Redesign of Environmental Stress Screening software

Tony Björkman
Abstract

Redesign of Environmental Stress Screening software

Tony Björkman

In this thesis a industrial test process used during the manufacturing of pressure sensors has been examined. The examination is aimed to find areas of improvements related to the software that is used in the process. This test process separates faulty components, in this case circuit boards, before they are assembled into a finished product. The separation is made by letting a software application monitor how the functionality of the circuit boards is affected during stress by high temperature and vibration. How the process is performed from a operator viewpoint was examined in the production plant and suggestions from the operators collected. A preliminary design specification, that was supplied at the beginning of the thesis, was completed with the information that was gathered. The key improvements to the software were that the evaluation of test results should be automatic and the software solution should be more scalable in terms of the amount of circuit boards that it can monitor. The completed specification has been implemented in LabVIEW as a prototype application that contains the key improvements of increased scalability and automatic evaluation of results.
# Contents

1 Introduction .................................................................................................................. 5  
  1.1 Background ........................................................................................................... 5  
  1.2 Problem description .............................................................................................. 5  
  1.3 Objective and tasks ............................................................................................... 5  
2 ESS system and process ................................................................................................. 6  
  2.1 Thermal and vibration stress equipment ............................................................... 8  
  2.2 Electronics to be screened .................................................................................... 8  
  2.3 Jigs ....................................................................................................................... 9  
  2.4 Interface Devices ................................................................................................ 9  
  2.5 Logging software ................................................................................................ 10  
  2.6 Log files ............................................................................................................. 12  
  2.7 Operator work flow to conduct ESS ................................................................. 12  
  2.8 Time consumption for ESS .............................................................................. 15  
3 Method ........................................................................................................................... 15  
  3.1 Study of existing system ....................................................................................... 16  
  3.2 Specification of requirements ............................................................................. 16  
  3.3 Implementation of prototype application in LabVIEW ...................................... 16  
4 ESS system’s specifications and improvements ............................................................. 16  
  4.1 Areas of improvements ....................................................................................... 16  
  4.2 Improving the preliminary specification ............................................................ 17  
  4.3 Changes made on the preliminary specification .............................................. 18  
  4.4 New features added to the specification ......................................................... 19  
5 Interface application .................................................................................................... 20  
  5.1 Software activity diagram ................................................................................... 21  
  5.2 Software class diagram ...................................................................................... 22  
  5.3 Software implementations ............................................................................... 22  
  5.4 Clusters .............................................................................................................. 31  
6 Control Application ...................................................................................................... 34  
  6.1 Control Application GUI .................................................................................... 34  
  6.2 Software activity diagram .................................................................................. 36  
7 Performance test ........................................................................................................... 37  
8 Results and discussion .................................................................................................. 37  
  8.1 Reducing operator workload ............................................................................ 37  
  8.2 Improvement of capacity ................................................................................. 38  
  8.3 Source code ....................................................................................................... 38  
  8.4 Discussion .......................................................................................................... 39  
9 Suggestions for future work ......................................................................................... 39  
10 Reference .................................................................................................................... 39
11 Appendix.............................................................................................................................................40
  11.1 Specifications of software requirements................................................................................40
  11.2 Images.......................................................................................................................................45
  11.3 Preliminary design specification..............................................................................................48
  11.4 Performance test results........................................................................................................50
  11.5 Table of circuit board setup configuration..............................................................................51
## Definitions

<table>
<thead>
<tr>
<th>Term</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADC</td>
<td>Analog to digital converter</td>
</tr>
<tr>
<td>circuit board</td>
<td>Printed circuit board containing electrical components</td>
</tr>
<tr>
<td>Class diagram</td>
<td>A kind of UML diagram that describes the classes used within an object-oriented software design.</td>
</tr>
<tr>
<td>Climate chamber</td>
<td>An enclosure where the interior temperature is controllable in a wide temperature range and used for stress testing electronics.</td>
</tr>
<tr>
<td>Cluster</td>
<td>A datastructure in LabVIEW that holds multiple variables (indicators or controls).</td>
</tr>
<tr>
<td>Control</td>
<td>User operated input variable in LabVIEW. Usually visible in GUI.</td>
</tr>
<tr>
<td>DAQ</td>
<td>Data Acquisition – sampling a signal and converting it to a digital numeric value.</td>
</tr>
<tr>
<td>ESS</td>
<td>Environmental Stress Screening – A process to ensure the quality of final products in manufacturing by exposing a newly manufactured component to stresses such as thermal cycling and vibration. This in order to provoke latent defects to manifest themselves by permanent failure during the screening process.</td>
</tr>
<tr>
<td>Full error</td>
<td>Term used in this report to describe the state of a circuit board that has failed the screening criteria.</td>
</tr>
<tr>
<td>GUI</td>
<td>Graphical User Interface.</td>
</tr>
<tr>
<td>Indicator</td>
<td>Component in the GUI for indicating the value of a variable to the user.</td>
</tr>
<tr>
<td>INI-file</td>
<td>A format for text files defining keys with values, and used for saving and loading settings</td>
</tr>
<tr>
<td>Term</td>
<td>Definition</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Interface device</td>
<td>A term used in this report for DAQ-devices and other devices that are connected to a computer to provide I/O functionality.</td>
</tr>
<tr>
<td>Jig</td>
<td>A hardware fixture used to aid in testing of circuit boards by providing connectivity between circuit boards and a test computer.</td>
</tr>
<tr>
<td>Kjörningstabell</td>
<td>A document to describe which circuit boards and equipment that have been part of a particular screening session.</td>
</tr>
<tr>
<td>Partial Error</td>
<td>A series of consecutive point errors is a partial error. Several partial errors leads to a full error.</td>
</tr>
<tr>
<td>PCB</td>
<td>Printed circuit board.</td>
</tr>
<tr>
<td>Point error</td>
<td>Value outside limits for a tested circuit board.</td>
</tr>
<tr>
<td>Sub-VI</td>
<td>Sub-Virtual Instrument – part of a LabVIEW program encapsulated in it’s own stand-alone application that is accessible for other applications.</td>
</tr>
<tr>
<td>TAB0629</td>
<td>The name of a document containing error limits for the ESS of circuit boards at the company Presens.</td>
</tr>
<tr>
<td>UML</td>
<td>Unified modeling language – A general-purpose modeling language in the field of software engineering, providing a standard way to visualize the design of a system</td>
</tr>
<tr>
<td>Vibration device</td>
<td>A mechanical device capable of causing vibrations for vibration stress test purposes.</td>
</tr>
</tbody>
</table>
1 Introduction

1.1 Background
Presens is a company that was founded in 1996 with the aim of commercializing a technology for pressure sensing developed and patented by the Foundation for Scientific and Industrial Research (SINTEF) in Norway.

The company manufactures and develops high-pressure sensors used in sub sea oil fields and space where component reliability is important. One of the components which greatly affects the reliability is the circuit boards inside the sensors. Since the manufacturing process is not perfect, the resulting circuit boards have varying quality and life expectancy. It is known that the probability of failure for a complex electronic assembly is the highest during the first year of operation, lower in mid-life and then increasing with age. This characteristic is known as a “bath tub” curve with probability of failure as a function of time [5].

A screening process called environmental stress screening (ESS) is used to provoke cards of lower quality to fail prematurely. The ESS process consists of exposing the circuit boards to mechanical and thermal stress while monitoring circuit board function. A circuit board that loses precision or functionality during some point in the screening process shall be closer examined after the test to decide if the circuit board is to be used in a final product or not.

The thermal stress in the screening is generated by putting circuit boards in a climate chamber and the vibration stress is created by strapping the circuit boards on to a vibration device.

To enable the monitoring of circuit boards during screening, a variety of equipment and software is used. The equipment needs to interface to each kind of circuit board’s specific protocol and provide a connection to a computer to enable gathering of information. The software in the computer records the measurements of multiple circuit boards and saves these measurements in a log file on company servers. The log file is then analyzed to decide whether or not the circuit boards have passed or not. They are saved on the company servers to provide traceability for the final product.

Different circuit boards and equipment are illustrated in appendix (section 12).

1.2 Problem description
The current system for ESS is not satisfactory, e.g., user friendliness and added features that would automate some of the tasks performed during the ESS are not sufficiently good; and the system is not scalable, which hinders production rate that the company is going to increase. This indicates the needs of improvement on the current system.

1.3 Objective and tasks
The objective of the project is to design a new software solution capable of replacing the current ESS software used. The thesis work is expected to result in a detailed specification for software functionality as well as a prototype application that presents a solution to the problems that are
identified during the study of the ESS-process and current software. The overall objective of the software design is to:

- Minimize time consumption for test setup and result evaluation.
- Allow for higher capacity, specifically in terms of the number of passed circuit boards.

To achieve the objective, the present thesis will, in particular, be concerned with the following aspects:

- Investigating software and application possibilities.
- Specifying a solution based on the current hardware.
- Implementing part of the solution with LabVIEW as programming language.

## 2 ESS system and process

The ESS process is designed to provoke latent defects in electronics into becoming patent defects such that the defects can be detected prior to assembling the electronic components into a product. Both patent and latent defects are introduced in the manufacturing process. The patent defect is a defect that directly and permanently affects the product functionality. Patent defects can in most cases be detected through inspections or by simply conducting a functioning testing on the subject circuit board. A latent defect on the other hand is not necessarily affecting the functionality but risks developing into a patent defect over time which will result in early failure of the product.

ESS originate from environmental testing which is performed as an evaluation process to examine how an electronic product functions in the environment it is built for. This is achieved by using chambers that simulates the in-field environment by controlling temperature, humidity, pressure etc. Environmental testing was used by the American military during second world war to prove the reliability of a newly designed product in a faster way then waiting until the product broke down by itself during in-field use. ESS takes it one step further to increase the amount of stress that the product is exposed to and thus make the product experience an environment worse than in-field with the intent of screening out low quality products before they are put in to use. An example of this is the temperature cycling between hot and cold, and random vibrations which are the techniques that ESS rely on to prove the quality of components or products. Today ESS testing is also used as a cost-effective way of minimizing product failure in consumer electronics not only high-end military, space and offshore products.
At Presens the ESS process is implemented as a two stage process that is done on printed circuit boards prior to final assembly. The two stages are vibration ESS and thermal ESS. These two tests are done completely separately but with the same logging hardware and software. The difference is the environmental stress source. The former is a vibration device and the latter a climate chamber. The hardware setup for thermal ESS is illustrated in Figure 2.0.

![Climate chamber for used for conducting ESS. DAQs can be seen on top.](image)

**Figure 2.0:** Climate chamber for used for conducting ESS. DAQs can be seen on top.

The connections between the different hardware used in a ESS is shown schematically in Figure 2.1. The PCB’s are mounted to a jig and they are placed inside a climate chamber or is fastened to a

![Schematic overview of the ESS process.](image)

**Figure 2.1:** Schematic overview of the ESS process.
vibration device depending on if the screening is for thermal or vibration ESS. The jigs connects via ribbon cables to an interface device such as a DAQ which acquires the measurements from each of the PCB’s on that jig and sends them via USB to the PC. These measured values from a PCB can be either a voltage, current or digital data that comes from the circuit board’s microprocessor. The information output from the circuit boards as well as data-format differs between the different kinds of PCB’s and thus each kind of PCB requires its own specific jig, interface device and software.

The PCB output is measured and processed at a rate of 1 Hz by a logging software written in LabVIEW which outputs log files that are evaluated manually against a criteria contained in document TAB0629[3] to see if a circuit board has passed or failed the screening. Failed circuit boards are disposed of and passed ones are used in sensor products.

This thesis only deals with making improvements on the PC logging system that produces the log files as well as changing the layout of the log files themselves.

2.1 Thermal and vibration stress equipment
A climate chamber of the brand Vötsch is used to achieve the ESS thermal program. This program consists of temperature cycling and temperature burn-in. The cycling consists of 10 cycles of hot/cold where each hot/cold state lasts for 10 minutes. After 10 cycles the program enters the burn-in phase which is a hot state that lasts for 32 hours. In the climate chamber there is space for 20-30 jigs depending on jig-type. A climate chamber with higher capacity will be used in the future. For image of the climate chamber, see Figure 2.0.

To generate vibration stress for the ESS vibration program, two apparatuses of different sizes can be used. The smaller apparatus has capacity for one jig at a time and the bigger vibration device has the capacity of 3-5 jigs depending on jig type. The bigger vibration device emits some sort of electrical noise to the measurements and can not be used with some of the circuit boards due to this. The ESS program duration for vibration stress is 12 minutes.

2.2 Electronics to be screened
The ESS screening is conducted on the three main types of circuit boards with surface mounted components (SMC) and used in sensor assemblies:

- Presens Sensor Electronics Unit (PSEU)
- Power and communication unit (PCU-MCU)
- PCU Power unit (PCU-PWR)
Redesign of ESS system

Figure 2.2: circuit board parts to be tested. From top to bottom: PSEU, PCU, PCU-PWR.

All types above are used inside sensors but in different combinations depending on sensor products. The PSEU is an ADC that converts the analog measurement of the sensor element to digital values which are then passed on to the PCU-MCU. The PCU-MCU's task is communicating with the outside industrial control system in which the sensor is used. PCU-MCU comes in one version for each kind of communication interface. Some of the PCU-MCU versions requires an external voltage regulator, this external voltage regulator is the PCU-PWR. For image of each circuit board, see figure 2.2.

2.3 Jigs
To provide a data connection between a single interface device and multiple circuit boards, a hardware device known as “jig” is used. The jig consists of several sockets for circuit boards and connections for programming and screening. Connecting a circuit board to it’s specific jig makes it possible to program the circuit board microprocessor and access measurement pins on the circuit board that are used during the ESS. Each jig can connect to 5-10 circuit boards depending on jig type. There are three distinct jig types that correspond to the different circuit board types and several versions of each kind which supports different communication interfaces. For images of the different jigs, see Figure 11.1-11.3 in appendix.

2.4 Interface Devices
Interface devices are used to provide industrial I/O functionality to a computer and to enable the computer to acquire data from various sources as well as communicate with other devices.

In the ESS system the interface devices are used to provide a channel for communication between the circuit boards and the ESS application during the ESS. The circuit boards have different ways of communicating depending on both version and type. Therefore several different kinds of interface devices are required. In total there are five different
interface devices being used. Two of them are used for communicating with jigs requiring both SPI and DIO, and specifically described in Table 2.1.

<table>
<thead>
<tr>
<th>Type of Jig-communication</th>
<th>Interface Device product</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analog voltage and current</td>
<td>NI USB-6210</td>
</tr>
<tr>
<td>CAN-bus</td>
<td>NI USB-8473</td>
</tr>
<tr>
<td>RS 485-bus</td>
<td>Brainbox</td>
</tr>
<tr>
<td>SPI DIO</td>
<td>NI USB-8451 &amp; USB-6008</td>
</tr>
</tbody>
</table>

*Table 2.1: Interface devices and their communication/measurement type.*

Each interface device needs to be connected to the PC via an USB hub since the PC's used have too few USB ports on the motherboard.

### 2.5 Logging software

The software that performs the logging of the circuit board values during ESS is divided on several applications depending on what kind of jigs are logged. Almost all the applications used in the ESS are LabVIEW applications, with the only exception that “AsterixEsslogger.exe” is a command-line application. The applications are run on several PC's to help distribute the testing system load, and each PC runs 6-10 applications. In particular, the applications are the following:

- CAN studio.vi
- PSEU Studio.vi
- RS485 Studio.vi
- Asterix_PWR_PCU_ESS.vi
- Asterix_4-20ma_PCU_ESS.vi
- AsterixEsslogger.exe

The different applications support a different number of Interface devices. The “AsterixEsslogger.exe” supports four interface devices and “PSEU Studio.vi” supports five. All other applications support one connected Interface device. The appearance of a typical application GUI is shown in figure 2.1 and the number of supported interface devices for each software is shown in table 2.2.
Table 2.2: The number of jigs each application can log.

<table>
<thead>
<tr>
<th>Application</th>
<th>Maximum jigs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Asterix ESS logger.exe</td>
<td>4</td>
</tr>
<tr>
<td>PSEU Studio</td>
<td>5</td>
</tr>
<tr>
<td>Asterix PWR PCU</td>
<td>1</td>
</tr>
<tr>
<td>RS485 Studio</td>
<td>1</td>
</tr>
<tr>
<td>CAN-studio</td>
<td>0.5-1</td>
</tr>
<tr>
<td>Asterix 4-20ma</td>
<td>1</td>
</tr>
</tbody>
</table>

Figure 2.3: GUI of the “Asterix_PWR_PCU_ESS.vi” logging application.

The applications used in the ESS process differ depending on interface type but still have the same major functionality of entering settings and logging the data retrieved by the interface applications.
Redesign of ESS system

based on those settings. Below follows a list of the user controls and user indicators present in most applications.

**Controls**

- Select interface device that connects to jig
- Select which circuit boards on jig to log
- Set circuit board name for each position
- Select the type of circuit board
- Select file path for folder to save to
- Enable/disable error log
- Buttons to start/stop acquiring measurements
- Buttons to start/stop logging to files
- Button to stop application

**Indicators**

- LED indicators for indicating errors for each circuit board
- Current values acquired from each circuit board
- Chart presenting last 1000 values of each circuit board's parameters.

2.6 Log files

The software creates log files for each circuit board and possibly an error log file per jig (which is optional). The circuit board log file contains all values that have been logged during a screening session for one circuit board while the error log only contain values outside a pre-set limit but for all circuit boards on the same jig.

Both kinds of log files are saved in tab separated text-format and are examined in Microsoft Excel. The log files contains one row of data for each measurement and the columns consists of time stamp followed by the values of that measurement. The appearance of one row(with header) can be seen in Table 2.1. To provide traceability for the final products, each log file is kept on company servers for the full lifetime of the product. The size of a thermal stress log file is around 250 000 rows and a file size of up to 10MB while the vibration stress log file is less than 1000 rows with a file size of 50kb.

<table>
<thead>
<tr>
<th>Time stamp</th>
<th>Parameter1</th>
<th>Parameter2</th>
</tr>
</thead>
<tbody>
<tr>
<td>2013-04-44 23:23</td>
<td>4.87</td>
<td>1.12</td>
</tr>
</tbody>
</table>

*Table 2.3: Example of structure inside log files*

2.7 Operator work flow to conduct ESS
Redesign of ESS system

Figure 2.4: Activity diagram of the operator workflow for conducting ESS test.

The tasks involved in conducting a ESS on PCB’s consist of keeping track on hardware connections and configuring the ESS applications accordingly. The connection between equipment,circuit boards and PCs are written down to a document called “Kjörningstabell”. This document provides traceability to help the operator keep track of connections when entering settings to the applications. The tasks are presented in further detail and chronological order below.

1. Connect circuit boards to jigs.
2. Connect jig to its matching interface device with ribbon cables.
3. Connect interface device to computer, note which computer.
Redesign of ESS system

Figure 2.5: Measurement and automation explorer window showing connected interface devices.

4. Open the MAX(Measurement and Automation Explorer) application to see which interface device serial number corresponds to which “device number”. A “device number” is automatically assigned to each connected interface device that is recognized by MAX. Figure 2.5 illustrates the detected interface devices inside the MAX application.

5. Open the ESS application that supports the jig to be tested. Select type of circuit board, device number, type of measurement, jig number and circuit board serial numbers.

6. When all applications are properly setup, go through each of them once again and make sure that there are no circuit board measurements outside limits and that all the settings are matching the document “Kjörningstabell”. Then start the logging for each application by pressing the “start”-button.

7. The circuit boards are logged during the full screening duration. The operator keeps track of when the run is completed and then stops the logging and evaluate each log file by plotting it as a graph and examining it for errors as listed in the criteria in the document “TAB0629”. Figure 2.6 provides an example of the appearance of such a graph.

Figure 2.6: A log-file plotted as a graph for evaluation. The first part of the graph is the cycling, then burn-in and in the end the climate chamber is turned off but the application has kept logging.
2.8 Time consumption for ESS

The time required for conducting an ESS can be studied by dividing the process into several parts. As described earlier, an ESS consists of two separate screenings. Each screening requires setup and result evaluation. Setting up 20 jigs for thermal screening takes at least 3 hours and for vibration screening about 2 hours. The time varies depending on whether the operator have to deal with connectivity issues between interface devices, jigs and PC. An overview of the time consumption for different parts of the ESS is illustrated in Figure 2.7.

The thermal screening (cycling and burn-in) needs 72 hours and the vibration screening needs 12 minutes for each jig.

![Figure 2.7: Time consumption for ESS when testing 20 jigs.](image)

The evaluation of a thermal screening at full capacity (20 jigs) requires 8 hours, and the vibration screening 4 hours since the log files are smaller and have simpler error criteria.

3 Method

For this project the following activities are conducted.

- Identify how the ESS is performed together with problem areas and possible improvements.
- Define the specification of requirements for system and applications.
- Implement a prototype application in LabVIEW.
3.1 Study of existing system
To gain understanding of how to implement the project tasks, the existing process of ESS testing was studied, first theoretical by reading the process description[2] and then practically on site together with the operators during actual ESS runs on both thermal and vibration screening.

3.2 Specification of requirements
A new specification of requirements is written based on the preliminary specification, having identified how well the existing ESS process works, what the problem areas are and which improvements can be made.

The specification will be structured in tables where each requirement is numbered and divided into categories. The numbering of the requirements will later be used as a reference when describing the implementation.

3.3 Implementation of prototype application in LabVIEW
The purpose of this prototype application is to provide a base for further implementation. When implementing functionality in LabVIEW, special consideration will be taken to make sure that the implemented technical solutions are dynamic and easily configurable as well as easy to understand. This is done by extensive use of dynamic arrays and sub-VIs which will result in clean LabVIEW codes and good overview inside the source codes.

4 ESS system's specifications and improvements
In this chapter we discuss the work behind the specifications of requirements for the ESS system. The final specifications are given in detail in Chapter 11 Appendices.

4.1 Areas of improvements
Some of the improvements were already identified at Presens and collected in the preliminary design specification named “ESS Design SPCS document” (see appendix 11.3) that was supplied in the beginning of the thesis project.

More improvements were figured out by analyzing the process itself and talking to the operators and developers at Presens. No consideration has been taken to non-application problems such as malfunctioning jigs. The specific problems that were discovered are presented below.

A. Multiple instances of the same program
In the existing ESS system the scalability is achieved by opening up more instances of the logging applications. From a user experience perspective it is difficult to get an overview of the jigs and their settings when up to 20 fullscreen applications are running in parallel. From a capacity point of view it requires more processing power and memory to manage multiple applications compared to a single application because of overhead in the applications.

B. Manual folder creation
Log files are placed in folders for each run to keep them ordered by date and production order number. This is done manually and in the application the operator needs to change the file path to the newly created folder.

**C. Manual evaluation of log-files**

In the ESS system only the numbers of errors are counted, and more complex error criteria required for passing or rejecting a circuit board according to the TAB0629 criteria cannot be managed. Therefore all log-files must be examined manually to see whether a circuit board has passed or failed the screening.

**D. Log files without including thermal state/temperature/vibration data.**

In the existing thermal stress process the temperature state or the temperature is not included in the log file. Since the error limits for thermal ESS depends on thermal state (burn-in or cycling) according to the TAB0692, it is of interest for the evaluation of the log files to know when the transition between burn-in and cycling has occurred. Currently the point of transition is decided by the operator based on the appearance of the log file presented as a graph.

**E. Manual device selection**

Operator needs to make the mapping between interface device serial and interface device number manually and then select interface device number for each jig inside the applications.

### 4.2 Improving the preliminary specification.

The main points from the preliminary specification:

- One application for each kind of interface, controlled by a “control application”.
- Error criteria should be applied to the circuit boards that are measured and presented in real-time.
- Climate chamber should be controlled by the “control application” in order to keep track of current thermal screening state (burn-in/cycling).

In order to improve the quality and detail of the preliminary specification, the current work flow of the ESS process as well as the source code of the ESS applications were studied.

The specification is improved by adding details on requirements already present in the preliminary specification and adding new features.

Since the scope of this project is limited to the software applications of the ESS, the current applications were examined to see which functionality was needed.
4.3 Changes made on the preliminary specification

A few changes have been made on the preliminary specification. They include:

A) Instead of having one application for each jig-type, there will be one application per interface device-type.

B) Instead of a device list to choose from inside a application, make the applications automatically use all connected interface devices of the type that is supported by the application.

C) Instead of showing a graph for each PCB, the application will show one graph per jig and PCB parameter type.

D) Instead of using tabs in the GUI, the settings and view will be presented on the same page.

A. The number of windows per jig-type

In the preliminary specification: “Only one window per type of jig: If several of the same jigs are connected and logged at the same time it should be done in the same window.”

Instead of dividing the system with one application for each kind of jig, it would be possible to divide the system by the interface devices used by the jigs. As can be seen in the table of circuit board-to-device relationship in Appendix 11.5, the solution of having one application for each kind of interface leads to a maximum of 4 interface applications while using one application for each jig would lead to about 10 applications.

B. Selection of devices

In the preliminary specification: “The program needs to have a device list that lets the user choose which device(s) the program is currently logging from, and which measuring-/communication device that belongs to which jig.”.

Previously the operator would go inside an application and choose which interface device number to use. The list to choose from would contain all devices connected to the PC, even the ones not compliant with the application.

In the case of having one large GUI where each jig is presented as its own GUI element, it would save complexity to let each of the GUI elements belong to a distinct interface device.

This implies that all interface devices that corresponds to an application and is connected to the PC is going to be logged. This makes no difference compared to previously since the operator only connects the interface devices that are needed.
C. Visualization of circuit board values in plots.

“Real-time data should be plotted to graphs that display all measured data from each board (one graph per board) real-time.” - Preliminary specification

Instead of showing one graph per circuit board, all circuit boards from the same jig should be in the same graph so it is possible to compare them. If a circuit board has more than one parameter, the parameters should be separated into one graph per parameter. This is because the different parameters might have different range of values and that would make the plots auto scale to the one with the biggest range and thus lose resolution for the parameter with smaller range.

D. GUI appearance, separation between setup and viewing values.

“User friendly, clean look. Can have several tabs in one window, for example one tab for setup, one for graphs, one for real-time measured values and Booleans etc.).” - Preliminary specification

Tabs are not needed to separate the setup and real-time measured values since most of the information in the setup tab is needed to know which circuit board a measured value belongs to. According to the operators it would be enough to see the real-time values of one jig at a time. For the graphs however they would need to be opened up in a new window to get the biggest possible view for analyzing plots.

4.4 New features added to the specification

This section describes new features that have been developed and added to the preliminary design specification. They include:

A) Support for a dynamic number of jigs in each application.

B) The serials of the connected interface device serial are presented in the applications.

C) Possibility to only perform the later part of an aborted/failed screening.

D) Noise reduction to prevent 50hz noise in the measurements.

E) Config files that stores the settings for each kind of jig.

A. Support for a dynamic number of jigs

Allowing for a dynamic number of jigs would greatly enhance the scalability. Currently most of the ESS applications can only log from one jig. And the other two(PSEU and DIO) can log from a maximum of 5 and 4 jigs.

Even if some of the applications can log from multiple jigs it is still a static number of jigs per application. With the static approach one would have to add more code for each extra jig to monitor from. The easiest way would be to simply duplicate the code and thus get twice the code size. The problem with duplicating code to increase capacity is that of redundancy. If changes would need to be made in the program, the changes would need to be made on multiple places. It also opens up for inconsistencies where one jig will run on different code then the others.
Since the number of jigs in a screening session varies, there would often be unused “slots” in the static software solution. The variables and GUI components would still have to be there even if not used.

To combat these two problems it would be possible to have a dynamical approach to the number of jigs that can be supported.

**B. Keeping track of interface device connections**
The operator is required to manually keep track of the automatically assigned device numbers of the connected interface devices. This is an extra step that is unnecessary since it is possible to make the interface device serial (unique identifier printed on each interface device) visible inside LabVIEW applications.

**C. Jump to position in thermal test**
In some cases a thermal screening needs to be restarted due to technical errors. To avoid having to restart a test from the beginning it should be possible to continue a screening from when the error occurred. This brings further implications that is mentioned in the specification under the section “Jump to position” in the final specification.

**D. Noise reduced measurements**
During the study of the source code of the “Asterix 4-20ma.vi” application, it was discovered that a mean value of the acquired values was used to achieve noise filtering. After talking to one of the engineers it was told that this is to prevent 50hz noise.

**E. Config-files with unique parameters for each kind of circuit board**
Several parameters in the ESS applications are circuit board specific and may need to be changed over time. Config files containing these settings should be used to enable modifying the settings for a circuit board without having to change the LabVIEW code. Each “jig-element” in the dynamic GUI should have a list of possible config files to choose from.

**5 Interface application**
This section gives a few suggestions for implementing the specification to create applications that fulfill the requirements. To present program-flow, a UML Activity diagram is used and in the case of interface applications a UML Class diagram is used to show how to divide data structures into several clusters to achieve support for dynamic number of jigs.
5.1 Software activity diagram

![Diagram](image)

Figure 5.8: UML software activity diagram for an Interface Application.

The software activity diagram seen in figure 5.8 shows how the interface application is supposed to operate.
5.2 Software class diagram

![Diagram of UML Class Diagram]

**Figure 5.9: UML Class diagram of “classes” that should be implemented.**

The interface devices and jigs are to be treated as “objects”. To implement this property, a series of clusters contained inside arrays are used. The arrays of clusters required to implement an interface application are shown in Figure 5.6. For the “DAQ interface application” the cardinality between the jig and interface device array-objects are one-to-one. In that case the first elements of each array belongs together. In a Interface application where several jigs connect to the same device there needs to be a cardinality of one-to-many.

Interface device specific arrays(one array-element per interface device):

- Device settings (Cluster)
- Device viewer (Cluster)

Jig specific arrays(one array-element per jig):

- Main input (Cluster)
- Error array (Cluster)
- Read measurements (Array of doubles)
- Log file (Array of refnums)
- Error log file (Array of refnums)

For detailed description of each array of clusters, see section “5.6 Clusters”.

5.3 Software implementations

The implementation of the prototype shows important design ideas and is made up by several sub-VIs that are re-usable between the different kinds of interface applications with minor adjustments.

The prototype application implements all the sub-VI’s mentioned in chapter 6.4 and has the following functionality:
Redesign of ESS system

- Detect and initialize interface devices according to config files
- Acquire measurements from interface devices
- Save measured values to log files and error log files
- Evaluate errors against TAB0629 criteria and indicate errors.
- View measured values from one jig, presented in a table.

5.3.1 Detection of connected devices

Fulfills specification 2.2

After investigating LabVIEWs functionality it was discovered that it is possible to detect which interface devices that are currently connected to the computer running the application. This information is used to alter the GUI dynamically to the number of devices connected.

The Sub-VI that handles this functionality is named “tDetectDaq.vi” and can be seen in figure 5.10. It detects all National Instruments DAQ-devices that are connected to the computer. This includes:

- NI USB-6210
- NI USB-CAN
- NI USB-DIO

An improvement would be if it also detected the “Brainbox” interface device.

Reshaping the GUI during runtime

Fulfills specification 2.1 + 2.2

A solution for altering the GUI dynamically during runtime is achieved by storing GUI components inside an array of clusters. To resize the array in the GUI, a property node named “Number of Columns” is used. The limitations of this approach is that the GUI-elements is limited to cascade to the right.
One of the properties of a cluster in LabVIEW is that all elements inside a cluster must be either indicator or control. This makes it more difficult to have both input and output in the same cluster. To be able to get past this problem one needs to alter the whole cluster by first unbundling the cluster, then updating one of its controls and then bundling it back to the cluster. In this way a control can be used as an indicator. In the example in figure 5.3 above, the indicator LEDs “Full” and “Point” are in a control-cluster but they are still used as indicators.

Another approach is to separate indicators and controls when possible by storing them in different arrays. This can be seen in Figure 5.1 above where Device and DevSerialNum is stored in a separate array of clusters then the cluster containing “Arbetsorder”, “Jigbeteckning” etc.

**GUI-elements for each jig**

Fulfills specification 3.1
Redesign of ESS system

Example of controls are supplied in illustration a[]. The circuit board-type control is a Menu ring enum which is loaded with items from the config-file folder.

**GUI-elements error-indicators**

**Fullfills specification 3.2**

Example of error-indicators in illustration a[] and for overview of full GUI see illustration s[]. Since the error-indicators are put inside a control cluster the state of the indicators can only be changed by unbundling and rebundling the cluster.

**GUI-elements real-time value table**

**Fullfills specification 3.3**
Figure 5.4: Table for visualizing real-time data

A table with sliders is used to enable unlimited number of parameters and circuit boards in the visualization of the values. A menu ring containing all the jigs is used to select which jig to view. Parameter headers are loaded from the config-file.

Display of error-limits

Error limits are stored in the config files and loaded into an array of clusters from which it is used by the program to determine error limits and interface device settings. This array of clusters should be visible and thus allowing the operator to see which error limits are loaded.

Load Config files

Fulfills specification 2.3 + 7.1 + 7.2
Redesign of ESS system

INI-files is used for storing config values since it is simple and covers the needs of this application. INI files are easy to use since they are simple text files and edited as such. INI-files are supported by LabVIEW and its easy to select which properties to read from the INI-file.

![DeviceSettings](image)

**Figure 5.5: Appearance of a config ini-file inside a text editor**

The data from the config file is saved to an array of clusters (seen in illustration []) with the same element order as the device cluster and read value cluster. In this way the config files are only read once during the initialization to reduce hard drive usage.

**Reduction of 50 hz noise**

For analog DAQ measurements a 50hz power line noise might be present, disturbing the measurements. To combat this a sample rate of 100hz is chosen and 50 samples are taken. Since the noise is a sine wave, the average level is 0. The sample rate should be chosen as an integer multiple of the noise frequency to make sure the samples will be on both the negative and positive level of the sine wave. The mean error of the samples with the same level of error but with different sign will be 0. The number of samples should be chosen such that sample s/sample rate = time between
measurements. In this case the time between measurements is specified as 0.5 sec which results in 50 samples for a 100 Hz sample rate.

**Graphical user interface**

![Graphical User Interface Image](image)

Figure 5.7: The full GUI of the Interface Application for the Analog Interface Devices

The graphical user interface have a tab-free design which gives a full overview of the error status of all circuit boards. There is no limit in the GUI for how many interface devices that can be shown, the array of devices extends to the right indefinitely. If there are enough interface devices to cover the screen size, the horizontal scroll bar of the LabVIEW window is used to access the full GUI.
5.3.2  DAQ init

Creates tasks for selected Interface devices. Sets samplerate and type of sampling in the case of Analog DAQ. Output is a array of DAQmx tasks which is used by DaqRead.

5.3.3  Read Config

Loads settings from ini-file and puts it in array of cluster.

5.3.4  File maker

Creates folders and files according to Array3. One folder for each Arbetsorder and one file with the name [körningsnumer].txt. It also creates the header of each file.

Output from the VI is an array of filereferences.

5.3.5  DAQ Read
5.3.6 FileWrite

Saves measured values, timestamps and temperature for each circuit board. Each measurement for a circuit board is represented as a row in the log-file. The files that are saved to has been created earlier by “filemaker.vi”.

5.3.7 Error detect

If any of the circuit boards have a measured value that is outside the allowed ESS-limits then this should be alerted to the operator in real-time. The limits differ depending on which phase the test is in, either burn in or cycling.

5.3.8 Error criteria

The error criteria Sub-VI evaluates the accumulated errors according to TAB0692 and indicates results in the Main Input cluster.
5.4 Clusters

Clusters have the same function as structs in C, they are a data type that contains several primitive data types. In this project they are used to create objects for handling a dynamic number of jigs and interface devices.

The clusters that have been implemented are:

- Main input
- Device Settings
- Device Viewer
- Error Array

5.4.1 Main Input

The Main input cluster is where the operator inputs jig and circuit board information and selects which config file to load for the interface device. The indication of errors are placed within this cluster. This achieves a structured GUI where each row in the main cluster element belongs to one circuit board.
5.4.2 Device Viewer

Device Viewer holds information about the interface device and is only used to show which interface device a “Main input” element belongs to.

5.4.3 Error array

The error array holds all the errors for all the circuit boards from one jig. “Pointerror” and “partialAll”-arrays are counters for error duration (consecutive errors) of the circuit boards. “PartialError” and “FullError”-arrays contains number of errors and when they occurred. This information is required to evaluate the errors of a circuit board against the error criteria.
5.4.4  Device settings

Device settings holds all the config values required to configure an interface device. It also holds the jig settings with error limits and parameter names.
6 Control Application

The application for controlling the interface applications and the climate chamber during thermal screening has been sketched as a LabVIEW GUI. A program flow is made as a UML activity diagram to guide an actual implementation. The connection between the different applications and the climate chamber can be seen in figure 6.1.

![Figure 6.1: Connections between applications and climate chamber.]

6.1 Control Application GUI

![Figure 6.2: Sketch of Control Application GUI]
Seen above in figure 6.2 is a GUI for the control application.

The reasons for having a central control application:

- Detect disturbances in any of the circuit boards that requires the operators attention.
- Central control for start/stop of thermal screening (vibration is run as stand-alone with interface applications directly).
- Control of the climate chamber for performing thermal screening.

**Interface applications**
From this GUI the operator will be able to start the interface applications as well as monitor the number of jigs that are successfully connected and the number circuit boards that has failed. If a circuit board has failed inside an interface application, the “OK Led” that corresponds to that interface application will go dark. This is to alert the operator that a failure has occurred. After the operator has investigated the error, he/she can clear the “OK Leds” for all applications by pressing the “Clear button”. This will give the operator a quick way of keeping track of changes in terms of circuit boards that have failed.

**Test control**
The controls for the full test are at the bottom in the GUI and consists of three buttons and one listbox. The listbox is used together with the jump-function to select which part of the screening that the operator wishes to start from.

**Test details**
The “test details” of the GUI contains test specific information.
6.2 Software activity diagram

The program flow of the Control application is seen in figure 6.3 and consists of two loops. One of them is pre-screening when the operator is setting up the system and starting it and the other loop is during the actual screening.

**Pre-screening**

For the pre-test, the number of failed circuit boards is acquired via global variables from each of the interface applications and presented in GUI. It is possible to start interface applications from inside the GUI of the control application and see if the interface applications are initialized and ready to log. The run-state is entered from the pre-screening loop by either “Start logging” button or the “Jump to” button.
Run-state
Run-state provides the functionality of running the ESS program for the climate chamber at the same time as presenting information from the initialized interface applications.

7 Performance test
A performance test with the aim of evaluating the CPU efficiency of the prototype application compared to the application it will replace was conducted. The test consisted of first running 10 applications of the “Asterix_4-20ma.lvproj” and then restarting LabVIEW and running one instance of the new prototype application handling the same number of interface devices. The applications were logging from simulated NI USB-6210 interface devices while the average CPU load was monitored with Windows Resource Monitor. Screenshots from the test can be seen in appendix 1[].

The two tests resulted in a CPU usage of 28% in average for the current multiple application solution and a CPU usage of 10% in average for the new prototype application that handles all jigs with the same interface device from the same application.

The difference is believed to be related to the overhead required to run multiple applications, and the difference in usage of CPU resources will increase as the number of jigs increase.

8 Results and discussion
8.1 Reducing operator workload
As stated earlier in the rapport, the setup of a thermal screening takes between 3-5 hours and the evaluation of the results takes 8 hours.

The obvious way of reducing the workload for the operator is having automatic result evaluation. This feature is partially implemented by the two SubVI’s “Errordetect.vi”, “ErrorCriteria.vi” and “ConfigIni.vi”. Communication with the climate chamber is required for setting the right error-limits and stop the error-detection once the run is complete. This communication is beyond the scope of the thesis and has not been implemented.

The automatic error detection feature would in best case save the 8 hours of work required for examining log files. Best case is when all circuit boards of a screening has passed the TAB0629 criteria.

To reduce the amount of work required for setting up a test the solutions below have been provided.

- One interface application per interface device type
- One control application that controls logging of the interface applications
- Tab-free GUI for entering settings for all jigs
- Automatic file handling
- Tab-free overview for all circuit boards error statuses
8.2 Improvement of capacity
With the new design, a maximum of 5 applications are used regardless of the number of jigs to screen. Previously, the number of applications were one application per jig for most jigs. With the new solution a complete test could be run from a single PC because of the increased CPU efficiency of the application and fewer applications running in parallel.

With the new implementation of the interface application running with 10 simulated devices the CPU load was only one third of the current program ("Asterix_4-20ma.vi") running in 10 instances. This is important as the number of jigs to test will increase and the previous system would be limited by CPU power at an earlier point in time then it is with the new solution.

8.3 Source code
The project has resulted in a working prototype of the interface application for the analog NI-6810 and it is capable of:

- Detecting interface devices
- Load and use settings from INI-files
- Detect errors and compare against TAB0629 criteria.
- Log measured values to both circuit board-log files and error-log files.

Figure 8.3: LabVIEW sourcecode for a part of the interface application. This part includes reading of values from circuit boards, detecting and evaluating errors, logging to files and presenting the values.
For the interface application prototype to comply with the specification the following additional features must be implemented:

- Communication with control application for remote start/stop
- Jump to time period (controlled by control application)
- Graph view of circuit boards
- Generate report ("Kjörningstabell") at the end of a finished screening

8.4 Discussion

The new approach of having a dynamic number of interface devices connected to the same application helps to improve both the capacity and reduce the operator workload.

However, the current solution is still split on multiple applications to make the implementation easier. A more elegant but far more complex solution would be to have a single application, preferably written in a procedural language that have better support for building complex GUI’s.

This is however not to be considered cost-efficient since the workload for the operator is already at a minimal with the new solution and the capacity would not be affected.

9 Suggestions for future work

This thesis has covered a part of the design of the new ESS applications and the full implementation of the specification requires further development as listed in the specification document.

A suitable working order for finishing the system would be:

1. Implementing the analog interface application by continue working on the prototype
2. Implement the control application
3. Implement the rest of the interface applications, based on the analog interface application.

This work order is chosen because it allows an application to be completely finished before implementing the next application. This helps combat the need for redundant changes over multiple unfinished applications.

This thesis was limited to LabVIEW and the current hardware. A larger study could be done to investigate other options for ESS system implementation.

10 Reference


Appendix

11.1 Specifications of software requirements
Details the requirements for the software of the ESS system.

11.1.1 General requirements
The system should be able to ESS-test all of the current jigs, both for vibration and temperature.

<table>
<thead>
<tr>
<th>Num</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Four DAQ/Interface applications, one for each kind of interface.</td>
</tr>
<tr>
<td>1.2</td>
<td>Studio application that controls and indicated status of the interface applications with global variables. It also controls the current and new climate chamber.</td>
</tr>
<tr>
<td>1.3</td>
<td>The interface applications should be possible to use standalone for vibration test.</td>
</tr>
<tr>
<td>1.4</td>
<td>Will use the same kind of Interface Devices and jigs as currently but should be easily adapted to new CAN circuit board-jig with increased capacity (10 single circuit boards/interface device). The current Interface devices are: RS-485 Brainbox, NI-6210, NI-8472, NI-8451 and USB-6008</td>
</tr>
<tr>
<td>1.5</td>
<td>PC-connected to climate chamber for control and measurement.</td>
</tr>
<tr>
<td>1.6</td>
<td>The DAQ applications should be cpu and hdd efficient. Possible to use over multiple PC's</td>
</tr>
</tbody>
</table>

Jump to a time-period in test
It should be possible to restart a test manually and decide where in the climate chamber program it should start. For vibration screening there is no such need since the screening duration is less than 15 minutes.

<table>
<thead>
<tr>
<th>Num</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1b.1</td>
<td>Possibility to go to a user selectable timeperiod in test. Both back and forth from current period in time.</td>
</tr>
</tbody>
</table>
1b.2 Continues to write in the same log files, no data should be removed.

1b.3 When test is complete, log file is automatically stitched together and saved in a different file.

1b.4 Warning message is shown in the end of a restarted run, stating that this test has been restarted and that files need to be checked manually.

1b.5 Error tracking should still work after a "goto position" has been executed.

1b.6 Control in GUI should be "start logging and goto position".

### 11.1.2 Interface Application requirements

#### Device handling

The interface devices are detected and initialized with the correct jig-settings.

<table>
<thead>
<tr>
<th>Num</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Support dynamic number of interface devices.</td>
</tr>
<tr>
<td>2.2</td>
<td>Detects interface devices automatically and adds controls and indicators for each jig in the GUI dynamically.</td>
</tr>
<tr>
<td>2.3</td>
<td>Settings for each interface device is loaded from an ini-file for jig-specific settings. Logistic information and information that is non auto detectable is entered manually.</td>
</tr>
<tr>
<td>2.4</td>
<td>For analog data acquisition the sample rate and duration is to be made in such a way that 50hz noise is reduced.</td>
</tr>
<tr>
<td>2.5</td>
<td>Data acquisition of G-value for vibration or temperature value for thermal.</td>
</tr>
</tbody>
</table>

#### General user Indicators and controls

Errors and measurement values from each jig are presented in the GUI in real time.

<table>
<thead>
<tr>
<th>Num</th>
<th>Description</th>
</tr>
</thead>
</table>
| 3.1 | Controls for each jig to enter:  
  • Production order number  
  • Jig number  
  • Manufacturer for each circuit board  
  • Serial for each circuit board  
  • Config file to use |
| 3.2 | Two error-indicators for each circuit board, one for temporarily out of limits and one for circuit board fail according to TAB062 criteria. |
| 3.3 | Real time view of measured values from one selected interface device at a time. Presented in table with headers for each parameter. |
| 3.4 | Show limits that are used for TAB062 criteria. |
3.5 Graph with plots for each parameter from each circuit board from one jig at a time. See “Graph Visualizing” below.

3.6 Control for selecting which kind of screening is performed (temperature or vibration).

**Graph Visualizing**

It should be possible to plot graphs of the measured data to evaluate circuit boards.

<table>
<thead>
<tr>
<th>Num</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1</td>
<td>Opens a separate window with one graph / parameter showing history of measured values.</td>
</tr>
<tr>
<td>4.2</td>
<td>Do not need to show all data points, only values that have passed in or out of a limit and values that have changed more then a set value.</td>
</tr>
<tr>
<td>4.3</td>
<td>Do not need to show in real time. Loaded window is not updated.</td>
</tr>
<tr>
<td>4.4</td>
<td>Each parameter from a circuit board is put together with the same parameter from the other circuit boards to make it easy to compare.</td>
</tr>
<tr>
<td>4.5</td>
<td>It should be possible to select which plot lines (circuit board parameters) that should be included in graph.</td>
</tr>
<tr>
<td>4.6</td>
<td>Tool palette for zooming in and out on graphs.</td>
</tr>
<tr>
<td>4.7</td>
<td>Plot legends named after parameter and circuit board-name.</td>
</tr>
<tr>
<td>4.8</td>
<td>Show limits as lines in graphs.</td>
</tr>
</tbody>
</table>

**Circuit board log files**

Measured data is saved to log files for traceability and error check.

<table>
<thead>
<tr>
<th>Num</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.1</td>
<td>Save one log file per circuit board.</td>
</tr>
<tr>
<td>5.2</td>
<td>Saved in text-format so it is possible to open in excel.</td>
</tr>
</tbody>
</table>
| 5.3 | Header information in each log file:  
  - Production order number  
  - Jig type (which ini-file used)  
  - Jig number (written on jig)  
  - Manufacturer batch  
  - Date and time  
  - Device serial  
  - Circuit board serial  
  - Parameter headings  
  - Revision number of ini-file  
  - Error limits |
| 5.4 | Log files should never be removed or overwritten automatically. |
| 5.5 | Log filename and folder to /current/<Arbetsorder>_<jignumber>_<circuit board
Serial>.txt. When a logging successfully has completed, files are moved to folder /<date>/

5.6 Log files copied to new location after screening is complete.

5.7 Rows for each measurement, containing:
- Relative time
- Cycling period
- Temperature in climate chamber or G-value from vibrator.
- Each parameter value.

**Log file error log**

Measurements that are outside limits are saved in an error log-file.

<table>
<thead>
<tr>
<th>Num</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.1</td>
<td>Save all errors from one jig to the same error log file.</td>
</tr>
<tr>
<td>6.2</td>
<td>Save in text format so its possible to open in excel.</td>
</tr>
<tr>
<td>6.3</td>
<td>Header information in each log file:</td>
</tr>
<tr>
<td></td>
<td>- Production order for jig</td>
</tr>
<tr>
<td></td>
<td>- Timestamp</td>
</tr>
<tr>
<td></td>
<td>- Parameter headers</td>
</tr>
<tr>
<td></td>
<td>- Device serial</td>
</tr>
<tr>
<td>6.4</td>
<td>Log files should never be removed or overwritten automatically.</td>
</tr>
<tr>
<td>6.5</td>
<td>Log filename and folder during a run is saved to /current/&lt;Arbetsorder&gt;_&lt;jignumber&gt;_Errorlog.txt. When a logging successfully has completed, files are moved to folder /&lt;date&gt;/</td>
</tr>
<tr>
<td>6.6</td>
<td>Rows for each circuit board outside limits in jig, containing:</td>
</tr>
<tr>
<td></td>
<td>- Card serial</td>
</tr>
<tr>
<td></td>
<td>- Relative time</td>
</tr>
<tr>
<td></td>
<td>- Temperature from chamber or G-value from vibrator.</td>
</tr>
<tr>
<td></td>
<td>- Each parameters</td>
</tr>
</tbody>
</table>

**Config files**

The settings for the different kinds of circuit boards is loaded from config files.

<table>
<thead>
<tr>
<th>Num</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.1</td>
<td>Config files in ini-format. One for each kind of card type.</td>
</tr>
<tr>
<td>7.2</td>
<td>Keys inside ini-file:</td>
</tr>
<tr>
<td></td>
<td>- Revisionnumber of file</td>
</tr>
<tr>
<td></td>
<td>- Number of circuit boards on jig</td>
</tr>
<tr>
<td></td>
<td>- Max and Min limits for Cycling</td>
</tr>
<tr>
<td></td>
<td>- Max and Min limits for Burn-in</td>
</tr>
<tr>
<td></td>
<td>- Headings for each parameter</td>
</tr>
</tbody>
</table>
Redesign of ESS system

<table>
<thead>
<tr>
<th></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• Channel settings for the interface device.</td>
</tr>
<tr>
<td></td>
<td>• Allowed number of point errors before “full error” occurs.</td>
</tr>
<tr>
<td></td>
<td>• Allowed duration of a error before “full error” has occurred.</td>
</tr>
</tbody>
</table>

“Kjörningstabell” document
When finishing a screening the interface applications should generate report files detailing the results of the screening.

<table>
<thead>
<tr>
<th>Num</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7a.1</td>
<td>For each circuit board show:</td>
</tr>
<tr>
<td></td>
<td>• Device serial it belongs to</td>
</tr>
<tr>
<td></td>
<td>• Card serial</td>
</tr>
<tr>
<td></td>
<td>• Production order number</td>
</tr>
<tr>
<td></td>
<td>• Passed/not passed and reason.</td>
</tr>
</tbody>
</table>

Circuit board fail detection
Measured values of each circuit board are monitored to see if they are approved or not, and if they are not, this is indicated.

<table>
<thead>
<tr>
<th>Num</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>8.1</td>
<td>Each parameter of a circuit board is subjected to the limits that are set in the corresponding config-files. Note that the limits depend on the current run-state of the screening (burn-in och cycling).</td>
</tr>
<tr>
<td>8.2</td>
<td>A indicator for each circuit board shows if the latest measurement is outside limits.</td>
</tr>
<tr>
<td>8.3</td>
<td>A indicator for each circuit board shows if the sum of all measurements comply with the TAB062 criteria or if the circuit board has failed.</td>
</tr>
<tr>
<td>8.4</td>
<td>The error count does not differentiate between which parameters of a circuit board that has failed.</td>
</tr>
</tbody>
</table>

11.1.3 Control application

Climate chamber control
The climate chambers temperature is run by the control application in such a way that it performs the cycling and burn-in temperatures in the correct intervals.

<table>
<thead>
<tr>
<th>Num</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>9.1</td>
<td>Set and read temperatures to achieve the same ESS program that is currently used (cycling and burn-in).</td>
</tr>
</tbody>
</table>
Have variables to keep track of how far the test has progressed. This is to enable rewind feature and keep track of when the thermal screening transitions between cycling and burn-in.

Possibility to go to a desired time period in the ESS program.

**Interface application communication**

The control application controls the interface applications such that all the logging during thermal screening can be controlled from the same application.

<table>
<thead>
<tr>
<th>Num</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.1</td>
<td>Send information to interface applications:</td>
</tr>
<tr>
<td></td>
<td>• Period for thermal cycling and burn-in.</td>
</tr>
<tr>
<td></td>
<td>• Time elapsed since start</td>
</tr>
<tr>
<td></td>
<td>• Climate chamber temperature</td>
</tr>
<tr>
<td>10.2</td>
<td>Send commands to Interface applications:</td>
</tr>
<tr>
<td></td>
<td>• Start logging</td>
</tr>
<tr>
<td></td>
<td>• Stop logging</td>
</tr>
<tr>
<td></td>
<td>• Go to position in ESS program (time period)</td>
</tr>
<tr>
<td>10.3</td>
<td>Receive information from Interface applications:</td>
</tr>
<tr>
<td></td>
<td>• Jig error status</td>
</tr>
</tbody>
</table>

**Controls and Indicators**

In the Control application the GUI should provide functionality necessary to control the interface applications and show the current state of the test-run.

<table>
<thead>
<tr>
<th>Num</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>11.1</td>
<td>Show error status from each jig for each interface application. If one circuit board in a jig is bad, this is indicated.</td>
</tr>
<tr>
<td>11.2</td>
<td>Show graph for climate chamber temperature over time.</td>
</tr>
<tr>
<td>11.3</td>
<td>Controls for:</td>
</tr>
<tr>
<td></td>
<td>• Start logging</td>
</tr>
<tr>
<td></td>
<td>• Stop logging</td>
</tr>
<tr>
<td></td>
<td>• Goto position to [period]</td>
</tr>
</tbody>
</table>
Figure 11.1: Jig for eight PCU-PWR circuit board's
Figure 11.2: Jig with capacity for 10 PSEU circuit boards
11.3 Preliminary design specification
### Preliminary DSPC (Design Specification)

<table>
<thead>
<tr>
<th>Logging program(s) for ESS</th>
<th>Project Leader</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Designer</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Ports:**
- N/A

**DSPC written by:** Mets P. Jonassen  
**Date:** 09.09.13

---

**Known design specifications:**

- Programming language: Labview (latest available version).
- Shall be able to log data from each of the different ESS-jigs currently used.
- Logged data shall be saved to HDD, preferably as text files (one text file per board).
- All logged data should be monitored real-time by the logging program. The program shall use predetermined limits (LAB0629) to decide if each monitored board is approved or not and display this to the user during the ESS run.
  - For example, if a board fails the test (measured values are outside their limits) the program should show this to the user real-time. Could for example have a Boolean for each board that is true if the measured values have been within limits the whole time, and false if the values are (or have been) outside the limits.
  - Real-time data should be plotted to graphs that display all measured data from each board (one graph per board) real-time.
- Should have a main window with global variables (e.g. Presens Studio), that can open up the different programs used for each type of jig. The main window shall also control the climate chamber used for ESS: Controls temperature and hold times, i.e. If the climate chamber is in burn-in mode or cycling mode.
  - Since there are different limits for burn-in and cycling, the program should use a global variable to determine whether the monitored boards are in the cycling or burn-in phase of the ESS run and adjust the used limits for each board in relation to this.
- Only one window per type of jig: if several of the same jigs are connected and logged at the same time it should be done in the same window (see PSEU logging program).
  - The program needs to have a device list that lets the user choose which device(s) the program is currently logging from, and which measuring/communication device that belongs to which jig.
- Shall also output a file that logs all ESS results (kjøringstabell) with information about production order, which units in PO that are...
Redesign of ESS system

Figure 11.8: Preliminary design specification 2/2

11.4 Performance test results

![Resource Monitor]

Figure 11.10 Performance test result, 10 applications of Astrix_4-20ma.lvproj logging one interface device.
Figure 11.9 Performance test result, tMain.vi (new Interface Application Analog in) logging 10 interface devices.

### 11.5 Table of circuit board setup configuration

<table>
<thead>
<tr>
<th>circuit board</th>
<th>circuit board type</th>
<th>circuit boards/ jig</th>
<th>DAQ</th>
<th>LV Application</th>
<th>Parameters / circuit board</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSEU_5V</td>
<td>PSEU</td>
<td></td>
<td>NI-8451 (SPI)</td>
<td>PSEU Studio</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>NI USB-6008</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PSEU_3V</td>
<td>PSEU</td>
<td></td>
<td>NI-8451 (SPI)</td>
<td>PSEU Studio</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>NI USB-6008</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PCU pT/TT CAN</td>
<td>PCU</td>
<td>5</td>
<td>0,5xNI-8472 (1 CAN-kanal per box)</td>
<td>CAN Studio</td>
<td>6</td>
</tr>
<tr>
<td>PCU pT/TT CAN SIIS</td>
<td>PCU</td>
<td>5</td>
<td>0,5xNI-8472</td>
<td>CAN Studio</td>
<td>6</td>
</tr>
<tr>
<td>Description</td>
<td>Interface</td>
<td>Count</td>
<td>Model</td>
<td>Notes</td>
<td></td>
</tr>
<tr>
<td>-----------------------------------------------------------------------------</td>
<td>-----------</td>
<td>-------</td>
<td>-------</td>
<td>----------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>PCU-MCU Dual pT CAN (2 single CAN på ett circuit board)</td>
<td>PCU-MCU</td>
<td>5</td>
<td>1xNI-8472 (CAN)</td>
<td>CAN Studio</td>
<td></td>
</tr>
<tr>
<td>PCU-MCU Dual pT/TT CAN (2 single CAN på ett circuit board)</td>
<td>PCU-MCU</td>
<td>5</td>
<td>1xNI-8472 (CAN)</td>
<td>CAN Studio</td>
<td></td>
</tr>
<tr>
<td>PCU-MCU Dual pT/TT CAN SIIS (2 single CAN på ett circuit board)</td>
<td>PCU-MCU</td>
<td>5</td>
<td>1xNI-8472 (CAN)</td>
<td>CAN Studio</td>
<td></td>
</tr>
<tr>
<td>PCU pT/TT RS485</td>
<td>PCU</td>
<td>0,5</td>
<td>BrainBox</td>
<td>RS 485 studio</td>
<td></td>
</tr>
<tr>
<td>PCU-MCU Dual pT/TT RS-485</td>
<td>PCU-MCU</td>
<td>1</td>
<td>Brainbox</td>
<td>RS 485 studio</td>
<td></td>
</tr>
<tr>
<td>PCU-MCU Dual pT/TT RS-485 (no term R)</td>
<td>PCU-MCU</td>
<td>1</td>
<td>Brainbox</td>
<td>RS 485 studio</td>
<td></td>
</tr>
<tr>
<td>PCU pT/TT 4-20ma</td>
<td>PCU</td>
<td></td>
<td>NI-6210 AIN</td>
<td>Asterix PCU 4-20ma</td>
<td></td>
</tr>
<tr>
<td>PCU dual pT 4-20ma</td>
<td>PCU</td>
<td></td>
<td>NI-6210 AIN</td>
<td>Asterix PCU 4-20ma</td>
<td></td>
</tr>
<tr>
<td>PCU-p dual pT/TT 4-20ma</td>
<td>PCU-p</td>
<td></td>
<td>NI-6210 AIN</td>
<td>Asterix PCU 4-20ma</td>
<td></td>
</tr>
<tr>
<td>PCU-t dual pT/TT 4-20ma</td>
<td>PCU-t</td>
<td></td>
<td>NI-6210 AIN</td>
<td>Asterix PCU 4-20ma [sensortyp: PCU-T]</td>
<td></td>
</tr>
<tr>
<td>PCU-PWR Dual pT/TT CAN</td>
<td>PCU-PWR</td>
<td>8</td>
<td>NI-6210 AIN</td>
<td>Asterix_CAN_SIIS_PWR_PCU_ESS_v0.2_RSE_DevXX</td>
<td></td>
</tr>
<tr>
<td>PCU-PWR Dual pT/TT RS-485</td>
<td>PCU-PWR</td>
<td>8</td>
<td>NI-6210 AIN</td>
<td>Asterix_DUAL RS-485_POWE_ESSRS</td>
<td></td>
</tr>
</tbody>
</table>