IAC-06-B5.7.05

CDHS DESIGN FOR A UNIVERSITY NANO-SATELLITE

Author
Gerard Aalbers
Computer Engineering, EEMCS, TU Delft, The Netherlands
g.t.aalbers@delfic3.nl

Co-Author(s)
Georgi N. Gaydadijev
Computer Engineering, EEMCS, TU Delft, The Netherlands
g.n.gaydadijev@ce.et.tudelft.nl

Rouzbeh Amini
System Integration, AE, TU Delft, The Netherlands
r.amini@lr.tudelft.nl

Abstract

The design starting points for the Delfi-C3 university nano-satellite are the absence of a battery, use of a commercial off the shelf structure and on-board computer, single failure survivability strategy and limited operational flexibility. Two architectures were considered during the design of the command and data handling subsystem: a star architecture, with the on-board computer in the center, and a distributed architecture, with the on-board computer as and microcontrollers for subsystem control. The microcontrollers are connected to the on-board computer by a serial databus based on the I2C standard. To improve its reliability, a simple error detection and correction approach based on parity bits will be used. The star architecture proved too cumbersome to work with and the distributed architecture was selected as the baseline design for Delfi-C3.

Full text

1 INTRODUCTION

The Delfi-C3 satellite is the first Dutch university nano-satellite and is currently under development at the Delft University of Technology in The Netherlands [1]. The project was initiated in November 2004 and is a cooperation between the faculties of Aerospace Engineering (AE) and Electrical Engineering, Mathematics and Computer Science (EEMCS). The project is primarily run by a mix of MSc. and BEng. students, although for some parts the help of industry or university supervisors is used. The Delfi-C3 satellite is based on the 3-unit cubesat package which has been bought from Pumpkin inc. This packages includes a structure of approximately 10x10x30 centimeters and an electronics board including an On-board Computer (OBC); this package can be considered Commercial Off The Shelf (COTS).

Delfi-C3 is currently under contract for a launch on the 30th of June 2007 from an Indian launchsite. This launch will place Delfi-C3 in a sun-synchronous low earth orbit at an altitude of approximately 600 km.

1.1 Delfi-C3 mission and payloads

The mission of the Delfi-C3 satellite mainly concerns its three payloads. These payloads are:

- Thin Film Solar Cells (TFSC), developed and

provided by the Dutch company Dutch Space. Primary payload and first flight opportunity for these cells.

- The Autonomous Wireless Sun Sensor (AWSS), a sun sensor with a wireless connection for data transfer to the Command and Data Handling Subsystem (CDHS) and its own power provision. This sun sensor is developed by the Dutch research institute TNO.
- A highly efficient on-chip radio transceiver, developed by the faculty of EEMCS of the TU Delft. Due to time constraints, Delfi-C3 will only fly its power-amplifier.

The mission objectives are separated into technical and educational objectives. The educational objectives are fairly straightforward, and come down to succeeding in actually building and launching the satellite with a team of students. Since all payloads are in-orbit demonstrations of new technologies, the technical objectives are mostly concerned with getting the data from these experiments to the ground. Of these, the primary technical objective of the Delfi-C3 mission is to get the experiment data of the TFSC payload to the ground. The fourth technical objective is to establish a network of ground stations, which helps to increase coverage for low earth-orbit missions.

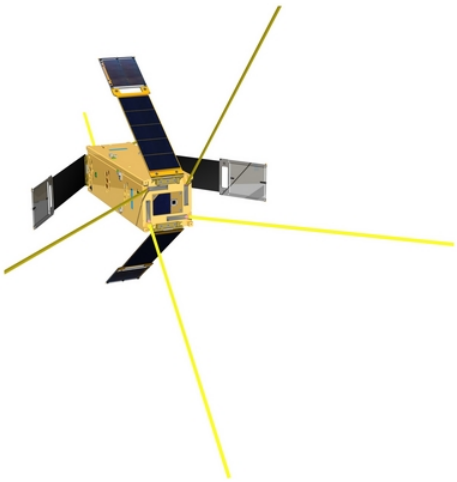

Figure 1: The Delfi-C3 satellite

1.2 The Command and Data Handling Subsystem

The Delfi-C3 Command and Data Handling Subsystem (or On Board Data Handling subsystem) can be characterized by the following functionality:

- Processing, storage, distribution and execution of received telecommands
- Acquisition, processing and storage of data presented by different subsystems and on-board experiments
- Control and synchronization of the measurements performed on the on-board experiments
- Decoding data received from and encoding data transmitted to the communications platform
- Control of satellite operation based on available data

All items mentioned above are standard functions for a CDHS to perform. The Delfi-C3 CDHS distinguishes itself from others not by functionality but by implementation, which will be discussed in the remainder of the paper. First we will introduce the starting points of the design of the CDHS. This leads to the next two sections in which we will discuss the two architectures that were considered for the organization of the CDHS. In these chapters we will discuss the basic concept, details and drawbacks of the architecture, all having the Delfi-C3 satellite in mind. Since one of these architectures uses a data bus, a section on the databus is included to discuss its details. Finally, there will be conclusions and recommendations.

2 DESIGN STARTING POINTS

In this section we will present the design starting points of the CDHS. These starting points are derived from higher level mission concepts or requirements and pose constraints on the design of the CDHS.

2.1 Commercial Off The Shelf flightboard

As mentioned in the introduction, the Delfi-C3 satellite is based on a COTS structure and OBC. The OBC resides on a flightboard, which can be seen in figure 2. This board provides the following:

- Remove Before Flight (RBF) pin
- USB type B connector (connected to the OBC)
- External power connector
- Mounting sockets for 2.4 GHz transceiver
- Texas Instruments MSP430F169 type microcontroller

- JTAG programming and testing connector
- Several crystal oscillators
- Connector for the CubesatKit bus

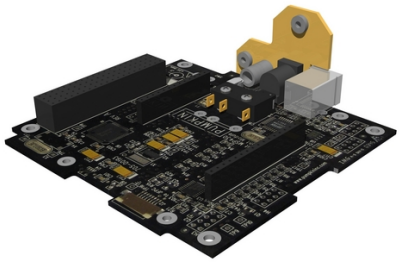

Figure 2: The flightboard including OBC

The use of this board, which was decided upon at the very start of the project to provide a head-start in the design, affects the satellite's and CDHS design in several ways. The flightboard is based on the PC-104 standard and is placed in a stack with all other Printed Circuit Boards (PCB). The design of the structure and flightboard force the other PCBs to be of the same size and take stacking connectors into account when laying out their components. The flightboard also provides several external connectors but these allow only limited freedom when a custom connector design is required. With the choice of the flightboard, the Texas Instruments microcontroller that is incorporated, also became part of the satellite's design.

2.2 Absence of power storage

One of the more interesting aspects of the Delfi-C3 project is the absence of a battery in the satellite. This was done for three reasons. The first is that all the experiments only produce useful data when the satellite is in sunlight, this all but eliminates the need for the satellite to function in eclipse. Second, including a battery and control circuitry would have a large impact on the already very limited available mass (up to 3 kilograms). Finally, the required range of operational temperatures for a battery is smaller than what the thermal control subsystem of Delfi-C3 can provide or is required by other components. Since the thermal control subsystem of Delfi-C3 is entirely passive, enhancing it with active components to accommodate a battery would even further increase the complexity accompanying a battery.

The absence of power storage means that there will be a loss of power during every orbit as soon as the satellite enters eclipse. For the CDHS this means that all data stored in volatile memory will be lost and that all electronic components, e.g. microcontrollers, will be reset. Since the exact time of eclipse is very hard to predict on-board the satellite, we can not safely assume when this reset will occur. On top of that, failure of one of the solar panels will also result in unexpected resets, reaffirming the ability to survive resets at any given point in time.

2.3 Redundancy

One of the underlying philosophies across the entire design of the satellite is to eliminate Single Point Failures (SPF). The elimination of an SPF is only applied when the elimination is fairly easy or absolutely necessary, because unlimited eliminations of SPFs will eventually lead to flying two identical satellites. Another aspect of this design aim is to look at the SPFs at satellite level; the satellite design must be able to cope with one failure, i.e. one failure will not affect the primary mission objective.

Since the data from the TFSC experiment is considered to be the most important, there would need to be some redundant solution for getting that data to the ground. The primary means is the CDHS that collects the data and then transmits it through the communications platform. In that case a failure of the OBC would be an SPF as no data would be processed. With the redundancy and SPF philosophy in mind, there always needs to be secondary means of getting the TFSC experiment data to the communications platform.

2.4 Operational flexibility

From a mission operations point of view, a certain amount of flexibility in the satellite control is desirable. This helps to solve problems or to deal with unexpected situations that occur during the mission. The degree of desired flexibility is dependent on the uncertainty in the mission environments and circumstances that is expected during the mission design. For low earth orbit missions like Delfi-C3, the uncertainty is low as this type of mission has been flown before and so the required flexibility is low. Since the CDHS is in control of the satellite's subsystems, the flexibility is usually applied there.

A high degree of flexibility in the CDHS would essentially mean the ability to update all of the CDHS software on orbit. This would require a dedicated controller that would be in charge of the communications platform through which it would receive the software updates. This dedicated controller ensures the capability to communicate with the satellite, even while reprogramming the CDHS, and it is usually implemented as a non-erasable component. In this case, however, the required degree of flexibility is not that high and because of the COTS flightboard and OBC the freedom to implement such a dedicated controller is very limited.

The limited flexibility and required and the complexity of in-orbit software updates led to the decision to limit the flexibility of the CDHS to changing operational parameters. These parameters are embedded in the software at the time of programming but they can be changed during the mission by telecommands. The parameters can be changed temporarily, i.e. until the next eclipse, or permanently, i.e. until they are updated again or reset to their original value. Changing a parameter temporarily simply overwrites its current value in program execution memory. Changing a parameter permanently means that the value needs to be written in a non-volatile memory in the CDHS, in addition to overwriting the current value. The non-volatile memory thus contains a list of updated values for certain operational parameters that will be loaded when the CDHS is activated.

3 STAR ARCHITECTURE

In this section we present the first of two architectures, the star architecture. This architecture was the original baseline design for the Delfi-C3 satellite. Although the star architecture is well-known, we will still discuss some of its details with respect to Delfi-C3.

3.1 Basic concept

The star architecture concept places the OBC in the center of the satellite and all the subsystems are directly connected to it. All of the CDHS functions are performed by the OBC; it is in control of all the subsystem, performing all data transfers with them and is in charge of all communications with the ground. The connections through the satellite will run over the CubesatKit bus that is supported by the flightboard. A visual representation of the star architecture as it would be implemented in Delfi-C3 can be seen in figure 3.

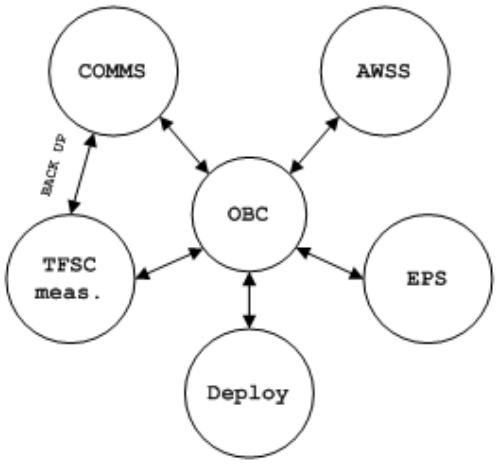

Figure 3: Delfi-C3 star architecture

3.2 Details

3.2.1 Boot sequence

As soon as the CDHS receives power, it will switch off all subsystems. This is done in order to prevent the subsystems from consuming all available power. This can occur when the satellite is just coming out of eclipse or when a subsystem has failed during the last orbit and started consuming all available power in the satellite. After the OBC has completed its own boot procedure it will start switching on and configuring subsystems sequentially. After each subsystem is configured, the OBC will write a status bit to the non-volatile memory. This will enable detection of a subsystem that for some reason resets or shuts down the satellite when it is receives power.

After this phase has been completed, nominal operation of the satellite will commence.

3.2.2 Handling of telecommands

All telecommands received by the communications platform will be passed to the OBC without any local processing. The OBC will verify and process the received command to prepare it for execution. Verification will be done by means of forward error correction and a process involving echoing the command to the ground. The OBC is also in charge of executing all of the commands because it controls all subsystems.

3.2.3 Measurement synchronization

One of the key aspects of all the experiment data is that they can be related in the time domain. The data from the TFSC experiment, for example, will

be related to the data from the AWSS experiment in order to get a better idea of the performance of the solar cells under different lighting conditions. One of the requirements for Delfi-C3 states these measurements have to be taken within one second of each other in order for them to be meaningful.

With the star architecture, the OBC is in charge of controlling all the measurements and acquiring the measurement data. This means that is relatively easy to fulfill this requirement by simply following the right sequence when acquiring the data and making sure that this happens in one uninterrupted chain(?).

3.2.4 Back-up mode

The center node is clearly the weak point of the star architecture. If this node fails, all subsystems will be completely isolated and no communication is possible between them. The other nodes, however, are isolated from each other and failure of one of these will not affect the others. Any failures along an interface between the central nodes and another node will also not affect the remaining nodes, assuming the central node can deal with this failure. The center node represents the OBC while the subsystems are the surrounding nodes.

To allow for the transmission of TFSC experiment data in case of OBC failure, there needs to be a secondary path from the TFSC measurement circuitry to the communications platform. To keep this as simple as possible, an analog transmission of the TFSC data to the communications platform was envisioned. The voltages are converted to frequencies before transmission to the communications platform, which in turn can transmit these to the ground.

In order to switch to this back-up mode, the OBC would need to be connected to watchdog circuitry that checks whether the OBC is still functioning properly. If the watchdog detects that this is not the case, it would reset the OBC . After several resets the watchdog would activate the back-up mode. In this back-up mode the OBC is powered down because it could interfere with subsystem operation in the back-up mode. In back-up mode, all subsystems will be switched to their default on or off modes by the Electrical Power Subsystem (EPS), depending on the required subsystems.

3.3 Drawbacks

The star architecture discussed in this section was the first CDHS design baseline for the Delfi-C3 satellite. As the design of the CDHS and other subsystems grew more mature, the team discovered that this particular architecture had several disadvantages.

The subsystems inside the satellite are placed on PCBs, which in turn are part of a stack. This means that the subsystems are independent in terms of hardware design and changes in one subsystem affect the other subsystems less. The star architecture of the CDHS, however, is highly dependent upon all of these subsystems and their changes. If the decision is made to add an additional measurement or control signal to a subsystem, the interface between the OBC and that subsystem will change. This means a change in hardware at both the subsystem and OBC, a change in software on the OBC and possibly an additional pin on the satellite bus.

Because all subsystems are directly connected to the OBC, the number of pins on it soon proved to be insufficient to accommodate all the connections. Especially the number of pins available for analog to digital conversion proved troublesome. Signal multiplexing and demultiplexing was applied for several signals but it was still questionable whether the number of connections would stay within limits. The number of connections to and from the subsystems also grew beyond the number of channels that could be accommodated on the satellite bus. Beside these physical problems, managing all these signals became increasingly complex.

The second issue that arose with respect to the satellite bus was electro magnetic interference. In the star architecture all measurement signals will have to run from their respective subsystem to the OBC. All of these signals will travel relatively long distances and in close proximity of other analog and digital signals with only a minimal amount of shielding. This would almost certainly lead to electro magnetic interference, especially with the analog signals, which in turn would affect the obtained precision for the measurements.

4 DISTRIBUTED ARCHITECTURE

In this section the second architecture concept is presented, the distributed architecture. This architecture is the current baseline design for the Delfi-C3 satellite.

4.1 Basic concept

In the distributed architecture, the OBC still fulfills a central role but is no longer directly connected to all the subsystems. Instead, each subsystem has been given a local microcontroller that handles all direct interaction with the subsystem. All microcontollers, including the OBC, are connected to a serial data bus. The data bus is based on the I2C [3] standard and is discussed in more detail in section 5. A visual representation of the distributed architecture as it would be implemented in the Delfi-C3 satellite can be seen in figure 4.

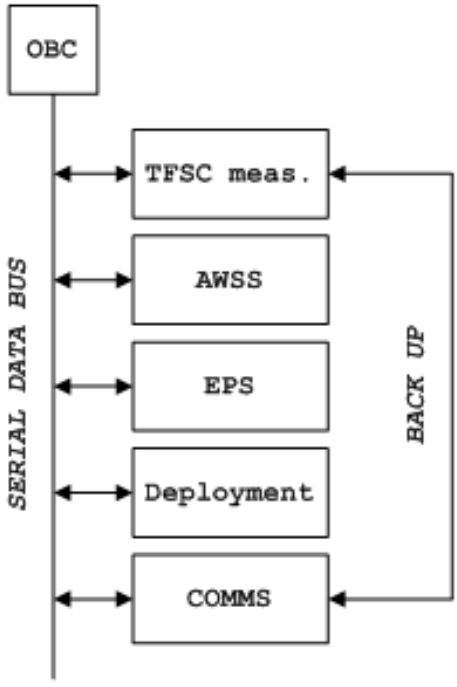

Figure 4: Delfi-C3 distributed architecture

4.2 Details

4.2.1 Boot sequence

As soon as the CDHS receives power, the local microcontrollers will keep the subsystems switched off, again to prevent the subsystems from potentially consuming all available power. Instead of directly proceeding to the sequential powering up of the subsystems, the local microcontrollers will first wait for a message from the OBC on the data bus. This message will indicate that the OBC and the databus are functioning properly. After the message has been received, the OBC will check if all microcontrollers are functioning and then start with the remainder of the boot sequence, which includes sending updated operational parameters to the microcontrollers if these are present in the On-board Database (OBDB). This remainder will essentially be the same as with the star architecture, except that the commands will be sent over the data bus to the local microcontrollers. If the mictrocontrollers have not received a message from the OBC after a certain amount of time, they will each activate their back-up mode. This back-up mode will be different for each microcontroller: some will power down their subsystem while others will perform either a reduced or nominal set of functions. A sequential powering up of the subsystems is possible in back-up mode, but will need to be based on timers instead of commands.

4.2.2 Handling of telecommands

Although this architecture is in essence a distributed system, some of its aspects do not adhere to this concept. The first, as mentioned in section 4.2.4, is the data transfer connection used during the back-up mode. The second is the transfer of telecommands from the communications platform to the OBC, which is also a direct connection.

A direct connection was chosen here to put the OBC in charge of verification and processing of the received commands, similar to the star architecture. The difference lies in the execution of those commands as this is done by both the OBC, a microcontroller or both, depending on the actual command.

4.2.3 Measurement synchronization

With the distributed architecture also comes a distributed acquisition of the experiment measurement data. Beside this acquisition the microcontrollers initiate and possibly control the measurements, but synchronization of these measurements is not as straightforward as with the star architecture.

In short, there needs to be a central mechanism to synchronize the measurements. With the star architecture this was the software in the OBC,

but this is not an option here. One possibility could be to have a central notion of time in the satellite, for example by having a clock line available in the satellite. This solution however is more complex than using the serial data bus to transmit a dedicated synchronization message. Upon reception of this message, the microcontrollers will initiate the measurements for their subsystem, thus making sure that measurements are related in the time domain. This message can also serve as the indication to the microcontrollers that the OBC and data bus are still functioning.

4.2.4 Back-up mode

The approach toward the back-up mode in the distributed architecture is somewhat different. First of all, since we now have microcontrollers available at both the measurement system and the communications platform, the back-up data transfer from one to the other can be slightly more complicated. We could still choose to implement the simple analog data transfer, but digital data transfers are also possible. No decision has been made on this subject yet but a secondary I2C bus, if supported by the microcontrollers, is one of the options.

While the microcontrollers are operating in the back-up mode, communication with the OBC could be restored. This could be the case if the OBC's watchdog triggers a reset after a software freeze has occurred. The OBC start to perform the normal boot sequence of the satellite by sending out a message on the data bus that it is functioning. Once the microcontrollers receive this message, they will stop operating in back-up mode and switch to nominal mode. If for some reason the OBC or databus fails after the boot procedure has been completed, the microcontrollers need to detect this and switch to back-up mode. This is achieved by using a simple timer that will trigger the switch to back-up mode when, after a certain amount of time, no message from the OBC has been received.

The OBC must remain active in the back-up mode. If the OBC would be deactivated when the satellite is in back-up mode, the possibility of the OBC recovering and being able to resume nominal operations is eliminated. Chances of the OBC unintentionally interfering with that mode are slim. This requires a meaningful data transfer on the data bus to which the microcontrollers would response.

4.2.5 Modularity

Placing a microcontroller at each subsystem or PCB has greatly improved modularity. Each subsystem is almost completely independent of other subsystems from a hardware point of view. The hardware interfaces to the OBC are the same for all subsystems and do not change when the subsystem hardware changes. The hardware interface to the local microcontroller does change, but this is only minor in comparison and only involves local changes in routing and occupied pins on the microcontroller. This also implies that subsystems can be in different stages of development, as is often the case in cubesat projects.

The interfaces between the subsystems and the OBC have been moved to the software layer of the CDHS. This has several advantages, beside the hardware modularity mentioned earlier:

Software simplicity The software that is needed to interact with the subsystems has become less complex, although the the CDHS software as a system becomes more complex. Dividing functionality is still advantageous however. First, the software in the microcontrollers will be smaller and less complex, enabling a better overview of its behaviour. Second, it enables enables sharing of code segments between the microcontrollers because some of their functionality is identical, e.g. communication with the data bus. Using the same code segments in multiple microcontrollers will make these segments less error-prone, because of a larger range of testcases. Finally, having multiple semi-independent, smaller, software packages instead of a single monolithic system, allows multiple people to work on the CDHS independently.

Interface flexibility Having the data interfaces to the subsystems in the software increases their flexibility, because changing in the later stages of the development process is easier than changing hardware.

Subsystem testability Each subsystem can now be tested completely independent of the other subsystems. Since each subsystem now has a standard hardware data interface and only responds to requests from the OBC, it is possible to develop a generic test platform. This test platform is currently under development and involves a computer connected to one or more subsystems by means of the serial data bus.

4.3 Drawbacks

The distributed architecture is the current baseline of the Delfi-C3 CDHS and was put in place after the star architecture's drawbacks were becoming too much of a problem. This architecture, however, is not entirely without complications itself and these will be discussed below.

The most obvious effect of introducing this architecture is that the number of components for the CDHS has significantly increased. More components mean more occupied surface area on the PCBs and in this case also an increase in power consumption. The increase in power consumption is the most critical because the available power in Delfi-C3 is, like with all cubesats, very limited. Several optimizations in the EPS and the ability to reduce the clock speeds of the microcontrollers were needed to reduce the power consumption to a more acceptable level.

The increase in microcontrollers also means an increase in software packages, as they relate one-to-one. A considerable amount of code segments can be shared, but each one will still need to have at least its own unique address for communicating on the data bus. This means that very careful attention has to be paid when putting these packages together. Mistakes in the databus addresses can lead to unreachable, thus uncontrollable parts of the satellite. In short, the management of the CDHS software layer becomes quite complex with the identical yet different parts.

5 DATA BUS

In this section we will discuss the databus that is used in the distributed architecture presented in the previous section. Since the distributed architecture is the current baseline of the Delfi-C3 CDHS, this databus is as well.

5.1 Bus functionality

In the distributed architecture the data bus is the backbone of the CDHS, allowing the OBC to communicate with each microcontroller. The only function of the bus it to get data from the sender to the receiver, preferably in a reliable fashion. The databus of Delfi-C3 is based on te I2C standard [3]. This means that on the physical level there will be at least two wires running through the satellite: one for the clock signal, one for the data signal. Each controller in the satellite will have a unique address but it is also possible to adress all controllers at once. A controller can be either a master or a slave on the bus, a master can initiate data transfers while a slave can only respond. To preserve central control and simplify the design of the entire CDHS, the OBC was designated as the only master.

5.2 Choice of bus type

With the introduction of the distributed architecture as the CDHS baseline design, the standard upon which the databus would be based was clear. The distributed architecture would still be using the OBC and that only has hardware support for the I2C and SPI standards. The same is true for the type of microcontrollers that were chosen for the subsytems, although there are also types available that support the CAN bus.

One of the features that is not present in the I2C standard is a means of error detection. Although each sent byte has to be acknowledged by the receiver, this does not guarantee an error-free transmission of that byte. Looking at serial bus standards that are used in space, in particular Spacewire [2] and CAN [4], we can see that these do provide error-detection on the transmissions. These two protocols are more extensive in general, as they also provide error-handling and restarting procedures in case such an error is detected. To compensate for this lack of error detection in the protocol it will be implemented in the CDHS software, creating a new protocol layer on top the original I2C protocol.

5.3 Error Detection and Correction strategy

In choosing a method for the error-detection functionality that will be implemented on top of the I2C protocol, two choices have to be made. The first is the choice of the error correction algorithm itself. Spacewire uses the odd-parity scheme while CAN employs a CRC-16 checksum. Parity is the simpler method of the two while CRC is the more powerful one. The use of an even more powerful Error Detection and Correction (EDAC) scheme for the data transfers on the Delfi-C3 databus is not required: the data transfers are only very limited in length which would mean that such a scheme introduces a significant overhead on the data transfers and the additional sensitivity detection would go unused. The second is the choice error handling approach. This can be either straightforward, i.e. simply retransmitting the entire message, or more complex, i.e.

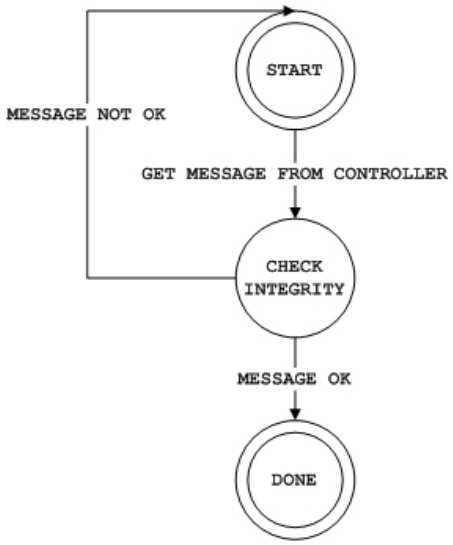

Figure 5: State diagram for master transmitter

make use of forward error correction or only resend the corrupted data instead of the entire message. The use of forward error correction is directly related to the selection of the error correction algorithm

The parity scheme has been selected as error-detection mechanism for the databus on Delfi-C3. This scheme requires the least computational power and because the microcontrollers will be running at low clock speeds to save power, this is a scarce resource. Each byte transferred on the databus, including the address, will be accompanied by a parity bit to check its integrity. The error handling was kept as simple as possible, a corrupted message will be retransmitted entirely.

5.4 Transmission scenarios

Transmissions on the I2C databus are always initiated by a master, i.e. the OBC, and never by a slave. This makes the handling of detected errors a bit more complicated. The following two transmission scenarios will occur.

1. **Master transmitter and slave receiver.** The OBC sends a message to a microcontroller. Upon reception of the message, the microcontroller will check the integrity of the content. Because it can not initiate a tranmission itself, it waits for the OBC to request information about the integrity of the message. This message is sent to the OBC without the use of the error-detection scheme and therefore needs to be insensitive to corruption.

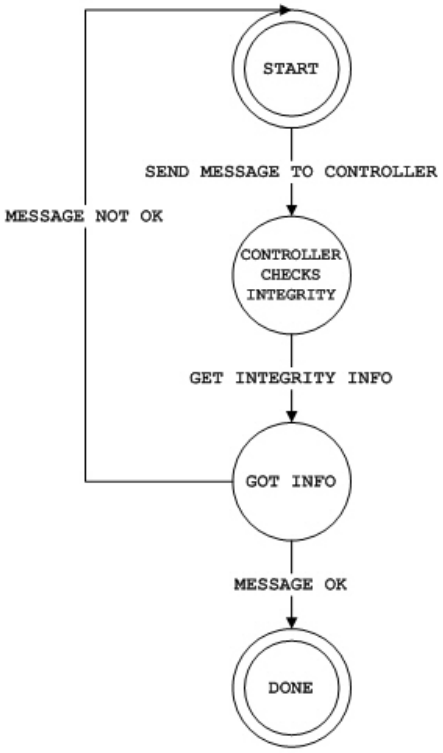

Figure 6: State diagram for master receiver

2. **Master receiver and slave transmitter.** A microcontroller sends a message to the OBC on request. Upon reception of the message by the OBC, the integrity is checked. If the message has been received uncorrupted the OBC sends a message to the microcontroller indicating that the content has been received uncorrupted and the transmission is complete. If there is a corruption of the message content, the OBC will restart the transmission process by requesting the data again.

5.5 Handling of internal commands

With the data bus, a notion of internal commands has been introduced to the satellite. These commands bare close resemblance to the telecommands that are sent from the ground. The telecommands sent to the satellite are always verified before execution and a similar procedure needs to be followed for the interal commands. If a command becomes corrupted during or after transmission to the destination or is incorrectly decoded there, resuming with execution could have catastrophic results. The telecommands are verified by means of forward error correction and echoing of the commands to the ground. Since there

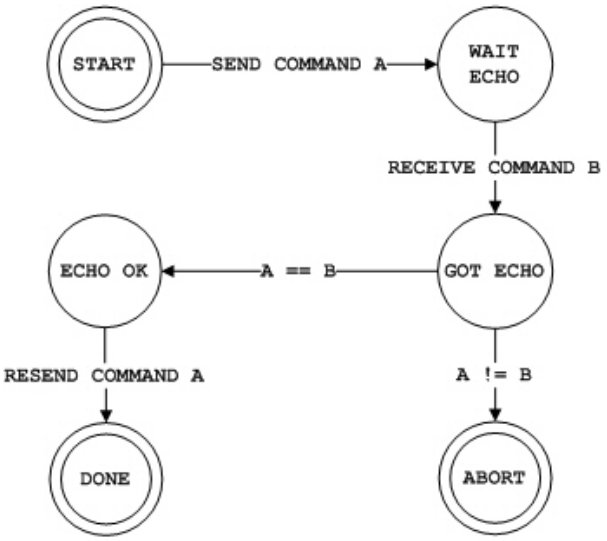

Figure 7: State diagram for internal command handling by the OBC

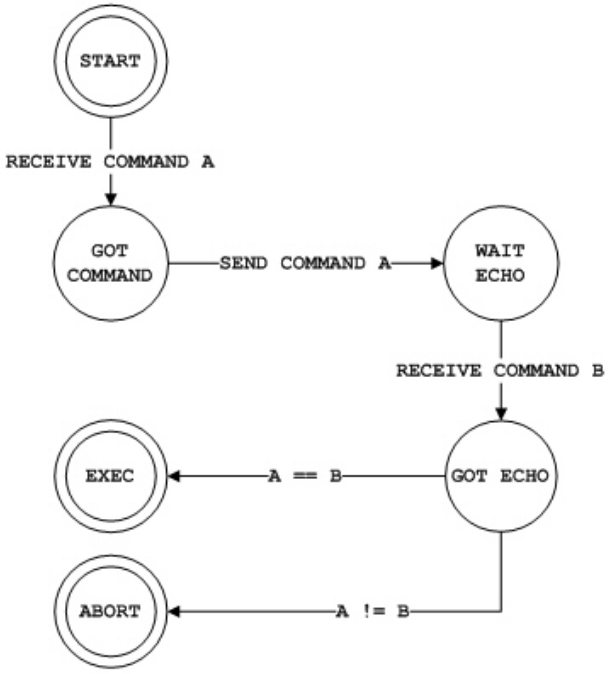

Figure 8: State diagram for internal command handling by a microcontroller

will already be an EDAC layer in place on top of the data bus, the internal commands will only be verified using an echoing procedure. The procedure is illustrated in figures 7 and 8.

First, a certain command is sent from the OBC to a microcontroller, which stores the command. This stored command is then sent back to the OBC where it is checked against the original command that was sent. If there these do not match, the OBC can restart the procedure by sending a new command. If the response is satisfactory the OBC will again send the command to the microcontroller. The microcontroller checks this command against its memory and starts execution if these two are identical.

This command handling procedure derives additional reliability due to the fact that:

- All messages are protected by the EDAC layer as the command handling process is placed on top of that layer
- Execution of a command is only possible when all copies of the command are identical in both memories and transmissions, limiting the sensitivity of this procedure for memory corruption in between transmissions
- Each message carries an indication of the message type, e.g. acknowledge, which helps to ensure the proper sequence of messages is observed

6 CONCLUSION

In this paper we have introduced the design starting points for the Command and Data Handling Subsystem design of the Delfi-C3 university nanosatellite. We have presented the two architectures that were considered during the design phase and discussed several resulting implementation concepts. We have also presented the databus used in the distributed architecture and have presented a simple protocol layer for the I2C protocol that will increase the reliability of the data transfers on this bus.

For Delfi-C3, a relatively large cubesat, the distributed architecture has been selected over a star architecture for the CDHS organization. With the number of subsystems present in the satellite, the star architecture simply became too cumbersome to work with. The distributed architecture maps elegantly onto the internal stack of subsystems,

improving modularity, testability and flexibility. For larger nano-satellites a distributed architecture with reliable internal communication is recommended, while for smaller nano-satellites the star architecture can still be considered.

REFERENCES

[1] Abe R. Bonnema et al., The delfi-c3 student nanosatellite, IAC-05.B5.6A.02.

[2] European Space Agency, Spacewire - links, nodes, routers, and networks, January 2003.

[3] Philips Semiconductors, The i2c bus specification, version 2.1, January 2000.

[4] Robert Bosch GmbH, Can specification, version 2.0, September 1991.

AWSS Autonomous Wireless Sun Sensor

AE Aerospace Engineering

CDHS Command and Data Handling Subsystem

COTS Commercial Off The Shelf

EDAC Error Detection and Correction

EEMCS Electrical Engineering, Mathematics and Computer Science

EPS Electrical Power Subsystem

MORON Malfunctioning OBC Recovery OptioN

OBC On-board Computer

OBDB On-board Database

OBDH On Board Data Handling

OBM On-board MORON Mode

PCB Printed Circuit Boards

RBF Remove Before Flight

SPF Single Point Failures

TFSC Thin Film Solar Cells

VCO Voltage Controlled Oscillator