Development of Electronics for Space Debris Detector SOLID

Morten Olsen
2013

Master of Science (120 credits)
Space Engineering - Space Master

Luleå University of Technology
Department of Computer Science, Electrical and Space Engineering
Development of Electronics for Space Debris Detector SOLID

Thesis

A thesis submitted in partial fulfilment of the requirements for the degree Master of Science with a Major in Space Technology

By

Morten Olsen

Division of Space Technology
Department of Computer Science, Electrical and Space Engineering

September 2013
Dipl.-Ing. Waldemar Bauer  
Supervisor  
Deutches Zentrum für Luft- und Raumfahrt

Prof. Peter Von Ballmoos  
Supervisor  
Université Toulouse III Paul Sabatier

Assistant Prof. Thomas Kuhn  
Examiner  
Luleå University of Technology
Abstract

Knowledge of small-particle distributions in space environment models, such as MASTER2009, is limited, due to difficulty in detecting these particles. To verify and improve data for Micro-Meteoroids and Orbital Debris (MMOD) models, Deutches Zentrum für Luft- und Raumfahrt (DLR) is developing the Solar Generator Based In-Orbit Space Debris Detector (SOLID) in-situ space debris detector which is expected to fly as payload in 2014 on the TechnoSat satellite built by Technisches Universität Berlin. This work focuses on the analysis and development of the embedded electronics for SOLID.

The expected space environment conditions, encountered during the mission, are analysed using radiation models (SHIELDOSE-2 and AP-8), MMOD model (MASTER2009) and potential EMC issues are investigated according to ECSS standards. The functional and technical requirements, for the space debris detector payload, are also established.

A previous SOLID prototype suffered from noise susceptibility. An active filter circuit is therefore designed to attenuate noise from the satellite Switch-Mode Power Supply (SMPS). A Short-Circuit-protection circuit is designed, to avoid excessive fault-currents and resulting performance loss in the satellite Electrical Power Subsystem (EPS). A new Multiplexer-circuit is developed, using a part with smaller PCB-footprint. All circuits are simulated with PSPICE, implemented as prototypes and their functionality successfully verified in lab tests, which also showed very good agreement with theoretical models.

A new C-program, for the SOLID Microcontroller Unit, is written, to control the detector circuits, improve software stability, and in the future, to handle the TechnoSat CAN-bus interface. No instabilities were observed in the new software. The implemented embedded electronics also met the strict power requirements.

For future work, it is recommended to begin qualification testing on component level, to improve the knowledge of the noise environment from the satellite EPS and to specify, in more detail, the system configuration including circuit board locations, mechanical-, electrical- and software-interfaces.
I would like to thank my supervisor, Waldemar Bauer, for the opportunity to conduct my thesis on this project under the System Analysis Space Segment (SARA) department at the Institute of Space Systems in Deutches Zentrum für Luft- und Raumfahrt (DLR), Bremen.

I would also like to acknowledge the European Space Agency Human Spaceflight and Operations directorate for granting me a scholarship to undertake this thesis work.

Throughout my two years in the SpaceMaster program, Dr. Victoria Barabash has always been a great support and mentor to me, which I am very grateful for.

A thank also goes to Vanessa Jane Clark and Carl Johann Treudler for their feedback on my thesis work and to Lars-Christian Hauer for his support and technical discussions in the Electronics-Lab at DLR.
## Contents

List of Figures vii  
List of Tables ix  
Acronyms x  
Definitions xiii

1 Introduction  
1.1 In-situ Detectors  
1.2 SOLID  
1.2.1 First Prototype  
1.2.2 Flight Opportunity  
1.3 Thesis Work Scope  
1.4 Document Structure  

2 Requirements  
2.1 Goals and Vision  
2.2 Functional Requirements  
2.3 Environmental Requirements  
2.3.1 Space Standards  
2.3.2 Radiation Effects  
2.3.3 Electromagnetic Compatibility  
2.3.4 Impact Rate  
2.3.5 Atomic Oxygen  
2.4 Technical Requirements  
2.5 Project Planning  
2.5.1 Work Breakdown Structure  
2.5.2 Project Time Schedule  
2.6 Verification Methods  
2.6.1 Model Philosophy  
2.6.2 EGSE  

iv
3 Design

3.1 Detector Board .................................................... 18
   3.1.1 Isolation Diodes ............................................. 19
   3.1.2 Multiplexers .................................................. 21
   3.1.3 Charging Effects ............................................ 22
   3.1.4 Transient Switching Time ................................... 22

3.2 Time-of-Impact Detector ......................................... 24
   3.2.1 Input Noise Characterization ................................. 26
   3.2.2 Input Filter ................................................... 30
   3.2.3 Short Circuit Protection ..................................... 40
   3.2.4 Sunlight Detector ............................................. 41

3.3 Microcontroller .................................................. 44
   3.3.1 Microcontroller vs. FPGA .................................... 44
   3.3.2 Microcontroller Requirements ................................. 45
   3.3.3 Prototype MCU Board ......................................... 46

3.4 Software .......................................................... 47
   3.4.1 Software Tools ................................................ 47
   3.4.2 Software Requirements ....................................... 49
   3.4.3 Software Structure and Implementation ...................... 50
   3.4.4 Software Outsourcing ......................................... 51

3.5 System Layout ..................................................... 54
   3.5.1 Internal and External Interfaces .............................. 57

3.6 Integration of Attitude Disturbance Data ......................... 59

4 Prototype Verification .............................................. 60
   4.1 Engineering Models and Functional Tests ...................... 60
      4.1.1 Input Filter of ToI Detector ............................... 60
      4.1.2 ToI Detector with MCU ................................... 63
      4.1.3 SC-protection Circuit ...................................... 65
      4.1.4 MUX Board with MCU ...................................... 69

4.2 Future Tests ....................................................... 72
      4.2.1 EMC-test with EPS from TU-Berlin .......................... 72
      4.2.2 CAN-bus on new MCU Board ................................ 73
      4.2.3 MCU Board with new Detector Board ...................... 74
      4.2.4 EGSE ......................................................... 74

5 Conclusion .......................................................... 76

6 Future Perspectives ................................................ 77

References ............................................................ 79

Appendices ............................................................ 83
# A Detector Line Impedance Calculator

# B Space Environment Simulations
- B.1 Total Ionization Dose (SHIELDOSE-2) ........................................ 87
- B.2 Single Event Upset Rates (AP-8) .................................................. 87
- B.3 Impact Rate (MASTER2009) ......................................................... 89
- B.4 Erosion by Atomic Oxygen (NRLMSISE-00) .................................. 91

# C Circuit Schematics and Simulations
- C.1 Short-circuit Protection Circuit with Sun Detector .......................... 92
- C.2 Input Filter ................................................................................. 95
- C.3 ToI Detector .............................................................................. 97

# D Prototype Hardware
- D.1 Input filter and ToI Detector ....................................................... 99
- D.2 MUX Board .............................................................................. 99
- D.3 MCU ......................................................................................... 103

# E Parts List

# F Software
- F.1 Source Code ................................................................. 107
  - F.1.1 main.c ................................................................. 107
  - F.1.2 SOLID.c ................................................................. 108
  - F.1.3 SOLID_adc.c ........................................................... 114
  - F.1.4 SOLID_gpio.c .......................................................... 118
  - F.1.5 SOLID_interrupts.c .................................................... 120
  - F.1.6 SOLID_pwr.c ............................................................ 123
  - F.1.7 SOLID_rcc.c ............................................................. 125
  - F.1.8 SOLID_rcc.c ............................................................. 126
  - F.1.9 SOLID_usart.c ........................................................... 129

# G Maple Source Code
List of Figures

1.1 Current and future prospected situation for the space debris environment . . . . 1
1.2 Illustration of the SOLID detector concepts . . . . . . . . . . . . . . . . . . . . . 3
1.3 Hardware of the first SOLID prototype . . . . . . . . . . . . . . . . . . . . . . . 4

2.1 SOLID Work Breakdown Structure . . . . . . . . . . . . . . . . . . . . . . . . . . 14
2.2 SOLID overall project time schedule . . . . . . . . . . . . . . . . . . . . . . . . . 15

3.1 Connection principle of detector lines and MUXs . . . . . . . . . . . . . . . . . . . 19
3.2 Bias conditions and return path for diode reverse currents on detector board . . 20
3.3 Limitations of MUX switching transient times due to parasitic capacitances . . 24
3.4 ToI pulse detector circuit of first SOLID prototype . . . . . . . . . . . . . . . . . 26
3.5 Characterization of input noise sources originating from the TechnoSat EPS . . 28
3.6 Fourier series of estimated input current ripple to the EPS . . . . . . . . . . . . . 29
3.7 Simplified block diagram of ToI detector filters . . . . . . . . . . . . . . . . . . . . 31
3.8 Bodeplot of input filter for the ToI detector . . . . . . . . . . . . . . . . . . . . . 36
3.9 Input filter transient responses on noise- and pulse inputs . . . . . . . . . . . . . 37
3.10 Schematic of ToI detector circuit . . . . . . . . . . . . . . . . . . . . . . . . . . . 38
3.11 Digital trimming of ToI detector threshold voltage . . . . . . . . . . . . . . . . . 40
3.12 EAGLE schematic of the SC protection circuit . . . . . . . . . . . . . . . . . . . . 42
3.13 Sun detector to detect shaded detector boards . . . . . . . . . . . . . . . . . . . . 43
3.14 STM32-Discovery development board . . . . . . . . . . . . . . . . . . . . . . . . . 47
3.15 Main loop software flowchart . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 51
3.16 Software flowchart of the interrupt functions . . . . . . . . . . . . . . . . . . . . 51
3.17 Software flowchart for the initialization function . . . . . . . . . . . . . . . . . . 52
3.18 Software flowchart for periodic scan- and ToI detection functions . . . . . . . . 52
3.19 Software flowchart of functions for reading/sending messages over UART or CAN- bus . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 53
3.20 SOLID system mounted inside satellite structure . . . . . . . . . . . . . . . . . . 55
3.21 Possible system layout configurations for SOLID . . . . . . . . . . . . . . . . . . 55
3.22 SOLID system blockdiagram . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 58
<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1</td>
<td>Test setup for measuring frequency response of ToI detector input filter</td>
<td>61</td>
</tr>
<tr>
<td>4.2</td>
<td>Lab test of ToI detector input filter</td>
<td>62</td>
</tr>
<tr>
<td>4.3</td>
<td>Testing of transient behaviour of ToI detector together with the MCU</td>
<td>64</td>
</tr>
<tr>
<td>4.4</td>
<td>Transient tests of ToI detector input filter</td>
<td>66</td>
</tr>
<tr>
<td>4.5</td>
<td>Test setup for the SC-protection circuit tests</td>
<td>68</td>
</tr>
<tr>
<td>4.6</td>
<td>Functionality of SC-protection circuit</td>
<td>70</td>
</tr>
<tr>
<td>4.7</td>
<td>Transient switch time, when reading out MUXs, before and after reducing the pull-down resistor value</td>
<td>71</td>
</tr>
<tr>
<td>4.8</td>
<td>Transient switch time, when reading out MUXs</td>
<td>72</td>
</tr>
<tr>
<td>4.9</td>
<td>Block diagram of test setup for EPS noise measurements</td>
<td>73</td>
</tr>
<tr>
<td>4.10</td>
<td>Block diagram of suggested CAN-bus test setup</td>
<td>73</td>
</tr>
<tr>
<td>4.11</td>
<td>Block diagram of SOLID system test</td>
<td>74</td>
</tr>
<tr>
<td>A.1</td>
<td>Online impedance calculator for microstrip traces</td>
<td>86</td>
</tr>
<tr>
<td>B.1</td>
<td>TID simulation, using SHIELDOSE-2 model, for parts with limited radiation shielding</td>
<td>87</td>
</tr>
<tr>
<td>B.2</td>
<td>SEU rates for varying orbit inclinations</td>
<td>88</td>
</tr>
<tr>
<td>B.3</td>
<td>Impact rate simulation, using MASTER2009 model, for 640 km orbit altitude and 98° inclination</td>
<td>90</td>
</tr>
<tr>
<td>B.4</td>
<td>Simulation of AO erosion rate on Kapton surface, using NRLMSISE-00 model</td>
<td>91</td>
</tr>
<tr>
<td>C.1</td>
<td>Short-Circuit protection circuit schematic and simulation</td>
<td>94</td>
</tr>
<tr>
<td>C.2</td>
<td>PSPICE schematic and simulation results of input filter</td>
<td>96</td>
</tr>
<tr>
<td>C.3</td>
<td>OrCAD schematic of ToI detector circuit with input filter and Short-Circuit protection circuit</td>
<td>97</td>
</tr>
<tr>
<td>C.4</td>
<td>OrCAD circuit schematic of reference voltage filter</td>
<td>98</td>
</tr>
<tr>
<td>C.5</td>
<td>OrCAD circuit schematic of ToI detector comparator circuit</td>
<td>98</td>
</tr>
<tr>
<td>C.6</td>
<td>PSPICE simulation results of ToI detector circuit</td>
<td>98</td>
</tr>
<tr>
<td>D.1</td>
<td>PCB layout of prototype ToI detector</td>
<td>99</td>
</tr>
<tr>
<td>D.2</td>
<td>PCB layout of the MUX for the second SOLID prototype</td>
<td>100</td>
</tr>
<tr>
<td>D.3</td>
<td>Circuit schematic of MUX board</td>
<td>101</td>
</tr>
<tr>
<td>D.4</td>
<td>Terminal output from MCU software</td>
<td>104</td>
</tr>
</tbody>
</table>
## List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>TechnoSat mission specifications</td>
<td>5</td>
</tr>
<tr>
<td>2.1</td>
<td>SOLID functional requirements</td>
<td>8</td>
</tr>
<tr>
<td>2.2</td>
<td>Impact rates and probabilities for various orbits</td>
<td>12</td>
</tr>
<tr>
<td>2.3</td>
<td>SOLID technical requirements</td>
<td>13</td>
</tr>
<tr>
<td>3.1</td>
<td>Input filter requirements for the ToI detector</td>
<td>32</td>
</tr>
<tr>
<td>3.2</td>
<td>Input filter parameters for the ToI detector</td>
<td>35</td>
</tr>
<tr>
<td>3.3</td>
<td>MCU pin numbers and functions for eight SOLID detector boards</td>
<td>45</td>
</tr>
<tr>
<td>3.4</td>
<td>Software development tools</td>
<td>48</td>
</tr>
<tr>
<td>3.5</td>
<td>Quantitative trade-off analysis of system configuration</td>
<td>56</td>
</tr>
<tr>
<td>3.6</td>
<td>Qualitative trade-off analysis of system configuration</td>
<td>56</td>
</tr>
<tr>
<td>3.7</td>
<td>SOLID internal and external interfaces</td>
<td>57</td>
</tr>
<tr>
<td>4.1</td>
<td>Lab equipment for frequency response measurement of ToI input filter test</td>
<td>61</td>
</tr>
<tr>
<td>4.2</td>
<td>Test setup for ToI detector with MCU</td>
<td>63</td>
</tr>
<tr>
<td>4.3</td>
<td>Power consumption measurements</td>
<td>67</td>
</tr>
<tr>
<td>4.4</td>
<td>Test setup for ToI detector with MCU</td>
<td>67</td>
</tr>
<tr>
<td>4.5</td>
<td>Test setup for ToI detector with MCU</td>
<td>69</td>
</tr>
<tr>
<td>4.6</td>
<td>Test setup for ToI detector with TU Berlin EPS</td>
<td>73</td>
</tr>
<tr>
<td>4.7</td>
<td>Lab equipment for suggested CAN-bus test setup</td>
<td>74</td>
</tr>
<tr>
<td>4.8</td>
<td>Lab equipment for suggested test with new detector board and MCU board</td>
<td>74</td>
</tr>
<tr>
<td>D.1</td>
<td>Data from MUX board readout</td>
<td>102</td>
</tr>
<tr>
<td>D.2</td>
<td>Pin mapping between MUX board and detector board</td>
<td>103</td>
</tr>
<tr>
<td>D.3</td>
<td>MCU pin layout for second SOLID prototype</td>
<td>104</td>
</tr>
<tr>
<td>E.1</td>
<td>Expected SOLID flight parts list</td>
<td>106</td>
</tr>
</tbody>
</table>
Acronyms

ACS  Attitude Control System .............................................................. 2
ADC  Analog-to-Digital Converter .......................................................... 33
AO   Atomic Oxygen ........................................................................ 12
CAN  Controller Area Network .............................................................. 5
CDR  Critical Design Review .............................................................. 6
CMOS Complementary Metal-Oxide Semiconductor ................................ 21
CRC  Cyclic Redundancy Check ............................................................ 46
COTS Commercial Of-The-Shelf ........................................................... 16
CPU  Central Processing Unit ............................................................... 44
DC   Direct Current ......................................................................... 11
DD   Displacement Damage ................................................................. 9
DLR  Deutches Zentrum für Luft- und Raumfahrt .............................. 2
DSC  Digital Signal Controller ............................................................. 31
DSP  Digital Signal Processor .............................................................. 31
DUT  Device Under Test .................................................................... 61
ECSS European Cooperation for Space Standardization .................... 8
EGSE Electrical Ground Support Equipment ................................... 16
EM   Engineering Model .................................................................. 16
EMC  Electro-Magnetic Compatibility .................................................. 8
EMI  Electromagnetic Interference ....................................................... 8
EPS  Electrical Power Subsystem ........................................................ 2
ESD  Electro-Static Discharge .............................................................. 9
ESA  European Space Agency .............................................................. 1
EU   European Union ....................................................................... 1
FDIR Fault Detection Isolation and Recovery ..................................... 49
FFT  Fast-Fourier Transform ............................................................... 31
FM   Flight Model ......................................................................... 18
FPGA Field Programmable Gate Array ............................................. 44
GEO  Geostationary Orbit
<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>GPIB</td>
<td>General Purpose Interface Bus</td>
<td>75</td>
</tr>
<tr>
<td>GPIO</td>
<td>General Purpose Input/Output</td>
<td>45</td>
</tr>
<tr>
<td>HVI</td>
<td>High-Velocity-Impact</td>
<td>4</td>
</tr>
<tr>
<td>IADC</td>
<td>Inter-Agency Debris Coordination Committee</td>
<td>1</td>
</tr>
<tr>
<td>IC</td>
<td>Integrated Circuit</td>
<td>22</td>
</tr>
<tr>
<td>IDE</td>
<td>Integrated Development Environment</td>
<td>48</td>
</tr>
<tr>
<td>I/O</td>
<td>Input/Output</td>
<td>22</td>
</tr>
<tr>
<td>IP</td>
<td>Intellectual Property</td>
<td>44</td>
</tr>
<tr>
<td>LC</td>
<td>Inductor-Capacitor</td>
<td>34</td>
</tr>
<tr>
<td>LED</td>
<td>Light-Emitting Diode</td>
<td>40</td>
</tr>
<tr>
<td>LEO</td>
<td>Low Earth Orbit</td>
<td>5</td>
</tr>
<tr>
<td>LOCOS</td>
<td>LOCal Oxidation of Silicon</td>
<td>21</td>
</tr>
<tr>
<td>MCU</td>
<td>Microcontroller Unit</td>
<td>4</td>
</tr>
<tr>
<td>MEO</td>
<td>Medium Earth Orbit</td>
<td></td>
</tr>
<tr>
<td>MIPS</td>
<td>Million Instructions Per Second</td>
<td>33</td>
</tr>
<tr>
<td>MMOD</td>
<td>Micro-Meteoroids and Orbital Debris</td>
<td>1</td>
</tr>
<tr>
<td>MOSFET</td>
<td>Metal-Oxide-Semiconductor Field-Effect Transistor</td>
<td>29</td>
</tr>
<tr>
<td>MUX</td>
<td>Multiplexer</td>
<td>18</td>
</tr>
<tr>
<td>OBC</td>
<td>On-Board Computer</td>
<td>44</td>
</tr>
<tr>
<td>OpAmp</td>
<td>Operational Amplifier</td>
<td>31</td>
</tr>
<tr>
<td>PCB</td>
<td>Printed Circuit Board</td>
<td>10</td>
</tr>
<tr>
<td>PDR</td>
<td>Preliminary Design Review</td>
<td>6</td>
</tr>
<tr>
<td>PFM</td>
<td>Proto-Flight Model</td>
<td>16</td>
</tr>
<tr>
<td>PWM</td>
<td>Pulse-Width-Modulation</td>
<td>39</td>
</tr>
<tr>
<td>RC</td>
<td>Resistor-Capacitor</td>
<td>26</td>
</tr>
<tr>
<td>RF</td>
<td>Radio Frequency</td>
<td>10</td>
</tr>
<tr>
<td>SC</td>
<td>Short-Circuit</td>
<td>40</td>
</tr>
<tr>
<td>SD</td>
<td>Space Debris</td>
<td>1</td>
</tr>
<tr>
<td>SD</td>
<td>Space Debris’s</td>
<td>1</td>
</tr>
<tr>
<td>SEE</td>
<td>Single Event Effect</td>
<td>9</td>
</tr>
<tr>
<td>SEL</td>
<td>Single Event Latch-up</td>
<td>21</td>
</tr>
<tr>
<td>SEU</td>
<td>Single Event Upset</td>
<td>9</td>
</tr>
<tr>
<td>SM</td>
<td>Structural Model</td>
<td>16</td>
</tr>
<tr>
<td>SMPS</td>
<td>Switch-Mode Power Supply</td>
<td>10</td>
</tr>
<tr>
<td>SVN</td>
<td>SubVersion</td>
<td>48</td>
</tr>
<tr>
<td>SOLID</td>
<td>Solar Generator Based In-Orbit Space Debris Detector</td>
<td>2</td>
</tr>
<tr>
<td>TBC</td>
<td>To Be Confirmed</td>
<td>6</td>
</tr>
<tr>
<td>TBD</td>
<td>To Be Decided</td>
<td>6</td>
</tr>
<tr>
<td>TID</td>
<td>Total Ionization Dose</td>
<td>8</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Description</td>
<td>Page</td>
</tr>
<tr>
<td>-------------</td>
<td>--------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>ToI</td>
<td>Time-of-Impact</td>
<td>2</td>
</tr>
<tr>
<td>TU</td>
<td>Technisches Universität</td>
<td>4</td>
</tr>
<tr>
<td>UART</td>
<td>Universal Asynchronous Receiver/Transmitter</td>
<td>46</td>
</tr>
<tr>
<td>UNOOSA</td>
<td>United Nations for Outer Space Affairs</td>
<td>2</td>
</tr>
<tr>
<td>USB</td>
<td>Universal Serial Bus</td>
<td></td>
</tr>
<tr>
<td>WBS</td>
<td>Work Breakdown Structure</td>
<td>13</td>
</tr>
<tr>
<td>WDT</td>
<td>WatchDog Timer</td>
<td>46</td>
</tr>
</tbody>
</table>
Definitions

Detector line
Single copper line placed on the detector board.

Detector board
Part of the satellite side panel with multiple detector lines and where the solar cells are mounted.

ToI detector
A circuit to detect the Time-of-Impact when the detector board is hit by a particle. This circuit also includes an input filter and Short-Circuit-protection circuit.

MCU board
The Printed Circuit Board where the Microcontroller Unit and CAN-transceivers are placed. This board is located in the electronics box inside the satellite.

MUX board
The Printed Circuit Boards where the Multiplexers, Time-of-Impact detectors are placed.

First SOLID prototype
A prototype SOLID system which was implemented and tested before the beginning this thesis work.

SOLID embedded electronics
All electronics of the SOLID system.
Micro-Meteoroids and Orbital Debris (MMOD) regularly hit satellites. Depending on the particle size, impacts can cause performance degradation, lead to complete instrument and satellite failures or potentially loss of human lives on manned spacecrafts[1]. Design engineers need accurate and timely updated models of the MMOD distribution to correctly analyse and mitigate the risk. Man-made Space Debris’s (SD) is becoming an increasing threat to the space assets of the international space community, as was concluded at the European Space Agency (ESA) 6th European Space Debris Conference in Darmstadt 2013. Figure 1.1a shows a simulation of SD evolution over the next two centuries clearly illustrating the present and increasing risk of impacts. Many initiatives are already being initiated to deal with the SD issue. The Inter-

Figure 1.1 – Current and future prospected situation for the space debris environment

Agency Debris Coordination Committee (IADC), compromising 12 leading international space agencies, is working for information exchange and cooperative activities on SD activities. The European Union (EU), with its growing space assets like the Galileo and Copernicus space
1. INTRODUCTION

Morten Olsen
Thesis

systems, is currently supporting a number of SD related projects[4][5] and United Nations for Outer Space Affairs (UNOOSA) have released their space debris mitigation guideline[6]. Larger objects > 10 cm are tracked by ground radars and optical telescopes[7]. Distribution data for smaller objects is collected from in-situ detectors[8][9] as well as hardware samples returned from space[10]. As seen in fig. 1.1b, there is a great amount of small particles with an estimated trillions of particles larger than 100 µm. This particle size is generally considered the threshold for when particles may cause considerable satellite damage[1]. Due to the challenge of detecting these small particles, the exact distribution is, however, very uncertain. Discrete events such as satellite breakups[11] or collisions[12] each cause hundreds or thousands of new SD objects. Regularly detecting MMOD with in-situ detectors will improve the data quality for the small particle distributions and provide close to real-time coverage on particle populations created by discrete events.

1.1 In-situ Detectors

Many different MMOD detectors have already flown in space. A comparison between the performance of these is given in [3]. Any spacecraft instrument is always required to be compact, low weight and with a low power consumption. MMOD detectors should preferable have large detector surfaces, to increase the impact data rate. This works against the requirement of a small instrument and satellite surface area is often already occupied by other subsystem parts, payloads and primarily solar panels.

1.2 SOLID

To improve distribution models for MMOD in the range 100 µm to 1 cm, Deutches Zentrum für Luft- und Raumfahrt (DLR) in Bremen is developing the Solar Generator Based In-Orbit Space Debris Detector (SOLID) detector. The SOLID concept makes use of existing satellite subsystems for impact detection and classification, as illustrated in fig. 1.2a. Upon MMOD impact, a disturbance is observed in the Electrical Power Subsystem (EPS) which activates the SOLID Ebox and provides a Time-of-Impact (ToI). Two detector line layers are integrated into the satellite solar arrays, as shown in figs. 1.2c and 1.2d. When the solar array has been hit by a large enough particle, one or several of the detector lines will be cut, as shown in fig. 1.2b. Broken detector lines are then recorded by the electronics. Applying damage equations[13] gives an estimation of the impact crater size as a function of the particle size and impact velocity. Data from the satellite Attitude Control System (ACS) provides information on the impact momentum. SOLID is a simple concept with many benefits[3]:

- **Large detector surface area** - the detector lines are integrated into the solar arrays which often cover the majority of the satellite surface.
1.2. SOLID

Derive of particle properties (e.g. diameter, velocity) and damaged area on solar generator (mechanical/ESD) by using damage equations and ACS data.

(a) SOLID system concept showing use of host satellite solar array, EPS and ACS for impact detection and classification. The SOLID Ebox contains detector electronics and communication interface to the satellite.

(b) Detector line spacing sets impact size resolution (dimensions in µm). Shows minimum damage area from 300 µm particle impact at 5 km s⁻¹.

(c) Detector mesh integrated below satellite solar array. The two copper line layers are isolated from each other and from the solar array structure by using layers of kapton film.

(d) Cross-section of detector lines integrated in solar array structure.

Figure 1.2 – Illustration of the SOLID detector concepts

Images source: [14]
1. INTRODUCTION

- **Low power consumption** - impact detection relies on mechanical damage, hence most electronics remain in a passive state during the majority of the operating time.
- **Low mass** - no additional supporting structure for the detector board is required.
- **Low cost** - only simple electronics required, which will reduce development time and costs.

1.2.1 First Prototype

A first SOLID prototype has already been build and successfully verified by High-Velocity-Impact (HVI)-tests in 2012[15] at the Fraunhofer Institute for High-Speed-Dynamics, Ernst-Mach-Institute. Figure 1.3a shows the detector board after a simulated space debris impact in laboratory. The electronics of the first SOLID prototype are shown in fig. 1.3b. They include analog electronics for reading the detector lines, a Microcontroller Unit (MCU) for system handling and communication and a pulse detector circuit for ToI detection.

![Detector board of the first SOLID prototype after a simulated particle impact (here shown without solar cells mounted)](image1)

![Electronics board developed for the first SOLID prototype](image2)

Figure 1.3 – Hardware of the first SOLID prototype

Source: [16]

1.2.2 Flight Opportunity

SOLID is scheduled to fly on the TechnoSat nanosatellite currently being designed and build by Technisches Universität (TU)-Berlin. The TechnoSat mission specifications are summarized in table 1.1.
1.3 Thesis Work Scope

The work for this master thesis will be undertaken from March, 2013 until the end of August, 2013. The main tasks are summarized below:

- **Optimization and miniaturization of electronics**
  The first SOLID prototype, discussed in section 1.2.1, was not optimized for small size and low power consumption. It also only supported one detector board. For the TechnoSat mission, the electronics and harnesses must support multiple detector boards, be very compact and with low power consumption.

- **Reducing noise susceptibility of the ToI pulse detector**
  During the HVI-tests of the first SOLID prototype, the ToI pulse detector was found to have some noise issues and unreliable performance. It is therefore necessary to improve the robustness of this circuit.

- **MCU software and CAN-bus configuration**
  The first SOLID prototype used a simple MCU code for detector line readout. Some code instability was observed and remains to be solved. A more complete code adapted for TechnoSat should also be developed along with configuration of the Controller Area Network (CAN)-bus communication interface.

- **Transient switching speed on detector lines**
  The transient switching speed for reading out different detector lines was limited to around 5 ms in the first SOLID prototype. Decreasing this readout time would also be desirable.

The first SOLID prototype did not fully account for the restriction from the host satellite nor challenges from the space environment. These aspects will also be considered in this work.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mission purpose</td>
<td>Technology demonstrator</td>
</tr>
<tr>
<td>Orbit</td>
<td>Low Earth Orbit (LEO), circular</td>
</tr>
<tr>
<td>Orbit altitude</td>
<td>440 km to 630 km</td>
</tr>
<tr>
<td>Orbit inclination</td>
<td>&gt; 53°</td>
</tr>
<tr>
<td>Lifetime</td>
<td>1 year</td>
</tr>
<tr>
<td>Mass</td>
<td>15 kg</td>
</tr>
<tr>
<td>Solar array voltage</td>
<td>≈ 14 V</td>
</tr>
<tr>
<td>Satellite shape</td>
<td>Octagonal structure</td>
</tr>
<tr>
<td>Solar panel configuration</td>
<td>14 panels, body mounted</td>
</tr>
<tr>
<td>Attitude control</td>
<td>Three-axis stabilized</td>
</tr>
<tr>
<td>Expected launch date</td>
<td>September 2014</td>
</tr>
</tbody>
</table>
1.4 Document Structure

This document is structured in the following way:

**Chapter 2** states the SOLID mission goals and provides an analysis and collection of functional-, environmental-, and technical requirements, based on these goals and the TechnoSat mission constraints.

**Chapter 3** provides analysis and design parameters of various suggested electronic systems, including both hardware and software, required to meet the mission expectations.

**Chapter 4** lists the methods and results of hardware tests on implemented prototypes as well as suggestions for future tests.

**Chapters 5 and 6** draws the main conclusions of the work and provides suggestions and recommendations for future work efforts.

**Appendices A to G** provides further details and backgrounds on various design calculations and model simulations.

It is the author's intention that parts of this document may be reused for a SOLID Preliminary Design Review (PDR) report and serve as a template for the Critical Design Review (CDR) report and serve as a general SOLID design reference document.

A number of design recommendations are also given throughout this report. These are aspects which are currently not included or solved by the proposed design, but should be considered for future work. The document contains numerous statements of To Be Decided (TBD) or To Be Confirmed (TBC). These should not be interpreted as unfinished work of this master thesis, but rather reflect the nature of the real space project depending on many external factors and design information which as of the current writing is not available.
Requirements

This chapter discusses the functional, environmental and technical requirements for the SOLID detector, mainly considering the electronics. A brief overview of the project plan is also provided.

2.1 Goals and Vision

The TechnoSat mission is an opportunity for in-flight verification of SOLID and to demonstrate the usefulness of this detector technology. The results should support the decision to develop a larger scale SOLID detector eventually to be deployed on large satellite platforms such as communication satellites. The SOLID mission success criteria are:

- Make clear detection of several MMOD particles with defined ToI,
- Have stable detector performance throughout the mission,
- Realize the detector concept with minimum system requirements and impact on the host satellite design.

2.2 Functional Requirements

Section 2.2 lists the SOLID functional requirements.
2. REQUIREMENTS

Table 2.1 - SOLID functional requirements

<table>
<thead>
<tr>
<th>Function</th>
<th>Specifications/Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>MMOD particle size detection range:</td>
<td>100 µm to 1 cm TBD</td>
</tr>
<tr>
<td>Particle diameter accuracy</td>
<td>&lt; ±200 µm TBC</td>
</tr>
<tr>
<td>ToI detection accuracy</td>
<td>&lt; 1 s TBC</td>
</tr>
<tr>
<td>ToI detection reliability</td>
<td>&gt; 95%*TBD</td>
</tr>
<tr>
<td>Flexible design</td>
<td>Easy to adapt for other satellites</td>
</tr>
<tr>
<td>No negative impact on other satellite subsystems</td>
<td>Mitigation of failure scenarios TBD</td>
</tr>
</tbody>
</table>

*Based on low expected MMOD impact rate and possible limitations of ToI detection use, as will be discussed in section 2.3.4 and section 3.2.4 respectively.

2.3 Environmental Requirements

Satellite payloads are exposed to extreme working environments. Potential design issues, due to the space environment, should be identified and mitigated as early as possible. Failure to due so may lead to significant cost overruns and time delays in later project phases. Some potential issues for the SOLID design have been identified:

- Compliance with space standards like European Cooperation for Space Standardization (ECSS) etc.
- Radiation exposure - especially parts located near the satellite surface to the space environment
- Electromagnetic Interference (EMI) including spacecraft charging
- MMOD impact rate
- Atomic oxygen - only external surfaces

Each of these environmental factors are evaluated in the following sections.

2.3.1 Space Standards

Compliance to space standards like ECSS is not required by TU-Berlin. Due to the author’s limited experience in developing flight hardware, the ECSS standards are used as reference to support design decisions or when evaluating potential design and environmental issues.

2.3.2 Radiation Effects

The high orbit inclination in the expected orbit of TechnoSat takes it frequently through the auroral polar regions with increased radiation exposure. In [17, Table 2.6], the TechnoSat Total Ionization Dose (TID) is estimated for different orbits and shielding thicknesses. These estimates
are valid for components placed inside the satellite structure. Some of the SOLID detector parts are placed on or near the satellite surface with little or no protection from the satellite shield. To estimate the radiation environment, for these parts, additional simulations are performed using the SPENVIS online software. The applied simulation parameters are listed in appendix B.1. Simulations show that TID can be more than ten times higher when the aluminium-shield is decreased from 0.5 mm to 0.2 mm or less. Parts located on the surface may also be subjected to Displacement Damage (DD)[18, Sec. 8]. Different electronics are more prone to some radiation threats and less to others. They also have different failure and degradation effects[18, Table 4.3]. Single Event Effects (SEEs) cause sporadic changes to bit-states in logic circuits, particularly flash memory blocks[19]. The SOLID Single Event Upset (SEU) rate is also estimated using SPENVIS. Simulation parameters and outputs are listed in appendix B.2. Simulations show a low expected SEU rate between 22 to 130 bit errors over the entire mission duration, depending on the satellite orbit chosen. Due to the relatively short mission duration, radiation exposure is limited. Depending on the final chosen orbit and satellite shielding, some parts may need further analysis or testing to verify that they will operate as intended. This applies in particular to parts located outside the satellite radiation shield.

2.3.3 Electromagnetic Compatibility

2.3.3.1 Spacecraft Charging

The SOLID detector lines are placed very close to the solar arrays, which are vital parts of the satellite system. It is therefore important that SOLID does not increase the risk of damage to itself or nearby circuits when affected by the space environment. With the TechnoSat orbit inclination most likely residing above 55°, spacecraft charging effects should be considered[20, B.2.2]. Charging effects include surface and internal charging. These may cause Electro-Static Discharges (ESDs) which can disturb or damage electronics. At solar array voltages > 10 V, ESDs generated near solar arrays may cause secondary arcs[21, p. 7.1]. The TechnoSat solar array voltage of ≈ 14 V is only slightly higher than this value, hence secondary arcs are not considered a major concern.

The majority of the detector line surfaces are covered by solar cells, including cover glass with a total thickness of around 280 µm[14]. Here, it is not necessary to consider surface charging effects. Due to the shape and spacing of the solar cells, parts of the detector line surfaces are more directly exposed to the space environment, and surface charging should be considered. If the detector lines are coated, the coating materials should also comply to [21, sec. 6.3.3]. To avoid build-up of high voltages, detector lines close to the surface should always have a current leakage path to ground. No conductor surface should be left floating[21, Sec. 6.3].

The TechnoSat solar arrays employ protection circuits which limit propagation of ESD voltages. It should still be ensured that the SOLID electronics and detector lines do not couple to, or are damaged by, these transient ESD events.
2.3.3.2 Conducted EMI from TechnoSat Power Subsystem

The first SOLID prototype used a direct connection to the satellite EPS, for measuring the ToI. The Switch-Mode Power Supplies (SMPSs), inside the EPS, are inherently noisy circuits. Conducted and radiated EMI should therefore be considered, to prevent noise issues and performance degradations.

2.3.3.3 Antenna Induced EMI

During transmission, the TechnoSat communication antenna may induce noise in the detector lines, since these can effectively form a receiving antenna. The communication system transmits 1W at 430 to 440 MHz[17] equal to a wavelength of 68.2 to 69.8 cm. It can be assumed that the detector line length is approximately the same as the detector board dimension, 14 to 17 cm. Thus, the detector lines may form an efficient quarter-wavelength Whip Printed Circuit Board (PCB) antenna[22] which according to [23] is the wavelength at which most noise is coupled. To reduce the effect of this parasitic receiving antenna, a number of measures could be considered:

- Running a ground plane close underneath the detector lines will make it behave more like a transmission line, rather than an antenna.
- Depending on the polarization of the TechnoSat transmitting antenna, it may be possible align the detector lines in opposite polarization.
- Radio Frequency (RF) chokes can be placed at the detector line outputs to filter out RF noise.
- Reduce loop areas between detector lines and ground connections, since these may form loop antennas.

The detector lines are all covered by a thin Kapton surface protection layer, and are many places obscured by the solar cells. This helps to avoid noise pickup from the communication antenna.

**Recommendation 1 (EMC testing)** Since EMI noise issues are almost impossible to analytically determine, it is recommended, if possible, to perform both radiated and conducted EMI susceptibility tests of the SOLID detector together with the other TechnoSat subsystems.

2.3.3.4 Induced Effects by Orbital Motion

Assuming a perpendicular magnetic field alignment and an estimated Earth magnetic field strength of 30 µT, the potential induced in long straight detector lines is calculated as[21,
2.3. Environmental Requirements

\[ V = l_{\text{line}} (v_{\text{sat}} \times \vec{B}) = 16 \, \text{cm} \cdot 7.8 \, \text{km/s} \cdot 45 \, \mu \text{T} \approx 56 \, \text{mV} \] (2.1)

where \( l_{\text{line}} \) is the detector line length, \( \vec{B} \) the Earth’s magnetic field vector at the orbit location and \( v_{\text{sat}} \) the satellite orbital velocity vector. Orbital induced voltage is a noise signal with very low frequency which is easily filtered out as long as precise Direct Current (DC) measurements are not required. Induced voltage is therefore not currently considered an issue. In larger scale implementation of the SOLID concept, with longer detection lines, induced voltage will have to be considered. Since only very low currents are flowing in the detector lines, no attitude disturbance is expected due to magnetic torques.

In appendix G, other potential noise sources were also evaluated, however none of these were found to pose any considerable design challenges.

### 2.3.4 Impact Rate

The MMOD impact rate affects the SOLID data rate and the potential quality of the SOLID verification results. The TechnoSat solar panel dimension is expected to be 167 mm × 145 mm[17, fig. 4.4]. Each of the eight satellite sides can accommodate up to two solar panels. When mounting eight detector boards, the total detector surface area is approximately

\[ A_{\text{detector}} = 8 \cdot 0.167 \, \text{m} \cdot 0.145 \, \text{m} = 0.193 \, \text{m}^2 \] (2.2)

The probability that at least a minimum number of particles hit the detector, during the mission lifetime, is calculated using a cumulative Poisson distribution

\[ P(X \geq k) = 1 - P(X < k) = 1 - \sum_{x=0}^{k-1} \frac{\exp(-\lambda \tau) (\lambda \tau)^x}{x!} \] (2.3)

where \( \lambda [\text{m}^{-2} \text{year}^{-1}] \) is the impact rate, \( \tau [\text{year}] \) the mission lifetime and \( k \) the number of impacts.

Table 2.2 lists expected impact rates and impact probabilities. The impact rates are simulated using the MASTER2009 model. Simulation parameters and outputs are listed in appendix B.3. As seen, impact rates and resulting probabilities are much lower in the low orbits.

**Recommendation 2 (Impact rates)** If a low altitude orbit is confirmed by TU-Berlin, it is recommended to increase the detector surface area and the expected lifetime of the detector. Results from these calculations could also serve as an argument for choosing a higher altitude orbit for the TechnoSat mission.

The simulations should still be taken with some care, since MASTER2009 model data are extrapolated from different orbits and time epochs thus actual particle fluxes may be different.
2. REQUIREMENTS

Table 2.2 – Impact rates and probabilities for various orbits. Particle size > 100 \( \mu \text{m} \), duration is one year (mission lifetime).

<table>
<thead>
<tr>
<th>( h ) (km)</th>
<th>( i ) ((^\circ))</th>
<th>Impact Rate (m(^{-2}) year(^{-1}))</th>
<th>( \text{P(} \geq 1 \text{ impact)} ) (%)</th>
<th>( \text{P(} \geq 5 \text{ impacts)} ) (%)</th>
<th>( \text{P(} \geq 10 \text{ impacts)} ) (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>430</td>
<td>53(^{\circ})</td>
<td>11</td>
<td>87</td>
<td>6</td>
<td>&lt; 1</td>
</tr>
<tr>
<td>640</td>
<td>53(^{\circ})</td>
<td>25</td>
<td>99</td>
<td>51</td>
<td>2</td>
</tr>
<tr>
<td>430</td>
<td>98(^{\circ})</td>
<td>12</td>
<td>90</td>
<td>9</td>
<td>&lt; 1</td>
</tr>
<tr>
<td>640</td>
<td>98(^{\circ})</td>
<td>39</td>
<td>99</td>
<td>86</td>
<td>22</td>
</tr>
</tbody>
</table>

2.3.5 Atomic Oxygen

At orbital altitudes between 300 to 700 km, Atomic Oxygen (AO) is the dominating neutral specie in the Earth’s atmosphere[7, fig. 3.9]. Due to the very reactive behaviour of AO, it may be a threat to externally exposed surfaces[24, p. 5.1.13]. Copper is not very susceptible since it creates a thin oxide layer, which protects the copper layer underneath from further oxidation[25]. Detector lines directly exposed to the space environment, for example lines neighbouring an impact area where protective film might be broken, will not be susceptible to AO corrosion as long as they are made of copper. Materials like kapton or other surface coatings may be subject to AO corrosion. Simulations show that a Kapton surface, permanently oriented on the satellite wake side, may be subject to an erosion rate of around 30 \( \mu \text{m} \) per year. Simulations parameters and outputs are listed in appendix B.4. With TechnoSat scheduled for launch in late 2014, solar cycle 24 is expected to go towards its minimum. This will reduce the AO corrosion rate due to the shrinking (cooling) of the Earth’s atmosphere. Since the mission duration is also short, AO corrosion will be less severe.

**Recommendation 3 (Surface cover material AO corrosion)** If one or several detector board surfaces are almost permanently oriented towards the satellite wake side, where AO corrosion is most intense, it is recommended to consider the erosion properties of the selected surface materials. This is to avoid synergistic issues, for example surface charging issues due to corroded surface cover materials.

2.4 Technical Requirements

Table 2.3 lists the SOLID technical requirements. These are mainly derived from the TechnoSat PDR report[17].
Table 2.3 – *SOLID technical requirements*[17]

<table>
<thead>
<tr>
<th>Specification</th>
<th>Requirement</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Power supply</td>
<td>5 V or 3.3 V</td>
<td>Nanosat standards.</td>
</tr>
<tr>
<td>Power consumption (active)</td>
<td>&lt; 550 mW</td>
<td></td>
</tr>
<tr>
<td>Power consumption (sleep)</td>
<td>&lt; 100 mW</td>
<td></td>
</tr>
<tr>
<td>Operating temperature</td>
<td>−45 to 85 °C</td>
<td></td>
</tr>
<tr>
<td>MCU PCB Size</td>
<td>80 mm × 60 mm</td>
<td>Including connectors.</td>
</tr>
<tr>
<td>MUX PCB Size</td>
<td>&lt; Detector board size</td>
<td>TBC</td>
</tr>
<tr>
<td>Detector board size</td>
<td>166.76 mm × 145.4 mm</td>
<td>Same as solar array.</td>
</tr>
<tr>
<td>Number of detector boards</td>
<td>8</td>
<td>TBD.</td>
</tr>
<tr>
<td>Weight</td>
<td>550 g</td>
<td>Including detector boards.</td>
</tr>
<tr>
<td>Component types</td>
<td>COTS</td>
<td>Cost limitation.</td>
</tr>
<tr>
<td>Data rate</td>
<td>≪ 200 kB per day</td>
<td></td>
</tr>
<tr>
<td>Communication interface</td>
<td>Redundant CAN-bus</td>
<td></td>
</tr>
<tr>
<td>TID</td>
<td>&lt; 5.547 kRad</td>
<td>Depends on shielding thickness.</td>
</tr>
<tr>
<td>Lifetime</td>
<td>&gt; 1 year</td>
<td></td>
</tr>
</tbody>
</table>

2.5 Project Planning

This section contains suggestions for project time schedules and testing of SOLID system, focusing primarily on the electronics.

2.5.1 Work Breakdown Structure

The SOLID Work Breakdown Structure (WBS) is shown in fig. 2.1. It provides an overview of the main work tasks in implementing and testing the SOLID system.

2.5.2 Project Time Schedule

Having an agreed project time schedule allows the project progress to be monitored, and provides a common understanding of the short term goals and time constraints. Figure 2.2 shows a proposed SOLID project schedule. The project team is currently small, development is therefore more flexible and information-sharing easier. For this reason, it is suggested to combine the SOLID PDR and CDR and leave some margin in Phase C to correct any findings from this review.
2. REQUIREMENTS

Figure 2.1 – SOLID Work Breakdown Structure
2.5. Project Planning

Figure 2.2 – SOLID overall project time schedule
2.6 Verification Methods

Up until now and to the author's knowledge, no SOLID-type MMOD detector has previously flown in space, which indicates that many of the applied system concepts per definition do not have flight heritage. The SOLID project is highly cost- and time constrained. Consequentially, at component level, space-rated parts can in general not be used, due to their high cost. Available resources to test and qualify commercial parts are also limited.

To respond to these limitations, it is suggested that verification should primarily be achieved from analysis. This involves providing sufficient proof that components and systems will operate as intended, under all operational conditions, by using input from datasheets and theoretical analysis. Critical components, which might severely degrade the detector performance or influence other satellite subsystems, should all be verified by tests. This includes qualification testing at component level (e.g. radiation testing).

2.6.1 Model Philosophy

From [17], the following model philosophy is expected by TU-Berlin:

- Engineering Model (EM) (named DM in reference document)
- Structural Model (SM)
- Proto-Flight Model (PFM)

One or several SOLID EMs are expected, depending on the outcomes of functional tests of the implemented designs. When a satisfactory design has been found, a PFM will be implemented. This will undergo a combined qualification and acceptance test. Since the SOLID systems are implemented using Commercial Of-The-Shelf (COTS) parts, EM and PFM models will in general use the same components. EM models may therefore also be used for qualification testing, assuming that design changes does not alter the characteristic being tested for.

2.6.2 EGSE

For the first SOLID prototype, a software Electrical Ground Support Equipment (EGSE) was implemented in LabView to test the functionality of the MCU software, detector line readout and ToI pulse detector. Currently, no EGSE development is included in the project plan.

Recommendation 4 (EGSE) It is recommended to include at least one EGSE for testing of the complete SOLID detector system without requiring access to any of the remaining TechnoSat subsystems, e.g. the EPS and solar arrays.
This EGSE should have a suitable communication and control interface to a PC and the ability to provide stimuli inputs for impact pulse signals and noise. It could also include options for endurance testing with tests running for several days/weeks to determine the stability and performance of the detector. The EGSE will be a very useful tool during qualification/acceptance tests of the SOLID PFM to measure changes in system performance following a particular qualification test.
Design

This chapter describes the design and development of both hardware and software for the SOLID embedded electronics. A list of chosen parts is provided in table E.1, appendix E and a complete set of design calculations is listed in appendix G.

3.1 Detector Board

A new detector board is currently being developed in parallel to this master thesis work. The design and layout of the detector board will also influence the SOLID embedded electronics in particular the MCU board. This section briefly describes the detector board concept along with circuit analysis, environmental considerations and design suggestions.

Figure 3.1 illustrates the detector board circuit layout for detector lines and Multiplexers (MUXs) and the MUXs signal interfaces to the MCU board. The detector board is illustrated with 16-pin MUXs whereas in the SOLID Flight Model (FM) 32-pin MUXs are expected. The detector lines are arranged in groups of 16 and each group shares the same output, OUT_1 to OUT_16. The input of the first detector line, IN_1, is shared with the inputs from the first detector lines of all the other groups. This configuration allows many detector lines to be monitored by relatively few MUXs. The input MUX, MUX_IN, selects a detector line from each of the 16 groups to be biased by the 3.3 V bias voltage. The output MUX, MUX_OUT, then sequentially selects the output of each group and checks if the biased detector line is intact, that is, reading 3.3 V on the MUX READ_OUTPUT. The diodes prevent the remaining 240 detector lines from being biased at the same time. The first SOLID prototype implemented a very similar detector board circuit concept, just with more detector lines[16]. The new detector board has a slightly larger detector line spacing of 300 µm. Using the solar panel dimensions from table 2.3, the number of required detector lines for each detector board is estimated to

\[
\begin{align*}
  n_{linesX} &= \frac{166.76 \text{ mm}}{300 \text{ µm}} = 556 \text{ detector lines} \\
  n_{linesY} &= \frac{145.4 \text{ mm}}{300 \text{ µm}} = 484 \text{ detector lines} \\
  n_{total} &= 1040 \text{ detector lines}
\end{align*}
\]
3.1.1 Isolation Diodes

To electrically isolate the overlapping detector lines, each line must be equipped with a diode, thus around 1040 diodes are needed per detector board. To accommodate all these diodes, their footprint must be extremely small, as discussed in [16]. A potential issue is diode reverse currents. At normal temperature, this is in the range of few nA but at elevated temperatures may increase by a factor 10,000 or more. These diodes may be subjected to more extreme temperatures, due to their location near the satellite surface. Connecting more than thousand diodes in parallel could result in unacceptable reverse currents.

Figure 3.2 shows a simplified circuit diagram of the detector board with three diodes, one switch closed in each MUX and several current leakage paths. The leakage resistors, \( R_{\text{LEAK}} \), are either intentionally placed leakage resistors or parasitic leakage through the detector board insulator material. The first three diodes are forward biased from the input and the remaining are reversed biased. There are two possible scenarios: The leakage resistors can sink all the reverse currents from the six reversed biased diodes thus keeping the voltage of the two inactive pins of \( \text{MUX\_IN} \) around zero volt. The second situation is when the diode reverse currents are greater than what the leakage resistor can sink and the reverse currents starts to charge the parasitic capacitances until the diodes connected to the output load become forward biased and starts to pass the reverse current to the output. In both cases, reverse current will flow in the reverse biased diodes.
Equation (3.2) shows the calculation of worst case diode reverse current per detector board,

\[ I_{\text{reverse,diode}} = I_{\text{reverse, max}} \cdot \eta_{\text{diodes}} \]

\[ I_{\text{reverse, max}}(125 \degree \text{C}) = I_{\text{reverse,diode}} \cdot n_{\text{diodes}} \]

\[ = 30 \text{nA} \cdot 1040 = 31.2 \text{µA} \]

If the reverse current become too high it might also cause slower MUXs switching times, due to increased discharging times as will be discussed in section 3.1.4.

The diodes are also possibly exposed to a more extreme radiation environment, as discussed in section 2.3.2. Leakage current of bipolar diodes increase after DD[18, Table 8.1]. The datasheet does not specify if the device is bipolar or not.

**Recommendation 5 (Testing of isolation diodes)** Even though equation (3.2) shows plenty of margin for the reverse currents, thermal and radiation testing of the diodes is recommended, to ensure their performance in the harsh space environment.
3.1.2 Multiplexers

The new detector board has reduced the required amount of MUX channels, by simultaneously biasing both the X- and Y-axis detector lines with the same input MUX. Further, assuming that the number of detector lines per detector line group and the number of detector line groups are equal, the minimum required number of MUX channels is calculated to

\[
\begin{align*}
 n_{\text{channelsMUXIN}} &= \sqrt{n_x} = \sqrt{556} = 24 \text{ channels} \\
 n_{\text{channelsMUXOUT}} &= \sqrt{n_x} + \sqrt{n_y} = \sqrt{556} + \sqrt{484} = 46 \text{ channels} \\
 n_{\text{channelsMUXtotal}} &= 70 \text{ channels}
\end{align*}
\]

It would therefore require three 32-pin MUXs to implement the detector board, using these dimensions.

3.1.2.1 Radiation Effects and Failure Modes

The Complementary Metal-Oxide Semiconductor (CMOS) switches inside the MUXs may be susceptible to SEE and degradation from TID\cite{18}. TID in some CMOS devices, for example a COTS LOCal Oxidation of Silicon (LOCOS) CMOS device, showed performance degradation after a few kRad, greatly increased leakage currents after 10 kRad, and after 30 kRad complete device failure due to shifted gate threshold voltage\cite{26}. From table E.1 in appendix E, the suggested MUX part is the ADG732BSUZ. Unfortunately, the datasheet does not specify the applied CMOS technology.

**Recommendation 6 (MUX radiation dose)** *If the MUXs are located near the satellite surface, it is recommended that the TID is kept well below a few kRad. Otherwise, radiation test on the selected part is recommended.*

CMOS switches are also susceptible to Single Event Latch-ups (SELS). A current limiting resistor is placed at the 3.3 V bias line on the input MUX, MUX\_IN. This prevents damage to the circuit should one of the lines accidentally be shorted to ground, due to either a SEL or shortened wires after an impact. The MUX part has an input current limit of 30 mA. The value of the current limiting resistor should then be

\[
R_{\text{limit}} \geq \frac{V_{\text{bias}}}{I_{\text{limit}}} = \frac{3.3 \text{ V}}{30 \text{ mA}} = 110 \Omega
\]

A 470Ω resistor is selected to leave some margin. SEL error states can be removed by power cycling the device. It should therefore be possible to easily cycle the power on the MUXs, for example before every scan of the detector lines. A failure scenario with short-circuit between two supply pins is also considered\cite{27}.
Recommendation 7 (Supply current limitation) To prevent device damage from a short-circuit error between the supply pins, it is recommended to place a current-limiting Integrated Circuit (IC), to monitor the common supply line to all MUXs. An error state can be detected and handled by the IC and the circuit can possibly be reset by a power cycling.

3.1.3 Charging Effects

Charge build-up on floating detector lines, should be prevented, as discussed in section 2.3.2. Each line should have a leakage path to ground. Using the recommendations from [21, 6.3.1g], the maximum resistance to ground is calculated as

\[ R_{\text{leakage}} \leq \frac{V_{\text{max}}}{n_{\text{cluster}} A_{\text{line}} J} = \frac{3.6 \text{ V}}{32 \cdot 0.15 \text{ cm}^2 \cdot 10 \text{ nA cm}^{-2}} = 75 \text{ M\Omega} \]  

(3.5)

where \( V_{\text{max}} \) is the maximum allowed voltage charge level limited by the MUX maximum ratings (0.3 V above supply voltage), \( n_{\text{cluster}} \) the number of detector lines for each leakage resistor, \( A_{\text{line}} \) the estimated copper area of each detector line and \( J \) the maximum allowed current density taken from the ECSS recommendations. From the MUX datasheet, the input leakage current is specified to 10 pA @ 5 V, which is equivalent to a leakage resistance of 500 G\( \Omega \). The leakage path, which was also shown in fig. 3.2, is therefore insufficient. The MUX datasheet states that the device has over-voltage protection diodes on all Input/Output (I/O) pins. These will clamp voltage build-up on detector lines since they are all connected to a MUX I/O pin, as long as the detector line is intact. When a detector line is broken, the detector line stretch between the impact damage area to the cathode of the reverse blocking diode only has two leakage paths; through the reverse leakage current of the diode, which may be less than a few nA, and through the kapton insulation film to structure which has very high resistivity. The chosen diode has a built-in Zener-clamp which limits the voltage build-up to 3.6 V. Furthermore, the insulation and adhesive materials expected to be used (see table E.1 in appendix E), have dielectric strengths of 40 kV mm\(^{-1}\) or more. 25.4 \( \mu \)m insulation layer thickness is expected. Charge build-up, on the detector lines, should therefore not be an issues. If other than the parts specified here are selected for the final PFM, it must be ensured that the new parts have similar protection features. Otherwise, design changes might be necessary. Another potential issue is the copper-clad coating material. In table E.1, appendix E, a polyimide kapton coating material is selected. [21, p. 6.3.3.6] recommends not using polyimide materials with a thickness above 50 \( \mu \)m. If this recommendation cannot be met, it might be necessary to add another ESD protective coating. From the datasheet of the copper-clad coating, the material resistivity exceeds the limit value recommended in [21, 6.3.3.6b]. From [21, B.2.2], the surface coating requirements may somewhat be relaxed.

3.1.4 Transient Switching Time

In the first SOLID prototype, the transient switching and readout time was limited to around 5 ms. This is could be due to parasitic capacitances in the long detector lines combined with a
3.1. Detector Board

relatively large 33 kΩ pull-down resistor. In fig. A.1 the parasitic capacitance of each detector line is estimated to 3 pF cm⁻¹. With a detector line length around 50 cm this equals 150 pF per detect line (dimensions of the first SOLID prototype where somewhat larger than the dimensions expected for TechnoSat).

Figure 3.3 shows the circuit along with parasitic capacitances of the detector lines and MUXs[28]. Considering the following situation: The switches S1 and S2 are initialled closed, all other switches are open and the MUX_OUT output is initially high. Switch S2 is then opened and S3 closed. The output of MUX_OUT now reads out the broken detector line as shown in the figure. This would require the three parasitic detector line capacitors, connected to switch S3, to be discharged before the voltage can be read out correctly (in the real detector there are around 32 parasitic capacitors connected to S3). Calculations show that the discharge time is much faster than 5 ms hence this situation cannot describe the complete problem. From the datasheet of the CM1213-08MR diodes used in the first SOLID prototype, the reverse leakage current is around 100 nA. Having around thousand diodes in parallel and connected to a 33 kΩ pull-down resistor can drive an output voltage of

\[ V_{\text{outleakage}} = 1000 \cdot 100 \text{nA} \cdot 33 \text{kΩ} = 3.3 \text{V} \]  

(3.6)

This voltage is indeed enough to cause problems with the detector line readout. Due to the relatively high reverse currents of these diodes, all the detector line capacitors needs to discharge whenever reading out a broken detector line. This discharge time-constant is roughly calculated to be in the order of 2.5 ms which does indeed correlate well with the laboratory measurements. Calculations are listed in appendix G.

One way to improve the switching speed is to reduce the pull-down resistor such that the output voltage driven by the leakage current is below the "logic low" voltage of the MCU digital input. This slightly increases the circuit power consumption when reading out the detector lines, however since this is only done periodically, average power consumption should not increase significantly. It will also help by changing to diodes with lower leakage current, which is the case for the new chosen diodes. Another thing which can greatly improve the switching speed is to place the diodes close to the MUX_OUT, to reduce the parasitic capacitance near this MUX and add some leakage resistors to each output of MUX_IN, which should have small enough resistance to sink at least the leakage current of all diodes connected to it thus allowing the parasitic capacitances to discharge when the detector lines are left disconnected.

To improve the speed when changing switch positions in MUX_IN, a large capacitor can be added right at the MUX input to work as a "charge bank" to quickly charge all the parasitic capacitors when a switch has been closed. This capacitor value should be considerably larger than the combined parasitic capacitance on the switch being closed.

If the capacitance of the cable between the MUX_OUT and the MCU is causing unacceptable propagation delays, an output buffer can be placed at the output of MUX_OUT, which effectively would provide a very low output impedance for charging and discharging the parasitic capacitances of the cable.
The performance of the readout system might also depend on changes in the space environment as the satellite goes through different orbit regions. It is important that the integrity of the readout system is high. Otherwise, important impact data might not be registered by the electronics.

**Recommendation 8 (MUX readout timing)** If fast readout times for the MUXs is required, it is recommended to include a few detector lines which has been pre-cut from manufacturing and use these to make in-flight tests for the integrity of the MUX readout circuit. These could be read out at each detector line scan and produce error or warning if the read out value is incorrect. There should also be included an easy function to change the readout speed by the SOLID software.

![Figure 3.3 – Limitations of MUX switching transient times due to parasitic capacitances](image)

3.2 Time-of-Impact Detector

To properly model the MMOD distributions, it is important to know the ToI. For the SOLID detector concept, there are two ways to determine this:

- Frequent scans of the detector lines
- Impact detection from disturbance signal caused by the impact

The first option requires continuous MCU operation at high clock-frequency, in order to provide proper time resolution. This will increase the MCU power consumption. The frequent switching could potentially create system noise. This method scales poorly for larger systems, due to increasing switching requirements. The second option requires a disturbance signal source. There are several potential physical sources, as explained in [1]. For the SOLID detector concept, one method is to bias each detector line with a current signal and then detect a change in the current level when one or more detector lines are broken as suggested in [29]. The number of current-to-voltage converters and current change detectors required, per detector board, is
3.2. Time-of-Impact Detector

It was expected that a disturbance signal would be created in the satellite EPS upon an impact which was confirmed during HVI testing of the first SOLID prototype\cite{3}. It was observed, that particles impacting an illuminated solar array produce a transient voltage disturbance pulse on the solar array output voltage. The exact data from these experiments are still being analysed as of this writing. In general, the transient disturbance pulse has a signal frequency in the range 2 to 10 MHz with an amplitude of approximately 400 to 1000 mV. It was then suggested to use this disturbance pulse as ToI detection. The advantage is that only one ToI detector circuit is required, for each independently controlled solar array, irrespective of size. This reduces the required number of components, especially for larger detector systems. The detector lines only needs to be scanned after an impact has been detected. The MCU power requirements are therefore also reduced. However, this method also has some challenges:

- **SMPS and ESD noise from the satellite EPS**
  SMPSs are inherently noisy circuits. The ToI detector circuit must therefore be designed to reject SMPS noise. The solar arrays may produce and conduct high voltage transient ESDs. These might cause false triggering or worse damage the ToI detector electronics.

- **High frequency, broadband disturbance pulse signals**
  The disturbance pulse has a frequency in the approximate range 2 to 10 MHz. High-speed electronics are therefore required which consume significantly more power than low-speed electronics and are more tedious to design. The broadband nature of the signal makes it harder to efficiently filter out adjacent noise without also effecting the signal itself.

- **Possible failure scenarios**
  The ToI detector connection point is directly at the solar array output. A short-circuit failure here would compromise the whole solar array. This kind of failure should therefore be mitigated as part of the ToI detector design.

The first SOLID prototype already included a pulse detector circuit to detect these disturbance pulses. This circuit is shown in fig. 3.4. It comprised two simple voltage dividers, a comparator, a reference signal and an output latch controlled by the MCU. As mentioned in section 1.2.1, this circuit experienced some glitches during the HVI-tests. Possible reasons for these glitches, and potential solutions, are discussed below:

- **High source impedance**
  The solar array voltage will exceed the 5 V supply (in the new design 3.3 V is used) for the ToI pulse detector electronics. A voltage divider was therefore added, to scale down the input voltage. Unfortunately, this also reduces the measured signal amplitude and results

\[
\sqrt{n_{\text{total}}} = \sqrt{1040} = 33 \text{ circuits} \tag{3.7}
\]

where \(n_{\text{total}}\) is the number of detector lines from equation (3.1). Each detector line will also require both an isolation diode and a current-limiting resistor. This method therefore also scales poorly for large systems, due to increased component count.
in a high source impedance $> 6.67 \, \text{k}\Omega$. This is problematic in high-speed circuits, due to the Resistor-Capacitor (RC) time constants originating from a combination of parasitic capacitance in wires and electronics together with the high source impedance\textsuperscript{[30]}. Making the resistors smaller would draw too much current from the solar arrays.

**Solution:** Use DC-blocking capacitor to remove the high DC-level voltage of the solar array. A DC bias-circuit is then needed, to place the input signal between the 0 V to 3.3 V supply voltage range of the ToI detector.

- **Noise susceptibility**
  No filter and minimal hysteresis were added to the comparator making the system noise-prone. This already became clear during HVI-tests, when the circuit was upset by noise from the 230 V 50 Hz domestic supply. Noise will be even more pronounced when connecting the ToI detector to the satellite SMPS which operates at frequencies much closer to the signal frequencies.

**Solution:** Use filters to remove unwanted EPS noise and add more hysteresis to the comparator.

- **Manual setting of reference voltage**
  The comparator reference signal had to track changes in the solar array voltage, thus requiring continuous regulation.

**Solution:** The DC-blocking capacitor discussed above makes the circuit independent from the solar array voltage. The reference voltage can then easily be taken from the DC-bias circuit. Adding a small DC-offset will set the threshold voltage trigger level.

### 3.2.1 Input Noise Characterization

Before designing the ToI detector filters, the EPS noise sources are briefly discussed in an attempt to characterize the noise signals. Obtaining detailed knowledge of the noise signals is...
very difficult, without the exact EPS hardware, which is still being developed at the time of this writing. Figure 3.5a shows some of the possible noise sources and how their signal path connects to SOLID. As seen, it is mainly the input noise from the EPS and noise from the solar array which is of concern. Unfortunately, the input noise of SMPSs on satellites is poorly defined, since design engineers are normally only concerned about the output noise coupling to other satellite subsystems and payloads. The noise signals depend on many factors like converter type, EPS filters, used materials/components, the physical layout of the PCBs, cable wiring, operation mode and effects from the space environment.

The TechnoSat EPS will be based on heritage from previous satellites built by TU-Berlin. Detailed knowledge of previous or current designs is not yet known or officially available. Known are the converter switch frequency, it is a boost-type converter and the voltage and power levels of the in- and output. A boost converter, applying the basic topology shown in fig. 3.5b, has a more clean input due to the input inductor which ensures a continuous input current. Other boost-converter topologies might have more noisy inputs.

### 3.2.1.1 Switching Fundamental Noise and Harmonics

One critical noise source is the switching inside the SMPS since this noise is continuous and the frequency close to the signal frequency. If the ToI detector is not designed to cope with this noise, it will be useless. The fundamental switching frequency is $300 \text{kHz}$ (TBC) with smaller superimposed higher frequency harmonics. The current signal, of the input switching ripple, has a triangular waveform which can be represented by a Fourier-series of sine-waves as

$$i_{\text{input ripple}}(t) = \frac{8}{\pi^2} \sum_{n=1,3,5,...}^{\infty} \frac{(-1)^{n+1}}{n^2} \sin \left( n \frac{2\pi}{f_{\text{switch}}} t \right)$$

where $f_{\text{switch}}$ is the converter switching frequency and $n$ the odd harmonics number. In an attempt to somewhat quantify the noise level, the solar panel input current is first estimated. From [17], each panel will deliver a maximum of $7.626 \text{W}$. Assuming one satellite side with perpendicular solar incidence, two sides with $45^\circ$ incidence and using the Kelly Cosine[31] taking into account the efficiency loss from solar incidence, the input power and corresponding current is calculated to

$$P_{\text{in}} = 2 \cdot 7.6 \text{W} + 4 \cdot 7.6 \text{W} \cdot 0.7 = 37 \text{W}$$

$$I_{\text{in}} = \frac{P_{\text{in}}}{V_{\text{panel}}} = \frac{37 \text{W}}{15.4 \text{V}} = 2.4 \text{A}$$

(3.9)

A rule-of-thumb states that the current ripple of a power converter should be limited to around 10\% of the input current, thus 0.24 A ripple current could be expected. Figure 3.6 shows the estimated fourier components from equation (3.8). As seen, the noise current quickly drops at higher harmonics. This fortunately makes it more possible to detect pulse signals in the presence of noise from higher harmonics at the same frequency.
3. DESIGN

Morten Olsen
Thesis

EPS
SOLID
Subsystem
18 V Mainbus
Filter

Solar Panel Noise
- Eclipse Transient: 0 - 15 V<sub>DC</sub>, < 10 Hz
- ESD: High, High

Subsystems Noise
- Load switching: < 5 % of V<sub>MAINBUS</sub>
- Fuse blowing: < 10 % of V<sub>MAINBUS</sub>
- Current latches
- ...

Impact Pulse Signal
- 400 - 1000 mV<sub>P-P</sub>, 2 - 10 MHz

Mainbus Ripple
- < 0.5 % of V<sub>MAINBUS</sub>, 300 kHz

(a) Potential EPS noise sources that may disturb SOLID

(b) Simplified circuit diagram for dynamic signals in system of solar panel and EPS. Here a simple boost converter topology is assumed. It is currently unknown if an input filter is also applied.

Figure 3.5 – Characterization of input noise sources originating from the TechnoSat EPS
The input noise voltage is determined from the charging and discharging of the input ripple current through the input capacitance which in a simplified estimation is given as

\[ v_{\text{noise}}(t) = \frac{1}{C_{\text{input}}} \int_{0}^{t_{\text{switch}}} i_{\text{ripple}}(t) \, dt \]  

where \( t_{\text{switch}} \) is the converter switching time period (3.33\( \mu \)s). As seen, the amplitude of the input noise voltage is inverse proportional to the input capacitance, when assuming no input filter is applied. Unfortunately, the input capacitance depends heavily on the specific components used, solar cells, circuit layout, cabling etc. and is therefore poorly defined and any estimation of it’s value, at this time, might be orders of magnitude from the real value.

**Recommendation 9 (EPS noise)**  
Due to the very poorly defined EPS noise characteristics, it is strongly recommended to make thorough testing of the ToI detector electronics together with the TechnoSat EPS. Making accurate measurements of the input noise on the SMPS hardware is also highly recommended as this will help to design an optimal filter for the ToI detector electronics.

### 3.2.1.2 Noise Transients

Together with the continuous noise from the SMPS switching ripple, a number transient noise sources exists. These are briefly described below.

Switching of the power Metal-Oxide-Semiconductor Field-Effect Transistors (MOSFETs) inside the SMPS produces sharp transients at much higher frequencies\footnote{32}. This transient noise depends on parasitic components of the MOSFETs and is again very hard to predict. The transients

![Fourier series of estimated input current ripple to the EPS](image-url)
occur whenever the SMPS is switching, hence with the same rate as the fundamental switching frequency. The ToI detector noise susceptibility against these signals must therefore be completely eliminated.

Load switching on the mainbus could potentially also induce noise, however this noise is most likely filtered out, when going through the EPS, before reaching the ToI detector input.

When the satellite is entering/leaving an eclipse, the solar array voltage will have slow voltage transients. However, these transients are comparably slow and should be easy to filter out. ESDs on the solar arrays are likely much harder to completely filter out and may indeed cause false triggering of the ToI detector. These are discrete events and are expected to be comparatively rare. A limited amount of false pulse triggering can be tolerated, since subsequent detector line scans would reveal if it was a false trigger, i.e. no change in the detector lines is observed. A limit for the tolerated false trigger rate has not yet been established. There are two main concerns for false triggering:

- **Power consumption**
  If the ToI detector is triggered regularly, the low power advantage of SOLID is somewhat lost.

- **Lost information on real events**
  If the ToI detector is already triggered by a false event, it might miss a real event and lose the ToI data.

One definition of a limit could be to keep the false trigger rate less than a small fraction of the time it takes to scan the detector lines and reset the ToI detector circuit. The detector line scan time is still unknown but would not be expected to exceed more than a few seconds. A maximum false rate of one event per 5 min. could therefore be a suitable limit.

Another concern for solar array ESDs is damage to the ToI detector electronics. The TechnoSat solar arrays include electronics to suppress the high voltages from an ESD\[17\], thus the input to the ToI should be protected. One could further require that all electronic devices in the ToI detector, which has a direct path to the input, should have built-in or external ESD protection capability.

### 3.2.2 Input Filter

To allow the ToI detector to operate successfully in the noisy EPS environment, an input filter is placed on front of this circuit. This section describes the filter design process which can be summarized as:

- **Step 1** Set filter requirements
- **Step 2** Decide whether to use digital or analog filters
- **Step 3** Decide whether to use active or passive filters
3.2. Time-of-Impact Detector

Step 4 Set sequence of cascaded filters

Step 5 Set filter order and response types

Step 6 Calculate and choose filter components

Step 7 Manual Fine tuning of filter parameters

Below are described the considerations and outcome of each design step.

3.2.2.1 Input Filter Requirements

One design challenge is that neither the noise nor the signal of interest are well specified in frequency nor amplitude. This makes it difficult to determine the input filter design criteria. Table 3.1 lists some of the identified input filter requirements.

Figure 3.7 shows a simplified block diagram of the suggested input filter for the ToI detector. A low- and high-pass filter are combined to effectively produce a band-pass filter passing only the impact pulse signals. A single Operational Amplifier (OpAmp) bandpass filter can be created however, since a high bandwidth in the pass-band is required, using a cascaded low- and high-pass filter will yield better results\cite{33}. The notch-filter further suppresses the fundamental frequency of the SMPS. The notch filter is required due to the limitations of the band-pass filter to fully reject the 300kHz SMPS switching frequency due to its proximity to the impact pulse signal frequency. The low-pass filter at the comparator negative input generates the reference voltage for the ToI detector, by filtering out the impact pulse signal but maintaining the 1.5 V DC-level. A small DC-gain is added to set the trigger voltage threshold. This reference voltage signal could also have been derived directly from the 1.5 V DC-bias circuit however by deriving it from the filter output allows it to better track changes in the filter component parameters due to temperature, radiation etc.

![Simplified block diagram of ToI detector filters](image)

**Figure 3.7 – Simplified block diagram of ToI detector filters**

3.2.2.2 Digital or Analog Filter

A Digital Signal Processor (DSP) could be programmed as a digital filter running a Fast-Fourier Transform (FFT) algorithm. A Digital Signal Controller (DSC) combining both DSP and MCU in one device could also be used\cite{34}. A main advantage for SOLID is the flexibility of a digital filter allowing the filter type and parameters to be changed during the experiment flight in space.
### Table 3.1 – Input filter requirements for the ToI detector

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single supply filters</td>
<td>3V supply</td>
</tr>
<tr>
<td>Compatibility with MCU supply</td>
<td>50 mW per filter TBC</td>
</tr>
<tr>
<td>Low power consumption</td>
<td>320 mW TBC ≥ 1 TBB</td>
</tr>
<tr>
<td>Compact size</td>
<td>TBD</td>
</tr>
<tr>
<td>Avoid higher than 2nd order filters and advanced topologies, when possible</td>
<td>TBD</td>
</tr>
<tr>
<td>High selectivity (Q)</td>
<td>TBD</td>
</tr>
<tr>
<td>Pass impact pulse signals</td>
<td>f_c = 300 kHz</td>
</tr>
<tr>
<td>High frequency noise attenuation</td>
<td>f_c &gt; 20 MHz</td>
</tr>
<tr>
<td>Transient noise spikes attenuation</td>
<td>τ_pulse &lt; 10 ns</td>
</tr>
<tr>
<td>Immune to high input voltage</td>
<td>V_in max &lt; 17.5 V Solar array voltage</td>
</tr>
<tr>
<td>Active filter tuning (optional)</td>
<td>TBD</td>
</tr>
<tr>
<td>Due to poorly specified signal and noise parameters the required pass bandwidth might be higher than 10 MHz TBC</td>
<td></td>
</tr>
<tr>
<td>Power consumption requirements from Table 2.3, assuming eight ToI detectors</td>
<td>TBD</td>
</tr>
<tr>
<td>Single supply filters</td>
<td>TBD</td>
</tr>
<tr>
<td>TBD</td>
<td>TBD</td>
</tr>
</tbody>
</table>

Table 3.1 – Input filter requirements for the ToI detector
The main disadvantage is high power consumption. The required sampling frequency is twice the signal frequency, $f_{\text{sample}} \approx 2 \times 10\,\text{MHz}$, according to the Nyquist-Shannon sampling theorem. This requires a rather fast running DSC thus increasing power consumption. For example, the dsPIC30F6010 DSC from Microchip consumes 120 mA @ 3.3 V when operating at 20 Million Instructions Per Second (MIPS). Since the impact pulse signal only lasts a few periods, the benefits of a FFT are less significant and selecting an adequate "window-function" becomes difficult. A digital filter could also lead to aliasing problems if higher frequency noise is not sufficiently filtered out before the signal enters the Analog-to-Digital Converter (ADC). A digital filter solution is therefore discarded.

### 3.2.2.3 Active or Passive Filter

An active filter combines passive RC components and OpAmps. The advantage with active filters is that they can provide more than unity gain and high Q-values. They generally have high input impedance and low output impedance thus minimizing loading effects and making it easier to cascade filters. The disadvantage is power consumption, especially for high-speed operation, and the input signal range is limited to voltage levels within the supply range. They also introduce several non-ideal effects due to OpAmps physical limitations such as, limited bandwidths, bias current effects, temperature variations etc.

It was decided to implement the high frequency low-pass filter as a passive filter since PSPICE simulation showed it to be extremely difficult to implement this efficiently as an active filter. The reason is as follows; It is generally recommended to use OpAmps with bandwidths 10 to 20 times the signal frequency. This means at least 200 MHz bandwidth. OpAmps in this range have high power consumption. Furthermore, most high-frequency OpAmps have non-ideal gain peaking at their roll-off frequency meaning that high-frequency noise spikes may actually be amplified before being filtered.

The notch-filter is implemented as an active filter. It was initially intended to implement a passive notch filter to minimize components and power consumption. PSPICE simulations however showed unstable behaviour, when cascading the notch filter with the other filter sections. An output buffer solved the problem but effectively made it an active filter. The reason for this filter instability is not yet completely understood, or if it only originates from the behaviour of the simulation software.

The high-pass filter is implemented as an active filter. This allows setting of a higher Q-value (filter selectivity) in order to suppress the slightly lower-frequency SMPS noise. An active filter also allows gain to be added at the signal frequencies, making it easier for the ToI detector circuit to distinguish signal from noise.

The low-frequency low-pass filter for the reference voltage is also implemented as an active filter since a small DC-gain is needed, to set the threshold voltage.
3.2.2.4 Filter Sequence

As was discussed in section 3.2.2.3, OpAmp bandwidth limitations makes it difficult to correctly pass high frequency signals (and noise) which effectively distorts these and shifts them to a lower frequency thus making filtering of high frequency noise less efficient. The high-frequency passive low-pass filter is therefore placed first, to remove any high frequency noise, before feeding the input signal to the OpAmps. The passive filter is also insensitive to the high solar array DC voltage. The high-pass filter is then added to remove the high DC-level. This effectively works like the DC-blocking capacitor discussed earlier in section 3.2. A DC-bias circuit then sets a constant 1.5 V DC-level. The notch-filter is added last.

3.2.2.5 Filter Order and Response Types

The passive low-pass filter is implemented as a simple second-order Inductor-Capacitor (LC) filter. If more attenuation of high frequency noise is required, third order Π- and T-filters or even higher order filters can be applied. However, the main concern for high frequency noise are the MOSFET transients where higher order filters did not show noticeable improvement during PSICE simulations.

The high-pass filter is designed using the approach from [35]. A 2nd order Sallen-Key filter is selected due to its simplicity and low components counts. Filter types like Multiple-FeedBack or multi-state were discarded due to higher component counts. 2 dB-ripple Chebyshev filter response is selected, since fast roll-off is required whereas signal distortion from phase delay and amplitude ripples are of less significance. The Q-level is selected as a compromise between high filter selectivity and limitation of gain peaking at the filter cut-off frequency. The two extra capacitors, $C_8$ and $C_9$ in fig. 3.10, are added to roll-off gain at frequencies below 2 MHz and above 10 MHz respectively which also maintains the 1.5 V DC-level with desired unity gain.

The notch-filter is implemented as a 2nd order Twin-T filter using the approach from [36]. The notch filter is made relatively large bandwidth (low Q) to also provide some suppression of the higher harmonics but still without affecting the lowest signal frequencies around 2 MHz. If active filter tuning is required, a Fliege notch filter might be more suitable.

The active low-pass filter that generates the reference voltage, is implemented as a first order RC active filter. The cut-off frequency is selected well below the switching noise and signal frequencies.

3.2.2.6 Filter Components Selection

The required component values of the resistors and capacitors are calculated using a simple Maple script listed in appendix G. Based on; filter type, filter response type, cut-off frequency, Q-value and gain, the script calculates required components values. The script produces bodeplots of the
### 3.2. Time-of-Impact Detector

#### Table 3.2 – Input filter parameters for the ToI detector

<table>
<thead>
<tr>
<th>Filter: Response</th>
<th>Type</th>
<th>Order</th>
<th>Gain</th>
<th>$f_c$</th>
<th>$f_{norm}$/f$_{BW}$</th>
<th>$f_{aw}$</th>
<th>f$_{BW}$</th>
<th>Q</th>
</tr>
</thead>
<tbody>
<tr>
<td>High-pass</td>
<td>Chebyshev</td>
<td>2nd order</td>
<td>2.114</td>
<td>2 MHz</td>
<td>1.103</td>
<td>25 kHz</td>
<td>300 kHz</td>
<td>0.5</td>
</tr>
<tr>
<td>Low-pass(input)</td>
<td>Sallen-Key active</td>
<td>1st order</td>
<td>1.075</td>
<td>16 MHz</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Low-pass(ref. voltage)</td>
<td>2nd order</td>
<td>1</td>
<td>2 MHz</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Notch</td>
<td>Twin-T active</td>
<td>1</td>
<td>30 kHz</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

a) Higher than required 10 MHz, due to practical component values

b) Depends on input source resistance
Figure 3.8 – Bodeplot of input filter for the ToI detector. Phase margin is around 30° and gain margin infinite since phase lag never exceeds 180° (theoretically).
3.2. Time-of-Impact Detector

Filter, as shown in fig. 3.8 and the filter transient responses, shown in fig. 3.9. The complete ToI detector circuit, including input filter, is shown in fig. 3.10.

For accurate filter response, it is important that the filter component values are precisely trimmed/known and that the drift of over time and temperature is limited. Therefore, quality capacitors like NP0/C0G shall be used and good tolerance resistors.

3.2.2.7 Manual Fine-tuning of Filter

It is often not practically possible to obtain the exact required component values, calculated in section 3.2.2.6. Furthermore, parasitic capacitance, series resistance and OpAmp imperfections also cause the real filter to deviate slightly from its theoretical predictions. Some manual trimming of component values is therefore usually required. Performing the manual fine-tuning does not make sense until the filter requirements from table 3.1 are more precisely defined.

To validate the constructed filter, a network analyzer can be used to measure the frequency and phase response, and its transient behaviour by applying step input and measuring the overshoot and settling time of the output, with an oscilloscope.

3.2.2.8 Active Filter Tuning

In addition to the manual fine-tuning discussed in section 3.2.2.7, it might also be considered to add active tuning capability into the input filter. This would allow the filter specifications to be somewhat optimized once the satellite has been launched into space, as an assurance against the poorly known noise specifications. Active tuning of analog filters can be achieved by replacing
3. DESIGN

Morten Olsen
Thesis

RLC low-pass filter
High-pass filter
Notch filter
Reference voltage filter
Comparator
DC-bias circuit

LMV641

Figure 3.10 – Schematic of ToI detector circuit
certain filter resistors with digital potentiometers. A suggested part is listed in appendix E. The digital potentiometer is controlled from the MCU. Three particular places have been identified, where active filter tuning could be suitable:

- **Notch filter tuning**
  For the notch filter to be efficient, it is important that the center frequency is very close to the SMPS fundamental frequency. The SMPS frequency might vary in the order of \(\pm 10\%\)\(^{[37]}\), due to variations in the associated Pulse-Width-Modulation (PWM)-controller. The notch filter parameters may also drift slightly, due to component changes over time/temperature. In fig. 3.8, if the notch filter is 10\% misaligned, it provides about 15 dB less attenuation which might be unacceptable. The Twin-T notch filter currently used is not very suitable for active tuning, since three resistor values would have to be adjusted. Instead, a Fliege topology could be used where the center frequency can be controlled by a single variable resistor.

- **High-pass filter cut-off frequency tuning**
  Using one digital potentiometer, the cut-off frequency of high-pass filter can be adjusted. In the Sallen-Key topology, the Q-value also changes when adjusting the frequency. If this cannot be tolerated, a second digital potentiometer will be needed which somewhat increases the circuit complexity. The high-pass multi-feedback topology allows frequency control, with constant Q-value, using only one potentiometer, however at least one source does not recommend this topology for high-pass filtering\(^{[38]}\).

- **Threshold voltage setting of comparator reference voltage**
  The threshold voltage of the comparator can be controlled by replacing resistor \(R_{14}\) in fig. 3.10 with a digital potentiometer. This effectively just varies the gain of the OpAmp as shown in equation (3.11). Figure 3.11 shows the possible setting of the threshold voltage which is achievable with the selected components.

\[
V_{\text{threshold}} = V_{DC} \frac{R_{13}}{R_{13} + R_{\text{POT}}} = 2.5\, \text{V} \frac{1.5\, \text{k}\Omega}{1.5\, \text{k}\Omega + R_{\text{POT}}} \quad (3.11)
\]

where \(R_{\text{POT}}\) is the resistance of the digital potentiometer.

Including active filter tuning is a trade-off between increased circuit complexity and input filter performance degradation from component variations.

**Recommendation 10** After passing the input signals through the input filter, a suitable noise margin should exist, which also takes into account some filter degradations. If this cannot be sufficiently achieved, it is recommended to include some degree of active filter tuning capability.
3.2.3 Short Circuit Protection

As was seen in fig. 3.5a, the input to the ToI detector is placed somewhere between the solar array output and the EPS input. Depending on the final system configuration, as will be discussed in section 3.5, this connection is a potential single-point failure\cite{20, 5.7.2a}. A Short-Circuit (SC) failure to ground, at this point, will cause complete failure of the solar array or the entire EPS input, depending on the exact location of the input connection point. To mitigate this risk, a SC-protection circuit is proposed. The SC-protection circuit comprises a current sensor, logic circuitry and a switch. The circuit must be reliable, compact and low power. A further challenge is that the input voltage will exceed the $3.3\, \text{V}$ logic supply. This makes it more difficult to operate the switch which normally requires a control voltage at the same level as the input voltage. Two possible solutions to the switch challenge are considered:

- **High-side MOSFET switch**
  High-side MOSFET switches are usually applied in automotive application to protect loads from SC failures, using a current limiting circuit. Some parts provide a current sense telemetry output. The MOSFET control voltage is generated via a charge-pump. The advantage is that everything is integrated into a single IC and additional protection against ESDs is provided. The disadvantage is that the SC-protection only limits the current but does not completely open the MOSFET switch, when a failure is detected. It is also a more complicated circuit with many unknown failure modes and degradation effects.

- **Optocoupler**
  Optocouplers are basically a MOSFET switch with a tiny solar cell, controlled by a small Light-Emitting Diode (LED). This allows control of the switch, independent from the input voltage level. The input current is measured using an external current sense resistor, and a comparator is configured to trigger at a predefined current level which in turn controls
3.2. Time-of-Impact Detector

The advantage is that the circuit behaviour and failure modes are well understood and monitoring/control from MCU is easily implemented. The disadvantages are higher power consumption, due to the operation of the LED and larger PCB footprint due to several external parts and the optocoupler itself which is a comparably large part.

The optocoupler solution is finally chosen, also due to the difficulty of finding a suitable high-side MOSFET switch. The SC-protection circuit is shown in fig. 3.12. Component value calculations and PSPICE simulations are listed in appendix C.1.

The SC-protection circuit works as follows. Top-comparator, of $U_6$, sees the differential voltage across current sense resistor, $R_{30}$. The input voltages are scaled down through two voltage dividers; $R_{18}$, $R_{19}$ and $R_{20}$, $R_{21}$. Capacitors $C_1$ and $C_{19}$ form RC-filters, together with $R_{18}$ and $R_{20}$ respectively, with cut-off frequency around 1 kHz. This removes high frequency noise and sets the reaction time of the SC triggering to around 1 ms. The value of the current sense resistor is chosen relatively large to add significant gain to the current measurement. To avoid the current sense resistor affecting the behaviour of the ToI detector input filter, capacitor $C_{20}$ is added, to bypass any high frequency signal above $\approx 5$ kHz.

Resistor $R_{22}$ adds hysteresis to the comparator which effectively sets the differential voltage at which the SC-protection circuit triggers. The lower ends of resistors $R_{19}$ and $R_{21}$ connects to a 1.5 V bias voltage. This avoids incorrect resetting after triggering the SC-protection circuit. If not there, when switch $S_3$ opens, both comparator inputs would go to zero which would ignore the hysteresis and the comparator output will return to its initial state (high output).

When the SC-protection is triggered, the circuit automatically latches and a telemetry output pulled low, to indicate a fault condition. The circuit is reset by pulling the telemetry signal high via the MCU.

### 3.2.4 Sunlight Detector

It is currently unknown if disturbance signals from impacts are generated on solar arrays while in eclipse or in shadow on the backside of the satellite. If not generated, these detector boards must rely on frequent scans of the detector lines. A sunlight detector circuit is then needed to determine if a detector board is in eclipse/shadow and set the monitoring method: ToI detection or frequent detector line scanning. The requirements and options for the sunlight detector further depend on the system configuration, as discussed in section 3.5.

Assuming the solar array voltage is always significantly lower when in eclipse/shadow and that the ToI detector has direct connection to the solar array output. The output can then easily be used to determine the sunlight status. This is actually already included in the SC-protection circuit in fig. 3.12 as the second comparator. The solar array voltage is scaled down by the voltage divider and the reference voltage is generated from the 1.5 V bias and divided through resistors $R_{24}$ and $R_{25}$. The comparator will indicate a sun lit solar array when the voltage
Figure 3.12 – EAGLE schematic of the SC protection circuit.
If only a single ToI detector is used or the solar array voltage cannot easily be characterized during eclipse/shadow, two alternative ways to detect presence of sunlight have been considered:

- **Temperature sensor**
  Sunlit detector boards are warmer than when in shadow. This can be measured with a thermistor and bias circuit. The temperature transients might be relatively slow, since heating of the detector board takes time. This is problematic if the satellite is rotating or otherwise have frequently varying sunlight conditions. A differentiator circuit could improve this, but adds complexity. This solution would also depend on the satellite thermal design, to correctly set the threshold temperatures.

- **Photodiode sensor**
  A photodiode is in effect a tiny solar cell. When placed in sunlight and connected to a high impedance load, it provides an open-circuit voltage of a few hundred mV which is measured by a comparator.

The photodiode solution is recommended due to its relative simplicity, independence of the satellite design and fast response time. A suggested circuit is shown in fig. 3.13. A reference voltage is generated from the 3.3 V supply and compared to the photodiode output voltage using a comparator. Suggested circuit parts can be found in appendix E. The prototype MUX board, shown in fig. D.2 already includes a PCB layout to mount and test the sun detector circuit. Switching between ToI detection and frequent scanning should automatically be decided by the SOLID MCU software. To reduce power consumption, the ToI detectors on shaded detector boards should also automatically be switched off.

Finally, if the above solutions are not possible or pose too many requirements on the system, i.e. power or PCB space for components, SOLID might have to depend on information from
the TechnoSat On-Board Computer (OBC) and ACS for orbit location and satellite attitude to determine which detector boards can be monitored using ToI detectors.

The SOLID concept is mainly intended for larger satellites with large wing-mounted solar arrays. These satellites usually use drive mechanisms to ensure that solar arrays always point towards the sun, except during eclipses. For these applications, sunlight detection circuitry might not be required by SOLID.

3.3 Microcontroller

This section describes the MCU and communication hardware selection. The MCU is the "brain" of SOLID and controls the communication interface to the TechnoSat data CAN-bus.

3.3.1 Microcontroller vs. FPGA

A trade-off was considered between using Field Programmable Gate Array (FPGA) or MCU. This is a well debated topic on various online electronics communities. FPGAs are often chosen for applications requiring large amounts of arithmetic calculations, for example image analysis or handling large amounts of data in time critical applications. FPGAs can be programmed with Intellectual Property (IP)-cores emulating a Central Processing Unit (CPU) similar to that of MCUs. For example, Micorsemi offer a free ARM Cortex-M1 IP-core\[39\]. For TechnoSat, a CAN-controller IP-core would be needed. The CAN protocol is governed by Bosch Gmbh patents and only available through various license agreements\[40\]. ESA also offers a CAN IP-core through their own license agreement, but only for projects realized as a part of an ESA-programme\[41\]. Many MCUs feature built-in CAN-controllers thus avoiding extra license fees. Using an FPGA as CPU will therefore have some limitations when a CAN-controller is required. An FPGA could also be considered for the MUXs, to offer a more compact and flexible solution. One important consideration, for the trade-off, is the FPGA power consumption. A method to estimate the power consumption is provided in \[42, p. 2-12\]. A quick summary of some FPGA features is listed below:

- Less risk of part obsolescence - FPGAs can be updated with new IP-cores.
- Flexible and adaptable pin layout - each FPGA pin function is customizable which might be advantageous in compact designs.
- More compact design by integrating MUXs and other simple circuitry into the FPGA.
- Parallel operation for time-critical or arithmetically heavy applications.
- FPGAs, in general, are said to have higher power consumption than similar MCU solution, except for applications requiring a very large amount of simple operations.
• Complex programs, algorithms and conditional operation is more difficult to implement in FPGAs unless a soft IP-core processor is included.

• Longer development times and considerable testing efforts are required with FPGAs.

Frequently scanning thousands of detector lines seems like an obvious FPGA task. The SOLID concept should, however, ideally not rely on frequent detector line scanning, but rather depend on ToI detectors. Considering the above stated features and the authors limited experience with FPGAs, an MCU solution has been chosen.

### 3.3.2 Microcontroller Requirements

A main MCU requirement is a large number of General Purpose Input/Outputs (GPIOs), in order to control and read out the MUXs, ToI detectors, sun detectors and SC- protection circuits. In table 3.3 the MCU pin requirements are listed along with their related functions. The number of required pins also depends on the number of detector boards and the SOLID system layout, as will be discussed in section 3.5. The two different pin count numbers represent the dependence on system configuration and number of detector boards.

<table>
<thead>
<tr>
<th>No. pins</th>
<th>Function</th>
<th>Direction</th>
<th>Config.</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>0/8</td>
<td>Sun detector status</td>
<td>Input</td>
<td>none</td>
<td>Digital</td>
</tr>
<tr>
<td>1</td>
<td>ToI detector status</td>
<td>Input</td>
<td>PD</td>
<td>Digital</td>
</tr>
<tr>
<td>1/8</td>
<td>ToI detector shutdown</td>
<td>Output</td>
<td>PP</td>
<td>Digital</td>
</tr>
<tr>
<td>1</td>
<td>ToI detector reset</td>
<td>Output</td>
<td>PP</td>
<td>Digital</td>
</tr>
<tr>
<td>1/8</td>
<td>SC protection status/control</td>
<td>Input/output</td>
<td>none/PP</td>
<td>Digital</td>
</tr>
<tr>
<td>16</td>
<td>MUX readout for x and y</td>
<td>Input</td>
<td>PD</td>
<td>Digital</td>
</tr>
<tr>
<td>1</td>
<td>MUX enable/disable</td>
<td>Output</td>
<td>PP</td>
<td>Digital</td>
</tr>
<tr>
<td>1</td>
<td>MUX write enable/disable</td>
<td>Output</td>
<td>PP</td>
<td>Digital</td>
</tr>
<tr>
<td>5</td>
<td>Output-MUX x,y address control</td>
<td>Output</td>
<td>PP</td>
<td>Digital</td>
</tr>
<tr>
<td>5</td>
<td>Input-MUX x,y address control</td>
<td>Output</td>
<td>PP</td>
<td>Digital</td>
</tr>
<tr>
<td>1-2</td>
<td>Filter tuning CS pin</td>
<td>Output</td>
<td>PP</td>
<td>Digital</td>
</tr>
<tr>
<td>1-2</td>
<td>Filter tuning Up/Down pin</td>
<td>Output</td>
<td>PP</td>
<td>Digital</td>
</tr>
<tr>
<td>2</td>
<td>Redundant CAN-bus (CANH, CANL)</td>
<td>Input/output</td>
<td>none/PP</td>
<td>Digital</td>
</tr>
<tr>
<td>2</td>
<td>UART(TX, RX) - testing purpose</td>
<td>Input/output</td>
<td>PU/PP</td>
<td>Digital</td>
</tr>
</tbody>
</table>

Table 3.3 – MCU pin numbers and functions for eight SOLID detector boards (directions are seen from the MCU perspective).

\[a\] To Be Decided. Dependant on the SOLID system configuration.

\[b\] PD = Pull-Down resistor on pin input
PU = Pull-Up resistor on pin input
PP = Push-Pull output
To reduce the number of required pins for ToI detectors, each output can be fitted with a diode thus creating a wired OR-gate in combination with the MCU input pull-down resistor. If any of the eight ToI detectors get triggered, the MCU input goes high and the sequent detector line scan determines which detector board was impacted, i.e. where a change has occurred.

Below are listed the identified requirements for the MCU. The functions of the MCU are relatively simple, thus the requirements are moderate.

<table>
<thead>
<tr>
<th>Microcontroller Unit requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>MCU 1   Opt: 8 bit architecture</td>
</tr>
<tr>
<td>MCU 2   Opt: 1 ADC(≥ 10 bits)</td>
</tr>
<tr>
<td>MCU 3   Req: 1 timer</td>
</tr>
<tr>
<td>MCU 4   Opt: 1 Universal Asynchronous Receiver/Transmitter (UART)</td>
</tr>
<tr>
<td>MCU 5   Req: 1 real-time clock</td>
</tr>
<tr>
<td>MCU 6   Opt: Digital high/low input detection with interrupt</td>
</tr>
<tr>
<td>MCU 7   Req: WatchDog Timer (WDT)</td>
</tr>
<tr>
<td>MCU 8   Req: Low power consumption</td>
</tr>
<tr>
<td>MCU 9   Req: &gt; 62 GPIOs</td>
</tr>
<tr>
<td>MCU 10  Req: Embedded CAN controller</td>
</tr>
<tr>
<td>MCU 11  Req: ≥ 16 kB program memory</td>
</tr>
<tr>
<td>MCU 12  Req: ≥ 1 kB data memory</td>
</tr>
<tr>
<td>MCU 13  Req: 5 V or 3.3 V supply voltages</td>
</tr>
</tbody>
</table>

*Req:* Required  
*Opt:* Optional

### 3.3.2.1 Radiation Effects

Flash NAND technology is more susceptible to space radiation than NOR. Possible reduction of TID degradation include not biasing (powering off) the device when not in use. Also, Cyclic Redundancy Check (CRC) can be applied to improve storage integrity. The STM32F427, listed in appendix E, features a power-saving mode with Flash power down, which might reduce unwanted radiation effects.

### 3.3.3 Prototype MCU Board

The development of a dedicated MCU board is currently planned. For initial hardware tests, it was decided to use the STM32F4-Discovery development board, shown in fig. 3.14. The board includes a build-in ST-LINK-V2 interface for programming and debugging which may also be used for external devices, via the SWD connectors. One limitation of this board is that many of the pins on the STM32F407VGT6 MCU are already connected to external modules such as
push-buttons, LEDs, clocks etc. This greatly limits the flexibility and number of available pins. This board is therefore not suitable for full scale SOLID tests with all MUXs, ToI detectors etc.

![STM32-Discovery development board](Source: [43])

Figure 3.14 – STM32-Discovery development board

3.4 Software

This section describes the planning and implementation of the SOLID software.

3.4.1 Software Tools

A number of different programming languages and software development tools have been considered for SOLID. These are briefly discussed in the following sections.

3.4.1.1 Programming Language

Choosing between different programming languages for embedded projects is a topic widely discussed on online fora[44], [45]. Three programming languages are considered for the SOLID software:

- **C++**
  An object-oriented programming language usually applied in larger and more complex projects.

- **C**
  Often preferred for smaller embedded projects where code size and speed optimizations are important. C somewhat allows code to be migrated to new hardware. C code is relatively easy and fast to write.
3. DESIGN

Morten Olsen
Thesis

- **Assembler**
  Allows very fast and optimized code to be written. However, using a good C compiler can often provide comparably efficient code. Assembler code is more complex and time-consuming to write. Assembler is also very hardware specific and therefore does not allow code migration to other hardware devices.

In addition to the above statements, the author is more experienced in programming C rather than C++ or assembler. C is therefore chosen as the programming language for **SOLID**.

### 3.4.1.2 Software Toolchain

It has been considered whether to use a commercial or open-source toolchain. Commercial toolchains have the following advantages:

- Ready-to-use: minimum resources on Integrated Development Environment (IDE) setup,
- Well proven: Commercial vendors depend on selling reliable software,
- Customer support (for licensed users).

The main drawback to utilising a commercial toolchain is the cost. However, considering the relatively short project schedule, the author’s limited expertise in setting up toolchains and to minimize project risks due to unforeseen toolchain issues, a commercial product is preferred. The MDK-Lite from Keil offers a free license for programs up to 32 kB which can later be upgraded to a full license, should a larger program or support be needed. A license quote from Keil has already been received. A possible future task could be to set up a reliable open-source toolchain for the code development - current DLR employees have already done so. Table 3.4 lists the chosen software development tools.

<table>
<thead>
<tr>
<th>Tool</th>
<th>Choice</th>
<th>Manufacturer</th>
</tr>
</thead>
<tbody>
<tr>
<td>IDE</td>
<td>MDK-Lite(^a)</td>
<td>Keil</td>
</tr>
<tr>
<td>Compiler</td>
<td>armcc</td>
<td></td>
</tr>
<tr>
<td>Development board</td>
<td>STM32F4-Discovery</td>
<td>STMicroelectronics</td>
</tr>
<tr>
<td>Programming/debugging</td>
<td>ST-LINK/V2</td>
<td>STMicroelectronics</td>
</tr>
<tr>
<td>Language</td>
<td>C</td>
<td></td>
</tr>
<tr>
<td>SubVersion (SVN) software</td>
<td>TortoiseSVN(^b)</td>
<td></td>
</tr>
<tr>
<td>PC Test Software</td>
<td>HTerm</td>
<td></td>
</tr>
<tr>
<td>UART to USB connector</td>
<td>TTL-232RG-VSW3V3-WE</td>
<td>FTDIChip</td>
</tr>
</tbody>
</table>

\(^a\) 32 kB free license

\(^b\) Not yet implemented
3.4.2 Software Requirements

The MCU software task are to control and read out detector lines via the MUXs; to control the ToI detectors, sun detectors and SC-protection circuits, and to handle communication over the CAN-bus. Below is shown a list of identified software requirements.

<table>
<thead>
<tr>
<th>Software requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>General</strong></td>
</tr>
<tr>
<td>SOF 1 Req: Modular code for easy modification or migration to different hardware</td>
</tr>
<tr>
<td>SOF 2 Opt: Software in-flight update capability</td>
</tr>
<tr>
<td><strong>Communication and data handling</strong></td>
</tr>
<tr>
<td>SOF 3 Req: Receive and transmit messages on redundant CAN-bus</td>
</tr>
<tr>
<td>SOF 4 Req: MCU wakeup when receiving CAN messages</td>
</tr>
<tr>
<td>SOF 5 Req: Send status message upon power-ON</td>
</tr>
<tr>
<td>SOF 6 Req: Store impact data</td>
</tr>
<tr>
<td>SOF 7 Req: Only send data when new impacts are detected</td>
</tr>
<tr>
<td>SOF 8 Req: Allow manual data requests</td>
</tr>
<tr>
<td>SOF 9 Req: Support debugging via RS232 interface</td>
</tr>
<tr>
<td>SOF 10 Req: Send acknowledgement after receiving and executing commands</td>
</tr>
<tr>
<td><strong>Timing</strong></td>
</tr>
<tr>
<td>SOF 11 Req: Interrupt on output from ToI detectors</td>
</tr>
<tr>
<td>SOF 12 Req: Provide real-time timestamps for ToI detection</td>
</tr>
<tr>
<td>SOF 13 Req: Nested interrupts should be avoided and shall not cause error or data loss</td>
</tr>
<tr>
<td>SOF 14 Req: Real-time clock synchronization with TechnoSat OBC</td>
</tr>
<tr>
<td><strong>Error handling</strong></td>
</tr>
<tr>
<td>SOF 15 Req: Internal WDT for system latchup recovery</td>
</tr>
<tr>
<td>SOF 16 Req: Software shall run stably and autonomously for extended time periods</td>
</tr>
<tr>
<td>SOF 17 Opt: Software shall use Fault Detection Isolation and Recovery (FDIR)</td>
</tr>
<tr>
<td>SOF 18 Opt: Memory checks to correct SEUs from space radiation</td>
</tr>
<tr>
<td>SOF 19 Opt: Accidental short-circuit on two MCU neighbour pins shall not cause damage</td>
</tr>
<tr>
<td>SOF 20 Opt: Safe mode option</td>
</tr>
<tr>
<td><strong>Time-of-Impact</strong></td>
</tr>
<tr>
<td>SOF 21 Req: Automatic reset of ToI detectors</td>
</tr>
<tr>
<td>SOF 22 Opt: Automatic choose between ToI detector or frequent scanning, on detector boards, based on sun detector outputs</td>
</tr>
<tr>
<td>SOF 23 Opt: Configuration of time interval for frequent detector line scanning</td>
</tr>
<tr>
<td>SOF 24 Opt: Manual disable of ToI detector(s)</td>
</tr>
<tr>
<td>SOF 25 Req: Manual detector line scan request</td>
</tr>
</tbody>
</table>
3. DESIGN

**Power**

**SOF 26** *Req:* Enter sleep mode when MCU is idle

**SOF 27** *Opt:* Power off ToI detectors on shaded detector boards

**SOF 28** *Opt:* Control SC-protection circuit

**Multiplexers**

**SOF 29** *Req:* Digital high/low readout of MUXs - avoids time-consuming ADCs

**SOF 30** *Opt:* Manual MUX readout using ADC - for diagnostic purposes

**SOF 31** *Req:* Time delay between MUX switching and readout - removes transient effects

**SOF 32** *Req:* Power cycling of MUXs to clear SEL errors

*Req:* Required

*Opt:* Optional

### 3.4.3 Software Structure and Implementation

This section describes some considerations and arguments for the software structure. The complete source code is listed in appendix F.1. Software flowcharts are shown in figs. 3.15 to 3.19, where red boxes indicate that the functionality is not yet implemented.

#### 3.4.3.1 Action Flags

As mentioned in section 1.3, the first SOLID prototype had some software instability issues. One concern is execution of long interrupt routines. These routines can interrupt each other and the code can end up running several partially-completed interrupt routines simultaneously. If these routines access and modify the same global variables, system behaviour might be unpredictable and stability compromised. To avoid this, it is suggested to execute all larger interrupt routines, for example scanning of detector lines, in the main loop. This is achieved by letting the interrupt set an action flag and when the program returns to the main loop it checks pending actions flags and executes these. This way, only one detector line scan function or other large routine will be active at a time, thus minimizing the time spent in interrupt routines.

#### 3.4.3.2 Power Saving Modes

To reduce the MCU power consumption, three power saving modes have been implemented. Ordered from highest to lowest power consumption, these are: sleep-, stop- and standby-mode. The sleep-mode can be exited by any interrupt for example when receiving a CAN-message or when the ToI detector is triggered. The stop-mode can only be exited when an "external interrupt line" changes state and the standby-mode only on "RTC tamper events" which in both cases can be generated by the ToI detector output or SC-protection circuit state, but not

---

1See the STM32F4xxxx Reference manual, RM0090
3.4. Software

when receiving a CAN-message. Thus, if a timely acknowledgement is required, when receiving CAN-messages, only the sleep-mode can be used.

![Main loop software flowchart](image)

![Software flowchart of the interrupt functions](image)

3.4.4 Software Outsourcing

It is currently expected that implementation of SOLID flight-software will be outsourced to a partner institution. This relieves some of the work load at DLR in Bremen, but poses some project management challenges. Some suggestions on the outsourcing is given below:

- **Requirements Control**
  
  To successfully outsource any software project, it is very important to have very clear expectations and definitions of the software requirements and the interfaces between software and hardware. Due to the typical nature of space projects, the software requirements are likely to change, as the project matures. There should therefore be an agreement on how to control changes in software requirements and interfaces[46, Sec. 8.5.2].

- **SubVersion Control System**
  
  A software SVN system should be used. This will make the source code available both to developers and DLR. This allows code development to be continuously tracked and the newest code is always available for testing with the hardware at DLR. A free SVN software was suggested in table 3.4.
Figure 3.17 – **Software flowchart for the initialization function**

**Action: Periodic detector line scan**
- Determine panels in shade or eclipse
- Read MUXs
- Get current time
- Prepare data packet
- CAN send data packet
- Update impact data
- Clear action flag
- Return

**Action: Impact pulse detected**
- Read MUXs
- Determine new impact?
- Yes
  - Prepare data packet
  - CAN send data packet
  - Update impact data
  - Clear action flag
  - Return
- No
  - PreVIOUS IMPACT DATA
  - new impact?
  - Yes
    - Prepare data packet
    - CAN send data packet
    - Update impact data
    - Clear action flag
    - Return
  - No
    - Reset pulse detector
    - Clear action flag
    - Return

Figure 3.18 – **Software flowchart of for periodic scan function(left image) and ToI detection function(right image)**
Figure 3.19 – Software flowchart of functions for sending/reading message over UART (figure a and b) or CAN-bus (figure c and d)
3. DESIGN

• Source Code Documentation
To make source code functionality as transparent as possible, good documentation standards should be applied. The documentation of the current source code uses Doxygen\cite{47} software and standards. This free software can also generate html or LaTex outputs of the documentation allowing a fast an easy overview of the source code.

Recommendation 11 (Software Requirements) It is recommended that the software requirements, listed in section 3.4.2, are discussed with the DLR project responsible, the external developer and the TU-Berlin, to decide on missing or unnecessary requirements

3.5 System Layout

The system layout of the SOLID embedded electronics determine the number of required PCBs, the location of these inside TechnoSat and their interfaces to other PCBs and TechnoSat subsystems. From table 2.3, eight detector boards are expected. To integrate these with the remaining SOLID electronics, a number of trade-offs are considered:

• Number of ToI detectors
Two options are considered: one central ToI detector on the MCU board, shown in fig. 3.20, or one for each detector board. Having just one central ToI detector would greatly reduce the system power consumption. Noise issues will, however, be even greater, since signals have to travel through longer cables that might pick up noise. It has also been a concern, that taking the signal from the output of the parallel connection of all the solar arrays would reduce the signal strength in contrast to taking the output directly at each individual solar array output (individual solar array outputs are isolated from each other using reverse blocking diodes, as described in \cite{17}).

• Location of MUXs
The MUXs can either be located on the MCU board, internally on a separate MUX board or externally outside on the detector board/separate MUX board. These options are illustrated in fig. 3.20.

Placing the MUXs on the MCU board has been discarded, due to the many connections required between MUXs and the detector lines which in this case would lead to many long and bulky cables. Discarding the MCU board by placing MCUs on each detector board has also been rejected due to increased power consumption, more components and lack of significant advantages. A number of potential system layout configurations are identified and shown in fig. 3.21. These are briefly described below:

  Config 1: One central ToI detector located on the MCU board and MUXs placed externally together with the detector board.

  Config 2: MUXs located internally but on a separate MUX board.
3.5. System Layout

Figure 3.20 – *SOLID* system mounted inside satellite structure

<table>
<thead>
<tr>
<th>Solar panel (external)</th>
<th>Satellite side panel (internal)</th>
<th>Electronics box (internal)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Config 1:</strong> Central ToI detector</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Detector lines Sun detector MUX</td>
<td>MUX ToI detector SC-protection</td>
<td>MCU CAN controller</td>
</tr>
<tr>
<td>~16 wires</td>
<td>~26 wires</td>
<td></td>
</tr>
</tbody>
</table>

| **Config 2:** MUX internal |
| Detector lines Sun detector | MUX ToI detector SC-protection | MCU CAN controller |
| ~73 wires | ~26 wires |

| **Config 3:** MUX external |
| Detector lines Sun detector MUX | Tol detector SC-protection | MCU CAN controller |
| ~16 wires | ~26 wires |

Figure 3.21 – Possible system layout configurations for *SOLID*
Config 3: MUXs located externally on or next to the detector board.

To quantitatively evaluate each system configuration, an Excel workbook has been created which contains specifications for all suggested parts and calculates the resulting budget requirements. The results are listed in table 3.5. The outcome heavily depends on the parts selected, the number of detector panels. Furthermore, the mass estimate does not consider the detector lines, kapton layers nor mechanical support parts, such as glue, bolts, screws etc., costs do not consider spare parts nor development models and no design margins are included in these values. Table 3.5 – Quantitative trade-off analysis of system configuration (assuming: eight detector boards, 3.3 V supply voltage, PCB area five times greater than the combined parts PCB footprint, 10 cm cable length between MCU- and MUX board, 60€/hour labour cost for PCB manufacturing + components soldering and an average of one minute spent on soldering small components)

<table>
<thead>
<tr>
<th>Config</th>
<th>Mass (g)</th>
<th>APcb (cm$^2$)</th>
<th>hmax (mm)</th>
<th>Pmax (mW)</th>
<th>Ptyp (mW)</th>
<th>Pstdb. (mW)</th>
<th>Cost(parts) (K€)</th>
<th>Cost(labour) (K€)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>175</td>
<td>130</td>
<td>6.8</td>
<td>175</td>
<td>48</td>
<td>14</td>
<td>2.3</td>
<td>16.1</td>
</tr>
<tr>
<td>2</td>
<td>440</td>
<td>535</td>
<td>6.8</td>
<td>401</td>
<td>187</td>
<td>40</td>
<td>3.1</td>
<td>16.6</td>
</tr>
<tr>
<td>3</td>
<td>364</td>
<td>405</td>
<td>6.8</td>
<td>401</td>
<td>187</td>
<td>40</td>
<td>3.1</td>
<td>16.6</td>
</tr>
</tbody>
</table>

* Without cable plugs connected to PCB sockets

Table 3.5 shows a qualitative trade-off analysis between the three system configurations. Power consumption is well below the design limit, even in worst-case scenario. Config 2 is currently the preferred configuration. However, the estimated system mass around 440 g is very close to the mass limit from table 2.3, when considering that several parts are not included in this calculation. The future design driver should therefore be to limit or decrease the system mass, if the allocated mass budget is not to be exceeded. The main mass contributors, according to these calculations, are the PCBs and harness. Thus, efforts should be put in to making very compact PCB layouts, use parts with small PCB footprint and use very lightweight cables and connectors.

Table 3.6 – Qualitative trade-off analysis of system configuration

<table>
<thead>
<tr>
<th>Config</th>
<th>Advantages</th>
<th>Disadvantages</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Low power, low mass</td>
<td>Noise prone, poor signal quality</td>
</tr>
<tr>
<td>2</td>
<td>More redundant</td>
<td>Highest mass, high power consumption, many connections required through satellite side panels</td>
</tr>
<tr>
<td>3</td>
<td>More redundant</td>
<td>High mass, high power consumption, more risk of radiation damage and temperature stresses on MUXs</td>
</tr>
</tbody>
</table>
3.5.1 Internal and External Interfaces

Figure 3.22 shows a simplified block diagram of the SOLID internal and external electrical interfaces. Table 3.7 lists the identified internal and external interfaces. The selected parts/materials are specified in Table E.1 in appendix E. Many of the interfaces still remain to be identified and defined.

Table 3.7 – SOLID internal and external interfaces

<table>
<thead>
<tr>
<th>Interface</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Solar cells to detector board</td>
<td>Mechanical</td>
<td>TBD</td>
</tr>
<tr>
<td>MCU board to TechnoSat electronics box</td>
<td>Mechanical</td>
<td>TBD</td>
</tr>
<tr>
<td>Detector board to TechnoSat side panel</td>
<td>Mechanical</td>
<td>TBD</td>
</tr>
<tr>
<td>Power from TechnoSat bus</td>
<td>Electrical</td>
<td>Bus connector TBC</td>
</tr>
<tr>
<td>Communication with TechnoSat bus</td>
<td>Electrical</td>
<td>CAN-bus via bus connector TBC</td>
</tr>
<tr>
<td>Power and communication to MUX boards</td>
<td>Electrical</td>
<td>Cable and PCB connectors</td>
</tr>
<tr>
<td>Power and communication to detector boards</td>
<td>Electrical</td>
<td>Connector/cable through satellite side panel TBD</td>
</tr>
<tr>
<td>ToI detector input from TechnoSat EPS</td>
<td>Electrical</td>
<td>Cables from solar array outputs TBD</td>
</tr>
</tbody>
</table>
Figure 3.22 – SOLID system block diagram

- **8 x Detector board**
- **8 x MUX board**
- **TBD**
- **Detector Lines**
- **MCU board**
- **MCU**
- **CAN Transceiver**
- **RS232** (testing purpose)
- **Solar Arrays Voltages**
- **Short Circuit Protection**
- **3 x MUX**
- **CAN** (redundant)
- **3.3V**
- **Sun detector**
- **Switch**
- **CAN Transceiver**
- **MCU board**
- **3.3V Supply**
- **Short Circuit Protection**

---

3. DESIGN

Morten Olsen

Thesis
3.6 Integration of Attitude Disturbance Data

So far, no work has been undertaken on how to integrate the use of attitude data together with SOLID. This section briefly discusses some of the design considerations in order to realize the use of attitude data for SOLID.

- It should be possible, by telecommand, to request data of the attitude disturbance or control compensation, from the ACS, after detecting an impact.

- If SOLID cannot automatically request attitude data from the ACS, attitude data must be stored by the ACS for later retrieval by manual telecommand.

- The relation between the impacting particle’s kinetic energy and the momentum transfer to the satellite should be analysed. The momentum transfer efficiency depends on particle size and velocity[48].
Prototype Verification

This chapter describes the testing activities on the implemented hardware and software prototypes as well as suggestions for future verification activities.

4.1 Engineering Models and Functional Tests

The following prototypes have been implemented:

- ToI Detector with input filter and SC-protection circuit
- MUX board
- MCU software

To confirm the functionality of the prototypes and their analytical design, from chapter 3, a number of lab tests have been conducted. These are described in the following sections.

4.1.1 Input Filter of ToI Detector

4.1.1.1 Test Purpose

- Test functionality of ToI detector input filter
- Validate frequency response of real hardware input filter against theoretical models simulated in PSPICE and analytically derived in Maple

4.1.1.2 Test Setup

A block diagram of the test setup is shown in fig. 4.1, the real test setup in fig. 4.2 and the lab equipment used are listed table 4.1. The signal generator is set to sweep a sine-wave with frequency from 100 kHz to 50 MHz. The digital oscilloscope then measures the filter input and outputs, performs an FFT and calculates the phase between the two signals. The phase measurement additionally applied an averaging function, due to significant fluctuations in this signal.
### 4.1. Engineering Models and Functional Tests

#### 4.1.1.3 Test Results

Below are listed some comments on the test results:

- **Varying Source Output**
  In fig. 4.2b it is seen, that the input signal amplitude is not constant with frequency. This behaviour is not yet completely understood but is likely related to the signal generator being loaded by the active filter and failing to maintain constant output level.

- **Amplitude Response**
  Comparing fig. 4.2b with figs. C.2c and 3.8, some differences are noticed. Firstly, the oscilloscope frequency scale is linear making the curve look longer, as compared to the logarithmic scale in the two theoretical plots. As the frequency increases, the difference between input and output is close to constant. However, since the input signal amplitude is decreasing, in the range 5 to 13 MHz, this actually means that gain is increasing accordingly, as expected from the theoretical plots. At around 17 MHz gain drops to unity and falls steadily at even higher frequencies, as expected.

- **Phase Response**
  In fig. 4.2b is seen a minimum phase of around \(-135^\circ\) at 13 MHz, corresponding to \(45^\circ\) phase margin at this point. At the 3 dB crossover frequency, the phase is around zero, which however doesn’t seem realistic since both the low-pass filter and OpAmp limitation should cause the phase to further decrease at higher frequencies. Since the curve never crosses \(-180^\circ\), it is not possible to estimate the gain margin. At higher frequencies, the phase curve becomes very distorted. This curves is already generated using an averaging function, since real values very even more noisy. The issue with correct phase reading might be due to the use of quite long cables in the setup and possible issues with the signal generator, as discussed above. Using better suitable equipment and setup would likely improve the measurement.

---

**Figure 4.1 – Test setup for measuring frequency response of ToI detector input filter**

**Table 4.1 – Lab equipment for frequency response measurement of ToI input filter test**

<table>
<thead>
<tr>
<th>Device Under Test (DUT)</th>
<th>Lab equipment</th>
</tr>
</thead>
<tbody>
<tr>
<td>• ToI detector</td>
<td>• Signal generator, Agilent 33250A</td>
</tr>
<tr>
<td></td>
<td>• Digital oscilloscope, LeCroy 204MXi</td>
</tr>
<tr>
<td></td>
<td>• Power supply, Agilent E3631A</td>
</tr>
</tbody>
</table>
4. PROTOTYPE VERIFICATION

Morten Olsen
Thesis

(a) Test setup

(b) Frequency response measurement on oscilloscope

Figure 4.2 – Lab test ToI detector input filter
Frequency responses of active OpAmp systems can be efficiently measured using LF network analyzers[49]. The reference provides recommendations for correct measurement setup including cabling and probes, which is important for good measurement results.

**Recommendation 12 (Frequency response measurements)** To verify and optimize the final design of the ToI detector active filter, it is recommended to perform measurements using a suitable LF network analyzer or similar, possible also using active probes, to cope with the very high input impedance of the OpAmps.

### 4.1.2 ToI Detector with MCU

#### 4.1.2.1 Test Purpose

- Test functionality of interface between MCU and ToI detector, including SC-protection circuit and input filter.
- Test functionality of pulse detection system: hardware + software
- Test power saving modes and measure system power consumption

#### 4.1.2.2 Test Setup

A block diagram of the test setup is shown in fig. 4.3a, the real test setup is shown in fig. 4.3b and lab equipments used are listed in table 4.2. A multimeter is placed in series with the MCU power supply, to measure the input current. The ToI detector is powered by the 3 V supply from the STM32F4-Discovery board. Triangular pulses, with varying time periods, are applied to the ToI detector input by the pulse generator, to simulate impact pulses. The filter output, of the ToI detector, is then measured with the oscilloscope. Pulse detection interrupts are recorded in the Hterm program running on the laptop, where from the ToI detector is also controlled.

<table>
<thead>
<tr>
<th>DUT</th>
<th>Lab equipment</th>
</tr>
</thead>
<tbody>
<tr>
<td>ToI detector</td>
<td>Pulse generator, Agilent 33250A</td>
</tr>
<tr>
<td>STM32F4-Discovery + software</td>
<td>Oscilloscope, Hameg HM1008-2</td>
</tr>
<tr>
<td></td>
<td>Multimeter, Fluke 87V</td>
</tr>
<tr>
<td></td>
<td>Hterm v0.8.1a (running on laptop)</td>
</tr>
<tr>
<td></td>
<td>UART-to-USB converter, FTDIChip TTL-232RG-VSW3V3</td>
</tr>
</tbody>
</table>

Table 4.2 – Test setup for ToI detector with MCU
4. PROTOTYPE VERIFICATION

Morten Olsen
Thesis

Signal Generator
10 MHz
0.4 – 1 V

SOLID
Prototype
EPS
(Clydespace/Gomspace)

Oscilloscope
~100 MHz
<15 V

Network Analyzer
100 kHz - 200 MHz

Solar Cells
?V

PC with:
HTerm (or similar)
CANoe (or similar)
UART
(RS232)
5V
CAN
Level
Converter
UART
(RS232)

SOLID EGSE
Signal Generator (TUB)
2 - 10 MHz
0.4 - 1 V

SOLID pulse
detector prototype

Oscilloscope (TUB)
~10 MHz
> 5 V

3.3V or 5V

Signal Generator
2 - 10 MHz
0.4 - 1 V

SOLID pulse
detector prototype

Oscilloscope
~10 MHz
> 5 V

3.3V or 5V

MUX
board
Pulse reset
Short-circuit
TM/TC
Pulse detect
Shutdown
Sun detect
3.3V
MUX control/output
(≥ 14 pins)

Pulse
Generator
SOLID Detector board
(with MUXs and pulse
detector)
3.3V
Control/data
(≥ 19 pins)

UART
3.3V
Level
converter
UART
STM32F4-Discovery
PC
CANDIP
PC
CAN-bus
UART

(a) Block diagram of test setup

(b) Test setup

Figure 4.3 – Testing of transient behaviour of ToI detector together with the MCU
4.1.2.3 Test Results

Below are listed some comments on the test results:

- **Transient Responses**
  Figures 4.4a to 4.4f show transient responses of the ToI detector input filter, for input pulse signals with varying time periods. The curves show that low frequency pulses are efficiently filtered. Signals at 1 MHz are only slightly attenuated and at 5 MHz completely passed through, as expected. At 50 MHz, the filter output still responds somewhat to the input signal, showing the filter limitation in efficiently removing higher frequency noise spikes. If this is a concern, a higher order low-pass filter might be necessary or the filter cut-off frequency could be lowered. It should however also be noted, that the pulse generator was nearing is limit, as seen in the input signal, thus potentially disturbing the measurement. The input pulses are approximately triangular waves meaning that some higher frequency harmonics are present. This is particularly seen in fig. 4.4b, where the fundamental is completed filtered out, but higher harmonics persist. If this is unacceptable, a higher-order high-pass filter or an elliptic high-pass filter may be needed. Comparing these curves to fig. 3.9 shows good correlation between measurements and the analytical model.

- **Power Consumption**
  Measured power consumption, for different system modes, are listed in table 4.3. These values show good agreement with theoretical predictions from table 3.5. Estimating MCU power consumption however heavily depends on the operating frequency and the amount of tasks being executed by the MCU. It is seen, that significant power reduction is possible, when using of MCU power saving modes and disabling unused ToI detectors, as discussed in section 3.2.4. MCU power consumption could further be reduced, by lowering the operating frequency, currently at 16 MHz, provided that the MCU tasks can be executed in a timely manner.

- **Software Control of ToI Detector**
  The test showed stable and correct behaviour of the MCU software. Interrupts were correctly generated on input signals on no code instabilities or "freezes" were experienced. However, high frequency spikes also caused ToI detector triggering and interrupts due to the filter limitations, as discussed above. Figure D.4 shows the Hterm output from the MCU software.

4.1.3 SC-protection Circuit

4.1.3.1 Test Purpose

- Verify SC-protection circuit functionality
- Verify control of SC-protection circuit from the MCU
- Verify analytical calculations and simulations for fault-current trigger levels
Figure 4.4 - Transient tests of ToI detector input filter. Notice changes in amplitude and time scales between plots. Filter input is the CH1 upper trace. Filter output is the CH2 lower trace.

(a) $\tau_{in} \approx 2 \cdot 6 \mu s \rightarrow f_c = 84 \text{kHz}$

(b) $\tau_{in} \approx 2 \cdot 1.65 \mu s \rightarrow f_c = 300 \text{kHz}$

(c) $\tau_{in} \approx 2 \mu s \rightarrow f_c = 500 \text{kHz}$

(d) $\tau_{in} \approx 2 \cdot 500 \text{ns} \rightarrow f_c = 1 \text{MHz}$

(e) $\tau_{in} \approx 2 \cdot 100 \text{ns} \rightarrow f_c = 5 \text{MHz}$

(f) $\tau_{in} \approx 2 \cdot 10 \text{ns} \rightarrow f_c = 50 \text{MHz}$
4.1.3.2 Test Setup

A block diagram of the test is shown in fig. 4.5a, the lab setup is shown in fig. 4.5b and table 4.4 lists the lab equipment used during the test. To simulate a fault-current, a ground-connected potentiometer, included in the ToI detector prototype, is varied until the fault-current limit is reached. Two multimeters measure the two comparator inputs of the SC-protection circuit. The ToI detector is powered by an external power supply, in order to monitor the applied fault-current. The SC-protection circuit is controlled from the laptop via the Hterm program.

<table>
<thead>
<tr>
<th>DUT</th>
<th>Lab equipment</th>
</tr>
</thead>
<tbody>
<tr>
<td>ToI detector</td>
<td>2 x Multimeter, Fluke 87V</td>
</tr>
<tr>
<td>STM32F4-Discovery + software</td>
<td>Power supply, Agilent E3631A</td>
</tr>
<tr>
<td></td>
<td>Hterm v0.8.1a (running on laptop)</td>
</tr>
<tr>
<td></td>
<td>UART-to-USB converter, FTDIChip</td>
</tr>
<tr>
<td></td>
<td>TTL-232RG-VSW3V3</td>
</tr>
</tbody>
</table>

4.1.3.3 Test Results

Below are described the results of the test:

- **Fault-current detection and trigger**
  Figures 4.6a to 4.6c show how the SC-protection circuit responds to fault currents. At a current below 5 mA and 8 V input voltage, the switch remains closed. When the fault-current exceeds 5 mA, the switch is automatically opened and latched. The fault-current trigger-level agreed well with the calculations from appendix C.1. The trigger-level also decreased for increasing input voltage, as simulated in fig. C.1.

- **Manual override**
  The possibility to manually force the switch open or closed, independent of fault-current
Figure 4.5 – Test setup for the SC-protection circuit tests
level, was also verified. This could be useful, should unexpected conditions, noise etc., cause unwanted behaviour of the SC-protection circuit.

- **Noise susceptibility**
  After the addition of the RC low-pass filters at the comparator inputs, no unwanted triggering of the SC-protection circuit were experienced, when applying high-frequency input pulses and waves.

### 4.1.4 MUX Board with MCU

#### 4.1.4.1 Test Purpose

- Test functionality of MUX board
- Test interface between MUX board and MCU + software
- Evaluate MUX readout-time improvements

#### 4.1.4.2 Test Setup

A block diagram of the test is shown in fig. 4.7a, the test setup is shown in fig. 4.7b and a list of used lab equipment is provided in table 4.5. The MUX board and detector board are connected with two ribbon cables; one input cable and one output cable. The MUX board is controlled via the Hterm program on the laptop and the oscilloscope measures the transient switching time of the MUX board output.

<table>
<thead>
<tr>
<th>DUT</th>
<th>Lab equipment</th>
</tr>
</thead>
<tbody>
<tr>
<td>• MUX board</td>
<td>• Oscilloscope, Hameg HM1008-2</td>
</tr>
<tr>
<td>• STM32F4-Discovery + software</td>
<td>• Power supply, Agilent E3631A</td>
</tr>
<tr>
<td>• Detector board from first SOLID prototype</td>
<td>• Hterm v0.8.1a (running on laptop)</td>
</tr>
<tr>
<td></td>
<td>• UART-to-USB converter, FTDIChip</td>
</tr>
<tr>
<td></td>
<td>TTL-232RG-VSW3V3</td>
</tr>
</tbody>
</table>

#### 4.1.4.3 Test Results

- **Read-out values and algorithm**
  The data output, from the detector line scan, is shown in table D.1. The output was stable between consecutive scans. A ‘1’ indicates that a lines was detected broken/disconnected and ‘0’ that the line was intact.
4. PROTOTYPE VERIFICATION

(a) Before triggering - the fault-current is at its limit where both comparator inputs are equal.

(b) After triggering - when the fault-current exceeds the threshold, the switch is opened thus electrically isolating SOLID from the EPS.

(c) Switch forced closed by MCU - this allows inhibit of the SC-protection function, in case of unexpected behaviour. Fault-currents exceeding the threshold are possible, thus this function should be used with care.

Figure 4.6 – Functionality of SC-protection circuit - left multimeter shows the comparator positive input and right multimeter the negative input. The power supply shows the circuit input voltage and current.
Figure 4.7 – Transient switch time, when reading out MUXs, before and after reducing the pull-down resistor value
4. PROTOTYPE VERIFICATION

4.2 Future Tests

4.2.1 EMC-test with EPS from TU-Berlin

Due to concern for EMI susceptibility, as was discussed in section 3.2, a test of the ToI detector together with an EPS from TU Berlin was planned. Unfortunately, it has not yet been possible to realize this test. This test aims to:

- Measure the characteristics of the noise generated by an EPS similar to the one being developed for TechnoSat,
- Test that noise generated by the EPS is efficiently filtered by the input filter of the SOLID ToI detector circuit, such that false pulse triggering is avoided.

Before executing the test a suitable connection interface between the ToI detector and the EPS must be defined, in order to prevent additional noise pickup. Table 4.6 lists the required lab equipments and fig. 4.9 shows a diagram of the suggested test setup.
4.2. Future Tests

Table 4.6 – Test setup for ToI detector with TU Berlin EPS

<table>
<thead>
<tr>
<th>DUT</th>
<th>Lab equipment</th>
</tr>
</thead>
<tbody>
<tr>
<td>ToI detector</td>
<td>Oscilloscope, ≈ 200 MHz</td>
</tr>
<tr>
<td>EPS from TU Berlin</td>
<td>Power supply, 3 V</td>
</tr>
<tr>
<td></td>
<td>Solar cells (for EPS)</td>
</tr>
<tr>
<td></td>
<td>Cable between EPS and the ToI detector</td>
</tr>
</tbody>
</table>

Figure 4.9 – Block diagram of test setup for EPS noise measurements

4.2.2 CAN-bus on new MCU Board

A new MCU board is planned to be designed and built. This MCU board should also include the two CAN-bus transceivers for testing of the communication interface between SOLID and TechnoSat. The purpose of this test is to:

- Test the CAN-bus controller using both various test modes (e.g. loopback) and a second test CAN-node
- Test interface connectors between the SOLID MCU board and detector board

Figure 4.10 shows a block diagram of the suggested test setup and table 4.7 lists the required lab equipments.

Figure 4.10 – Block diagram of suggested CAN-bus test setup
4. PROTOTYPE VERIFICATION

Table 4.7 – Lab equipment for suggested CAN-bus test setup

<table>
<thead>
<tr>
<th>DUT</th>
<th>Lab equipment</th>
</tr>
</thead>
<tbody>
<tr>
<td>New MCU board</td>
<td>Power supply, 3 V</td>
</tr>
<tr>
<td>STM32F4-Discovery</td>
<td>Hterm v0.8.1a (running on laptop)</td>
</tr>
<tr>
<td></td>
<td>UART-to-USB converter, FTDIChip TTL-232RG-VSW3V3</td>
</tr>
</tbody>
</table>

4.2.3 MCU Board with new Detector Board

A new detector board is currently being built. This board should be tested together with the new MCU board, with the purpose of

- testing the interface between the detector board and the MCU board,
- testing functionality of detector board with new isolation diodes.

Figure 4.11 shows a block diagram of the suggested test setup and table 4.8 lists the required lab equipment.

![Block diagram of SOLID system test](image)

Table 4.8 – Lab equipment for suggested test with new detector board and MCU board

<table>
<thead>
<tr>
<th>DUT</th>
<th>Lab equipment</th>
</tr>
</thead>
<tbody>
<tr>
<td>New detector board (including MUXs)</td>
<td>Oscilloscope</td>
</tr>
<tr>
<td>New MCU board</td>
<td>Power supply</td>
</tr>
<tr>
<td></td>
<td>Hterm v0.8.1a (running on laptop)</td>
</tr>
<tr>
<td></td>
<td>UART-to-USB converter, FTDIChip TTL-232RG-VSW3V3</td>
</tr>
</tbody>
</table>

4.2.4 EGSE

In fig. 4.10 is illustrated some components which could be part of a SOLID EGSE. The STM32F4-Discovery board provides a UART-interface to the laptop, a programming interface (ST-Link) to the new MCU board, a CAN-bus via a small PCB with two CAN-transceivers and a 3 V supply.
In future, the Hterm program could be replaced by a more sophisticated test-program developed in LabView or Nokia Qt, which could allow execution of test-scripts etc. The EGSE should also include a pulse generator, for example the Agilent 33250A used during previous lab tests which supports both a General Purpose Interface Bus (GPIB) and RS232 interface.
Conclusion

The purpose of the thesis was to further develop the SOLID embedded electronics with the aim of launching the detector as payload in 2014 on the TechnoSat satellite built by TU Berlin. Some instability issues with the previous prototype also had to be solved.

In chapter 2, the SOLID requirements and mission goals were defined, but also revealed that many parameters are yet to be specified. The environmental conditions for the payload, during the mission, were investigated and showed that only a limited number of space debris impacts is expected.

In chapter 3, several potential performance issues of the detector were discussed, including switch-timing, diode reverse currents and spacecraft charging effects. Recommendation for solving and further analysis of these issues were provided. An active filter circuit was designed, to solve noise issues from the satellite EPS. Tools were provided for easy adjustment of the filter design, when more knowledge of the noise characteristics is available. A Short-circuit protection circuit was also designed, to prevent a complete or partial loss of the satellite EPS, should a short-circuit failure occur. A new MCU software program was developed, but many parts still remain to be implemented.

In chapter 4, the implemented prototypes were tested. All tests verified the intended functionality and performance of the circuits and showed good agreement with theoretical models. Development plans and future tests were also suggested.

The new developed embedded electronics for SOLID provide a more stable and robust detector which will be easier and more suitable to integrate into a larger satellite system. This document provides a collection of the requirements and specifications for SOLID, which prior to this work, were not defined in written anywhere. This document will therefore be useful as a common design reference, for new people joining the project allowing them to follow and understand the design development and when defining the external interfaces to the TechnoSat subsystems.

It is the authors hope, that this work has matured the SOLID embedded electronics and provided a beneficial roadmap for future development, to achieve a high performing payload and successful mission.
Future Perspectives

Below are listed some suggestions for future work on the SOLID embedded electronics:

- **Attitude Data**
  - If attitude data from the ACS is to be used, work should be initiated on how to acquire these data and how to integrate them with the SOLID system.

- **Software**
  - A robust open-source toolchain for compilation of the C-code, should be set up, if the software is developed in-house at DLR in Bremen.
  - Finally decide on the software requirements and specify the protocol for the CAN-bus interface.
  - Set up an SVN server, for file-sharing, if the software development is outsourced.

- **Testing**
  - Develop an EGSE to allow complete system level testing and qualification testing, without requiring access to any of the TechnoSat subsystems.
  - Qualification testing on vital components (MUXs, isolation diodes, optocoupler).
  - Design and implement a new MCU board to allow testing of the CAN-bus.

- **Requirements and System Configuration**
  - Further investigate if impact disturbance pulses are generated on solar arrays, when these are not sunlit and if a sunlight detector is then needed.
  - Decide on the final number of detector boards that will provide sufficient impact data, to meet the expected scientific returns and still stay within the defined budget limits.
  - Further specify the SOLID system configuration, that is, where the different circuits should be located and how many to be included in total. This should lead to more detailed identification and definition of all external interfaces to TechnoSat (power, data, mechanical etc.).
Finally decide if the SC-protection circuit should be included or if the risk of no short-circuit protection is acceptable.
References


[41] *CAN - HDL, HurriCANe*. European Space Agency. URL: %5Curl%7Bhttp://www.esa.int/TEC/Microelectronics/SEMHOX8L6VE_0.html%7D (visited on 08/07/2013).


Appendix

A Detector Line Impedance Calculator 85

B Space Environment Simulations 87
B.1 Total Ionization Dose (SHIELDOSE-2) ................................................................. 87
B.2 Single Event Upset Rates (AP-8) ................................................................. 87
B.3 Impact Rate (MASTER2009) ................................................................. 89
B.4 Erosion by Atomic Oxygen (NRLMSISE-00) ................................................ 91

C Circuit Schematics and Simulations 92
C.1 Short-circuit Protection Circuit with Sun Detector ....................................... 92
C.2 Input Filter .................................................................................................. 95
C.3 ToI Detector ............................................................................................. 97

D Prototype Hardware 99
D.1 Input filter and ToI Detector .................................................................... 99
D.2 MUX Board ........................................................................................... 99
D.3 MCU ....................................................................................................... 103

E Parts List 105

F Software 107
F.1 Source Code ............................................................................................ 107
F.1.1 main.c ................................................................................................. 107
F.1.2 SOLID.c ............................................................................................ 108
F.1.3 SOLID_adc.c ..................................................................................... 114
F.1.4 SOLID_gpio.c ................................................................................... 118
F.1.5 SOLID_interrupts.c ............................................................................ 120
F.1.6 SOLID_pwr.c .................................................................................... 123
F.1.7 SOLID_rcc.c ...................................................................................... 125
F.1.8 SOLID_rtc.c ...................................................................................... 126
Detector Line Impedance Calculator

Figure A.1 shows the input parameters for estimation of the detector line capacitance between the two detector line layers. For simplicity, one detector line layer is considered to be a full ground plane, which leads to a somewhat higher calculated capacitance.
### Embedded Microstrip Impedance Calculator

Provide values for the four parameters $H_1$, $H$, $T$, $W$, and the relative permittivity of the dielectric. Click the button corresponding to the characteristic impedance to calculate its value. ($0.1 < WH < 3.0, 1 < \varepsilon_r < 16$)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>$H_1$ (Dielectric thickness)</td>
<td>0.100</td>
<td>mm/mils</td>
</tr>
<tr>
<td>$H$ (Height above reference plane)</td>
<td>0.035</td>
<td>mm/mils</td>
</tr>
<tr>
<td>$T$ (Trace thickness)</td>
<td>0.025</td>
<td>mm/mils</td>
</tr>
<tr>
<td>$W$ (Trace width)</td>
<td>0.1</td>
<td>mm/mils</td>
</tr>
<tr>
<td>$\varepsilon_r$ (Relative permittivity)</td>
<td>3.4</td>
<td></td>
</tr>
<tr>
<td>$Z_0$ (Characteristic impedance)</td>
<td>50.8</td>
<td>ohms</td>
</tr>
<tr>
<td>$C_0$ (Capacitance)</td>
<td>1.901</td>
<td>pF/cm/pF/inch</td>
</tr>
<tr>
<td>$T_{pd}$ (Propagation delay time)</td>
<td>50.8</td>
<td>psec/cm/psec/inch</td>
</tr>
</tbody>
</table>

### Notes

- The formulas used here are taken from the Design Guide for Electronic Packaging Utilizing High-Speed Techniques (4th Working Draft, IPC-2221, February 2003)

$$
Z_0 = \frac{87}{\sqrt{\varepsilon_r + 1.41}} \ln \left( \frac{5.98H}{0.8W + T} \right) \times \left( 1 - \frac{H_1 - T - H}{0.1} \right) \text{ ohms}
$$

$$
T_{pd} = 84.75 \times \sqrt{0.475 \times \varepsilon_r \times \left( 1 + e^{-1.55H_1/H} \right) + 0.67} \text{ psec/inch}
$$

$$
C_0 = \frac{T_{pd}}{Z_0} \text{ pF/inch}
$$

- Range of valid parameters specified in the Design Guide: $0.1 < WH < 3.0, 1 < \varepsilon_r < 16$
- Note the equations used here are proposals currently being evaluated by IPC. The embedded microstrip impedance formula is an approximation and the results are highly sensitive to the values of the variables. The accuracy of the equations was observed to be slightly better for higher permittivities.
- For typical PCB parameters $\varepsilon_r = 4, H_1 = 61.37 \text{ mil}, H = 30 \text{ mil} \text{ and } T = 1.37 \text{ mil}$, the deviation of the calculated results obtained using a 2D numerical field solver, is listed below:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Z_0$</td>
<td>$0.1 &lt; WH &lt; 1.0$</td>
<td>within 20%</td>
</tr>
<tr>
<td></td>
<td>$1.0 &lt; WH &lt; 3.0$</td>
<td>20%--40%</td>
</tr>
<tr>
<td>$C_0$</td>
<td>$0.1 &lt; WH &lt; 1.0$</td>
<td>within 5%</td>
</tr>
<tr>
<td></td>
<td>$1.0 &lt; WH &lt; 2.2$</td>
<td>within 20%</td>
</tr>
<tr>
<td></td>
<td>$2.2 &lt; WH &lt; 3.0$</td>
<td>20%--40%</td>
</tr>
</tbody>
</table>

---

Figures A.1 – Online impedance calculator for microstrip traces

Source: [50]
Space Environment Simulations

This section lists the simulation input parameters used in the various space environment models as well as the results from the simulation outputs.

B.1 Total Ionization Dose (SHIELDOSE-2)

SPENVIS 4.6.6.2655  21-May 2013  18:11:14
Ionizing dose
Project: TEST1
Test 1 project

<table>
<thead>
<tr>
<th>Target material: Si</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shield configuration: Transmission surface of finite Al slab shield</td>
</tr>
<tr>
<td>Proton results without nuclear attenuation</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Total mission dose (rad)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\text{Al absorber thickness} )</td>
</tr>
<tr>
<td>------------------------</td>
</tr>
<tr>
<td>0.010</td>
</tr>
<tr>
<td>0.050</td>
</tr>
<tr>
<td>0.100</td>
</tr>
<tr>
<td>0.200</td>
</tr>
<tr>
<td>0.300</td>
</tr>
<tr>
<td>0.400</td>
</tr>
<tr>
<td>0.500</td>
</tr>
<tr>
<td>0.750</td>
</tr>
<tr>
<td>1.000</td>
</tr>
<tr>
<td>1.500</td>
</tr>
<tr>
<td>2.000</td>
</tr>
<tr>
<td>2.500</td>
</tr>
</tbody>
</table>

Figure B.1 – TID simulation, using SHIELDOSE-2 model, for parts with limited radiation shielding

B.2 Single Event Upset Rates (AP-8)
### SEU rates in SOLID electronics for 640 km orbit altitude and 53° inclination

- **Overview input**
  - Particle spectra:
    - AP-8 MAX trapped protons
    - ESP-PSYCHE total flux solar particles (H - U)
    - ISO 15390 GCR particles (H - U)
  - Spacecraft shielding thickness (Al equivalent): 0.20 cm
  - No of devices: 1

<table>
<thead>
<tr>
<th>Device name</th>
<th>Heavy ion method</th>
<th>Proton method</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOL DEFAULT (user defined)</td>
<td>$L_A = 5.4 \times 10^{-6}$ MeV cm²/mg</td>
<td>$A = 4.08$ MeV,B = 7.06 MeV</td>
</tr>
<tr>
<td>SPP:36.30 x 30.0 x 100 (µm²)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Total mission single event upsets

<table>
<thead>
<tr>
<th>Device</th>
<th>Effect</th>
<th>(# outfile)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEFAULT</td>
<td>Direct ionization</td>
<td>2.15E+01</td>
</tr>
<tr>
<td></td>
<td>Proton induced ionization</td>
<td>2.259E+01</td>
</tr>
<tr>
<td></td>
<td>Total</td>
<td>2.173E+01</td>
</tr>
</tbody>
</table>

### SEU rates in SOLID electronics for 640 km orbit altitude and 98° inclination

- **Overview input**
  - Particle spectra:
    - AP-8 MAX trapped protons
    - ESP-PSYCHE total flux solar particles (H - U)
    - ISO 15390 GCR particles (H - U)
  - Spacecraft shielding thickness (Al equivalent): 0.20 cm
  - No of devices: 1

<table>
<thead>
<tr>
<th>Device name</th>
<th>Heavy ion method</th>
<th>Proton method</th>
</tr>
</thead>
<tbody>
<tr>
<td>TOTAL</td>
<td>$L_A = 5.4 \times 10^{-6}$ MeV cm²/mg</td>
<td>$A = 4.08$ MeV,B = 7.06 MeV</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Total mission single event upsets

<table>
<thead>
<tr>
<th>Device</th>
<th>Effect</th>
<th>(# outfile)</th>
</tr>
</thead>
<tbody>
<tr>
<td>TOTAL</td>
<td>Direct ionization</td>
<td>1.20E+01</td>
</tr>
<tr>
<td></td>
<td>Proton induced ionization</td>
<td>7.63E+01</td>
</tr>
<tr>
<td></td>
<td>Total</td>
<td>1.30E+02</td>
</tr>
</tbody>
</table>

(a) SEU rates in SOLID electronics for 640 km orbit altitude and 53° inclination
(b) SEU rates in SOLID electronics for 640 km orbit altitude and 98° inclination

Figure B.2 – SEU rates for varying orbit inclinations
B.3 Impact Rate (MASTER2009)

Impact rates were initially simulated using the SPENVIS online tool. Unfortunately, it only supports a "tumbling plate target" for MMOD flux simulations whereas a "sphere target" is desired. Therefore simulations were performed using the ESA-MASTER2009 graphical user interface developed by Institute of Aerospace Systems at TU-Braunschweig\cite{51}. Simulation parameters are shown in fig. B.3. For simplification, the orbit altitude is considered constant throughout the mission lifetime.
Figure B.3 – Impact rate simulation using MASTER2009 model, for 640 km orbit altitude and 98° inclination.
B.4 Erosion by Atomic Oxygen (NRLMSISE-00)

The AO simulation input parameters are:

- 450 km orbit altitude,
- 98° inclination,
- one year mission duration.

Figure B.4 – Simulation of AO erosion rate on a Kapton surface, using the NRLMSISE-00 model
Circuit Schematics and Simulations

Below are shown, for various SOLID circuits, the Cadence OrCAD 16.c circuit schematics and PSPICE simulation results. For some of the circuits, parameter calculations are also provided.

C.1 Short-circuit Protection Circuit with Sun Detector

Below is shown calculations of the SC protection circuit trigger point for the maximum allowed input current. Figure 3.12 shows the circuit and component values used in the calculations. Assuming a solar array voltage of 15 V, the comparator negative input voltage is

\[ V^- = V_{bias} + (V_{solar} - V_{bias}) \frac{R_{19}}{R_{18} + R_{19}} = 1.5 V + (15 V - 1.5 V) \frac{7.7 \text{k}\Omega}{7.7 \text{k}\Omega + 91 \text{k}\Omega} = 2.553 V \]  

(C.1)

where \( V_{solar} \) is the input voltage and \( V_{bias} \) the 1.5 V bias voltage from the input filter. Assuming the comparator output is initially high, the positive input voltage is calculated as

\[ V^+ = \left( \frac{V_{filter,IN}}{R_{20}} + \frac{V_{OH}}{R_{22}} + \frac{V_{bias}}{R_{21}} \right) \frac{1}{R_{20} + R_{21} + R_{22}} \]  

(C.2)

where \( V_{filter,IN} \) is the signal on the filter input, after the current sense resistor \( R_{30} \). Equalling the positive and negative comparator input voltages gives the threshold voltage for \( V_{filter,IN} \) that triggers the comparator

\[ V^+ = V^- \rightarrow 

V_{T_HL} = \left( V^- \left( \frac{1}{R_{21}} + \frac{1}{R_{20}} + \frac{1}{R_{22}} \right) - \frac{V_{bias}}{R_{21}} - \frac{V_{OH}}{R_{22}} \right) R_{20} \]  

(C.3)

\[ = (2.553 V \left( \frac{1}{7.7 \text{k}\Omega} + \frac{1}{91 \text{k}\Omega} + \frac{1}{100 \text{k}\Omega} \right) - \frac{1.5 V}{7.7 \text{k}\Omega} - \frac{3.1 V}{100 \text{k}\Omega})91 \text{k}\Omega = 14.502 V \]

where \( V_{T_HL} \) is the threshold voltage from high-to-low and \( V_{OH} \) is the high level output voltage of the comparator. A threshold voltage of 14.502 V corresponds to a fault current exceeding

\[ i_{fault} = \frac{V_{solar} - V_{T_HL}}{R_{30}} = \frac{15 V - 14.502 V}{100 \Omega} = 5 mA \]  

(C.4)

The current level can be changed by increasing/decreasing the sense resistor, \( R_{30} \). The trigger current level depends somewhat on the solar array input voltage and will decrease with increasing voltage. Alternatively, the negative comparator input voltage could be derived from the 3.3 V supply and a differential amplifier placed in front of the comparator. To drive the comparator output from low to high, a voltage of 20.4 V is required on the positive input, which should never occur - particularly since the solar array voltage will be disconnected. Hence the short-circuit trigger output is latched. To reset the latch, a control signal from the MCU is required. To close the switch, \( S_3 \), up to 0.5 mA LED supply current is needed. The bias resistor \( R_{17} \) is then calculated as

\[ R_{17} = \frac{V_{supply} - V_{LED} - V_F}{0.2 mA} = \frac{3.3 V - 1.4 V - 0.3 V}{0.2 mA} = 3.2 \text{k}\Omega \]  

(C.5)
where $V_{LED}$ and $V_F$ are the diode forward voltage drops from the optocoupler and diode $D_1$ respectively. A slightly smaller resistor is chosen, to draw extra current, to allow some relay degradation due to radiation. $R_{23}$ allows the the MCU to reset the comparator by pulling its output high, even if the comparator is trying to keep it low. The diode $D_1$ blocks any high voltage from a failure in the solid-state relay, making the SC protection circuit a single-point failure free circuit, considering all failure modes from [27, Annex G]. Figure C.1 shows PSPICE schematic and simulation results for the SC-protection circuit. The solar array input voltage has superimposed high frequency noise signal and noise spikes to simulate the robustness of the circuit against wrong triggering from noise. A short circuit condition is provoked by slowly shorting the output after 10 ms. After 30 ms the output condition are reset to normal.
(a) PSPICE schematic of the SC protection circuit

(b) Input and output conditions for simulation of the SC protection PSPICE circuit

(c) PSPICE simulation of SC protection circuit. **Bottom plot:** Solar array voltage with superimposed high frequency noise and spike voltages. 2nd plot from bottom: SC protection circuit output voltage with cut off at 14 ms. Slight voltage increase at 33 ms is the 1.5 V bias voltage seen at input, when fault is removed but solar array still cut off, illustrating desired circuit latch effect. 3rd plot from bottom: Differential voltage on comparator inputs, showing 50 mV hysteresis for triggering the SC protection. When triggered, -250 mV is required to reset the latch which is impossible by the circuit itself thus preventing erroneous self reset. 2nd plot from top: Telemetry/control output of SC protection circuit showing ON and CUT OFF states. Pull this signal high with MCU to reset SC protection circuit. Top plot: SC protection circuit triggering at fault current exceeding around 10 mA.

Figure C.1 – PSPICE SC protection circuit schematic and simulation
C.2 Input Filter
(a) PSPICE schematic of input filter for the ToI detector circuit

(b) Simulation inputs for PSPICE model of input filter circuit. SC protection circuit included to simulate filter loading effects from this circuit.

Date/Time run: 08/01/13 11:28:48
Temperature: 27.0

(c) simulation results of input filter PSPICE model. **Bottom plot:** dB bode plot showing the bandpass filter around 1.5 MHz to 15 MHz and notch filter at 300 kHz. A few dB gain is added in the bandpass region. **Top plot:** Phase of the output voltage, showing around 60° phase margin at the 0 dB cross over frequency around 15 MHz.

Figure C.2 – PSPICE schematic and simulation results of input filter
Figure C.3 – OrCAD schematic of ToI detector circuit including input filter and SC protection circuit. The many voltage sources and ramps simulate incoming impact pulse signals at different frequencies, noise signals and spikes as well as slow variations in the solar array voltage.

From the above simulation can be learned that the ToI detector circuit generally works as intended. However, noise from higher harmonics of the switch converter and high frequency voltage spikes might still cause problems. Especially the high frequency spikes are heavily affected by the final circuit layout, component selection etc. of the TechnoSat EPS, so the results from this simulation should also be compared with lab measurements on real hardware, before final conclusions are made.
**Figure C.4** – OrCAD circuit schematic of reference voltage filter

**Figure C.5** – OrCAD circuit schematic of ToI detector comparator circuit

**Figure C.6** – PSPICE simulation results of ToI detector circuit. **Bottom plot:** The solar array input voltage with converter switching noise, noise spikes and impact pulses superimposed. **2nd from bottom:** Impact pulses signals. **2nd from top:** The two comparator inputs. The lower signal is the direct output from the input filter. The noise present is the higher harmonics noise of the switching of the converter as well as the very high frequency noise spikes. The constant higher signal is the reference voltage with a slight positive offset to set the detection threshold voltage level. **Top plot:** The output of the ToI detector circuit. As seen, it always reacts on the impact pulses. The very high frequency noise spikes also triggers the circuit.
Prototype Hardware

D.1 Input filter and ToI Detector

![Prototype ToI detector](image)

(a) Top side  (b) Bottom side

Figure D.1 – PCB layout of prototype ToI detector

D.2 MUX Board

Table D.2 shows the pin mapping between the MUX board and the detector board of the first SOLID prototype. Only the Y-axis is scanned. The MUX_OUT connects to the detector board Y_IN and the MUX_IN to the detector board Y_OUT.
Figure D.2 – PCB layout of the MUX for the second SOLID prototype
Figure D.3 – Circuit schematic of MUX board
Table D.1 – Data from MUX board readout. ‘1’ represents a broken/disconnected detector line and ‘0’ an intact detector line.

<table>
<thead>
<tr>
<th>Line: 1 9 17 25</th>
<th>Group 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>29</td>
<td>0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1</td>
</tr>
<tr>
<td>25</td>
<td>0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1</td>
</tr>
<tr>
<td>21</td>
<td>0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1</td>
</tr>
<tr>
<td>17</td>
<td>5 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 1</td>
</tr>
<tr>
<td>13</td>
<td>9 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 1</td>
</tr>
<tr>
<td>9</td>
<td>0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 1</td>
</tr>
<tr>
<td>5</td>
<td>0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 1</td>
</tr>
<tr>
<td>1</td>
<td>1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1</td>
</tr>
<tr>
<td>9</td>
<td>1 0 0 0 0 1 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1</td>
</tr>
<tr>
<td>17</td>
<td>1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1</td>
</tr>
<tr>
<td>25</td>
<td>0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 1</td>
</tr>
<tr>
<td>29</td>
<td>0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 1</td>
</tr>
</tbody>
</table>
Table D.2 – Pin mapping between MUX board and detector board of the first SOLID prototype[16]

<table>
<thead>
<tr>
<th>MUX board</th>
<th>Detector board</th>
<th>MUX board</th>
<th>Detector board</th>
</tr>
</thead>
<tbody>
<tr>
<td>32</td>
<td>1</td>
<td>24</td>
<td>17</td>
</tr>
<tr>
<td>16</td>
<td>2</td>
<td>8</td>
<td>18</td>
</tr>
<tr>
<td>31</td>
<td>3</td>
<td>23</td>
<td>19</td>
</tr>
<tr>
<td>15</td>
<td>4</td>
<td>7</td>
<td>20</td>
</tr>
<tr>
<td>30</td>
<td>5</td>
<td>22</td>
<td>21</td>
</tr>
<tr>
<td>14</td>
<td>6</td>
<td>6</td>
<td>22</td>
</tr>
<tr>
<td>29</td>
<td>7</td>
<td>21</td>
<td>23</td>
</tr>
<tr>
<td>13</td>
<td>8</td>
<td>5</td>
<td>24</td>
</tr>
<tr>
<td>28</td>
<td>9</td>
<td>20</td>
<td>25</td>
</tr>
<tr>
<td>12</td>
<td>10</td>
<td>4</td>
<td>26</td>
</tr>
<tr>
<td>27</td>
<td>11</td>
<td>19</td>
<td>27</td>
</tr>
<tr>
<td>11</td>
<td>12</td>
<td>3</td>
<td>28(only out)</td>
</tr>
<tr>
<td>26</td>
<td>13</td>
<td>18</td>
<td>29(only out)</td>
</tr>
<tr>
<td>10</td>
<td>14</td>
<td>2</td>
<td>30(only out)</td>
</tr>
<tr>
<td>25</td>
<td>15</td>
<td>17</td>
<td>NC&lt;sup&gt;a&lt;/sup&gt;</td>
</tr>
<tr>
<td>9</td>
<td>16</td>
<td>1</td>
<td>NC&lt;sup&gt;a&lt;/sup&gt;</td>
</tr>
</tbody>
</table>

<sup>a</sup> Not connected

D.3 MCU
Table D.3 – **MCU pin layout for second SOLID prototype**

<table>
<thead>
<tr>
<th>Pin name</th>
<th>Pin function</th>
</tr>
</thead>
<tbody>
<tr>
<td>PA0</td>
<td>Wakeup interrupt&lt;sup&gt;a&lt;/sup&gt;</td>
</tr>
<tr>
<td>PA1</td>
<td>Shutdown control</td>
</tr>
<tr>
<td>PA2</td>
<td>SC-protection status</td>
</tr>
<tr>
<td>PA3</td>
<td>Sun detection</td>
</tr>
<tr>
<td>PB1</td>
<td>ToI detector output</td>
</tr>
<tr>
<td>PB2</td>
<td>ToI detector reset</td>
</tr>
<tr>
<td>PB4</td>
<td>MUX enable/disable ((E_N))</td>
</tr>
<tr>
<td>PB5</td>
<td>MUX write enable/disable ((WR))</td>
</tr>
<tr>
<td>PB7</td>
<td>MUX (CS)</td>
</tr>
<tr>
<td>PB11 to PB15</td>
<td>MUX_OUT address control</td>
</tr>
<tr>
<td>PC4</td>
<td>MUX read output (x-axis)</td>
</tr>
<tr>
<td>PC5</td>
<td>MUX read output (y-axis)</td>
</tr>
<tr>
<td>PD0</td>
<td>CAN-bus RX-pin&lt;sup&gt;b&lt;/sup&gt;</td>
</tr>
<tr>
<td>PD1</td>
<td>CAN-bus TX-pin&lt;sup&gt;b&lt;/sup&gt;</td>
</tr>
<tr>
<td>PD8</td>
<td>UART TX-pin</td>
</tr>
<tr>
<td>PD9</td>
<td>UART RX-pin</td>
</tr>
<tr>
<td>PE3 to PE7</td>
<td>MUX_IN address control</td>
</tr>
</tbody>
</table>

<sup>a</sup> Connected to 'blue' push-button on STM32F4-Discovery board

<sup>b</sup> Not yet implemented

---

Figure D.4 – **Terminal output from MCU software**

---

104
Parts List

Table E.1 shows the expected flight parts for SOLID.
<table>
<thead>
<tr>
<th>Function</th>
<th>Part</th>
<th>Manufacturer</th>
<th>Qualification</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Detector board</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Detector line diodes</td>
<td>ESD5V3U1U-02LS E6327</td>
<td>Infineon</td>
<td>?</td>
</tr>
<tr>
<td>MUX</td>
<td>ADG732</td>
<td>Analog Devices</td>
<td>?</td>
</tr>
<tr>
<td>RF choke</td>
<td>TBD</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Silicon adhesive</td>
<td>RTV-S691</td>
<td>Wacker Silicons</td>
<td>F</td>
</tr>
<tr>
<td>Copper clad</td>
<td>Pyralux AP</td>
<td>DuPont</td>
<td>F</td>
</tr>
<tr>
<td>Copper coverlay</td>
<td>Pyralux LF</td>
<td>DuPont</td>
<td>F</td>
</tr>
<tr>
<td><strong>MCU board</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MCU</td>
<td>STM32F427VG</td>
<td>STMicroelectronics</td>
<td>F</td>
</tr>
<tr>
<td>CAN transceiver</td>
<td>MAX3051</td>
<td>Maxim</td>
<td>F</td>
</tr>
<tr>
<td>Resistors</td>
<td>TBD</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Capacitors</td>
<td>TBD</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Connectors (detector- and MUX</td>
<td>G125</td>
<td>Harwin</td>
<td>Q</td>
</tr>
<tr>
<td>boards)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cable wire</td>
<td>TBD</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bus-connector (TechnoSat bus)</td>
<td>TBD</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PCB</td>
<td>FR4 TBC</td>
<td>TBD</td>
<td>F</td>
</tr>
<tr>
<td><strong>ToI detector</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fast OpAmp</td>
<td>LTC6246HS6</td>
<td>Linear Technology</td>
<td>?</td>
</tr>
<tr>
<td>Slow OpAmp</td>
<td>LMV641</td>
<td>Texas Instruments</td>
<td>?</td>
</tr>
<tr>
<td>ToI detector comparator</td>
<td>ADCMP601</td>
<td>Analog Devices</td>
<td>?</td>
</tr>
<tr>
<td>Transistor latch</td>
<td>BSS83</td>
<td>NXP</td>
<td>?</td>
</tr>
<tr>
<td>Transistor reset</td>
<td>2PC4617Q</td>
<td>NXP</td>
<td>?</td>
</tr>
<tr>
<td>Filter inductor</td>
<td>IMC0805ER1R0J01</td>
<td>Vishay Dale</td>
<td>?</td>
</tr>
<tr>
<td>Filter capacitors</td>
<td>TBD</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Filter resistors</td>
<td>TBD</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Digital potentiometer</td>
<td>MCP4024</td>
<td>Microchip</td>
<td>?</td>
</tr>
<tr>
<td><strong>SC-protection circuit and sun detector</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Comparator</td>
<td>MCP6542-E/MS</td>
<td>Microchip</td>
<td>?</td>
</tr>
<tr>
<td>Solid state relay</td>
<td>LH1525AAB</td>
<td>Vishay Semiconductor</td>
<td>?</td>
</tr>
<tr>
<td>Current sense resistor</td>
<td>TBD</td>
<td></td>
<td>?</td>
</tr>
<tr>
<td>Signal diode</td>
<td>1PS76SB10</td>
<td>NXP</td>
<td>?</td>
</tr>
<tr>
<td>Photo diode</td>
<td>BPW34S</td>
<td>Osram</td>
<td>?</td>
</tr>
</tbody>
</table>

*a F = Flight heritage
Q = Qualified, but no flight heritage
N = None
? = Unknown
Software

F.1 Source Code

F.1.1 main.c

C_source_code/main.c

TO BE DONE (suggestions):
- Several functions use a "while(some_status_flag)" loop to wait for a flag to be set/cleared. If this fails, software might get stuck forever in this loop – hence a general timeout function is needed.
- Must functions return in integer ('0') as a status of successful function completion. The software should be able to react when a failure/error occurs and the function returns a non-zero value. The macros EXIT_SUCCESS and EXIT_FAILURE can be used together with the exit() function.
- Make an error handling/check function for all divisions, to prevent a runtime error if trying to divide by zero. (The Cortex M4 processor has a 'divide-by-zero' exception which can be used for this purpose).
- Check all limit values in loop counters that they are valid and incorrect setting will not crash the program. Also, that loop counters will not continue indefinitely if a, for some reason, a wrong starting point is provided e.g. wait for something like while (i != 10) i++ should be converted to: while (i < 10) i++
- Arguments passed to functions shall, whenever practically possible, make use of 'typedef' (maybe together with 'enum') values to avoid wrong type of value passed as argument.
- The Cortex M4 Cyclic Redundancy Check (CRC) module should be used to detect memory errors due to space radiation. Possible some Error Correction Code (ECC) could be implemented or memory has to be updated from ground, whenever a bit-error is detected in memory.
- Calibration routine for internal Low Speed oscillator (LS1).
- Instead of using EXTI1 external interrupt for pulse detection, use of tamper event with TimeStamp function, level detection and filtering function. This will be more efficient and robust against noise.
- Configure the CAN controller module.
- remove all functions/dependence on STM32F4–Discovery board for future development – these are: LEDs, pushbuttons from solid_gpio.c

@file main.c
@author Morten Olsen <mortenolsen100@hotmail.com>
@version 1.0
@date 08–August–2013
@brief SOLID main function

SOLID main function code. Code designed for use with STM32F4–Discovery board.

(C)COPYRIGHT 2013 Deutches Zentrum fur Luft und Raumfahrt(DLR)

*/

#include "stm32f4xx.h"
#include "SOLID_pwr.h"
#include "SOLID.h"
/suggested header files for error handling and debugging */
#include "errno.h" // error return values
#include "assert.h" // to check for arguments passed to functions (e.g. avoid divide by zero)
#include "setjmp.h" // functions to jump out of a function, if an error state is detected

// #define NDEBUG // to turn assertion checking off

 /*************************************************************************/
 /* SOLID MAIN FUNCTION */
 /*************************************************************************/

/**
@brief SOLID main function.

Before calling this routine, the startupfile, stm32f4xx.s is executed which calls the Config_SysClk() function that sets up the microcontroller system clock.

Main function initializes microcontroller, checks for active action flags, executes these and enters sleep mode (if enabled) when no more action flags are pending.
*/
int main(void)
{
    SOLID_init(); // initialize all MCU registers

    while (1) // infinite loop
    {
        check_pending_action_flags(); // check if actions are pending

        if (ACTIVE_PWR_MODE != PWR_MODE_RUN)
        {
            GPIOD->ODR &= 0xEFFF; // clear PD12 (turn OFF green LED = enable power saving)
            PWR_EnterPowerSavingMode(ACTIVE_PWR_MODE); // enter power saving mode
        }

        GPIOD->ODR |= 0x1000; // set bit PD12 (turn ON green LED -> RUN mode enabled)
    }

}/*
END OF FILE

C_source_code/SOLID.c

 /*************************************************************************/
/*
@file SOLID.c
@author Morten Olsen
@version 1.0
@date 08—August—2013
@brief General SOLID functions

General SOLID functions

COPYRIGHT (C) 2013 by Deutches Zentrum fur Luft und Raumfahrt (DLR)
*/

#include "SOLID.h"
#include "stm32f4xx.h"
#include "SOLID_usart.h"
#include "SOLID_gpio.h"
#include "SOLID_rcc.h"
#include "SOLID_interrupts.h"
#include "SOLID_rtc.h"
#include "SOLID_pwr.h"
#include "SOLID_adc.h"
#include <stdio.h>
#include "core_cm4.h"

/**
 @brief Variable declaration for array of pending action flags.
 Array size of pending action flags should be at least equal to the highest
 index value of any action flag index.
 */
int action_flags[10] = {0};

/**
 @brief Declare data arrays

 Information of broken lines for each axis (X and Y) is stored in integer
 arrays, with dimensions fitted to the number of detector lines, with nX being
 the number of detector lines for the X-axis, as illustrated below:

\[
\begin{array}{cccccc}
\text{group number} & \text{line number} & 1 & 2 & 3 & 4 & 5 & 6, \ldots, (nX/32) \\
\text{in groups} & 1 & 0 & 0 & 0 & 0 & 0 & 0 \quad 0 \\
(1..32) & 2 & 0 & 0 & 0 & 0 & 0 & 0 \quad 0 \\
| & 3 & 0 & 0 & 0 & 0 & 0 & 0 \quad 0 \\
| & 4 & 0 & 0 & 0 & 0 & 1 & 0 \quad 0 \\
v & 5 & 0 & 0 & 1 & 0 & 0 & 0 \quad 0 \quad <--- \text{structure of two-dimensional data array for X-axis}
\end{array}
\]

'0' means: detector line is intact
'1' means: detector line is broken

In the above example, two lines are detected as broken: line 32*2+5 = '69' and line 32*4+4 = '132'

For example, if 840 detector lines are needed, the array dimension (number of groups) is calculated as:
840 / 32 = 26.25 --> 27 groups (rounding up)
*/
uint32_t impact_data_panel1_X[32] = {0};
uint32_t impact_data_panel1_Y[32] = {0};

/----------------------------------------------- SOLID FUNCTIONS -----------------------------------------------*/

/**
 @brief Checks for pending action flags

 Checks of any action flags are set in the action_flags[] array and executes
 the pending action. This function is called continuously when in RUN mode.
 When in one of the power saving modes, this function is called once before
 MCU enters power saving mode. Upon interrupt (wakup from power saving mode),
 this function is again called.

 @return execution status (NOT IMPLEMENTED YET)
 */
int check_pending_action_flags(){
    int i = 0;
    // get number of elements in action flags array
    int n_action_flags = sizeof(action_flags) / sizeof(action_flags[0]);
    for (i = 0; i < n_action_flags; i++) // for each action flag
    { // if(action_flags[i] != 0) // check if action flag is pending
        execute_action(i); // execute pending action
        action_flags[i] = 0; // reset action flag
        i = 0; // reset loop counter(while executing one action, a previous could become pending)
```c
return 0;

/**
 * @brief Simple delay function.
 * Simple delay function. HSI clock frequency (system clock) = 16 MHz
 */
void Delay(__IO uint32_t nCount){
  /* Decrement nCount value */
  while (nCount != 0)
  {
    nCount --;
  }
}

/**
 * @brief Sets the address of the MUXs.
 * @param group_num detector line group number. Valid range: 0 to 31 (TBD)
 * @param line_num detector line in selected detector line group: Valid range: 0 to 31 (TBD)
 * Sets the input and output addresses of the MUXs.
 * @return execution status (NOT IMPLEMENTED YET)
 */
int MUX_setAddress(int group_num, int line_num){
  // unlock MUX address setting
  GPIOB->ODR &= (uint16_t) 0xFFDF;  // enable MUX write access (PB5, WR=low)
  GPIOB->ODR &= (uint16_t) 0xFF7F;  // CS pin low (PB7)
  
  /* Config input MUX address (pin PE3 to PE7)
   Selects the detector line group to be voltage biased */
  GPIOE->ODR &= 0xFF07;  // clear address outputs
  GPIOE->ODR |= ((uint16_t) ((group_num << 3) & 0x00F8) );  // set detector group to be biased
  
  /* Config output MUX address (pins PB11 to PB15)
   Selects the line number (in the current group) to be read at the MUX output */
  GPIOB->ODR &= 0x07FF;  // clear address outputs
  GPIOB->ODR |= ((uint16_t) ((line_num << 11) & 0xF800) );  // set detector line to be read

  // lock MUX address
  GPIOB->ODR |= (uint16_t) 0x0020;  // disable MUX write access (PB5)
  GPIOB->ODR |= (uint16_t) 0x0080;  // CS pin high (PB7)

  return 0;
}

/**
 * @brief Checks for broken detector lines.
 * @param data_panel_X pointer to data array for X-axis detector lines
 * @param data_panel_Y pointer to data array for Y-axis detector lines
 * IMPORTANT: dimensions(data array size) of X- and Y-axis must be equal,
 * otherwise algorithm might try to store values beyond array dimensions and result in undefined behaviour! (TO BE FIXED)
 * NOTE: Indication of broken lines is stored, but if by mistake a line is once detected as broken (even if intact), data will not be reset on later scans. Maybe update this algorithm or create 'fix-algorithm' to avoid the problem. Writing the current state ('0' or '1') for each line on each scan, is undesired since it would require a significant amount of unnecessary write accesses to the data array. Especially since this function is expected to be called frequently.
 * FUTURE: When more SOLID detector panels are used, this function should be called for each panel individually. NOTE2: all MUXs (also for different panels) are controlled using the same physical
communication wires, hence setting the address of the MUXs on one panel, automatically sets the same address on all other panels. Alternatively, many more communication wires will be needed.

```c
int MUX_checkLines(uint32_t *data_panel_X, uint32_t *data_panel_Y){
    // group number and detector line number
    int nLine, nGroup = 0;
    const int size_data = 32; // set number of elements in data array
    const int lines_per_group = 32; // set number of detector lines per group

    GPIOB->ODR &= (uint16_t) 0xFFFE; // enable MUX (enable/disable pin PB4)

    for (nGroup = 0; nGroup < size_data; ){ // check each group of detector lines
        for (nLine = 0; nLine < lines_per_group; ){ // check each detector line in the group
            MUX_setAddress(nGroup, nLine); // update MUX address

            // add delay: wait 5 ms (for transient voltage on detector line to settle after MUX switching)
            Delay(0x00FF);

            // read pin PC4 status (X-axis) and check if current line is broken (low voltage)
            if ( !(GPIOC->IDR & 0x00000010) ){
                data_panel_X[nGroup] |= ((uint32_t) (1 << nLine)); // line broken: set bit in data array
            }

            // read pin PC5 status (Y-axis) and check if current line is broken (low voltage)
            if ( !(GPIOC->IDR & 0x00000020) ){
                data_panel_Y[nGroup] |= ((uint32_t) (1 << nLine)); // line broken: set bit in data array
            }

            nLine++;
        }
        nGroup++;
    }
    GPIOB->ODR |= (uint16_t) 0x0010; // disable MUX (enable/disable pin PB4)

    return 0;
}
```

@brief Initializes the microcontroller

Initialization functions for the various MCU hardware modules used by SOLID.
Order of the functions should not matter (TO BE TESTED/VERIFIED).

```c
int SOLID_init(){
    Config_GPIO();
    Config_USART();
    Config_RTC();
    ConfigInterrupts();
    Config_EXTI();
    Config_Power();
    Config_ADC1();

    // to be implemented:
    //Config_IWDG();
    //Config_Timers();
    //Config_CAN();
    //Config_DMA();?
    //SysTick_Config(); ?

    return 0;
}
```

@brief Enables/disables the pulse detector and filter circuits

```c
int MUX_checkLines(uint32_t *data_panel_X, uint32_t *data_panel_Y){
    // group number and detector line number
    int nLine, nGroup = 0;
    const int size_data = 32; // set number of elements in data array
    const int lines_per_group = 32; // set number of detector lines per group

    GPIOB->ODR &= (uint16_t) 0xFFFE; // enable MUX (enable/disable pin PB4)

    for (nGroup = 0; nGroup < size_data; ){ // check each group of detector lines
        for (nLine = 0; nLine < lines_per_group; ){ // check each detector line in the group
            MUX_setAddress(nGroup, nLine); // update MUX address

            // add delay: wait 5 ms (for transient voltage on detector line to settle after MUX switching)
            Delay(0x00FF);

            // read pin PC4 status (X-axis) and check if current line is broken (low voltage)
            if ( !(GPIOC->IDR & 0x00000010) ){
                data_panel_X[nGroup] |= ((uint32_t) (1 << nLine)); // line broken: set bit in data array
            }

            // read pin PC5 status (Y-axis) and check if current line is broken (low voltage)
            if ( !(GPIOC->IDR & 0x00000020) ){
                data_panel_Y[nGroup] |= ((uint32_t) (1 << nLine)); // line broken: set bit in data array
            }

            nLine++;
        }
        nGroup++;
    }
    GPIOB->ODR |= (uint16_t) 0x0010; // disable MUX (enable/disable pin PB4)

    return 0;
}
```
@param enable_state enable or disable

Enables/disables the pulse detector and filter circuits (for power saving).

@return execution status (NOT IMPLEMENTED YET)

*/

int Shutdown_EnableDisable(ShutdownState_TypeDef enable_state){
  if (enable_state == SHUTDOWN_ENABLE)
  {
    GPIOA->ODR |= (uint16_t) 0x0002; // set pin PA1 (SHUTDOWN)
  }
  else if (enable_state == SHUTDOWN_DISABLE){
    GPIOA->ODR &= (uint16_t) 0xFFFF; // clear pin PA1 (SHUTDOWN)
  }
  else {
    //USART_printString("ERROR! SHUTDOWN: unknown argument");
    return 0;
  }

  //
  //*
  @brief Executes a specified action flag
  @param action_flag action to be executed

  Takes an integer as input and executes a corresponding action.

  @return execution status (NOT IMPLEMENTED YET)
  */
  int execute_action(int action_flag){
    int ERROR_state = 0; // no error

    switch (action_flag)
    {
      case USART_READ_ACTION_FLAG_INDEX:
        USART_call_command(USART_getBufferChar()); // pass buffer to command function
        break;

      case PULSE_DETECT_ACTION_FLAG_INDEX:
        USART_printString("PULSE_DETECT_ACTION\called\n");
        //PulseDetector_Reset();
        GPIOD->ODR |= 0x2000; // set bit PD13 (turn ON orange LED)
        break;

      case PERIODIC_SCAN_ACTION_FLAG_INDEX:
        GPIOD->ODR ^= (uint16_t) 0x4000; // toggle PD14 (red LED = RTC wakeup interrupts and running code)
        break;

      default:
        USART_printString("unknown\action\flag\called\_\Received\:\_*");
        ERROR_state = 1;
    }

    return ERROR_state;
  }

  //
  //*
  @brief Resets the pulse detector output

  When an impact pulse is detected, output (logic high), the pulse detector
  will be latched until this reset function is called.

  @return execution status (NOT IMPLEMENTED YET)
  */
  int PulseDetector_Reset(void){
    // toggle reset pin
    /**if (GPIOB->ODR & 0x0004)
```c
if (state == SC_PROTECTION_AUTOMATIC) {
    // configure short-circuit pin PA2 as input
    GPIOA->MODER &= (uint32_t) 0xFFFFFCF; // use pin PA2 as input
    GPIOA->PUPDR &= (uint32_t) 0xFFFFFCF; // no pull-up/down resistor on pin PA2
}
else {
    // configure short-circuit pin PA2 as output
    GPIOA->MODER |= (uint16_t) 0x00FF; // use pin PA2 as output
    GPIOA->OTypeR &= (uint16_t) 0xF0; // use push-pull output on pin PA2
    GPIOA->OSPEEDR &= (uint32_t) 0xFFFFFCF; // use 2 MHz clock freq. on pin PA2
    GPIOA->PUPDR &= (uint32_t) 0xFFFFFCF; // no pull-up resistor on pin PA2

    if (state == SC_PROTECTION_ENABLE) {
        GPIOA->ODR |= (uint16_t) 0x0004; // set pin PA2 (short-circuit protection ON)
    }
    else if (state == SC_PROTECTION_DISABLE) {
        GPIOA->ODR &= (uint16_t) 0xFFFB; // clear pin PA2 (short-circuit protection OFF)
    }
    else {
        USART_printString("ERROR! \_SC-PROTECTION\_\_unknown\_argument");
    }
}
return 0;
}
```

@file SOLID.h
@author Morten Olsen

C_source_code/SOLID.h
```c
#ifndef SOLID_H
#define SOLID_H

// includes
#include "stm32f4xx.h"

/*
** define constants for action flag indexes: used access corresponding flag in action_flags[] array*/
define USART_READ_ACTION_FLAG_INDEX ((int) 1)
define PERIODIC_SCAN_ACTION_FLAG_INDEX ((int) 2)
define PULSE_DETECT_ACTION_FLAG_INDEX ((int) 3)

/*
** global variable containing pending action flags*/
extern int action_flags[10];

/*
** global data arrays for storing info on broking detector lines*/
extern uint32_t impact_data_panel1_X[32], impact_data_panel1_Y[32];

/*
** enum type definition for possible shutdown pin states*/
typedef enum {
    SHUTDOWN_ENABLE=1,
    SHUTDOWN_DISABLE=0
} ShutdownStateTypeDef;

/*
** enum type definition for possible Short-Circuit Protection pin states*/
typedef enum {
    SC_PROTECTION_AUTOMATIC=2,
    SC_PROTECTION_ENABLE=1,
    SC_PROTECTION_DISABLE=0
} ShortCircuitProtectionTypeDef;

// function declarations
int Shutdown_EnableDisable(ShutdownStateTypeDef enable_state);
int PulseDetector_Reset(void);
int ShortCircuit_EnableDisable(ShortCircuitProtectionTypeDef state);
int SOLID_Init(void);
void Delay(__IO uint32_t nCount);
int execute_action(int action_flag);
int check_pending_action_flags(void);
int MUX_checkLines(uint32_t *data_panel_X, uint32_t *data_panel_Y);
int MUX_setAddress(int group_num, int line_num);
#endif /* SOLID_H */
```

F.1.3 SOLID_adc.c

```c
// SOLID_adc.c

C_source_code/SOLID_adc.c
```

The ADC is mainly used as troubleshooting functions by reading exact values (voltage) of MUX outputs, if these show inconsistent behaviour during normal operation.

(C) COPYRIGHT 2013 Deutches Zentrum fur Luft und Raumfahrt(DLR)
```c
#include "stm32f4xx.h"
#include "SOLID_adc.h"
#include "SOLID.h"

/*
* ------------------ SOLID ADC FUNCTIONS ------------------
*/

/**
@brief Configures the ADC
@return execution status (NOT IMPLEMENTED YET)
*/
int Config_ADC1()
{
    SCB->CPACR |= 0x000F0000; // enable Floating Point Unit (set bit 20–23 (CP10 and CP11 Full Access))
    ADC->CCR &= 0xFFFFF000; // ADC clock is PCLK2 / 2
    RCC->APB2ENR |= 0x00000001; // enable the ADC1 clock
    // reset ADC1 configuration
    ADC1->CR1 &= 0x00000000; // use 10-bit ADC1 resolution
    ADC1->CR2 &= 0x000000FF; // set EOCS bit = EOC is set at completion of each channel scan
    // set channel sampling times (Tconv = t_sample + 12 ADCCLK cycles)
    ADC1->SMPR2 |= 0x00000012; // set sampling time of channel '0' and '1' to 28 cycles
    ADC1->SMPR1 |= 0x00000008; // set sampling time of channel '16' and '17' to 144 cycles
    // use DMA?
    return 0;
}

/**
@brief Reads the output voltage of the two addressed MUX lines (X and Y axis)
Uses the ADC to read the MUX output value. This is mainly intended as a troubleshooting function to test correction functionality of the detector line circuit and MUXs.
@return Pointer to an array with the two ADC output values
*/
int* ADC_readMuxInput()
{
    int i, data[2];
    int* data_p = data;
    ADC1->CR2 |= 0x00000001; // enable ADC1
    // select conversion channels and order
    ADC1->SQR1 |= 0x00010000; // perform 2 conversions for each ADC scan sequence
    ADC1->SQR3 &= 0x00000000; // clear register
    ADC1->SQR3 |= 0x0000000E; // select ADC channel '14' to be the 1st channel in the scan sequence
    ADC1->SQR3 |= 0x0000000C; // select ADC channel '15' to be the 2nd channel in the scan sequence
    ADC1->CR1 |= 0x00000010; // use SCAN mode (convert multiple channels at a time)
    // ----------- connect MUX input pins to ADC alternate function -----------
    // config MUX read input pins: 1 x X-axis
    GPIOD->MODER |= (uint32_t) 0x00000300; // use pin PC4 as analog pin
    // config MUX read input pins: 1 x Y-axis
    GPIOD->MODER |= (uint32_t) 0x00000C00; // use pin PC5 as analog pin
    ADC1->CR2 |= 0x40000000; // start conversion
    for (i = 0; i < 2; i++)
    {
        while( !(ADC1->SR & 0x00000002) ){} // wait until EOS bit channel conversion is complete
        data[i] = (0x00000FFF & ADC1->DR); // read converted value
    }
    return data_p;
}
```
ADC1->CR1 &= 0xFFFFFEFF; // clear the SCAN bit
ADC1->CR2 &= 0xFFFFFFF0; // disable ADC1

/* reconfigure MUX input pins to normal inputs */

// config MUX read input pins: 1 x X-axis
GPIOC->MODER &= (uint32_t) 0xFFFFFCFF; // use pin PC4 as input

// config MUX read input pins: 1 x Y-axis
GPIOC->MODER &= (uint32_t) 0xFFFFF3FF; // use pin PC5 as input

return data_p;
}

/**
@brief Reads the value of the internal reference voltage

Reads and returns the value of the internal reference voltage source.

TO BE DONE: so far the values received are incorrect.

@return Returns the value (in volts) of the reference voltage
*/
float ADC_getRefVoltage(){
  int data;
  ADC1->CR2 |= 0x00000001; // enable ADC!
  ADC->CCR |= 0x00800000; // enable temperature sensor and reference voltage sensor
  ADC1->CR1 |= 0xFFFFFEFF; // disable SCAN mode (clear SCAN bit). only scan one channel at a time

  ADC1->SQR1 &= 0xFFFF0000; // perform 1 conversion for each ADC scan sequence
  ADC1->SQR3 &= 0xC0000000; // clear register
  ADC1->SQR3 |= 0x00000011; // select ADC channel '17' to be converted

  ADC1->CR2 |= 0x40000000; // start conversion (set SWSTART bit)
  while( !(ADC1->SR & 0x00000002) ){} // wait until EOS bit channel conversion is complete
  data = (0x0000FFFF & ADC1->DR); // read converted value
  data = ADC_data2Volt(data);
  ADC->CCR &= 0xFFFF7FFF; // disable temperature sensor and reference voltage sensor
  ADC1->CR2 &= 0xFFFFF000; // disable ADC!

  return data;
}

/**
@brief Converts ADC output value into volts

@param adc_val the raw output value of the ADC

Converts ADC output value to the corresponding measured voltage
ADC range = (Vref+) - (Vref-) = 3.3V - 0V

Sensed voltage is calculated as: V_sense = (adc_range / adc_resolution) * adc_val

@return
*/
float ADC_data2Volt(int adc_val){
  float v_sense;
  int adc_resolution = 1024; // valid for 10-bit ADC1 resolution which is set in ADC1->CR1 register.
  // TO BE IMPLEMENTED: above value should automatically be set by reading the configuration register
  int adc_voltage_range = 3;
  v_sense = ((float) (adc_val * adc_voltage_range) / adc_resolution);

  return v_sense;
}
@brief Reads the microcontroller temperature

Reads the voltage of the internal temperature sensor output and returns the value of the corresponding
temperature in celsius. Temperature voltage ADC input address is ADC1_IN16 (ADC1_IN18 on stm32f42x).

Conversion range is between 1.8 and 3.6 V (STM32F407 datasheet).

TO BE DONE: read temperature values does not seem to be correct. (not problem
with conversion formula which is already tested but with the read voltage value).

@return Returns the microcontroller temperature (degree celsius)

float ADC_getTemperature(){
    float data_float;
    int data_int;

    // temperature = ((Vsense - V25)/Avg_Slope) + 25
    float V25 = 0.76; // Vsense @ 25C
    float avg_slope = 2.5; // in units of mV/C

    ADC1->CR2 |= 0x00000001; // enable ADC1
    ADC->CCR |= 0x00000000; // enable temperature sensor and reference voltage sensor
    ADC1->CR1 &= 0xFFFFFEFF; // disable SCAN mode (clear SCAN bit). Scan one channel at a time

    // wait till startup completed < 10 us
    Delay(0x1FFFF);
    ADC1->SQR1 &= 0xFFF0FFFF; // perform 1 conversion for each ADC1 scan sequence
    ADC1->SQR3 &= 0xC0000000; // clear register
    ADC1->SQR3 |= 0x00000010; // select ADC1 channel "16" to be converted

    ADC1->CR2 |= 0x40000000; // start conversion (set SWSTART bit)
    while( !(ADC1->SR & 0x00000002) ){} // wait until EOS bit channel conversion is complete
    data_int = (0x0000FFFF & ADC1->DR); // read converted value

    ADC->CCR &= 0xFFFFFEFF; // disable temperature sensor and reference voltage sensor
    ADC1->CR2 &= 0xFFFFF000; // disable ADC1

    data_float = ((ADC_data2Volt(data_int) - V25) / avg_slope ) + 25; // convert temperature value

    return data_float;
}

@end

C_source_code/SOLID_adc.h

/**
 * @file    SOLID_adc.h
 * @author  Morten Olsen
 * @version 1.0
 * @date    08-August-2013
 * @brief   Header for SOLID ADC functions
 *
 (C) COPYRIGHT 2013 Deutches Zentrum fur Luft und Raumfahrt(DLR)
 */

// include guard
#ifndef SOLID_adc_H
#define SOLID_adc_H

#include "stm32f4xx.h"

int Config_ADC1(void);
int* ADC_readMuxInput(void);
float ADC_getRefVoltage(void);
float ADC_getTemperature(void);
float ADC_data2Volt(int adc_val);

F.1.4 SOLID_gpio.c

#include "stm32f4xx.h"

//---SOLID_GPIO_FUNCTIONS---*

//@brief Configures the GPIOs
Sets up all the General Purpose Input/Output (GPIO) pins of the microcontroller. These are used to:
- Create interrupt on pulse detector output
- Control shutdown of analog circuits
- Read MUX outputs
- Read sun detector outputs
- Control MUX addresses
- Reset pulse detector
- Control/read short-circuit protection status
- Configure USART3 and CAN pins
- Configure ADC input pin(s)

@return execution status (NOT IMPLEMENTED YET)

int Config_GPIO(){
RCC->AHB1ENR |= (uint32_t) 0x0000001F; // enable peripheral clocks on ports GPIOA to GPIOE

// reset settings (factory values are non-zero)
GPIOB->MODER = (uint32_t) 0x00000000; // all pins are inputs (high impedance)
GPIOB->PUPDR = (uint32_t) 0x00000000; // No pull-up nor pull-down resistors

// configure 1 x wakeup pin (blue push-button on Discovery board)
GPIOA->MODER &= (uint32_t) 0xFFFFFFF1; // use pin PA0 as input
GPIOA->PUPDR &= (uint32_t) 0xFFFFFFF0; // no pull-up resistors on pin PA0

// configure 1 x shutdown control pins
GPIOA->MODER |= (uint32_t) 0x00000004; // use pin PA1 as output
GPIOA->OTYPER &= (uint16_t) 0xFFFFD; // use push-pull output on pin PA1
GPIOA->OSPEEDR &= (uint32_t) 0xFFFFFFF3; // use 2 MHz clock freq. on pin pin PA1
GPIOA->PUPDR &= (uint32_t) 0xFFFFFFF3; // no pull-up resistors on pin PA1
GPIOA->ODR = (uint16_t) 0x0002; // pin 1 high (ON)

// configure 1 x short-circuit monitor pins
// configure green LED (to indicate wakeup pin)
GPIOA->MODER &= (uint32_t) 0xFFFFFCF; // use pin PA2 as input
GPIOA->PUPDR &= (uint32_t) 0xFFFFFCF; // no pull-up/down resistors on pin PA2

// configure 1 x sun detect inputs
GPIOA->MODER &= (uint32_t) 0xFFFFF3F; // use pin PA3 as input
GPIOA->PUPDR &= (uint32_t) 0xFFFFF3F; // no pull-up/down resistors on pin PA3

// configure 1 x pulse reset output
GPIOB->MODER |= (uint32_t) 0x00000010; // use pin PB2 as output
GPIOB->OTYPER &= (uint16_t) 0x0FFB; // use push-pull outputs on pin PB2
GPIOB->OSPEEDR &= (uint32_t) 0xFFFFF0FF; // 2 MHz peripheral clock freq. on pin PB2
GPIOB->PUPDR &= (uint32_t) 0xFFFFF3FF; // no pull-up/down resistors on pin PB2
GPIOB->ODR &= (uint16_t) 0xFFFC; // reset inactive

// configure 1 x pulse detect input
GPIOB->MODER &= (uint32_t) 0xFFFFF3F; // use pin PB1 as input
GPIOB->PUPDR |= (uint32_t) 0x00000080; // use pull down resistor on pin PB1

// config MUX control output pins: 1 x enable/disable(PB4), 1 x write enable/disable(PB5)
GPIOB->MODER |= (uint32_t) 0x00000500; // use pin PB4(EN) to PB5(WR) as outputs
GPIOB->OTYPER &= (uint16_t) 0xFFF0FF0F; // use push-pull outputs on pin PB4(EN) to PB5(WR)
GPIOB->OSPEEDR &= (uint32_t) 0xFFFFF0FF; // 2 MHz peripheral clock freq. on pin PB4(EN) to PB5(WR)
GPIOB->PUPDR &= (uint32_t) 0xFFFFF0FF; // no pull-up/down resistors on pin PB4(EN) to PB5(WR)
GPIOB->ODR &= (uint16_t) 0xFFFC; // disable both

// config MUX control output pins: 1 x CS (PB7)
GPIOB->MODER &= (uint32_t) 0x00004000; // use pin PB7 as output
GPIOB->OTYPER &= (uint16_t) 0xFFFFF3FF; // use push-pull output on pin PB7
GPIOB->OSPEEDR &= (uint32_t) 0xFFFFF3FF; // 2 MHz peripheral clock freq. on pin PB7
GPIOB->PUPDR &= (uint32_t) 0xFFFFF3FF; // no pull-up/down resistor on pin PB7
GPIOB->ODR &= (uint16_t) 0xFF7F; // CS output low

// config MUX control output pins: 5 x output address
GPIOC->MODER &= (uint32_t) 0x55400000; // use pin PB11 to PB15 as outputs
GPIOC->OTYPER &= (uint16_t) 0x07FF; // use push-pull outputs on pin PB11 to PB15
GPIOC->OSPEEDR &= (uint32_t) 0x003FFF; // 2 MHz peripheral clock freq. on pin PB11 to PB15
GPIOC->PUPDR &= (uint32_t) 0x003FFF; // no pull-up/down resistors on pin PB11 to PB15
GPIOC->ODR &= (uint16_t) 0x07FF; // set initial address to '0'

// config MUX control output pins: 5 x input address
GPIOE->MODER &= (uint32_t) 0x00005540; // use pin PE3 to PE7 as outputs
GPIOE->OTYPER &= (uint16_t) 0xFFF07F; // use push-pull outputs on pin PE3 to PE7
GPIOE->OSPEEDR &= (uint32_t) 0xFFFF007F; // 2 MHz peripheral clock freq. on pin PE3 to PE7
GPIOE->PUPDR &= (uint32_t) 0xFFFF007F; // no pull-up/down resistors on pin PE3 to PE7
GPIOE->ODR &= (uint16_t) 0xFFFF007F; // set initial address to '0'

// configure 2 x CAN1-bus pins
GPIOD->AFR[0] &= (uint32_t) 0x000000A4; // Alternative function on PD0(CAN1-RX) , PD1(CAN1-TX)
GPIOD->AFR[1] &= (uint32_t) 0x00000099; // Alternative function AF9(CAN1): PD0(CAN1-RX) , PD1(CAN1-TX)
GPIOD->OTYPER &= (uint16_t) 0xFFFFFC; // Use push-pull outputs on PD0(CAN1-RX) and PD1(CAN1-TX)
GPIOD->OSPEEDR &= (uint32_t) 0xFFFFF0FF; // 2 MHz peripheral clock freq. on PD0(CAN1-RX) and PD1(CAN1-TX)
GPIOD->PUPDR &= (uint32_t) 0xFFFFF0FF; // No pull-up/down resistors on PD0(CAN1-RX) and PD1(CAN1-TX)

// configure 2 x USART3 pins
GPIOD->AFR[0] &= (uint32_t) 0x00004000; // Alternative function on PD8(USART3-TX) , PD9(USART3-RX)
GPIOD->AFR[1] &= (uint32_t) 0x00000077; // Alt. function AF7(USART3): PD8(USART3-TX) , PD9(USART3-RX)
GPIOD->OTYPER &= (uint16_t) 0x0EFF; // use push-pull outputs on pins PD8(USART3-TX)
GPIOD->OTYPER &= (uint16_t) 0x0FDF; // use push-pull outputs on pins PD9(USART3-RX)
GPIOD->OSPEEDR &= (uint32_t) 0xFFFFF0FF; // 2 MHz clock freq. on PD8(USART3-TX) , PD9(USART3-RX)
GPIOD->PUPDR &= (uint32_t) 0xFFFFF0FF; // Use pull-up resistors on PD8(USART3-TX) , PD9(USART3-RX)

// configure green LED (to indicate wakeup pin)
/* LED GPIOs (only for testing purpose on STM32-Discovery board): */

// configure 2 x X-axis
GPIOC->MODER &= (uint32_t) 0xFFFFFCF; // Use pin PC4 as input
GPIOC->PUPDR &= (uint32_t) 0x00000200; // Use pull-down resistor on pin PC4

// configure 2 x Y-axis
GPIOC->MODER &= (uint32_t) 0xFFFFFCF; // Use pin PC5 as input
GPIOC->PUPDR &= (uint32_t) 0x00000800; // Use pull-down resistor on pin PC5

// configure green LED (to indicate wakeup pin)
```c
GPIOD->MODER |= (uint32_t) 0x10000000; // use pin 12 as output
GPIOD->OTYPER &= (uint16_t) 0xFF; // use push–pull output on pin 12
GPIOD->OSPEEDR &= (uint32_t) 0xF0000000; // use 2 MHz clock freq. on pin 12
GPIOD->PUFDR &= (uint32_t) 0xF0000000; // no pull-up resistors on pin 12
GPIOD->ODR &= (uint16_t) 0xFF; // clear bit PD12 (turn OFF green LED)

// configure orange LED (to indicate external interrupt i.e. pulse detection)
GPIOD->MODER |= (uint32_t) 0x00000000; // use pin 13 as output
GPIOD->OTYPER &= (uint16_t) 0xFF; // use push–pull output on pin 13
GPIOD->OSPEEDR &= (uint32_t) 0xF0000000; // use 2 MHz clock freq. on pin 13
GPIOD->PUFDR &= (uint32_t) 0xF0000000; // no pull-up resistors on pin 13
GPIOD->ODR &= (uint16_t) 0xFF; // clear bit PD13 (turn OFF orange LED)

// configure red LED (to indicate RTC wakeup interrupt)
GPIOD->MODER |= (uint32_t) 0x10000000; // use pin 14 as output
GPIOD->OTYPER &= (uint16_t) 0xFF; // use push–pull output on pin 14
GPIOD->OSPEEDR &= (uint32_t) 0xF0000000; // use 2 MHz clock freq. on pin 14
GPIOD->PUFDR &= (uint32_t) 0xF0000000; // no pull-up resistors on pin 14
GPIOD->ODR &= (uint16_t) 0xFF; // clear bit PD14 (turn OFF red LED)

return 0;
}

/F.1.5
SOLID_interrupts.c

C_source_code/SOLID_gpio.h

/∗∗
@file    SOLID_gpio.h
@author   Morten Olsen
@version 1.0
@date    08–August–2013
@brief   SOLID header file for GPIO functions

(C) COPYRIGHT 2013 Deutches Zentrum fur Luft und Raumfahrt (DLR) */

// include guard
#ifndef SOLID_gpio_H
#define SOLID_gpio_H

int Config_GPIO(void);
#endif /* SOLID_gpio_H */

F.1.5 SOLID_interrupts.c

C_source_code/SOLIDInterrupts.c

/∗∗
@file    SOLID_interrupts.c
@author   Morten Olsen
@version 1.0
@date    08–August–2013
@brief   SOLID interrupt routine handlers

SOLID interrupt handlers. The routines are kept very short and often does just
set a pending action flag and returns to main loop. This is done to minimize
risk of issues by having several large interrupt routines running simultaneously.

(C) COPYRIGHT 2013 Deutches Zentrum fur Luft und Raumfahrt (DLR) */

#include ’stm32f4xx.h’
#include ’SOLID_uart.h’
#include ’SOLID.h’
```
#include "SOLID_pwr.h"
#include "misc.h"

/*
  Private define
   *  
  */
#define AIRCR_VECTKEY_MASK ((uint32_t)0x05FA0000)

SOLID INTERRUPT HANDLERS
/**
 @brief Enables the various interrupts and sets priorities
 Enables the various interrupt sources and configures the priorities.
 @{
 return execution status (NOT IMPLEMENTED YET)
 */
int Config_Interrupts(){

    // Set vector table location and Offset.
    SCB->VTOR = NVIC_VectTab_FLASH | (0x00 & (uint32_t)0xFFFF080);

    // Set group priority: no group priority, 16 subpriorities
    SCB->AIRCR = AIRCR_VECTKEY_MASK | NVIC_PriorityGroup_0;

    NVIC->IP[9] |= 0x01000000;  // set priority of interrupt channel ’39’(USART3) to ’1’
    NVIC->IP[0] &= 0x00FFFFF;  // set priority of interrupt channel ’3’(RTC_WKUP) to ’0’
    NVIC->IP[1] |= 0x00510000;  // set priority of interrupt channel ’6’(EXTI0) to ’1’
    NVIC->IP[1] |= 0x00100000;  // set priority of interrupt channel ’7’(EXTI1) to ’1’

    NVIC->ISER[0] |= 0x00000008; // enable RTC wakeup interrupt, EXTI line ’22’ (vector table position 3)
    NVIC->ISER[0] |= 0x00000040; // enable EXTI0 interrupt (vector table position: 6)
    NVIC->ISER[0] |= 0x00000080; // enable EXTI1 interrupt (vector table position: 7)
    NVIC->ISER[1] |= 0x00000080; // enable USART3 interrupt (vector table position: 39)

    return 0;
}
/**
 @brief Enables the external line interrupts
 Configures the connections for External Interrupt sources, sets the edge trigger (rising or falling) and clears the masking of the interrupts.
 @{
 return execution status (NOT IMPLEMENTED YET)
 */
int Config EXTI(){

    EXTI->EMR &= 0xFF700000;  // mask all lines for event requests (= reset configuration)

    // enable external interrupt for pulse detection:
    SYSCFG->EXTICR[0] |= 0x00000010;  // connect External interrupt line ’1’(EXTI1) to PB1
    EXTI->IMR |= 0x00000000;  // unmask EXTI1 -> enable pulse detection interrupt
    EXTI->RTSR |= 0x00000000;  // enable rising edge trigger on line ’1’ (pin PB1)

    // enable external interrupt for RTC wakeup:
    EXTI->IMR |= 0x00040000;  // unmask exti line ’22’ -> enable RTC wakeup interrupt line
    EXTI->RTSR |= 0x00040000;  // enable rising edge trigger on line ’22’ (RTC wakeup interrupt)

    // enable external interrupt for Wakeup pin (push–button on PA0):
    SYSCFG->EXTICR[0] &= 0xFFFF0000;  // connect External interrupt line ’0’(EXTI0) to PA0
    EXTI->IMR |= 0x00000001;  // unmask EXTI0 -> enable PWR Wakeup pin interrupt
    EXTI->RTSR |= 0x00000001;  // enable rising edge trigger on line ’0’ (Wakeup pin)

    return 0;
}
/**
 @brief RTC wakeup interrupt handler()
 Interrupt routine for the Real–Time Clock (RTC) wakeup event. This function
 is used to set an action flag for a perioded detector line scan.
 */
void RTC_WKUP_IRQHandler(void){

}
if (PWR->CSR & 0x00000002)  // check if SBF flag is set (wakeup occurred from Standby mode)
{
    // TO BE IMPLEMENTED: perform any additional initialization as required
    PWR->CR |= 0x00000008;  // clear SBF flag
}
// set actions to be performed at each wakeup event:
action_flags[PERIODIC_SCAN_ACTION_FLAG_INDEX] = 1;  // set corresponding action flag
EXTI->PR |= 0x00400000;  // clear interrupt flag
RTC->ISR &= 0xFFFFFBFF;  // clear wakeup interrupt flag
} //
/**
@brief USART interrupt handler

Interrupt routine for USART3 used when receiving new characters over the USART line. Calls a USART receive function and sets an action flag to indicate a new character is received.

TO BE DONE: Extend function such that multiple character commands can be accepted, including command parameters. Possibly use a cyclic FIFO buffer for this.
*/
void USART3_IRQHandler(void){
if (USART3->SR & 0x00000020)  // check for RXNE flag (new character received)
{
    USART_receive();  // function automatically resets interrupt flag
    action_flags[USART_READ_ACTION_FLAG_INDEX] = 1;  // set action flag
}
else{
    USART3->SR &= 0xFFFFFDFF;  // reset interrupt flag
}
} //
/**
@brief External line 0 interrupt handler

EXTI0 interrupt handler used for STM32-Disclosure board when the "blue" push-button is pressed. This button is configured as wakeup pin and will activate this interrupt routine which clears any power saving mode.
*/
void EXTI0_IRQHandler(void){
if (EXTI->PR & 0x00000001)  // check for pending interrupt flag (wakeup pin)
{
    EXTI->PR |= 0x00000001;  // clear interrupt flag
    ACTIVE_PWR_MODE = PWR_MODE_RUN;  // clear any power saving mode (return to RUN mode)
}
} //
/**
@brief External line 1 interrupt handler

Handles EXTI1 global interrupt request. This is used for pulse detection.

TO BE IMPLEMENTED: In future, the RTC Tamper events might replace this function.
*/
void EXTI1_IRQHandler(void){
if (EXTI->PR & 0x00000002)  // check for pending interrupt flag (pulse detection)
{
    action_flags[PULSE_DETECT_ACTION_FLAG_INDEX] = 1;  // set action flag
    EXTI->PR |= 0x00000002;  // clear interrupt flag
}
} //
/* ------------------------------------------------ END OF FILE ------------------------------------------------*/
/*@
@file SOLID_interrupts.h
@author Morten Olsen
@version 1.0
@date 08-August-2013
@brief SOLID header file for interrupt handlers and functions

(C) COPYRIGHT 2013 Deutches Zentrum fur Luft und Raumfahrt (DLR)
*/

// include guard
#ifndef SOLID_interrupts_H
#define SOLID_interrupts_H

int Config_Interrupts(void);
int Config_EXTI(void);

#endif /* SOLID_interrupts_H */

F.1.6 SOLID_pwr.c

/*@
@file SOLID_pwr.c
@author Morten Olsen
@version 1.0
@date 08-August-2013
@brief Functions for control of microcontroller power saving modes

These functions are used to control the power saving modes of the microcontroller.

(C) COPYRIGHT 2013 Deutches Zentrum fur Luft und Raumfahrt (DLR)
*/

#include "SOLID_usart.h"
#include "stm32f4xx.h"
#include "SOLID_pwr.h"

/∗∗ declare global variable for control of active power mode*/
enum PWR_MODES ACTIVE_PWR_MODE = PWR_MODE_RUN; // initialized not to enter any power saving mode

/∗-------------------------- SOLID POWER SETTING FUNCTIONS--------------------------*/

/__@
@brief Configures general behaviour of the power saving modes

Sets the initial behaviour of the power saving modes

@return execution status (NOT IMPLEMENTED YET)
__*/
int Config_Power()
{
// configure behaviour of power down modes:
PWR->CR |= 0x00000200; // set FPDS to power down flash memory during Stop/Standby mode
PWR->CR |= 0x00000001; // set LPDS to power down voltage regulator during Stop/Standby mode
SCB->SCR &= 0xFFFFFDFF; // clear SLEEPONEXIT bit: do not enter sleep mode when no pending interrupts
SCB->SCR &= 0xFFFFFFF0; // clear SEVONPEND bit: pending interrupt is not wakeup event

//RCC->APB1LPENR |= 0x00000000; // enable USART3 clock during sleep mode

return 0;
}

//
/__@
@brief Changes the current power saving mode

return 0;
*/
@param mode the power saving mode to be activated

Sets the corresponding power saving mode and calls the __WFI() (Wait For Interrupt) routine to enter power saving mode.

@return

*/

void PWR_EnterPowerSavingMode(PWR_MODE_TypeDef mode){

    if (mode == PWR_MODE_SLEEP) {
        SCB->SCR &= 0xFFFFFB;  // clear SLEEPDEEP bit
    }

    else if (mode == PWR_MODE_STOP) {
        SCB->SCR |= 0x00000004; // set SLEEPDEEP bit; enter Stop or Standby mode upon WFI() call
        PWR->CR |= 0xFFFFFD;  // clear PDDS bit; enter Stop mode upon WFI() call
        // all pending interrupts must be cleared to enter stop mode
    }

    else if (mode == PWR_MODE_STANDBY) {
        SCB->SCR |= 0x00000004; // set SLEEPDEEP bit
        PWR->CR |= 0x00000020; // set PDDS bit; enter Standby mode upon WFI() call
        // upon exit: resets all registers except PWR->CSR. PWR->CSR.SBF indicates exit from standby mode
    }

    else {
        //USART_printString ("ERROR! PWR: unknown power saving mode\n");
    }

    PWR->CSR |= 0x00000100; // set WKUP (wakeup pin = blue pushbutton on Discovery board)

    PWR->CR |= 0x00000004; // clear WUP (Wakeup flag)
    RTC->ISR &= 0xFFFFBFFFF; // clear RTC WUTF flag

    /* This option is used to ensure that store operations are completed */
    #if defined (__CC_ARM )
        __force_stores();
    #endif

    __WFI();  // execute Wait For Interrupt (WFI) instruction (enter power saving mode)
}

/*---------------------------------------------------- END OF FILE ----------------------------------------------------*/
/** global variable storing the active power saving mode */
extern enum PWR_MODES ACTIVE_PWR_MODE;

int Config_Power(void);
void PWR_EnterPowerSavingMode(PWR_MODETypeDef mode);

@endef /* SOLID_pwr_H */

F.1.7 SOLID_rcc.c

#include "stm32f4xx.h"

C_source_code/SOLID_rcc.c

/*---------------------------------- SOLID CLOCK SETTING FUNCTIONS ----------------------------------*/

/*
@brief Configures the system clocks

Configures the system clock to HSI (16 MHz High Speed Internal oscillator).
This function is called from the startup assembly file before the main function is called.

@return execution status (NOT IMPLEMENTED YET)
*/

int Config_SysClk(){

  // Disable all clock interrupts
  RCC->CIR &= 0x000000FF;

  // Reset CFGR register - selects HSI as system clock
  RCC->CFGR &= 0x00000300;

  // Reset the RCC clock configuration to the default reset state
  RCC->CR &= (uint32_t) 0xFFFFFFFF;

  // Set HSION bit
  RCC->CR |= (uint32_t) 0x00000001;
  while( !(RCC->CR & 0x00000002) ){} // wait until internal oscillator ready (HSIRDY set)

  RCC->APB1ENR |= 0x10000000; // enable clock for Power interface (necessary RTC write protection unlock)

  RCC->APB2ENR |= 0x00000000; // enable SYSCFG clock

  // Configure Flash prefetch, Instruction cache, Data cache and wait state
  FLASH->ACR = FLASH_ACR_ICEN | FLASH_ACR_Dcen | FLASH_ACR_LATENCY_5WS;
  //FLASH->ACR &=~FLASH_ACR_PrfTen; // Disable prefetch buffer to decrease power consumption

  return 0;
}

="/"
int Config_SysClk(void);

F.1.8 SOLID_rtc.c

#include "stm32f4xx.h"
#include "SOLID_rtc.h"
#include "SOLID.h"

TIME_DATE_TypeDef INIT_TIME_DATE;

/* ------------------------------- SOLID REAL-TIME CLOCK FUNCTIONS ------------------------------- */

/* @brief Configures the Real-Time-Clock. 
Sets up and enables the Real-Time Clock (RTC). The internal low-speed oscillator (LSI) is used.
TO BE DONE: Calibration function for the LSI. 
*/
int Config_RTC(){
    // after system reset:
    PWR->CR |= 0x00000100; // set DBP bit 8(allow write access to RTC registers)
    /* Clear Wakeup flag */
    PWR->CR |= 0x00000004; // clear wakeup flag
    // after power-on (enable write access to RTC registers):
    RTC->WPR |= 0xA0;
    RTC->WPR |= 0x53;
    if ((PWR->CSR & 0x00000002)) // Check if the StandBy flag is set
        {
            PWR->CSR |= 0x00000008; // Clear StandBy flag
            /* Disable the write protection for RTC registers */
            RTC->WPR = 0xA0;
            RTC->WPR = 0x53;
            RTC->ISR &= 0xFFFFFDFD; // Clear RSF flag , bit 5
            while ( !(RTC->ISR & 0x00000020) ){} // wait till RSF flag is set again (calendar values updated)
Enable the write protection for RTC registers */
RTC->WPR = 0xFF;
}
else {
    RCC->CSR |= 0x00000001; // enable LSI (internal low speed oscillator)
    while ( !(RCC->CSR & 0x00000002) ){} // wait until LSI oscillator is stable
    RCC->BDCR |= 0x00000000; // use LSI as RTC clock
    RCC->BDCR |= 0x00080000; // enable RTC clock (set bit 15)
/* Wait for RTC APB registers synchronisation (needed after start-up from Reset)*/
RTC->ISR &= 0xFFFFDFDF; // Clear Rsf flag, bit 5
while ( !(RTC->ISR & 0x00000020) ){} // wait until Rsf flag is set again (calendar values updated)
RTC->CR &= 0x00000000; // reset RTC configuration
RTC->CR &= 0xFFFFFBBF; // set time formatFmt to 24 hour/day
RTC->ISR |= 0x00000080; // set INIT bit (enter initialization mode)
while ( !(RTC->ISR & 0x00000040) ){} // wait until initialization mode is entered

// f_CK_SPRE = f_RTCCLK / ([PREDIV_S + 1] x [PREDIV_A + 1]) = 32 kHz / (128 x 256) = 0.98 Hz
RTC->PRER |= (255 & 0xFFFFF7FF); // set PREDIV_S = 255
// f_CK_APRE = f_RTCCLK / (PREDIV_A + 1) = 32 kHz / 128 = 248.1 Hz
RTC->PRER |= ((127 << 16) & 0xFFFF0000); // set PREDIV_A = 127
if ( !(RTC->ISR & 0x00000010) ) // check if calendar is initialized (INITS bit 4)
{
    /* set initial time and date to Monday 02:56:23 on 21st July 1969
    will only be executed after power-on reset, not after system reset */
    INIT_TIME_DATE_SEC = 0x23;
    INIT_TIME_DATE_MIN = 0x56;
    INIT_TIME_DATE_HOUR = 0x02;
    INIT_TIME_DATE_WEEKDAY = 0x01;
    INIT_TIME_DATE_DAY = 0x21;
    INIT_TIME_DATE_MONTH = 0x07;
    INIT_TIME_DATE_YEAR = 0x69;

    RTC_setTimeDate(&INIT_TIME_DATE);
}
// configure wakeup timer:
RTC->CR &= 0xFFFFFBFF; // clear WUTE bit 10 (disable wakeup timer)
while ( !(RTC->ISR & 0x00000004) ){} // wait until access to wakeup counter is allowed (WUTWF bit 2)
RTC->CR |= (uint32_t) 0x00000004; // set WUCKSEL[2:0] to '10' -> 1Hz (wakeup time from 1s to 18h)
RTC->WUTR &= (uint32_t) 0xFFFFF000; // clear WUTr periodic counter
RTC->WUTR |= (uint32_t) 0x00000008; // set wakeup interrupt interval to 8 /1Hz
RTC->CR |= (uint32_t) 0x00000000; // enable WUTIE (wakeup interrupt)
RTC->CR |= (uint32_t) 0x00000000; // enable WUTE bit 10 (periodic wakeup)

// start RTC
RTC->ISR &= 0xFFFFF7FF; // clear INIT bit (leave initialization mode)

// Enable the write protection for RTC registers
RTC->WPR = 0xFF;
}
return 0;
}*/
/**
@brief Converts BCD to decimal
@param bcd input value in binary-coded-decimal format

Converts two digit BCD time format to two digit decimal format

@return Decimal format of input BCD
*/
int Convert_BCD2Decimal(char bcd){
/*
int decimal = 0;

while (bcd >= 16) {
    decimal += 1; // increase digit for tens
    bcd -= 16;
} // set digit for tens
decimal += bcd; // add digit for units

return decimal;

//

/**
@brief RTC_getTimeDate(TIME_DATE_TypeDef* time_date)
@param time_date placeholder for the time and date values to be stored

Saves the current date and time into the input time_date struct.

@return execution status (NOT IMPLEMENTED YET)
*/
int RTC_getTimeDate(TIME_DATE_TypeDef* time_date){
    uint32_t tmp_time, tmp_date;

    RTC->ISR &= 0xFFFFFDF; // clear RSF bit 5
    while ( !(RTC->ISR & 0x00000020) ){} // wait until RSF bit is set again (calendar and time values updated)

    // read most recent time and date values
    tmp_time = RTC->TR;
    tmp_date = RTC->DR;

    // load time and date values into time_date struct
    time_date->SEC = (tmp_time & 0x0000007F);
    time_date->MIN = ((tmp_time & 0x000007F0) >> 8);
    time_date->HOUR = ((tmp_time & 0x0003F000) >> 16);
    time_date->DAY = (tmp_date & 0x0000003F);
    time_date->MONTH = ((tmp_date & 0x00001F00) >> 8);
    time_date->YEAR = ((tmp_date & 0x00FF0000) >> 16);

    return 0;
}

/**
@brief Updates the RTC to the input time and date values
@param time_date The time and date values to be set

Sets RTC the time and date values from the input values given.

@return execution status (NOT IMPLEMENTED YET)
*/
int RTC_setTimeDate(TIME_DATE_TypeDef* time_date){

    // set time
    RTC->TR = (uint32_t) ( (time_date->HOUR << 16) + (time_date->MIN << 8) + time_date->SEC );

    // set date
    RTC->DR = (uint32_t) ( (time_date->YEAR << 16) + (time_date->WEEKDAY << 13) + (time_date->MONTH << 8) +
                          time_date->DAY );

    return 0;
}

///**
@file     SOLID_rtc.h
@author   Morten Olsen
@version  1.0
@date     25−June−2013
@brief    SOLID header file for Real−Time−Clock functions

(C) COPYRIGHT 2013 Deutches Zentrum fur Luft und Raumfahrt(DLR)

#include guard
#ifndef SOLID_rtc_H
#define SOLID_rtc_H

/**
@brief Holds a set of date and time values
*/
typedef struct
{
    char SEC;
    char MIN;
    char HOUR;
    char WEEKDAY;
    char DAY;
    char MDNTH;
    char YEAR;
} TIME_DATE_TypeDef;

int Config_RTC(void);
int Convert_BCD2Decimal(char bcd);
int RTC_getTimeDate(TIME_DATE_TypeDef* time_date);
int RTC_setTimeDate(TIME_DATE_TypeDef* time_date);
#endif /* SOLID_rtc_H */

F.1.9  SOLID_usart.c

C_source_code/SOLID_usart.c

/**
@brief SOLID USART FUNCTIONS
*/

/* declare local variable for USART read buffer*/
char global_USART_buffer;

rophy configurations for testing

The SOLID UART functions are only intended for testing purpose. The are not intended to be included in the final flight software. UART is a more simple and easy interface to use than the CAN−bus, hence used for initial testing.

(C) COPYRIGHT 2013 Deutches Zentrum fur Luft und Raumfahrt(DLR)

#include "stm32f4xx.h"
#include "SOLID.h"
#include "SOLID_rtc.h"
#include "SOLID_pwr.h"
#include <stdio.h>
#include "SOLID_usart.h"
#include "SOLID_adc.h"

phy declare local variable for USART read buffer*/
char global_USART_buffer;

---------------------------------------------------------------------- SOLID USART FUNCTIONS ----------------------------------------------------------------------*/

/**
@brief Configures the UART controller.

Set up USART3 (only used for testing/debug purposes)
Oversampling = 16
Word length, 1 start bit, 8 data bits, 1 stop bit
Wakeup = Idle line
No parity control
Interrupt on received character
Three sample bit method
Full duplex
Tx/Rx baud = fCK/(8 x [2−OVER8] x USARTDIV) = 16 MHz /(8 x 2 x 104.167) = 9.6 kbps

@return execution status (NOT IMPLEMENTED YET)
*/
int Config_USART(){
    int DIV_Mantissa, DIV_Fraction;
    RCC->APB1ENR |= (uint32_t) 0x00040000; // enable UART3 peripheral clock, set bit 18
    USART3->CR2 &= (uint32_t) 0xFFFF809F; // reset pin 0 to 15
    USART3->CR1 &= (uint32_t) 0xFFFF0000; // reset configuration
    USART3->CR1 |= (uint32_t) 0x00000008; // enable USART3 transmitter -> will send an idle frame
    USART3->CR1 |= (uint32_t) 0x00000004; // enable receiver and start searching for a start bit
    // reset configuration
    USART3->CR3 &= (uint32_t) 0xFFFFF000;
    DIV_Mantissa = 0x068; // 104
    DIV_Fraction = 0x3; // 16 * 0.167 = 2.672 -> 3.00
    USART3->BRR = ((uint32_t) (DIV_Mantissa << 4) + DIV_Fraction); // set baudrate to 9.6 kbps
    USART3->Ch1 |= (uint16_t) 0x2000; // enable USART3 (set bit 13)
    USART_printString ("SOLID_MCU initialized\n");
    return 0;
}

//
/**
@brief Executes a command corresponding to the input character.
@param cmd command to be executed

Reads an input char and executes a corresponding command.
TO BE IMPLEMENTED: This function should be able to take multiple character commands with optional command parameters.

@return execution status (NOT IMPLEMENTED YET)
*/
int USART_call_command(char cmd){
    char string_buffer[255];
    TIME_DATE_TypeDef time_date;
    int group_num, line_num, i, k; // for testing of MUXs
    static int j = 0;
    switch(cmd)
    {
    // enable shutdown
    case 'a':
        Shutdown_EnableDisable(SHUTDOWN_ENABLE);
        USART_printString("power_enabled\n");
        break;
    // disable shutdown
    case 'b':
        Shutdown_EnableDisable(SHUTDOWN_DISABLE);
        USART_printString("shutdown_enabled\n");
        break;
    // pulse reset
    case 'c':
        PulseDetector_Reset();
    }
}
// short-circuit protection: manual enable (pin pulled low)
case 'd':
    ShortCurrent_EnableDisable(SC_PROTECTION_ENABLE);
    USART_printString("short-circuit_protection_switch_manually_closed\n");
    break;

// short-circuit protection: manual disable (pin pulled high)
case 'e':
    ShortCurrent_EnableDisable(SC_PROTECTION_DISABLE);
    USART_printString("short-circuit_protection_switch_manually_opened\n");
    break;

// short-circuit protection: automatic (MCU pin floating -> used as input to determine state)
case 'f':
    ShortCurrent_EnableDisable(SC_PROTECTION_AUTOMATIC);
    USART_printString("short-circuit_protection_set_to-automatic\n");
    break;

// get time and date
case 'g':
    RTC_getTimeDate(&time_date); // update time and date struct
    // convert time and date formats from BCD to decimal format and outputs values on USART line
    sprintf(string_buffer, "Time[HHMMSS]: %02u:%02u:%02u\n", Convert_BCD2Decimal(time_date.HOUR), Convert_BCD2Decimal(time_date.MIN), Convert_BCD2Decimal(time_date.SEC));
    USART_printString(string_buffer);
    break;

// enter power saving Sleep mode
case 'h':
    ACTIVE_PWR_MODE = PWR_MODE_SLEEP;
    USART_printString("enabling_Sleep_mode\n");
    break;

// enter power saving Stop mode
case 'i':
    ACTIVE_PWR_MODE = PWR_MODE_STOP;
    USART_printString("enabling_Stop_mode\n");
    break;

// enter power saving Standby mode
case 'j':
    ACTIVE_PWR_MODE = PWR_MODE_STANDBY;
    USART_printString("enabling_Standy_mode\n");
    break;

// disable power saving modes (stay in RUN mode)
case 'k':
    ACTIVE_PWR_MODE = PWR_MODE_RUN;
    USART_printString("disabling_power_saving_modes\n");
    break;

// get MCU temperature
case 'l':
    sprintf(string_buffer, "MCU_temperature_is:%f_degC\n", ADC_getTemperature());
    USART_printString(string_buffer);
    break;

// get value of reference voltage
case 'm':
    sprintf(string_buffer, "MCU_reference_voltage_is:%f_V\n", ADC_getRefVoltage());
    USART_printString(string_buffer);
    break;
// get voltages of MUX outputs
switch ('n') {
    printf(string_buffer, "MUX_output.voltages.are:Ex=.%f, Ey=.%f\n", ADC_data2Volt(ADC_readMuxInput()[0]), ADC_data2Volt(ADC_readMuxInput()[1]));
    USART_printString(string_buffer);
    break;
}

switch ('o') {
    if (((GPIOC->IDR & 0x00000010) )){
        USART_printString("MUX_output.is.high\n");
    }
    else {
        USART_printString("MUX_output.is.low\n");
    }
    break;
}

switch ('p') {
    GPIOB->ODR &= (uint16_t) 0xFFEF; // enable MUX (PB4)
    USART_printString("MUX_enabled\n");
    group_num = 5;
    line_num = 4;
    MUX_setAddress(group_num, line_num);
    USART_printString("Group.set.to:5,line.set.to:4\n");
    break;
}

switch ('q') {
    group_num = 3;
    line_num = 3;
    MUX_setAddress(group_num, line_num);
    USART_printString("Group.set.to:3,line.set.to:3\n");
    break;
}

switch ('r') {
    USART_printString("Detector,line.scan.began.at:\n");
    RTC_gettimeDate(&time_date); // update time and date struct
    // convert time and date formats from BCD to decimal format and outputs values on USART line
    sprintf(string_buffer, "Time[HHMMSS]: %02u:%02u:%02u\n", Convert_BCD2Decimal(time_date.HOUR), Convert_BCD2Decimal(time_date.MIN), Convert_BCD2Decimal(time_date.SECOND));
    USART_printString(string_buffer);
    sprintf(string_buffer, "Date[DDMMYY]: %02u/%02u-%02u\n", Convert_BCD2Decimal(time_date.DAY), Convert_BCD2Decimal(time_date.MONTH), Convert_BCD2Decimal(time_date.YEAR));
    USART_printString(string_buffer);
    MUX_checkLines(impact_data_panel1_X, impact_data_panel1_Y);
    RTC_gettimeDate(&time_date); // update time and date struct
    USART_printString("Detector,line.scan.completed.at:\n");
    // convert time and date formats from BCD to decimal format and outputs values on USART line
    sprintf(string_buffer, "Time[HHMMSS]: %02u:%02u:%02u\n", Convert_BCD2Decimal(time_date.HOUR), Convert_BCD2Decimal(time_date.MIN), Convert_BCD2Decimal(time_date.SECOND));
    USART_printString(string_buffer);
    sprintf(string_buffer, "Date[DDMMYY]: %02u/%02u-%02u\n", Convert_BCD2Decimal(time_date.DAY), Convert_BCD2Decimal(time_date.MONTH), Convert_BCD2Decimal(time_date.YEAR));
    USART_printString(string_buffer);
    break;
}

switch ('s') {
    USART_printString("Impact.data,X-axis.reads:\n");
    for(i=0; i<32){
        sprintf(string_buffer, "%u\n", impact_data_panel1_X[i]);
        USART_printString(string_buffer);
        i++;
        break;
    }
for (i = 0; i < 32; ){
    sprintf(string_buffer, "%u\n", impact_data_panel1_Y[i]);
    USART_printString(string_buffer);
    i++;
}
break;

// list all available commands
case '? ' :
    USART_printString(" Possible/SOLID/MCU/commands/are:
");
    USART_printString(" 'a '/enable/shutdown\n");
    USART_printString(" 'b '/disable/shutdown\n");
    USART_printString(" 'c '/reset/SC-protection\n");
    USART_printString(" 'd '/manually.deactivate/SC-protection\n");
    USART_printString(" 'e '/manually.deactivate/SC-protection\n");
    USART_printString(" 'f '/set/SC-protection/to/automatic/control\n");
    USART_printString(" 'g '/get.time_and_date\n");
    USART_printString(" 'h '/enter.sleep_mode_upon.completion_of_tasks\n");
    USART_printString(" 'i '/enter.stop_mode_upon.completion_of_tasks\n");
    USART_printString(" 'j '/enter.standby_mode_upon.completion_of_tasks\n");
    USART_printString(" 'k '/disable.power_saving.modes\n");
    USART_printString(" 'l '/get.temperature\n");
    USART_printString(" 'm '/get.reference.voltage.value\n");
    USART_printString(" 'n '/read.out.MUX.outputs.with_ADC\n");
    USART_printString(" 'o '/get.MUX.output.value\n");
    USART_printString(" 'p '/enable_MUXs_and_set_addresses\n");
    USART_printString(" 'q '/set_MUXs_addresses\n");
    USART_printString(" 'r '/scan.all.detector_lines\n");
    USART_printString(" 's '/get_impact.data\n");
    break;

default :
    USART_printString("ERROR! USART:unknown_command\n");
}

return 0;

/**
@brief Prints and input number onto the UART line.
@param num number to be printed
Simple function to take an input integer and send it over the UART3.
@return execution status (NOT IMPLEMENTED YET)
*/
int USART_printNumber(int num){

    USART3->CR1 |= 0x8; // enable USART3 transmitter -> will send an idle frame
    USART3->DR = (num & (uint16_t)0x01FF); // will clear TXE bit
    while( !(USART3->SR & 0x04) ){} // wait until last transmission is complete (TC bit is set)
    USART3->CR1 &= 0xFFFFFFF7; // disable USART3 transmitter
    return 0;
}

/**
@brief Prints an input string to the UART line.
@param *str string to be printed
Takes an input string and sends it over the UART3.
@return execution status (NOT IMPLEMENTED YET)
*/
int USART_printString(char *str){

}
int i = 0;

USART3->CR1 |= 0x8; // enable USART3 transmitter will send an idle frame

while (str[i] != \0) {
    while (!((USART3->SR & 0x80))){} // wait until data register is empty (TXE bit is set)
    USART3->DR = str[i]; // will clear TXE bit
    i++;
}

while (!((USART3->SR & 0x40))){} // wait until last transmission is complete (TC bit is set)

USART3->CR1 &= 0xFFFFFFFF; // disable USART3 transmitter

return 0;
}

//
/**
 * @brief Returns stored character in UART buffer.

 * Function to return the character stored in the global UART read buffer.

 * @return Character in UART read buffer
 */
char USART_getBufferChar(){
    return global_USART_buffer;
}

//
/**
 * @brief Reads a character from UART line and stores in buffer

 * Receives a single character from the UART line and stores this in a read buffer.

 * TO BE IMPLEMENTED: Function should be able to receive multiple characters and store these in a cyclic FIFO buffer.

 * @return execution status (NOT IMPLEMENTED YET)
 */
int USART_receive(){
    if (USART3->SR & 0x20) // check if new character has been received (RXNE bit is set)
    {
        global_USART_buffer = USART3->DR; // read character (automatically clears interrupt flag)
    }

    return 0;
}

//
/*-------------------------- end of file --------------------------*/

C_source_code/SOLID_usart.h

/**
 * @file   SOLID_usart.h
 * @author Morten Olsen
 * @version 1.0
 * @date   25–June–2013
 * @brief   SOLID header file for UART functions

 * (C) COPYRIGHT 2013 Deutches Zentrum fur Luft und Raumfahrt(DLR)
 */

// include guard
#ifndef SOLID_usart_H
#define SOLID_usart_H

int Config_USART(void);

#endif
int USART_receive(void);
int USART_printString(char *str);
int USART_printNumber(int num);
char USART_getBufferChar(void);
int USART_call_command(char cmd);

#endif /* SOLID_usart_H */
Maple Source Code
Pulse detector circuit

Noise calculations
Noise calculations for the pulse detector circuit to find potential design concerns.

Shot noise
Bandwidth, $B$, taken slighter above low-pass filter cut-off, dc current, $i_{DC}$, taken as smallest (shot noise highest for small currents) expected DC-current in any pulse detector OpAmp.

$$k := 1.38 \times 10^{-23} \left[ \frac{m^2 \text{ kg}}{s^2 \text{ K}} \right]$$

$$E_{\text{SHOT NOISE}} := 0.000797 \ [V]$$

Thermal noise
Using Nyquist's relation, through 100 kΩ resistor @ 25 °C

$$E_{\text{THERMAL}} := 0.000222 \ [V]$$

Orbit induced voltage

$$B_{\text{MAX}} := \text{evalf} \left( \sqrt{4 \cdot B_{\text{EQUATOR}} \cdot \left( \frac{R_{\text{EARTH}}}{R_{\text{EARTH}} + h_{\text{ORBIT}}} \right)^3} \right);$$
Estimating input ripple current and voltage based on solar panel power and boost converter component values:

> \( P_{\text{panel}} := 7.6 \, [\text{W}] \):

Input power to the EPS converter. Assuming direct perpendicular sun on one satellite side and two sides with 45 deg angular illumination and using the "Kelly Cosine" equal to 0.7@45deg which considers real solar cell dependence on sunlight angle.

> \( \epsilon_{\text{kelly@45 deg}} := 0.7 : \)

\[
P_{\text{in max}} := 2 \cdot P_{\text{panel}} + 4 \cdot P_{\text{panel}} \cdot \epsilon_{\text{kelly@45 deg}} ;
\]

\[
P_{\text{in max}} := 36.5 \, [\text{W}] \]

Calculating input current and input current ripple assuming ripple is limited to 10% of the input current DC level (rule of thumb design criteria).

> \( E_{\text{panel mpp}} := 15.4 \, [\text{V}] : \)

\[
i_{\text{in max}} := \frac{P_{\text{in max}}}{E_{\text{panel mpp}}} ;
\]

\[
i_{\text{ripple}} := 0.1 \cdot i_{\text{in max}} ;
\]

\[
i_{\text{in max}} := 2.37 \, [\text{A}] \]

\[
i_{\text{ripple}} := 0.237 \, [\text{A}] \]

Calculating input voltage ripple assuming no input filter is connected and combined input capacitance (solar panel + cables) is 100nF:

> \( C_{\text{input}} := 100 e^{-9} \, [\text{F}] : \)

\[
f_{\text{switch}} := 300 e3 \, [\text{Hz}] ;
\]

\[
t_{\text{switch}} := \frac{1}{f_{\text{switch}} ;}
\]

\[
E_{\text{ripple}} := \frac{1}{C_{\text{input}} \cdot i_{\text{ripple}} \cdot t_{\text{switch}} ;}
\]

\[
E_{\text{ripple}} := 7.90 \, [\text{V}] \]

A ripple voltage of almost 8 V is improbable. As seen, the ripple voltage is inversely proportional to the input capacitance which could easily vary orders of magnitude depending on circuit layout, components and possibly added input filter. Hence more insight into this issue is difficult without knowing further details of the power subsystem design.

Calculating input current harmonics using Fourier Series and assuming perfect triangular waveform and 50% duty cycle (symmetric triangular waveform):
Print first fundamental and first 10 harmonics of a triangular waveform:

```maple
> ALL := 0:
> for j from 1 to 10 do
>   ALL := ALL + SERIES(2*j - 1)
> end do; 'First ten components of fourier series' = ALL;
> p_ALL := plot(ALL, x = 0 .. 2*t_SWITCH, color = coral, legend = "All(till 23rd harmonic)", thickness = 2):
```

First ten components of fourier series:

\[
\begin{align*}
&\frac{1920 \sin(600000\pi x)}{\pi^2} - \frac{640 \sin(1.80 \times 10^6 \pi x)}{3\pi^2} \\
&+ \frac{384 \sin(3.00 \times 10^6 \pi x)}{5\pi^2} - \frac{1920 \sin(4.20 \times 10^6 \pi x)}{49\pi^2} + \frac{640 \sin(5.40 \times 10^6 \pi x)}{27\pi^2} \\
&- \frac{1920 \sin(6.60 \times 10^6 \pi x)}{121\pi^2} + \frac{1920 \sin(7.80 \times 10^6 \pi x)}{169\pi^2} - \frac{128 \sin(9.00 \times 10^6 \pi x)}{15\pi^2} \\
&+ \frac{1920 \sin(1.02 \times 10^7 \pi x)}{289\pi^2} - \frac{1920 \sin(1.14 \times 10^7 \pi x)}{361\pi^2}
\end{align*}
\]

\[
(1.1.4.4)
\]

\[
> p_F := plot(\text{SERIES}(1), x = 0 .. 2\cdot t_{\text{SWITCH}}, \text{color} = \text{green}, \text{legend} = \text{"Fundamental(300kHz)"}, \\
\text{linestyle} = \text{dot}, \text{thickness} = 3):
\]

\[
p_3 := plot(\text{SERIES}(3), x = 0 .. 2\cdot t_{\text{SWITCH}}, \text{color} = \text{yellow}, \text{legend} = \text{"3rd harmonic(900kHz)"}, \\
\text{linestyle} = \text{dash}, \text{thickness} = 2):
\]

\[
p_5 := plot(\text{SERIES}(5), x = 0 .. 2\cdot t_{\text{SWITCH}}, \text{color} = \text{red}, \text{legend} = \text{"5th harmonic(1.5MHz)"}, \\
\text{linestyle} = \text{longdash}, \text{thickness} = 2):
\]

\[
p_7 := plot(\text{SERIES}(7), x = 0 .. 2\cdot t_{\text{SWITCH}}, \text{color} = \text{blue}, \text{legend} = \text{"7th harmonic(2.1MHz)"}, \\
\text{linestyle} = \text{dashdot}, \text{thickness} = 2):
\]

\[
p_{\text{FILL}} := \text{plot}([-250, 250], x = 0 .. 2\cdot t_{\text{SWITCH}}, \text{filled} = \text{true}, \text{color} = \text{"Gray")}:
\]

\[
> \text{display}(p_{\text{ALL}}, p_F, p_3, p_5, p_7, p_{\text{FILL}}, \text{axes} = \text{boxed}, \text{title} = \text{"Input Current Fourier Series"}, \text{labels} = [\text{"t (s)"}, \text{"i (mA)"}] );
\]

The ripple amplitude quickly reduces for higher harmonics. Since the voltage ripple (capacitor charging/discharging) is the integral over this current, the voltage ripple amplitude will drop even faster (the voltage ripple resembles more a sine-wave hence less higher harmonics).

\section*{Input filter}

Design calculations for the input filter. Several types of filters have been considered. Filter design equations are from:
2nd order Sallen-Key High-pass (Chebyshev)

Chebyshev response, 2nd order (2db ripple). Quality factor (filter selectivity), $Q_{DAMP}$, is set to 2, desired filter gain, $K$, (in passband) is set to 2.114.

```
> restart:
> K_{HPcheb} := 2.114:
> \text{n\text{RATIO}} := 1:
> Q_{DAMP} := 2:
> f_{NORMALIZATION} := 1/0.907:
> f_{CUTOFF} := 2e6:
> C_{FILTER} := 100e-12:
> C_{COMPENSATION} := 47e-12:
> R_{GAIN} := 750:
```

Calculating filter resistor values based on filter capacitor values, gain, quality factor and desired cut-off frequency:

```
> m_{HPcheb} := solve\left(\frac{Q_{DAMP}}{1 + \text{n\text{RATIO}} + m_{HPcheb} \cdot \text{n\text{RATIO}} \cdot (1 - K_{HPcheb})}, m_{HPcheb}\right);
> R_2 := \text{evalf}\left(\frac{1}{2 \cdot \pi \cdot C_{FILTER} \cdot \sqrt{m_{HPcheb} \cdot \text{n\text{RATIO}} \cdot f_{CUTOFF} / f_{NORMALIZATION}}}\right);
> R_1 := R_2 \cdot m_{HPcheb}:
```

```
m_{HPcheb} := 1.29
R_2 := 636.
R_1 := 819.
```

Calculating feedback resistor value (filter gain setting):

```
> R_{FEEDBACK} := R\text{GAIN} \cdot (K_{HPcheb}^{-1});
> R_{FEEDBACK} := 836.
```

Filter dynamic system:

```
> C_1 := C_{FILTER}:
> C_2 := C_{FILTER}:
> \text{SysHPcheb2dB} := \left(\frac{K_{HPcheb} \cdot (s)^2 \cdot R_1 \cdot R_2 \cdot C_1 \cdot C_2}{(s)^2 \cdot (R_1 \cdot R_2 \cdot C_1 \cdot C_2) + (s) \cdot (R_2 \cdot C_2 + R_1 \cdot C_1 + R_1 \cdot C_2 \cdot (1 - K_{HPcheb})) + 1}\right):
```

```
> \text{BodePlot}(\text{TransferFunction}(\text{SysHPcheb2dB}), \text{hertz}, \text{title} = "High-Pass filter bodeplot", \text{range} = 2e4 .. 100e6, \text{output} = \text{dualaxis}):
```

Calculating high-pass filter transfer function including compensation capacitor:

---

Page 4 of 22
2nd order Sallen-Key High-pass (Bessel)

Bessel response, 2nd order

\[ K_{HP\text{bess}} := 1.268 ; \]
\[ n_{\text{RATIO}} := 1 ; \]
\[ Q_{\text{DAMP}} := 1 ; \]
\[ f_{\text{NORMALIZATION}} := 1 / 1.272 ; \]
\[ f_{\text{CUTOFF}} := 5e6 ; \]
\[ C_{\text{FILTER}} := 100e-12 ; \]
\[ R_{\text{GAIN}} := 2e3 ; \]

Calculating filter resistor values based on filter capacitor values, gain, quality factor and desired cut-off frequency:

\[ m_{\text{HP\text{bess}}} := \text{solve} \left( Q_{\text{DAMP}} = \frac{\sqrt{m_{\text{HP\text{bess}}} \cdot n_{\text{RATIO}}}}{1 + n_{\text{RATIO}} + m_{\text{HP\text{bess}}} \cdot n_{\text{RATIO}} \cdot \left( 1 - K_{\text{HP\text{bess}}} \right)}, m_{\text{HP\text{bess}}} \right) ; \]

\[ R_2 := \text{evalf} \left( \frac{1}{2 \cdot \pi \cdot C_{\text{FILTER}} \cdot \sqrt{m_{\text{HP\text{bess}}} \cdot n_{\text{RATIO}} \cdot f_{\text{CUTOFF}} \cdot f_{\text{NORMALIZATION}}} \right) ; \]
\[ R_1 := R_2 \cdot m_{\text{HP\text{bess}}} ; \]
\[ m_{\text{HP\text{bess}}} := 2.08 \]
\[ R_2 := 281 . \]
\[ R_1 := 584 . \]  

Calculating feedback resistor value (filter gain setting):

\[ R_{\text{FEEDBACK}} := R_{\text{GAIN}} \cdot \left( K_{\text{HP\text{bess}}} - 1 \right) ; \]
\[ R_{\text{FEEDBACK}} := 536 . \]  

2nd order Sallen-Key Band-pass

\[ f_{\text{MIDBAND}} := 6e6 ; \]
\[ B := 8e6 ; \]
\[ Q_{\text{DAMP DESIRED}} := f_{\text{MIDBAND}} / B ; \]
\[ K_{\text{BP}2\text{nd}} := 2 ; \]
\[ A_{\text{GAINMIDBAND}} := K_{\text{BP}2\text{nd}} / \left( 3 - K_{\text{BP}2\text{nd}} \right) ; \]
\[ Q_{\text{DAMP}} := 1 / \left( 3 - K_{\text{BP}2\text{nd}} \right) ; \]
\[ C_{\text{FILTER}} := 100e-12 ; \]
Two options for feedback resistor calculation: design for quality factor (a) or design for midband gain (b):

\[
R_{\text{FEEDBACK}a} := \frac{2 \cdot Q_{\text{DAMP}} - 1}{Q_{\text{DAMP}}} \cdot R_{\text{GAIN}};
\]

\[
R_{\text{FEEDBACK}b} := \frac{2 \cdot A_{\text{GAINMIDBAND}}}{1 + A_{\text{GAINMIDBAND}}} \cdot R_{\text{GAIN}};
\]

\[
R_{\text{FEEDBACK}a} := 10000, \quad R_{\text{FEEDBACK}b} := 13300.
\]  

\(4th\) order Multi-FeedBack Band-pass (bessel)

Band pass filter by combining two 2nd order filters (staggered tuning)

\[
> f_{\text{MIDBAND}} := 5e6;
\]

\[
B := 8e6;
\]

\[
Q_{\text{DESIRSED}} := \frac{f_{\text{MIDBAND}}}{B};
\]

\[
A_{\text{MIDBAND}} := 1;
\]

\[
\Delta \Omega := \frac{1}{Q_{\text{DESIRSED}}};
\]

\[
C_{\text{FILTER}} := 100e-12;
\]

\[
R_{\text{GAIN}} := 2e3;
\]

\[
a_1 := 1.3617;
\]

\[
b_1 := 0.618;
\]

\[
Q_{\text{DESIRSED}} := 0.625
\]

\[
\Delta \Omega := 1.60
\]  

Use successive approximation to find value for \(\alpha\): (long equation should be zero)

\[
> \alpha := 1.961;
\]

\[
\alpha^2 + \left( \frac{\alpha \cdot \Delta \Omega \cdot a_1}{b_1 \cdot (1 + \alpha^2)} \right)^2 + \frac{1}{\alpha^2} - 2 - \left( \frac{\Delta \Omega}{b_1} \right)^2 = -0.00120
\]

Determine cut-off frequency for each of the two filter parts (high pass and low pass):

\[
> f_{\text{MIDBAND}1} := \frac{f_{\text{MIDBAND}}}{\alpha};
\]
Determine quality factor and midband gain for both filters:

\[ Q_i := \frac{Q_{\text{desired}}}{\alpha \cdot a_1} \left( 1 + \alpha^2 \right) \cdot b_1 \]

\[ A_{\text{midband}} := \frac{Q_i}{Q_{\text{desired}}} \sqrt{\frac{A_{\text{midband}}}{b_1}} \]

\[ Q_i := 0.701 \]

\[ A_{\text{midband}} := 1.43 \]  

Calculating filter resistor values and feedback resistor values. The latter may be designed for midband gain or quality factor (not both simultaneously).

\[ R_{21} := \text{evalf} \left( \frac{Q_i}{\pi \cdot f_{\text{midband}} \cdot C_{\text{filter}}} \right) \]

\[ R_{11} := \frac{2 \cdot A_{\text{midband}}}{R_{21}} \]

\[ R_{31} := \frac{A_{\text{midband}}}{2 \cdot Q_i^2 + A_{\text{midband}}} \cdot R_{11} \]

\[ R_{22} := \text{evalf} \left( \frac{Q_i}{\pi \cdot f_{\text{midband}} \cdot C_{\text{filter}}} \right) \]

\[ R_{12} := \frac{2 \cdot A_{\text{midband}}}{A_{\text{midband}} \cdot R_{12}} \]

\[ R_{32} := \frac{A_{\text{midband}} \cdot R_{12}}{2 \cdot Q_i^2 + A_{\text{midband}}} \]

\[ R_{21} := 875. \]
\[ R_{11} := 307. \]
\[ R_{31} := 182. \]
\[ R_{22} := 228. \]
\[ R_{12} := 79.8 \]
\[ R_{32} := 47.2 \]  

**4th order Multi-FeedBack Band-pass (Butterworth)**

\[ f_{\text{midband}} := 6 \cdot 10^6 \]
\[ B := 20 \cdot 10^6 \]

\[ Q_{\text{desired}} := \frac{f_{\text{midband}}}{B} \]
Determining $\alpha$:

> $\alpha := 3.372$:

$$
\alpha^2 + \left( \frac{\alpha \cdot \Delta \Omega \cdot a_1}{b_1 \cdot (1 + \alpha^2)} \right)^2 + \frac{1}{\alpha^2} - 2 - \frac{(\Delta \Omega)^2}{b_1} = -0.00162
$$

Determine cut-off frequency for each of the two filter parts (high pass and low pass):

> $f_{MIDBAND1} := \frac{f_{MIDBAND}}{\alpha}$

$$
f_{MIDBAND2} := f_{MIDBAND} \cdot \alpha
$$

$$
f_{MIDBAND1} := 1.78 \times 10^6
$$

$$
f_{MIDBAND2} := 2.02 \times 10^7
$$

Determine quality factor and midband gain for both filters:

> $Q_i := \frac{Q_{DESIRED} \cdot (1 + \alpha^2) \cdot b_1}{\alpha \cdot a_1}$;

$$
A_{MIDBANDi} := \frac{Q_i}{Q_{DESIRED}} \cdot \frac{A_{MIDBAND}}{b_1},
$$

$$
Q_i := 0.778
$$

$$
A_{MIDBANDi} := 2.59
$$

Calculating filter resistor values and feedback resistor values. The latter may be designed for midband gain or quality factor (not both simultaneously).

> $R_{21} := \text{evalf} \left( \frac{Q_i}{\pi \cdot f_{MIDBANDi} \cdot C_{FILTER}} \right)$;

$$
R_{11} := \frac{R_{21}}{2 \cdot A_{MIDBANDi}};
$$

$$
R_{31} := \frac{A_{MIDBANDi}}{2 \cdot Q_i^2 + A_{MIDBANDi}} \cdot R_{11};
$$

\[Q_{DESIRED} := 0.300\]

\[\Delta \Omega := 3.33\]
2nd order Band-reject (twin-T, active)

> \( f_{\text{MIDBAND}} := 300e3 \):
> \( B := 600e3 \):
> \( K := 1 \):

\[
Q_{\text{DESIRED}} := \frac{f_{\text{MIDBAND}}}{B};
\]

\[
Q_{\text{FILTER}} := \frac{1}{2 \cdot (2 - K)};
\]

\[
C_{\text{FILTER}} := 1000e-12;
\]

\[
R_{\text{GAIN}} := 2e3;
\]

\[
Q_{\text{DESIRED}} := 0.500
\]

\[
Q_{\text{FILTER}} := \frac{1}{2}
\]

Calculating filter resistor values:

> \( R_{\text{FILTER}} := \text{evalf}\left(\frac{1}{2 \cdot \pi \cdot f_{\text{MIDBAND}} \cdot C_{\text{FILTER}}}\right)\);

\[
R_{\text{FEEDBACK}a} := (K - 1) \cdot R_{\text{GAIN}};
\]

\[
R_{\text{FEEDBACK}b} := \left(1 - \frac{1}{2 \cdot Q_{\text{FILTER}}}\right) \cdot R_{\text{GAIN}};
\]

\[
R_{\text{FILTER}} := 531.
\]

\[
R_{\text{FEEDBACK}a} := 0.
\]

\[
R_{\text{FEEDBACK}b} := 0.
\]

Notch filter transfer function:
\textbf{2nd order Band-reject (Wien-Robinson, active)}

\begin{align*}
> & \quad f_{\text{MIDBAND}} := 500e3 : \\
& \quad C_{\text{FILTER}} := 100e-12 : \\
& \quad R_2 := 10e3 : \\
& \quad Q_{\text{FILTER}} := 5 : \\
& \quad A_0 := 1 : \\
& \text{Calculate filter resistor value:} \\
> & \quad R_{\text{FILTER}} := \text{evalf} \left( \frac{1}{2 \cdot \pi \cdot f_{\text{MIDBAND}} \cdot C_{\text{FILTER}}} \right) ;
\end{align*}

\begin{equation}
R_{\text{FILTER}} := 3180. \tag{1.2.7.1}
\end{equation}

\begin{align*}
& \text{Calculate coefficients:} \\
> & \quad \alpha := 3 \cdot Q_{\text{FILTER}} - 1 ; \\
& \quad \beta := A_0 \cdot 3 \cdot Q_{\text{FILTER}} ; \\
& \quad \alpha := 14 \\
& \quad \beta := 15 \tag{1.2.7.2}
\end{align*}

\begin{align*}
& \text{Calculate remaining filter resistor values:} \\
> & \quad R_3 := \frac{R_2}{\alpha}; \\
& \quad R_4 := \frac{R_2}{\beta}; \\
& \quad R_3 := 714. \\
& \quad R_4 := 667. \tag{1.2.7.3}
\end{align*}

\textbf{1st order Low-pass passive (RC)}

\begin{align*}
> & \quad f_{\text{CUTOFF}} := 10e6 : \\
& \quad C_{\text{FILTER}} := 100e-12 : \\
& \quad R := \text{evalf} \left( \frac{1}{2 \cdot \pi \cdot C_{\text{FILTER}} \cdot f_{\text{CUTOFF}}} \right) ; \\
& \quad R := 159. \tag{1.2.8.1}
\end{align*}
2nd order Low-pass passive (RLC butterworth - including inductor losses)

Formulas and design values taken from "Electornic Filter Design Handbook", 4th edition, A. Williams and F. Taylor. In contrary with active filters, passive filter responses depend on load and source resistances. A 2nd order butterworth (40dB / decade roll-off) is considered sufficient as a first design approach.

Since the source impedance (solar cell + cable) is much less than load impedance (active filter input), the infinite load impedance termination is selected from "Electronic Filter Design Handbook". The filter requirements are:

\[ f_{3dB} := 10e6 : \]
\[ f_{40dB} := 100e6 : \]
\[ A_s := \frac{f_{40dB}}{f_{3dB}} ; \]
\[ R_{SOURCE} := 50 : \]
\[ R_L := 2.13 : \]

\[ A_s := 10.0 \]

(1.2.9.1)

The 50 Ω source impedance is an estimation from the solar cell source impedance + cable + optocoupler + a small added series resistor to the LC filter to stabilize the filter.

Calculating maximum Q value of chosen 1 uH inductor (caused by loss in equivalent series resistance) and calculating required predistortion, \( d \):

\[ Q_{L,MAX} := \text{evalf} \left( \frac{2 \cdot \pi \cdot f_{3dB} \cdot 10^{-6}}{R_L} \right) ; \]
\[ d := \frac{1}{Q_{L,MAX}} ; \]

\[ Q_{L,MAX} := 29.5 \]
\[ d := 0.0339 \]

(1.2.9.2)

Normalized LC values from table 11.2 (2nd order butterworth, unequal terminations) in book:

\[ L_{1,NORM} := 0.7200 : \]
\[ C_{2,NORM} := 1.390 : \]

Calculating the frequency-scale-factor (FSF) (Denominator is the normalized frequency (rad/s) at which the 2nd order butterworth response has 3dB attenuation):

\[ FSF := \text{evalf} \left( \frac{f_{3dB}}{1 \cdot 2 \cdot \pi} \right) ; \]

\[ FSF := 6.28 \times 10^7 \]

(1.2.9.3)

Setting the impedance transformation, \( Z \), for the source impedance which initially was normalized to 1 Ω.

\[ Z := 50 : \]

Calculating required LC component values:

\[ C_{FILTER} := \frac{C_{2,NORM}}{FSF \cdot Z} ; \]
\[ L_{FILTER} := \frac{L_{1,NORM} \cdot Z}{FSF} ; \]

\[ C_{FILTER} := 4.42 \times 10^{-10} \]

(1.2.9.4)
\[ L_{\text{FILTER}} := 5.73 \times 10^{-7} \]  

In the initial prototype the following component values are used:

\[ C_{\text{FILTER PROTOTYPE}} := 100e-12 : \]
\[ L_{\text{FILTER PROTOTYPE}} := 1e-6 : \]
\[ f_{\text{CUTOFF}} := \text{evalf} \left( \frac{1}{2 \cdot \pi \cdot \sqrt{C_{\text{FILTER PROTOTYPE}} \cdot L_{\text{FILTER PROTOTYPE}}} \right) ; \]
\[ Q_{\text{RLC}} := \frac{1}{R_{\text{SOURCE}}} \cdot \sqrt{\frac{L_{\text{FILTER PROTOTYPE}}}{C_{\text{FILTER PROTOTYPE}}} ;} \]
\[ f_{\text{CUTOFF}} := 1.59 \times 10^7 \]
\[ Q_{\text{RLC}} := 2.00 \]  

\[ SysLP2ndOrder := \frac{1}{s \cdot C_{\text{FILTER PROTOTYPE}}} + \frac{1.00 \times 10^{10}}{s \left( \frac{1.00 \times 10^{10}}{s} + 1.10^{-6} s + 50 \right)} \]

\[ BodePlot(\text{TransferFunction}(\text{SysLP2ndOrder}), \text{hertz}, \text{output} = \text{dualaxis}) ; \]

\[ 3\text{rd order Low-pass passive (RCLC Chebyshev - including inductor losses)} \]

\[ f_{3dB} := 10e6 : \]
\[ f_{60dB} := 100e6 : \]
\[ A_s := \frac{f_{60dB}}{f_{3dB}} ; \]
\[ R_{\text{SOURCE}} := 50 : \]
\[ R_L := 2.13 : \]
\[ Q_{L\text{MAX}} := \text{evalf} \left( \frac{2 \cdot \pi \cdot f_{3dB} \cdot 1e-6}{R_L} \right) ; \]
\[ A_s := 10.0 \]
\[ Q_{L\text{MAX}} := 29.5 \]  

Required predistortion:

\[ d := \frac{1}{Q_{L\text{MAX}} ;} \]
\[ d := 0.0339 \]  

Setting normalization factors (from table values)

\[ C_{1\text{NORM}} := 0.5250 : \]
\[ L_{2\text{NORM}} := 1.370 : \]
\[ C_{3\text{NORM}} := 1.475 : \]
Filter bodeplots and pulse responses

Plotting filter system step responses:

> with(plot):

Generate pulse disturbance signal using the "Sinc" function:

> \(f_{\text{NOISE}} := 50e6\):
> \(f_{\text{PULSE}} := 56\): 
> \(\tau_{\text{SIMULATION}} := 2e-6\):
> \(\text{noise}_{\text{SINUSERP}} := 2 \cdot \sin(2 \cdot \pi \cdot f_{\text{NOISE}} t) \cdot \exp(-0.7 \cdot f_{\text{NOISE}} t)\):
> \(\text{pulse}_{\text{SINUSERP}} := \sin(2 \cdot \pi \cdot f_{\text{PULSE}} t) \cdot \exp(-0.7 \cdot f_{\text{PULSE}} t)\):

\[
\text{noise}_{\text{TRIANGULAR}} := \text{convert}
\begin{cases}
4 \cdot f_{\text{NOISE}} t & 0 \leq t < \frac{1}{4 \cdot f_{\text{NOISE}}} \\
2 - 4 \cdot f_{\text{NOISE}} t & \frac{1}{4 \cdot f_{\text{NOISE}}} \leq t < \frac{1}{2 \cdot f_{\text{NOISE}}}
\end{cases}, \text{piecewise, } t
\]

\[
\text{pulse}_{\text{TRIANGULAR}} := \text{convert}
\begin{cases}
4 \cdot f_{\text{PULSE}} t & 0 \leq t < \frac{1}{4 \cdot f_{\text{PULSE}}} \\
2 - 4 \cdot f_{\text{PULSE}} t & \frac{1}{4 \cdot f_{\text{PULSE}}} \leq t < \frac{1}{2 \cdot f_{\text{PULSE}}}
\end{cases}, \text{piecewise, } t
\]

Complete filter:

> SysComplete := TransferFunction(SysBRtwinT \cdot SysHPcheb2dBCOMPENSATED 
\cdot SysLP2ndOrder):
> BodePlot(SysComplete, hertz, title = "Input filter bodeplot", range = 100e3 .. 100e6, output 
\quad = dualaxis, axis_2 = [gridlines = 15], axis_1 = [gridlines = 30, mode = log, numpoints = 200);
Simulating a noise spike input:

\[
\text{sim} := \text{Simulate} \left( \text{SysComplete}, \left[ \text{noise_{SINUS EXP}} \right], \text{dsolveargs} = \left[ \text{maxfun} = 500000 \right] \right);
\]

\[
p \text{outnoise} := \text{plots} \left[ \text{odeplot} \left( \text{sim}, \left[ \left[ t, y \left( t \right) \right] \right], t = 0 .. \tau_{\text{SIMULATION}}, \text{numpoints} = 50000, \text{color} = \text{blue}, \text{legend} = \text{"filter output"} \right) ;
\]

\[
p \text{noise} := \text{plot} \left( \text{noise_{SINUS EXP}}, t = 0 .. \tau_{\text{SIMULATION}}, \text{legend} = \text{"noise input"}, \text{numpoints} = 500 \right) ;
\]

\[
\text{display} \left( \text{poutnoise}, \text{pnoise}, \text{axes} = \text{boxed}, \text{labels} = \left[ \text{"time (s)"}, \text{"amplitude"} \right], \text{title} = \text{"Transient response on high-frequency noise spike"}, \text{view} = \left[ 0 .. 2 \times 10^{-7}, -1.5 .. 2 \right], \text{gridlines}, \text{labeldirections} = \left[ \text{horizontal}, \text{vertical} \right] \right) ;
\]

Simulating an impact pulse input:

\[
\text{sim} := \text{Simulate} \left( \text{SysComplete}, \left[ \text{pulse_{TRIANGULAR}} \right], \text{dsolveargs} = \left[ \text{maxfun} = 500000 \right] \right);
\]

\[
p \text{outpulse} := \text{plots} \left[ \text{odeplot} \left( \text{sim}, \left[ \left[ t, y \left( t \right) \right] \right], t = 0 .. \tau_{\text{SIMULATION}}, \text{numpoints} = 50000, \text{color} = \text{blue}, \text{legend} = \text{"filter output"} \right) ;
\]

\[
p \text{pulse} := \text{plot} \left( \text{pulse_{TRIANGULAR}}, t = 0 .. \tau_{\text{SIMULATION}}, \text{legend} = \text{"pulse input"} \right) ;
\]

\[
\text{display} \left( \text{poutpulse}, \text{ppulse}, \text{axes} = \text{boxed}, \text{labels} = \left[ \text{"time (s)"}, \text{"amplitude"} \right], \text{title} = \text{"Transient response on mid-frequency pulse signal"}, \text{view} = \left[ 0 .. 1 \times 10^{-6}, -1.5 .. 1.5 \right], \text{gridlines}, \text{labeldirections} = \left[ \text{horizontal}, \text{vertical} \right] \right) ;
\]

\[
\text{ImpulseResponsePlot} \left( \text{SysComplete}, 1 \times 10^{-6} \right) ;
\]

### Additionally added components to active filters

\[
\text{restart} ;
\]

\[
\text{with} \left( \text{Units} \left[ \text{Standard} \right] \right) ;
\]

\[
\text{UsingSystem} \left( \text{SI} \right) ;
\]

High-pass pole at gain resistor. A series capacitor is added to the gain resistor of the high-pass filter OpAmp to roll off the gain at frequencies below the filter pass frequency. Roll off frequency is set to 10 % of the filter cut-off frequency.

\[
\text{f_{CUTOFF LOW}} := 2 \times 10^6 \left[ \text{Hz} \right] ;
\]

\[
\text{C_{ROLLOFF LOW}} := \text{evalf} \left( \frac{1}{2 \times \pi \times 1 \times \left[ \Omega \right] \times 0.1 \times \text{f_{CUTOFF LOW}}} \right) ;
\]

\[
\text{C_{ROLLOFF LOW}} := 7.96 \times 10^{-7} \left[ \text{F} \right]
\]

Extra low-pass pole added to feedback resistor. A parallel resistor is added to the OpAmp feedback resistor to roll off the filter gain at frequencies above the high cut-off frequency. A frequency somewhat above the filter cut-off frequency is chosen

\[
\text{f_{CUTOFF HIGH}} := 10 \times 10^6 \left[ \text{Hz} \right] ;
\]

\[
\text{R_{GAIN}} := 750 \left[ \Omega \right] ;
\]

\[
\text{C_{ROLLOFF HIGH}} := \text{evalf} \left( \frac{1}{2 \times \pi \times \text{R_{GAIN}} \times 2 \times \text{f_{CUTOFF HIGH}}} \right) ;
\]

\[
\text{C_{ROLLOFF HIGH}} := 1.06 \times 10^{-11} \left[ \text{F} \right]
\]

An extra low-pass pole at the active filter output can be added. A small series isolation resistor is already added to stabilize the OpAmp.

\[
\text{R_{ISOLATION}} := 20 \left[ \Omega \right] ;
\]
\[ f_{\text{CUTOFF,LP}} := 15e6 \, \text{[Hz]} : \]
\[ C_{LP} := \text{evalf} \left( \frac{1}{2 \cdot \pi \cdot R_{\text{ISOLATION}} \cdot f_{\text{CUTOFF,LP}}} \right) ; \]
\[ C_{LP} := 5.31 \times 10^{-10} \, \text{[F]} \] (1.2.12.3)

PSPICE simulations however showed only little extra improvement with this extra low-pass pole hence it is discarded from the implemented prototype.

\textbf{Reference Voltage Filter}

\textbf{RC active Low-pass (with DC gain for hysteresis setting)}

\[ \text{with} \left( \text{Units} \left[ \text{Standard} \right] \right) : \]
\[ \text{UsingSystem} \left( \text{SI} \right) : \]

\[ C_{\text{FILTER}} := 100e-12 \, \text{[F]} : \]
\[ f_{\text{CUTOFF}} := 25e3 \, \text{[hertz]} : \]
\[ R_{\text{GAIN}} := 20e3 \, \text{[\Omega]} : \]
\[ E_{\text{DC}} := 1.5 \, \text{[V]} : \]
\[ E_{\text{THRESHOLD}} := 0.2 \, \text{[V]} : \]

Calculating the required resistor values for the RC filter, threshold level of pulse detector and calculating the resulting OpAmp gain:

\[ R_{\text{FEEDBACK}} := \frac{E_{\text{THRESHOLD}}}{E_{\text{DC}}} \cdot R_{\text{GAIN}} ; \]
\[ R_{\text{FILTER}} := \text{evalf} \left( \frac{1}{2 \cdot \pi \cdot f_{\text{CUTOFF}} \cdot C_{\text{FILTER}}} \right) ; \]
\[ K := 1 + \frac{R_{\text{FEEDBACK}}}{R_{\text{GAIN}} ;} \]
\[ R_{\text{FEEDBACK}} := 2670. \, \text{[\Omega]} \]
\[ R_{\text{FILTER}} := 63700. \, \text{[\Omega]} \]
\[ K := 1.13 \] (1.3.1.1)

Adding possibility to trim the threshold voltage using a digital potentiometer. Replacing the gain resistor (the one connected to GND) by a linear potentiometer leads to a discrete logarithmic behaviour in the threshold voltage trim setting as shown below. The potentiometer has 64 steps with maximum 50 kΩ resistance. For this particular potentiometer, one end must be connected to GND, hence it cannot replace the feedback resistor (to avoid the logarithmic behaviour).

\[ \text{with} \left( \text{DynamicSystems} \right) : \]

\[ N_{\text{STEPS}} := 64 ; \]
\[ R_{\text{POT,MAX}} := 50 ; \]
\[ R_{\text{FEEDBACK}} := 2.5 ; \]
\[ E_{\text{DC}} := 1.5 ; \]
\[ X := \text{Vector} \left( N_{\text{STEPS}}, t \rightarrow \frac{R_{\text{POT,MAX}}}{N_{\text{STEPS}}} \cdot t \right) ; \]
\[ Y := \text{Vector} \left( N_{\text{STEPS}}, t \rightarrow E_{\text{DC}} \cdot \frac{R_{\text{FEEDBACK}}}{R_{\text{FEEDBACK}} + \frac{R_{\text{POTMAX}}}{N_{\text{STEPS}}}.t} \right) ; \]

\[ > \text{DiscretePlot}(X, Y, \text{style} = \text{stair}, \text{labels} = ["Potentiometer Resistance (k\Omega)", "Threshold (V)"], \text{axes} = \text{boxed}, \text{title} = "Threshold voltage trimming with digital potentiometer") ; \]

\section*{Short-circuit protection}

Calculating the threshold voltage which triggers the short-circuit (SC) protection circuit putting its output from High-to-Low:

\[ > \text{restart} : \]
\[ \text{with}((\text{Units}[\text{Standard}])) : \]
\[ \text{Using}((\text{System}[\text{SI}])) : \]

\[ > R_{\text{DIV1}} := 91e3[\Omega] ; \]
\[ R_{\text{DIV2}} := 7.7e3[\Omega] ; \]
\[ R_{\text{FEEDBACK}} := 100e3[\Omega] ; \]
\[ E_{\text{OUT LOW}} := 0.2[V] ; \]
\[ E_{\text{OUT HIGH}} := 2.96[V] ; \]
\[ E_{\text{DC BIAS}} := 1.485[V] ; \]
\[ E_{\text{SOLAR}} := 8[V] ; \]

Calculating the comparator negative input voltage (reference setting) and the two threshold voltages on the positive comparator input going from high-to-low (THL) and low-to-high (TLH):

\[ > E_{\text{NEG}} := E_{\text{DC BIAS}} + (E_{\text{SOLAR}} - E_{\text{DC BIAS}}) \cdot \left( \frac{R_{\text{DIV2}}}{R_{\text{DIV1}} + R_{\text{DIV2}}} \right) ; \]
\[ E_{\text{THL}} := \left( E_{\text{NEG}} \cdot \left( \frac{1}{R_{\text{DIV2}}} + \frac{1}{R_{\text{DIV1}}} + \frac{1}{R_{\text{FEEDBACK}}} \right) - \frac{E_{\text{DC BIAS}}}{R_{\text{DIV2}}} - \frac{E_{\text{OUT HIGH}}}{R_{\text{FEEDBACK}}}) \cdot R_{\text{DIV1}} ; \]
\[ E_{\text{TLH}} := \left( \frac{E_{\text{DC BIAS}} - E_{\text{OUT LOW}}}{R_{\text{FEEDBACK}}} + \frac{E_{\text{DC BIAS}}}{R_{\text{DIV2}}} + \frac{E_{\text{DC BIAS}}}{R_{\text{DIV1}}} \right) \cdot R_{\text{DIV1}} ; \]

\begin{equation}
\begin{align*}
E_{\text{NEG}} & := 1.99[V] \\
E_{\text{THL}} & := 7.12[V] \\
E_{\text{TLH}} & := 20.2[V]
\end{align*}
\end{equation}

Calculating corresponding trip-current which forces the high-to-low transition (triggering the SC-protection circuit):

\[ > R_{\text{SENSE}} := 100[\Omega] ; \]
\[ i_{\text{SC TRIGGER}} := \frac{(E_{\text{SOLAR}} - E_{\text{THL}})}{R_{\text{SENSE}}}; \]
\[ i_{\text{SC TRIGGER}} := 0.00880[A] \]

(2.2)

Setting the required LED bias current resistor:

\[ > E_{\text{SUPPLY}} := 3.3[V] ; \]
\[ E_{\text{FLED}} := 1.4[V] ; \]
\[ E_{F,\text{DIODE}} := 0.3 \, [V] \]
\[ i_{\text{BIASED}} := 0.5e^{-3} \, [A] \]
\[ R_{\text{BIASED}} := \frac{E_{\text{SUPPLY}} - E_{F,\text{LED}} - E_{F,\text{DIODE}}}{i_{\text{BIASED}}} \]
\[ R_{\text{BIASED}} := 3200. \, [\Omega] \] (2.3)

Sun detector threshold voltage for solar array input voltage:

\[ > R_{24} := 20e3 \, [\Omega] \]
\[ R_{25} := 20e3 \, [\Omega] \]
\[ R_{18} := 91e3 \, [\Omega] \]
\[ R_{19} := 7.7e3 \, [\Omega] : e \]
\[ E_{\text{SOLAR DETECT THRESHOLD}} := E_{\text{DC,BIAS}} \cdot \frac{R_{25}}{R_{24} + R_{25} + R_{19}} \]
\[ E_{\text{SOLAR DETECT THRESHOLD}} := 9.61 \, [V] \] (2.4)

Calculating filter capacitor components values. Bypass Capacitor in parallel with 100 Ohm sense resistor (to pass high frequency signals around the sense resistor, such that the 100 Ohm resistor does not change the high frequency signal source impedance.

\[ > f_{\text{BYPASS}} := 15e3 \, [Hz] \]
\[ C_{\text{BYPASS}} := \frac{1}{2 \cdot \pi \cdot f_{\text{BYPASS}} \cdot 1 \, [\Omega]} \]
\[ C_{\text{BYPASS}} := 0.0000106 \, [F] \] (2.5)

Using a 22 uF bypass capacitor, the pass frequency is around 7.5 kHz.

Addind RC filter capacitors to input pins of comparator for SC-protection circuit. These are needed to avoid triggering of this circuit due to high-frequency noise. These also set the response time of the triggering of the SC-protection circuit to around 1 ms. The capacitor values may be increased to set this slower.

\[ > f_{\text{CUTOFF}} := 1e3 \, [Hz] \]
\[ C_{\text{FILTER}} := \frac{1}{2 \cdot \pi \cdot f_{\text{CUTOFF}} \cdot R_{18}} \]
\[ C_{\text{FILTER}} := 1.75 \times 10^{-9} \, [F] \] (2.6)

**Impact probability**

\[ > \text{restart} : \]
\[ \text{with(Units[Standard]) :} \]
\[ \text{UsingSystem(SI)} : \]

Using model Master2009, time epoch: 2014-09 to 2015-09, sphere target, flux [m²/year]. Simulation output for impact rates for various orbit configurations:

Altitude: h=430, inclination: i=53 deg. :

\[ > F_1 := 10.53 \, [m^{-2} \, \text{year}^{-1}] : F_1 := \text{convert}(F_1,'\text{units}','m^{-2} \, \text{year}^{-1}) ; \]
\[ F_1 := 3.34 \times 10^{-7} \, \left[ \frac{1}{m^2 \, s} \right] \] (3.1)
$F_1 = 10.5 \left[ \frac{1}{m^2 \text{yr}} \right]$ (3.1)

$h=640, \text{i}53 \text{deg}:

$F_2 := 24.6 \left[ m^{-2} \text{year}^{-1} \right] ;
F_2 := 7.80 \times 10^{-7} \left[ \frac{1}{m^2 \text{s}} \right];
F_2 = 24.6 \left[ \frac{1}{m^2 \text{yr}} \right]$ (3.2)

$h=430, \text{i}98 \text{deg}:

F_3 := 12.1 \left[ m^{-2} \text{year}^{-1} \right] ;
F_3 := 3.83 \times 10^{-7} \left[ \frac{1}{m^2 \text{s}} \right];
F_3 = 12.1 \left[ \frac{1}{m^2 \text{yr}} \right]$ (3.3)

$h=640, \text{i}98 \text{deg}:

F_4 := 38.6 \left[ m^{-2} \text{year}^{-1} \right] ;
F_4 := 1.22 \times 10^{-6} \left[ \frac{1}{m^2 \text{s}} \right];
F_4 = 38.6 \left[ \frac{1}{m^2 \text{yr}} \right]$ (3.4)

Detector surface area:

$A_{\text{DETECTOR}} := 0.193 \left[ m^2 \right]$.

Poisson distribution for calculating impact probabilities for any minimum number of impacts e.g. probability that at least 5 impacts are observed, during the full mission duration of one year.

$$f_{\text{POISSON}}(k) \rightarrow \frac{(\lambda \cdot \tau)^k \cdot e^{-\lambda \cdot \tau}}{k!} ;$$

$\tau := 1 \left[ \text{year} \right]$:

$$P := (n) \rightarrow 100 \cdot \left( 1 - \sum_{k=0}^{n-1} f_{\text{POISSON}}(k) \right) ;$$

Impact probabilities for at least; 1, 5 or 10 impacts. Calculated for various orbit configurations

$h=430$, inclination: $i=53 \text{deg}.$:

$\lambda := F_1 \cdot A_{\text{DETECTOR}}$

$P(1)$;

$P(5)$;

$P(10)$;

$\lambda := 6.44 \times 10^{-8} \left[ \frac{1}{s} \right]$:

86.9

5.56
h=640, i53 deg:
\[
> \lambda := F_2 \cdot A_{\text{DETECTOR}}; \\
P(1); \\
P(5); \\
P(10);
\]
\[
\lambda := 1.50 \times 10^{-7} \left[ \frac{1}{s} \right]
\]
99.1
51.4
2.36

h=640, i98 deg
\[
> \lambda := F_3 \cdot A_{\text{DETECTOR}}; \\
P(1); \\
P(5); \\
P(10);
\]
\[
\lambda := 7.40 \times 10^{-8} \left[ \frac{1}{s} \right]
\]
90.3
8.79
0.0162

h=640, i98 deg
\[
> \lambda := F_4 \cdot A_{\text{DETECTOR}}; \\
P(1); \\
P(5); \\
P(10);
\]
\[
\lambda := 2.36 \times 10^{-7} \left[ \frac{1}{s} \right]
\]
99.9
86.4
21.8

\section*{Detector lines}

\subsection*{Spacecraft charging concerns}

Maximum leakage resistance to ground on detector lines, due to charging effects

\[
> \text{restart} : \\
> \quad \text{with(Units[Standard]) :} \\
> \quad \text{UsingSystem(ST)} : \\
> \quad E_{\text{MAX ALLOWED}} := 3.6 \left[ V \right] ; \\
> \quad i_{\text{LEAKAGE MUX}} := 10e^{-12} \left[ A \right] ; \\
> \quad J_{\text{CHARGE MAX}} := 10e^{-5} \left[ A \ m^{-2} \right] ; \\
> \quad A_{\text{DETECTOR LINE GROUP}} := 32 \cdot 100e^{-6} \left[ m \right] \cdot 15e^{-2} \left[ m \right] ;
\]
Leakage resistance path from MUX input leakage current:

\[ R_{\text{LEAKAGE MAX}} := \frac{E_{\text{MAX ALLOWED}}}{A_{\text{DETECTOR LINE GROUP}} J_{\text{CHARGE MAX}}} ; \]

\[ i_{\text{LEAKAGE MUX}} := 1.0 \times 10^{-11} \text{ [A]} \]

\[ A_{\text{DETECTOR LINE GROUP}} := 0.000480 \text{ [m}^2\text{]} \]

\[ R_{\text{LEAKAGE MAX}} := 7.50 \times 10^7 \text{ [\Omega]} \]

\[ (4.1.1) \]

E-field between two detectorlines, one charged to 5.3V other held at ground potential:

\[ > R_{\text{LEAKAGE MUX}} := \frac{5 \text{ [V]}}{i_{\text{LEAKAGE MUX}}} ; \]

\[ R_{\text{LEAKAGE MUX}} := 5.00 \times 10^{11} \text{ [\Omega]} \]

\[ (4.1.2) \]

E-field between two detectorlines:

\[ > d_{\text{LINE2LINE}} := 25.4e-6 \text{ [m]} ; \]

\[ E_{\text{DETECTOR LINE}} := 3.6 \text{ [V]} ; \]

\[ E_{\text{LINE2LINE}} := \frac{E_{\text{DETECTOR LINE}}}{d_{\text{LINE2LINE}}} ; \]

\[ E_{\text{LINE2LINE}} := 142000. \text{ [} \frac{\text{kg m}}{\text{s}^3 \text{A}}\text{]} \]

\[ (4.1.3) \]

Product limit from AP copper clad datasheet (dielectric strength per meter V/m):

\[ > \text{Dielectric}_{\text{APPYRALUX}} := 6e3 \text{ [V mil}^{-1}\text{]} ; \]

\[ E_{\text{MAX ALLOWED DETECTORLINE}} := \text{Dielectric}_{\text{APPYRALUX}} d_{\text{LINE2LINE}} ; \]

\[ \text{Dielectric}_{\text{APPYRALUX}} := 2.36 \times 10^8 \text{ [} \frac{\text{kg m}}{\text{s}^3 \text{A}}\text{]} \]

\[ E_{\text{MAX ALLOWED DETECTORLINE}} := 6000. \text{ [V]} \]

\[ (4.1.4) \]

\[ \text{Number of lines required} \]

\[ > \text{restart} : \]

\[ \text{with(Units[Standard]) :} \]

\[ \text{UsingSystem(SI) :} \]

\[ \text{line width: w[m]} \]

\[ > d_{\text{LINECENTER TO LINECENTER}} := 300e-6 \text{ [m]} ; \]

\[ l_{\text{PANEL}} := 166.76e-3 \text{ [m]} ; \]

\[ w_{\text{PANEL}} := 145.4e-3 \text{ [m]} ; \]

\[ n_{\text{LINES X}} := \frac{l_{\text{PANEL}}}{d_{\text{LINECENTER TO LINECENTER}}} ; \]

\[ n_{\text{LINES Y}} := \frac{w_{\text{PANEL}}}{d_{\text{LINECENTER TO LINECENTER}}} ; \]

\[ n_{\text{LINES X}} := 556. \]

\[ n_{\text{LINES Y}} := 485. \]

\[ (4.2.1) \]

\[ \text{Switching transients and parasitic capacitances} \]
Microstrip (detector line): 3pF / cm
Ribbon cable 12pF / ft

> \( C_{\text{CABLE}} := 50 e^{-12} \, \text{[F]} \):
> \( C_{\text{DETECTOR LINE}} := 150 e^{-12} \, \text{[F]} \):
> \( C_{\text{MUX IN}} := 50 e^{-12} \, \text{[F]} \):
> \( C_{\text{MUX SD}} := 10 e^{-12} \, \text{[F]} \):
> \( C_{\text{MUX OUT}} := 5 e^{-12} \, \text{[F]} \):
> \( C_{\text{MCU}} := 5 e^{-12} \, \text{[F]} \):
> \( R_{\text{MUX}} := 180 \, \text{[\Omega]} \):
> \( R_{\text{PULLDOWN}} := 33 e3 \, \text{[\Omega]} \):
> \( R_{\text{SOURCE}} := 470 \, \text{[\Omega]} \):
> \( E_{\text{LOGIC HIGH}} := 2 \, \text{[V]} \):
> \( E_{\text{LOGIC LOW}} := 0.8 \, \text{[V]} \):
> \( E_{\text{SUPPLY}} := 3.3 \, \text{[V]} \):
> \( n_{\text{LINES PER GROUP}} := 32 \):
> \( n_{\text{LINES}} := 1024 \):

Calculating diode reverse leakage current and the potential drop across the load pulldown resistor this current causes.

> \( i_{\text{LEAK DIODE}} := 100 e^{-9} \, \text{[A]} \):
> \( i_{\text{LEAK TOTAL}} := n_{\text{LINES}} \cdot i_{\text{LEAK DIODE}} \):
> \( V_{\text{OUT LEAK}} := i_{\text{LEAK TOTAL}} \cdot R_{\text{PULLDOWN}} \):

\[
\begin{align*}
  i_{\text{LEAK TOTAL}} &:= 0.000102 \, \text{[A]} \\
  V_{\text{OUT LEAK}} &:= 3.38 \, \text{[V]} \\
\end{align*}
\quad (4.3.1)
\]

The leakage current through the diodes is strong enough such that all detector line capacitors need to discharge in order to drive the output voltage low.

Switching from a intact detector line (output high) to a broken detector line (output going low):

> \( R_{\text{TOTAL}} := R_{\text{PULLDOWN}} + R_{\text{MUX}} \):
> \( C_{\text{TOTAL}} := C_{\text{MUX IN}} + n_{\text{LINES PER GROUP}} \cdot C_{\text{MUX OUT}} + C_{\text{CABLE}} + C_{\text{MCU}} + n_{\text{LINES}} \cdot C_{\text{DETECTOR LINE}} \):
> \( R_{\text{TOTAL}} := 33200 \, \text{[\Omega]} \):
> \( C_{\text{TOTAL}} := 1.54 \times 10^{-7} \, \text{[F]} \):

\[
\begin{align*}
  v_{\text{DISCHARGE}} &:= (t) \rightarrow E_{\text{SUPPLY}} \left( -e^{-\frac{R_{\text{TOTAL}}}{C_{\text{TOTAL}}} \cdot t} \right); \\
  t_{\text{DISCHARGE}} &:= \text{solve} \left(v_{\text{DISCHARGE}}(t) = E_{\text{LOGIC HIGH}}, t\right) \, \text{[s]} \\
\end{align*}
\quad (4.3.2)
\]

\[
\begin{align*}
  v_{\text{DISCHARGE}}(t) &= 3.3 \cdot e^{-196 \cdot t} \, \text{[V]} \\
  t_{\text{DISCHARGE}} &:= 0.00256 \, \text{[s]} \\
\end{align*}
\quad (4.3.3)
\]

Going from initial state of MUX completely disconnected (all capacitors discharged) and to closing one in- and out
MUX switch:

\[ R_{TOTAL} := R_{SOURCE} + R_{MUX} \]
\[ C_{TOTAL} := C_{MUXIN} + 2 \cdot n_{LINES \ PER \ GROUP} \cdot C_{MUXOUT} + C_{CABLE} + C_{MCU} + n_{LINES} \cdot C_{DETECTORLINE} \]

\[ R_{TOTAL} := 650 \ [\Omega] \]
\[ C_{TOTAL} := 1.54 \times 10^{-7} \ [\text{F}] \] (4.3.4)

> \[ v_{\text{CHARGE}} := (t) \rightarrow E_{SUPPLY} \left( 1 - e^{-\frac{t}{R_{TOTAL} \cdot C_{TOTAL}}} \right) \] \[ \frac{\text{d}[s]}{[s]} \]

\[ v_{\text{CHARGE}}(t) \approx v_{\text{CHARGE}}(t) \]

\[ t_{\text{CHARGE}} := \text{solve} \left( v_{\text{CHARGE}}(t) = E_{\text{LOGIC HIGH}}, \ [s] \right) \]

\[ v_{\text{CHARGE}}(t) = (3.3 - 3.3 \times e^{-9990 \cdot t}) \ [V] \]

\[ t_{\text{CHARGE}} := 0.0000933 \ [s] \] (4.3.5)