Automated production test of BLDC motors in LabVIEW

Anton Söderblom
Abstract

Automated production test of BLDC motors in LabVIEW

Anton Söderblom

The goal determined to be achieved for this project was to construct an assembly of hardware and software to automatically test BLDC motors at Allied Motion AB. The main purpose of this is to get away from the current manually controlled test equipment which is testing the motors. The parts included in the assembly more precisely is a sensor which decides the direction, position and the velocity of the shaft belonging to the motor. A Data Acquisition hardware which is used to converting from analogue to digital signal and vice versa. A computer and also a fixture which keeping the motor at place during the test. The fixture also keeps all the hardware on place. The computer is used to execute the software code developed which decides how the motor should act. The sensor is connected such that it’s sensing the rotation from the shaft belonging to the motor. The assembly described is supposed to achieve pre-determined tests. E.g rotation, direction, position and errors created by the steering electronics belonging to the motor.

The test equipment is operating as wanted and have fulfilled all the desired conditions that was specified.

Handedare: Thomas Garpebo
Anmargivare: Uwe Zimmerman
Examinator: Tomas Nyberg
ISBN LUTH-INGUT8-EX- F-201/18-SF
Populärvetenskaplig sammanfattning av projektet

Detta projekt togs fram i syfte att konstruera en uppställning som ska testa in och utgångarna för signalerna tillhörande BLDC-motorer automatiskt. Detta test är idag manuellt utfört av operatörer och skall med den nya testutrustningen då kunna utföras automatiskt. Det kompletta systemet består i synnerlighet av en dator, givare och någon motor. Datorn exekverar all kod som är programmerad i LabVIEW genom en Data Aquisition hårdvara och motorn tar in all info från koden och behandlar infon. Det motorn utför, tolkar sedan sensorn som avgör riktningen, positionen och hastigheten.

Koden är uppbyggt på en objekt orienterad programmeringsstruktur. Vilket innebär att metoder skapas i olika klasser och dessa kommer sedan att utnyttjas i en slutgiltig kod för att i så stor utsträckning kunna göra koden lättare att konfigurera på olika nivåer. Det testfall som skapats med koden kommer att testa det olika egenskaperna position, riktning och hastighet. Det finns även metoder inbyggda för att kunna testa strömmarna in i motorn samt fel koder som kan uppstå i motorn genom avläsning av servo driven.
Table of contents

Abstract a

Populärvetenskaplig sammanfattning av projektet b

1 Introduction 1
   1.1 Background 1
   1.2 Project description 1
   1.3 Theory 2
      1.3.1 PWM (Pulse width modulation) 2
      1.3.2 Rotational Sensor (Digital shaft encoder) 2
      1.3.3 Hexadecimal form into byte and vice versa 3
      1.3.4 CAN (Controller Area Network) 3
      1.3.5 Service Data Object (SDO) 3
      1.3.6 Relay 4
      1.3.7 Fuse 4
      1.3.8 DAQ 5
      1.3.9 DAQ modules 5
      1.3.10 Programmable voltage supply 6
      1.3.11 D/A and A/D converter 6
      1.3.12 LabVIEW 7

2 Method & Measurements 8
   2.1 Layout of the system as a whole 8
      2.1.1 The system as a whole 8
      2.1.2 Schematic of the whole system 9
         2.1.2.1 PST Power and Enable 9
         2.1.2.2 PST Analog inputs and Lord-brake outputs 10
         2.1.2.3 PST Reference switch 10
         2.1.2.4 Manually steering box 10
         2.1.2.5 CAN communication 10
         2.1.2.6 cDAQ modules and rotational sensor 11
         2.1.2.7 Power supply 11
   2.2 Programming 12
      2.2.1 The sequence of the code 12
         2.2.1.1 Direction and speed-measurement 13
         2.2.1.2 Position mode test 13
         2.2.1.3 Current measurement 14
         2.2.1.4 Sound test 14
         2.2.1.5 Error codes 14
   2.3 Logging CAN communication of the old test equipment 15
      2.3.1 Manual performance of tests on the old test-equipment 17
1 Introduction

1.1 Background

The product created in this project will operate in the production as a final test of the complete mounted PST (Power Steering). There are several tests during the whole mounting of the “box” and this product will be one of them, the final one. There are several ways to construct a product like this, but as mentioned earlier there is already a manual test equipment available that’s based on CAN. That product worked and where operating in the production for many years, but now a decision of making it automatic have been made.

**CAN** is the acronym for Controller Area Network is today used in many applications and is mostly associated with the automobile industry, but the truth is that it’s actually used in many different areas. E.g. the medical industry, in factory automation and the building industry. Why CAN? Because it is one of the fastest network systems and the speed of transmission can operate on baud rates up to 1 Mbit/sec, which is by far faster than other usual connections used to transmit big oodles of data. There’s lot of benefits when CAN is used, but the biggest advantage of using CAN as a field-bus system is that this method is used by many companies worldwide. This makes the availability of components that use CAN easy to acquire for a long time to come. Depending on what sort of application that would like to use field-bus system as serial communication, there are different sorts of field bus system more suitable for different applications. Which also operates on the layers between Data Link Layer and Application software. CAN is only operating on these two layers and the physical layer. This makes CAN more effective by using this savings of memory and use it on other features and also to make the costs lower. Options to CAN is CANopen, DeviceNet and SAE J1939.

1.2 Project description

This project was established to create an automated control which will replace the already existing control, which are handled manually. The tests handled by the new automated control, consists of rotation, direction, position, sound and current measurements. The new automated control will also handle error codes, created by the hardware and the PST. The possibility of connecting a manually handled box like the old equipment is also of desire, so that the operator can confirm the tests or if the automated test-equipment is in need to be repaired. The new automated control is designed to perform the tests more safely regarding to approve the conditions of the tests, but also increase the speed of the test. The user is also given less responsibility because the tests are approved using conditions that are predetermined in the new automatic tests. The project will consist of two areas, partly with existing materials, but also expand with new features. The existing parts will get analyzed and based on that, the new construction will get a design to suite the new application areas. The goal of analyzing the old test-equipment is to get knowledge about its way of operating. Which is done by using SDO (Service Data Object) that are transmitted through CAN (Controller Area Network), these SDOs is interpreted as various tests/commands in hexadecimal form stored into the microprocessor of the PCB. All the objects that is collected from the old test-equipment will be taken into account and will be used for the “new” tests used in the new test-equipment. These “new” tests will be programed in an environment created in LabVIEW, which is meant to represent CAN. The hardware performing the tests will be done by using the old test-equipment, analyze its scheme and use it as a reference. The scheme for the new test-equipment will be created in Tina, a drawing program from Hungary. At last when the programing and the design is finalized, constructors will build it and a test made by the new test-equipment will be performed.
1.3 Theory

1.3.1 PWM

Pulse width modulation, is a time duration modulated signal. Most often it is regulated by using a switch that’s switching on and of the circuit where the signal is operating. Depending on how long time the switch is on for each interval of operation, the PWM Duty cycle could be determined if the switch is on for 95% of the period in operation mode the duty cycle is 95%. The longer the switch is turned on, the power supplied to a motor (if e.g. voltage is used) will increase. ¹

![Figure 1. Three PWM signal describing duty cycle and HI/LOW.²](http://binaryupdates.com/pwm-in-lpc2148-arm7/pwm-duty-cycle-pulse/)

1.3.2 Rotational Sensor (Digital shaft encoder)

Rotational sensor based on digital shaft encoders has a disc with different segments in it, it could either be only high and low. I.e. conducting or non-conducting parts in the disc. Or segment on the disc representing gray code. I.e. it depends on how much of conducting area each segment has. E.g. 0000 has no conducting area and 1111 has no non-conducting area.³

![Figure 2. Two encoders with different properties. To the left non-conducting or conducting areas and to the right an encoder disc with gray segments conducting areas.⁴](http://www.netrino.com/Embedded-Systems/How-To/PWM-Pulse-Width-Modulation)

² (Figure 1) (http://binaryupdates.com/pwm-in-lpc2148-arm7/pwm-duty-cycle-pulse/)
1.3.3 Hexadecimal form into byte and vice versa

To save a lot of data one can change the base of what the data should be stored in. The bigger base the bigger amount of data can be stored into fewer numbers. Hexadecimal form uses more characters than decimal and therefore can store larger amounts of data than bases below such as decimal form. Hexadecimal uses 0-9 and up to fifteen a-f.\(^5\)

E.g. a hexadecimal number could look like below:\(^5\):

\[ FFF_{16} \]  \hspace{1cm} (1)

Converted into bytes looks like the number below:\(^5\):

\[ 1111\ 1111\ 1111\ 1111_2 \]  \hspace{1cm} (2)

1.3.4 CAN (Controller Area Network)

CAN is a bus used most often in systems that’s embedded. It is a half-duplex, two-wire with fast communication. Its main purpose is to within embedded systems supply communication with high baud rates betwixt MCUs and by its superior serial characteristics it abolish other technologies within serial communication.\(^6\)

1.3.5 Service Data Object (SDO)

These are messages send and received in CAN which has low priority in contrary to process data object. When the process data object has been transmitted the low priority data transfer will continue. The size of a message send or received as an SDO is maximum 8 bytes.\(^7\)

An SDO message is stated as follow: 3 bytes | 1 byte | 8 bytes

<table>
<thead>
<tr>
<th>Base + node number</th>
<th>DLC</th>
<th>Data (D0-D7)</th>
</tr>
</thead>
<tbody>
<tr>
<td>60[7]</td>
<td>8</td>
<td>23 75 60 01 78 56 34 12</td>
</tr>
</tbody>
</table>

All messages uses hexadecimal as its base.

E.g. 0D24 written in hexadecimal converted into decimal form equals:

\[
(0 \times 16^3) + (13 \times 16^2) + (2 \times 16^1) + (4 \times 16^0) = 3364
\]

---


1.3.6 Relay

A Relay is an electronic component used for switching on and off signals belonging to a system that it's coupled to. Usually it can have either one or two poles and one throw or two throws.\(^8\)

![Figure 3. Schematic symbols of (to the left) Single Pole Single Throw relay and (to the Right) Single Pole Double throw.\(^9\)](image)

1.3.7 Fuse

The Fuse is a component used for safety regarding electronic circuits. It works such that when high current operating on the metal element inside it will start to heating and when it reaches a certain temperature it will melt and the circuit is than open.\(^10\)

![Figure 4. Different kinds of symbols of fuses drawn schematically.\(^10\)](image)

---


\(^9\) (Figure 3) (Charles Platt. Emcyclopedia of Electronic Components Volume 1. Edition 1. Sebastopol: O’Reilly Media, 2013, 65.)

1.3.8 DAQ

When a measurement has been performed and the measurement have been collected, the measurement is supposed to get analyzed or maybe is needed to get processed in a computer. Then conversion is needed, this because of adaption to get compatible to the processing unit. The conversion from e.g. analogue to digital is then executed by a DAQ (Data Aquisition). A DAQ is a hardware that can either be integrated with different properties converting or receiving the data collected (Figure 5 to the left), it can also be an empty chassis that has to be connected with the computer by e.g. USB and equipped with slots where modules with different properties is connected (Figure 5 to the right).

![Figure 5. DAQs in different forms (to the left) integrated DAQ and (to the right) chassis DAQ.](http://www.ni.com/data-acquisition/what-is/)

1.3.9 DAQ modules

A DAQ-module is a tool utilized to overcome the problematics with integrated I/O pins in acquisition tools. A Data acquisitions chassis which uses modules as I/O pins will be more effective regarding to flexibility, because the user can switch the modules and therefore be able to receive or transmit whatever the data/measurements wanted.

![Figure 6. Different modules belonging to various DAQ chassis. First to the left AIN from National Instruments (NI), the second from the left is an AOUT module also a NI product, the second from the right is a Relay module also a NI product and the lest module is a multifunction module with 80 I/O ports fabricated by 6TL.](http://www.ni.com/sv-se/support/model.ni-9263.html & http://www.ni.com/sv-se/support/model.ni-9403.html, http://www.ni.com/sv-se/support/model.ni-9481.html &)

---

1.3.10 Programmable voltage supply

Also called bench-top power supply is an electronic product which a connection between the unit and a computer is possible. This is possible because of the use of an e.g. digital interface and microcomputers where integrated into the product.

![Programmable voltage supply](image)

Figure 7. The backside of a programmable voltage supply. Offering 1. commutable VAC. 2. Commutable Voltage supply. 3. Analog DSUB. 4. SW1 setup switch. 5. Remote sense connector. 6. GPIB connector. 7. LAN connector. 8. USB connector. 9. Ground screw & nut. 14

1.3.11 D/A and A/D converter

Digital to Analog and Analog to Digital converter is functions of converting as the name says where a circuit process the signals coming in and converts it into the opposite unit. In figure 6 with the DAQ modules, the two on the left hand side of the picture are representing A/D and D/A converters.

![D/A and A/D converter](image)

Figure 8. Two schematics representing a digital to analogue circuit to the left 15 and an analogue to digital circuit to the right. 16 Both the D/A & A/D converter schematics is roughly approximated.

http://www.massinterconnect.de/6tl-produkte/yav-module/digitale-und-analoge-i-os/digitale-und-analoge-i-os.html

14 (Figure 7) [http://www.matsolutions.com/Portals/0/Product%20documents/Agilent%20Technologies/N8736A/N8736A%20User’s%20Guide.pdf]

15 (Figure 8 to the left) [http://www.next.gr/circuits/Digital-to-analog-converter-DAC-I38488.html]

16 (Figure 8 to the right) [https://www.quora.com/How-can-you-create-an-AC-signal-from-DC-with-the-Arduino-Unos-PWM-considered-an-AC-signal-since-it-produces-a-square-wave]
1.3.12 LabVIEW

Laboratory Virtual Instrument Engineering, is a programming language created by National Instruments which is visualized graphically, in contrary to many other programing languages which usually uses text-symbols to create methods, objects and other programing functions. It was originally developed to make it easier for engineers and scientists to manage code faster, so that more time could be spent on the actual work that had to be done. ¹⁷

2 Method & Measurements

2.1 Layout of the system as a whole

2.1.1 The system as a whole

The system as built as in figure 9. How all the different parts of the system where selected, was determined by using the specifications of the PST I/O pins. A picture of the I/O pins and the specifications can be seen in appendix D page 39 (I/O pins PST & specifications).

The test is performed by at first the user has to configure a couple of settings in the Graphical User Interface (GUI). After that the automatic test is started. The software is interacting with the power supply all of the time during the test, because of the current measurements done to protect the system from damages. The Power supply is also programed such that it can supply both 24V PSTs and 48V PSTs.

The sensor is used during the velocity/direction test to confirm that the PST is working as desired. The sensor is supplied with voltage from the power supply during the test and also placed such that the PST makes contact with the rotating part of the sensor. The sensor is depicted in figure 11 on page 11.

The Aggregation/Relay is a hop of relays depending on the same source, this aggregation of relays determines whether the system should run automatically or manually.
The Software is the code developed in LabVIEW with focus to work as a CAN solution for the PST.

One part of the system that is optional, is if the user want to maneuver the system manually, it can be done by plugging in a box installed and constructed such that the user can steer the BLDC motor by hand and make a conclusion self if the motor is running as expected. The schematic for the box with all the components can be seen in appendix A on page 28.

The Data Acquisition in the middle that’s transmits data and collecting the measured physical units, is done by the use of National Instruments cDAQ with external modules for different purposes. The modules picked to operate in this system is analog I/O and a module containing relay channels.

2.1.2 Schematic of the whole system

The whole system was drawn as a scheme in a drawing program called Tina. Which contains all components that is supposed to operate in the system. This was done to get a greater view of the connections of the in and out pins from the hardware operating in the system and what components needed to create the quantity and safety for a functional system as seen from the specifications from all terminals operating in the system. By analyzing the specifications of the I/O of the PST, the basis was created for which modules needed to operate in the cDAQ. A miniature of the schematic can be seen in figure 10 below and for a better view, all schematics can be find in Appendix B.

![Schematic of the PST](image)

*Figure 10. The Whole system drawn schematically.*

The schematic in figure 10 is divided into subgroups because of their applications.

2.1.2.1 PST Power and Enable

The square A contains the Inputs +Battery, Enable and –Battery. Their purpose is to receive power, start the motor, enable it and disable it from running. Enable and +Battery is connected to the same node and all three is connected to the power supply which is programmed such that the correct power is supplied to the PST (24V or 48V). The Enable input and Battery+ input is equipped with a fuse each so that no currents flowing in the reverse direction can destroy the power supply. The current flowing into the PST is that current that is measured in the test named current measuring, this test is more closely explained in the next title named programming. To get a closer look at the schematic, see Appendix B.7, page 32.
2.1.2.2 PST Analog inputs and Lord-brake outputs

The square B is where the type of controlling the motor is determined. It is equipped with six relays switching between automatic controls and manually controlling the steering and all the six of these relays is connected to the same relay channel on the relay module. The default state of the relay channel is set to manual control which is set in the GUI. It is in manually controlling mode by default because the manual controlling should not be dependent on the automatic control at all. The analog inputs are equipped with one diode each connected to ground to protect if there will be a reverse current and if it becomes too high. To get a closer look the schematic, see Appendix B.6, page 31.

2.1.2.3 PST Reference switch

The square C in the schematic handles +12_out, Ref and GND_Ref. The circuit is equipped with a relay and three resistors that will give the right amount of voltage when the switch is put on, otherwise it is led to GND_Ref. To get a closer look at the schematic, see Appendix B.8, page 32.

2.1.2.4 Manually steering box

The square D representing the circuit which offer the possibility for manual steering of the motor. Equipped with double potentiometers, a Lord-brake, 6 relays belonging to a rotational switch and a circuit controlling the reference. The Relays is used to decide if the PST connected to the system should be controlled by the potentiometers or the Lord-brake. The Lord-brakes positive output from the PST is equipped with both a diode and a PTC thermistor to protect the Lord-brake from high currents. The box with the whole circuit for manual steering is connected into the system by plugging in the DSUB contact into the system. To get a closer look at the schematic, see Appendix B.2, page 29.

2.1.2.5 CAN communication

The square E is the part of the system contains the can communication which consists of two cables representing CAN HI and LOW and the PCI “module” which is inserted in to the CPU. More information about the CAN PCI card can be find in Appendix A. To get a closer look at the schematic, see Appendix B.1, page 29.
2.1.2.6 cDAQ modules and rotational sensor

In square F is all the hardware belonging to the compact DAQ and the rotational sensor placed. It contains the sensor, an analog input module, an analog output module and a relay module (The datasheets of the modules can be find in appendices 0). The rotational sensor which will operate in the system can be seen in figure 11 below (The datasheet can be find in appendices A). In this area of the schematic is also the “decision” if a PST with version of Jura or indra/tartan/Jupiter is supposed to be connected to the system, if the automatic control is performing the tests. The decision is represented as one of the connections of the relay module which is programmed such that the user can choose what unit is connected into the system. The area also contains how the hardware of the lord-brake is simulated. By a nominal load, a relay channel from the relay module and an extra resistor which makes the right current flowing into the analog inputs of the PST when the Jura version is connected to the system. To get a closer look at the schematic, see Appendix B.3 & B.4, page 30.

![Figure 11. The sensor used to measure the rotation of the PST.](https://www.elfa.se/en/mini-measuring-wheel-system-kuebler-05-2400-0040-1000-5045/p/11049233)

2.1.2.7 Power supply

The component in square G is the power supply, which is an Agilent Technologies System DC Power Supply Series N8700. Which is going to supply different parts of the system and the PST with voltage and current. There are many different connection possibilities built into this product and all the connections possible can be seen in the data sheet which can be find in Appendix A under Power supply N8700. To get a closer look at the schematic, see Appendix B.5, page 31.

---

2.2 Programming

All the tests performed on the PST was programmed by using different methods developed in LabVIEW. Which is based on what purpose they make. E.g. Resources, test cases and test sequences. Below in figure 12 there is a flow chart depicting the structure of the sequence of the code when it was developed, this was done to get an understanding of how it could have been executed. In the future the tests will be executed separately in whatever order the user wants.

Figure 12. Flow chart of the code as a whole. The Grey bubble describes the start of the test. Error code check orange bubbles. Velocity/Direction test is symbolized by the turquoise bubble. The Position test is symbolized by the azure bubble and the Sound test is depicted as a light blue bubble. The current measurements is visualized as the green bubbles and the end/pass of the test is the light green bubble. Indication of error/fail is the red square in the middle.

2.2.1 The sequence of the code

The code was programmed such that each test is going through a startup which also includes the performance of the test. i.e. firstly start all the hardware needed for the test and then the specific measurements will get completed.

All tests also have a case of disable it from running. E.g. “End position mode test”. After the tests is disabled a “shut down” case is passed as well, to disable enable and all the electronics around the PST. All the tests also passing through a case where the main setup is configured and two cases handling the report and evaluation of the test. All specific tests is explained in the upcoming subtitles.
2.2.1.1 Direction and speed-measurement

The case testing the direction and the speed of the PST, is executed such that the power is enabled and different velocities of RPM is applied during different loops with first a clockwise movement and this is repeated with the only difference that the motor is running counterclockwise instead. To confirm the direction and movement of the motor, a sensor that’s sensing rotation is placed such that the tip of the shaft belonging to the motor (Figure 13) and the tip which is magnetic of the sensor is touching each other during the test. The sensor is collecting the time it takes for the motor to rotate 360 degrees. The collection of rotation is in PWM such it’s easy to see the frequency of the rotation. The direction is confirmed by using the polarity between clockwise rotation and counterclockwise rotation (positive numbers for clockwise movement and negative numbers for counter clockwise movement). The conditions set for passing the test was determined by calculations of the applied RPM converted into frequency and the directions conditions was set such that if the numbers didn’t have the correct sign it was not approve. If the outcome of the tests is not granted the GUI is displaying this by showing a non-lighted led for that test and which one of the properties tested that is faulty and both if neither is granted.

If a PST with the version of Jura is connected to the system the test also measuring the voltage dropped over the resistor which is in series with the relay which represents the lord-brake (See page 30 appendix B figure B.4). This is measured to control whether the +brake and –brake outputs of the PST is working as supposed.

![Figure 13. A BLDC motor with a PST mounted on the top of it.](image)

2.2.1.2 Position mode test

The position mode test is executed such that a continuous dc voltage is altering the position which were the motor should travel. I.e. for a specific voltage applied this voltage is converted into a position. This position is the target of where the shaft of the motor is travelling. The test is approved if the position given is reached, and a failure of the test is obtained on the GUI if the intended target is not reached. The test starts with a reference run, where the PST is searching for the initial position. When the position of origin is reached the PST is automatically switched into position mode. When the motor is in position mode the motor is following each position given to it.
2.2.1.3 Current measurement

The current measurement in the code is included with the direction and velocity test. It’s located in that test because it should not be neglected. It is also located in that test because of the different velocities applied to the motor, which gives different currents depending on the velocity of the motor. If the current measurement was located in the sound test, the different velocities wouldn’t be achieved (Only maximum speed is applied). It would therefore not give a good spread of current-measurements depending on the velocities. The measurement of the physical unit is done by the power supply used in the system and that measurement is then transferred into the cDAQ and then the measurement is handled by two conditions working as a lower and an upper limit that will decide if the current is in the right proportions that will not damage the PST. The specification of the max current fed into the PST was given as 2 Amperes. If the current is operating over the 2 Ampere condition, the system shuts down and an error message is activated and displayed on the GUI.

2.2.1.4 Sound test

The Sound measurement of the code is the only part of the code that is manually determined whether the unit should pass or not. The code was created such that the power stage is enabled to the PST and the BLDC motor coupled to the PST is after that running with the specified maximal RPM set for the PST as long as the user wants to listen, so the user can make sure that the decision is the correct one. If the user decides that the unit shouldn’t pass, the user have to press the button connected such that the test is paused and continues to “shut down” the unit of the test. This test is done in both clockwise movement and counter clockwise movement.

2.2.1.5 Error codes

The error handling was created such that the SDO address error-word in the CAN communication was read and the response from that method was collected and filtered out in a method named ErrorCodes (See page 36 appendix C figure C.8 & C9 ). This method was created to work such that the response/Error collected, a bit representation was filtered by priority of the possible errors occupying the PST. I.e. if there’s several Errors active on the same time, the error with the highest priority is shown. The priority was already determined and was available in the CAN communication info from the company handling the PST fundamentally and this was utilized in the error handling to sort out the error with the highest priority in this project.

One problem with The SDO address “error-word”, is that it only handles the errors arising from the Power STeering. There’s also error arises from the hardware which the “error-word” doesn’t handle. The most common error created by the hardware, is the one detecting if the test-equipment have communication with CAN or not. To find this error in the new system, a methodology that was created by programming. Is sending a SDO message to the PST and if the PST doesn’t answering in a couple of milliseconds, this error is displayed on the GUI. (See page 35 appendix C figure C.6 & C7).
2.3 Logging CAN communication of the old test equipment

By logging the CAN communication, the behavior of the manual test equipment can be understood. The test was performed by connecting the test-equipment, a PST and a DC motor. The PST was exchanged from test to test, because there are different versions of PST and the log was expected to look differently depending on what version of PST that were coupled to the test-equipment. In figure X the components of the test can be seen.

![Figure 14. The complete set-up that was used to log the CAN communication. 1 Test-equipment. 2 CAN transmitter/receiver. 3 PST. 4 DC Motor.](image)

To transmit the CAN communication a serial com port from the test-equipment were used together with a Kvaser Linx J1587 (appendix A), which convert from serial to USB. The software used to interpret the CAN communication is called kvaser can king, The inter face of kvaser CAN king can be seen in figure 14.
To log the CAN communication that is operating in the test-equipment, one have to perform manually all of the operations needed to receive the condition that is desired. During the performance the Kvaser leaf professional converter is receiving all of the data and this can be seen in the log of Kvaser can king in the output window. This window can be seen in figure 15.

Figure 15. Interface of Kvaser CAN King, with the output window in the upper right corner.
2.3.1 Analyze and performance check of the tests on the old test-equipment

First of all the user have to decide what version of PST that is coupled into to the test-equipment, which is done by rolling a button by choosing Indra, Jupiter, Tartan or Jura. This button can be seen at number 1 in figure X. Next step the user have to select the voltage level, which is done by switching HM. Depending on whether it is a 48 V PST or a 24 V PST, this switch can be seen at number 2 in Figure X. When this is done the test-equipment is starting and the user have to use the button named reference signal. The button can be seen at number 3 in figure 16. After that the test-equipment can be used to navigate position. This is performed differently, depending on what version that is coupled into the test-equipment. If it is a Jura it’s navigated by the steering-wheel as one can see at number 5 in figure 16 (Lordbrake) and if it is another version e.g. Tartan, the user should use the potentiometer that can be seen at number 4 in figure 16. If rotation or sound would like to be obtained one should press button number 7 in figure 16. This mode give the user opportunity to control the motors rotation. During this test the user can also listen to the sound coming from the gearing wheel and make the conclusion whether it’s operating optimally or not.

Figure 16. Old test-equipment. 1 choosing version of PST. 2 select voltage-level of the PST. 3 Send reference signal. 4 Change direction and speed for Indra, Jupiter and Tartan version. 5. Steering-wheel changing direction and speed of the Jura version. 6 Indicator of error-codes. 7 Toggle between position and rotation. 8 Analogue current indicator. 9 Fuse. 10 Indicator of high current reached. 11 Indicates that to high current is flowing in the system.
3 Results

3.1 CAN-log of the old test-equipment during startup and running

The behavior of the CAN-communication was saved into text files for each test made, and is presented below.

3.1.1 PST 24 CAN log

3.1.1.1 SDO packages during start-up

With 24 V and Indra Jupiter Tartan mode chosen. The log shows two command that represents the read function (40 in the first byte) and two commands that represents answers (4B in the first byte). See figure 17 below.

<table>
<thead>
<tr>
<th>Identifier</th>
<th>Flg</th>
<th>DLC</th>
<th>D0...1...2...3...4...5...6...D7</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>00000607</td>
<td>4</td>
<td>40</td>
<td>05 30 02</td>
<td>7.200840</td>
</tr>
<tr>
<td>0000587</td>
<td>6</td>
<td>4B</td>
<td>05 30 02 89 00</td>
<td>7.201860</td>
</tr>
<tr>
<td>00000607</td>
<td>4</td>
<td>40</td>
<td>05 30 03</td>
<td>7.205830</td>
</tr>
<tr>
<td>0000587</td>
<td>6</td>
<td>4B</td>
<td>05 30 03 8E 00</td>
<td>7.206850</td>
</tr>
</tbody>
</table>

Figure 17. Log showing the initial state of the operation right after the power is turned on.

The Identifier showing 607 and 587 which declares if it’s the corresponding PST controller or if it’s a command sent to the controller. Number 607 stands for a command send to the controller and 587 stands for that an answer has been obtained from the PST controller. D1 and D2 should change place and represent 3005. D3 is a sub-index, D4 and D5 is bits containing values and should also switch place to have the correct representation. 3005_2 is standing for FLASH Parameter Checksum and 3005_3 is standing for Progressive Steering Parameter Checksum. The answers says the same thing with the exception that it also sends the value of the parameters in hexa-decimal representation (D5 D4).

3.1.1.2 SDO packages during run

If the toggle button is pushed (change from position mode to velocity mode), the same SDO readings are sent and the same answers are received as during the start-up. When the toggle button is pushed to change from position to rotation mode nothing happens as was mentioned in the previous sentence, but when the user starts to spin the potentiometer to increase/or decrease the velocity the log receives two new SDO messages. The messages can be seen in figure 18 below.

| 00000607 | 6 | 2B | 70 | 20 | 02 | 01 | 00 | 181.963140 |
| 0000587  | 4 | 60 | 70 | 20 | 02 | 181.964930 |

Figure 18. Log showing one SDO message writing and one SDO message of the answer from the PST controller.

D0 in the first row stands for a write command (2B) and the second rows D0 bit (60) stands for an answer from the PST controller, when a write command has been sent. First row representing 2070_2 representing Selector for source of desired value (velocity) and the answer says the same by confirming. The value sent in the first row is 1, hexa-decimal representation 0x0001.
3.1.2 PST 48 CAN log

3.1.2.1 Service Data Objects

If a 48V PST is connected to the manually handled test-equipment and Indra tartan Jupiter version is chosen, the log of CAN is almost the same as for a 24V PST.

3.1.2.2 SDO during start-up

The difference between the two PSTs during start-up and reference-button are pushed is that the checksum parameter is set to 0x001C for a 48V PST instead of 0x00B9 which is set for a 24V PST.

3.1.2.3 SDO packages during run

During the operation the same SDO message as a 24V PST is registered in the CAN log. I.e. that when the toggle between position and velocity mode is pushed, the CAN communication log is changed in the same way in the operation of executing the test performed by the test-equipment for both these kind of PST.

3.1.3 PST 24 JURA CAN log

3.1.3.1 Error-codes

When a faulty PST is connected to the test-equipment or the test-equipment itself causes error this will be indicated by the test-equipment. This is shown by a display, placed on the test-equipment.

The analysis of the error code 3130, which stands for failure phase cut. Was analyzed to see what the CAN communication receives when an error code is detected by the test-equipment.

3.1.3.2 Service Data Objects

The CAN-log is the same as a PST which not have errors, but with one difference. See figure 19 below.

```
0 0000087 8 30 31 00 00 00 00 00 46.041790
```

*Figure 19. One row from the CAN-log during a run with a faulty PST connected to the test-equipment (D1 D0 = 31 30).*

The other SDO that appears usually, couldn’t be analyzed because the PST had such an error that the PST couldn’t start. When the PST isn’t starting there is no printout in the CAN log.
3.2 All couplings

The completed system with all hardware can be seen in figure 20 below. To see the schematics and correlations with the results please look at the schematics in appendix B.

![Figure 20. The system as a whole.](image)

The preliminary fixture with a sliding function in the bottom to get a good placement to receive optimal measurements from the rotational sensor can be seen in figure 21 below.

![Figure 21. The PST with moveable fixture and the rotational sensor.](image)
In figure 22 is the cDAQ with modules placed on the din-rail depicted.

![Figure 22. compact DAQ with all modules used in the project.](image1)

To get connection with the manual control steering boxes I/O, the relays that’s determining whether the system should be controlled automatic or manual have terminals available for this purpose and this can be seen in figure 23 below.

![Figure 23. The relays determining automatic or manual system control. There six terminals is empty on the upper row, where the manual steering box is supposed to be connected.](image2)
All terminals, relays and a grid connected power-supply was placed on the din-rail, which can be seen in figure 24 below.

**Figure 24.** To the left is a voltage supply delivering 12V DC used to supply the rotation sensor and the current measuring unit. Terminal with number 12 is coupled to the plus potential of the DSP left of it. Terminal with number 21 is coupled to the negative potential of the same DSP. Terminal with number 24 is coupled to the power supply delivering 24V DC. The terminal with the fuse coupled above on it, is the terminal connected to the pin of enable. Terminal 42 is connected to ground (GND) of the power supply delivering 24V DC. Terminal 1, 2 & 3 is used for prolongation of the wires. The 6 relays with double latch is the relays determining if the system is controlled manually or automatic. The two relays furthest to the right is the relays latching between different connection depending on what kind of steering servo that is connected to the system.

**Figure 25.** The PCB that is supplied with the components performing the current measurement of the main DC supply.

The solution of the current measurement above is just preliminary, then the actual power supply unit was not available due to lead times.
3.3 GUI LabVIEW

3.3.1 Initial GUI for selection of test and configuration

When the PST is connected and ready for testing, the GUI below will pop up as a selector. There the user have the possibility to choose one of the tests. The configuration needed of the parameters of CAN (Test setup tab) have to be done before starting a test. It’s the same when configuration of the control settings is needed. In the Direction/velocity test the operator can change the frequency and the current tolerances. The three tests Position mode Test, Direction/Velocity Test and Sound Test is selected after any configurations, because the tab with the desired test has to be the one active/displayed to the user in the GUI to get started.

![Initial GUI](image)

*Figure 22. Initial GUI.*
3.3.2 GUI displayed during the test performance

When one of the tests has been selected the GUI below is displayed during the test. Depending on what test the operator have chosen, is it only one of the three squares that is needed to be controlled. The led in the right upper corner is active for any of the tests and is the control indicator if CAN is available or not. The led in the right bottom corner is all for indication if the test conditions were approved or not.

![GUI](image)

*Figure 23. GUI that’s displayed during the test sequence.*
3.4 Programming – Test cases -- Error handling

3.4.1 The code

The result of the code ended up in one “start case” for each test and one “end case” for each test. As it says the start case initiating and perform the tests and the end tests disable and ending the tests. The state “Is Ready” is where all local variable is placed to get there default values before a new test is performed. The state “SetupGui” is where the CAN communication configuration is located and the shift between manual or automatic control is placed. The state “Shut down unit” is where the relays operating for the enable pin and the type of control wanted is disabled. The state “Evaluate Test” is where the evaluation of the data collected from the various measurements is processed. The state “Report” is where the data is stored into files and send to maps where the files is stored. More details and pictures of the code can be seen in appendix C.

![Figure 24. The case structure selector containing all the cases handled in the code.](image)

3.4.2 Test cases

As already mentioned in the first sentence in the forgoing paragraph (3.4.1 “The code”) the test cases is divided into a start and an end case. To see how they are structured with conditions, functions and methods in code look at appendix C.

3.4.3 Error handling

The Errors caused by the PST, is in code handled by the readings from the Error word. The Error word is then analyzed with different functions of LabVIEW and then sent to a case selector. The case selector have one error for each case. These cases consists of two parameters, one string and a number constant. The two parameters mentioned in the forgoing sentence is then displayed on the GUI showed during the tests. If an error wasn’t detected the display shows “no errors was found”. To see the result of these method created please look at appendix C8 and C9.

3.4.4 How the PST Jura is handled

If a PST with characteristics of Jura is connected to the system, the pin 1 and 9 is switched from serial I/O to Brake +/- and the serial I/O of a Jura is located on the PCB instead. This is treated such that a relay is switching these pins from one state to another and in the code this is done by the methods created for cDAQ modules. This is executed in SetupGui and configured while the code is running under the tab “test setup”.

25
4 Discussion

4.1 Comparison between the old test equipment with the new one

The biggest difference between the new and the old test equipment is that the control is changed from manual to almost non manual control. Some buttons is needed to be pushed by the mouse but more than that nothing else has to be performed by hand more than put the PST on the test rig and slide towards the sensor until they touch each other. The size of all hardware is almost the same now as it was before. This will change in the future because this is just a prototype and will maybe get larger or maybe smaller depending on the size of eventual hardware needed for added features to the system or changed hardware for already existing features. The tests performed is maybe not much faster than before but will be more precise in the way of approve the tests. Now the test condition is the same for each operator handling the tests, which was not the case before there it changed from operator to another.

4.2 The Code

The code in total and the end product feels like it never will be finished but for now it’s at least making all the tests and following more or less the structure of what the company asked for. There will always be things needed to look in too much deeper and optimize to make it more effective. Here is time a big thing to take into account. The total time of running the code which is from the operator makes its choice of what test should be executed until when the code is finished. I.e. when the operator is able to see the end result, is not as fast as predetermined. Still the execution of the tests is running fast, but as mentioned earlier with a little more time to debug changes could have been made. It was a big project with many parts involved so the programming was lacking due to time constraints. The end result is working and this is just a prototype and there will be more time for optimization.

4.3 The planning of the system

The planning consisted of flowcharts, drawing programs for schematics and some calculations for test conditions and tolerances for different hardware. The most difficult thing to take into account was the size of the project. This was something that had to be accepted because the project seemed interesting and very rewarding because of all different activities that it included. The plan was from the beginning to finish the project already in the beginning of June 2018, but this was already in the end of May something that couldn’t be achieved. The biggest problem was the lead time of hardware that was ordered. Another thing that was taken into account was the debugging. This is something that cannot be planned that accurate, but a plan has to be provided. In the end the planning of this project worked out great, but this project should have had more time to get an even greater end product.
5 conclusions

The project was finalized a month after the end that was planned. This was caused by too much new areas to get familiar with, but also that ordinary work was done beside of the project for the same company. This was of course a choice made by me the author and have giving me a lot of understanding of how it is to be involved in many working activities at the same time. The conclusion of the project described briefly is that the planning can never get to much time, this is important and will make it easy because then there is always something to fall back onto. The thing about spending too much time about the planning is that the project will get finished more quickly with a good plan. Sometimes this is changed of course because of debugging and lead times of ordered equipment, but this is actions that can never be avoided most often. Nothings is ideal which probably everyone with a good plan hopes fore. There should always be time for debugging taken into account when projects handling electronics and programming. Talking about the project with respect to each part involved, the calculations of what units and quantity each hardware component needed was the action that took the most amount of time. Because the data-sheets given from various hardware is very often not very generous talking about information. E.g. internal resistance, currents and power. The sensor that was used in the project made a lot of headaches, therefore the data-sheet was so ungenerous with information, so that actually two different sensors have been used throughout the project. This was because of that the first one was damaged when too much current was supplied to it. This was caused by the data-sheet containing to less information regarding units and quantities of them. Positive things about the project was the experience working in LabVIEW as a software. The positive attitude to LabVIEW was maybe not the graphical programming more the way it was used in the project. The code was object oriented and was created in many instances before it could be used in the actual code. This have given a lot of understanding of LabVIEW and how one can use LabVIEW more than just “hård kåda” something like “Hard-programming” in English. This where one of the parts that took a lot of time to get into before understanding it properly. Before the project my object oriented programming was only experienced in other languages such as PYTHON and Java. The schematic drawn in the drawing program TINA worked out fine, because of the similarities of other drawing program used before. There was some functions that was a little bit different but worked as any other already experienced drawing program, such as LT-spice and OrCAD. When drawing the schematics in the planning stage all components is considered to work ideal, which simplifies the drawing. When the connections of the different physical hardware and edit the already existing schematics complexity is reached. This is when simulations doesn’t correspond to the changes made between different physical hardware of the real construction. Finally the test-equipment created in this project is now working as requested. It can perform automatic steering and measurement evaluation of the tests wanted of the direction, velocity, position and current. It also automatically collects the errors from the PST. The sound test is driven automatically but the test approval is still needed to get approved by an operator (manually). In the future the test-equipment has to get optimized by making the evaluation of the sound test automatic. Last thing to say, the test-equipment created is a prototype. It operates as requested, but many improvements are expected in the future.
7 Appendices

A

Components detailed list

Rotational sensor
Mini measuring wheel system: 05.2400.0040.1000.5045, Kübler.

CAN transmitter/receiver
https://www.kvaser.com/product/kvaser-linx-j1587/

Current transducer
G2.00.23.103.0

Power supply TDK lambda
DSP10-12 AC/DC

Fuse terminal Weidmuller
WSI 6

Terminal block Weidmuller
WDU 4

Relay module Phoenix
DEK-REL-624 21

DCDC converter 12-5
PM05S050A

compact DAQ National instruments
NI9174

NI modules
Analogue IN – NI 9482
Analogue OUT – NI 9263
Relay – NI 9402
DIO – NI 9401
B

Schematics TINA

Figure B.1 The components handling the CAN communication.

Figure B.2 Schematic representing the manual control of the system.
**Figure B.3** Schematic representing the connections of the analogue in pins of the PST and the rotation sensor.

**Figure B.4** Schematic representing the connections of RS232, current measurement, reference signal, Enable and so on.
Figure B.5 Symbol and pins representing the Power supply that's supposed to be used in the future.

Figure B.6 Schematic representing the relays determining the control of the system (Auto/Manual) and different pins I/O to the PST.
Figure B.7 Schematic representing the connections of enable, Bat+ and Bat- of the PST.

Figure B.8 Schematic representing the reference switch circuit.
Figure C.1. An example of one state used to activate the test by selecting the source of determination steering the unit (in this code 1, standing for analogue input) and activation of what function to perform (in this code speed).

Figure C.2. All test is started with an reference run and this is depicted in the figure above.
Figure C.3 All test conditions needed to be fulfilled for passing or failing is determined in code such as the picture above is depicting.

Figure C.4 This is how the cDAQ modules are disabled.
Figure C.5 How the PST is disabled. The two tests containing velocity is also a command send to change the source of steering the PST in the disabling stage. This is done by not blocking this choice for the Position test.

Figure C.6 Here is the CAN communication checked, by performing a reset of the PST and then read the command word to see if the reset was done properly. If the answer is greater than zero CAN communication is available.
Figure C.7 This is the end part of the code from figure C.6 which is depicting the configuration of the start settings given to the PST if the CAN communication is available.

Figure C.8 Here is the start part of the code handling the error numbers read out from the error word of the PST.
Figure C.9 Ending part of the code handling the error numbers from the error word of the PST. Here the state machine gives different explanations of errors and what priority the error has due to the directives from the PCB supplier.

Figure C.10 An example of code performing a DO command for a cDAQ module. (In this code a write command is given).
**Figure C.11** An example of code initiating a channel for a module belonging to the cDAQ. In this code a digital output is initiated.

**Figure C.12** An example of code representing one of the tests automatic control. In this code the velocity and direction is steered.

**Figure C.13** How the relay that’s determining Auto or manual steering is disabled to the left and also the Enable of the PST relay to the right.
I/O pins of the PST & specifications

1. Brake+/Serial in
2. Can Hi
3. +12V out
4. +5V out
5. AIN0
6. Enable
7/15. Bat-
8/16. Bat +
9. Brake-/Serial out
10. Can Low
11. Ref
12. GND Ref
13. AIN1
14. GND out
Glossary

**Allied motion** – A Company designing, manufacturing and sells different services for electronic, electro-magnetic and mechanical motion control.

**PST** - Power steering.

**Jura** - One of the different steering servo drives that is handled by Allied motion.

**Indra** - One of the different steering servo drives that is handled by Allied motion.

**Tartan** - One of the different steering servo drives that is handled by Allied motion.

**Jupiter** - One of the different steering servo drives that is handled by Allied motion.

**PCB** – Printed circuit board.

**cDAQ** – compact Data AcQuisition.

**GUI** – Graphical user interface.

**CAN** – Controller area network.

**RPM** – Revolutions per minute (A unit including frequency)

**I/O** – In/Outs (pins or terminals on the specific hardware)

**MCU** – Microcontroller unit.

**Chassis** – Internal framework of an artificial object.

**CPU** – Central processing unit.

**Ethernet** – An area of computer network technologies often used in computer networks within a limited area.

**Lordbrake** – A certain sort of steering wheel used for one of the PSTs handled by Allied motion.

**Diode** – Semi-conductor component denying the current to flow in both directions of the branch.

**PTC** – Positive Temperature Coefficient thermistor – Resistance increases with increased temperature.

**DSUB** – Electrical connector

**Kvaser can king** – Company handling CAN solutions.

**BLDC** – Brushless Direct current