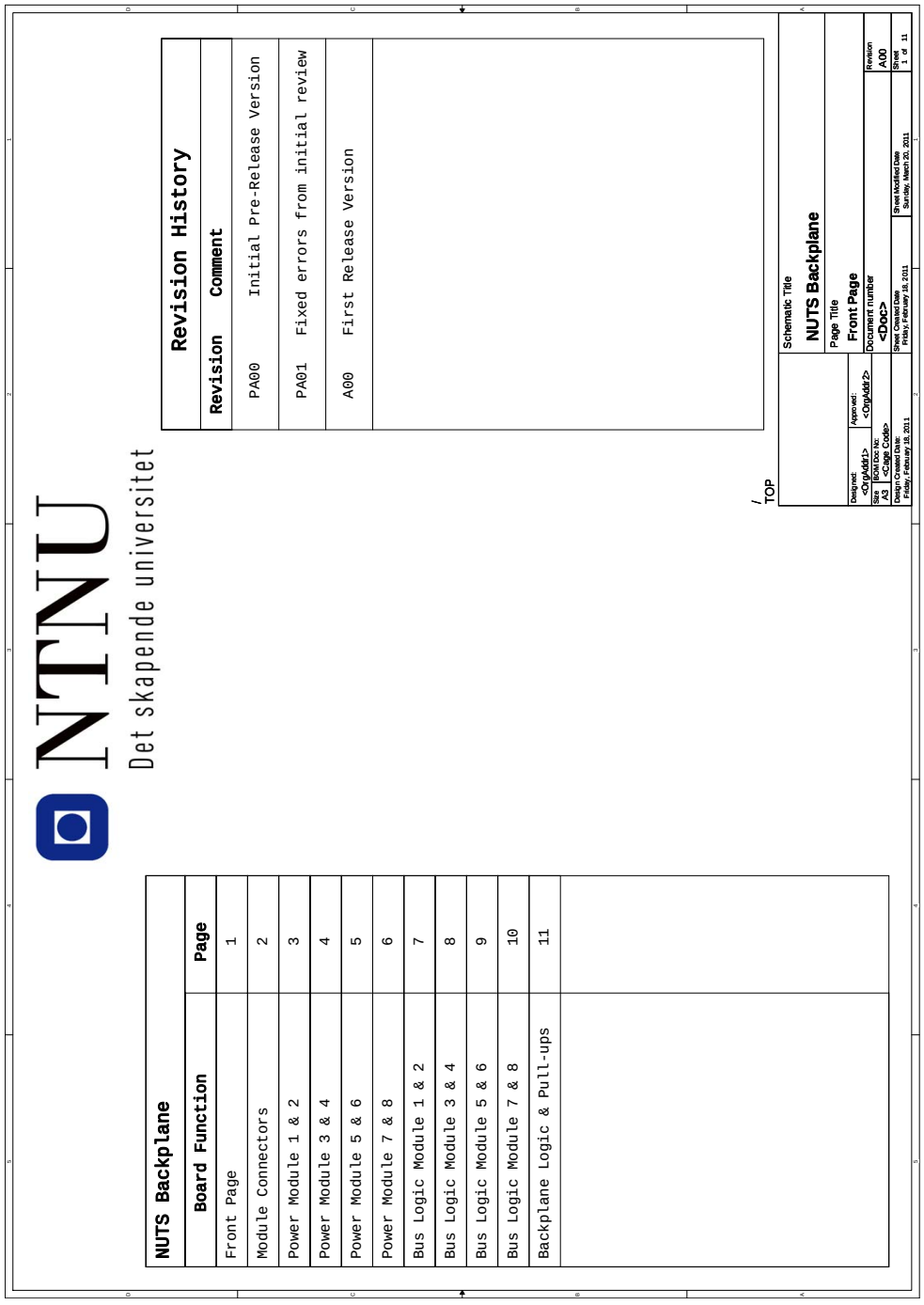

Figure A.1: Backplane schematic, page 1.

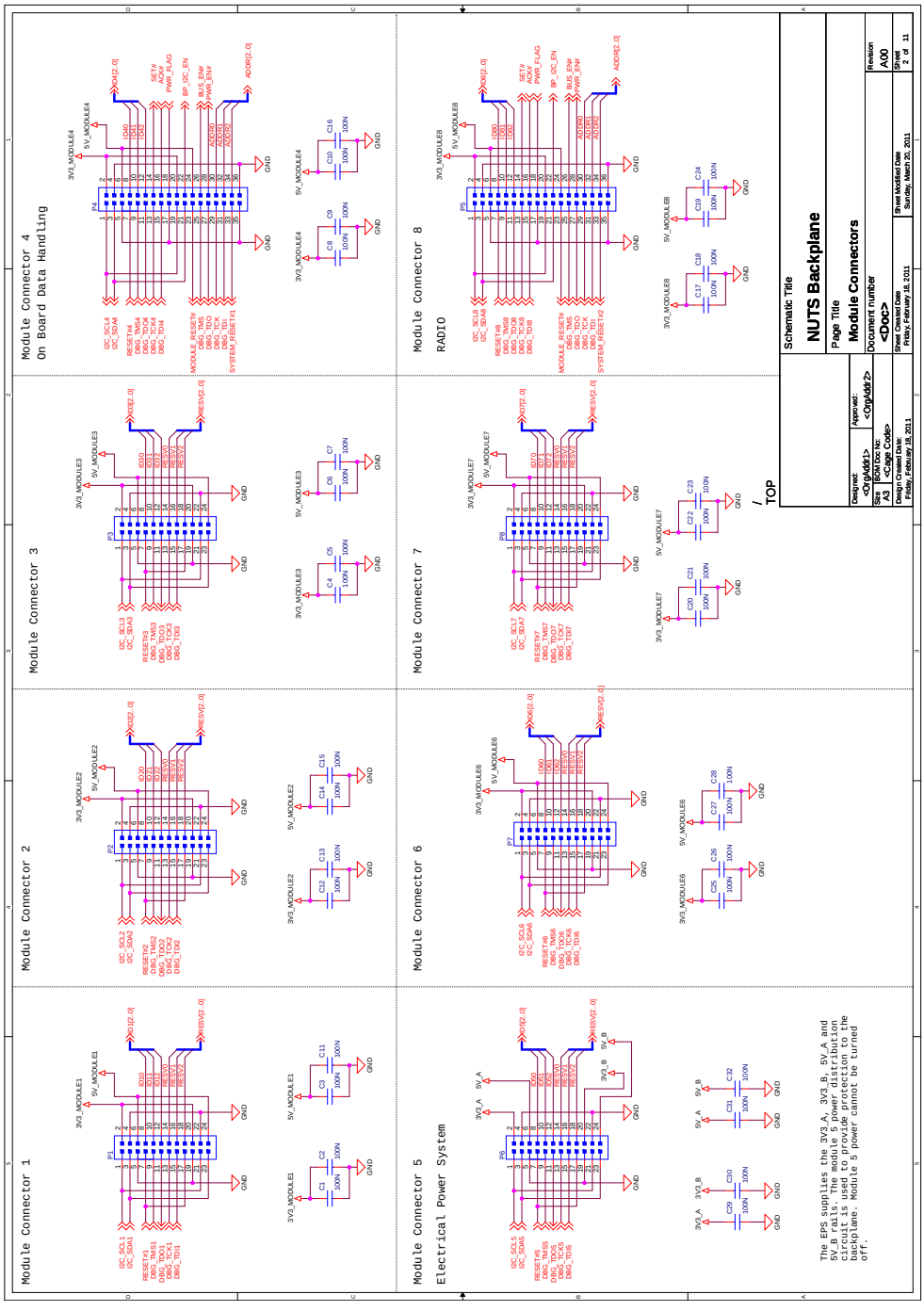

Figure A.2: Backplane schematic, page 2.

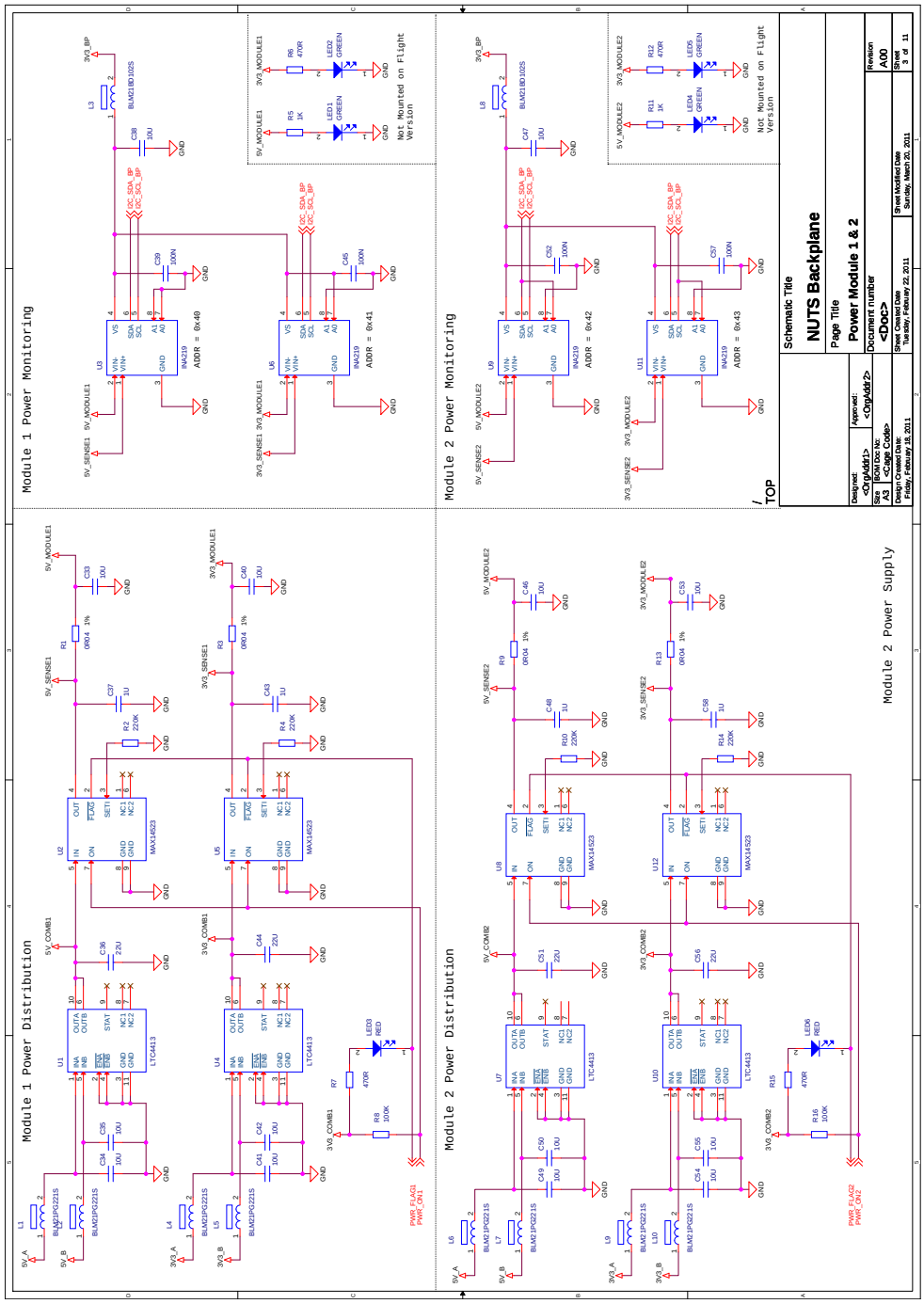

Figure A.3: Backplane schematic, page 3.

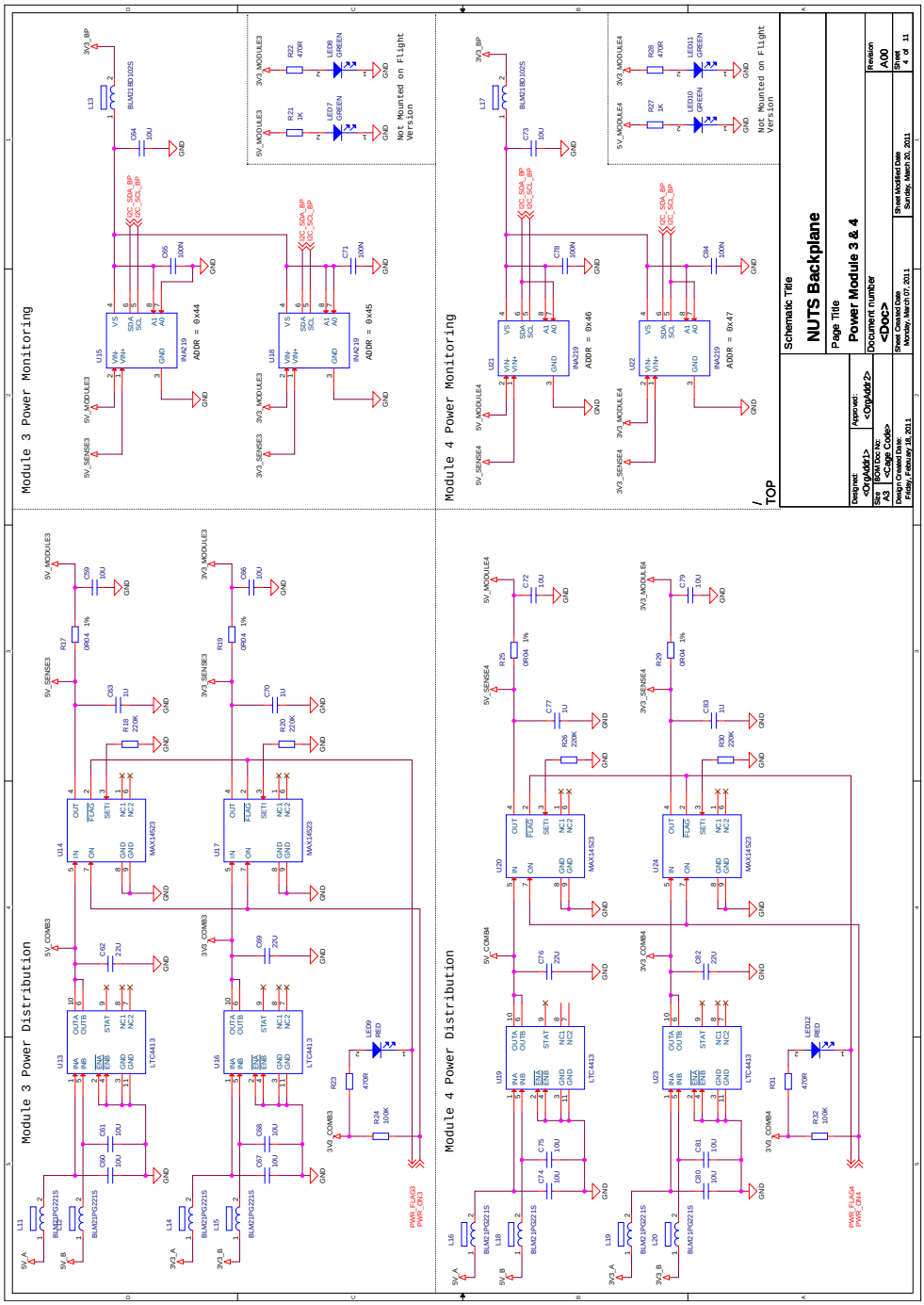

Figure A.4: Backplane schematic, page 4.

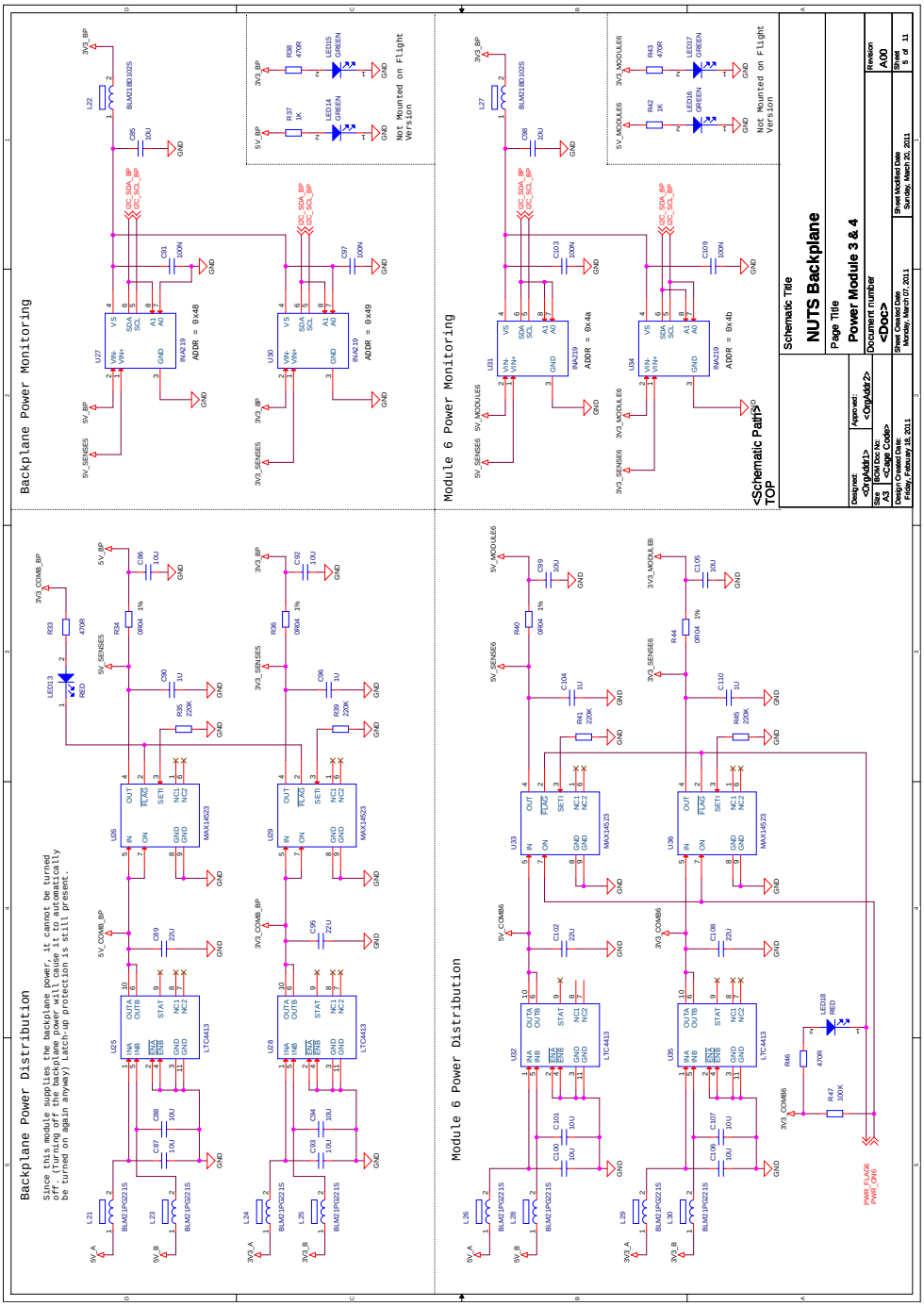

Figure A.5: Backplane schematic, page 5.

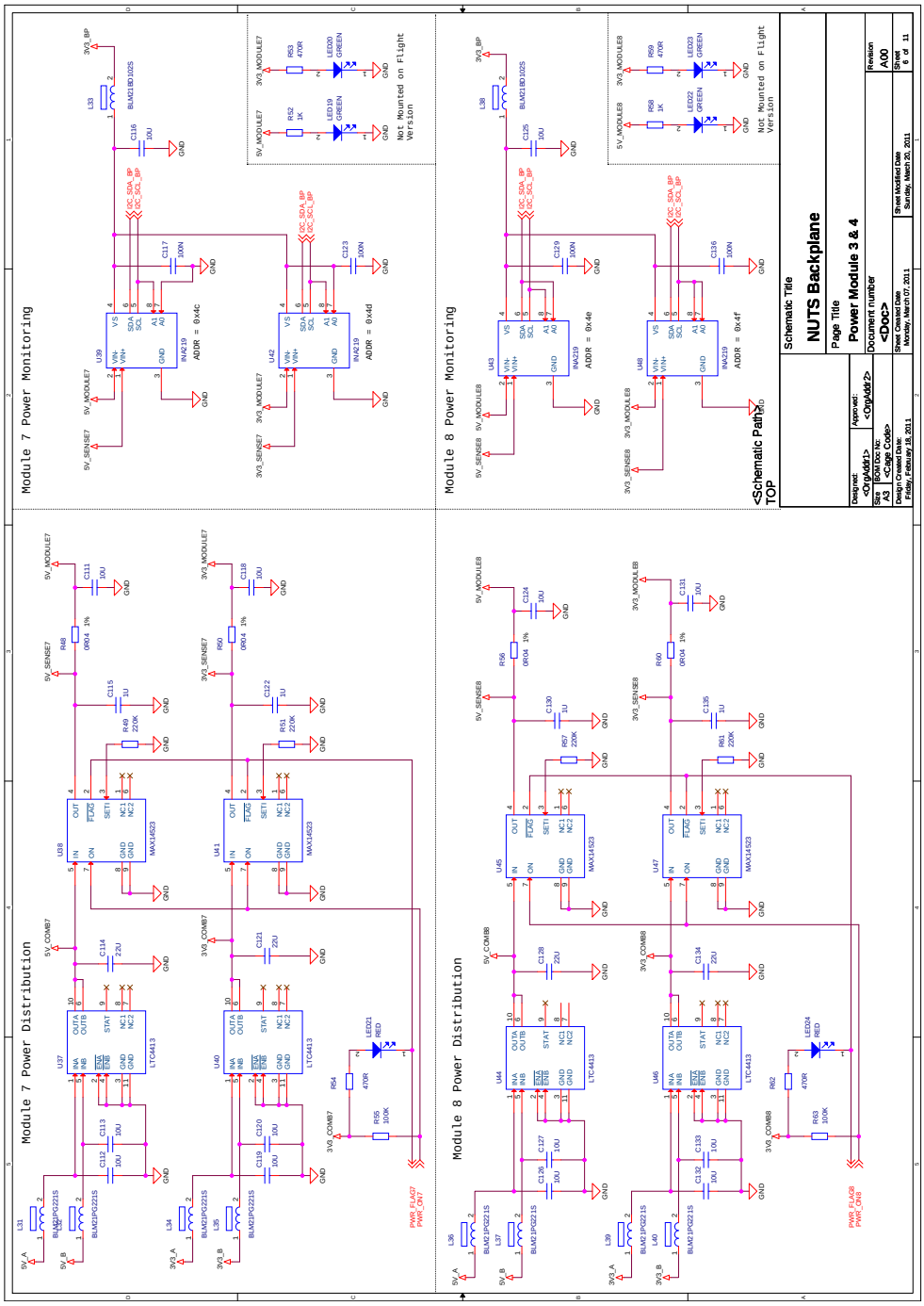

Figure A.6: Backplane schematic, page 6.

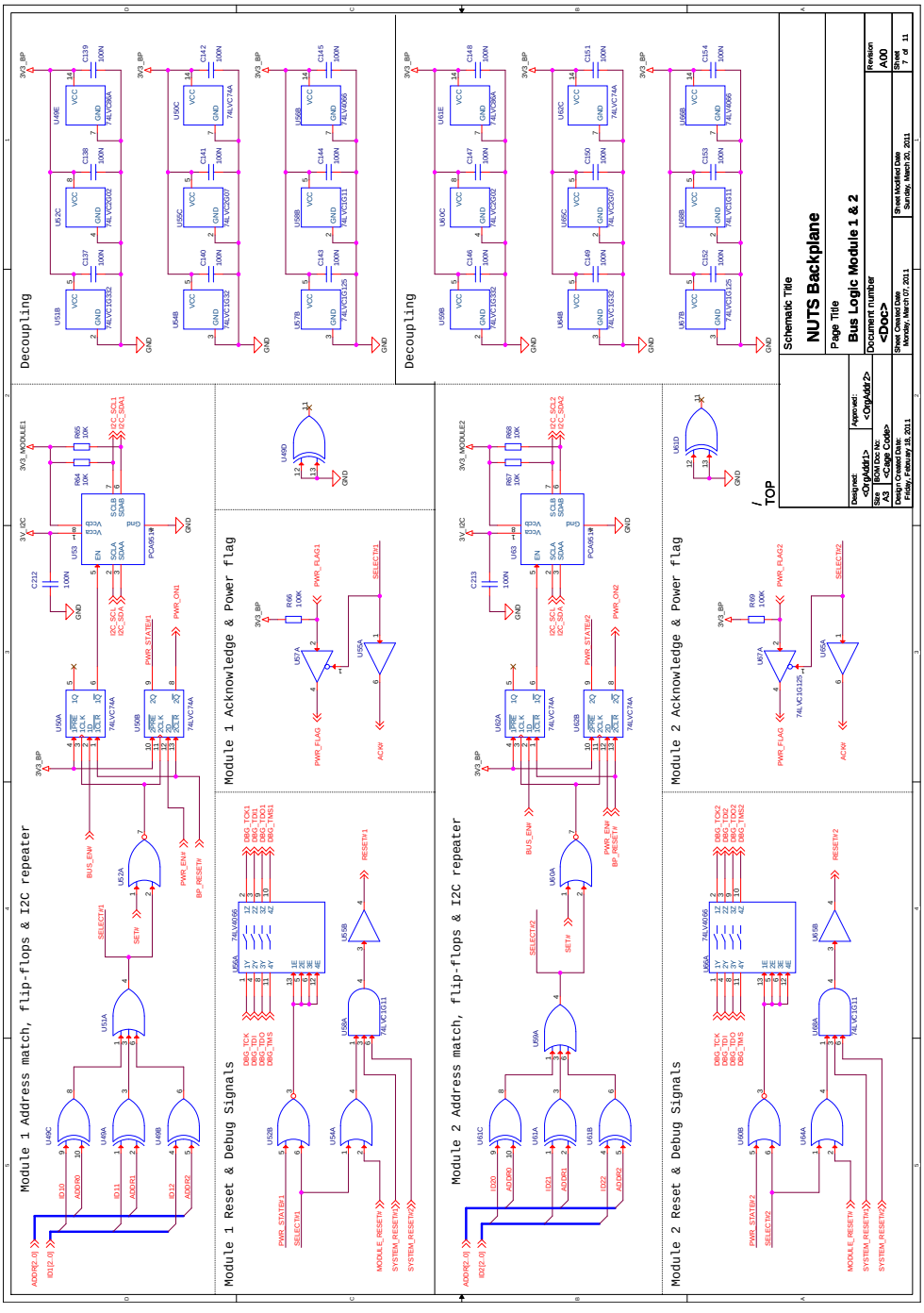

Figure A.7: Backplane schematic, page 7.

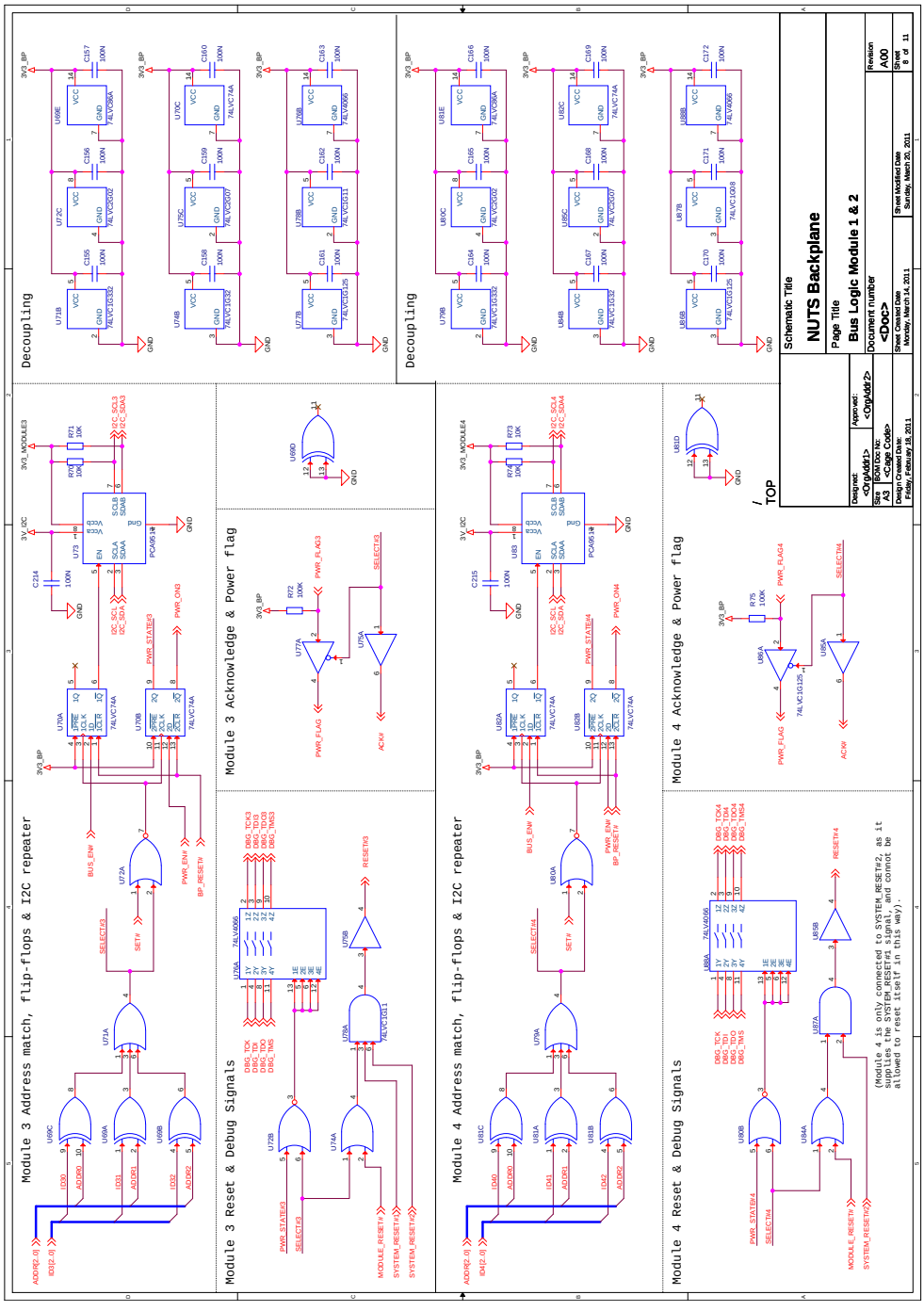

Figure A.8: Backplane schematic, page 8.

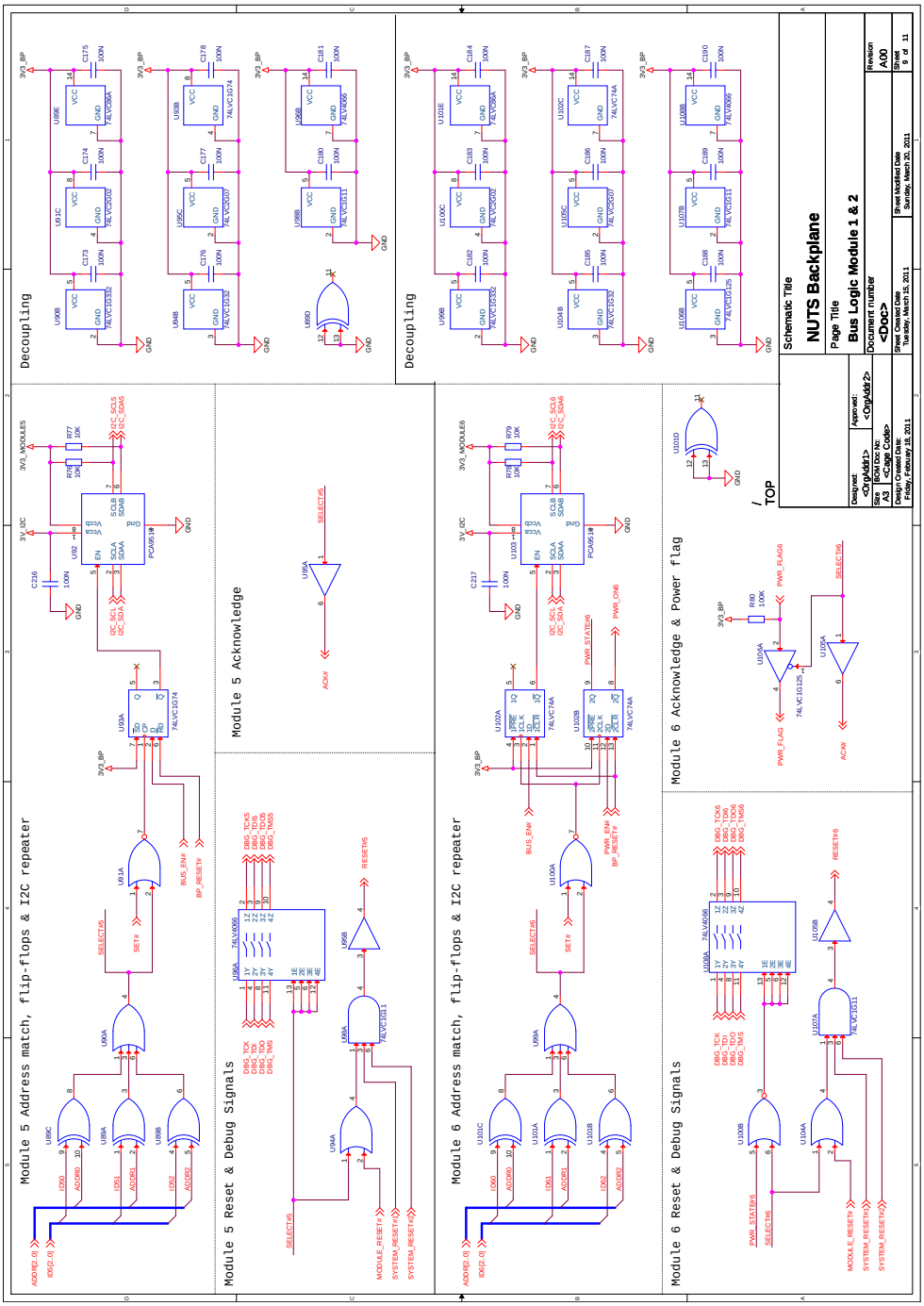

Figure A.9: Backplane schematic, page 9.

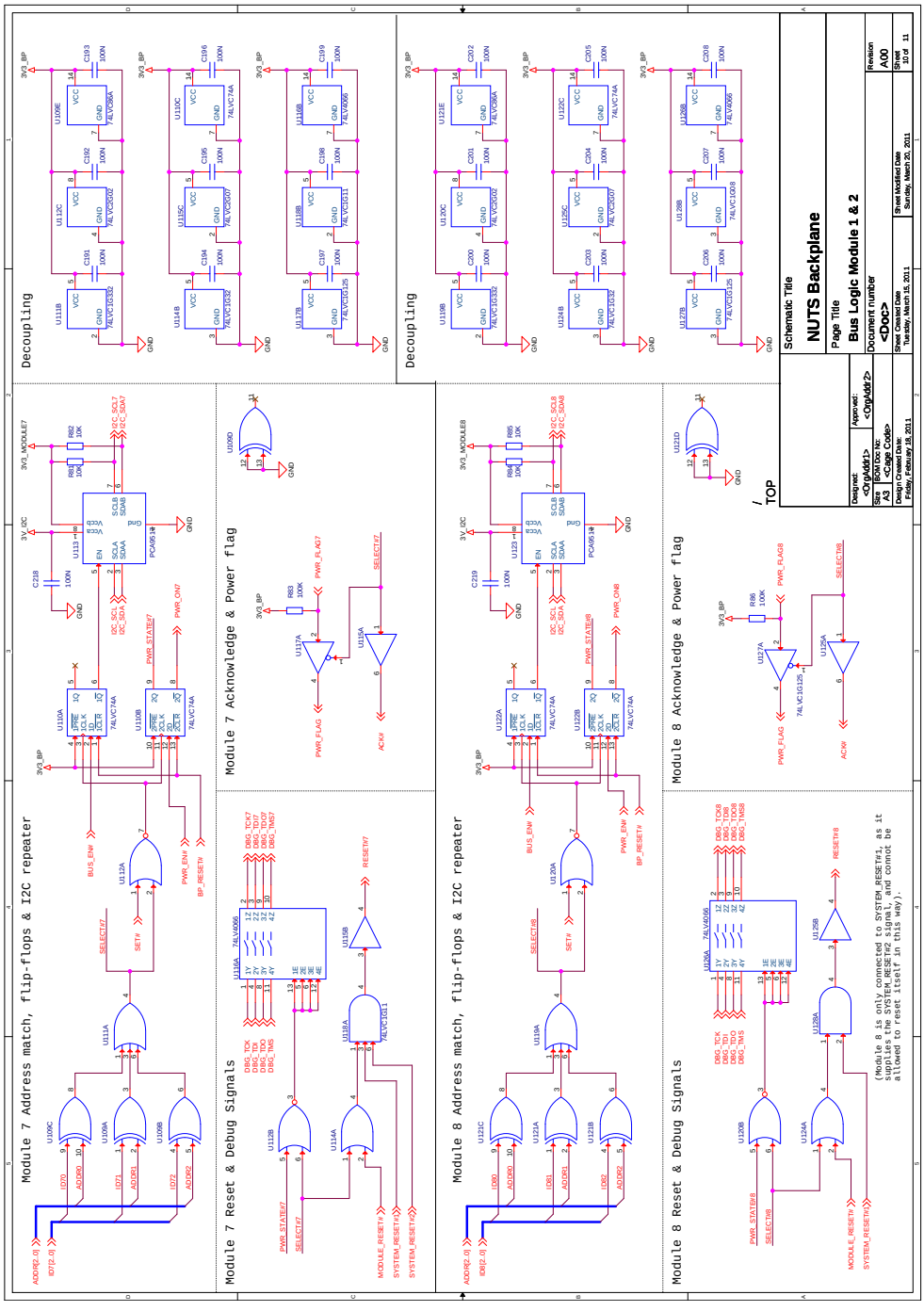

Figure A.10: Backplane schematic, page 10.

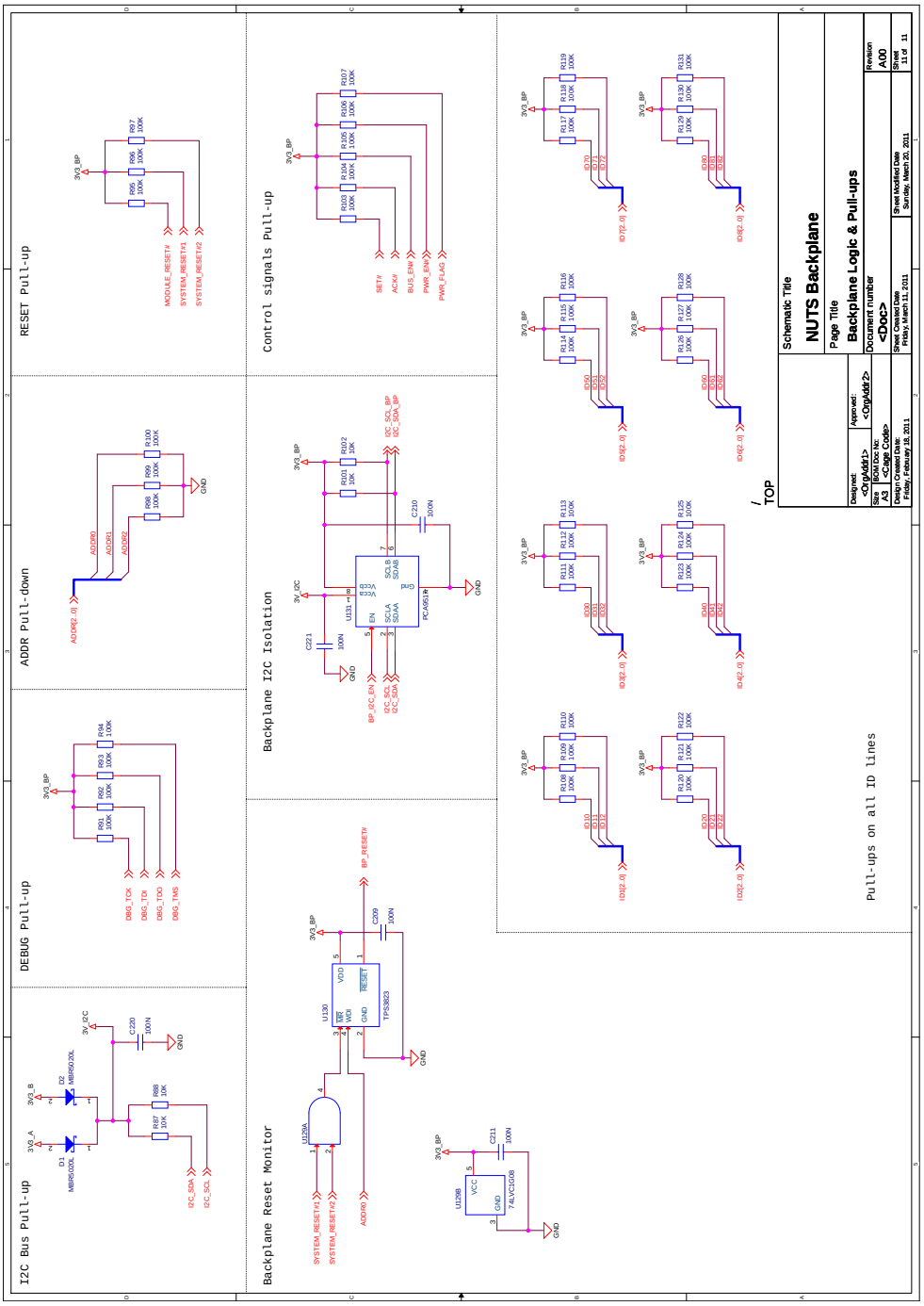

Figure A.11: Backplane schematic, page 11.

Appendix B

On-Board Controller

The source drawings for the hardware of the design is located in this section.

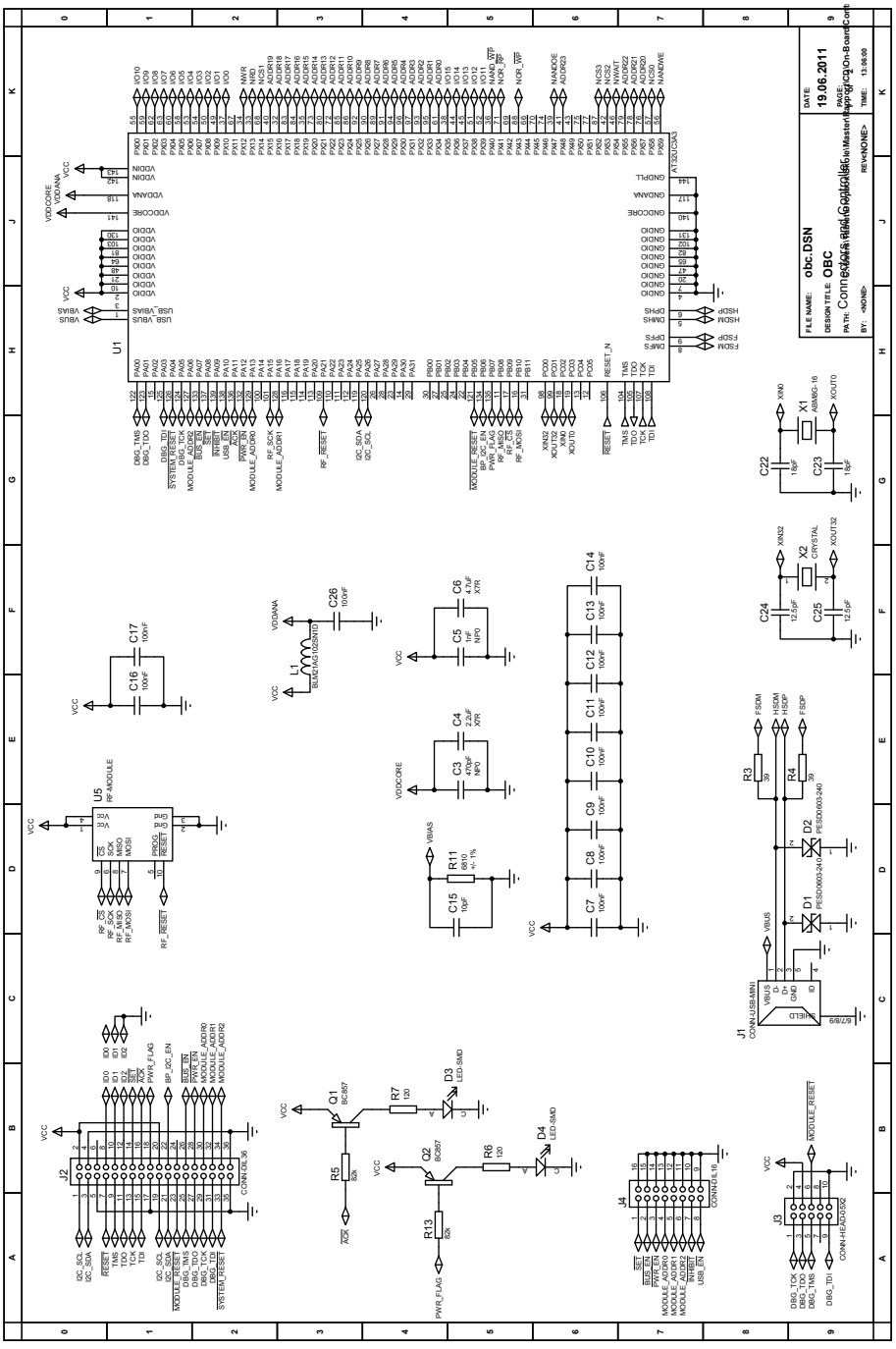

Figure B.1: On-Board Controller schematic, page 1.

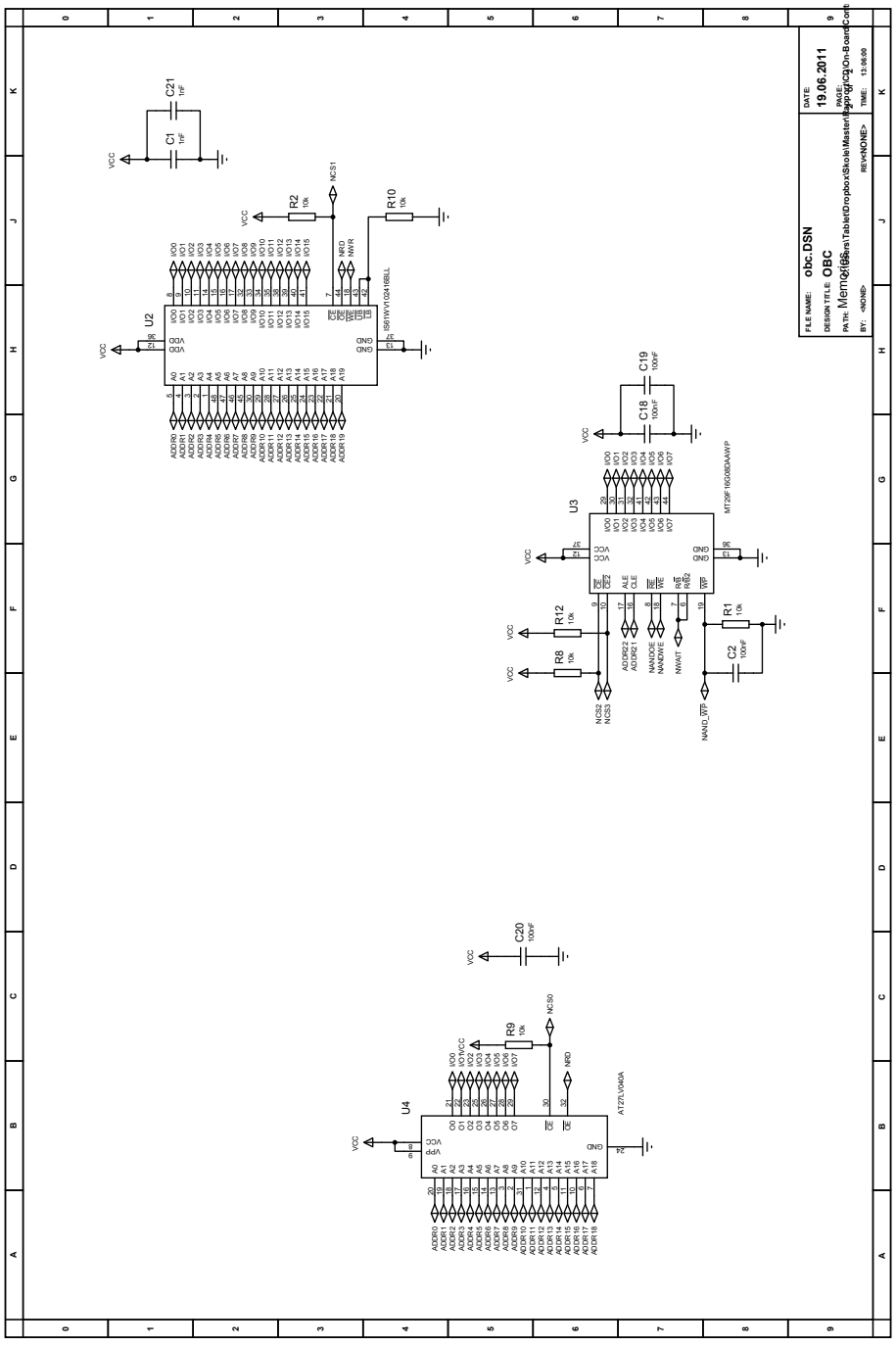

Figure B.2: On-Board Controller schematic, page 2.

Appendix C

Wireless Internal Communication Module

The source drawings for the hardware of the design is located in this section.

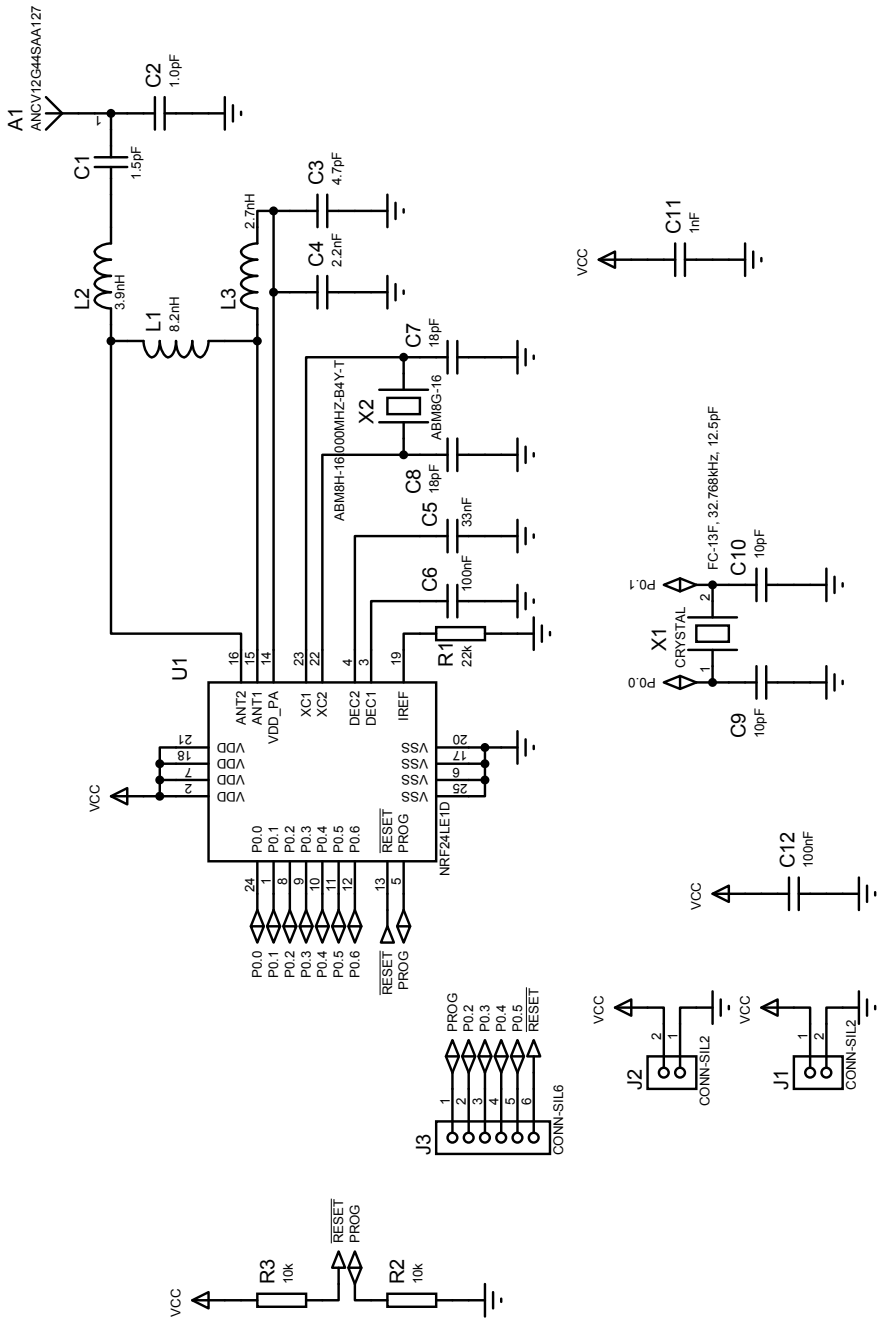

Figure C.1: Intra-satellite Communication Module schematic.