University of Tartu

Faculty of Science and Technology

Institute of Technology

Computer Engineering

Mari Allik

Software Development for the Mechanical Shock Testing System at Tartu Observatory

Master’s thesis (30 ECTS)

Supervisors:

MSc Viljo Allik

MSc Teet Tilk

PhD Riho Vendt

Tartu 2016

Software Development for the Mechanical Shock Testing System at Tartu Observatory

Abstract

Components used in space are subject to high frequency and high amplitude mechanical shocks that occur during the launch and separation phases of the space vehicles. Shock testing is done to ensure that components can withstand the shock events.

A shock can be described by an acceleration and time history, but in shock testing it is difficult to manage and quantify shocks by only looking at the acceleration and time history of a shock. Instead, a shock response spectrum is used, which plots the peak accelerations of multiple single degree of freedom systems with their own unique frequencies over a range of frequencies, usually from 100 Hz to 10 000 Hz. The shocks that occur in space can reach up to acceleration levels of 10 000 g's.

The main objective of this work was to develop a software application for the mechanical shock testing system at Tartu Observatory. The shock testing system at Tartu Observatory consists of a resonant plate, pendulum hammer, guiding rod, weights and a data acquisition module with acceleration sensors. The software was developed in Microsoft Visual Studio .NET 2003 using the C++ programming language and external libraries. The software is required to acquire the data from the sensors connected to the data acquisition module and process and plot it. The shock response spectrum and acceleration-time histories can be plotted.

An important aspect of shock testing is to be able to recreate shocks of various levels. If the shock testing system manages to do this, the system is considered reliable. To test the reliability of the system, the developed software was tested by performing shock testing on test objects and plotting the results.

CERCS: T320, Space Technology

Keywords:

Shock testing, shock response spectrum, signal processing, data acquisiton

Tarkvara arendamine Tartu Observatooriumi põrutuskindluse katsesüsteemile

Lühikokkuvõte

Kosmosetööstuses kasutatavad komponendid peavad vastu pidama kõrge sageduse ja amplituudiga mehhaanilistele põrutustele, mis esinevad kanderakettide üleslennutamise ja eraldusfaaside ajal. Tagamaks komponentide vastupidavus põrutusele, teostatakse põrutus-kindluse testimist.

Põrutust (e. šokki) saab esitada kiirendus-aeg graafikul, kuid põrutuskindluse testimisel on vajalik erineva tasemega šokke korrata ning seda on keeruline kiirendus-aeg graafikut kasutades teha. Selle tõttu on kasutusele võetud šoki kostespekter, mis võimaldab esitada ühe vabadusastmega süsteemide, millel igal ühel on oma ainulaadne loomulik sagedus, maksimaalsed kiirendused määratud sagedusvahemiku jaoks, tavaliselt vahemikus 100 Hz kuni 10 000 Hz. Kosmoses esinevate šokkide kiirendus võib ulatuda kuni 10 000 g-ni.

Käesoleva töö eesmärgiks on arendada tarkvaralahendus Tartu Observatooriumi mehaanilisele põrutuskindluse katsesüsteemile. Katsesüsteem koosneb resonantspinnast, pendliga haamrist, juhtvardast, raskustest ja andmehõive süsteemist koos kiirendusanduritega. Tarkvara arendati Microsoft Visual Studio .NET 2003 keskkonnas, kasutades C++ programmeerimiskeelt ning erinevaid teeke. Tarkvara peamiseks ülesandeks on andmete kogumine andmehõivesüsteemiga ühendatud anduritest ning andmete töötlemine ja esitamine. Tarkvara võimaldab esitada šoki kiirendus-aeg ja kostespektri graafikuid.

Põrutuskindluse testimises on oluline, et erinevate tugevusastmetega šokke oleks võimalik taasluua. Kui katsesüsteem võimaldab seda teha, loetakse süsteem usaldusväärseks. Süsteemi usaldusväärsuse testimiseks teostati katseobjektil põrutuskindluse süsteemi ning arendatud tarkvara kasutades põrutuskindluse testimine ning testide tulemused esitati graafikutel.

CERCS: T320, Kosmosetehnoloogia

Võtmesõnad:

Põrutuskindluse testimine, šoki kostespekter, signaalitöötlus, andmehõive

Table of contents

List of figures

List of tables

Acronyms and abbreviations

AC – Alternating Current

DASS – Data Acquisition Subsystem

DC – Direct Current

ECSS – European Cooperation for Space Standardization

ESA – European Space Agency

FFT – Fast Fourier Transform

ICP – Integrated Circuit-Piezoelectric

IEPE – Integrated Electronic Piezoelectric

ISIS – Innovative Solutions In Space

MFC – Microsoft Foundation Classes

PCB – Printed Circuit Board

PGL – Plot Graphics Library

PNG – Portable Network Graphics

SDI – Single Document Interface

SDK – Software Development Kit

SDOF – Single Degree of Freedom System

SRS – Shock Response Spectrum

1 Introduction

1.1 Background

During the launch of a space vehicle, high frequency and high amplitude shocks occur. The satellites on board the space vehicle have to withstand these shocks. The shocks usually occur during the launcher stages separation, fairing being jettisoned and separation of the satellite from the launcher [1].

Shock testing is not only performed on satellites but also on different components that are used when building satellites, to ensure that the components can withstand the shock events occurring during space missions. Shocks of various levels usually occur during the launch of the space vehicle. The satellites have to withstand shocks up to 10 000 g depending on the category and specific mission. Shocks have caused failures resulting in total or partial loss of many space missions.

1.2 Purpose and aim of the work

The main objectives of this work is to develop a software application for the mechanical shock testing system in Tartu Observatory space technology laboratory. The software has to be compatible with the signal acquisition module used for registering shocks and also contain a graphical user interface that enables the user to change different parameters and plot results. The shock response spectrum (SRS) has to be calculated from the acceleration and time histories of the shock to enable evaluation of the shock. The spectrum has to be plotted to enable comparison with different level shocks and compare shocks to specification levels. [1].

It is necessary to develop our own software for the shock testing system because software that fulfills the shock testing requirements and is compatible with the used signal acquisition module is hard to find. The developed software will make it possible to perform shock testing by using the shock test system in Tartu Observatory's space technology laboratory.

1.3 Organization of the thesis

This thesis consists of 8 different chapters. The outline of each of the chapters is described in the following paragraphs.

Chapter 2 describes the main problem to be solved with this thesis and gives an overview of the existing shock testing system in Tartu Observatory.

Chapter 3 gives an overview about what a shock response spectrum is and why it is the most used method in shock testing systems.

Chapter 4 lists all the requirements that the developed software has to accomplish. Also an overview of existing software solutions is given.

Chapter 5 describes the software development process. It describes all the main software components and their functionality.

Chapter 6 gives an overview of the graphical user interface of the software application. All button events are described and commented and the main steps of the user interface are explained.

Chapter 7 outputs results from testing the shock testing system and software.

Chapter 8 concludes the thesis work and gives some suggestions about how to improve the software in the future.

2 Shock testing

2.1 Importance of shock testing

Shocks are characterized by high frequency and acceleration amplitudes with short durations. In the aerospace industry, the severity of a shock can become a source of concern when designing shock sensitive equipment and components [1].

The potential damage that can be caused by shocks is as follows [1]:

- Electronics are impacted (printed circuit boards (PCBs) might malfunction due to failure of components such as relays, quartz and transformers);
- Structures are impacted (cracks and fractures in fragile materials, plastic deformations);
- Mechanisms are impacted (bearings, gears, worm wheels and endless screws can be affected).

2.2 Overview of the shock testing system

The shock testing system in Tartu Observatory was developed by Paul Liias at RadiusSpace [2] and assembled in the summer of 2015. Unfortunately, a shock testing system does not work without some accompanying software that receives the shock data from the sensors and calculates and plots the results of the shocks. The main problem to be solved with this work is to create software that consists of a graphical user interface for the mechanical shock testing system at Tartu Observatory space technology laboratory. The created software will also be tested in a real shock testing environment.

The mechanical shock testing system in Tartu Observatory is a metal to metal impact facility that meets the shock testing standards given by ECSS, ESA, NASA. Figure 2.1 describes the shock testing setup. The system enables to test different equipment and components against shock. It consists of a resonant plate and pendulum hammer seen in figure 2.2. The testing system setup is almost identical to the shock testing facility at ISIS developed by Martin Jonsson [3].

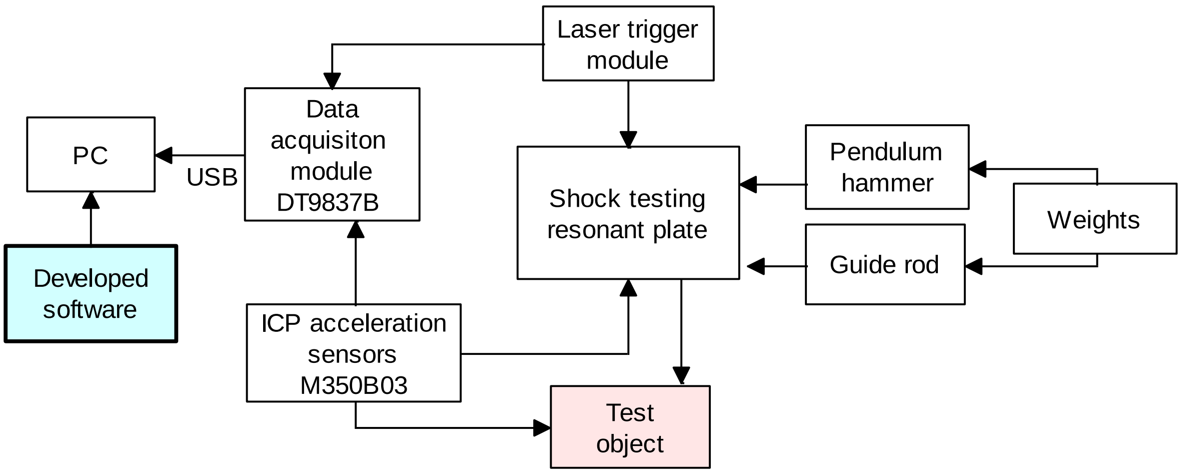

Laser trigger module
Data acquisiton module DT9837B
PC
USB
Developed software
ICP acceleration sensors M350B03
Shock testing resonant plate
Pendulum hammer
Weights
Guide rod
Test object

Figure 2.1: Shock testing setup

2.2.1 Horizontal test setup

The horizontal test setup can be seen in figure 2.2. The pendulum hammer is released from a chosen angle and when it strikes one side of the resonant a shock takes place.

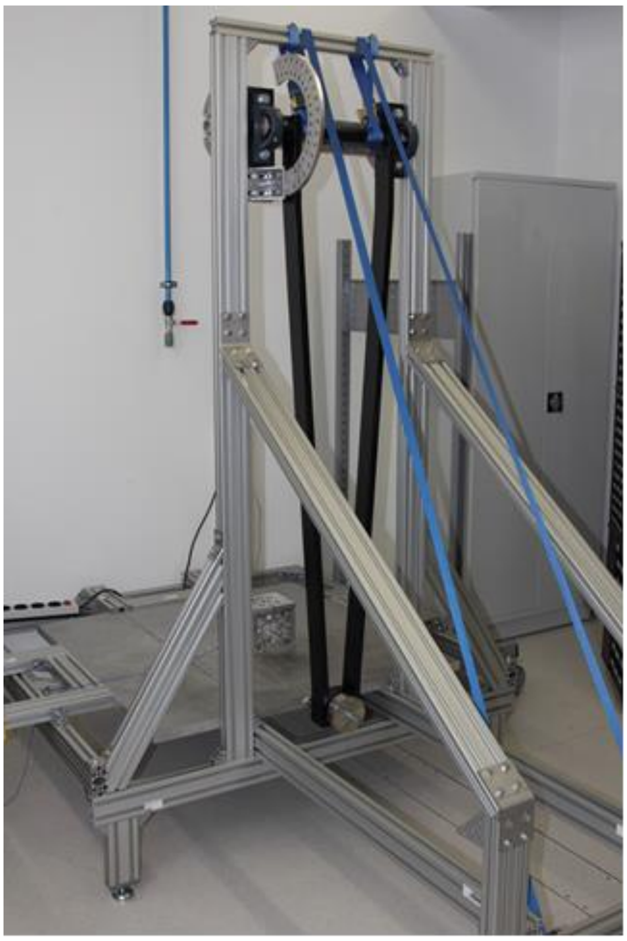


Figure 2.2: Shock testing system at Tartu Observatory’s space technology laboratory

2.2.2 Vertical test setup

The vertical test setup is a metal rod mounted on the resonant plate. The test setup can be seen in figure 2.3. The shock occurs when a mass of 1 kg is dropped on the resonant plate from some chosen height. Another option for the vertical test setup is to use the hammer (from the horizontal setup) and mount it vertically, so that the hammer strikes the pendulum plate vertically.

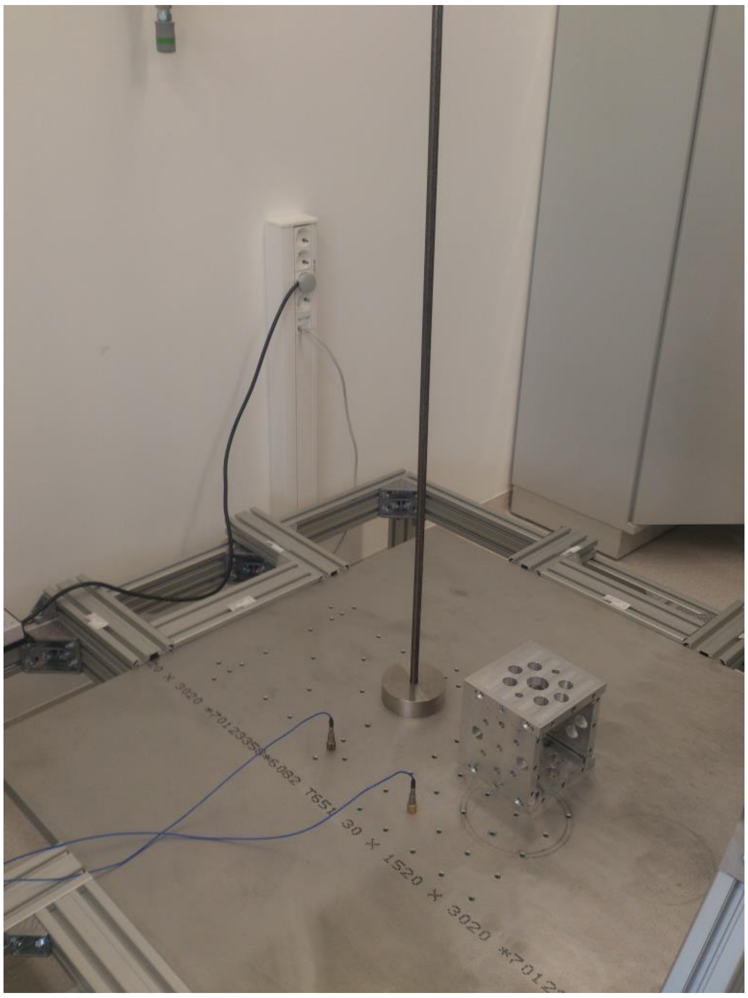


Figure 2.3: Vertical test setup

2.2.3 Laser trigger

An external trigger was built for the shock testing system. The trigger consists of two modules. One module emits a laser beam that the other module detects it. When the pendulum hammer gets closer than about 2.6 cm from the impact point, it breaks the laser beam and triggers the data acquisition. The trigger enables to calculate the velocity of the pendulum hammer. The external trigger modules can be seen in figure 2.4.

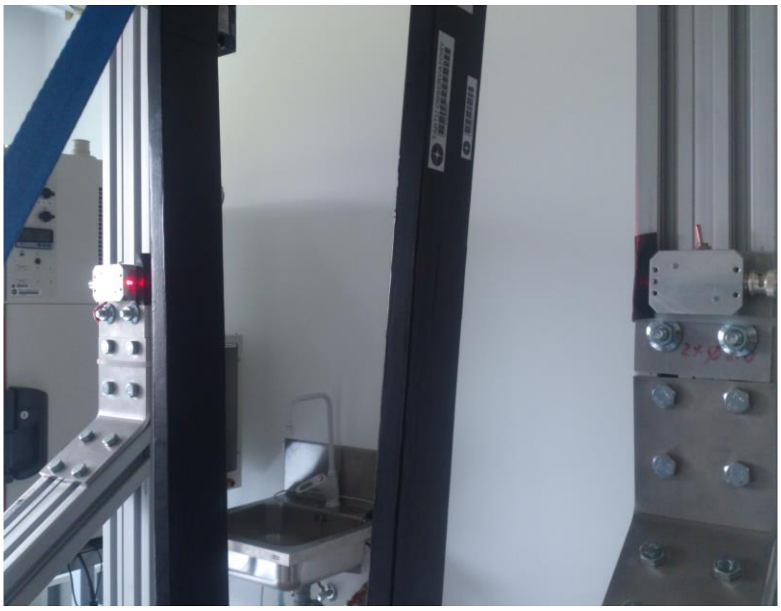


Figure 2.4: Laser trigger modules

2.3 Data acquisition system

To process the data from a shock, a data acquisition system consisting of a data acquisition module and acceleration sensors must be used. The sensors are mounted to the resonant plate and the test object and connected to a high-performance data acquisition module and a PC.

2.3.1 Signal acquisition module

The Data Translations DT9837B data acquisition module (seen in figure 2.5) is used at the shock testing facility. The maximum A/D throughput of the module is 105.429 kHz. The module has four simultaneous 24-bit A/D converters (4 channels) and support for Integrated Electronic Piezoelectric (IEPE) inputs. DT9837 Series modules support an input signal range of ±10 V (using a gain of 1) or ±1 V (using a gain of 10). [9]

An internal excitation current source of 4 mA is used by the module. The module supports both AC and DC coupling. By default, AC coupling is selected for the DT9837B module. The module supports the use of external positive edge (threshold) trigger and software trigger. The use of triggers enables the data acquisition to start after the trigger event has occurred. The external positive edge trigger occurs when the module detects a rising-edge transition on the signal connected to the external trigger connector on the module [10].

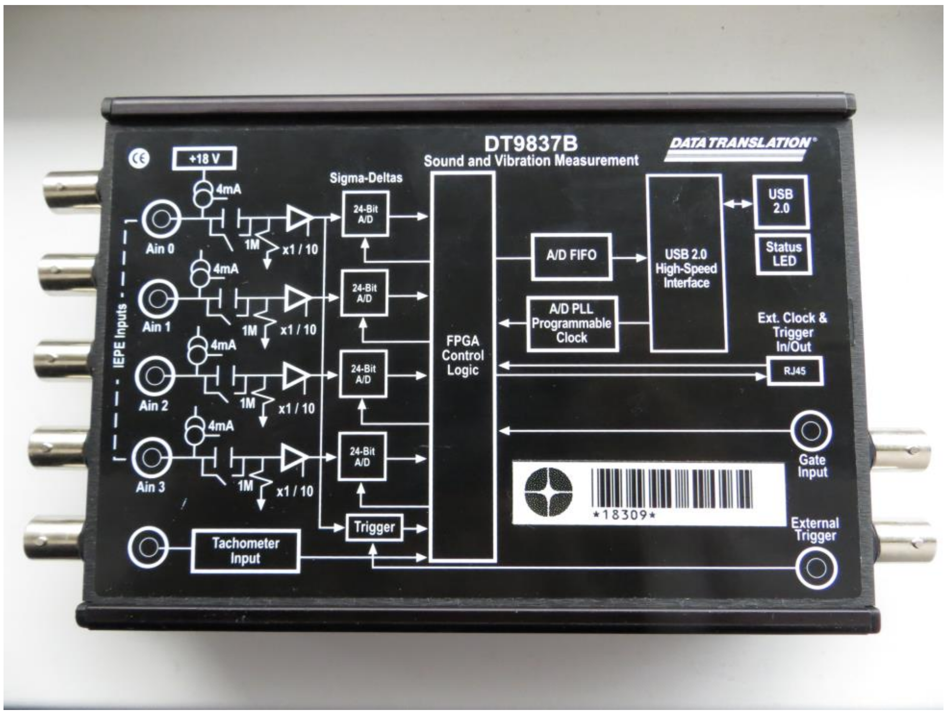


Figure 2.5: Data Translations data acquisition module DT9837B

2.3.2 Acceleration sensors

The shock testing facility uses shock IEPE (ICP) accelerometers. Currently, two piezoelectric acceleration sensors from PCB Piezotronics with model number M350B03 are used. The acceleration sensors are compatible with the chosen data acquisition module. The accelerometers have built-in signal-conditioning electronics that convert the high-impedance charge signal generated by the piezoelectric sensing element into a usable low-impedance voltage signal that can be transmitted over ordinary two-wire or coaxial cables to any voltage recording device. The electronics within ICP accelerometers require excitation power from a constant-current regulated, DC voltage source [13]. The sensitivity of the sensors is 0.5 mV/g, the measurement range is ±10 000 g and the frequency range is 0.4 to 10 kHz [14]. The sensors have been calibrated by PCB Piezotronics on the 16th of February 2015 [24] [25].

3 Theoretical background

A shock is usually measured as an acceleration time history, but the time histories are very hard to compare when quantifying and managing different shocks. A very important aspect of shock testing is to be able to recreate shocks of the same severity. By looking at different acceleration-time plots (see figures 3.1 and 3.2), they all look similar and do not give enough unique information about each shock. The most common way to specify a shock test is the use of the maximax SRS with a standard Q-factor of 10 that is equal to a damping ratio of 5% [1]. The SRS is calculated from the acceleration time history of a shock.

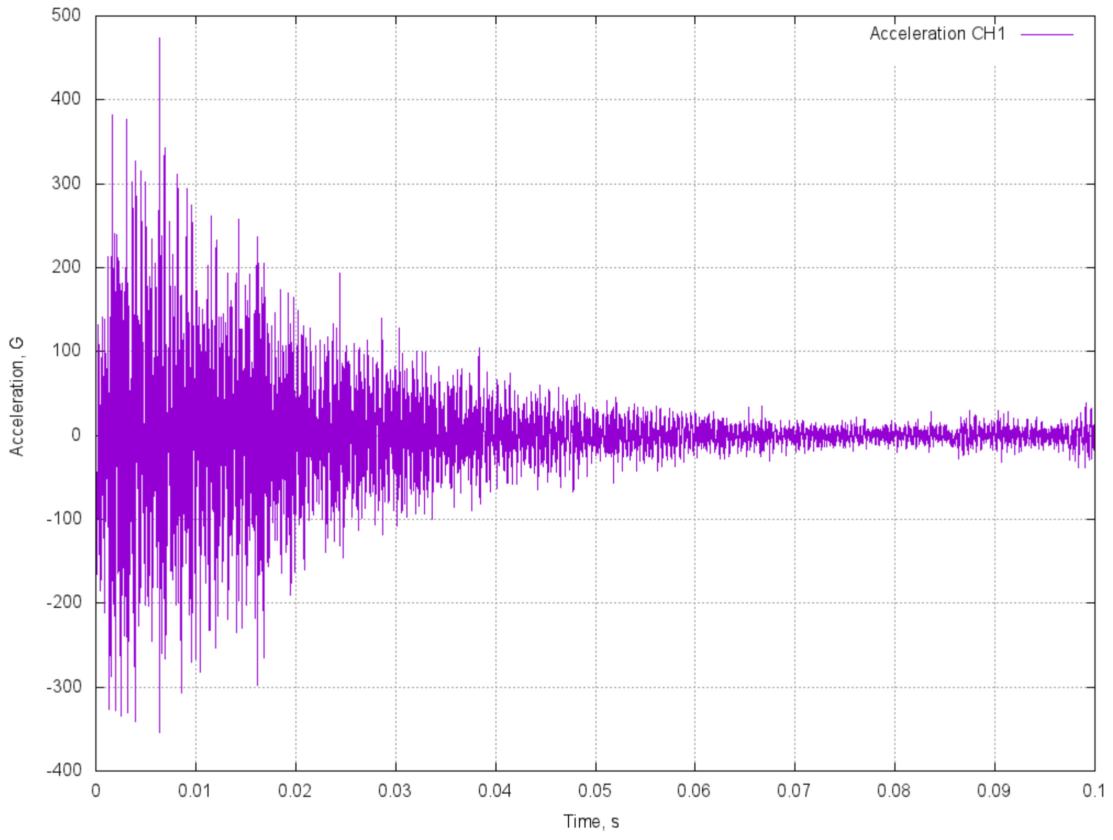


Figure 3.1: A horizontal shock from a 25 degree angle

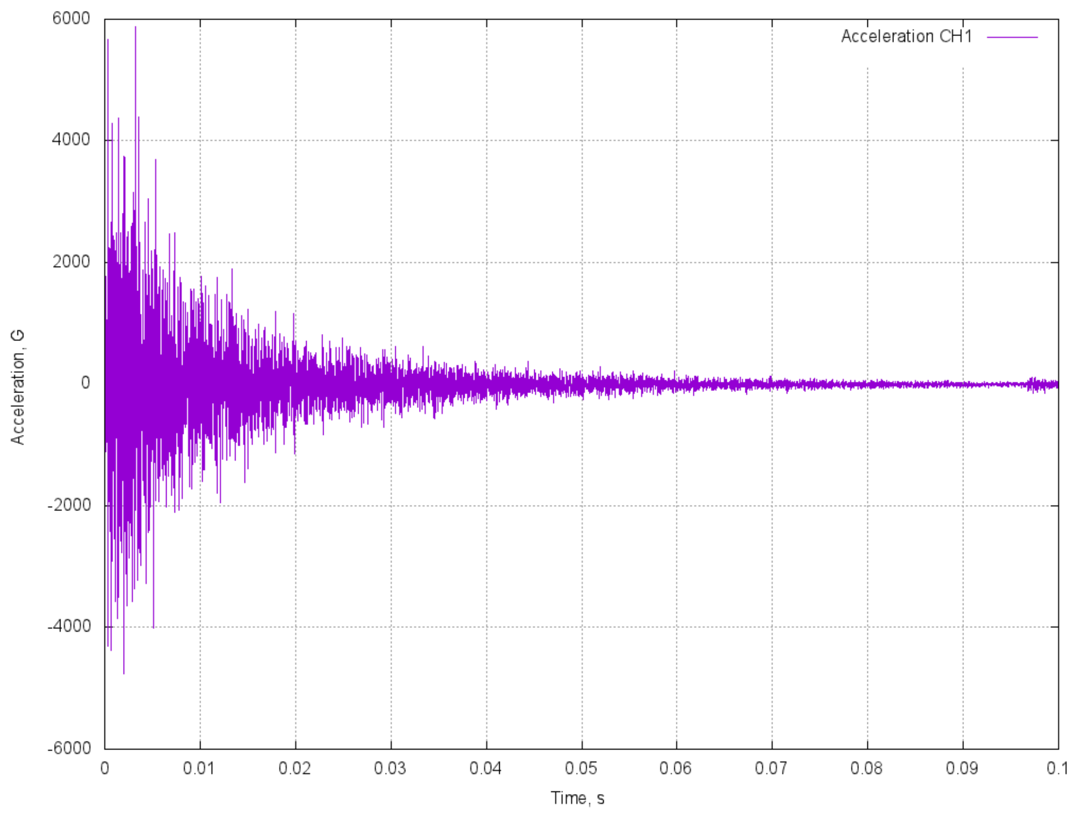


Figure 3.2: A vertical shock from a height of 100 cm

3.1 Shock response spectrum

The SRS applies an acceleration time history as a base excitation to an array of single-degree-of-freedom (SDOF) systems. The calculation of the SRS is performed for a number of independent SDOF systems, each with a unique natural frequency [4]. The SDOF is defined by two parameters, natural frequency and damping ratio (or Q-factor). When calculating the SRS for for example 1000 different components (with different parameters and resonance frequencies) mounted to the shock testing resonant plate, each of the SDOF components will respond with its own unique acceleration transient. The peak response acceleration level is computed for each of the SDOF components. By collecting all those peak levels together, the spectrum across the frequency range of interest (for example 100 – 10000 Hz) can be formed [5]. This is called SRS. The spectrum is calculated for both the negative and positive acceleration time series (as seen in the example SRS in figure 3.3). Then, the maximax SRS is calculated by finding the maximum values of the “positive” and absolute “negative” spectrum. The maximax SRS has become a standard for shock characterization [1].

In the aerospace industry, usually the damping is chosen to be 5%, which is equal to an acceleration amplification factor Q of 10. In situations when damping is known, Q can be chosen accordingly [1].

The number of unique natural frequencies used in the range (100 – 10000 Hz) is arbitrary, but typically each successive natural frequency is $2^{1/6}$ times the previous natural frequency [4].

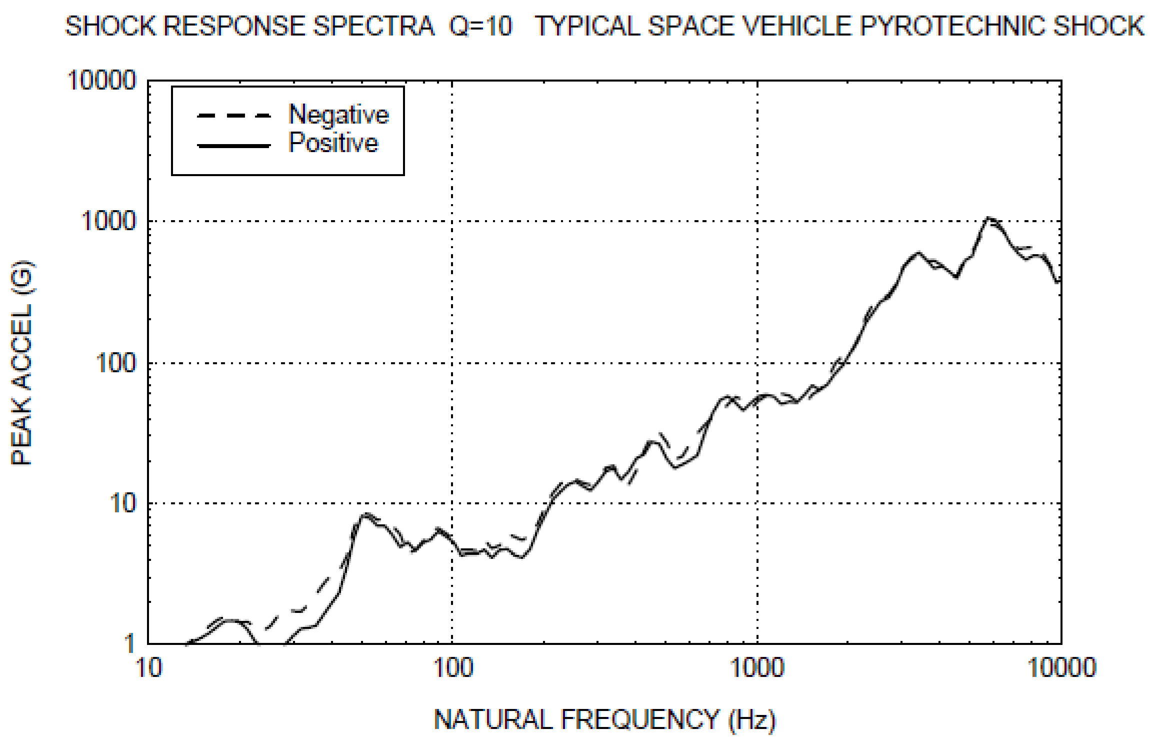


Figure 3.3: Example of a SRS: The SRS of a Pyrotechnic input pulse [4]

3.2 Description of the used algorithms

The SRS is calculated in the time domain via a convolution integral. The end result is represented in terms of the frequency domain [4]. There are many different algorithms that can be used to calculate the shock response spectrum. The main algorithms used in the developed shock testing software for calculating the SRS are the Smallwood algorithm [8] and the Runge Kutta method (of order 4) [23]. The Smallwood algorithm was chosen because it has shown to be the most representative one (specially at higher frequencies) when comparing to other algorithms [3]. The Runge Kutta method is well suited for use near non-smooth or discontinuous behavior such as shocks, because the method has a nonlinear stability property that prevents false oscillations or other non-physical behavior [23].

The basic part of a SRS algorithm is a filtering mechanism of a damped SDOF system. The following procedure is commonly used when calculating the SRS [15]:

1) The input acceleration is filtered by SDOF transfer function.

2) The maximum response of the system and its corresponding natural frequency is taken and a point is marked on the SRS. This is done by sweeping the natural frequency until the maximum frequency region.

3) The points that have been marked on the SRS are merged together.

3.3 Drawbacks of the SRS

The SRS is irreversible, that means that a time history is related to a unique SRS but the SRS does not relate to a unique time history. The SRS only provides the amplitude information, but the Fourier transform provides both amplitude and phase. From the Fourier transform it is possible to calculate the exact time and acceleration history. Another drawback is that the SRS is very complex and a highly nonlinear method and requires special attention when being used [1]. It is impossible to perform a “true” SRS calculation in the real world. Different hardware systems have different capabilities [6].

3.4 Alternative method considered

As an alternative method, Fourier transform was considered for SRS calculation. An equivalency can be seen between the convolution integral and multiplication of Fourier transforms. It is possible to calculate the SRS by using Fourier transforms. The calculation however requires additional calculation steps. The preferred calculation method for the SRS used by the industry is the convolution method. [4]

4 Software requirements and available software solutions

ESA shock testing requirements for the qualification level are described in table 4.1. The component or equipment being tested shall survive shocks up to 3400 g. The shock testing system in Tartu Observatory has to be able to produce shocks that are up to 3400 g at higher frequencies (10 kHz).

Table 4.1: ESA qualification level requirements

	Qualification level	
SHOCK	100 Hz	20 g
	1000 Hz	1400 g
	2000 Hz	3400 g
	10000 Hz	3400 g

4.1 Software requirements

The developed software is required to do the following:

1. be compatible with the Data Translations DT9837B signal acquisition module;
2. receive data from the accelerometers connected to the module after a shock has occurred;
3. convert the data into acceleration;
4. support the use of at least 2 accelerometers (2 channels) at the same time;
5. support sampling frequencies up to 100 kHz;
6. support the use of both a software trigger and an external trigger;
7. load system configuration parameters from a configuration file;
8. calculate the SRS with at least 2 different algorithms;
9. output the results (for all channels):
    - acceleration vs time plot,

- SRS plot,
- required SRS plot (+ tolerances);

10. create a test log (saved as a text file);
11. generate a preliminary test report;
12. handle errors and display error messages.

4.1.1 Requirements for the configuration file

The configuration file must contain the following parameters that the test operator can change:

1. sampling rate;
2. damping ratio;
3. number of channels;
4. the duration of the sampling;
5. test profiles with required SRS;
6. tolerances for the test profiles;
7. information about sensors used:
    a. sensor identification name,
    b. sensor sensitivity (mV/g),
8. units for results (m/s^2 or g).

4.2 Additional requirements

The created software application will be used by test engineers at Tartu Observatory. It is crucial for them to know how to use the software. This is why a user guide must be created that explains the basic steps of the software.

4.3 Available software solutions

To be able to use the DT9837B data acquisition module, a driver needs to be downloaded from the Data Translations web site [11]. There are two drivers available, one is for the DT9837 module and the other for the DT9837A, DT9837B and DT9837C modules. The device driver allows you to use the module with any of the supported packages or utilities [9].

4.3.1 Quick DataAcq application

The Quick DataAcq application can be downloaded on the Data Translations web site [11]. The application enables the user to test key features of the DT9837 modules, display data on screen and save the data for later processing. This application does not support configuring AC or DC coupling or the excitation current source for IEPE inputs [9]. The lack of support for configuring IEPE inputs is one of the reasons why this application cannot be used as shock testing software. Another reason is the nonexistent SRS calculation.

4.3.2 QuickDAQ FFT Analysis Option

QuickDAQ Fast Fourier Transform (FFT) Analysis can be enabled with a purchased license key and it adds FFT analysis features to the QuickDAQ Base version. It allows the user to perform single-channel FFT operations (such as AutoSpectrum, Spectrum and Power Spectral Density) on the acquired data. Parameters for the FFT can be configured (FFT size, windowing type and averaging type) [9]. The SRS can unfortunately not be calculated or plotted with this application.

4.3.3 QuickDAQ Advanced FFT Analysis Option

QuickDAQ Advanced FFT Analysis can also be enabled with a purchased license key and it adds some additional FFT analysis features to the application. The application does not support calculating or plotting of the SRS.

4.3.4 DAQ Adaptor for MATLAB

The Data Translation module comes with a data acquisition library for MATLAB (DAQ Adaptor). To enable the use of the DAQ Adaptor, as a minimum requirement, MATLAB (32-bit) Version 7 (R14) Service pack 3 or higher has to be used. The MATLAB Data Acquisition

Toolbox Version 2.7 or higher is also required for the data acquisition. The DAQ Adaptor has to be installed after the necessary software has been installed. [7]

The shock testing facility at ISIS that is very similar to Tartu Observatory's test system is using MATLAB and the DAQ Adaptor as the main software for recording and plotting shocks and calculating the SRS [3].

In the scope of this thesis, the DAQ Adaptor was tested in MATLAB (version r2013a) It was decided not to use MATLAB as the program for the development of shock testing software because of the following reasons:

- independence of any external software that requires a working license such as MATLAB;
- the test operator is not expected to have experience with MATLAB;
- possibility to run the software in different computers, without the need of having the correct version or any version of MATLAB available;
- ability to work with different data formats;
- ability to generate a preliminary test report.

4.3.5 DataAcq SDK

The DataAcq SDK supports the use of Visual Studio 6.0 and 7.0 environments and Microsoft C or C++ to develop software for the DT9837 series modules using Windows operating systems. The DataAcq SDK complies with the DT-Open Layers standards [9]. The DataAcq SDK was chosen as the main software development kit because it supports the development of Windows applications and is compatible with the Visual Studio .NET 2003 environment, which was available in Tartu Observatory for software development.

5 Software development

The software was developed in the Microsoft Visual Studio .NET 2003 (VC7) environment using the C++ programming language. A MFC application implementing a graphical user interface was created. It also has support for document/view architecture.

5.1 External libraries and software used

5.1.1 The DataAcq SDK

The DataAcq SDK provides functions for system operations such as initializing a device, specifying a subsystem, configuring a subsystem, calibrating a subsystem, error and message handling and releasing of a subsystem and driver. It also contains information functions to determine the capabilities of the installed Data Translation devices and subsystems and channels. [12]

5.1.2 PGL

PGL is a library that is compatible with MFC projects in VC6 and VC7 [20]. It enables to plot data generated in a project without external software. PGL uses GDI+ available as Microsoft SDK [19].

PGL was chosen for this project because of its compatibility with MFC and Visual Studio 2003. Later it was discovered that PGL does not support logarithmic scales. The developers of the library state that the logarithmic scales are still under development. This was discovered after the library had been used for a while. The library was still chosen for plotting the results, because of its easy implementation in the shock testing software and its free distribution. A solution for the logarithmic-scales/axes problem was found by converting the acceleration data into dBg and converting the frequency data to logarithmic scale. Logarithmic labels were added on the correct spots on the x-axis. In the future, there is a plan to improve the PGL library to implement the logarithmic scales correctly.

5.1.3 gnuplot

Because of the non-working logarithmic scales, gnuplot was chosen as an external plotting software for generating high quality plots for the shock test reports. Gnuplot is a free graphing utility that offers a wide range of plotting configurations [21]. The logarithmic scales/axes are working correctly in the program. Gnuplot was downloaded and installed on the shock testing facility's control PC. The developed shock testing software calls gnuplot and launches a script file (with ".plt" extension), that has been created separately for every graph. The ".plt" file is a plot control file that contains gnuplot commands that are necessary for creating and viewing a plot in gnuplot. A script file that has been created for the shock testing software can be seen in Appendix C.

5.1.4 CMarkup

CMarkup is a fast and simple C++ XML parser. It is a library that enables you to create and modify new XML documents from your own C++ program [22]. In the shock testing application, the CMarkup library is used for generating a preliminary test report as an XML document directly from the application.

5.2 Software design

The developed software consists of 8 source files (with ".cpp" extension) and 9 header files (with ".h" extension) and can be found in Appendix A. The software flowchart, that describes all the main processes of the shock testing software application is described in figure 5.1.

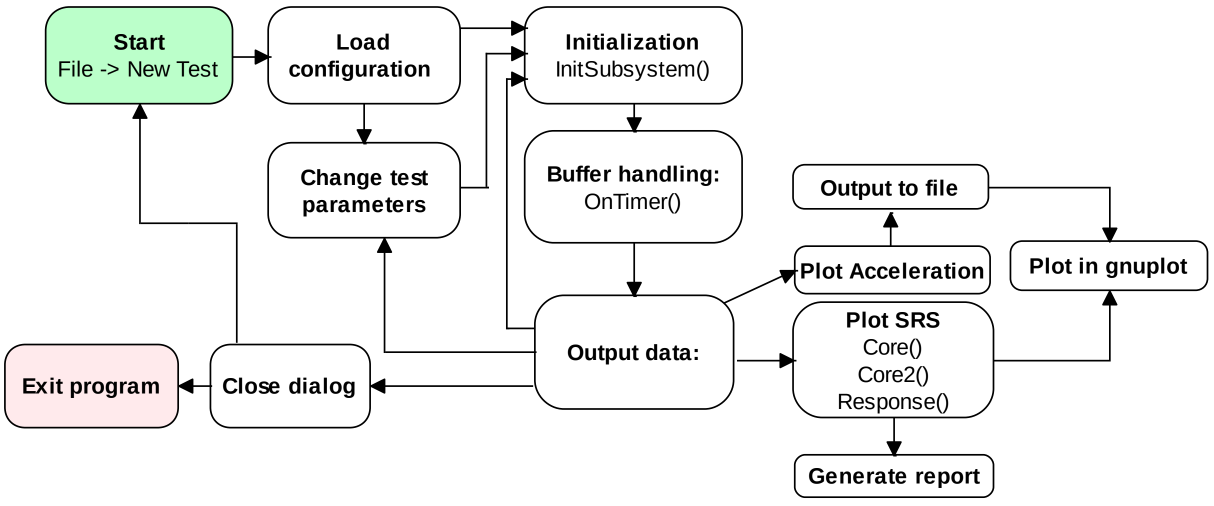


Figure 5.1: Software flowchart

5.2.1 Driver

The driver of the DT9837B module is initialized with the ***GetDriver()*** function. The initialization is done very similarly to the sample code "Initialize the Driver" in the Data Acquisition SDK [12]. The ***GetDriver()*** function gets the name of the device.

5.2.2 Initialization

The ***InitSubsystem()*** function first calls the ***GetDriver()*** function, which lists the name of the device that is available for use. The ***GetDriver()*** function calls the ***olDaInitialize*** function to initialize the DT9837B data acquisition device driver and returns a device handle called HDEV. The ***InitSubsystem()*** function configures the module with the following:

- parameters that are read from the configuration file or entered into the graphical user interface,
- data flow (set to continuous),
- coupling type (AC coupling is chosen),
- channel type (single-ended type is chosen),
- excitation current source (set to internal),
- range (-10 to 10),
- external trigger usage, if an external trigger is used,
- buffer size.

5.2.3 Dialog

The developed software is a SDI application where support for a dialog has been added. The main graphical user interface consists of a dialog window, where parameters can be entered. The dialog is initiated from the main application window (main frame). When the user selects New Test from the menu in the main frame, the ***OnFileTesting()*** function is called. The function creates a new ***ShockTestDlg()*** (IDD_SHOCKTEST_DLG). All dialogs in a Visual C++ MFC program are usually created as modal dialogs, which means that the user can not interact with any other parts of the application's user interface while the dialog is active [18]. When the shock test dialog is created, it is necessary for the dialog to be created as a modeless dialog instead, which allows the dialog to interact with other application windows. The user and the program needs to be able to interact with the main frame of the application. For example program commands and parameters are written into the main frame while the shock test dialog is active. The ***OnFileTesting()*** function creates the shock test dialog as a modeless dialog.

The shock testing application also uses dialogs when asking the user to wait when the system is being initialized and when error messages are shown. These dialog windows are modal.

5.2.4 Message handling

Messages are generated by the data acquisition device when different events happen. When a buffer is done (buffer is in the done queue), a OLDA_WM_BUFFER_DONE message is sent from the device. Unfortunately, the messages are sent as Windows User Messages (WM_USER+100 to WM_USER+114), which cause some problems when used in the MFC VC6 C++ application. The application itself also uses Windows User messages for different button events. This means that the messages telling the application that the buffer is ready (done) do not come through. Instead, a Windows timer was chosen as a solution. The Windows timer (WM_TIMER) is started in the application when the button "Initialize System" is clicked. The timer (with ID 456) calls the ***OnTimer()*** event every 2 seconds where it checks if there is at least one buffer waiting in the ready queue.

5.2.5 Buffers

There are 4 buffers used (#define NUM_BUFFERS 4) for the data acquisition operations of the module. The size of each buffer is configured in the ***InitSubsystem()*** function. The size of one buffer is equal to the amount of samples (sample rate) that the user defines. If two channels are used, the size of the buffer is multiplied by two. How the samples are stored in the buffer when one or multiple channels are used is described in figure 5.2.

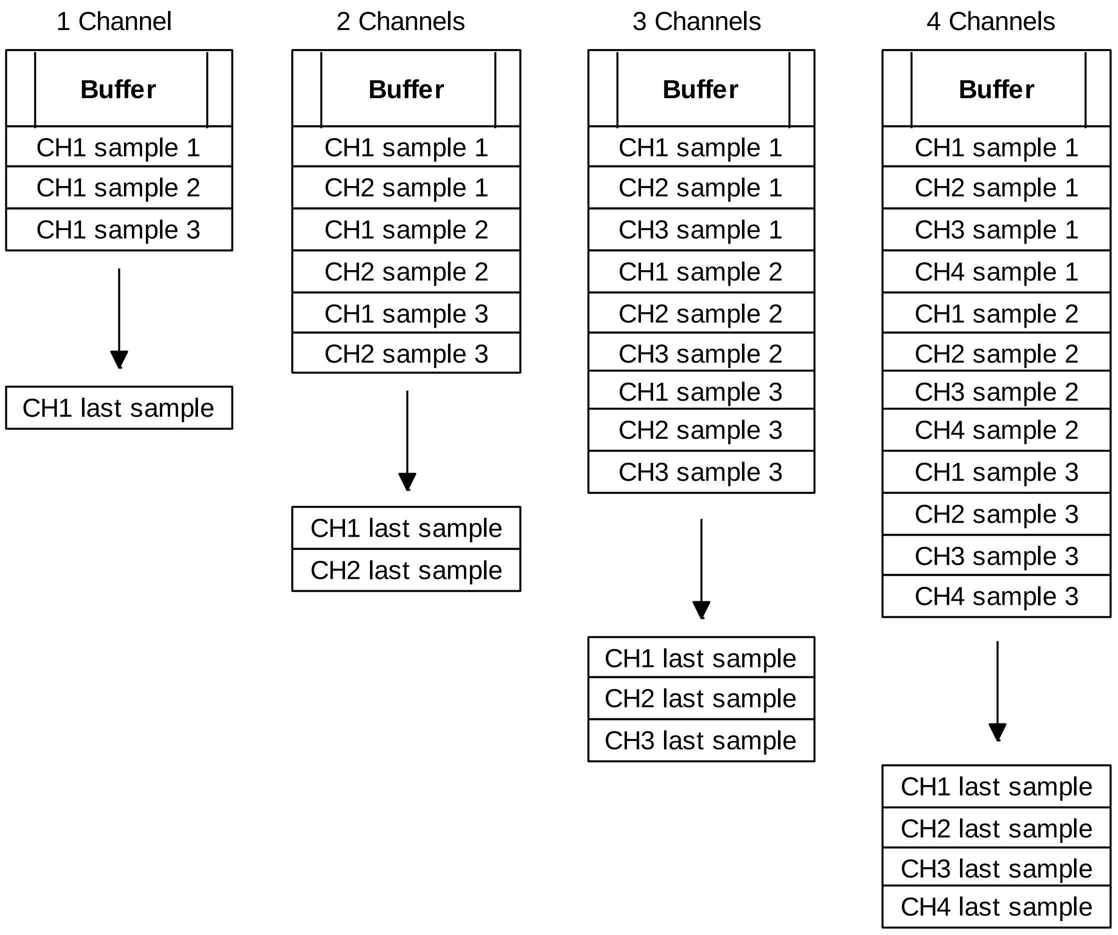


Figure 5.2: Storing samples in buffer

The values of the samples are first converted into voltages according to the example in the Data Acquisition SDK manual in the “Convert Values to Voltage” section [12]. Then the acceleration value is calculated by dividing the voltage value (in millivolts) by the acceleration sensor sensitivity value (that has been obtained from the configuration file). All the operations that handle the buffers are described in figure 5.3.

The ready queue contains buffers that are empty and ready for input. The in-process queue holds the buffer that the device is filling with samples. When the module is done filling the buffer, the buffer is moved from the in-process queue to the done queue. After a buffer has been processed, the buffer is put back into the ready queue.

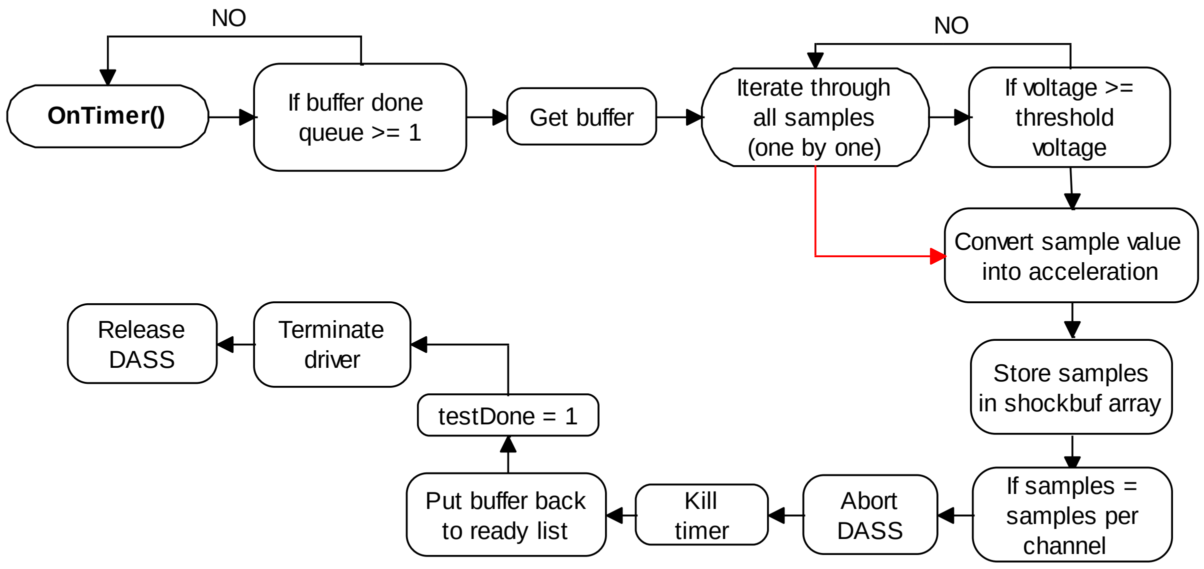

NO
OnTimer()
If buffer done queue >= 1
Get buffer
NO
Iterate through all samples (one by one)
If voltage >= threshold voltage
Convert sample value into acceleration
Store samples in shockbuf array
If samples = samples per channel
Abort DASS
Kill timer
Put buffer back to ready list
testDone = 1
Terminate driver
Release DASS

Figure 5.3: Buffer handling operations

5.2.6 Triggers

Two triggering methods are used in the application. One trigger is an external positive rising edge trigger that is configured in the ***InitSubsystem()*** function. To use the external trigger, the user selects the external positive edge trigger in the user interface (in the “Trigger” section). The trigger occurs when the pendulum hammer hides the laser beam. The external trigger makes it possible to calculate the velocity of the hammer.

DataAcq SDK also supports the use of a software trigger, but the software trigger is not supported by the DT9837B module. A software trigger has instead been created in the C++ application. The software trigger occurs when the voltage level of the signal from the accelerometer is greater than a threshold value in millivolts. The user assigns the trigger threshold level (in g’s) in the user interface (in the “Trigger” section). If the threshold level is achieved, the data acquisition is started.

5.2.7 Error handling

Each of the DataAcq SDK functions return an error code (ECODE ec). If the function executes correctly, the error code is 0 (no error occurs). Other error codes indicate that an error has occurred. The shock test application checks for errors every time it calls a DataAcq SDK function. The errors are checked with the ***checkError()*** function and error messages are shown with the ***showError()*** function. After an error has occurred and the error message is shown, the data acquisition subsystem is released and the device driver is terminated. All the error codes are defined in the DataAcq SDK library header file “OLERRORS.h”. The most commonly occurring errors are described in table 5.1.

Table 5.1: Most commonly occurring errors in the application and how to solve them

Error definition	**Error**	**Comment**	**Solution**
OLSUBSYSINUSE	Subsystem in use	Some other application in the PC might use the module	Check if any other application uses the DT9837B module, close that program
OLBADSSHANDLE	Invalid Subsystem Handle	The module has been disconnected to the PC during the testing procedure	Connect the module to the PC again and restart the application
OLNOBOARDSINSTALLED	No DT-Open Layers boards installed	The module is not connected to the PC	Connect the module to the PC

5.2.8 Configuration file handling

The program file “Config.cpp” and header file “Config.h” are used for handling the configuration file. The files are a part of an external library, used for configuration file handling and has not been created during this work. The files have been imported to the project to call the functions from other program files. When the “Load Configuration” button is clicked, first a dialog window appears that gives the user the chance to select which configuration file he wants to use. Then the ***ResetConfigFile()*** function is called that resets the memory for the configuration files. Thereafter the ***SetConfigFile()*** function is called to set the chosen configuration file. The ***ReadconfigFiles()*** reads the configuration files. The ***GetConfigPar()*** looks through the defined section, name and value of the configuration parameter and checks if the value exists (is greater than 0). If the parameter exists, the value is assigned to a system variable to be used in the program and written to the graphical user interface. A sample configuration file can be seen in Appendix B.

5.2.9 Document/View architecture

The shock test application uses the MFC document/view architecture that consists of a document that stores data and a view that has access to the data. The maintenance of data is separated from the displaying of the data. In the application, the document/view architecture is used when displaying the system parameters and log from the application. It gives the user the opportunity to follow the process of the application.

The file “MechShockView.cpp” contains a function ***AddData()*** that is called from other files and functions when there is data that needs to be displayed in the main window of the application.

5.2.10 Resources

Resource.h is a header file that has been generated by Microsoft Visual C++. It includes definitions for the resources used by the project. The resources are for example the button and variable definitions that are used by the graphical user interface.

5.2.11 Logging

The shock testing process is automatically logged by the application. The following is logged:

- the date and time of the test procedure,
- the parameters that the module was initialized with:
    - sampling frequency per channel,
    - the duration of the data acquisition,
    - number of channels,
    - number of samples per channel,
    - damping ratio,
- if external trigger is used, the velocity of the hammer.

The test log gives a very good overview of the parameters used during shock testing and can be attached to the test report. The logging is done using an ofstream object that saves the parameters that the system was initialized with into a log file called “testlog.txt”. If many tests are done in a row, the test log contains information about all of the tests. The different tests can are separated by the time information. An example test log can be seen in Appendix D.

5.2.12 Generating reports

CMarkup library is used to generate data for the shock test reports. Currently, by clicking the "Generate Report" button, an xml file is generated that contains test requirements data, such as the required acceleration values at different frequencies.

5.3 Implementation of the SRS Algorithms

5.3.1 Smallwood algorithm

The Smallwood algorithm code is written by Tom Irvine [16]. The code has been modified to calculate the maximax response from the positive and absolute negative response spectrum. Instead of only calling one ***Core()*** function that calculates the response, the shock test application uses two functions, ***Core()*** and ***Core2()***. The functions are both the same but use different buffers for the shock response spectrum. ***Core()*** is used for calculating the shock response spectrum from the acceleration time data of the first channel and ***Core2()*** for the second channel.

5.3.2 The Runge Kutta 4th order method

The Runge Kutta 4th order method code (algorithm) is written by Jean-Pierre Moreau [17]. The C++ code has been adapted and improved for the use in the mechanical shock testing software. The code written by Jean-Pierre Moreau calculates the minimum and maximum response values (negative and positive spectrum), but not the maximax spectrum of interest. The calculation of the maximax spectrum from the positive and absolute negative spectrum values has been added to the code. The ***Response()*** functions takes the frequency, acceleration signal, sampling step, damping ratio and number of SRS points as input and outputs the response as the result. When the button "Plot SRS (Runge Kutta)" is clicked, the ***Response()*** function is called and the shock response spectrum is calculated. Both the negative and positive responses are stored and the maximax spectrum is found. The *Spect1* buffer stores the absolute negative response, *Spect2* buffer stores the maximum response and *Spect3* stores the maximax response spectrum values. The buffers *Spect1_2*, *Spect2_2* and *Spect3_2* are used accordingly if two channels are used.

6 Graphical user interface

6.1 System configuration parameters

The parameters used for shock testing are stored in a configuration file. The user can easily change the test parameters either by editing the configuration file or the edit control fields in the graphical user interface. When the "Load Configuration" button is clicked, the user can choose the configuration file he wants to use. After the file has been chosen, the parameters in the configuration file are scanned into the application and saved as system variables. All the configuration parameters and their functions are described in table 6.1. The graphical user interface can be seen in figure 6.2.

Table 6.1: System configuration parameters and their functions

Parameter in configuration file	**Parameter in program**	**Type**	**Value range**	**Function**	**Can be edited in**
samplerate	sampl	long	1000 - 100000	Sampling frequency per channel (Hz)	configuration file, graphical user interface
damping	damp	double	0 - 0.99	Damping Ratio	configuration file, graphical user interface
ch	channels	int	1 - 4	Number of channels to be initialized	configuration file, graphical user interface
time	Tim	double	0.00001 - 1	Duration of the data acquisition in seconds	configuration file, graphical user interface
TESTfreq	TESTfreq[5]	double	100 - 10000	Shock test specification levels (frequency)	configuration file
TESTacc	TESTacc[5]	double	1 - 8000	Shock test specification levels (acceleration)	configuration file
posTolerance	posTol	int	not applicable	Tolerance in dB	configuration file
negTolerance	negTol	int	not applicable	Tolerance in dB	configuration file
sens1	sensor1[80], sensit1	char, double	not applicable	Sensor 1 serial number and sensitivity (mV/g)	configuration file
sens2	sensor2[80], sensit2	char, double	not applicable	Sensor 2 serial number and sensitivity (mV/g)	configuration file

sens3	sensor3[80], sensit3	char, double	not applicable	Sensor 3 serial number and sensitivity (mV/g)	configuration file
sens4	sensor4[80], sensit4	char, double	not applicable	Sensor 4 serial number and sensitivity (mV/g)	configuration file
unit	units[10]	char	not applicable	Acceleration units for displaying results (m/s^2 or g)	configuration file

6.1 Button events

The events that occur when a button is clicked are described in table 6.2.

Table 6.2: Button events

Button	Event	Enabled at Start	Comment
Load Configuration	Loads configuration parameters from config file	Yes	The user can choose the configuration file for loading the system parameters
Initialize System	Initializes the system with user-defined parameters	Yes	Disabled when clicked, enabled again when data acquisition is done
Plot Acceleration CH1	Plots the acceleration time history of channel 1	Yes	-
Plot Acceleration CH2	Plots the acceleration time history of channel 2	Yes	-
Plot Acceleration (All Channels)	Plots the acceleration time history of all channels	Yes	-
Output to file	Saves the acceleration time history to csv file	No	Enabled when Initialize System button is clicked
Plot Acceleration gnuplot	Plots the acceleration time histories using gnuplot	No	Enabled when Output to file button is clicked
Plot SRS (Smallwood)	Plots the SRS calculated with the Smallwood algorithm	Yes	-
Plot SRS (Runge Kutta)	Plots the SRS calculated with the Runge Kutta method	Yes	-

SRS gnuplot (Smallwood)	Plots the SRS calculated with the Smallwood algorithm using gnuplot	No	Enabled when Plot SRS (Smallwood) button is clicked
SRS gnuplot (Runge Kutta)	Plots the SRS calculated with the Runge Kutta method using gnuplot	No	Enabled when Plot SRS (Runge Kutta) button is clicked
SRS gnuplot (All Algorithms)	Plots the SRS calculated with both algorithms using gnuplot	No	Enabled when Plot SRS (Smallwood) and Plot SRS (Runge Kutta) buttons have been clicked
Generate Report	Generates an xml file for the test report	Yes	-

6.2 Software user guide

The goal was to create a graphical user interface for the application that is simple to understand and use. All the necessary steps for the user to understand the application are described in detail in the user guide that can be found in Appendix A. The main steps of the user interface are also described here:

Step 1: A new test can be started by running the shock testing application and selecting the option New Test from the File menu (see figure 6.1).

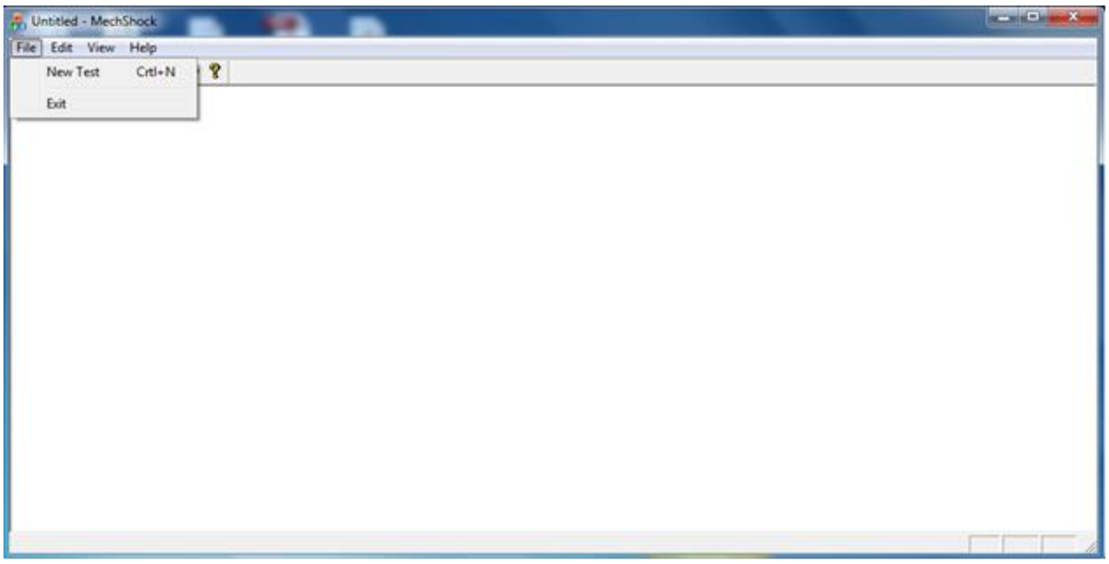
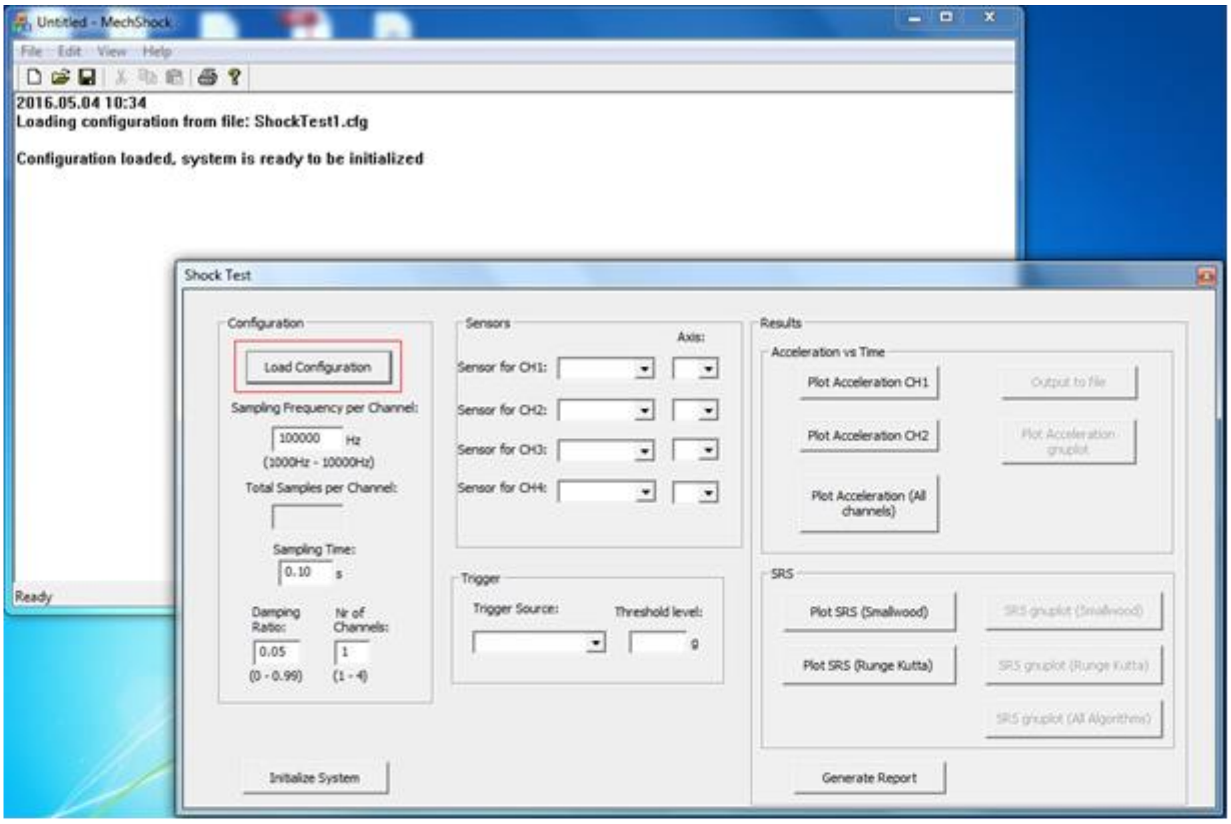


Figure 6.1: Step 1 of the shock testing application. Selecting New Test from the menu to open the dialog window to input parameters

Step 2: The configuration file is loaded by clicking the "Load Configuration" button as it is done in figure 6.2. The user is given the opportunity to choose the configuration file he wants to use.


```
2016.05.04 10:34
Loading configuration from file: ShockTest1.cfg

Configuration loaded, system is ready to be initialized
```

Sampling Frequency per Channel: 100000 Hz (1000Hz - 10000Hz)

Total Samples per Channel:

Sampling Time: 0.10 s

Damping Ratio: 0.05 (0 - 0.99)

Nr of Channels: 1 (1 - 4)

Figure 6.2: The graphical user interface dialog window for inputting parameters

Step 3: If necessary, the parameters loaded from the configuration file can be changed in the user interface dialog window.

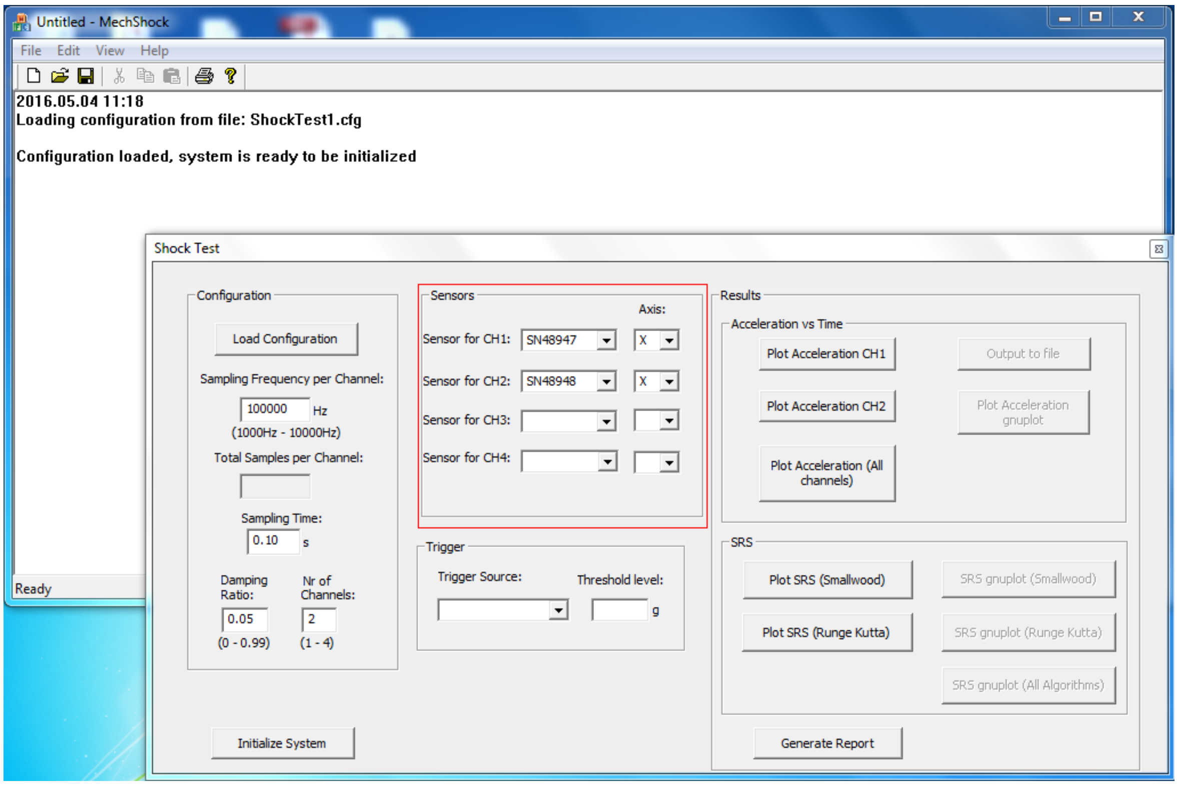
Step 4: The sensor identification names are chosen for the channels (see figure 6.3). This is necessary, because each sensor has a sensitivity parameter in the configuration file that is used when converting voltage to acceleration.

Step 5: The axes (X, Y or Z) are chosen for the sensors.

Step 6: The trigger source and threshold level is selected according to figure 6.4.

Step 7: If all the parameters are correct, the system is ready to be initialized by clicking the "Initialize System" button (see figure 6.5). This process takes 10 seconds.

Step 8: The system is ready for the shock.



```
2016.05.04 11:18
Loading configuration from file: ShockTest1.cfg

Configuration loaded, system is ready to be initialized
```

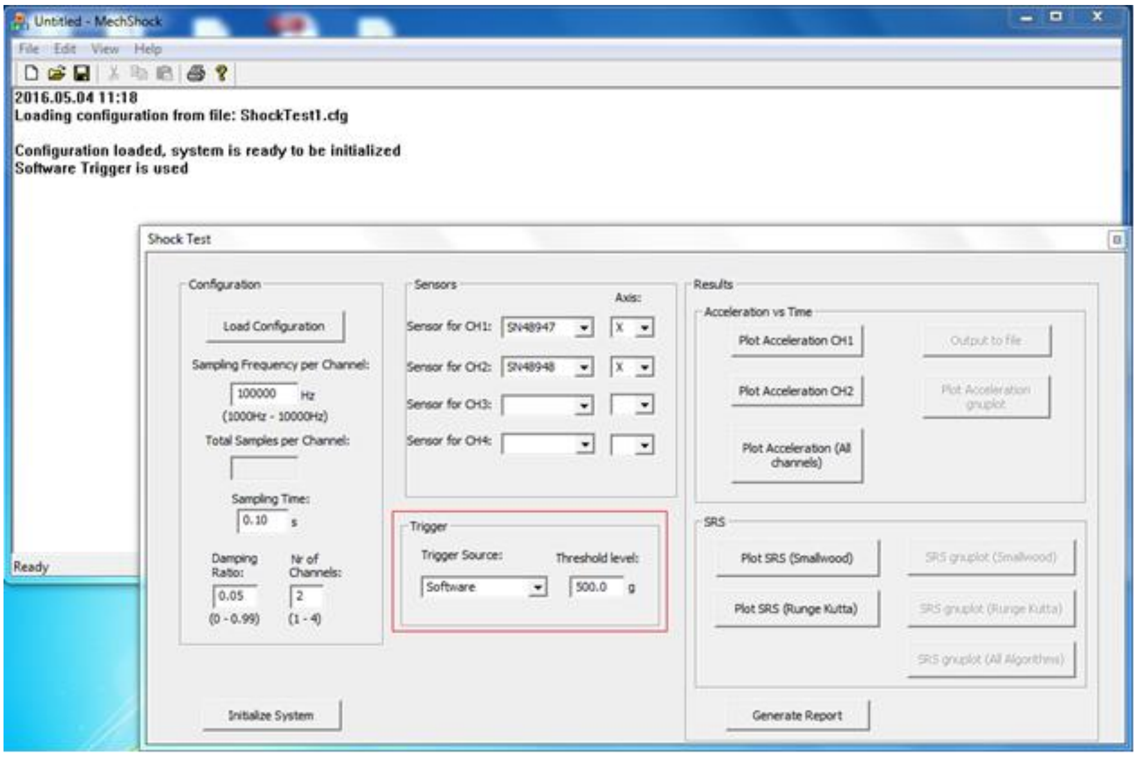
Sampling Frequency per Channel: 100000 Hz (1000Hz - 10000Hz)
Total Samples per Channel:
Sampling Time: 0.10 s
Damping Ratio: 0.05 (0 - 0.99)
Nr of Channels: 2 (1 - 4)

Sensors

		Axis:
Sensor for CH1:	SN48947	X
Sensor for CH2:	SN48948	X
Sensor for CH3:		
Sensor for CH4:		

Trigger
Trigger Source:
Threshold level: g

Figure 6.3: Sensor parameters and testing axes are chosen



```
2016.05.04 11:18
Loading configuration from file: ShockTest1.cfg

Configuration loaded, system is ready to be initialized
Software Trigger is used
```

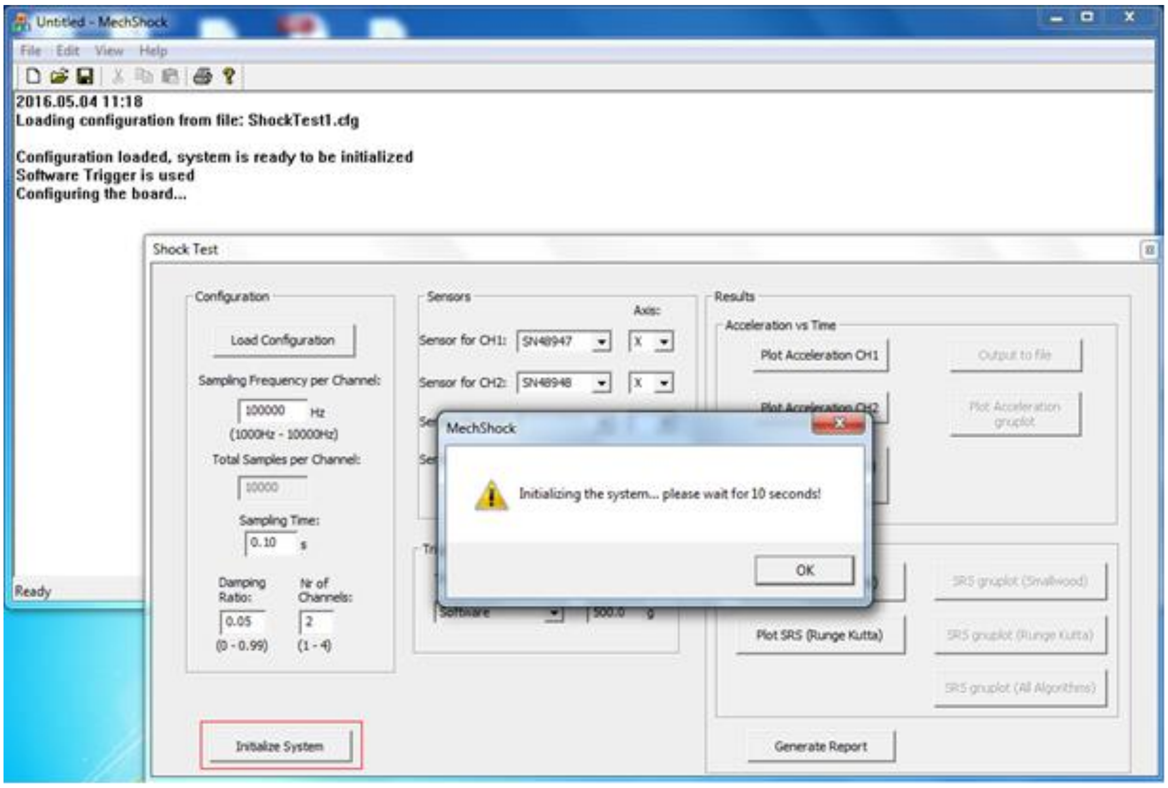
Sampling Frequency per Channel: 100000 Hz (1000Hz - 10000Hz)
Total Samples per Channel:
Sampling Time: 0.10 s
Damping Ratio: 0.05 (0 - 0.99)
Nr of Channels: 2 (1 - 4)

Sensors

		Axis:
Sensor for CH1:	SN48947	X
Sensor for CH2:	SN48948	X
Sensor for CH3:		
Sensor for CH4:		

Trigger
Trigger Source: Software
Threshold level: 500.0 g

Figure 6.4: Selection of the trigger source and threshold level



```
2016.05.04 11:18
Loading configuration from file: ShockTest1.cfg

Configuration loaded, system is ready to be initialized
Software Trigger is used
Configuring the board...
```

Sampling Frequency per Channel: 100000 Hz (1000Hz - 10000Hz)
Total Samples per Channel: 10000
Sampling Time: 0.10 s
Damping Ratio: 0.05 (0 - 0.99)
Nr of Channels: 2 (1 - 4)
Sensor for CH1: SN48947 Axis: X
Sensor for CH2: SN48948 Axis: X
Software 500.0 g

MechShock: Initializing the system... please wait for 10 seconds!

Figure 6.5: Initialization of the system

Step 9: Results from the shock testing can be found in the “Results” section of the user interface. An acceleration-time plot can be seen if “Plot Acceleration CH1” (or CH2) button is clicked. The data can be saved to a file by clicking “Output to file” button.

Step 10: To view the SRS of the shock, the “Plot SRS (Smallwood)” or “Plot SRS (Runge Kutta)” buttons are clicked. After the buttons have been clicked, the “SRS gnuplot (Smallwood)” and “SRS gnuplot (Runge Kutta)” buttons are enabled and display a plot with proper logarithmic scales.

Step 11: A preliminary report can be generated by clicking the “Generate Report” button. This outputs the required shock test specifications into an xml file to be used inside the shock test report.

For the next shock test, the parameters can be changed or left the same and the “Initialize System” button has to be clicked again.

7 Testing and results

The developed software has been tested in the Windows 7 64-bit operating system. When testing the software, it was decided to test the reliability of the shock testing system. It would be valuable to know if the shock system is reliable. The software was used to acquire data from the acceleration sensor and plot the SRSs of various level shocks to verify that the results are almost identical.

7.1 Repeatability of shocks

A very important aspect of shock testing is to be able to recreate shocks of the same severity and with an almost identical SRS by using the same system configuration parameters. For two shocks to be considered identical, the SRSs of the shocks shall lie within 1 dB of each other [3]. If the shock testing system is not able to produce almost identical shocks, the system is not reliable and should not be used for testing [3]. To test the developed shock testing application and also the reliability of the shock testing system, shocks were conducted with both horizontal and vertical testing setups.

7.1.1 Horizontal shock testing results

Three horizontal shocks were conducted from angles of 25°, 30° and 35°. Figure 7.1 shows a plot with three horizontal shocks from a 25° angle, calculated by using the Smallwood algorithm.

Figure 7.2 shows a plot with three horizontal shocks from a 30° angle, calculated by using the Smallwood algorithm. The required SRS and tolerances (+6 dB and -3 dB) are also plotted according to the ESA specifications. Three horizontal shocks from a 35° angle can be seen in figure 7.3.

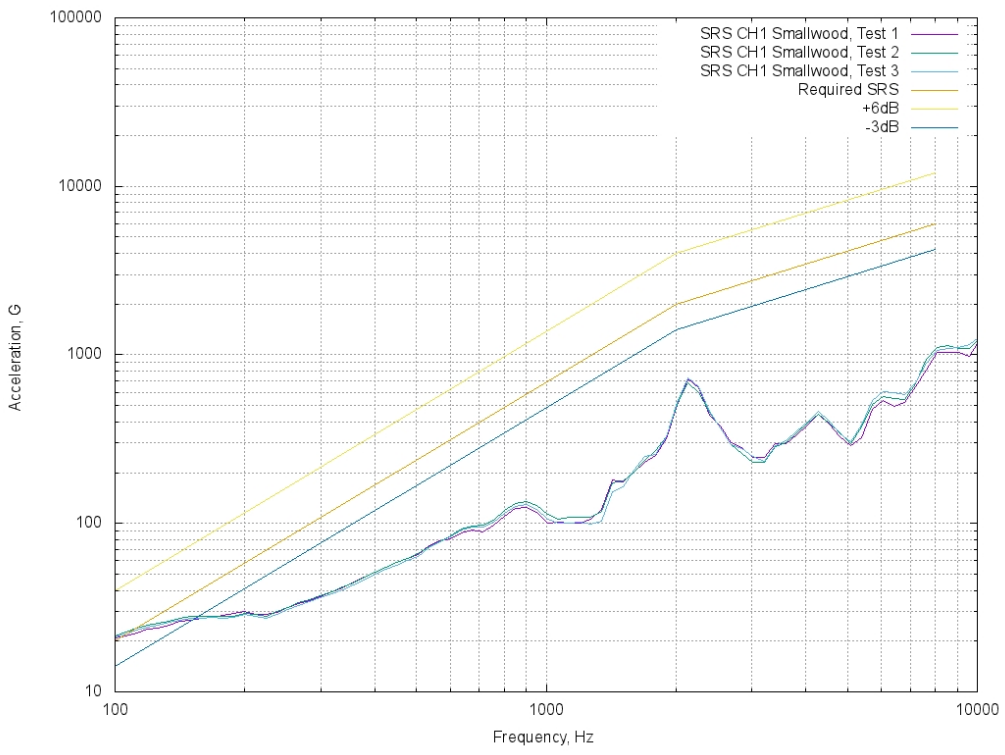


Figure 7.1: The SRS from three horizontal shocks from a 25° angle (using Smallwood algorithm)

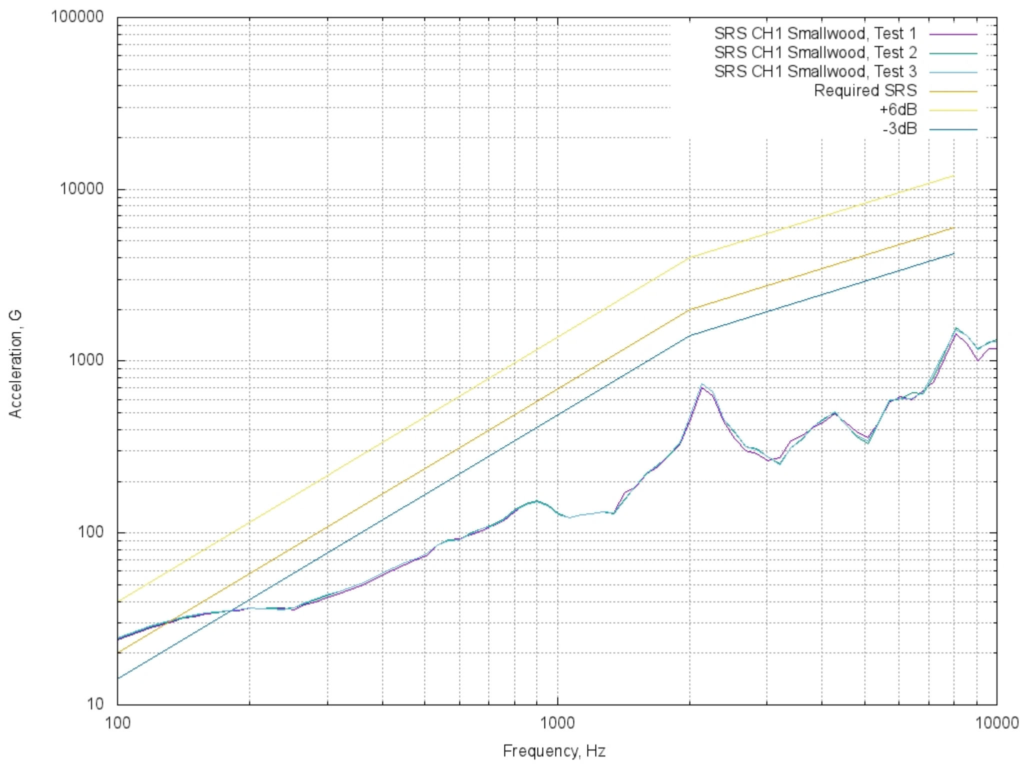


Figure 7.2: The SRS from three horizontal shocks from a 30° angle (using Smallwood algorithm)

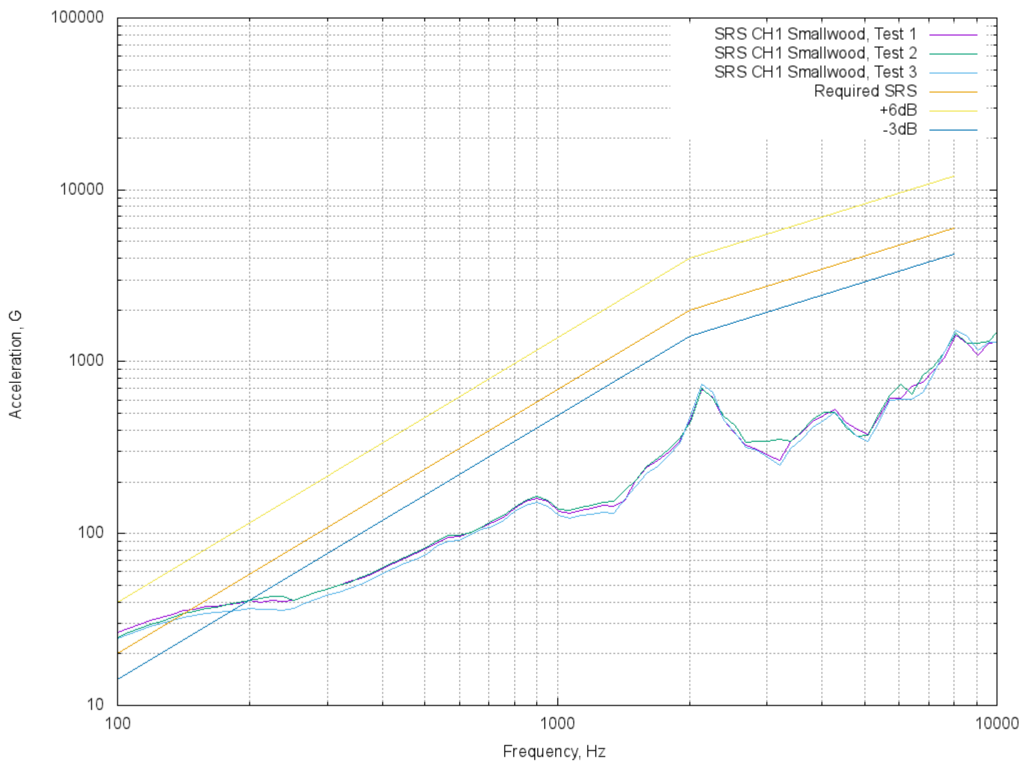


Figure 7.3: The SRS from three horizontal shocks from a 35° angle (using Smallwood algorithm)

7.1.2 Vertical shock testing results

The vertical test setup was used to produce drop shocks from heights from 90 to 110 cm. The required SRS and tolerances (+6 dB and -3 dB) are also plotted for all heights according to the ESA specification. Figure 7.4 shows the SRS plot of a three vertical shocks from a height of 90 cm calculated by using the Smallwood algorithm. The SRS plot of three shocks from a height of 100 cm calculated by Smallwood algorithm can be found on figure 7.5. Figure 7.6 shows the plot of three shocks from the height of 110 cm calculated by the Smallwood algorithm. Figure 7.7 shows the same plot calculated by both the Smallwood and Runge Kutta algorithms.

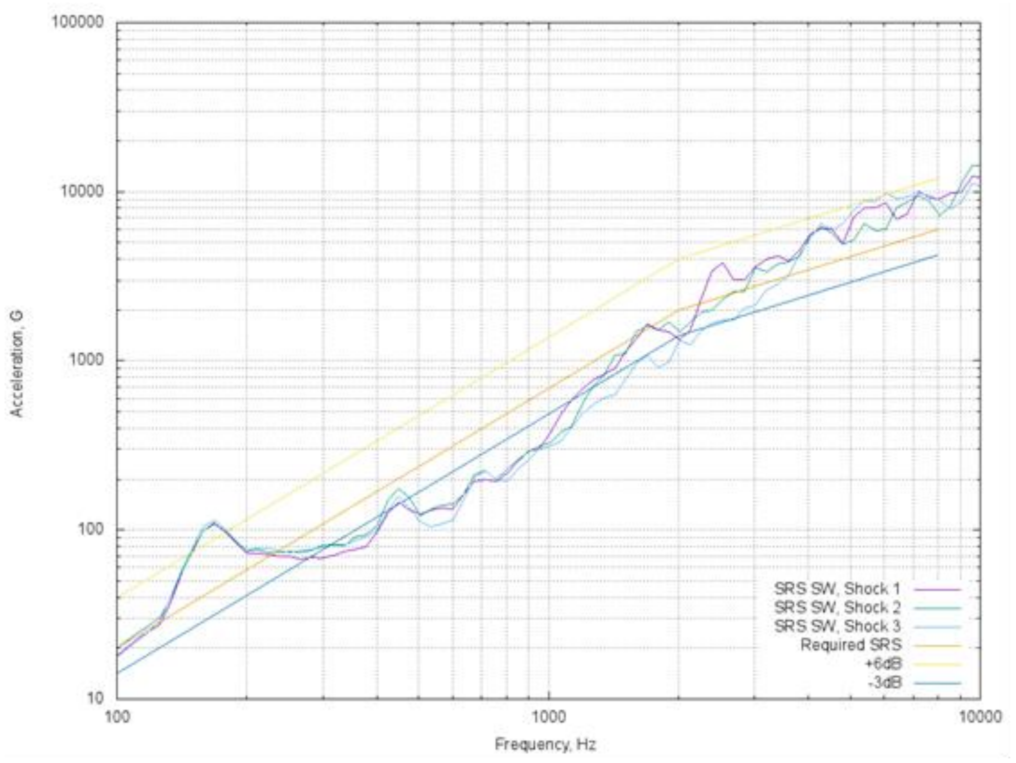


Figure 7.4: The SRS from three vertical shocks from a height of 90 cm (using Smallwood algorithm)

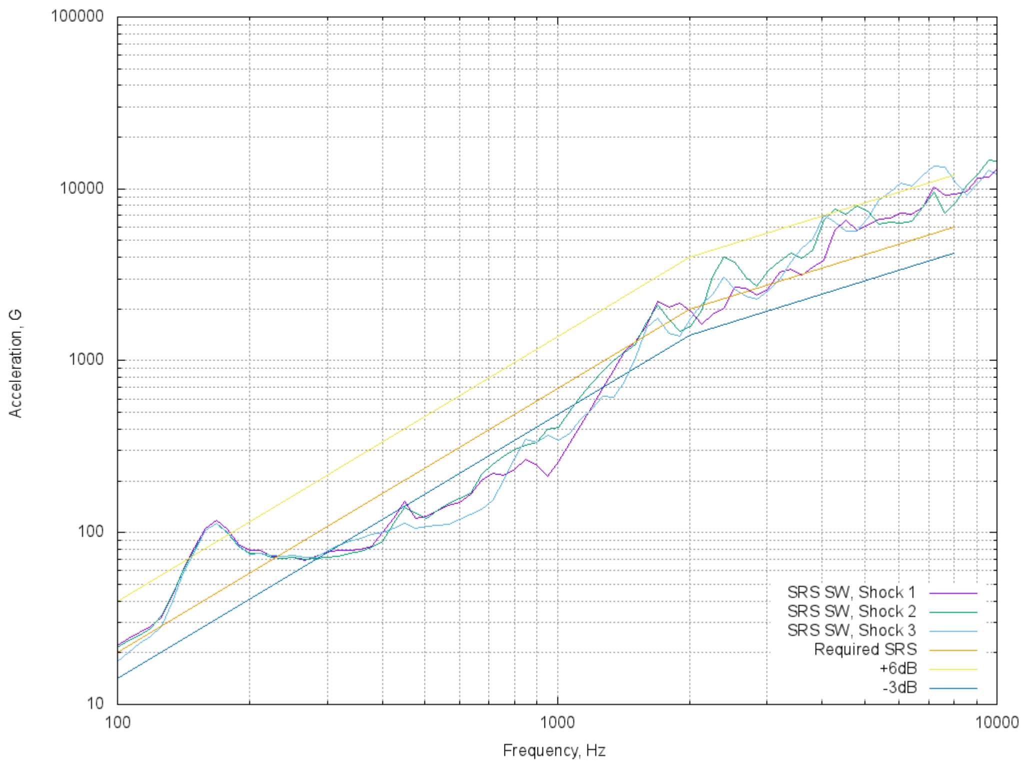


Figure 7.5: The SRS from 3 vertical shocks from a height of 100 cm (using Smallwood algorithm)

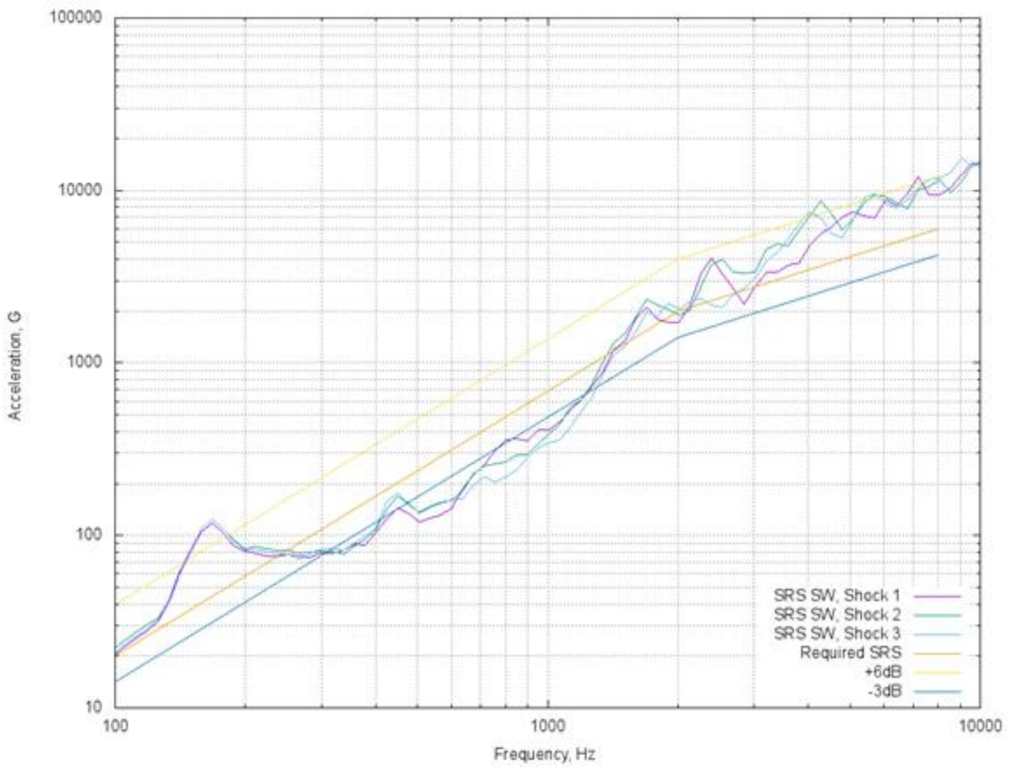


Figure 7.6: The SRS from three vertical shocks from a height of 110 cm (using Smallwood algorithm)

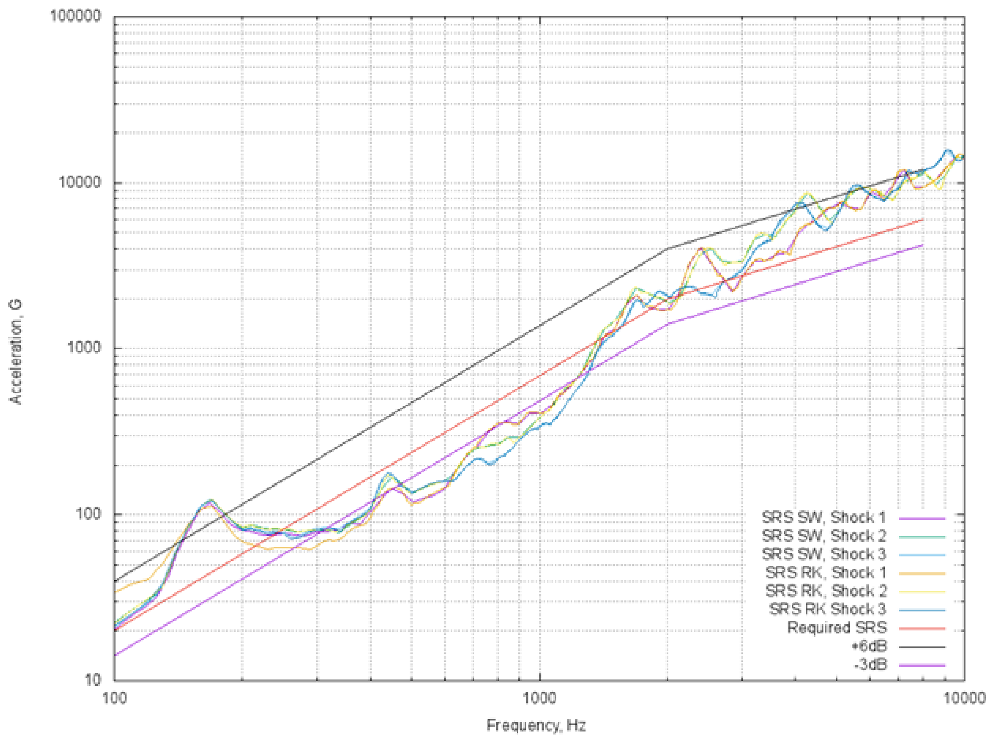


Figure 7.7: The SRS from three vertical shocks from a height of 110 cm (using both Smallwood algorithm and Runge Kutta method)

7.2 Comparing with MATLAB results

A MATLAB testing program was written in order to compare results calculated by using the Smallwood algorithm. The MATLAB Smallwood implementation was used, that is written by Tom Irvine [26]. The code was adapted and some additional code was created. The acceleration-time data from the 25° angle horizontal shock test was processed in the MATLAB program that calculates the SRS using the Smallwood algorithm. In figure 7.8 the same shock data has been used to calculate the SRS by using two different software implementations. The blue line shows the maximax SRS calculated by MATLAB and the green line shows the maximax SRS calculated by the shock testing application, both using the Smallwood algorithm.

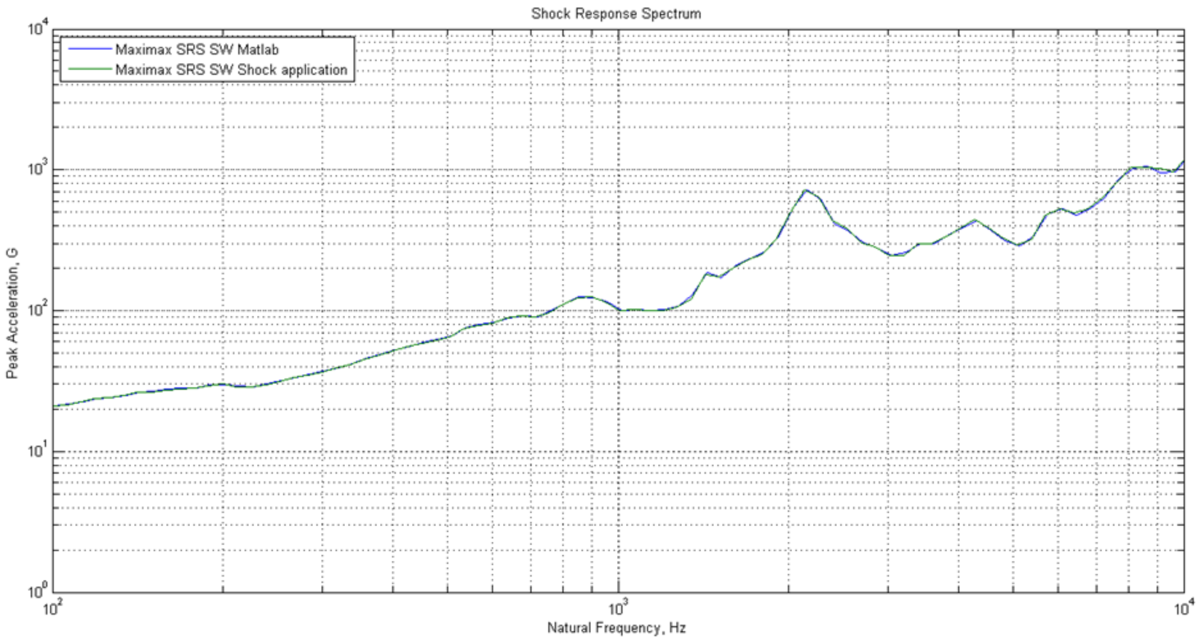


Figure 7.8: The SRS calculated with MATLAB and the shock testing application, using the Smallwood algorithm

7.3 Analysis of results

By looking at the shock test results from the horizontal and vertical test setup, the conclusion can be made that it is easier to recreate different severity shocks by using the horizontal test setup. The results from the horizontal test setup are more reliable than the results from the vertical test setup. The vertical test setup will be improved in the near future.

7.4 Test report template

A test report template was created for the shock testing system. Since the shock testing software has already been used for commercial testing, a sample test report that was created which can be found in Appendix A. The test report is in compliance with the Tartu Observatory test reports.

8 Conclusions and future work

In the scope of this work, the shock testing system in Tartu Observatory's space technology laboratory was described and software for the shock testing system was developed in C++ in Visual Studio .NET 2003. The main goal of the work was to make the shock testing system operational and ready for conducting shock tests for space technology components.

All of the requirements for the software were successfully fulfilled. The created graphical user interface is simple to understand and can easily be understood by reading the user guide that was also created in the scope of this thesis.

The software was tested by conducting shocks to test the reliability of the shock testing system. From the results it can be clearly seen that the horizontal shock testing setup is more reliable than the vertical setup. The vertical setup produces shocks with greater acceleration levels (up to 6000 g) with less effort than the horizontal setup. The results confirm that the shock testing system can be used for space technology testing

To make the shock testing system as efficient as possible for the user, there are a few improvements that will be done in the future such as:

- the PGL library will be improved and support for logarithmic scales will be added;
- the functionality of the report generation will be improved and a more advanced library might be used;
- calculation of how much of the SRS data is located outside the required SRS tolerances will be added;
- support for additionally two acceleration sensor channels will be added (a total of 4 channels will be supported at the same time);
- the ability to load shock test data (acceleration-time histories) from a file to be processed and plotted in the application will be added;
- the vertical test setup will be improved to achieve better repeatability of shocks.

Summary

Shock testing is used in the space industry to ensure that components used in space can withstand shocks that occur during the launch and separation phases of the space vehicles. A shock is usually described as an acceleration-time history, but in order to be able to compare and recreate shocks of various levels the shock response spectrum is used in the space industry. It plots the peak acceleration of different single degree of freedom systems with unique natural frequencies over a range of frequencies.

The result of this thesis work is a software solution which is compatible with the Data Translations DT9837B data acquisition module and the shock testing system at Tartu Observatory. The software is developed in Microsoft Visual Studio .NET 2003 using C++. It enables the user to easily perform shock testing and output the results as acceleration-time history and shock response spectrum plots. During the work, a software user guide has also been created to ensure that test engineers in Tartu Observatory are able to use the software and perform shock testing. A test report standard was also created where the results from the testing will be inserted.

The developed software enables to:

- select a configuration file from where to load the system configuration parameters into the application and/or edit the parameters in the graphical user interface,
- plot acceleration-time histories (in both g's and m/s^2) for up to 2 acceleration sensor channels,
- calculate the SRS by using two different algorithms (Smallwood algorithm and Runge Kutta 4th order method),
- save the plots on the computer,
- log commands and system parameters in the main window of the application and in a separate log file,
- generate data for the shock reports.

The software application was tested in real shock environment, by plotting the SRS from shocks of various levels. The results verify that the shock testing system at Tartu Observatory

is able to produce shocks up to 6000 g and that it is a reliable system to use when performing space technology shock testing.

Kokkuvõte

Tarkvara arendamine Tartu Observatooriumi põrutuskindluse katsesüsteemile

Kosmosetööstuses teostatakse põrutuskindluse testimist, et tagada kosmoses kasutatavate komponentide vastupidavus põrutustele (e. šokkidele), mis esinevad kanderakettide üleslennutamise ja astmete eraldamise etappidel. Šokke esitatakse tavaliselt kiirendus-aeg graafikutel, kuid selleks, et võimaldada erinevate tugevusastmetega šokke võrrelda ning taasluua on kosmosetööstuses kasutusele võetud šoki kostespekter. Kostespekter võimaldab esitada ühe vabadusastmega süsteemide, millel igal ühel on oma ainulaadne loomulik sagedus, maksimaalsed kiirendused määratud sagedusvahemiku jaoks.

Käesoleva töö tulemuseks on tarkvaralahendus, mis ühildub Data Translations'i DT9837B andmehõivemooduliga ja Tartu Observatooriumi põrutuskindluse süsteemiga. Tarkvara sai arendatud Microsoft Visual Studio .NET 2003 keskkonnas kasutades C++ programmeerimise keelt. Tarkvaralahendus võimaldab kasutajal teostada põrutuskindluse testimist ning väljastada tulemused kiirendus-aeg ja šoki kostespektri graafikutel. Töö käigus on ka valminud kasutusjuhend loodud tarkvarale, et Tartu Observatooriumi kosmosetehnoloogia labori insenerid saaksid süsteemi kasutamise selgeks ning selle abil viia läbi põrutuskindluse testimist. Samuti on valminud testimise jaoks katseraporti põhi, kuhu saab testimise tulemused sisestada. Loodud tarkvara võimaldab:

- kasutajal süsteemi käivitamise parameetrid konfiguratisoonifailist programmi lugeda või neid graafilisest kasutajaliidesest muuta,
- kuvada kiirendus-aeg graafikuid kuni kahe andmehõivesüsteemi kiirendusanduri jaoks,
- arvutada šoki kostespekter kasutades kahte erinevat algoritmi (Smallwoodi algoritm ja Runge Kutta 4. järku meetod),
- salvestada saadud graafikud arvutisse,
- väljastada programmi kasutamise ajal logi, kus süsteemi olulisemad parameetrid on välja toodud,
- genereerida raportite jaoks andmeid

Loodud tarkvara katsetati viies läbi erineva tugevusastmega šokke ning salvestades ja esitades šokkide kostepektreid. Tulemused kinnitavad, et Tartu Observatooriumi põrutuskindluse süsteem võimaldab tekitada šokke kiirendusega kuni 6000 g’d ning on usaldusväärne süsteem kosmosetehnoloogia põrutuskindluse testimise läbiviimiseks.

Acknowledgements

First of all, I would like to thank my supervisor Viljo Allik for always giving great advice for and suggestions on how to improve the developed software and his feedback on the thesis. Then I would like to thank my co-supervisor Riho Vendt for his guidance with the test report and all of his feedback. Also I would like to thank my co-supervisor Teet Tilk for giving me ideas on how to make the software application even better. I would also like to thank Astrid Pung, who has given valuable feedback on the thesis. I am grateful for having the chance to work in the space technology laboratory at Tartu Observatory.

I would last, but not least, thank my family and close ones who have been a great support during this intense period. Especially, I would like to thank my mother, Hiie Allik, who never ceases to believe in me, even when I doubt myself.

[illegible]

References

[1] Space engineering, Spacecraft mechanical loads analysis handbook, ECSS-E-HB-32-26A, Noordwijk, The Netherlands, February 2013

[2] Radius Space, http://www.radiusspace.com/ [Online; accessed April 2016]

[3] M. Jonsson (2012), Development of a Shock Test Facility for Qualification of Space Equipment (Master’s thesis, Chalmers University of Technology, Göteborg, Sweden)

[4] T. Irvine, An introduction to the Shock Response Spectrum, http://www.vibrationdata.com/tutorials2/srs_intr.pdf [Online; accessed November 2015]

[5] Signalysis, Shock Response Spectrum Analysis http://signalysis.com/company/signalysis-at-work/shock-response-spectrum-analysis/

[6] R. Melander, S. Smith, *Why Shock Measurements Performed at Different Facilities Don’t Agree*, Shock & Vibration Symposium October 1995

[7] DAQ Adaptor for MATLAB, Data Translation User Manual 22024-K, Tenth Edition, Marlboro, USA, January 2014

[8] D. O. Smallwood, *An Improved Recursive Formula for Calculating Shock Response Spectra*, Sandia National Laboratories, Albuquerque, New Mexico.

[9] DT9837 Series User’s Manual, Data Translation User Manual 22417-AE, Twenty Nineth Edition, Marlboro, USA, April 2015

[10] DT9837 Series Datasheet, Data Translation, Marlboro, USA

[11] Data Translation, www.datatranslation.com [Online; accessed November 2015]

[12] DataAcq SDK User’s Manual, Data Translation User Manual 18326-AC, Twenty Fifth Edition, Marlboro, USA, January 2015

[13] Introduction to Piezoelectric Accelerometers, PCB Piezotronics, http://www.pcb.com/techsupport/tech_accel [Online; accessed March 2016]

[14] Model M350B03 Shear ICP Shock Sensor Installation and Operating Manual 18292-B, PCB Piezotronics, NY, USA

[15] G. Berhanu (2011), Vibration Durability Testing and Design Validation Based on Narrow Frequency Band (Master’s thesis, Blekinge Institute of Technology, Karlskrona, Sweden)

[16] Vibration data, http://www.vibrationdata.com/ [Online, accessed October 2015]

[17] J-P. Moreau, Program to demonstrate the Acceleration Shock Spectrum, C++ version, Paris, http://jean-pierre.moreau.pagesperso-orange.fr/Cplus/tshocksp_cpp.txt [Online; accessed October 2015]

[18] Modeless Dialog Boxes, http://www.informit.com/library/content.aspx?b=Visual_C_PlusPlus&seqNum=64 [Online; accessed December 2015]

[19] Microsoft Dev Center, GDI+, https://msdn.microsoft.com/en-us/library/windows/desktop/ms533798(v=vs.85).aspx [Online; accessed November 2015]

[20] Code Project, Plot Graphics Library, http://www.codeproject.com/Articles/1546/Plot-Graphic-Library [Online; accessed October 2015]

[21] gnuplot, http://www.gnuplot.info/ [Online; accessed February 2016]

[22] Firstobject developer network, Cmarkup: fast simple C++ XML parser, http://www.firstobject.com/dn_markup.htm [Online; accessed March 2016]

[23] C. B. Macdonald (2003), Constructing High-Order Runge-Kutta Methods with Embedded Strong-Stability-Preserving Pairs (Master's thesis, Simon Fraser University, Canada)

[24] Calibration Certificate, "Calibration Certificate for ICP Acceleration Sensor 48947, PCB Piezotronics", 2015

[25] Calibration Certificate, "Calibration Certificate for ICP Acceleration Sensor 48948, PCB Piezotronics", 2015

[26] Mathworks, Shock response spectrum, a program that calculated the SRS of a time history, T. Irvine, 2006, http://www.mathworks.com/matlabcentral/fileexchange/7398-shock-response-spectrum/content/srs.m [Online; accessed August 2015]

Appendix A: Additional files

Additional files that are included with the thesis are described in table A.1.

Table A.1: Additional files

MechShock.rar	Source code of the developed software and additional files that the software uses (configuration files, gnuplot files)
User_guide_for_shock_testing_software.pdf	The user guide of the shock testing application
TO_SHOCK_003_16.pdf	The shock test report of a tested object

Appendix B: Application configuration file

A sample configuration file ("ShockTest1.cfg") which can be loaded when the user clicks the "Load configuration" button and the system configuration parameters are read from this file, can be seen here:

```
[Parameters]

samplerate = 100000
damping = 0.05
ch = 1
time = 0.1

[TestProfile]

// For ISIS
TESTfreq = 100.0, 2000.0, 8000.0, 8000.0
TESTacc = 20.0, 2000.0, 6000.0, 6000.0

// For ESA
//TESTfreq = 100.0, 1000.0, 2000.0, 10000.0
//TESTacc = 20.0, 1400.0, 3400.0, 3400.0

[Tolerances]

// + 6 dB and -3 dB
posTolerance = 6
negTolerance = 3

[Sensors]
// Sensor identification name, sensor sensitivity mV/g
sens1 = "SN48947", 0.4957
sens2 = "SN48948", 0.4880
sens3 = "None", 0.0
sens4 = "None", 0.0

[Units]
// choosing acceleration units for results
// SI = m/s^2
// Acc_g = g
//unit = SI;
unit = Acc_g;
```

Appendix C: Sample gnuplot script file

A Sample gnuplot script file (with “.plt“ extension) is described in this chapter. This particular plot file is launched by the shock testing application when the “SRS gnuplot (All algorithms)“ button is clicked by the user. The script creates a plot in gnuplot and views in and also saves the same plot as a PNG file.

```
# plot file for all SRS plots (both algorithms)


set terminal png size 1024,768
set output 'SRS_all.png'
set xrange [100:10000]
set log x
set log y
set xlabel "Frequency, Hz"
set ylabel "Acceleration, G"
set key right bottom
set grid xtics ytics mxtics mytics
plot 'CH1_SRS_SW.csv' with lines title "SRS CH1 Smallwood", \
        'CH2_SRS_SW.csv' with lines title "SRS CH2 Smallwood", \
        'CH1_SRS_RK.csv' with lines title "SRS CH1 Runge Kutta", \
        'CH2_SRS_RK.csv' with lines title "SRS CH2 Runge Kutta", \
        'test_profile1.csv' with lines title "Required SRS", \
        'test_profile2.csv' with lines title "+6dB", \
        'test_profile3.csv' with lines title "-3dB"
set datafile separator ","


set term windows
set xrange [100:10000]
set log x
set log y
set xlabel "Frequency, Hz"
set ylabel "Acceleration, G"

set key right bottom

set grid xtics ytics mxtics mytics

plot 'CH1_SRS_SW.csv' with lines title "SRS CH1 Smallwood", \

        'CH2_SRS_SW.csv' with lines title "SRS CH2 Smallwood", \

        'CH1_SRS_RK.csv' with lines title "SRS CH1 Runge Kutta", \

        'CH2_SRS_RK.csv' with lines title "SRS CH2 Runge Kutta", \

        'test_profile1.csv' with lines title "Required SRS", \

        'test_profile2.csv' with lines title "+6dB", \

        'test_profile3.csv' with lines title "-3dB"

set datafile separator ","

pause -1
```

Appendix D: Sample test log

```
2016.04.21 13:47

Initialization done with the following parameters:

Sampling frequency per channel: 100000

Sampling duration: 0.10 s

Number of channels: 1

Total samples per channel: 10000

Damping ratio: 0.05
```