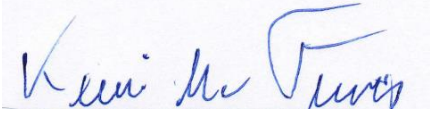




Universitetet  
i Stavanger

FACULTY OF SCIENCE AND TECHNOLOGY

## MASTER'S THESIS

Study program/specialization: Computer Science	Spring semester, 2016  Open / <del>Confidential</del>
Author: Kevin Mc Tiernan	 ..... (signature author)
IRIS Supervisor: Eric Cayeux  Faculty Supervisor: Terje Kårstad	
Title of Master's Thesis:  <b>Integration of process optimization and diagnostic functions in a real-time system that supervises the automatic drilling of heterogeneous formation rocks</b>	
ECTS: 30	
Subject headings: Electrical design, Autonomous system, Drilling Control, Drilling Performance, Real time systems, Finite automata	Pages: 50 + attachments/other: 92  Stavanger, 10/07-2016 Date/year

## Abstract

This thesis presents the technical challenges associated with creating an autonomous system for drilling in heterogeneous rock formations. And therein after describes the methods and solutions of which the UIS team used to design and build a prototype for this problem. This thesis is a result of an attempt to compete in the international Drillbotics competition.

# Acknowledgments

I would like to thank Chief Scientist Eric Cayeux and Senior Research Engineer Robert Ewald for their assistance and guidance during the thesis work done at IRIS, Associate Professor Terje Kårstad for his help and advice with this thesis report. A shout out to Patrik Seldal Bakke and Bjarte Odin Kvamme for their input and constructive criticism and finally I would like to thank my family and girlfriend for their support and help in keeping motivation at an all-time high during this thesis period.

# Preface

This thesis is submitted in partial fulfilment of the requirements to complete the Master of Science (M.Sc.) degree at the Department of Electrical and Computer Engineering at the University of Stavanger (UiS), Stavanger Norway.

The work has been done at the International Research Institute of Stavanger under the supervision of Chief Scientist Eric Cayuex and Associate Professor Terje Kårstad from the University of Stavanger. It contains work done from January to June 2016.

The study is on how to create an autonomous drilling rig which is self-diagnosing and optimizing to be able to drill through “a given” rock size in the fastest possible speed without jeopardizing any hardware incorporated in its build. An attempt to build a small scale prototype which incorporates these functions is made through the use of microcontrollers and backend software developed through the .NET framework.

This thesis might be helpful for those who think of competing in the international Drillbotic competition as well as those looking into automation of drilling.

June 15, 2016

Kevin Mc Tiernan

**Contents**

- Nomenclature ..... I
- Abbreviations ..... II
- List of Figures ..... III
- List of Tables ..... IV
- List of Equations ..... V
- Introduction ..... 1
  - 1.1 Problem description..... 2
  - 1.2 Related work..... 2
  - 1.3 Organization of the Thesis ..... 2
- 2. Analysis of the problem ..... 3
  - 2.1 Background ..... 3
    - 2.1.1 Definition of real time ..... 3
    - 2.1.2 Arduino..... 6
    - 2.1.3 The line between deterministic and nondeterministic ..... 7
    - 2.1.4 Interpolation[15]..... 7
  - 2.2 Communication protocols ..... 8
    - 2.2.1 TCP..... 8
    - 2.2.2 UDP ..... 8
    - 2.2.3 SPI[2,3,4]..... 9
    - 2.2.4 I<sup>2</sup>C[5] ..... 10
  - 2.2 Control technique[6]..... 11
  - 2.3 Safety envelopes..... 11
  - 2.4 Safety triggers..... 12
  - 2.5 Calibration ..... 12
  - 2.6 Diagnostic..... 12
  - 2.7 Optimization ..... 13
- 3. Software architecture for real-time management of a complex system ..... 14
- 4. Safety Envelops ..... 16
  - 4.1 Circulation system ..... 16
    - 4.1.1 Pressure loss formulas[7] ..... 17
    - 4.1.2 Component characteristics..... 19
    - 4.1.3 Calculation results based on software ..... 20
    - 4.1.4 Stochastic simulation..... 22
  - 4.2 Power Transmission – Rotary control ..... 23

5. Cybernetic work that had to be done .....	24
5.1 Initial versus finalized design .....	25
5.2 Overview .....	26
5.3 Main Control box(MCB).....	27
5.4 Directional control box(DCB).....	29
5.5 Amplifier circuits .....	30
6. Implementation.....	31
6.1 Microcontrollers .....	31
6.1.1 Fluid System.....	31
6.1.2 Hoisting System .....	33
6.1.3 Power Transmission system .....	35
6.1.4 Directional system[Appendix G].....	36
7. Calibration procedures .....	38
7.1 Physical Calibration .....	38
7.1.1 Fluid Calibration.....	38
7.1.2 Power transmission Calibration.....	39
7.1.2 Hoisting Calibration .....	40
7.2 Software Calibration procedures .....	41
8. Dynamic conditions.....	43
8.1 Stick-slip.....	43
8.2 Whirl.....	43
8.3 Buckling .....	43
8.4 Twist off .....	43
8.5 Overpull.....	43
8.6 Pack-Off[12].....	43
8.7 Washout[13] .....	43
9. Graphical user interface and logging.....	44
10. Competition results.....	47
11. Conclusion.....	48
References .....	50
Appendix A .....	51
Appendix B .....	52
Appendix C .....	70
Appendix D .....	75
Appendix E.....	76
Appendix F .....	77

Appendix G ..... 78  
Appendix H ..... 88  
Appendix H ..... 89

## Nomenclature

$CSS$	<i>Confined compressive strength</i>
$d_{bit}$	<i>Bit diameter</i>
$d_i$	<i>Internal diameter</i>
$d_o$	<i>Outside diameter</i>
$Q$	<i>Flow rate</i>
$\rho$	<i>Fluid density</i>
$v_{flow}$	<i>Flow velocity</i>
$Re$	<i>Reynolds number</i>
$\mu$	<i>Fluid viscosity</i>
$MSE$	<i>Mechanical Specific Energy</i>
$\Delta p$	<i>Pressure loss</i>
$L$	<i>Length</i>
$f_D$	<i>Darcy's friction factor</i>
$\varepsilon$	<i>Roughness coefficient</i>



## **Abbreviations**

<i>ACK</i>	Acknowledgment
<i>ADC</i>	Analog-to-digital converter
<i>BHA</i>	Bottom Hole Assembly
<i>CCS</i>	Confined Compressive Strength
<i>DFSA</i>	Deterministic Finite State Automaton
<i>Due</i>	Arduino Due
<i>FSM</i>	Finite State Automaton
<i>GUI</i>	Graphical User Interface
<i>HP</i>	Horse Power
<i>I2C</i>	Inter-Integrated Circuit
<i>ID</i>	Inside Diameter
<i>IRIS</i>	International Research Institute of Stavanger
<i>Mega</i>	Arduino Mega
<i>MSE</i>	Mechanical Specific Energy
<i>OD</i>	Outside Diameter
<i>OPC</i>	OLE for Process Control
<i>PDC</i>	Polycrystalline Diamond Compact
<i>PLC</i>	Programmable Logic Controller
<i>ROP</i>	Rate of Penetration
<i>RPM</i>	Rounds per minute
<i>RPS</i>	Rounds per second
<i>RTC</i>	Real Time Controller
<i>SDC</i>	Strategic Decision Controller
<i>SPI</i>	Serial Peripheral Interface
<i>SSR</i>	Solid State Relay
<i>TCP</i>	Transmission Control Protocol
<i>TFA</i>	Total Flow Area
<i>TTL</i>	Transistor-transistor logic
<i>UCS</i>	Uniaxial Compressive Strength
<i>UDP</i>	User Datagram Protocol or Universal Datagram Protocol
<i>UIA</i>	University of Agder
<i>WOB</i>	Weight on Bit

## List of Figures

Figure 1 - Aluminium Pipe.....	3
Figure 2 - Hoisting subsystem.....	4
Figure 3 - Arduino Mega.....	6
Figure 4- Simple spi setup, One master, one slave.....	9
Figure 5 - One Master, Multiple slaves .....	9
Figure 6 - I2C setup, One master, Multiple slaves .....	10
Figure 7 - DFA with 3 states .....	11
Figure 8 - Optimization by recognising MSE .....	13
Figure 9 - Hierarchical structure of the system .....	14
Figure 10 - Hierarchical structure within control box one(MCB).....	15
Figure 11 - Hierarchical control structure within control box two(DCB).....	15
Figure 12 - Fluid system overview .....	16
Figure 13 - Limits between laminar, Critical and turbulent flow .....	17
Figure 14 - Water/Diesel pump 12 V (Biltema 25987).....	20
Figure 15 - ITT Flojet Diaphragm Electric Positive Displacement Pump, 19L/min, 3.1 bar, 12 V dc. 21	
Figure 16 - Stochastic result of the Biltema pump .....	22
Figure 17 - Stochastic result of two identical pumps in series(BILTEMA PUMP USED).....	22
Figure 18 - Stochastic result, selected ITT pump.....	22
Figure 19 - Physical drill string.....	23
Figure 20 - Components for the electrical setup .....	24
Figure 21 - Table and graph displaying loop duration vs instructions .....	25
Figure 22 - Electrical setup overview.....	26
Figure 23 - MCB being constructed.....	27
Figure 24 - MCB finished .....	28
Figure 25 - DCB finished.....	29
Figure 26 - Custom amp(Left), Sparkfun amp(Right).....	30
Figure 27 - Fluid system state diagram .....	31
Figure 28 - Hoisting system state diagram .....	33
Figure 29 - Power transmission state diagram .....	35
Figure 30 - Directional control state diagram.....	37
Figure 31 - Ball valve.....	38
Figure 32 - Obstruction valve(Left), Leak valve(Right) .....	39
Figure 33 - Physical setup(left), Overview(Right) .....	39
Figure 34 - Hoisting load cells .....	40
Figure 35 - Hoisting setup.....	40
Figure 36- Hoisting overview.....	41
Figure 37 - Directional control overview .....	42
Figure 38 - IRIS GUI.....	44
Figure 39 - Competition result UIS .....	47
Figure 40 - Competition summary .....	47

## List of Tables

Table 1 - Ideal team setup .....	1
Table 2 - State transition table for figure 7.....	11
Table 3 - Component characteristics .....	19
Table 4 - Initial pressure loss calculations .....	19
Table 5 - Pressure loss calculation through sim. software .....	20
Table 6 - Pressure loss calculation through sim. software for two identical pumps in series .....	21
Table 7 - Pressure loss calculation sim. software selected pump .....	21
Table 8 - Lookup table torque .....	23
Table 9 - Component list MCB .....	27
Table 10 - Component list DCB.....	29
Table 12 - Budget .....	49

## List of Equations

Equation 1 - Power transmission loop time.....	4
Equation 2 - Hoisting system loop time .....	4
Equation 3 - Fluid system loop time.....	5
Equation 4 - Linear interpolation .....	7
Equation 5 - The Darcy-Weisbach equation .....	17
Equation 6 - Darcy friction factor, Reynolds number equation .....	17
Equation 7 - Laminar flow equation.....	18
Equation 8 - Iterativ solution, turbulent flow .....	18
Equation 9 - Pressure loss equation swivel .....	18
Equation 10 - Hydrostatic pressure loss equation .....	18
Equation 11 - Pressure loss equation BIT .....	19
Equation 12 - Sparkfun Amp Sampling Rate .....	30

## Introduction

This thesis is a result of the international competition DrillBotics2016 where the goal is to create a small scale autonomous drilling rig which can drill through a cubic rock of the size 76x76x76cm<sup>1</sup>.

*“Drillbotics is a competition that combines surface and down hole measurements with process control algorithms to automatically control oil well drilling machines to enable safer and more efficient drilling.”[1]*

The competition focuses on combining multiple disciplines and requires competing teams to have at least one petroleum’s engineer, besides that the teams are free to bring in which ever disciplines they deem necessary to get the task in question done without exceeding the five person limit.

The ideal team setup would be as listed below:

Discipline	Number of students
Petroleum	1
Computer Science	1
Cybernetics	1
Mechanical	1
Physics	1

TABLE 1 - IDEAL TEAM SETUP

This setup gives the team the best chance of success to overcome the various challenges which follows the task at hand.

In the case of our team, the “advertisement” for students to join the team came out late September/early December in a period where most students already had selected their thesis subject. Which lead to a team consisting of two petroleum’s engineers and one software engineer.

As the team didn’t manage to gather one student from each of the ideal disciplines, the remaining work was distributed amongst the team members based on their qualifications, and the work that should have been done by a cybernetics student fell to, me, the software engineer.

The first step in the competition was to create an initial design report (See Appendix A) which had its deadline the 31.December 2015. With the assistance of the scientists at IRIS the team managed to create the initial design report of the system and submit it before the announced deadline. Unfortunately our team was not selected to participate further in the competition, but managed to setup an unofficial competition with the UIA, where both teams agreed to follow the rules of Drillbotics.

With the team not competing in the main competition, it was decided to reanalyse the initial design and attempt to make it more cost efficient by opting out of industrial standard equipment and focusing on the use of cheap open-source hardware.

---

<sup>1</sup> The competition rules have been unclear as to how large the rock would be. Initially the rock block was supposed to be of the size 30x30x30 cm. But analysing the block which was used in 2015 it is more likely that the rock in question was more along the lines of 30x30x30 in. So if we’re assuming that the rock will be close to the one used during the competition in 2015, it is more likely that size will be around 76x76x76cm. The rules also specified that the block size could be increase by up to 20% which would mean that the block at it maximum size would be 91.2x91.2x91.2cm.

## **1.1 Problem description**

The problem the team tried to solve was, how to create a real-time system for automatic drilling in heterogeneous rock formations which is self-supervising. The system should detect and react to deteriorating drilling conditions while recalibrating itself to achieve optimal drilling conditions in terms of ROP, RPM, WOB and also to shut down if the system deemed it necessary to avoid damaging any hardware incorporated in it.

## **1.2 Related work**

In recent years as the cost of drilling has increased and the industry has been looking into automating larger portions of the drilling process, both to reduce the long term cost and to increase the reliability and security.

Previous and current work on this subject is hard to get hold of because most companies are developing in house solutions and still consider their research trade secrets.

Drillbotics releases videos and a short presentation of the designs previously submitted, but these only contain superficial overviews of the designs. This is to make sure that teams presently competing can't copy the designs completely.

## **1.3 Organization of the Thesis**

The thesis is organized in the following manner:

- Chapter 2 presents basic background and theory which the project is based on.
- Chapter 3 presents the control hierarchy of the system.
- Chapter 4 details the variables in the system which needs to be monitored and calculated for the system to operate in a safe state.
- Chapter 5 describes the cybernetic implementation which had to be done.
- Chapter 6 describes the state diagrams used in the microcontrollers.
- Chapter 7 details the calibration procedures.
- Chapter 8 discusses the dynamic conditions and failure states which the system strives to avoid.
- Chapter 9 describes how the graphical user interface.
- Chapter 10 presents the competition results.
- Chapter 11 presents the conclusion and budget.

## 2. Analysis of the problem

This section covers some of the more problematic aspects of the task and proposes solutions for the different aspects related to the problem in question.

The first subsection covers the background theory of the project, with the next section describing the different communication protocols which were considered and finally the last sections describe the key features which the system should contain.

### 2.1 Background

#### 2.1.1 Definition of real time

In general when referring to real time, one refers to the current time at this given point in time. One could say that real time is now. In a computing real time can be viewed in many different ways, the most general way of looking at it is the data that is generated at this point in time. As an example we can look at airline traffic, flights continuously send data to flight control relaying their position and altitude. This data is transmitted in real time so flight control can always view their current position.

In a real system there is no generalized answer for what can be defined as real time. The definition of real time in a given real time system will vary based on its needs and requirements.

The system in question, contains four subsystems; Hoisting, Power Transmission (Top Drive), fluid system and directional. Three of these systems have a set of required response times associated with them and the required response time for these three systems will be described briefly, the directional control will be described in a later chapter.

The main system constraint for this system is the aluminium drill pipe which is used in drilling. The wall thickness of this pipe which is only 0.016-in (0.4064 mm).



FIGURE 1 - ALUMINIUM PIPE

This makes the pipe very fragile and easy to both buckle and limits the amount of torque which the system can apply to it without twisting the pipe clean off.

Through calculations and the simulation software made available by IRIS, the petroleum engineers found that the maximum twist angle which the aluminium pipe can endure is 0.3°, this leads to the following limitation on the top drive.

The step motor used at the top drive has 10 000 steps, this results in an angular resolution of 0.036 degrees. If the bit gets stuck, the motor can perform approximately 8 steps before the pipe reaches its maximum twist angle. If the system ran at its maximum capacity of 180 RPM, it is shown that between each step there is an interval of 33 microseconds.

Assuming a maximum RPM of 180:

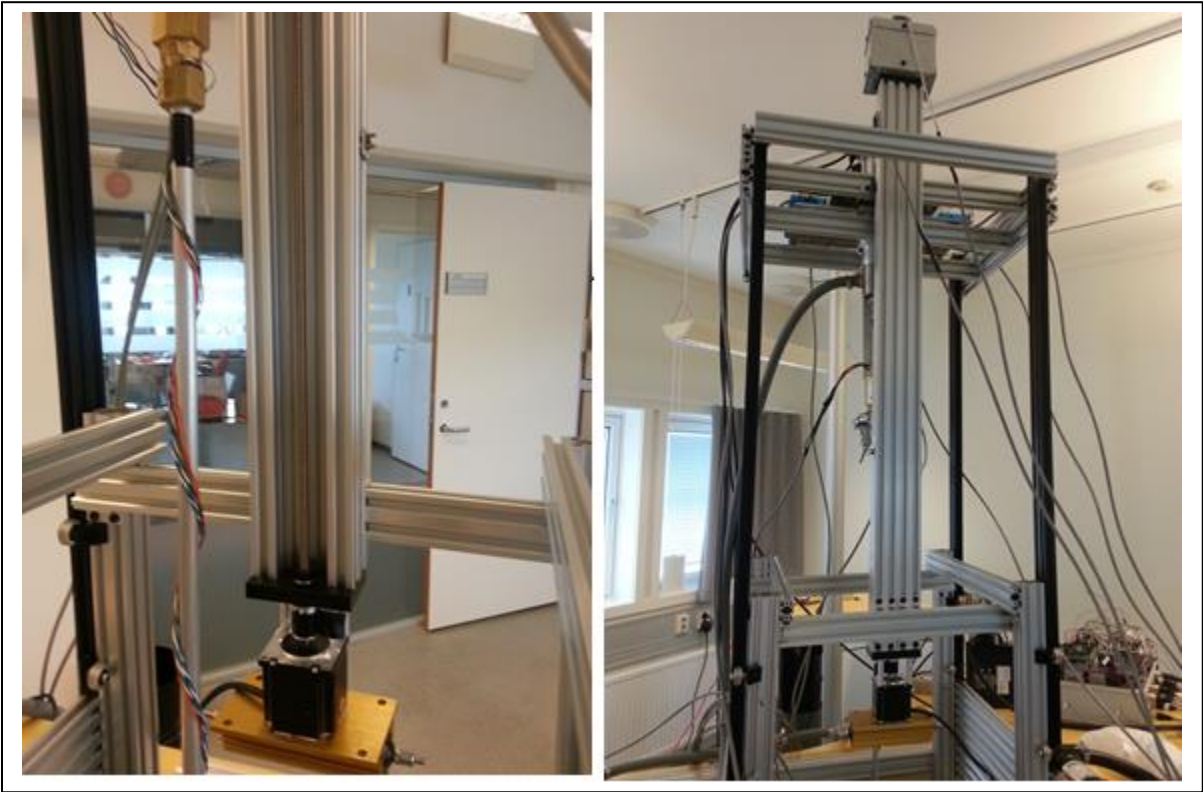
$$\frac{10000}{360^\circ} = 0.036^\circ \rightarrow \frac{0.3^\circ}{0.036^\circ} \approx 8 \rightarrow \text{Max } 180\text{rpm} \rightarrow \frac{180}{60} = 3\text{RPS} \rightarrow \frac{1000000}{30000} \approx 33\mu\text{s}$$

**EQUATION 1 - POWER TRANSMISSION LOOP TIME**

With this in mind, using the calculation below the maximum allowable loop time for the top drive can be derived as follows:

$$33 * 8 \approx 264\mu\text{s} = \text{Max Loop Time}$$

**EQUATION 2 - HOISTING SYSTEM LOOP TIME**



**FIGURE 2 - HOISTING SUBSYSTEM**



For the hoisting system, the loop time was found using the following simple arguments.

The pitch of the lead screw is 2mm. With microstepping there is 2000 steps per rotation. The highest speed movement is probably 10 mm/s, therefore the loop duration should be lower than 100  $\mu$ s.

The final subsystem which is part of the rig is the fluid system. To find the required response time, the following thought process was applied.

The compressibility of water is:  $0.19 \times 10^{-10}$  Pa and the pump will provide about 10l/min. The volume of water in the fluid system at any given time is approximately: 0.5 to 1l. For an instantaneous obstruction, the reaction time for a maximum pressure increase of 0.5bar gives that the max loop duration should be.

$$\left( \frac{0.5e^5}{0.19e^{10}} \right) * \left( \frac{0.5e^{-3}}{10e^{-3}/60} \right) = 78\mu s$$

**EQUATION 3 - FLUID SYSTEM LOOP TIME**

The microcontrollers that monitor each of these subsystems must react if any of these three parameters exceed their critical values.

### 2.1.2 Arduino

Originally the lower level components of the system were to be controlled by an Omron CP1L-EM60DT-D PLC, but as further analysis was made into the required response time of the system it was found that the PLC wouldn't be able to meet the system's needs. This is described in more detail in Chapter 5, Section 5.1.



FIGURE 3 - ARDUINO MEGA

The original PLC was instead replaced by 5 Arduinos. Arduino is an open-source hardware/software solution which was created to give end users, novices and professionals, easy access to readymade microcontrollers which can be used to interact with the environment through sensors and actuators.

Arduino uses a programming language called Processing which is based upon java. Processing has some overhead associated with it, but if this overhead becomes an issue one can also use native C to eliminate some of the overhead and create more optimized code.

### 2.1.3 The line between deterministic and nondeterministic

The system to be build is complex and must be able to react immediately if any of the critical parameters are exceeded. It is therefore important that the system is divided into layers which operate within segments where the algorithms applied are utilized at their maximum potential.

In computer science deterministic and non-deterministic is commonly applied to algorithms, both on their timing and output conditions.

The two categories can be defined as follows:

*A section of the system is said to be deterministic if and only if the section completes it task in a fixed amount of time and always completes it in the same fixed amount of time.*

*A section of the system is said to be non-deterministic if and only if the section completes it task in a variable amount of time and delivers no guarantees as to how much time is spent completing its tasks.*

With the calculated loop time limitations it is clear that the microcontrollers and data collections from the various sensors in the system need to work under a deterministic time regiment, while any computations which fall into the category of non-deterministic should be handled by an external PC.

The command structure for the system is further explained in Chapter 3.

### 2.1.4 Interpolation[15]

As established in chapter three, the system has to be divided in too three layers where one works within a non-deterministic time region and the other two, the microcontroller layers, works within a deterministic time region.

By calculation look up tables and pre calculating formulas which apply for the different sub systems in the system, the microcontrollers can take advantage of interpolation to find values which are not given through the supplied data.

The bigger the datasets passed from the backend layer to the lowest microcontroller layer the better accuracy the system will receive by using interpolation, but this comes with a tradeoff. The microcontrollers have limited storage space and network datagrams also have a limited size.

Ideally the system wants to be able to transmit all the necessary data through one UDP datagram which is 655kBytes.

The system in its current state only applies interpolation in the first dimension also referred to as linear interpolation and does this through this formula.

$$y = y_0 + (y_1 - y_0) * \frac{x - x_0}{x_1 - x_0}$$

**EQUATION 4 - LINEAR INTERPOLATION**

To apply linear interpolation, the system needs two reference points and one value to calculate a new value. In the formula above, that would be  $(x_0, y_0)$  and  $(x_1, y_1)$  as reference points and a value  $x$  which is used to find the value of  $y$ .

One of the subsystems which uses this type of linear interpolation is the fluid system described further in Chapter 4, Section 4.1.

## 2.2 Communication protocols

This sub section briefly describes the communication protocols that were considered for the system with some of their pros and cons.

### 2.2.1 TCP

The Transmission Control Protocol or TCP for short is a communication protocol which is connection-orientated and reliable.

TCP works by establishing a direct connection between sender and receiver; this is done through the use of handshakes, acknowledgements and sequence numbering. No data is passed between the two entities before the handshaking procedure is completed, and if it fails the process has to start over again.

Below is a joke to illustrate how TCP works.

*“Hi, I’d like to hear a TCP joke.”*  
*“Hello, would you like to hear a TCP joke?”*  
*“Yes, I’d like to hear a TCP joke.”*  
*“Ok, I will hear a TCP joke.”*  
*“Are you ready to hear a TCP joke?”*  
*“Yes, I am ready to hear a TCP joke.”*  
*“Ok, I am about to send the TCP joke. It will last for 10 seconds, it has two characters, it does not have a setting, it ends with a punchline.”*  
*“Ok, I am ready to get your TCP joke that will last for 10 seconds, has two characters, does not have an explicit setting, and ends with a punchline.”*  
*“I’m sorry, your connection has timed out... Hello, would you like to hear a TCP joke?”*

*-, Unknown Author*

TCP is a great protocol for applications which need reliable communication, but due to the overhead associated with TCP it was necessary to consider other options.

### 2.2.2 UDP

UDP (User Datagram Protocol) is the opposite of TCP. They are both communication protocols but this is pretty much where their similarity ends. UDP operates on a fire and forget premise.

UDP is a connectionless protocol which doesn’t establish a direct link between sender and receiver. Instead it simply broadcasts the data packet out on to the network and assumes that the data packet arrived at its destination.

Applying this method versus the communication method of TCP makes UDP a much faster protocol. Without providing any guarantees as to its reliability it leaves this responsibility to the application programmers and thus the programmer decides how much overhead to generate through messages.

UDP was chosen as the communication protocol to use between the backend layer and the lower layer sub systems, because the system wouldn’t be affected by the overhead related to TCP.

### 2.2.3 SPI[2,3,4]

Serial Peripheral Interface (SPI) is a synchronous serial data protocol which is used by microcontrollers to communicate with peripheral devices through high speed transmission. SPI is limited by its short range, but for the use in this system where all components are with a maximum two meter distance, the range limitation doesn't apply.

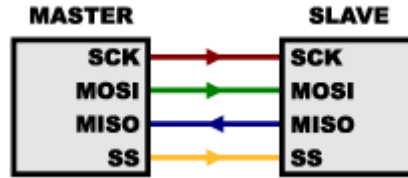


FIGURE 4- SIMPLE SPI SETUP, ONE MASTER, ONE SLAVE[14]

SPI uses four physical connections between the devices, three which are common to all devices:

- Master In Slave Out (MISO) – The slave line for sending data to master.
- Master Out Slave In (MOSI) – The master line for sending data to the peripherals.
- Serial Clock (SCK) – The clock pulse which synchronize data transmission generated by the master

And one which specifies which device to talk to:

- Slave Select (SS) – pin on each device that the master can toggle to enable and disabled the specific device.

By setting the state of the SS pin on the microcontroller to low, it enables the device to communicate with the master device. When setting the SS state to high, it ignores any messages from the master and thus allows the system to have multiple SPI devices connected to the master. On the Arduino it is possible to connect as many SPI devices as there are free digital outputs, this being the only limiting factor in terms of connectable devices.

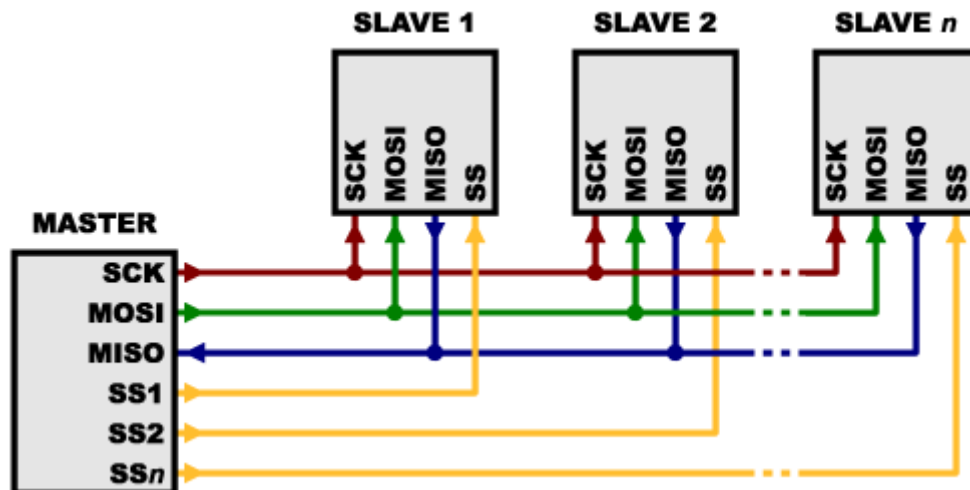


FIGURE 5 - ONE MASTER, MULTIPLE SLAVES[14]

The figure above illustrates how the process of interfacing multiple slave devices with a master device is done.

Due to the rapid transmission speed delivered by SPI, it was the first choice for intercommunication between the microcontrollers and peripheral devices. But due to instability's and a lack of compatibility with the Ethernet Shield<sup>2</sup> used on a few of the microcontrollers, it was decided that another communication protocol should be chosen for the communication between microcontrollers.

### 2.2.4 I<sup>2</sup>C[5]

I square C or I two C (I<sup>2</sup>C or I2C) is a protocol originally developed in 1982 by Philips for use in their chips. In contrast to SPI, I2C uses a two wire setup where the wire are as follows:

- Serial Data Line (SDA) – Which the data being transmitted travels through.
- Serial Clock Line (SCL) – Clock line which syncs the read/write operation

I2C works by giving an address to each device connected on the I2C bus. I2C offers an address space of 10 bits, which in turn means that the system could theoretically support 1024 devices<sup>3</sup>. There are of course physical restrictions such as bus space and like SPI, I2C is designed to work within short distances so this could become an issue if one were to try and connect two devices which aren't close together.

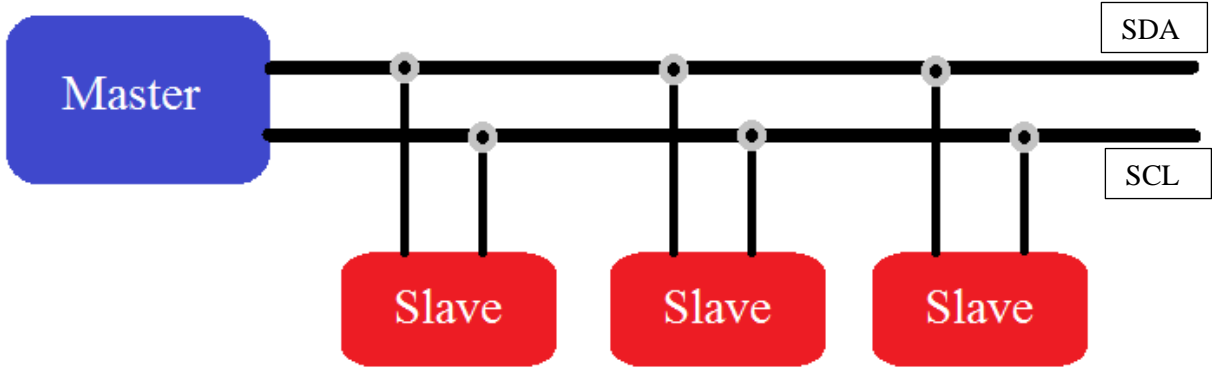


FIGURE 6 - I2C SETUP, ONE MASTER, MULTIPLE SLAVES

The figure above illustrates how an I2C setup using one master would look.

Because of the compatibility issues encountered when using SPI, I2C was tested and proved to show no such compatibility issues. A combination of SPI and I2C was there for selected for the communication in the two microcontroller layers (see chapter three). Where SPI is used to interact with certain peripheral devices and I2C is used for intercommunication between the microcontrollers.

<sup>2</sup> The Arduino Ethernet shield uses SPI to transfer data collected from Ethernet to the Arduino

<sup>3</sup> In reality one wouldn't be able to connect 1024 devices, because a certain number of addresses are reserved for special purposes. This however wasn't a problem for the system in question as it would never get close to the maximum address limit.

## 2.2 Control technique[6]

The control technique selected to handle the microcontrollers in the system was a finite state machine. This is a fairly common control technique when dealing with microcontrollers and is a proven method which yield high success for most situations and problems.

A finite state automata is a mathematical model of a system with discrete inputs and outputs. The system can be in one of a finite number of states at any time. The past inputs given to the system determines the behavior of the system on the next subsequent input.

The chosen implementation of this system is a deterministic finite state automata (DFSA), the fact that it is deterministic refers to the uniqueness of the computations. When applying this method to formulate a model of the system and defining the different states which are available, a number of tools become available such as state diagrams and state transition tables.

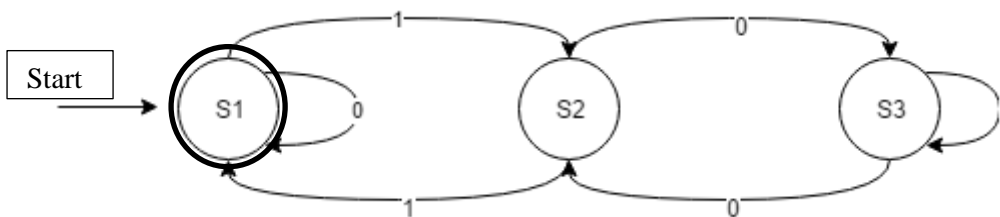


FIGURE 7 - DFA WITH 3 STATES

The figure above illustrates a simple DFA. State One is both the start state and the accept state for the input in this case.

<i>States</i>	<i>Input</i>	
	0	1
<i>S1</i>	S1	S2
<i>S2</i>	S3	S1
<i>S3</i>	S2	S3

TABLE 2 - STATE TRANSITION TABLE FOR FIGURE 7

Through the use of this type of state machine, the system will be able to model each subsystem and their corresponding states with inputs and outputs leading the desired states which will be illustrated later on in the thesis.

## 2.3 Safety envelopes

For the system to operate under “normal” drilling conditions, the system needs to define a state which is considered normal. Such a state should contain boundary conditions for each subsystem where sensor data is collected and these boundary conditions usually take the form of a minimum and maximum value.

Each of the systems subsystems are charged with monitoring their specific part of the system and for each of these subsystems the system needs to set in place safety envelopes related to their task in the system.

By calculating the breaking points of the physical components in the system and then applying a 10-25% adjustment to these limits, the systems safety envelopes were derived. As long as the sensor data falls within the established safety envelopes the system can proceed in its normal operating state.

## **2.4 Safety triggers**

As described in the previous section, the system needs to take action if it senses that the system is about to break any of the established thresholds in the system. Hence the system needs safety triggers which start a series of events to bring the system back to safe operating conditions.

These triggers should fire if the data collected from the systems sensors indicate that the current system settings are eventually going to break any of the safety thresholds currently in place in the system.

An example of such a safety trigger would be in the case of a stick slip situation, see chapter 8.1, where the drill bit gets stuck in the rock and as the motor applies more torque and energy builds up in the drill string, the bit will eventually skip as the energy level hits a peak and then repeat. If such a situation arises a safety trigger should fire and start an event where the system stops rotating the drill string and tells the hoisting system to lift off from the bottom of the borehole. Thereby releasing the built up energy in the system and resetting the drill parameters to a defined state of unknown rock formation.

This state assumes a worst case scenario and will restart the drilling and attempt to find new optimal drilling parameters.

This is one example of the safety trigger that should be implemented into the system.

## **2.5 Calibration**

With the system built and in its pre calibrated state, it won't be very efficient at its task. The system needs to be calibrated manually to a state where the system can be initialized and calibrate itself automatically.

During this initial state, the system should run a number of calibrating procedures for the system to collect baseline values for each sub system, as described further in Chapter 7.

With these calibration procedures completed, the system is ready to move in to the drilling state.

## **2.6 Diagnostic**

For the autonomous system to be in a state of completion it must be able to diagnose itself if it encounters errors and act accordingly. As the system is operating in normal state and monitoring the sensor data collect it should be able to make decisions which is in its best interest. These diagnostics are assisted by the safety triggers which will attempt to bring the system from a brink state and back to a stable state.

The system has limited possibilities when it comes to correcting errors detected, and if the established safety mechanisms fail, the system should shut itself down to prevent damage to equipment attached to it.

Through the diagnostic system it should be able to pinpoint the area which is in an erroneous state and communicate this to the backend software, in such a manner that the affected area can easily be found and the appropriate maintenance carried out.

The highest safety priority of the system should be that it should shut down to prevent equipment damage.



## 2.7 Optimization

For the system to operate at its peak performance level, the system should be able to optimize the drilling parameters to achieve the highest possible ROP while maintaining the verticality of the borehole.

For this to be achievable the system needs to analyse the data collected from the various sensors in the system and attempt to make an educated guess as to the strength of the current rock formation. The idea behind the optimization scheme in place is that when the system starts to drill the initial parameter settings are set to a worst case scenario which corresponds to the hardest formation rock placed inside the rock sample to be drilled.

As the system attempts to drill, and more sensor data becomes available the system should adjust the WOB and RPM to find a sweet spot where speed is maximized and all systems fall within safe thresholds. At this stage the system should claim the rock formation as known and store the drilling parameters along with its corresponding mechanical specific energy (MSE) value.

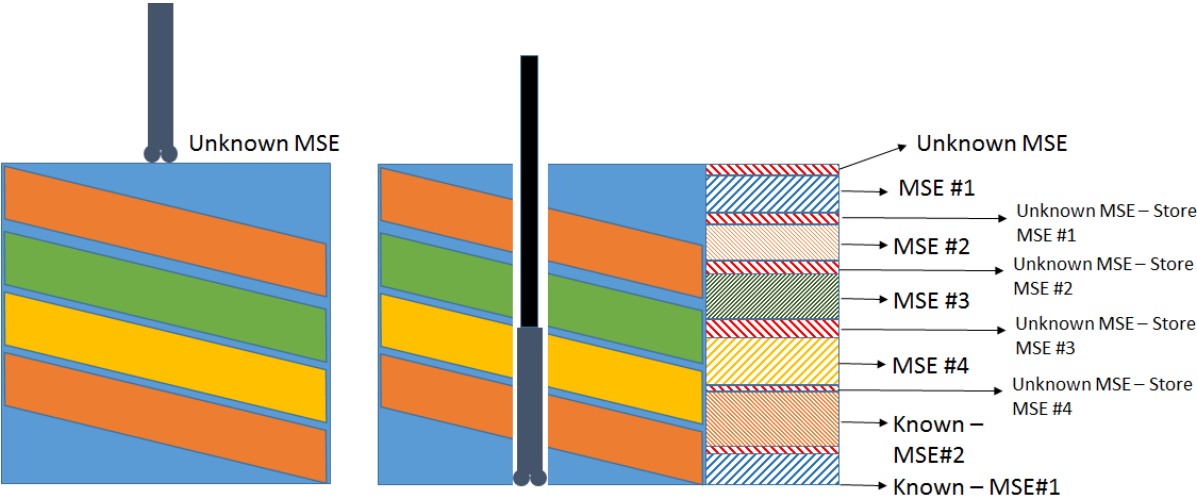


FIGURE 8 - OPTIMIZATION BY RECOGNISING MSE

The figure above illustrates the optimization method planned for the system. As seen in the figure the system starts by assuming an unknown rock formation strength and starts experimenting with the drilling parameters until it obtains a maximum stable ROP value. Once this maximum is found the system moves into a known MSE state sticks to these settings until the sensor data changes indicating a change in rock formation. The system then stores these drill parameters along with its MSE and moves back into the unknown MSE state and repeats the procedure.

### 3. Software architecture for real-time management of a complex system

The software architecture for the system is based on the hierarchical control scheme, which is a common engineering technique for decomposing complex systems.

Each element in a hierarchical control system can be viewed as a linked node in a tree structure, where commands and task flow down from superior nodes down to subordinate nodes. Results and sensor readings flow upwards from the bottom nodes to their superior nodes. The two distinguishing features of this type of systems are:

- Each higher layer of the tree operates with a longer planning and execution interval than its immediate lower layer.
- The lower layers have local tasks and their activities are planned and coordinated by the higher layers.

The final system was decomposed and configured hierarchically, below is a figure showing an overview of the complete system.

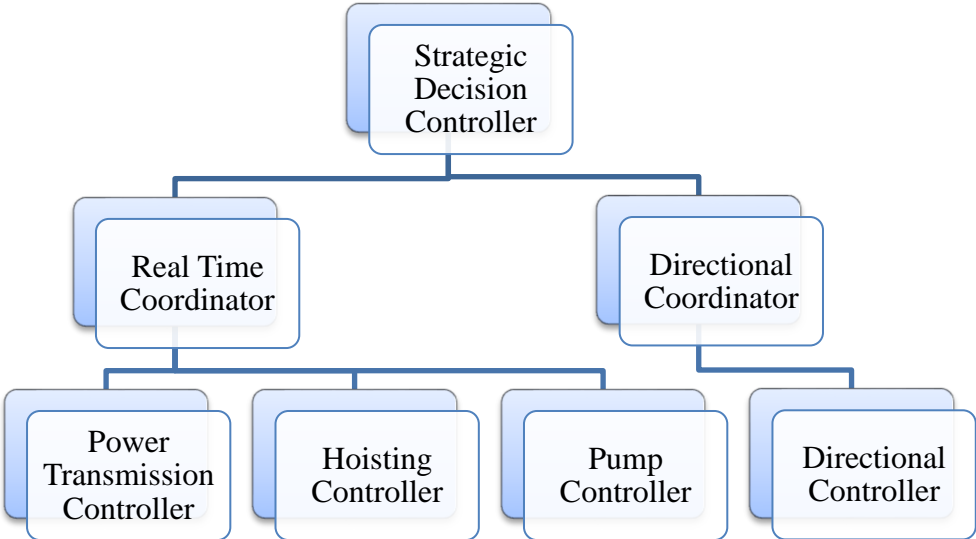


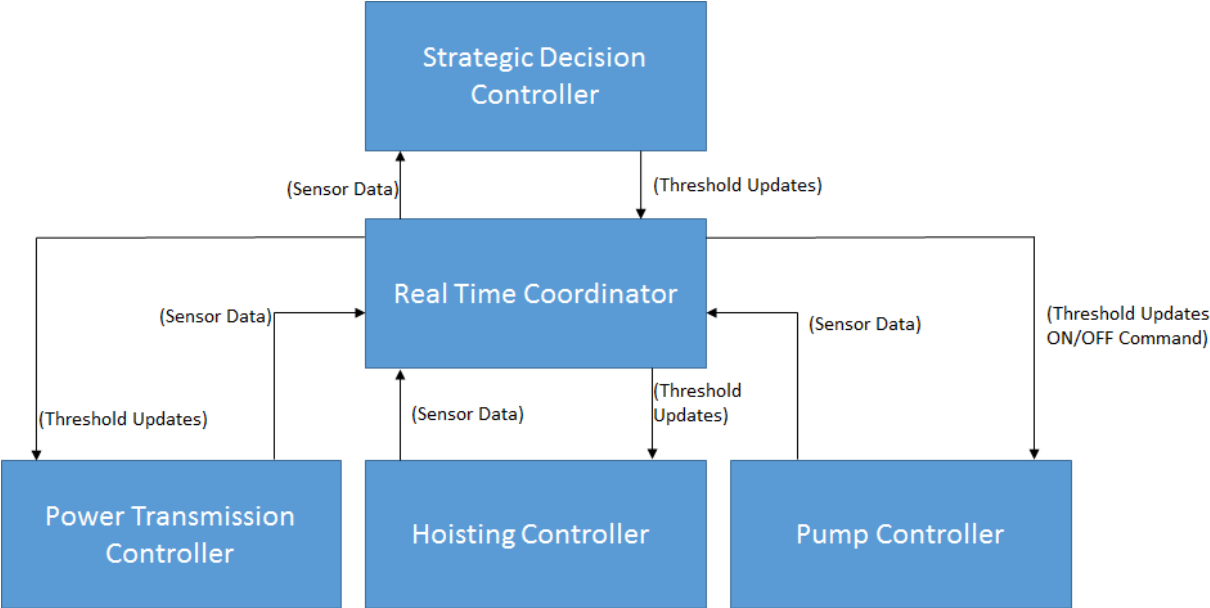
FIGURE 9 - HIERARCHICAL STRUCTURE OF THE SYSTEM

The highest layer contains a computer which handles the visualization of data transmitted from the lowest level nodes in the system. It processes this data through algorithms, which aren't deterministic and it therefore has no guarantee on the finishing time associated with it. An example of these algorithms would be the generation of lookup tables for the system torque which is then passed downwards in the system to the correct node.

The middle layer consists of a Real Time Coordinator (RTC) and a directional coordinator, which handles the communication between the lowest layer controllers and SDC. These coordinator can be viewed as information hubs, their only task is to forward information between the upper and lower layers of the system.

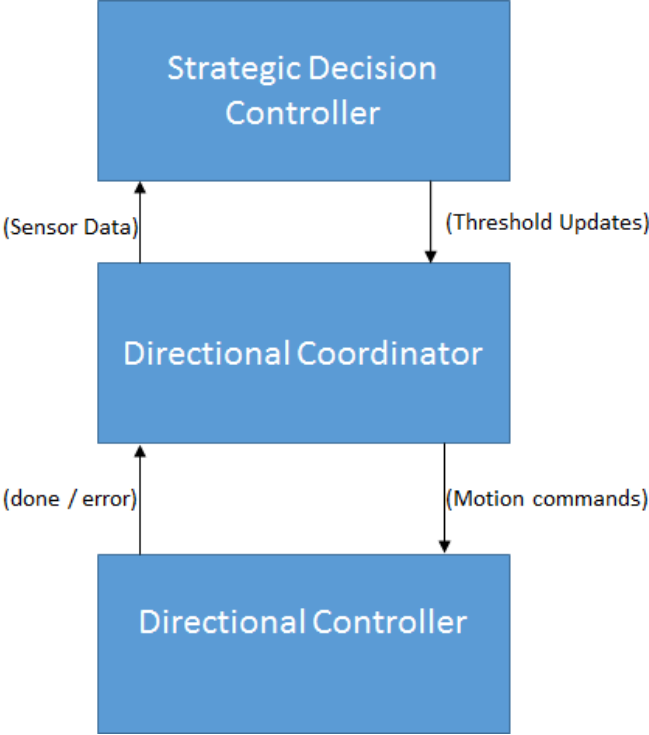
Finally the bottom layer in the system consists of the controllers which collect data from the various sensors in the system, the data is then transmitted through high speed connections between the microcontrollers, and controllers the actuators in the system.

As will be described in Chapter 5, the two lower layers are divided into two physical domains as indicated in the systems hierarchical structure. By further decomposing each branch in the system shown in figure 9, the branches can be formulated into the two control hierarchies shown below.



**FIGURE 10 - HIERARCHICAL STRUCTURE WITHIN CONTROL BOX ONE(MCB)**

The figure above illustrates the setup for the upper rig controller and gives an indication as to which type of data is being transmitted between the different layers and between the different nodes in the system. Similarly a figure for the directional subsystem is shown below.



**FIGURE 11 - HIERARCHICAL CONTROL STRUCTURE WITHIN CONTROL BOX TWO(DCB)**

## 4. Safety Envelops

With the limited amount of time available to create an initial design for the system, the team didn't have time to refine the calculation presented and therefore one of the first tasks at hand was to analyse and verify that the rough estimates were indeed correct. This chapter gives a brief introduction to the reanalysis work which was done. It also covers the software implementation which was done to speed up the work of verifying the pump calculations.

### 4.1 Circulation system

The main by-product from drilling is rock fragments created as the system grinds out the borehole. These fragments are referred to as cuttings and need to be cleared out of the borehole to prevent the drill getting stuck. To handle this the system was designed with a fluid system which handles the previously mentioned issue as well as cooling down the PDC bit during drilling.

This subsystem consists of six components;

- Pump
- Hose
- Swivel
- Aluminium drill-pipe
- Bottom Half Assembly (BHA)
- PDC Bit

The figure below shows an overview of how the components are connected in the physical prototype. Besides the components listed above there are two additional factors which affect the system, namely hydrostatic pressure and the annulus created through drilling. Common to all the components is that they generated pressure loss in the system, the next section describes how these pressure losses are calculated and what requirements the selected pump had to satisfy.

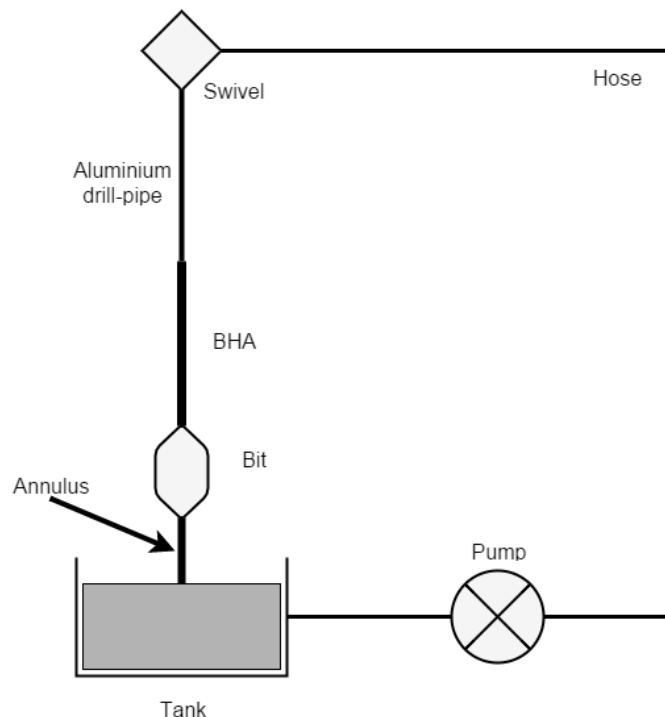


FIGURE 12 - FLUID SYSTEM OVERVIEW

#### 4.1.1 Pressure loss formulas[7]

The simulation software applies the Darcy-Weisbach equation with the application of Darcy's friction factor to determine the pressure loss associated most of the components in the system. The formula for this equation is given below.

$$\frac{\Delta p}{L} = f_D * \frac{\rho}{2} * \frac{v_{flow}^2}{d_i} \rightarrow \Delta p = \frac{f_D * \rho * L * v_{flow}^2}{2 * d_i}$$

EQUATION 5 - THE DARCY-WEISBACH EQUATION

Where  $\Delta p$  is the pressure loss as a function of:

- The fluid density  $\rho$  (kg/m<sup>3</sup>)
- The length of the component  $L$  (m)
- The hydraulic diameter<sup>4</sup>  $d_i$  of the component
- Flow velocity  $v_{flow}$ , measured as the volumetric flow rate  $Q$  per unit cross-sectional wetted area (m/s)
- Darcy's friction factor,  $f_D$

The software uses this formula to describe the pressure loss in all components in the system with the exception of the swivel, bit and hydrostatic pressure loss. Which have their own pressure loss formulas, described later in this chapter.

#### *Darcy friction factor & Reynolds number[8]*

The Darcy-Weisbach equation makes use of the Darcy friction factor which is a friction factor that depends on various characteristics, such as the inner diameter of the pipe, characteristics of the fluid in the system and the velocity of that fluid. Calculating the Reynolds number[6] with this information the friction factor can easily be deduced.

The Darcy friction factor is defined as follows:

$$f_D = \frac{64}{Re}, \text{ where } Re \text{ is the Reynolds number defined as } \Rightarrow Re = \frac{\rho * v_{flow} * d_i}{\mu}$$

EQUATION 6 - DARCY FRICTION FACTOR, REYNOLDS NUMBER EQUATION

The Reynolds number can fall into three different categories of flow, and these categories are as follows; Laminar, Critical and Turbulent. These categories are defined as;

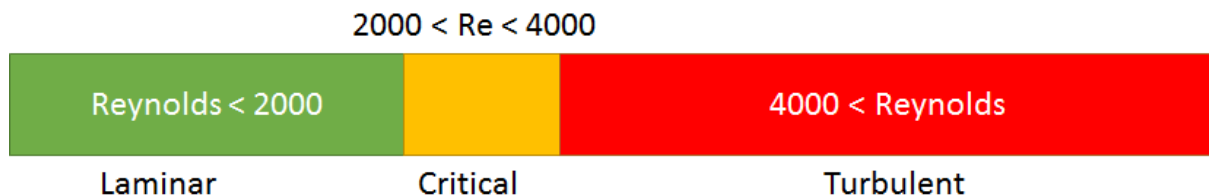


FIGURE 13 - LIMITS BETWEEN LAMINAR, CRITICAL AND TURBULENT FLOW

<sup>4</sup> The hydraulic diameter is defined by  $D_H=4A/P$  where  $A$  is the cross sectional area and  $P$  is the wetted perimeter of the cross-section. In our case when dealing with a circular pipe/tube the hydraulic diameter is equal to the inner diameter.

The calculation of the Darcy's friction factor depends on which segment the Reynolds number falls.

If the Reynolds number falls within the laminar segment the friction factor is calculated by;

$$f_D = \frac{64}{Re}$$

**EQUATION 7 - LAMINAR FLOW EQUATION**

If the calculated Reynolds number falls in to the turbulent segment, the friction factor is obtained through the use of the Colebrook-White equation [9]. This equation can be used to iteratively find the solution for the Darcy's friction factor in turbulent flow. The equation is expressed as:

$$\frac{1}{\sqrt{f_D}} = -2 \log_{10} \left( \frac{\varepsilon}{3.7 D_i} + \frac{2.51}{Re \sqrt{f_D}} \right)$$

**EQUATION 8 - ITERATIV SOLUTION, TURBULENT FLOW**

Where:

- $f_D$  is the Darcy friction factor
- $\varepsilon$  is the roughness coefficient
- $D_i$  is the inner diameter
- $Re$  is the Reynolds number

The simulation software applies the first equation using the inner diameter.

And finally if the Reynolds number falls within the critical segment, the friction factor is found by first calculating the laminar friction factor and then calculating the turbulent equivalent. With these two values established, one interpolates between the two to achieve the correct value.

### ***Swivel[10]***

The pressure loss equation for the swivel is modelled by the equation:

$$\Delta p = \left( \frac{Q}{C_v} \right)^2$$

**EQUATION 9 - PRESSURE LOSS EQUATION SWIVEL**

This was obtained through the datasheets swivel [10], where:

- $Q$  is the flow rate in gallons per minute of water
- $C_v$  is the flow factor in gallons of water per minute with a 1 psi pressure drop. In the case of the systems ½" swivel  $C_v \approx 5.25$
- $\Delta p$  is the pressure drop in PSI

### ***Hydrostatic***

The pressure loss equation for the hydrostatic head loss is given as;

$$\Delta p = \rho * g * h$$

**EQUATION 10 - HYDROSTATIC PRESSURE LOSS EQUATION**

Where  $g$  is the gravitational acceleration and  $h$  is the height at which the system is positioned.

When the system is at its highest peak the hydrostatic pressure loss will also be at its highest, and as the rig descends and drills deeper into the rock the hydrostatic pressure loss will reduce proportionally to the increase in pressure loss in the annulus.

**Bit**

The pressure loss equation for the loss through the bit is given as:

$$\Delta p = \frac{\rho Q^2}{1.975 * C^2 * n * \left(\frac{\pi * d_i^2}{4}\right)^2}$$

**EQUATION 11 - PRESSURE LOSS EQUATION BIT**

This formula was obtained by the chief scientist at IRIS through this datasheet [11].

- ρ, The fluid density (kg/m<sup>3</sup>)
- Q, the flow rate in gallons per minute
- 1.975, a silly constant
- C, the characteristics of the nozzle on the PDC bit
- n, number of nozzles on the bit
- d<sub>i</sub>, The inner diameter of the bit nozzle

**4.1.2 Component characteristics**

The table below list the initial assumptions about the component characteristics.

	Annulus	Alum. Pipe	BHA	Bit	Hose
Inner dia(mm)	22	8.7	8.7	5**	12.7
Outer dia(mm)	27.5		22	27.5	
Length(mm)	0*	914	450		2500
Roughness(mm)	0.15	0.03	0.15		0.03

**TABLE 3 - COMPONENT CHARACTERISTICS**

\*) Annulus length changes based on borehole depth

\*\*\*) The bit uses nozzle diameter instead of inner diameter

As can be seen from the table above, not all the fields in the table are filled. This is because they're of no interest for the calculations.

	Annulus	Alum. Pipe	BHA	Bit	Hose	Hydrostatic	Swivel	TOTAL
Pressure Loss(Bar)	>0.05*	0.1	0.11	0.7	0.04	0.2	0.12	1.27

**TABLE 4 - INITIAL PRESSURE LOSS CALCULATIONS**

The table represents pressure loss calculated in the initial design proposal (See appendix A).

### 4.1.3 Calculation results based on software

With the necessary formulas and characteristics established, everything was implemented into the systems simulation software to verify the earlier findings.

For the full software implementation take a look into Appendix B.

*All tables below shows that the pressure loss in the annulus is 0 bar. This is because the simulation assumes that the rig is at its peak position of two meter above ground. As the rig descends and starts drilling the hydrostatic pressure will decrease and the pressure in the annulus will increase proportionally.*

	Annulus	Alum. Pipe	BHA	Bit	Hose	Hydrostatic	Swivel	TOTAL	Flowrate in the system	Flow Velocity in the annulus
Pressure Loss(Bar)	0	0.122	0.431	0.497	0.05	0.196	0.022	1.516	9.389	0.47 m/s

TABLE 5 - PRESSURE LOSS CALCULATION THOUGH SIM. SOFTWARE

The table above shows the calculations extracted from the simulation software which indicate that a number of the initial calculations were incorrect.

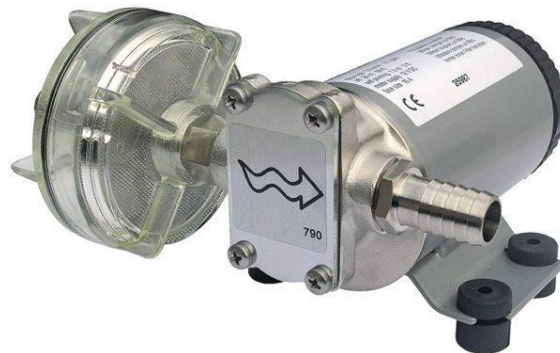


FIGURE 14 - WATER/DIESEL PUMP 12 V (BILTEMA 25987)

The original pump selected for the system was a water/diesel pump from Biltema, which should be able to provide about 9 l/min at the initial calculated pressure. This pump however turned out to be too weak as errors were made in the initial calculations show in table 4 above and the correct values are displayed in table 5. With these findings, it became clear that the pump selected for the system wouldn't be able to deliver the necessary flow rate to transport cuttings out of the annulus while drilling<sup>5</sup>.

<sup>5</sup> The necessary flow rate to transport cuttings out of the annulus has to be approximately 0.6m/s. This number was calculated by the scientists at IRIS and by using this formula the system was able to check the flow velocity in the annulus;  $Velocity = \frac{Q}{60000} * \frac{1}{OD_{bit}^2 - OD_{BHA}^2}$ .



This meant that a new pump had to be selected, but in the interest of cost a solution involving the connection of two pumps in series was proposed. With the simulation software already implemented, making the minor changes necessary to reflect this setup was fairly straightforward. This solution yielded the following results:

	Annulus	Alum. Pipe	BHA	Bit	Hose	Hydrostatic	Swivel	TOTAL	Flowrate in the system	Flow Velocity in the annulus
Pressure Loss(Bar)	0	0.081	0.28	0.321	0.034	0.196	0.014	0,926	10.759	0.53 m/s

TABLE 6 - PRESSURE LOSS CALCULATION THROUGH SIM. SOFTWARE FOR TWO IDENTICAL PUMPS IN SERIES

Unfortunately this solution did not yield the desired effect and it was instead decided to replace the water/diesel pump with a more powerful pump.



FIGURE 15 - ITT FLOJET DIAPHRAGM ELECTRIC POSITIVE DISPLACEMENT PUMP, 19L/MIN, 3.1 BAR, 12 V DC

The team chose to search online for a pump which would satisfy the necessary requirements. The pump shown in the figure above was finally selected. This pump delivers 19l/min.

	Annulus	Alum. Pipe	BHA	Bit	Hose	Hydrostatic	Swivel	TOTAL	Flowrate in the system	Flow Velocity in the annulus
Pressure Loss(Bar)	0	0.216	0.782	0.906	0.088	0.196	0.039	2,031	12.948	0.65 m/s

TABLE 7 - PRESSURE LOSS CALCULATION SIM. SOFTWARE SELECTED PUMP

The table above shows that the pump delivers just over the requirements specified by the system, and through calibration it was found that these values were very accurate.

#### 4.1.4 Stochastic simulation

Due to uncertainties in the characteristics of the physical components in the system, the software was extended to perform stochastic simulation, varying the different attributes of the components to get a better indication of what the actual pressure in the system would be.

The tables below show the stochastic results for each of the proposed pump solutions for the system.

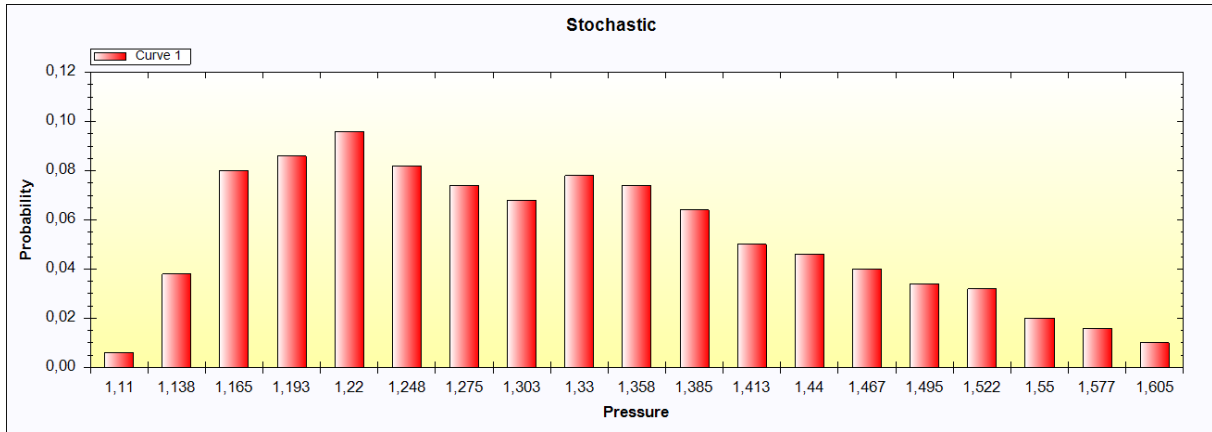


FIGURE 16 - STOCHASTIC RESULT OF THE BILTEMA PUMP

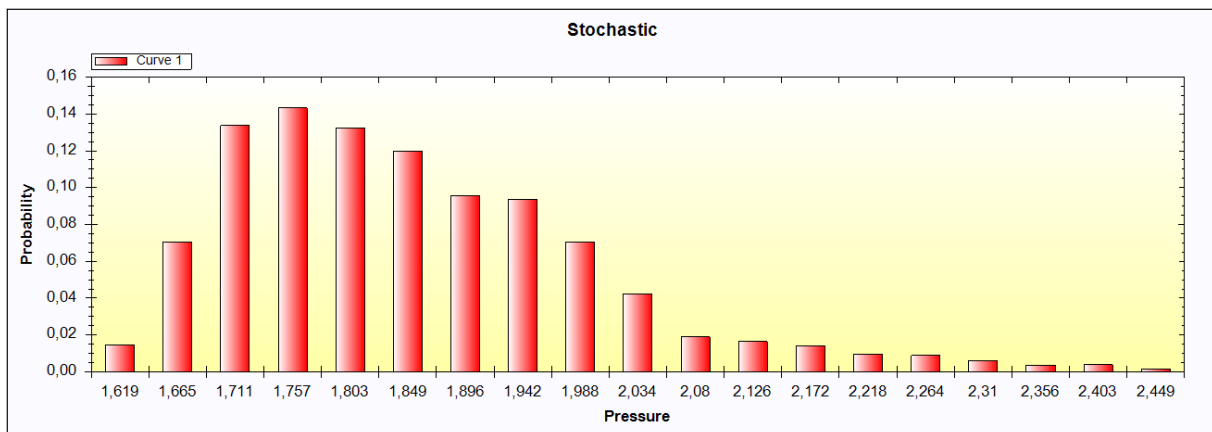


FIGURE 17 - STOCHASTIC RESULT OF TWO IDENTICAL PUMPS IN SERIES(BILTEMA PUMP USED)

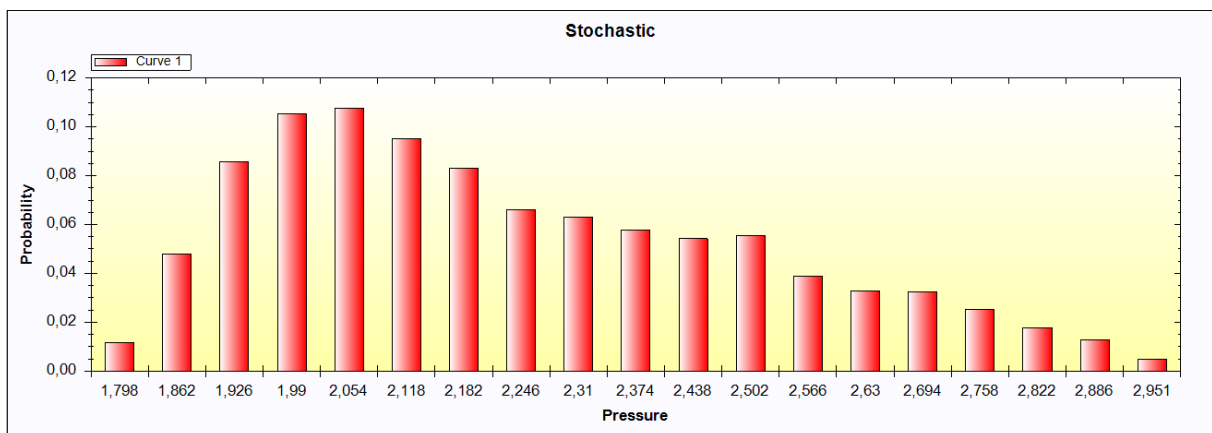


FIGURE 18 - STOCHASTIC RESULT, SELECTED ITT PUMP

## 4.2 Power Transmission – Rotary control

The system contains a power transmission subsystem which controls the rotation of the drill string and the RPM of the bit. The system should be able to adjust and respond to changes in torque while operating in the drilling state.

The drill string is made up of several components connected together to form the complete drill string attached to the top drive step motor.

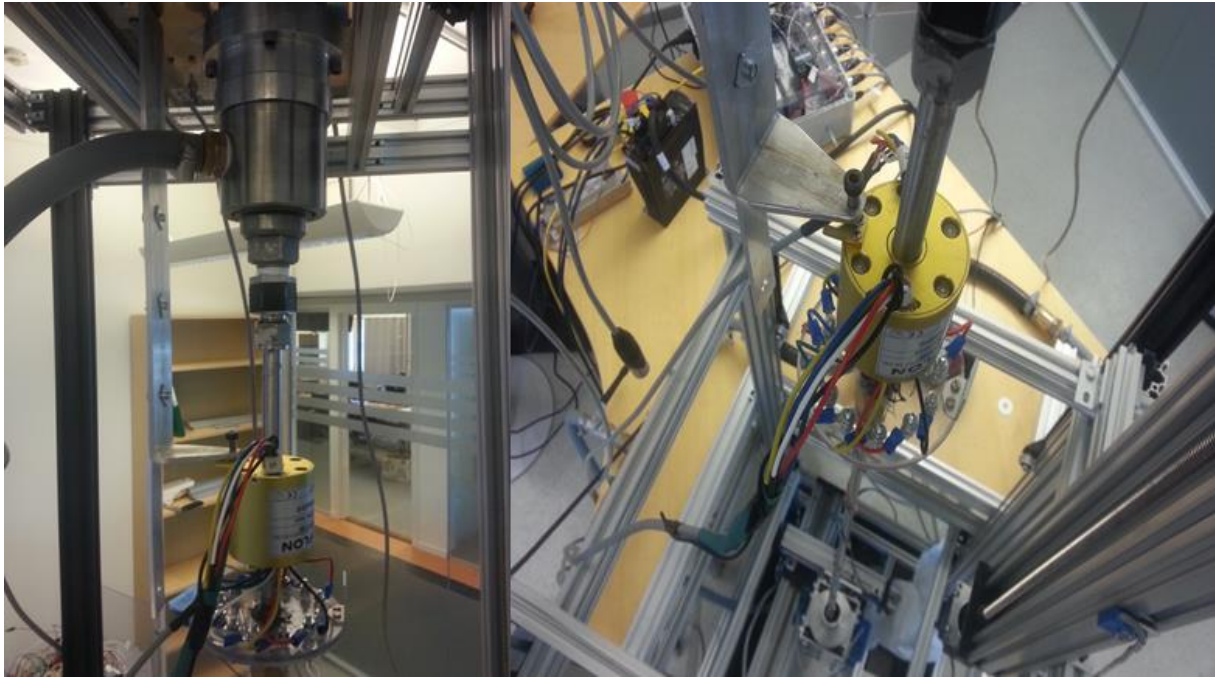


FIGURE 19 - PHYSICAL DRILL STRING

As mentioned in chapter 2.1, the primary limitation of the power transmission system is the aluminium drill pipe. This transient model should translate the torque readings from the sensors and predict the angle of twist of the components in the drill string. These predictions should be made available to the control unit of the top drive in form of a lookup table which it can be used to check the sensor readings.

The table below indicates what the lookup table should look like.

	$\tau_0$	$\tau_1$	$\tau_n$
$\theta_0$	Value	Value	Value
$\theta_1$	Value	Value	Value
$\theta_n$	Value	Value	Value

TABLE 8 - LOOKUP TABLE TORQUE

*The top row of the table denotes torque values which the system should use to check readings against and find the closes possible value.*

*The first column indicates the component in the drill string. The cells contain the predicted twist angle of the component.*

When the system reads torque values from the sensor, it will attempt to find the closest possible index which corresponds to the observed value in the system. If there are no matching indexes it should pick the two indexes which the observed value falls within and interpolate between them.

When the system finds a value which matches the observed reading, it should proceed to the check primarily the twist angle of the drill string. If the value falls within the safety threshold, the system can continue to drill. If it's predicted to breach the threshold the system should engage one of the safety mechanisms to avoid damaging the mechanical components in the system.

To establish the transient model required by the system the procedure detailed in Appendix C.

### 5. Cybernetic work that had to be done

In the team there should have been a cybernetics student, but the group was established without this technical competence. This meant that another student in the team had to do the work which should have been assigned to this person.

The only member in the team with some experience creating circuitry boards and planning electrical setups was me. Therefore in addition to developing the software system I also had to do the electrical engineering work required by the system.

This work consisted of;

- Planning the electrical setup
- Signal analyses
- Physical implementation of the system
- Component calibration

Unfortunately this work was very time consuming and the software development was delayed.



FIGURE 20 - COMPONENTS FOR THE ELECTRICAL SETUP

### 5.1 Initial versus finalized design

The electrical setup is the system which has undergone the largest amount of change from start to the end product. Starting out the system consisted of only one PLC controlling all aspects of the system. Further analysis of the problem discovered that the PLC’s loop time was far too slow to be able to react to changes in the system.

The fastest loop time the PLC could realistically achieve was in the range of 5-10ms. As a PLC’s loop time is dependent on the amount of code in the PLC show in the able below, using this approach the code written would have to be highly optimized for the system to achieve a respectable reaction time. The table below shows the correlation between number of instructions versus the loop time.

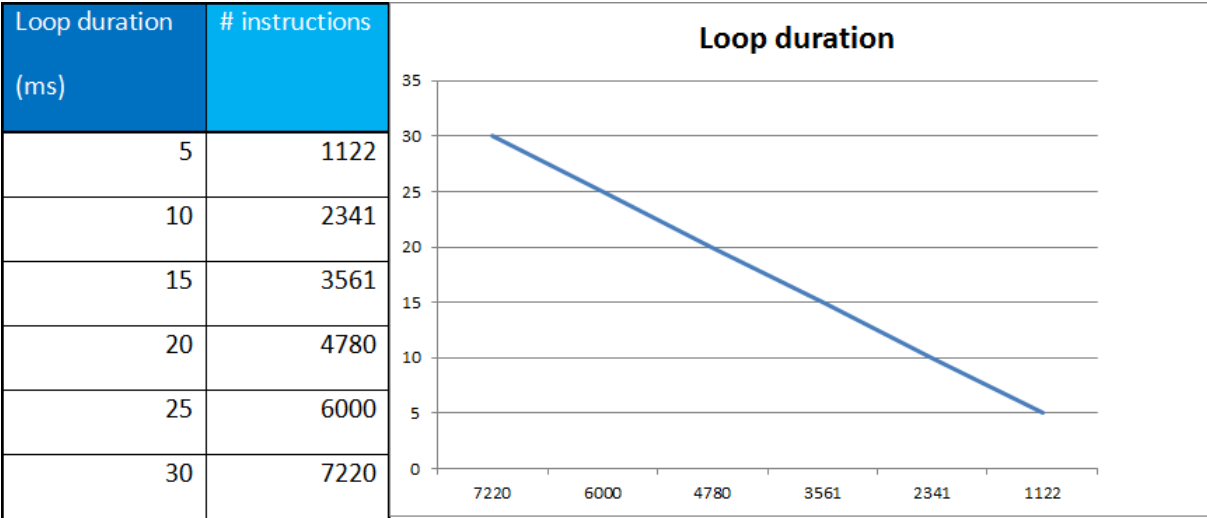


FIGURE 21 - TABLE AND GRAPH DISPLAYING LOOP DURATION VS INSRUCTIONS

In chapter 2.1 the necessary responds time were calculated. It was found that the necessary loop times for the subsystems were; 264, 100, 78 microseconds for the hoisting, power transmission and pump subsystems respectably. Seeing as the team didn’t get to compete in the official Drillbotics competition, the team moved away from the industrial PLC and looked at cheap open-source solutions which could provide the necessary loop times.

This lead to a setup using multiple Arduino Mega’s which has a processor clock speed of 16MHz. By dividing the number of instructions which would have resided within a single PLC into five different Arduinos, the number of instructions available increased dramatically.

However after experimenting with the readymade amplifier circuits bought with the load cells used in the system it became evident that the analog-to-digital converter (ADC) of these board could not support the sampling speed required by the hoisting and power transmission systems. Considering building custom amplifiers (Which will be covered later in this chapter), it was found that the even with these the ADC of the Mega wouldn’t be able to deliver the speed necessary for the system to operate under the required system specifications.

Thus the Arduino Due was procured which has an 84 MHz processor and has an ADC which can perform its task in 1-2µs. This combined with its processing power meant that even more instructions could be fitted within the microcontroller. The only drawback with the Due, is that it works on a different transistor-transistor-logic (TTL) then the Mega, namely 3.3V versus 5V.

The original idea was to use the Mega's in combination with the Due's, where the Mega's would act as coordinators and the Due's would act as the controllers. But because of the problem that the different TTL logic created, it was decided to use the Mega's in one control box and the Due's in the other. This circumvented the TTL problem as the boards had no direct communication.

By using the slow Mega boards in the directional control box and the Dues in the main control box the interaction would only be through the SDC if necessary. And thus the following system setup was created.

### 5.2 Overview

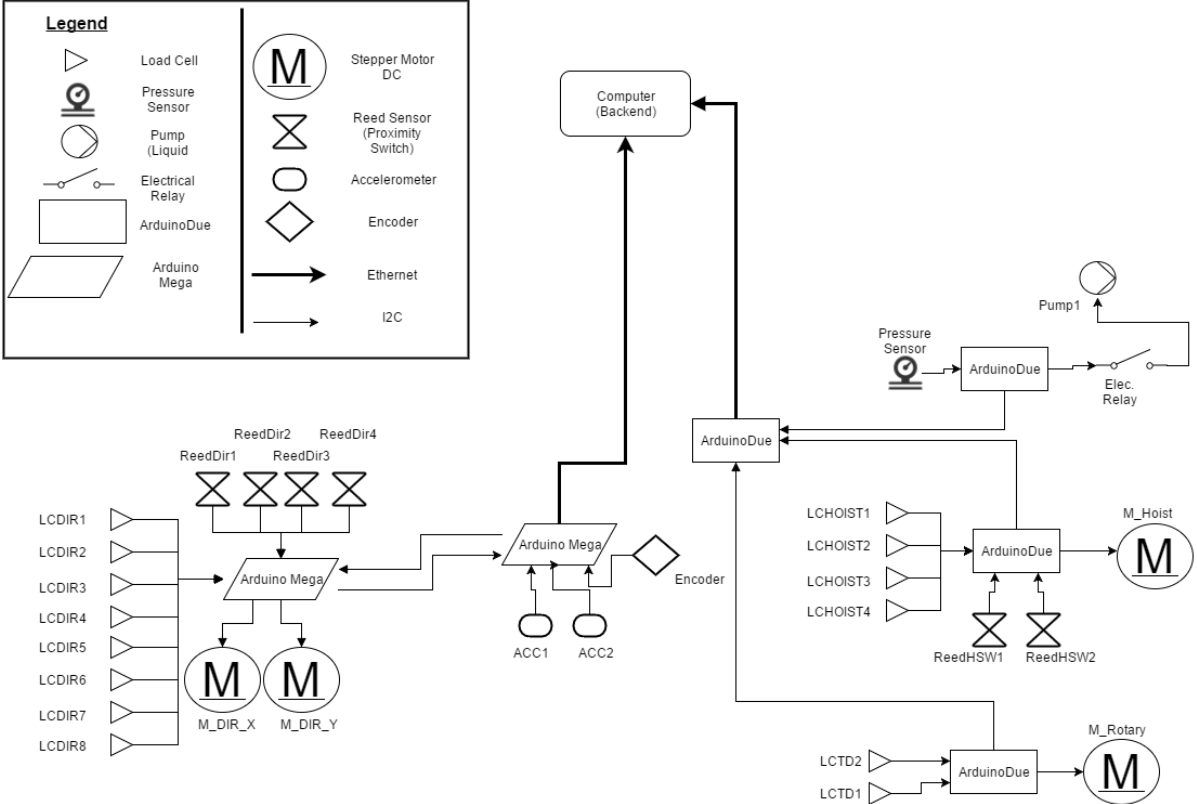


FIGURE 22 - ELECTRICAL SETUP OVERVIEW

The above illustrates the basics electrical setup of the system. The system consists of:

- 4x Arduino Due
- 2x Arduino Mega
- 14x Load cells
- 6x Reed Switches
- 2x Accelerometers
- 1x Encoder

The Arduino's control the motors and pull sensor readings from the various sensors in the system list above.

These sensors are placed in the rig to measure the different aspects such as torque, hook load, vibration and to simply stop the actuators from going out of bounds.

The hardware is split into two control boxes which, is described in the next two sections of the thesis.

### 5.3 Main Control box(MCB)

The MCB contains the most essential subsystems of the prototype. These subsystems are the power transmission system, the hoisting system and the fluid system.

Without these the rig would not be able to perform its primary task which is drilling through the rock.

The MCB contains the following components:

Component	#
Arduino Due	3
Arduino Ethernet Shield	1
Solid state relay	2
24V Power supply	1
12V Power supply	1
+/- 12V Power supply	1
Heat sink	1
12V Fan	1
XLR4 Connector	7
XLR3 Connector	4
XLR5 Connector	3
OptoCoupler	4
RJ45 Connector	1
USB Connector	1
5V LED (Power on/Off)	1
G250x Digital driver	1

TABLE 9 - COMPONENT LIST MCB

Because of the amount of components within the MCB, it was constructed with two floors. The floors were cut from plexiglas, where the ground floor was mounted directly to the box and contains the various power supplies, heat sink, cooling fan and outwards connectors. The second floor of Plexiglas was then mounted on top of the power supplies creating an elevated platform where the microcontrollers and prototype circuitry boards were mounted.

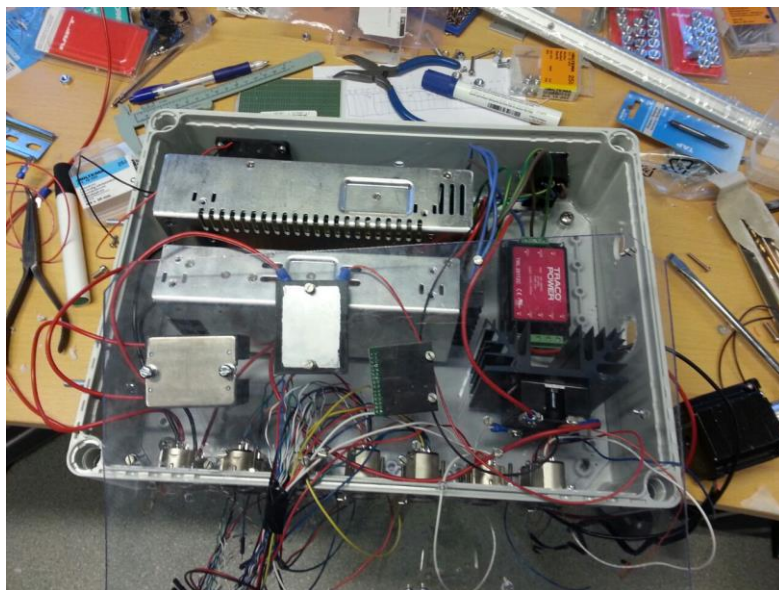
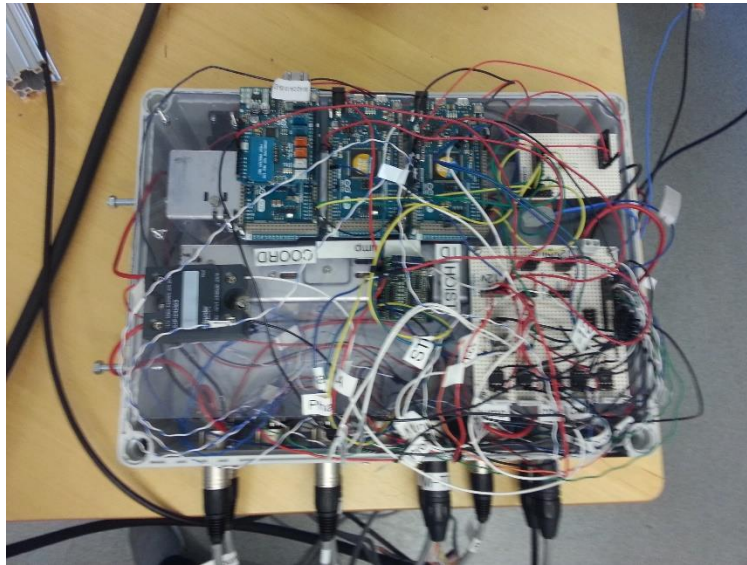


FIGURE 23 - MCB BEING CONSTRUCTED

Due to the 3.3V TTL logic of the Due it was necessary to insert optocouplers<sup>6</sup> between the controller and the components which operated on voltages above 3.3V. This applied to the solid state relays for the hoisting break, the pump and the Omron drive commands.



**FIGURE 24 - MCB FINISHED**

The coordinator communicates with the SDC through an Ethernet connection supplied by the Arduino Ethernet shield mounted on top of the controller. When receiving new data from the SDC, the data is forwarded to the correct controller using I2C. The controllers pull sensory data from the system and passes it back to the coordinator through the same communication path. Finally the coordinator forwards this data to the SDC through UDP datagrams.

For a complete description see Appendix D.

---

<sup>6</sup> An optocoupler is a component which transfers electrical signals between two isolated circuits through the application of light. This prevents low voltage systems being affected by high voltage systems, but enables the low voltage to impact and control the high voltage system.



## 5.4 Directional control box(DCB)

The DCB contains the directional control subsystem. The main purpose of this subsystem is to control the vertical position of the borehole. It is the least essential control system as it doesn't directly affect the rigs potential to drill and was the system with the lowest priority.

The DCB contains the following components:

Component	#
Arduino Mega	2
Arduino Ethernet Shield	1
24V Power Supply	1
12V Power Supply	1
XLR5 Connector	10
XLR4 Connector	2
Jack Connector	5
RJ45 Connector	1
USB Connector	1

TABLE 10 - COMPONENT LIST DCB

Like the MCB, the DCB contains a coordinator handling the communication with the SDC through an Ethernet shield and communicates with the directional controller through I2C. A difference in this setup though is that the coordinator is also in charge of pulling sensory data and issuing commands to the directional controller. The directional controller is connected to both the X and Y actuators and controls the amount of side force applied to the drill string.

For a complete description see Appendix E.

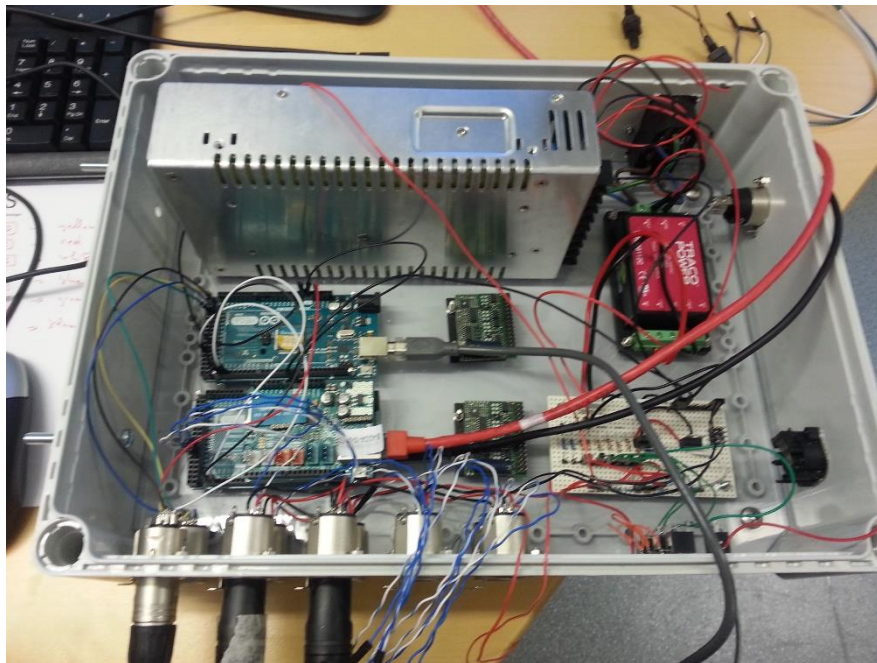


FIGURE 25 - DCB FINISHED

## 5.5 Amplifier circuits

The load cells in the system are connected to amplifier circuits to boost the signal. While ordering the hardware, pre made amplifiers were added for the load cells. These amplifiers can deliver 80 samples per second, this turns out to be:

$$\frac{1000000}{80} = 12500\mu s \text{ per sample}$$

EQUATION 12 - SPARKFUN AMP SAMPLING RATE

With the loop times established in chapter 2.1.1 this sampling rate is unacceptable. The system should be able to collect 20 samples within the loop time and have enough time left to analyse and make a sensible decision. Due to the fact that the premade amplifiers converted the analog readings into digital signals the sample rate is limited by the ADC on those amplifiers.

By constructing custom analog amplifiers, the sampling rate was limited only by the speed of the ADC on the Arduino's. Using the ADC on the Due which can deliver a sampling rate of one sample per  $2\mu s$ , the system now had a setup which fulfilled the requirements of the system.

For a complete description see Appendix F.

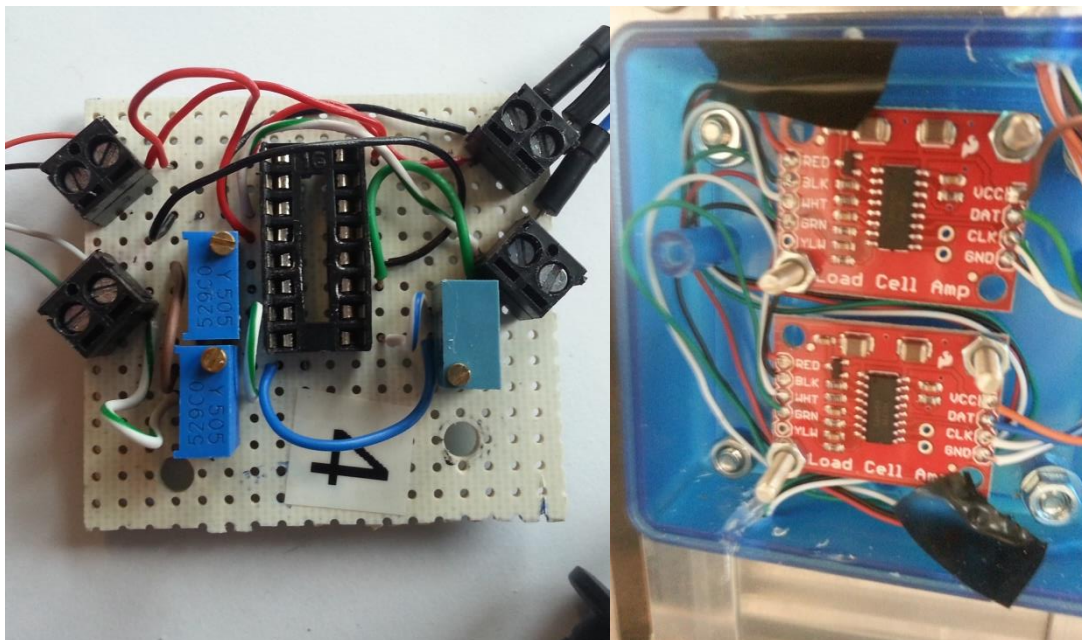


FIGURE 26 - CUSTOM AMP(LEFT), SPARKFUN AMP(RIGHT)

## 6. Implementation

This chapter covers the software implementation of the system in two stages. The first section describes the finite state machine model for each of the sub systems and the last section describes state machine approach used in the SDC. A short description of the software architecture is given in Chapter 3.

It is worth noting that the state machines presented in this chapter have not been implemented and tested in the prototype, the consequence of this being that some state machines presented might not be able to fulfil the intended task to an adequate level. These potential shortcomings should have been found and corrected during implementation and testing of the state machines.

### 6.1 Microcontrollers

Each microcontroller available in the system operates under the principles of a finite state machine. This section describes the different states of each controller and how they are triggered.

#### 6.1.1 Fluid System

The fluid controller states are:

- Initialized
- Pressurize
- Pressurization Error
- Monitor
- Leak
- Obstruction

Due to the fact that this subsystem can't control any aspects related to its operation besides turning itself on or off, this subsystem doesn't have a calibration state. The calibration had to be done manually to obtain sensible limits for the system, so it could recognize leakage and obstructions. How this calibration was performed is covered in chapter x.

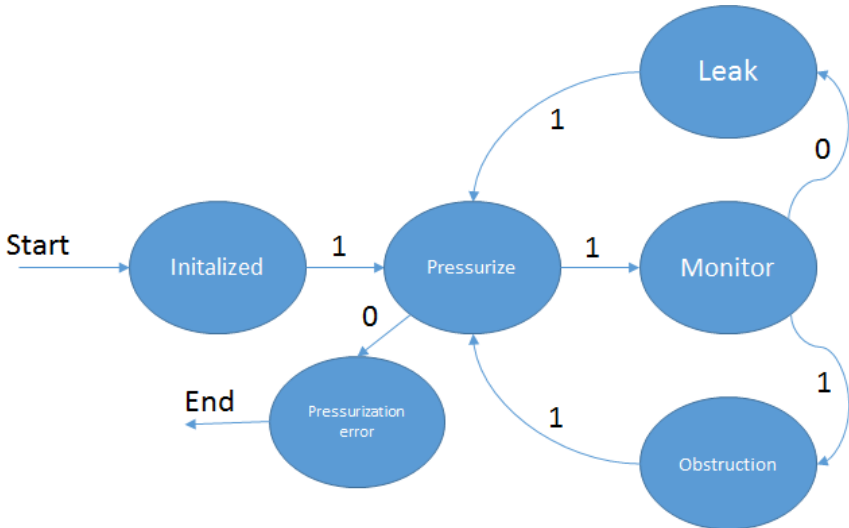


FIGURE 27 - FLUID SYSTEM STATE DIAGRAM

### ***Initialized***

The subsystem enters this state on start-up and waits in this state until an acknowledgement (ACK) message is requested by the main system, at this point it should acknowledge that it is on and wait for an ON command.

### ***Pressurize***

The system enters this state on receiving an ON command, and starts the pump while waiting a period of ten (10) seconds while the system pressurizes. After this delay, it checks the readings from the pressure sensor in the system to verify that the system is pressurized within the established pressure limits.

If the pressure falls outside these limits the controller responds with a pressurization error and halts operations, requiring a reset instigated by the SDC.

If the pressure falls within these limits the controller moves into the monitor state.

### ***Monitor***

The monitor state is the main state for the controller. While in this state, sensory data is collected from the pressure sensor and compared with the established thresholds as well as passed to the SDC through the coordinator controller.

If the pressure readings drop below the lowest pressure threshold, the controller moves into the leak state.

If the pressure readings pass above the highest pressure threshold, the controller moves into the obstruction state.

### ***Leak***

During this state the controller passes a leak error to the SDC, indicating that there is a possible leak in the system which prevents further drilling operations. This state can be exited if the controller receives a repressurize command, moving the controller back into the pressurize state.

### ***Obstruction***

During this state the controller passes an obstruction error to the SDC, indicating that the amount of cuttings in the hole is causing an increase in pressure falling outside the approved thresholds. This state can be exited if the controller receives a repressurize command, moving the controller back into the pressurize state.

### 6.1.2 Hoisting System

The hoisting system consists of the following states;

- Uncalibrated
- Calibration error
- Calibrated
- Hold Position
- Speed Control
- WOB Control
- WOB Error

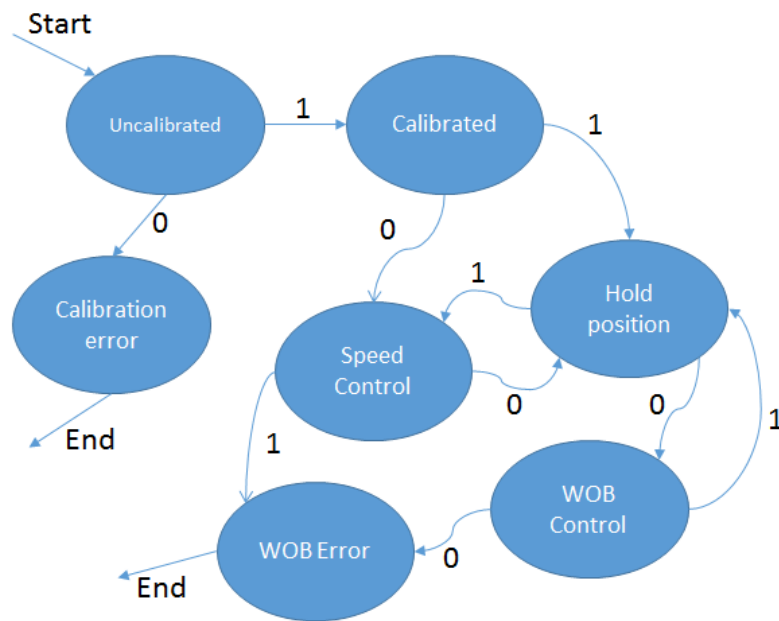


FIGURE 28 - HOISTING SYSTEM STATE DIAGRAM

#### ***Uncalibrated***

When the subsystem is turned ON, the system enters this state and waits for the SDC to request an ACK message that the subsystem is online. After having passed this ACK message, the controller begins its calibration procedure.

This calibration procedure is described in Chapter 7.

If the calibration fails, the controller moves into the Calibration error state. If the calibration is successful, the controller moves into the calibrated state.

#### ***Calibration Error***

The controller notifies the SDC that the calibration failed, and provides an error message stating which aspect of the calibration failed. This error message can then be analysed by the operator, to speed up the process of locating the affected area.

#### ***Calibrated***

While in the calibrated state, the controller notifies the SDC that it's calibrated and then waits for further instruction. From this state the controller can be moved in to one of two states. The "Hold Position" state or the "Speed Control" state.

***Hold Position***

The hoisting system holds its position and waits for further instruction. The system can move into either the “Speed Control” state or the “WOB Control” state depending on commands issued from the SDC.

***Speed Control***

In the “Speed Control” state, the system can control the speed at which the system elevates and descends. Collecting sensor data and validating it towards the boundaries established, if the values collected break these boundaries the controller moves into the WOB error state.

***WOB Control***

While in the “WOB Control” state, the system can specify a certain WOB and the subsystem will attempt to maintain the WOB value specified.

***WOB Error***

Entering this state the controller passes a WOB error message to the SDC. This error can indicate one of two scenarios. Either the drill string is buckling (Described in Chapter 8.3), or that the drill string has splintered.

The latter scenarios should be detected by the power transmission system before it happens.

### 6.1.3 Power Transmission system

The power transmission system consists of the following states;

- Uncalibrated
- Calibration error
- Calibrated
- Hold RPM
- Over Torque
- RPM Control

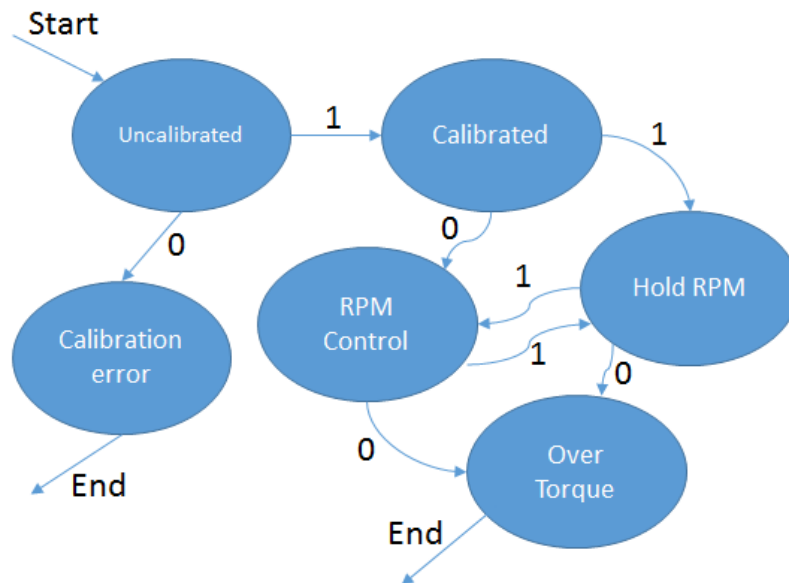


FIGURE 29 - POWER TRANSMISSION STATE DIAGRAM

#### ***Uncalibrated***

When the subsystem is turned ON, the system enters this state and waits for the SDC to request an ACK message that the subsystem is online. After having passed this ACK message, the controller begins its calibration procedure.

This calibration procedure is described in Chapter 7.

If the calibration fails, the controller moves into the Calibration error state. If the calibration is successful, the controller moves into the calibrated state.

#### ***Calibration Error***

The controller notifies the SDC that calibration has failed, and provides an error message stating which aspect of the calibration failed. This error message can then be analyzed by the operator, to speed up the process of locating the affected area.

#### ***Calibrated***

While in the calibrated state, the controller notifies the SDC that it's calibrated and then waits for further instruction. From this state the controller can be moved in to one of two states. The "Hold RPM" state or the "RPM Control" State.

### ***Hold RPM***

During this state the controller maintains the last RPM commanded and monitors the torque readings. If the system detects over torque, it moves the controller into the “Over Torque” state.

If the system needs to change the RPM, it issues a command to move the controller into the “RPM Control” state.

### ***RPM Control***

In this state the system can change to a RPM as it sees fit. When the controller reaches the issued RPM, it moves into the “Hold RPM” state. While changing the RPM, the controller monitors the torque readings and if over torque is detected it moves the controller into the “Over Torque” state.

### ***Over Torque***

This is an end state for the controller and it will require a reset by the SDC to start up again. Entering this state the controller passes an over torque message to the system indicating that a twist off situation might have occurred.

## **6.1.4 Directional system[Appendix G]**

*This section was written by Chief Scientist Eric Cayeux, IRIS.*

To correct any deviations from vertical, the directional controller applies a side force on the stabilizer in the opposite direction to the current borehole deviation. This will be referred to as “riser steering mode”. However, if the borehole does not deviate from vertical, no side force is applied. Similarly, when the drill-string is not drilling, the side force on the stabilizer is removed to facilitate the axial and rotational movement of the BHA. Removing the side force on the stabilizer will be referred to as “riser neutralization mode”. In order to place the riser in neutral position, it is necessary to calibrate the position of the riser. This activity will be called “riser calibration mode”.

The system has therefore the following main states:

- Riser position uncalibrated
- Riser position calibrated
- On bottom
- Off bottom
- Directional control
- Neutral mode



A more detailed deterministic finite automaton is given below. There are three possible fatal exits:

- Error while calibrating
- Error while neutralizing
- Error while steering

Otherwise, the finite automaton always starts by calibrating and then continues to the “off bottom” state which enforces a neutralization of the riser. Thereafter, the system can flip between the “on bottom” and “off bottom” states. Each time, it goes to the “off bottom”, it performs a neutralization. So if the drilling controller lift the bit off bottom, the directional controller will automatically neutralize the riser position. When the state changes to “on bottom”, the system check the verticality. If it is vertical, then it neutralizes the riser and returns to the “on bottom” state, to be ready for a new evaluation. If it is not vertical, it applies a steer command and return to the “on bottom” state to start a new evaluation.

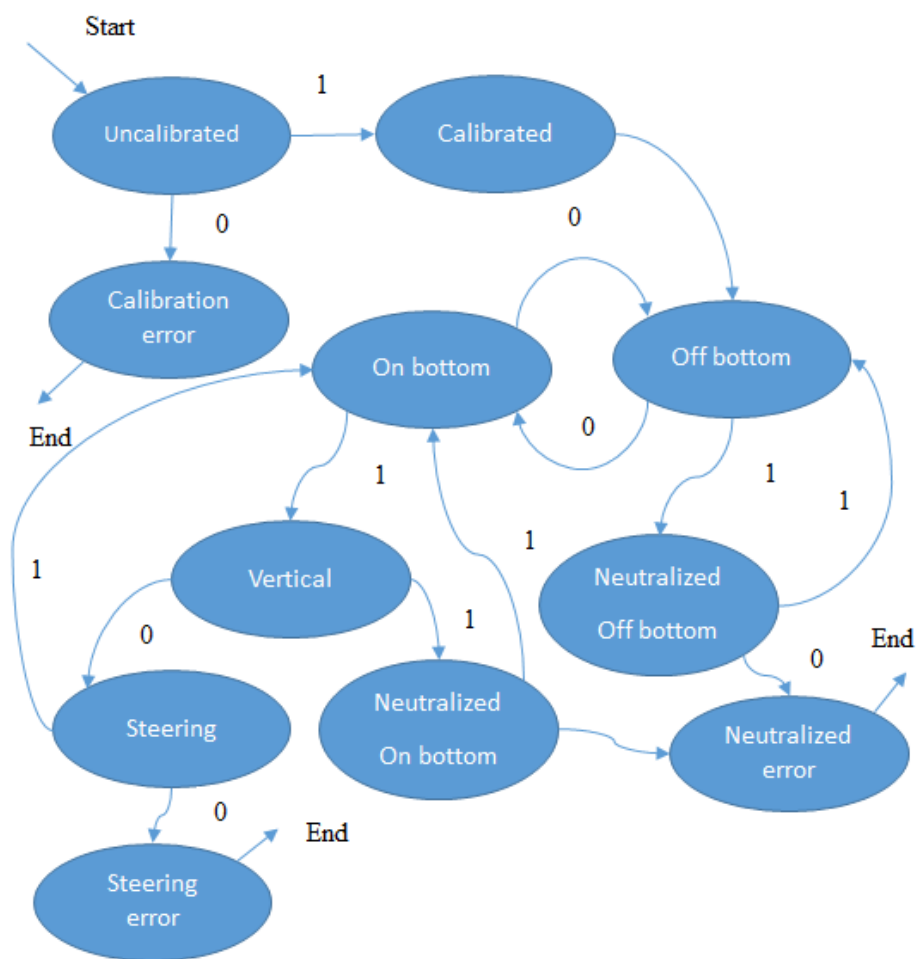


FIGURE 30 - DIRECTIONAL CONTROL STATE DIAGRAM

## 7. Calibration procedures

As stated earlier it is important for the system to be properly calibrated to perform at its highest level. This chapter describes in detail how the physical calibration of the system can be performed. It also gives a detailed description of the various calibration procedures.

### 7.1 Physical Calibration

Three of the subsystems needed to be physically calibrated before the system starts to prevent damage to the microcontroller and establish the pressure limits for the fluid system.

The custom amplifiers used in the hoisting and power transmission subsystems run on +/- 12V, the amplifiers must be calibrated to output voltage signals that the Arduino's can handle namely 3.3V.

#### 7.1.1 Fluid Calibration

Due to the poor datasheet following the selected pump, it was necessary to perform experiments to establish an accurate pump characteristics.

Using a simple setup of two buckets, filling one of the buckets with 10 litres of water. The fluid system was initiated and the time it took the system to pump the water from one bucket to the other was recorded. This experiment was performed several times and the data collected was then analysed by the petroleum's engineers to derive the correct pump equation.

The next calibration step of the fluid system was to determine the pressure thresholds which would indicate either a leak or an obstruction in the system.

When building the fluid system, two ball valves were added to the plumbing. Using ball valves one could limit the flow of water through the valve depending on a specific section of the plumbing to simulate either a leak or obstruction scenario.

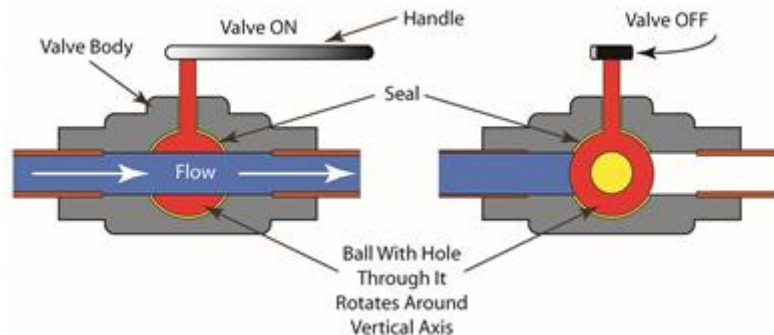


FIGURE 31 - BALL VALVE

Using these valves, the petroleum engineers performed experiments by varying angle of valve handle to measure the pressure in the system. This method was applied to both scenarios.



FIGURE 32 - OBSTRUCTION VALVE(LEFT), LEAK VALVE(RIGHT)

### 7.1.2 Power transmission Calibration

The custom amplifiers for the torque readings, had to be calibrated to avoid frying the Arduino's with too high voltage.

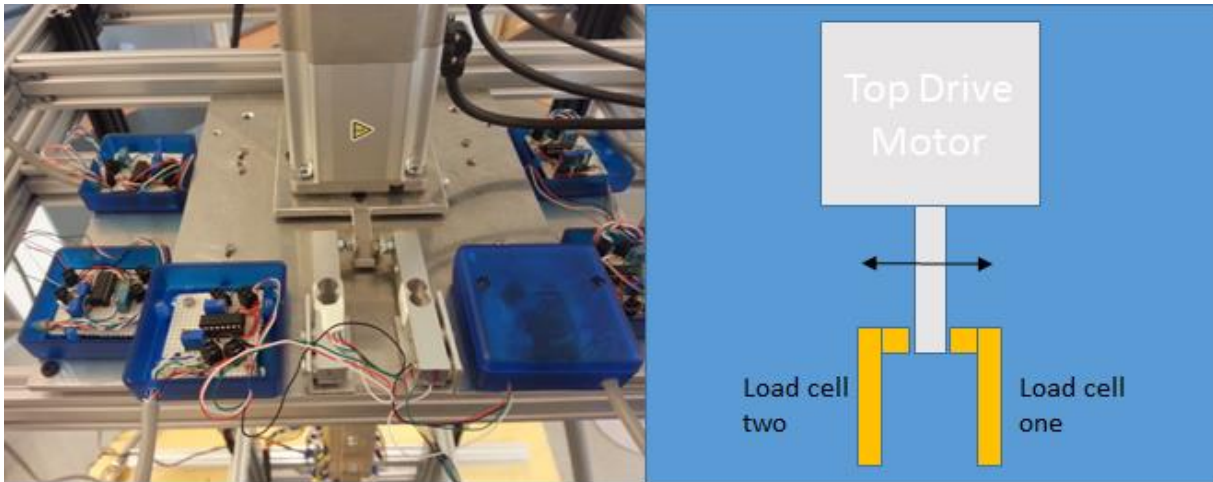


FIGURE 33 - PHYSICAL SETUP(LEFT), OVERVIEW(RIGHT)

The first step in the calibration procedure involved removing any weight applied to the load cells to calibrate it to a neutral zero value. This was achieved by adjusting the potentiometer controlling the offset of the amplifier.

With the first step done, the gain was calibrated by connecting a pulley to the framework and then fastening a string to the load cell in question. Attaching a 5kg weight on the end of that string, the weight was then left to hang freely while the team used a volt meter to check the output voltage of the amplifier.

Using the potentiometers controlling the gain, this was adjusted so that if the load cell experienced a weight of 5kg being applied, it would output the maximum established voltage of 3.3V.

The petroleum engineers estimated that the system would never exceed a torque equivalent to 10kg, and therefore this was used as a maximum value for the amplifier.

### 7.1.2 Hoisting Calibration

Like the power transmission subsystem, the hoisting subsystem also uses the custom amplifiers and these also had to be calibrated to avoid damage to the Arduino's.

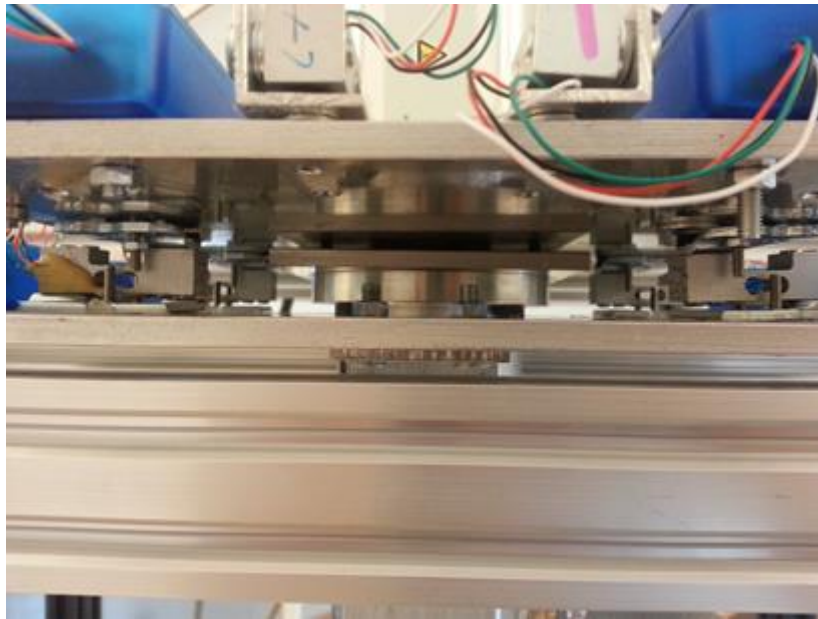


FIGURE 34 - HOISTING LOAD CELLS

The hoisting system uses four load cells for the WOB measurement and holds the weight of the power transmission system. The neutral calibration for these load cells therefore had to be zero valued at the weight of this system.

The power transmission system weighs in on 7.2kg, and the offset was therefore set to give WOB readings of zero with this weight.

It is important to note, that the WOB applied pushes the entire hoisting system up as indicated by the black arrows in the figure below.

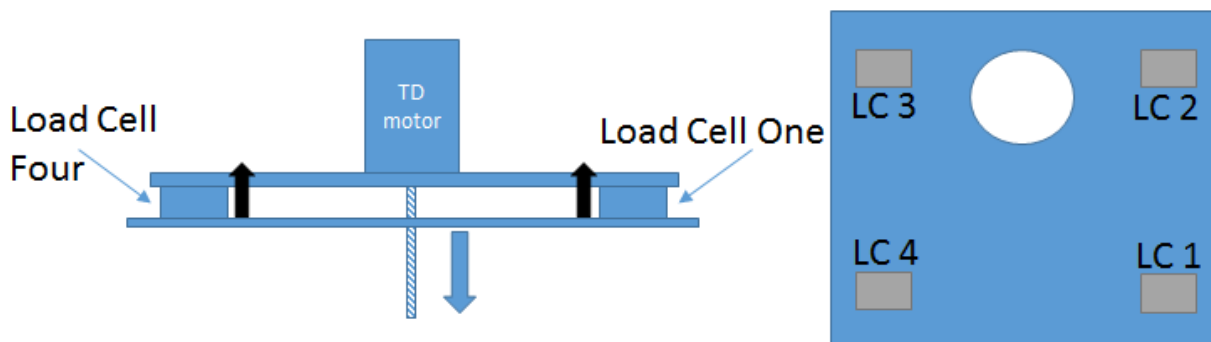
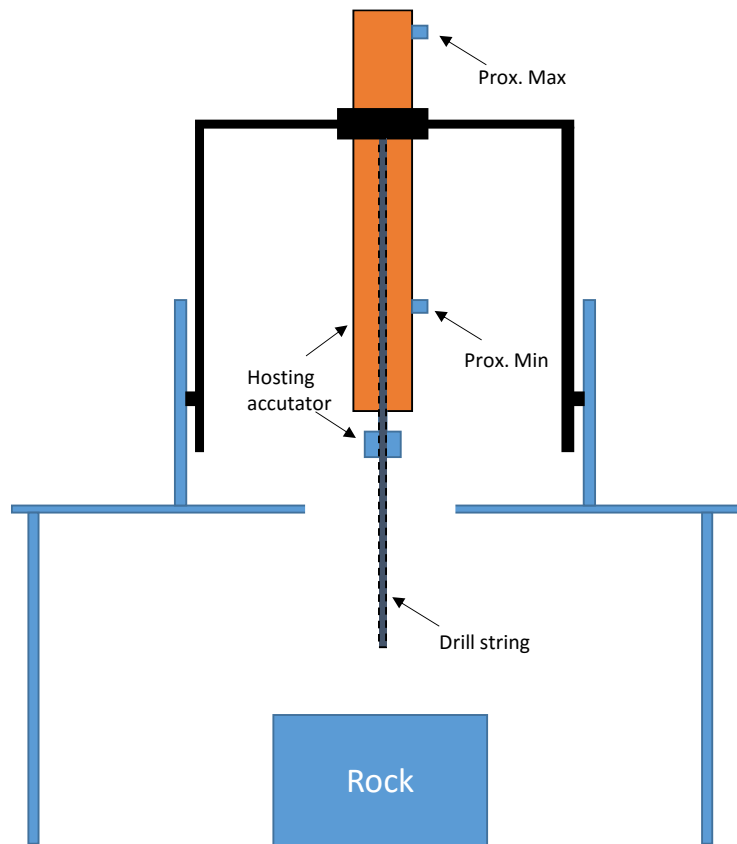


FIGURE 35 - HOISTING SETUP

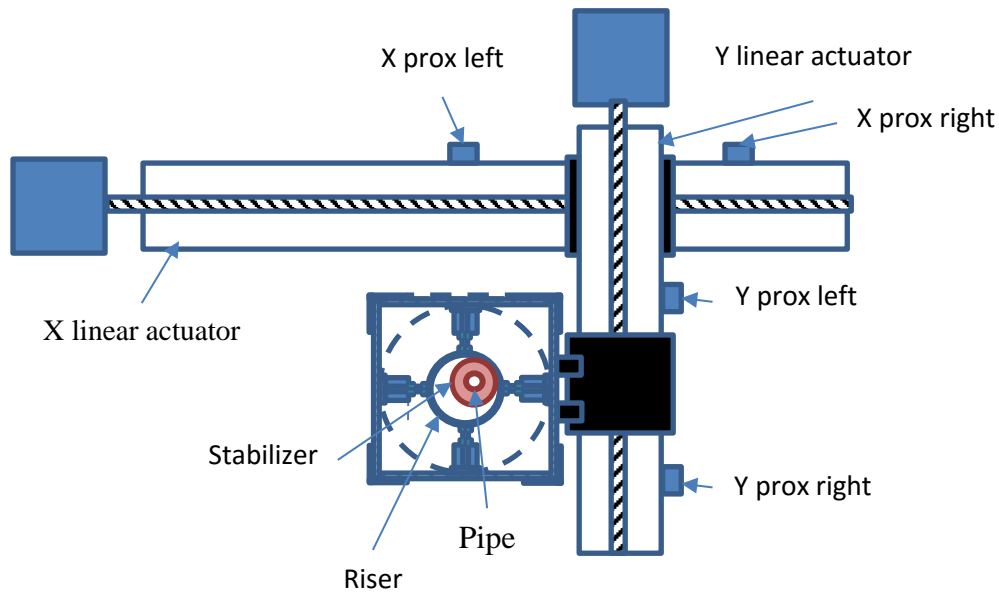
The gain was then calibrated by attaching a pulley much like the power transmission system lifting the power transmission section of the top drive segment upwards with a 10kg weight. This simulates the application of 10kg WOB and the gain for this was then set to 3.3V.

## 7.2 Software Calibration procedures



**FIGURE 36- HOISTING OVERVIEW**

The hoisting system should establish the maximum and minimum elevation of the system. The maximum is found by elevating the rig until it hits the proximity sensor at the highest elevation, then by moving slowly towards the rock it should tag the bottom hole, the top of the rock, which will give a WOB increase to indicate that the bit is now on top of the rock. With these two parameters the positional control of the hoisting system is calibrated.



**FIGURE 37 - DIRECTIONAL CONTROL OVERVIEW**

The hoisting system should then be instructed to tag the bottom hole again and lift off about 1-2cm, such that the BHA is placed within the riser. The system should then proceed to calibrate the directional control by measuring the side forces exerted on to the drill string and position the linear actuators of the directional control such that no side forces is applied. When this is achieved the directional control subsystem is calibrated.

With two of the four subsystems calibrated the hoisting system should now hoist the rig out of the riser to a point where the drill string is hanging in a neutral free position mid-air and the top drive can start its calibration procedure.

At this point the top drive can begin an RPM test, by spinning the drill string from a minimum RPM and increasing this RPM incrementally up to the maximum set point limit of 180. During this the system should collect data from the torque sensors to establish minimum torque readings for each given RPM. With these values established the system can move on to calibrate the fluid system.

Before the system initiates the fluid system, all other subsystems should be put in to an idle state, not to interfere with the baseline readings from the pressure sensor. The main calibration of the fluid system had to be done manually as there is no method of controlling this subsystem with the exception of turning it on or off. The goal of this calibration is to read the sensor data forwarded from the pressure sensor to get a base value for what the pressure in the system is like without interference and to check for any potential leaks or obstructions in the system.

Finally when this process is completed, the top drive should run a second RPM test to measure the effect of the fluid system on the torque in the system if any.

## **8. Dynamic conditions**

When operating in the drilling state, the conditions are dynamic and can within seconds. This makes it almost impossible to predict settings that will allow optimal drilling efficiency.

### **8.1 Stick-slip**

The stick-slip phenomenon occurs when the bit gets stuck and as the drill string is rotating it will build up energy, just as a spring if one end was fixed and the other end twisted. When the energy reaches a certain level it will overcome the friction which is holding the bit back and release the energy in one quick jerk or it will break.

Our method for handling this situation is to monitor the amount of torque being registered. If this exceeds a certain value, the system will assume that a stick-slip situation has occurred and then proceed to elevate the drill string of the bottom of the hole to release the tension.

At this point we could proceed drilling with the same settings as we had initially or assume that the situation will most likely occur again. Therefore the system will go back to its initial worst case conditions setting and attempt to recalibrate WOB, RPM etc. To find the optimal settings yet again.

### **8.2 Whirl**

There are two categories of this phenomenon, forward whirl and backwards whirl. Of the two categories forward whirl is the best case condition as it won't damage the drill bit straight away. Backwards whirl if left untreated for a minimal amount of time can destroy the drill bit.

Detecting whirl is done by analysing the sensor data retrieved from the load cells in the riser and the acceleration data from the accelerometers in BHA (drill collar). If the load cells record sporadic forces being applied as well as high frequency readings from the accelerometer then the system can assume that there is some form of whirl in the system.

By stopping the rotation in the system and lifting off the bottom letting the drill string settle back in to a free state we can eliminate the whirl and restart drilling operations

### **8.3 Buckling**

Buckling occurs if too much WOB is applied on the drill string. This causes stress on the aluminium pipe which if the WOB isn't decreased will strain the pipe until it breaks.

### **8.4 Twist off**

Twist off is the technical term for when over torque causes the drill string to break in two.

### **8.5 Overpull**

Overpull occurs when the bit gets stuck in the borehole and the rig attempts to pull the drill string out. This cause the drill string to stretch and will eventually break the pipe it is not detected and dealt with.

### **8.6 Pack-Off[12]**

Pack-off can happen for a variety of reasons, the most common being that the drilling fluid is not transporting cuttings properly.

### **8.7 Washout[13]**

Washout indicates an enlargement in the borehole. This occurs most commonly when drilling in soft formations and is caused by excessive bit jet velocity.

## 9. Graphical user interface and logging

The GUI was created through the use of the .NET framework, and due to the late finish of the electrical setup, the scientist assisting the team helped construct and design the GUI to ensure that some early testing would be feasible. This chapter describes the GUI and the different situations which should be displayed. All information during a session is displayed and also logged into a separate text file in the backend for debugging purposes.

The figure below is highlighted into different sections which are then described one section at a time.

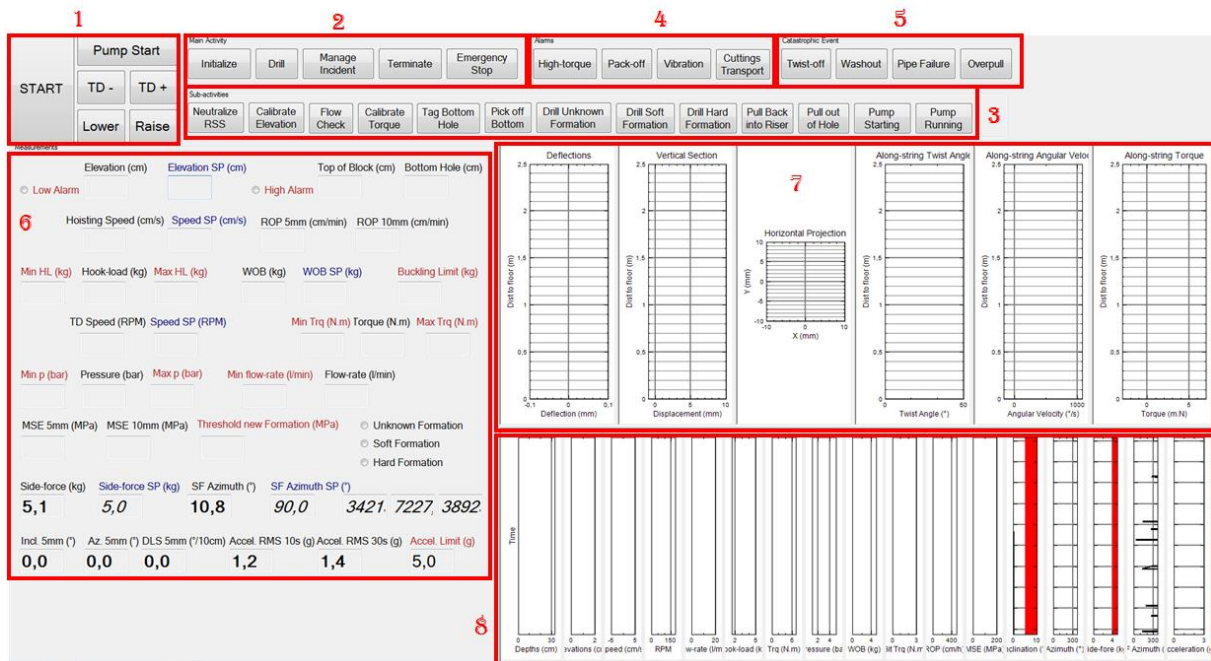


FIGURE 38 - IRIS GUI

1

Contains the START button to initiate the system and some manual controls for debugging and testing purposes.

<b>Pump Start</b>	Starts the pump
<b>TD- / TD+</b>	Changes the RPM of the power transmission system.
<b>Lower / Raise</b>	Changes the elevation of the hoisting system.



2

Indicates the main activities that the system will do.

<b>Initialize</b>	As described earlier sets up the system and calibrates it before commencing normal drilling operations.
<b>Drill</b>	Drilling state.
<b>Manage Incident</b>	If a safety trigger is triggered the system will indicate that it is now attempting to correct an issue.
<b>Terminate</b>	Indicates that the system has terminated operations. This is either because the system finished its task or a critical error has occurred forcing the system to terminate.
<b>Emergency Stop</b>	Indicates that the emergency stop has been triggered by the supervising operator.

3

Indicate which sub activity the system is currently engaged in.

<b>Neutralize RSS</b>	Directional control calibration, neutralizing the riser so no side forces are applied to the BHA.
<b>Calibrate Elevation</b>	Calibrating the hoisting system
<b>Flow Check</b>	Checking the flow rate of the system, based on pressure readings.
<b>Calibrate Torque</b>	Calibrating torque for the power transmission system.
<b>Tag Bottom Hole</b>	Indicates that the emergency stop has been triggered by the supervising operator.
<b>Drill Unknown Formation</b>	Indicates that the MSE is currently unknown.
<b>Drill Soft Formation</b>	Indicates that the MSE is known and it is recognized as soft formation.
<b>Drill Hard Formation</b>	Indicating that the MSE is known and that it is recognized as hard formation
<b>Pull Back into Riser</b>	Indicates that the system is pulling the BHA back into the riser.
<b>Pull out of Hole</b>	Indicates that the system is lifting the bit and BHA out of the hole, possible obstruction in the fluid system or a stick slip situation.
<b>Pump Starting</b>	Indicates that the pump has been started.
<b>Pump Running</b>	Indicates that the pump is running normally.

4

Lights up to indicate dangerous conditions occurring during drilling.

<b>High Torque</b>	The torque data collected indicates that it is close to breaking the safety envelope.
<b>Pack-off</b>	Indicates an obstruction in the borehole which decreases the system's ability to transport cuttings.
<b>Vibration</b>	Indicates vibration in the drill string, possible back or forwards whirl.
<b>Cuttings Transport</b>	Indicates that the flow velocity in the annulus is lower than the established limit for transporting cuttings out of the borehole.

5

This tab lights up if one of the catastrophic events listed below occur. Either of these events causes the system to terminate operations.

<b>Twist-off</b>	Described in Chapter 8.4
<b>Washout</b>	Described in Chapter 8.7
<b>Pipe Failure</b>	Indicates that the pipe has broken.
<b>Overpull</b>	Described in Chapter 8.5

6

This section of the GUI displays real time information gathered from the system and is divided into three categories.

<b>Red Label</b>	Displays the minimum and maximum thresholds currently in use by the system.
<b>Blue Label</b>	Changes the RPM of the power transmission system.
<b>Black Label</b>	Displays values currently being feed to the system from the various subsystems.

7

Section seven displays the last thirty seconds of data collected from the system.

8

This section displays the historical data from start to finish.

## 10. Competition results

Because neither UIS nor UIA was selected to participate in Drillbotics phase two, an internal competition between the two universities was arranged as stated in the introduction. The competition day for UIS was the 27.May 2016.

The day started off by the UIS team presenting their work and answering any questions the jury had about the design and implementation.

The jury was informed that due to the time consuming work finishing the mechanical and electrical setup that the prototype was only finished the day before and because of this the software currently implemented at that stage was highly experimental.

The UIS team decided not to attempt drilling because of health and safety reasons associated with the open electrical boxes and makeshift power supply, which marked the end of the juries visit.

After having reached a decision the final results for UIS is displayed below.

University in Stavanger
UIS had a smaller team which we think limited their progress, presentation was rather hurried and one appeared made at the last minute, with lots of abbreviations which were not fully defined. Much more focus on the math and physics, and the use of the data to optimize.
Construction looked a lot less study and it would be interesting how it handles drilling vibration.
In terms of HSE there were too many open electrical boxes and loose wiring, and apparently no end stops, just an emergency shutdown.
Good use of heavy BHA to help avoid buckling.
Novel handling of the combination of gravity + centrifugal + Coriolis + Euler forces from downhole sensors including noise filtering.
Had good flow regimes, and seemed to have precise control.
Had good alarm setups in software, though GUI /HMI is much more focused at engineers.
Well thought out time response strategy for shutdown of top drive.
Apparently had a self-calibrating system
Excellent that they have been looking at the use of downhole sensors, which allowed some form of directional control.
Unfortunately over budget and they could not show the machine in operation.
It was apparent that the late change in the drill-pipe size effected UIS more because of processor choice.

FIGURE 39 - COMPETITION RESULT UIS

Summary
In summary we have to award it to UiA as they demonstrated the machine, as requested, though they did have some limitations and described above. UiS had good focus on the control and data management and less on the design aspects, though what they have achieved is more than good enough for the task. UiS more clearly Petroleum focused, whilst the UiA ended up with the best rig which was clearly very movable.
As we did not see the the UiS rig in operation ,we would still like to see it in motion at a future date!
One main thing is that this is an excellent initiative and should continue, as Norway should continue to have a leading position in Automation and Autonomy in Drilling, and we need the ideas and support from Academia.

FIGURE 40 - COMPETITION SUMMARY

## 11. Conclusion

In this thesis, an autonomous system for drilling in heterogeneous rock formations was designed and implemented to a certain degree, due to lack of manpower and time the system was not completed before this thesis was concluded.

The thesis was driven by the international DrillBotics competition and the desire to automate the drilling industry to increase both safety and cut costs. The competition aims at building small scale prototypes which act as a proof of concept. Working on such a small scale prototypes, the problems associated with automation becomes much more apparent and time available to handle these issues goes down from minutes to microseconds.

There are a number of conclusions to note for this thesis, one being that a project of this scale requires a lot more planning ahead of project start. The process of gathering students for the team needs to start earlier than it did for this year. And it is imperative that all the necessary disciplines which are needed to finish the project is represented within the team. This year's lack of an electrical engineer had detrimental consequences for the project as none of the students in the team was fully prepared to handle this aspect of the prototype. It also became apparent that the students selected for this year's project lacked substantial knowledge necessary to complete such a project. It is in my opinion important that the University setups up some form of an introductory course for the next year's team, to make sure that all students have some basic knowledge about the challenges they might meet.

Another aspect which was found peculiar is the lack of information to be gathered on this type of project. The industry doesn't prioritize the sharing of information which is essential when one wants to further the technological development within a field. Without this information this year's team had to rely heavily on the expertise and experience of the scientists at IRIS.

It is also worth mentioning that the Drillbotics competition conducted their business in quite an unprofessional manner in this year's competition. Not meeting the deadlines they themselves established, poor formulation and unclear rules which were later changed at their discretion. The most notable rule change that was introduced in late April, was that subsequent change in the aluminium pipe's dimensions. Originally working with a wall thickness of 0.016-in (0.4064 mm), this was changed to a pipe with a wall thickness of 0.035-in (8.89 mm).

After having done an analysis of what this pipe could handle, it became clear that close to none of the dynamic conditions which the team had based their work around, could simply not occur. The old pipe could withstand a maximum twist angle of 0.3 degrees and a maximum WOB of a proximately 2-3kg, while the new pipe could handle a maximum twist angle of 36 degrees and a maximum WOB larger than 20kg!

In addition the lack of dynamic conditions that could occur, this change would also drastically change the necessary responds time available to the system. If this change had been announced earlier in the design process and not during the actually building of the prototype the team could have made a substantially simpler system and possible managed to finish the system before the conclusion of this thesis.

In summary, a project of this scale needs more planning especially when the team lacks manpower and the correct disciplines within the team. It is vital that more information be shared publicly in this field to further development and hopefully this thesis can contribute to this in a small way. The university should thoroughly consider if it wants to participate in the next year's competition of Drillbotics or possibly keep this project on for master students and formulate the problem themselves. When

working with a real time system, the components need to be analysed thoroughly to establish the correct responds time available at an early stage to avoid having to redesign the system several times over.

Appendix H contains pictures taken during the project and of the full rig.

**Budget**

The budget displayed below reflects the total cost of components used in the system.

<b>Section:</b>	<b>Cost:</b>
<b>Small Hardware</b>	7287,00
<b>Mostly frame and aluminium pipe</b>	26964,56
<b>Mostly electrical</b>	32316,60
<b>Mostly electrical and plumbing</b>	15906,43
<b>Total:</b>	88970,94

TABLE 11 – BUDGET

*Unfortunately over budget and they could not show the machine in operation.*

*-, Competition jury*

The above citation was mentioned by the jury in their final report, it is important to note that the budget above only represents the cost of components in the system. Outside these cost, there was a cost related to the man hours put in by the IRIS workshop which came out to about 50 000 NOK, and with this caused the team to exceed the budget.

## References

- [1] Drillboticscom. (2016). *Drillboticscom*. Retrieved 4 June, 2016, from <http://drillbotics.com/index.php/about-drillbotics/>
- [2] Nick Gammon. (2015). *SPI- Serial Peripheral Interface – For Arduino*. Retrieved 4.June 2016, from <http://www.gammon.com.au/forum/?id=10892>
- [4] Arduino. (Not Defined). *SPI Library*. Retrieved 4. June 2016, from <https://www.arduino.cc/en/Reference/SPI>
- [5] Motorola, Inc. (2000). *SPI Block Guide V03.06 (S12SPIV3/D)*. Motorola, Inc.
- [6] Nick Gammon. (2014). *I2C – Two-Wire Peripheral Interface – For Arduino*. Retrieved 5.June 2016, from <http://gammon.com.au/i2c>
- [7] Hopcroft, J. E., & Ullman, J. D. (1979) *Introduction to automata theory, languages and computation*, Reading, Mass: Addison-Wesley
- [8] Glenn Brown. (2000). *The Darcy-Weisbach Equation*. Retrieved 10.June 2016, from <https://bae.okstate.edu/faculty-sites/Darcy/DarcyWeisbach/Darcy-WeisbachEq.htm>
- [8] Subramanian, R. S., (2016) *Reynolds Number*, Clarkson University: Department of Chemical and Biomolecular Engineering.
- [9] Swaffield, J. A., & Bridge S. (1983) *Applicability of the Colebrook-White Formula to Represent Frictional Losses in Partially Filled Unsteady Pipeflow*, Uxbridge: Brunel University
- [10] John Henry Ford, (2016). *Duff Norton Rotary Unions*, Retrieved 10.June.2016, from [https://www.jhf.com/Catalog\\_v6/0403.pdf](https://www.jhf.com/Catalog_v6/0403.pdf)
- [11] Robinson, L. (2010) Drill Bit Nozzle Pressure Loss, AADE, *AADE Fluids Conference and Exhibition, Houston, Texas 2010*
- [12] Schlumberger. (Not Defined) Pack off definition. Retrieved 14.June2016, from [http://www.glossary.oilfield.slb.com/Terms/p/pack\\_off.aspx](http://www.glossary.oilfield.slb.com/Terms/p/pack_off.aspx)
- [13] Schlumberger. (Not Defined) Washout definition. Retrieved 14.June2016, from <http://www.glossary.oilfield.slb.com/en/Terms/w/washout.aspx>
- [14] Sparkfun (2015) *Serial Peripheral Interface (SPI)*. Retrieved 4.June 2016, from <https://learn.sparkfun.com/tutorials/serial-peripheral-interface-spi>
- [15] Bogomolny, A (1996) *Equations of a straight line*. Retrieved 5.June 2016, from <http://www.cut-the-knot.org/Curriculum/Calculus/StraightLine.shtml>

## **Appendix A**

See attachment, separate document. Drillbotics-2016-PhaseOne-Final.pdf

## Appendix B

### Components

Fluidcomponent.cs

```
public class Component
{
    public string name { get; set; } // Name of the component
    public double roughness { get; set; } // Fluid roughness (mm)
    public double innerDim { get; set; } // Inner dimension (mm)
    public double outerDim { get; set; } // Outer dimension (mm)
    public double length { get; set; } // Total length (m)
    public double re { get; set; } // Reynolds number
    public double fricFrac { get; set; } // Darcy Friction Factor
    public double currPloss { get; set; } // Current pressure loss for
the component
}
```

AlumPipe.cs

```
public class AlumPipe : Component
{
    public AlumPipe(string n, double rough, double innDim, double len)
    {
        name = n;
        roughness = rough;
        innerDim = innDim;
        length = len;
    }
}
```

Annulus.cs

```
public class Annulus : Component
{
    public double innerDimHole { get; set; } // InnerHole Dimension
    public double outerDimBHA { get; set; } // Outer Dim of the BHA

    public Annulus(string n, double idh, double odb, double len, double
rough)
    {
        name = n;
        innerDim = idh-odb;
        this.innerDimHole = idh;
        this.outerDimBHA = odb;
        length = len;
        roughness = rough;
    }
}
```



## BHA.cs

```
public class BottomHalfAssembly : Component
{
    public BottomHalfAssembly(string n, double rough, double innDim, double
len)
    {
        name = n;
        roughness = rough;
        innerDim = innDim;
        length = len;
    }
}
```

## Bit.cs

```
public class Drillbit : Component
{
    public int nozzles { get; set; }
    public double nozzleDim { get; set; }

    public Drillbit(string n, double dia, double nozDia, int nozz)
    {
        name = n;
        outerDim = dia;
        this.nozzles = nozz;
        this.nozzleDim = nozDia;
    }
}
```

## Hose.cs

```
public class Hose : Component
{
    public Hose(string n, double rough, double inDim, double len)
    {
        name = n;
        roughness = rough;
        innerDim = inDim;
        length = len;
    }
}
```

## Swivel.cs

```
public class Swivel : Component
{
    public Swivel(string name)
    {
        this.name = name;
    }
}
```

## System modelSystemmodel.cs

```
public class SystemModel
{
    public ICollection<Component> _theSystem;

    //Components
    public Drillbit bit;
    public Hose hose;
    public AlumPipe pipe;
    public BottomHalfAssembly bha;
    public Annulus ann;
    public Swivel swivel;

    public double systemElevation = 2.0;
    public double hydrostaticLoss = 0;

    public double rockSize = 0.4;
    public double maxElevation = 2.0; // Given by SysElev
    public double minElevation = 1.6; // Given by SysElev - rockSize;

    public SystemModel()
    {
        //bit = new Drillbit("bit", 0.0275, 0.005 , 1); // China
        bit = new Drillbit("bit", 0.028575, 0.003175, 2); // Baker
        hose = new Hose("Hose", 0.00003, 0.0127, 2.5);
        pipe = new AlumPipe("Pipe", 0.00003, 0.0087, 0.914);
        bha = new BottomHalfAssembly("BHA", 0.00015, 0.0065, 0.45);
        //ann = new Annulus("Annulus", 0.0275, 0.022,0,0.00015); // China
        ann = new Annulus("Annulus", 0.028575, 0.022, 0, 0.00015); // Baker
        swivel = new Swivel("Swivel");

        _theSystem = new List<Component>();

        _theSystem.Add(bit);
        _theSystem.Add(hose);
        _theSystem.Add(pipe);
        _theSystem.Add(bha);
        _theSystem.Add(ann);
        _theSystem.Add(swivel);
    }

    public void incSysElev(double val)
    {
        if (Math.Ceiling(systemElevation) > minElevation && systemElevation <
maxElevation)
        {
            systemElevation += val;
            ann.length -= val;
        }
    }

    public void decSysElev(double val)
    {
        if (systemElevation >= minElevation && systemElevation <= maxElevation)
        {
            systemElevation -= val;
            ann.length += val;
        }
    }
}
```

```
public double SystemElevation
{
    get { return systemElevation; }
    set { systemElevation = value; }
}

public double hydroStaticLoss
{
    get { return hydrostaticLoss; }
    set { hydrostaticLoss = value; }
}
}
```

## Formulas

FluidMathematics.cs

```
static class PressureLossFormula
{
    private static double PI = Math.PI;
    private static double PIsqrt = PI*PI;
    private static double bc = 0.95*0.95;
    private static double constNozzle = 1.03; // Characteristic of the nozzle
    private static double cv = 5.25;
    private static double pas2Bar = Math.Pow(10, -5);
    private static double lpm2m3s = 60000;

    // Gravity
    private static double grav = 9.81;
    // Swivel constants
    private static double swivConstOne = 6894;
    private static double swivConstTwo = 0.264;
    private static double swivConstV = 5.25;

    public static double lossHosePipeBHA(double ro, double q, double f, double
l, double d)
    {
        double fV = flowVelocity(q, d);

        double loss = (f * ro * fV * fV * l )/ (2 * d);

        return loss*pas2Bar;
    }
    public static double lossBit(double ro, double q, double n, double d)
    {
        q = q / lpm2m3s;
        double a = (PI * d * d) / 4;

        double res = (ro * q * q) / (1.975 * constNozzle * constNozzle
*((n*a)*(n*a)));

        res = res * pas2Bar;

        return res;
    }
    public static double lossAnnulus(double ro, double f, double q, double l,
double d)
    {
        double fV = flowVelocity(q,d);

        double res = (f * ro * fV * fV * l) / (2 * d);

        return res*pas2Bar;
    }
    public static double lossSwivel(double q)
    {
        double res = swivConstOne * (((q * swivConstTwo) / 2) * ((q *
swivConstTwo) / 2))/swivConstV;
        return res*pas2Bar;
    }
}
```

```

public static double hydrostaticPressure(double ro, double h)
{
    double res = ro * h * grav;
    return res*pas2Bar;
}

public static double flowVelocity(double q, double innerDim)
{
    return (q / lpm2m3s) * (1 / ((PI * innerDim * innerDim) / 4));
}
}

```

## Pressure calculator

PressureCalculator.cs

```

public class Pressure
{
    public double q = 13; // Flow rate

    // Constants
    public double ro = 998.2; // Liquid Density (kg/m^3)
    public double u = 0.001002; // Viscosity (Pa.s)

    public int darcy = 64; // Constant used in darcys friction formula

    // Friction factor limits
    public int laminar = 2300; // Re <= 2300
    public int turbulent = 4000; // Re => 4000

    // Something...
    public SystemModel model;
    public double currentPressureLoss;

    // Iterating to correct value
    public double pDelta = 0;
    public double pBar = 0;
    public double pumpBar = 0;

    // Pump Characteristics (Combined)
    private double a = -3.2669;
    private double b = 16.372;

    public double estPressureSystem;

    public Pressure()
    {
        model = new SystemModel();
    }

    public Pressure(SystemModel mod)
    {
        model = mod;
    }
}

```

```

public void estimatedPressure()
{
    q = 17;

    pBar = 100;
    while (Math.Abs(pBar - pumpBar) > 0.1)
    {
        pumpBar = linearInterpolation(0, 2.8, q, 17, 11.9);
        //pumpBar = linearInterpolation(0, 2, q, 14, 7);
        calcReyAndFricFrac();

        currentPressureLoss = calcLoss();

        //pDelta = 14 - (3.5 * currentPressureLoss);
        pDelta = 17 - (1.82 * currentPressureLoss);
        pBar = linearInterpolation(0, 2.8, pDelta, 17, 11.9);
        //pBar = linearInterpolation(0, 2, pDelta, 14, 7);

        if(q == 14 || q != 14)
        {
            q--0.1;
        }
        if(q < 7)
        {
            break;
        }
    }
    q = pDelta;
    estPressureSystem = pBar;

    var velo = (q / 60000) * (1 / ((model.ann.innerDimHole *
model.ann.innerDimHole) - (model.ann.outerDimBHA *
model.ann.outerDimBHA)));
}

```

```

public void estimatedPressureTwoIdenticalPumps ()
{
    q = 14;
    double watchlist; // Debug assist
    pBar = 100;
    while (Math.Abs(pBar - pumpBar) > 0.1)
    {
        pumpBar = linearInterpolation(0, 2, q, 14, 7); // This might
not be correct
        calcReyAndFricFrac();

        currentPressureLoss = calcLoss();

        pDelta = 14 - (3.5 * currentPressureLoss);
        pBar = linearInterpolation(0, 2, pDelta, 14, 7);
        watchlist = pBar;
        pBar += pBar;

        if (q == 14 || q != 14)
        {
            q -= 0.1;
        }
        if (q < 7)
        {
            break;
        }
    }
    q = pDelta;
    estPressureSystem = pBar;

    var velo = (q / 60000) * (1 / ((model.ann.innerDimHole *
model.ann.innerDimHole) - (model.ann.outerDimBHA *
model.ann.outerDimBHA)));
}

```

```

public void estimatedPressureTwoDifferentPumps ()
{
    q = 14;

    pBar = 100;
    pumpBar = 0;
    while (Math.Abs(pBar - pumpBar) > 0.1)
    {
        pumpBar = linearInterpolation(0.726071, 2.868775, q, 14, 7);
        calcReyAndFricFrac();

        currentPressureLoss = calcLoss();

        pDelta = (a * currentPressureLoss) + b;
        pBar = linearInterpolation(0.726071, 2.868775, pDelta, 14,
7);

        if (q == 14 || q != 14)
        {
            q -= 0.1;
        }
        if (q < 7)
        {
            break;
        }
    }
    q = pDelta;
    estPressureSystem = pBar;
}

```



```

public double calcLoss()
{
    double loss;
    foreach (Component c in model._theSystem)
    {
        if (c.name.ToLower().Equals("hose") ||
c.name.ToLower().Equals("pipe") || c.name.ToLower().Equals("bha"))
            c.currPloss = PressureLossFormula.lossHosePipeBHA(ro, q,
c.fricFrac, c.length, c.innerDim);

        if(c.name.ToLower().Equals("annulus"))
        {
            c.currPloss = PressureLossFormula.lossAnnulus(ro,
c.fricFrac, q, c.length, c.innerDim);
        }

        if (c.name.ToLower().Equals("swivel"))
        {
            c.currPloss = PressureLossFormula.lossSwivel(q);
        }

        if (c.name.ToLower().Equals("bit"))
        {
            c.currPloss = PressureLossFormula.lossBit(ro, q,
((Drillbit)c).nozzles, ((Drillbit)c).nozzleDim);
        }
    }

    loss = model.hydrostaticLoss =
PressureLossFormula.hydrostaticPressure(ro, model.systemElevation);

    foreach (Component c in model._theSystem)
    {
        loss += c.currPloss;
    }

    return loss;
}

public void calcReyAndFricFrac()
{
    foreach (Component c in model._theSystem)
    {
        if (c.roughness != 0)
        {
            c.re = ReynoldsNumb(PressureLossFormula.flowVelocity(q,
c.innerDim) ,c.innerDim);
            c.fricFrac = frictionFactor(c.re, c.innerDim,
c.roughness);
        }
    }
}

```

```

public double ReynoldsNumb(double fV, double d)
{
    return ((ro*fV*d)/(u));
}

public double frictionFactor(double rE, double d, double rough)
{
    double f, f1, f2 = 0;

    if(rE <= laminar){
        f = laminarFunc(rE);
    }
    else if (rE > laminar && rE < turbulent)
    {
        f1 = laminarFunc(rE);
        f2 = coleBrokFunc(rE, rough, d);

        f = interpolate(rE, laminar, turbulent, f1, f2);
        linearInterpolation(rE, laminar, turbulent, f1, f2);
    }
    else
    {
        f = coleBrokFunc(rE, rough, d);
    }

    return f;
}

public double laminarFunc(double re)
{
    return ((double)(darcy / re));
}
public double coleBrokFunc(double re, double rrough, double d)
{
    double tpfc = .00000001;
    double c = tpfc;
    double tpfc2 = 0;
    do
    {
        tpfc2 = tpfc;

        tpfc = 1/(Math.Pow(-2*Math.Log10((rrough/(3.7*d) +
            (2.51/(re*Math.Sqrt(tpfc2+c))))),2));
    }
    while(Math.Abs(tpfc-tpfc2) > .000001);

    return tpfc;
}

public double linearInterpolation(double x0, double x1, double y, double
y0, double y1)
{
    return x0 + (x1 - x0) * ((y - y0) / (y1 - y0));
}
}

```

## StochasticSim.cs

```
public class StochasticSim
{
    public SystemModel _system;
    public SystemModel _initialValues;
    public Pressure pressure;
    public Random rand;

    // Constants
    private double id_dev_pipe = 0.001;
    private double id_dev_hose = 0.001;
    private double id_dev_bha = 0.001;
    private double rough_dev_hose = 0.00001;
    private double rough_dev_pipe = 0.00001;
    private double rough_dev_bha = 0.0001;
    private double id_nozz_dev = 0.0001;

    public StochasticSim()
    {
        _system = new SystemModel();
        _initialValues = new SystemModel();
        rand = new Random();

        pressure = new Pressure(_system);
    }

    public SortedList<double, double> runSimulation(int simulations)
    {
        var res = stochSim(simulations);

        var finalList = groupUp(res);

        return finalList;
    }

    private double RandomNumberBetween(double minValue, double maxValue)
    {
        var next = rand.NextDouble();

        return minValue + (next * (maxValue - minValue));
    }
}
```

```

public List<double> stochSim(int sim)
{
    List<double> results = new List<double>();
    for (int i = 0; i < sim; i++)
    {
        resetSystem();
        foreach(Component c in _system._theSystem)
        {
            if (c.name.Equals("Pipe"))
            {
                c.innerDim += RandomNumberBetween(-id_dev_pipe,
id_dev_pipe);
                c.roughness += RandomNumberBetween(-rough_dev_pipe,
rough_dev_pipe);
            }
            if (c.name.Equals("Hose"))
            {
                c.innerDim += RandomNumberBetween(-id_dev_hose,
id_dev_hose);
                c.roughness += RandomNumberBetween(-rough_dev_hose,
rough_dev_hose);
            }
            if (c.name.Equals("BHA"))
            {
                c.innerDim += RandomNumberBetween(-id_dev_bha,
id_dev_bha);
                c.roughness += RandomNumberBetween(-rough_dev_bha,
rough_dev_bha);
            }
            if (c.name.Equals("Bit"))
                ((Drillbit)c).nozzleDim += RandomNumberBetween(-
id_nozz_dev, id_nozz_dev);
            pressure.estimatedPressure();
            //pressure.estimatedPressureTwoDifferentPumps();
            //pressure.estimatedPressureTwoIdenticalPumps();
            results.Add(pressure.estPressureSystem);
        }
    }
    return results;
}

```

```

private SortedList<double,double> groupUp(List<double> list)
{
    double min, max;
    min = 100;
    max = 0;

    foreach (double d in list)
    {
        if (d < min)
            min = d;
        if (d > max)
            max = d;
    }

    double sample = max - min;
    sample = sample / 20;

    SortedList<double, double> finalList = new SortedList<double,
double>();

    for (int i = 0; i < 20; i++)
    {
        double count = 0;

        foreach (double d in list)
        {
            if (d >= min + (sample * (i)) && d <= min + (sample * (i
+ 1)))
            {
                count++;
            }
        }

        finalList.Add(min + sample*i, count/list.Count);
    }

    return finalList;
}
public void adjustPipeRoughness(double val)
{
    if (val > 0)
        _system.pipe.roughness += _system.pipe.roughness * (1 - val);
    else
        _system.pipe.roughness -= _system.pipe.roughness * (1 + val);
}

public void adjustPipeInnerDim(double val)
{
    if (val > 0)
        _system.pipe.innerDim += _system.pipe.innerDim * (1 - val);
    else
        _system.pipe.innerDim -= _system.pipe.innerDim * (1 + val);
}

```

```

public void adjustPipeLength(double val)
{
    if (val > 0)
        _system.pipe.length += _system.pipe.length * (1 - val);
    else
        _system.pipe.length -= _system.pipe.length * (1 + val);
}

public void adjustHoseRoughness(double val)
{
    if (val > 0)
        _system.hose.roughness += _system.hose.roughness * (1 - val);
    else
        _system.hose.roughness -= _system.hose.roughness * (1 + val);
}

public void adjustHoseInnerDim(double val)
{
    if (val > 0)
        _system.hose.innerDim += _system.hose.innerDim * (1 - val);
    else
        _system.hose.innerDim -= _system.hose.innerDim * (1 + val);
}

public void adjustHoseLength(double val)
{
    if (val > 0)
        _system.hose.length += _system.hose.length * (1 - val);
    else
        _system.hose.length -= _system.hose.length * (1 + val);
}

public void adjustBHARoughness(double val)
{
    if (val > 0)
        _system.bha.roughness += _system.bha.roughness * (1 - val);
    else
        _system.bha.roughness -= _system.bha.roughness * (1 + val);

    _system.ann.roughness = _system.bha.roughness;
}

public void adjustBHAInnerDim(double val)
{
    if (val > 0)
        _system.bha.innerDim += _system.bha.innerDim * (1 - val);
    else
        _system.bha.innerDim -= _system.bha.innerDim * (1 + val);
}

```

```

public void adjustBHALength(double val)
{
    if (val > 0)
        _system.bha.length += _system.bha.length * (1 - val);
    else
        _system.bha.length -= _system.bha.length * (1 + val);
}

public void adjustBitInnerDim(double val)
{
    if (val > 0)
        _system.bit.nozzleDim += _system.bit.nozzleDim * (1 - val);
    else
        _system.bit.nozzleDim -= _system.bit.nozzleDim * (1 + val);
}

public void adjustBitNozzles(int val)
{
    _system.bit.nozzles = val;
}

public void resetSystemPipeValue()
{
    _system.pipe.length = _initialValues.pipe.length;
    _system.pipe.innerDim = _initialValues.pipe.innerDim;
    _system.pipe.roughness = _initialValues.pipe.roughness;
}

public void resetSystemHoseValue()
{
    _system.hose.length = _initialValues.hose.length;
    _system.hose.innerDim = _initialValues.hose.innerDim;
    _system.hose.roughness = _initialValues.hose.roughness;
}

public void resetSystemBHAValues()
{
    _system.bha.length = _initialValues.bha.length;
    _system.bha.innerDim = _initialValues.bha.innerDim;
    _system.bha.roughness = _initialValues.bha.roughness;
}

public void resetSystemBitValues()
{
    _system.bit.nozzles = _initialValues.bit.nozzles;
    _system.bit.nozzleDim = _initialValues.bit.nozzleDim;
}

public void resetIntervallValues()
{
    id_dev_pipe = 0.001;
    id_dev_hose = 0.001;
    id_dev_bha = 0.001;
    rough_dev_hose = 0.00001;
    rough_dev_pipe = 0.00001;
    rough_dev_bha = 0.0001;
    id_nozz_dev = 0.0001;
}

```

```

public void resetSystem()
{
    resetSystemBHAValues();
    resetSystemBitValues();
    resetSystemHoseValue();
    resetSystemPipeValue();
    resetIntervallValues();
}

/// <summary>
/// Get Set methods
/// </summary>

public double Id_dev_hose
{
    get { return id_dev_hose; }
    set { id_dev_hose = value; }
}

public double Id_dev_pipe
{
    get { return id_dev_pipe; }
    set { id_dev_pipe = value; }
}

public double Id_nozz_dev
{
    get { return id_nozz_dev; }
    set { id_nozz_dev = value; }
}

public double Id_dev_bha
{
    get { return id_dev_bha; }
    set { id_dev_bha = value; }
}

public double Rough_dev_hose
{
    get { return rough_dev_hose; }
    set { rough_dev_hose = value; }
}

public double Rough_dev_pipe
{
    get { return rough_dev_pipe; }
    set { rough_dev_pipe = value; }
}

public double Rough_dev_bha
{
    get { return rough_dev_bha; }
    set { rough_dev_bha = value; }
}
}

```



## KeyPair.cs

```
class KeyPair
{
    public double q {get; set;}
    public double p {get; set;}

    public KeyPair(double q, double p)
    {
        this.q = q;
        this.p = p;
    }
}
```

The code for the GUI is attached. See attachment.

## Appendix C

Chief Scientist Eric Cayeux, IRIS, Unpublished 2016

### General equation:

The angular movement of an elastic system is described by the following general partial differential equation (Newton's second law for moments):

$$\frac{\partial}{\partial x} \left( GJ(x) \frac{\partial \theta(x, t)}{\partial x} \right) + f(x, t) = I(x) \frac{\partial^2 \theta(x, t)}{\partial t^2}$$

with  $I(x) = \rho(x)J(x)$ ,  $\rho$  is the mass density,  $J$  is the second moment of area,  $f(x, t)$  the external torques, and  $G$  is the shear modulus.

### Decomposition of the system:

The power transmission chain is composed of different elements. Each element is characterized by:

- geometrical dimensions (length  $l_i$ , OD  $d_{oi}$ , ID  $d_{ii}$  which corresponds to a second moment of area  $J_i = \frac{\pi}{32} (d_{oi}^4 - d_{ii}^4)$ ),
- material properties (mass density  $\rho_i$ , shear modulus  $G_i$ , shear yield point  $\sigma_{sypl_i}$  and ultimate shear strength  $\sigma_{su_i}$ ),
- an angular position  $\theta_i$ ,
- a contact friction (static friction  $\mu_{si}$ , kinetic friction  $\mu_{ki}$ , Stribeck coefficient  $c_{vi}$  and viscous friction coefficient  $\eta_i$  such that the friction torque is  $\tau_{fi} = - \left( \mu_{ki} F_{ni} \frac{d_{oi}}{2} \text{sign}(\dot{\theta}_i) + \left( \mu_{si} F_{ni} \frac{d_{oi}}{2} - \mu_{ki} F_{ni} \frac{d_{oi}}{2} \right) e^{c_{vi}|\dot{\theta}_i|} \right) \text{sign}(\dot{\theta}_i) - \eta_i \dot{\theta}_i$
- a torque  $\tau_i$

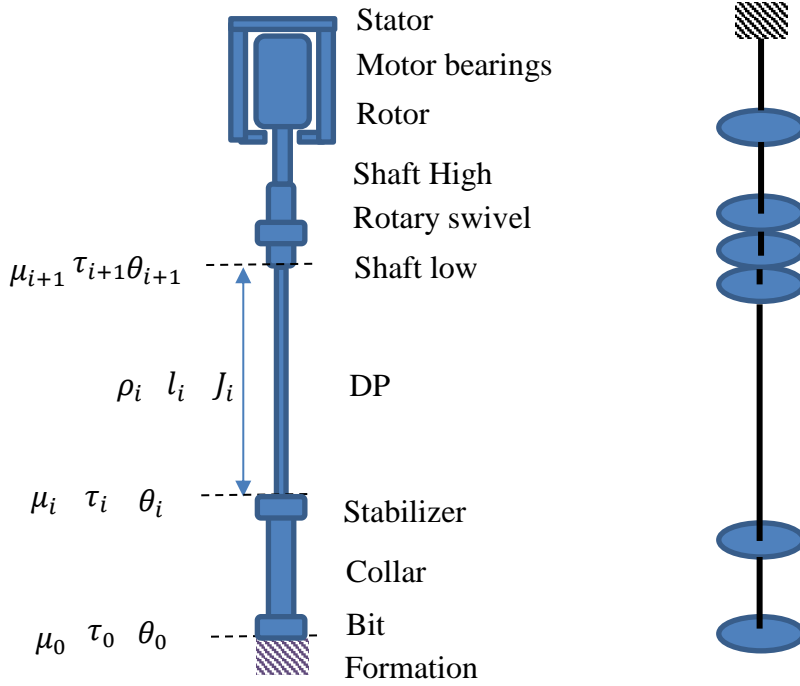
For instance the UiS DrillBotics 2016 rotating system can be decomposed in:

- bit
- collar
- stabilizer
- aluminum drill-pipe
- lower shaft
- rotary union
- upper shaft
- motor rotor with bearings

There are friction contacts at the bit, stabilizer, rotary union and the bearings supporting the motor rotor.

### Simplification:

A simplification consists in lumping each element in an equivalent disk having the same mass and the same second moment of area than the element itself and to connect each of the lumped disk to each other with a massless spring wire with a zero second moment of area but having a spring torsional constant  $k_i = \frac{G_i J_i}{l_i}$ .



### Equilibrium of one element

Considering a friction torque  $\tau_{f_i}$ , between element  $i$  and element  $i + 1$ , i.e. any elements that is neither the bit nor the rotor, the equilibrium is described by:

$$-\eta_i \frac{\partial \theta_i}{\partial t} + \tau_{f_i} - \frac{G_{i-1} J_{i-1}}{l_{i-1}} (\theta_i - \theta_{i-1}) + \frac{G_i J_i}{l_i} (\theta_{i+1} - \theta_i) = I_i l_i \frac{\partial^2 \theta_i}{\partial t^2}$$

We will now linearize in time, by considering discrete time instants (indexed  $j$ ) separated by a short time interval  $\Delta t$ . The Taylor expansion of the second derivative of the angle  $\theta_i$  is:

$$\frac{\partial^2 \theta_i}{\partial t^2} \approx \frac{\theta_{i,j+1} - 2\theta_{i,j} + \theta_{i,j-1}}{(\Delta t)^2}$$

And the Taylor expansion of the first derivative is:

$$\frac{\partial \theta_i}{\partial t} \approx \frac{\theta_{i,j+1} - \theta_{i,j}}{\Delta t}$$

It is then possible to rewrite the equilibrium of each elements as a linear function of the  $\theta_{i,j+1}$ ,  $\theta_{i,j}$  and  $\theta_{i,j-1}$ . At a given time step all the  $\theta_{i,j}$  and  $\theta_{i,j-1}$  are known (they are in the past) and therefore we can

easily calculate all the  $\theta_{i,j+1}$ . However, this explicit method tends to require very small time steps and can easily diverge due to numerical accuracy. It is much more stable to use a semi-implicit formulation as the one of Crank-Nicolson where the average of the sum of the moments between the current time step and the previous one is used as the second member:

$$\begin{aligned} I_i l_i \frac{\theta_{i,j+1} - 2\theta_{i,j} + \theta_{i,j-1}}{(\Delta t)^2} \\ = \frac{1}{2} \left( -\eta_i \frac{\theta_{i,j} - \theta_{i,j-1}}{\Delta t} + \tau_{i,j} + \frac{G_i J_i}{l_i} (\theta_{i+1,j} - \theta_{i,j}) - \frac{G_{i-1} J_{i-1}}{l_{i-1}} (\theta_{i,j} - \theta_{i-1,j}) \right. \\ \left. - \eta_i \frac{\theta_{i,j+1} - \theta_{i,j}}{\Delta t} + \tau_{i,j+1} + \frac{G_i J_i}{l_i} (\theta_{i+1,j+1} - \theta_{i,j+1}) - \frac{G_{i-1} J_{i-1}}{l_{i-1}} (\theta_{i,j+1} - \theta_{i-1,j+1}) \right) \end{aligned}$$

Which gives after regrouping:

$$\begin{aligned} - \left( \frac{G_{i-1} J_{i-1}}{2l_{i-1}} \right) \theta_{i-1,j+1} + \left( \frac{I_i l_i}{(\Delta t)^2} + \frac{G_i J_i}{2l_i} + \frac{G_{i-1} J_{i-1}}{2l_{i-1}} + \frac{\eta_i}{2\Delta t} \right) \theta_{i,j+1} - \left( \frac{G_i J_i}{2l_i} \right) \theta_{i+1,j+1} \\ = \frac{1}{2} \left( \eta_i \frac{\theta_{i,j-1}}{\Delta t} + \tau_{i,j} + \tau_{i,j+1} + \frac{G_i J_i}{l_i} (\theta_{i+1,j} - \theta_{i,j}) - \frac{G_{i-1} J_{i-1}}{l_{i-1}} (\theta_{i,j} - \theta_{i-1,j}) \right) \\ - \frac{I_i l_i}{(\Delta t)^2} (-2\theta_{i,j} + \theta_{i,j-1}) \end{aligned}$$

For the bit, the equilibrium is defined by:

$$\begin{aligned} I_0 l_0 \frac{\theta_{0,j+1} - 2\theta_{0,j} + \theta_{0,j-1}}{(\Delta t)^2} \\ = \frac{1}{2} \left( -\eta_0 \frac{\theta_{0,j} - \theta_{0,j-1}}{\Delta t} + \tau_{0,j} + \frac{G_0 J_0}{l_0} (\theta_{1,j} - \theta_{0,j}) - \eta_0 \frac{\theta_{0,j+1} - \theta_{0,j}}{\Delta t} + \tau_{0,j+1} \right. \\ \left. + \frac{G_0 J_0}{l_0} (\theta_{1,j+1} - \theta_{0,j+1}) \right) \end{aligned}$$

where  $\tau_{0,j}$  is the torque on bit. If the boundary condition is the torque at the bit, then this equation can be rearranged as a function of  $\theta_{0,j+1}$  and  $\theta_{1,j+1}$ :

$$\begin{aligned} \left( \frac{I_0 l_0}{(\Delta t)^2} + \frac{G_0 J_0}{2l_0} + \frac{\eta_0}{2\Delta t} \right) \theta_{0,j+1} - \left( \frac{G_0 J_0}{2l_0} \right) \theta_{1,j+1} \\ = \frac{1}{2} \left( \eta_0 \frac{\theta_{0,j-1}}{\Delta t} + \tau_{0,j} + \tau_{0,j+1} + \frac{G_0 J_0}{l_0} (\theta_{1,j} - \theta_{0,j}) \right) - \frac{I_0 l_0}{(\Delta t)^2} (-2\theta_{0,j} + \theta_{0,j-1}) \end{aligned}$$

However, if the boundary condition is the bit angle, then this equation can be rearranged as a function of  $\tau_{0,j+1}$  and  $\theta_{1,j+1}$ :

$$\begin{aligned} -\frac{1}{2} \tau_{0,j+1} - \frac{1}{2} \frac{G_0 J_0}{l_0} \theta_{1,j+1} \\ = \frac{1}{2} \left( -\frac{\eta_0}{\Delta t} (\theta_{0,j+1} - \theta_{0,j-1}) + \tau_{0,j} + \frac{G_0 J_0}{l_0} (\theta_{1,j} - \theta_{0,j} - \theta_{0,j+1}) \right) \\ - I_0 l_0 \frac{\theta_{0,j+1} - 2\theta_{0,j} + \theta_{0,j-1}}{(\Delta t)^2} \end{aligned}$$

The equilibrium of the next element is defined by:

$$\begin{aligned}
I_1 l_1 \frac{\theta_{1,j+1} - 2\theta_{1,j} + \theta_{1,j-1}}{(\Delta t)^2} &= \frac{1}{2} \left( -\eta_1 \frac{\theta_{1,j} - \theta_{1,j-1}}{\Delta t} + \tau_{1,j} - \frac{G_0 J_0}{l_0} (\theta_{1,j} - \theta_{0,j}) + \frac{G_1 J_1}{l_1} (\theta_{2,j} - \theta_{1,j}) \right. \\
&\quad \left. - \eta_1 \frac{\theta_{1,j+1} - \theta_{1,j}}{\Delta t} + \tau_{1,j+1} - \frac{G_0 J_0}{l_0} (\theta_{1,j+1} - \theta_{0,j+1}) + \frac{G_1 J_1}{l_1} (\theta_{2,j+1} - \theta_{1,j+1}) \right)
\end{aligned}$$

Which can be rearranged as:

$$\begin{aligned}
\left( \frac{I_1 l_1}{(\Delta t)^2} + \frac{\eta_1}{2\Delta t} + \frac{G_0 J_0}{2l_0} + \frac{G_1 J_1}{2l_1} \right) \theta_{1,j+1} - \frac{G_1 J_1}{2l_1} \theta_{2,j+1} &= \frac{1}{2} \left( \frac{\eta_1}{\Delta t} \theta_{1,j-1} + \tau_{1,j} + \tau_{1,j+1} - \frac{G_0 J_0}{l_0} (\theta_{1,j} - \theta_{0,j} - \theta_{0,j+1}) + \frac{G_1 J_1}{l_1} (\theta_{2,j} - \theta_{1,j}) \right) \\
- \frac{I_1 l_1}{(\Delta t)^2} (-2\theta_{1,j} + \theta_{1,j-1}) &
\end{aligned}$$

The equilibrium of the bottom part of the element just below the rotor is:

$$\begin{aligned}
I_n l_n \frac{\theta_{n,j+1} - 2\theta_{n,j} + \theta_{n,j-1}}{(\Delta t)^2} &= \frac{1}{2} \left( \eta_n \frac{\theta_{n,j} - \theta_{n,j-1}}{\Delta t} + \tau_{n,j} + \frac{G_n J_n}{l_n} (\theta_{n+1,j} - \theta_{n,j}) - \frac{G_{n-1} J_{n-1}}{l_{n-1}} (\theta_{n,j} - \theta_{n-1,j}) \right) \\
+ \eta_n \frac{\theta_{n,j+1} - \theta_{n,j}}{\Delta t} + \tau_{n,j+1} + \frac{G_n J_n}{l_n} (\theta_{n+1,j+1} - \theta_{n,j+1}) & \\
- \frac{G_{n-1} J_{n-1}}{l_{n-1}} (\theta_{n,j+1} - \theta_{n-1,j+1}) &
\end{aligned}$$

But  $\theta_{n+1,j+1}$  is known as it is imposed by the motor speed controller. The last equation can therefore be rewritten:

$$\begin{aligned}
- \left( \frac{G_{n-1} J_{n-1}}{2l_{n-1}} \right) \theta_{n-1,j+1} + \left( \frac{I_n l_n}{(\Delta t)^2} + \frac{G_n J_n}{2l_n} + \frac{G_{n-1} J_{n-1}}{2l_{n-1}} - \frac{\eta_n}{2\Delta t} \right) \theta_{n,j+1} & \\
= \frac{1}{2} \left( -\eta_n \frac{\theta_{n-1,j-1}}{\Delta t} + \tau_{n,j} + \tau_{n,j+1} + \frac{G_n J_n}{l_n} (\theta_{n+1,j} - \theta_{n,j}) \right) & \\
- \frac{G_{n-1} J_{n-1}}{l_{n-1}} (\theta_{n,j} - \theta_{n-1,j}) + \frac{G_n J_n}{l_n} \theta_{n+1,j+1} - \frac{I_n l_n}{(\Delta t)^2} (-2\theta_{n,j} + \theta_{n,j-1}) &
\end{aligned}$$

The equilibrium of the top of the last element below the rotor can be written as:

$$\begin{aligned}
I_{rotor} \frac{\theta_{n+1,j+1} - 2\theta_{n+1,j} + \theta_{n+1,j-1}}{(\Delta t)^2} &= \frac{1}{2} \left( -\eta_{n+1} \frac{\theta_{n+1,j} - \theta_{n+1,j-1}}{\Delta t} + \tau_{n+1,j} - \frac{G_n J_n}{l_n} (\theta_{n+1,j} - \theta_{n,j}) \right) \\
- \eta_{n+1} \frac{\theta_{n+1,j+1} - \theta_{n+1,j}}{\Delta t} + \tau_{n+1,j+1} - \frac{G_n J_n}{l_n} (\theta_{n+1,j+1} - \theta_{n,j+1}) &
\end{aligned}$$

For this equation, we know the  $\theta_{n+1,j}$  and  $\theta_{n+1,j+1}$  as they are defined by the motor controller, and the unknown are the angle  $\theta_{n,j+1}$  and the motor torque  $\tau_{n+1,j+1}$ . The last equation can be rearranged as:

$$\begin{aligned} & -\frac{1}{2} \frac{G_n J_n}{l_n} \theta_{n,j+1} - \frac{1}{2} \tau_{n,j+1} \\ & = \frac{1}{2} \left( -\eta_{n+1} \frac{\theta_{n+1,j+1} - \theta_{n+1,j-1}}{\Delta t} + \tau_{n,j} - \frac{G_n J_n}{l_n} (\theta_{n+1,j} - \theta_{n,j} + \theta_{n+1,j+1}) \right) \\ & \quad - I_{rotor} \frac{\theta_{n+1,j+1} - 2\theta_{n+1,j} + \theta_{n+1,j-1}}{(\Delta t)^2} \end{aligned}$$

To determine all the  $\theta_{i,j+1}, \forall i < n$ , and  $\tau_{n,j+1}$  we can solve the following system of equations:

$$\begin{pmatrix} a_{0,0} & a_{0,1} & 0 & 0 & 0 & 0 \\ a_{1,0} & a_{1,1} & a_{1,2} & 0 & 0 & 0 \\ 0 & a_{2,1} & a_{2,2} & a_{2,3} & 0 & 0 \\ 0 & 0 & a_{3,2} & a_{3,3} & a_{3,4} & 0 \\ 0 & 0 & 0 & a_{4,3} & a_{4,4} & 0 \\ 0 & 0 & 0 & 0 & a_{5,4} & a_{5,5} \end{pmatrix} \begin{pmatrix} \theta_{0,j+1} \\ \theta_{1,j+1} \\ \theta_{2,j+1} \\ \theta_{3,j+1} \\ \theta_{4,j+1} \\ \tau_{n,j+1} \end{pmatrix} = \begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \end{pmatrix}$$

$$\begin{aligned} \text{Where } a_{0,0} &= \left( \frac{l_0 l_0}{(\Delta t)^2} + \frac{G_0 J_0}{2l_0} + \frac{\eta_0}{2} \right), a_{0,1} = -\left( \frac{G_0 J_0}{2l_0} \right), \forall 0 < i < n, a_{i,i-1} = -\frac{G_{i-1} J_{i-1}}{2l_{i-1}}, a_{i,i} = \\ & \left( \frac{l_i l_i}{(\Delta t)^2} + \frac{G_i J_i}{2l_i} + \frac{G_{i-1} J_{i-1}}{2l_{i-1}} + \frac{\eta_i}{2\Delta t} \right), a_{i,i+1} = -\frac{G_i J_i}{2l_i}, a_{n,n-1} = -\frac{G_{n-1} J_{n-1}}{2l_{n-1}}, a_{n,n} = \frac{l_n l_n}{(\Delta t)^2} + \frac{G_n J_n}{2l_n} + \frac{G_{n-1} J_{n-1}}{2l_{n-1}} + \\ & \frac{\eta_n}{2\Delta t}, a_{n+1,n} = -\frac{1}{2} \frac{G_n J_n}{l_n}, a_{n+1,n+1} = -\frac{1}{2}, c_0 = \frac{1}{2} \left( \eta_0 \frac{\theta_{0,j-1}}{\Delta t \Delta t} + \tau_{0,j} + \tau_{0,j+1} + \frac{G_0 J_0}{l_0} (\theta_{1,j} - \theta_{0,j}) \right) - \\ & \frac{l_0 l_0}{(\Delta t)^2} (-2\theta_{0,j} + \theta_{0,j-1}), \forall 0 < i < n-1, c_i = \frac{1}{2} \left( \eta_i \frac{\theta_{i,j-1}}{\Delta t} + \tau_{i,j} + \tau_{i,j+1} + \frac{G_i J_i}{l_i} (\theta_{i+1,j} - \theta_{i,j}) - \right. \\ & \left. \frac{G_{i-1} J_{i-1}}{l_{i-1}} (\theta_{i,j} - \theta_{i-1,j}) \right) - \frac{l_i l_i}{(\Delta t)^2} (-2\theta_{i,j} + \theta_{i,j-1}) \text{ and } c_n = \frac{1}{2} \left( \eta_n \frac{\theta_{n-1,j-1}}{\Delta t} + \tau_{n,j} + \tau_{n,j+1} + \right. \\ & \left. \frac{G_n J_n}{l_n} (\theta_{n+1,j} - \theta_{n,j}) - \frac{G_{n-1} J_{n-1}}{l_{n-1}} (\theta_{n,j} - \theta_{n-1,j}) + \frac{G_n J_n}{l_n} \theta_{n+1,j+1} \right) - \frac{l_n l_n}{(\Delta t)^2} (-2\theta_{n,j} + \theta_{n,j-1}) \text{ and } c_{n+1} = \\ & \frac{1}{2} \left( -\eta_{n+1} \frac{\theta_{n+1,j+1} - \theta_{n+1,j-1}}{\Delta t} + \tau_{n,j} - \frac{G_n J_n}{l_n} (\theta_{n+1,j} - \theta_{n,j} + \theta_{n+1,j+1}) \right) - I_{rotor} \frac{\theta_{n+1,j+1} - 2\theta_{n+1,j} + \theta_{n+1,j-1}}{(\Delta t)^2} \end{aligned}$$

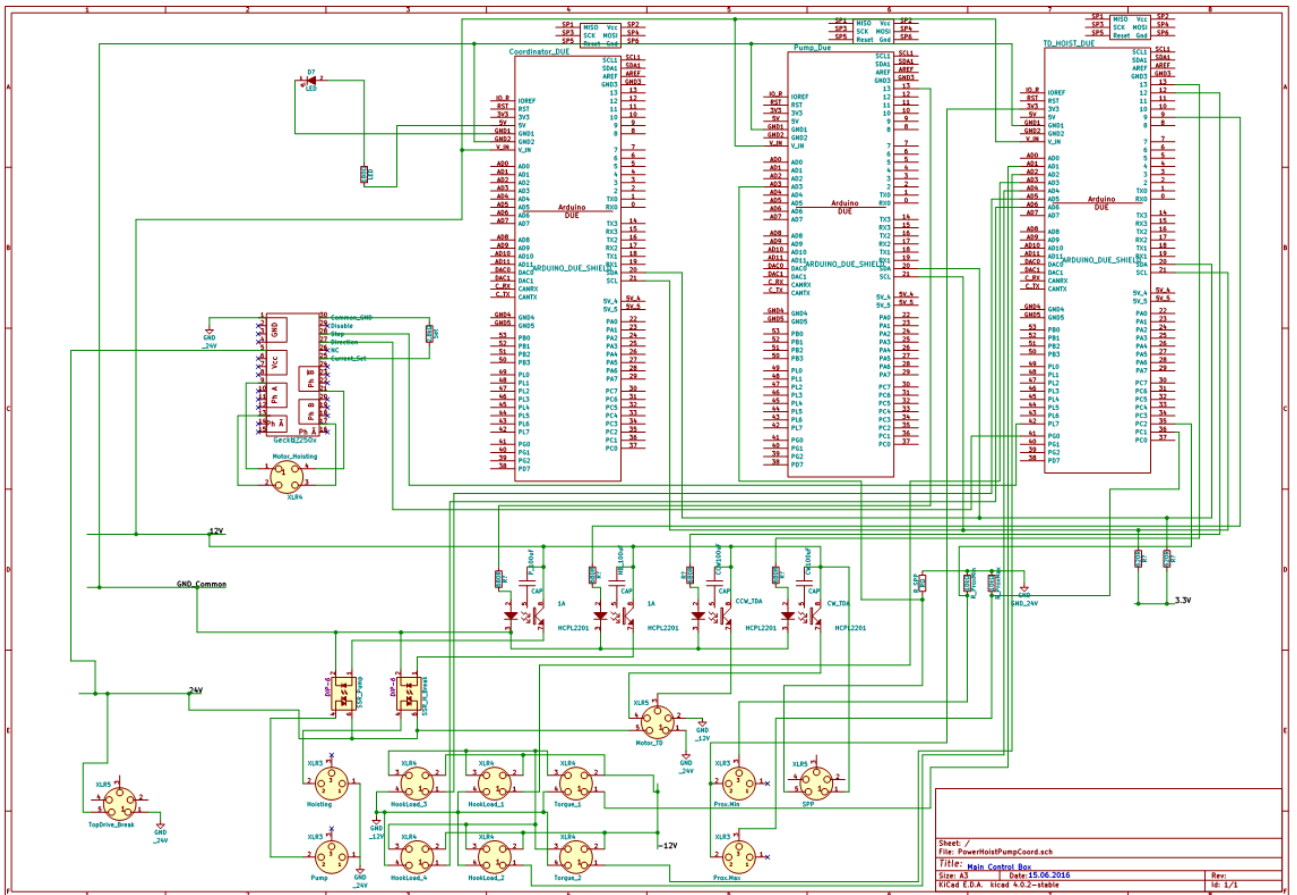
This is a tridiagonal linear system which can efficiently be solved numerically.

If the bottom boundary condition is the bit angle, then the unknown variables are  $\tau_{0,j+1}, 0 < i < n$   $\theta_{i,j+1}$  and  $\tau_{n,j+1}$ :

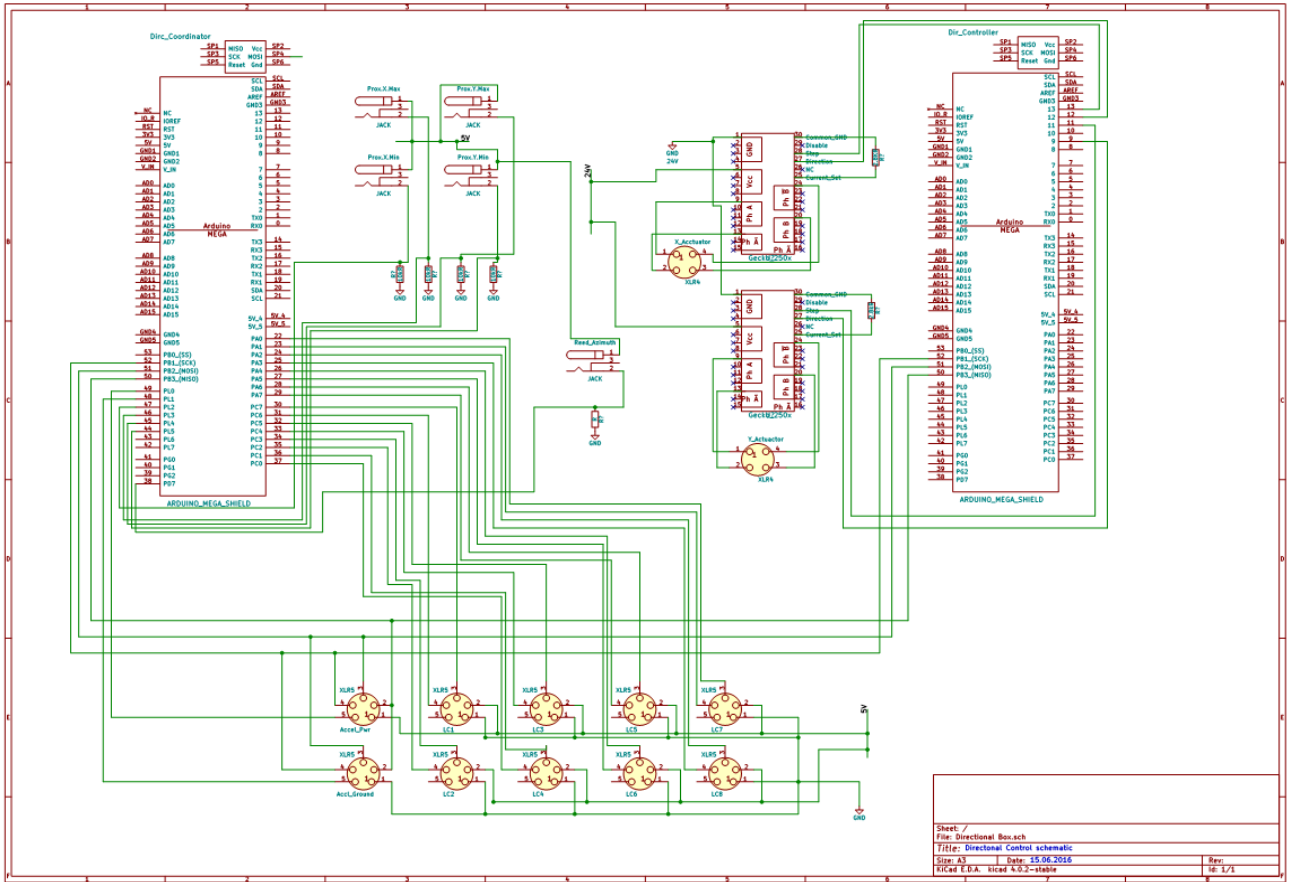
$$\begin{pmatrix} a_{0,0} & a_{0,1} & 0 & 0 & 0 & 0 \\ 0 & a_{1,1} & a_{1,2} & 0 & 0 & 0 \\ 0 & a_{2,1} & a_{2,2} & a_{2,3} & 0 & 0 \\ 0 & 0 & a_{3,2} & a_{3,3} & a_{3,4} & 0 \\ 0 & 0 & 0 & a_{4,3} & a_{4,4} & 0 \\ 0 & 0 & 0 & 0 & a_{5,4} & a_{5,5} \end{pmatrix} \begin{pmatrix} \tau_{0,j+1} \\ \theta_{1,j+1} \\ \theta_{2,j+1} \\ \theta_{3,j+1} \\ \theta_{4,j+1} \\ \tau_{n,j+1} \end{pmatrix} = \begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \end{pmatrix}$$

where  $a_{0,0} = -\frac{1}{2}$ ,  $a_{0,1} = -\frac{1}{2} \frac{G_0 J_0}{l_0}$  and  $c_0 = \frac{1}{2} \left( -\frac{\eta_0}{\Delta t} (\theta_{0,j+1} - \theta_{0,j-1}) + \tau_{0,j} + \frac{G_0 J_0}{l_0} (\theta_{1,j} - \theta_{0,j} - \theta_{0,j+1}) \right) - I_0 l_0 \frac{\theta_{0,j+1} - 2\theta_{0,j} + \theta_{0,j-1}}{(\Delta t)^2}$ ,  $a_{1,1} = \frac{l_1 l_1}{(\Delta t)^2} + \frac{\eta_1}{2\Delta t} + \frac{G_0 J_0}{2l_0} + \frac{G_1 J_1}{2l_1}$ ,  $a_{1,2} = -\frac{G_1 J_1}{2l_1}$  and  $c_1 = \frac{1}{2} \left( \frac{\eta_1}{\Delta t} \theta_{1,j-1} + \tau_{1,j} + \tau_{1,j+1} - \frac{G_0 J_0}{l_0} (\theta_{1,j} - \theta_{0,j} - \theta_{0,j+1}) + \frac{G_1 J_1}{l_1} (\theta_{2,j} - \theta_{1,j}) \right) - \frac{l_1 l_1}{(\Delta t)^2} (-2\theta_{1,j} + \theta_{1,j-1})$  the other coefficients being identical to the formulation based on boundary condition using the bit torque.

## Appendix D

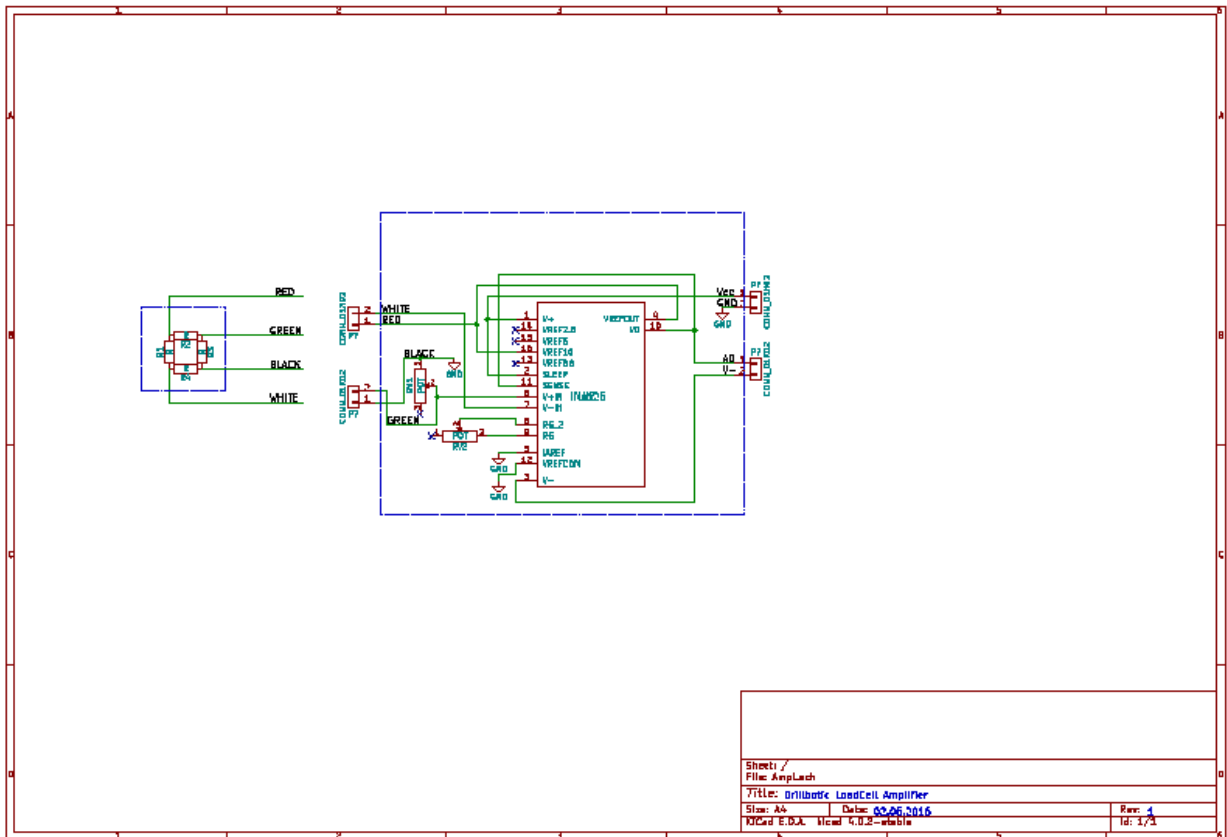


# Appendix E





# Appendix F

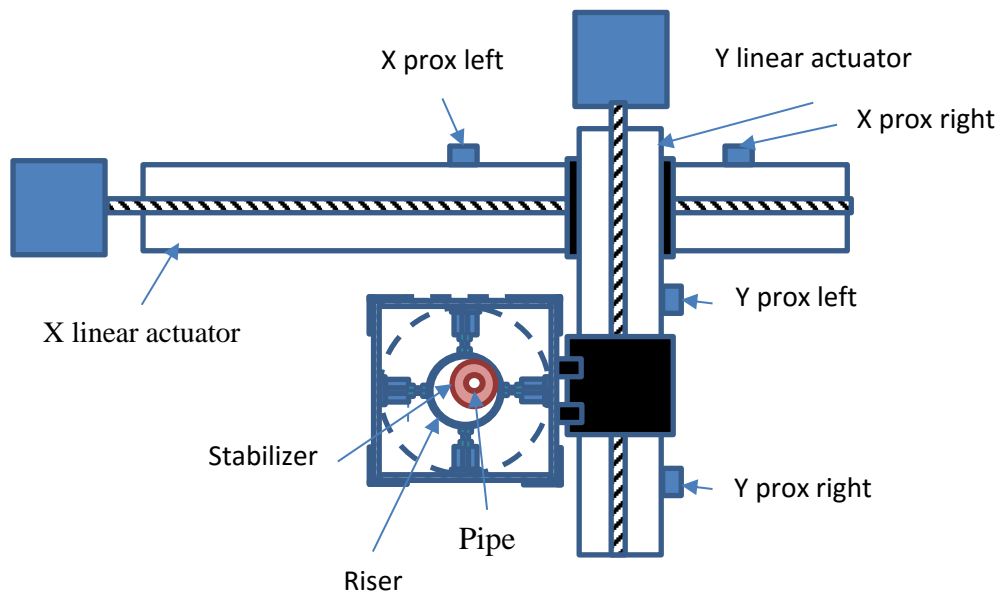


## Appendix G

### Design of the Directional Controller

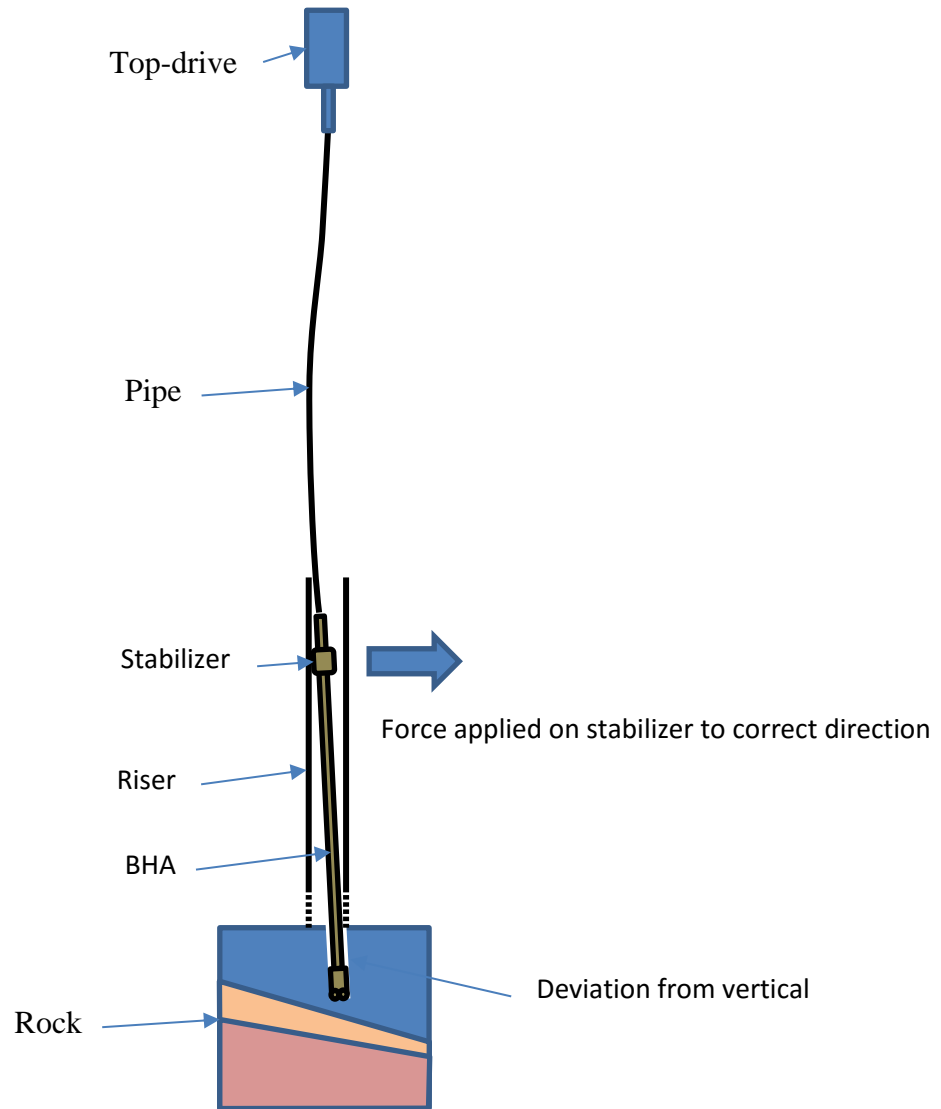
Chief Scientist Eric Cayeux, IRIS, Unpublished 2016

The directional controller steers two linear actuators in a X-Y configuration, i.e. perpendicular to each other, which are used to displace horizontally a riser (see **Error! Reference source not found.**).



#### X-Y LINEAR ACTUATORS AND ASSOCIATED RISER.

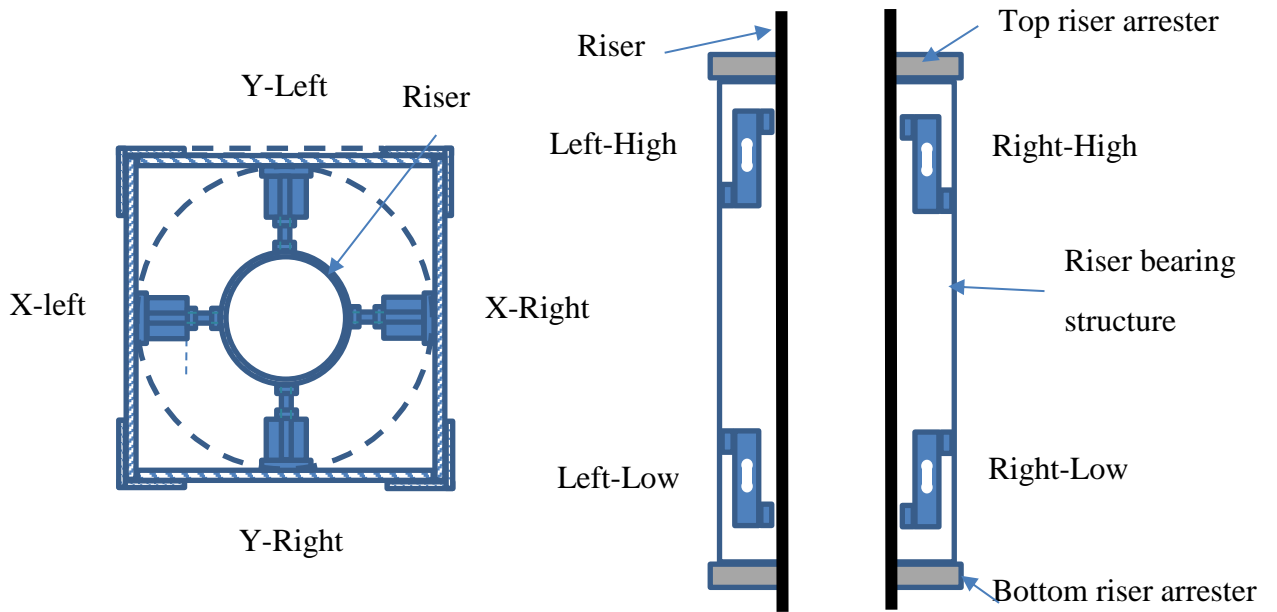
By the displacement of the riser, it is possible to exert a side force on a stabilizer at the top of the BHA and therefore actively control the drilling direction (see **Error! Reference source not found.**). By all practical means, this is the principle of a push the bit rotary steerable system, except that this is not the stabilizer that pushes on the borehole, but that the riser that pushes on the stabilizer.



**AN INVERTED RSS WHERE THE RISER APPLIES A FORCE ON THE STABILIZER**

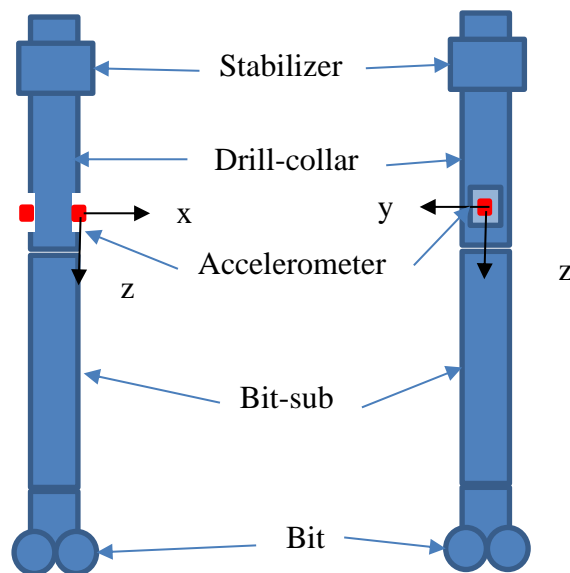
The riser is equipped with 8 load cells that read the current force applied on the stabilizer. These load cells are denoted as (see **Error! Reference source not found.**):

- X-left-high
- X-left-low
- X-right-high
- X-right-low
- Y-left-high
- Y-left-low
- Y-right-high
- Y-right-low



**POSITION OF THE LOAD-CELLS INSIDE THE BEARING STRUCTURE OF THE RISER**

The BHA is equipped with 3D accelerometers placed at mid distance along the BHA on each side of the drill-collar. These accelerometers are used to measure the inclination and the level of vibration.



In order to obtain an orientation, i.e. a toolface, of the accelerometer while the drill-string rotates, there is a proximity sensor that detects each complete rotation. In addition, the drilling controller send the current top-drive rotation angle at regular interval as a discrete value (10000 samples per revolution).

**Main functions of the controller**

The directional controller has two functions.

- Information source
- Directional control

### **Information source**

One of the function of the controller is to provide information to the rest of the system. This information is:

- The current inclination of the BHA
- The current azimuth of the BHA
- The current RMS (root mean square) value of vibrations (combined torsional and lateral)
- The side force applied on the stabilizer
- The direction of the side force applied on the stabilizer

### **Directional control**

To correct any deviations from vertical, the directional controller applies a side force on the stabilizer in the opposite direction to the current borehole deviation. This will be referred as “riser steering mode”. However, if the borehole does not deviate from vertical, no side force is applied. Similarly, when the drill-string is not drilling, the side force on the stabilizer is removed to facilitate the axial and rotational movement of the BHA. Removing the side force on the stabilizer will be referred as “riser neutralization mode”. In order to place the riser in neutral position, it is necessary to calibrate the position of the riser. This activity will be called “riser calibration mode”.

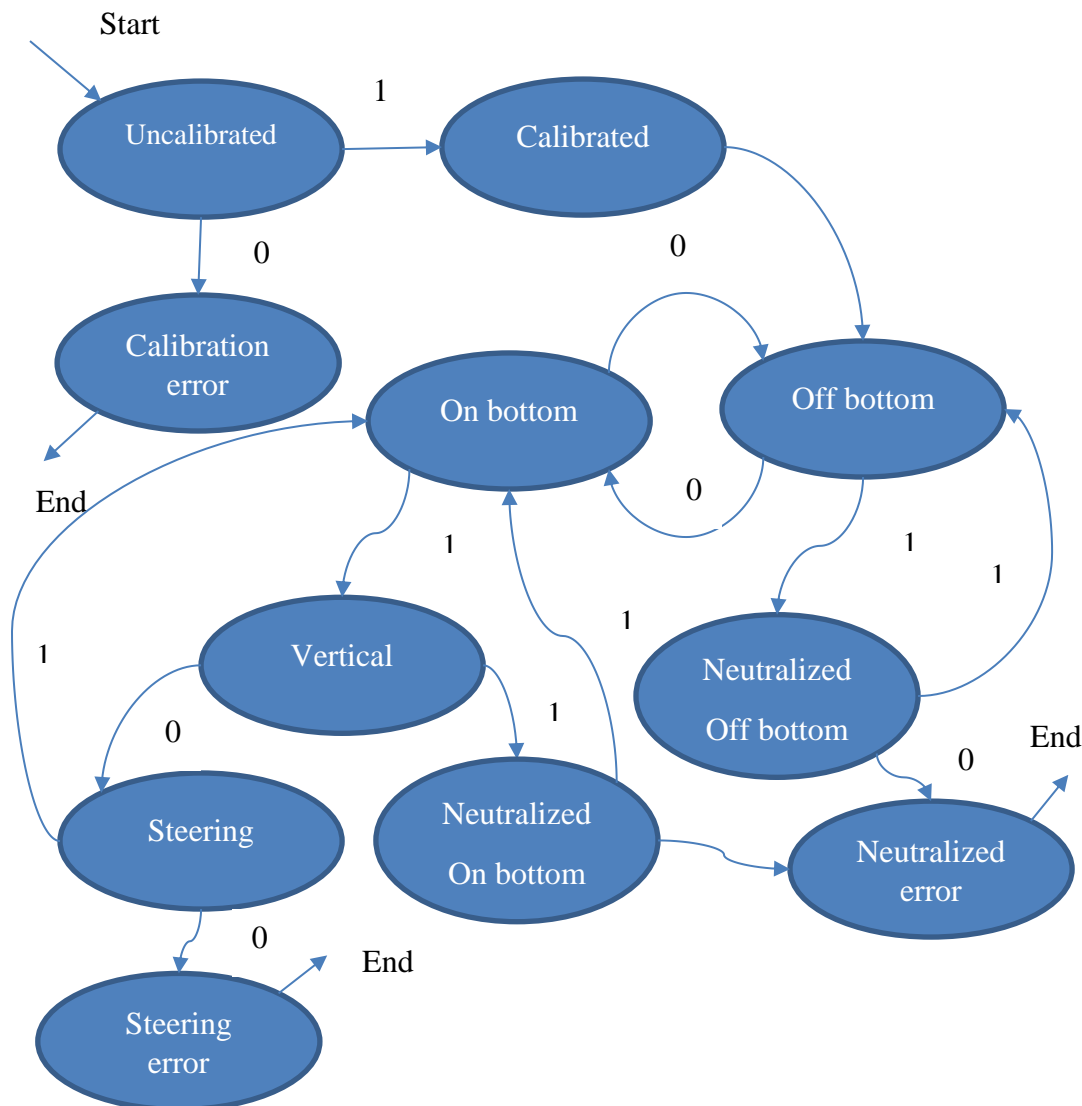
The system has therefore the following main states:

- Riser position uncalibrated
- Riser position calibrated
- On bottom
- Off bottom
- Directional control
- Neutral mode

A more detailed deterministic finite automaton is given below. There are three possible fatal exits:

- Error while calibrating
- Error while neutralizing
- Error while steering

Otherwise, the finite automaton always starts by calibrating and then continues to the “off bottom” state which enforces a neutralization of the riser. Thereafter, the system can flip between the “on bottom” and “off bottom” states. Each time, it goes to the “off bottom”, it performs a neutralization. So if the drilling controller lift the bit off bottom, the directional controller will automatically neutralize the riser position. When the state changes to “on bottom”, the system check the verticality. If it is vertical, then it neutralizes the riser and returns to the “on bottom” state, to be ready for a new evaluation. If it is not vertical, it applies a steer command and return to the “on bottom” state to start a new evaluation.



### Detailed description of each functions

#### Processing of Accelerometer Measurements

The acceleration observed by the sensor has the following origins:

- Gravitational field of the earth
- Axial vibrations like the one due to bit bouncing. It is a purely translational acceleration.
- Rotation of the pipe around its axis. It is a purely rotational acceleration which has two components: a centrifugal acceleration and a Euler acceleration. The Coriolis acceleration is necessarily zero as the distance of the sensor to the center of rotation cannot change.
- Lateral movement of the pipe in the overgauge hole. It is a combination of a translational and rotational acceleration. The rotational acceleration has potentially the all three sources, i.e. centrifugal, Coriolis and Euler.

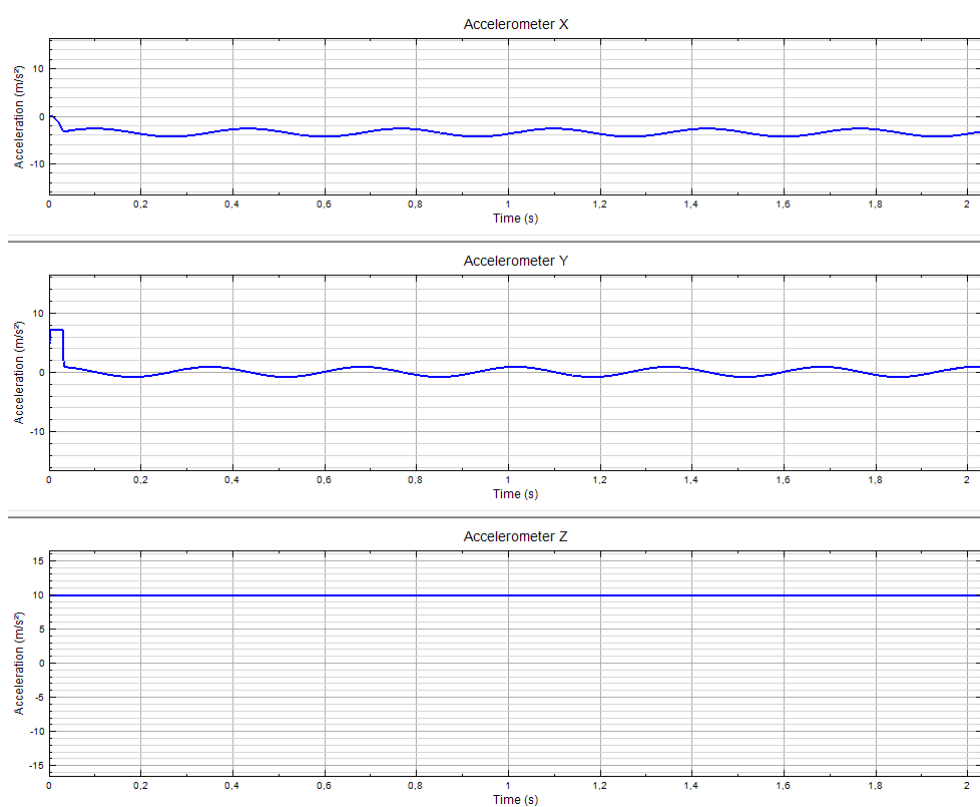
In the absence of vibrations and with a uniform pipe rotation, the components of the acceleration observed by the sensor are:

$$\begin{cases} \gamma_x = g \sin \beta \sin \varphi - r \dot{\theta}^2 \\ \gamma_y = g \sin \beta \cos \varphi \\ \gamma_z = g \cos \beta \end{cases}$$

where  $g$  is the earth gravitational acceleration,  $\beta$  is the inclination,  $\varphi$  is the toolface angle,  $r$  is the distance of the sensor to the pipe center,  $\dot{\theta}$  is the angular velocity of the pipe rotation.

If furthermore, the inclination is zero, then  $\gamma_x = -r \dot{\theta}^2$  reads only the centrifugal acceleration,  $\gamma_y$  is zero and  $\gamma_z$  is the gravitational field  $g$ .

However, when there is an inclination larger than zero, then  $\gamma_x$  and  $\gamma_y$  oscillates with a period  $\frac{2\pi}{\dot{\theta}}$  (see **Error! Reference source not found.**). For a small inclination,  $\cos \beta$  can be approximated to  $1 - \frac{\beta^2}{2!}$  and since  $\beta$  is small,  $\beta^2$  is even smaller, so it will be hardly possible to measure the inclination using  $\gamma_z$ . On the other hand,  $\sin \beta$  can be approximated to  $\beta$  for values close to 0. So both  $\gamma_x$  and  $\gamma_y$  can provide an estimation of the inclination by estimating the amplitude of the oscillations, except that with  $\gamma_x$ , we shall also remove the effect of the centrifugal acceleration.



#### ACCELEROMETER READINGS WHEN ROTATING UNIFORMLY AT 180RPM WITH AN INCLINATION OF 5°.

As we use a stepper motor to drive the drill-string, there are large angular rotational accelerations for each new step of the motor. This translates as follow on the equations:

$$\begin{cases} \gamma_x = g \sin \beta \sin \varphi - r \dot{\theta}^2 \\ \gamma_y = g \sin \beta \cos \varphi + r \ddot{\theta} \\ \gamma_z = g \cos \beta \end{cases}$$

The  $\gamma_y$  acceleration gets a Euler acceleration term in addition to the component due to the gravitational field.  $\ddot{\theta}''$  is alternating from positive to negative as the stepper motor accelerates and decelerate for each motor step. The frequency of that signal is equal to the motor rotational speed multiplied by the number of steps per revolution, i.e. at 180 rpm (3Hz) the Euler acceleration due to a stepper motor with 10000 steps per revolution has a frequency of 30kHz (see **Error! Reference source not found.**).

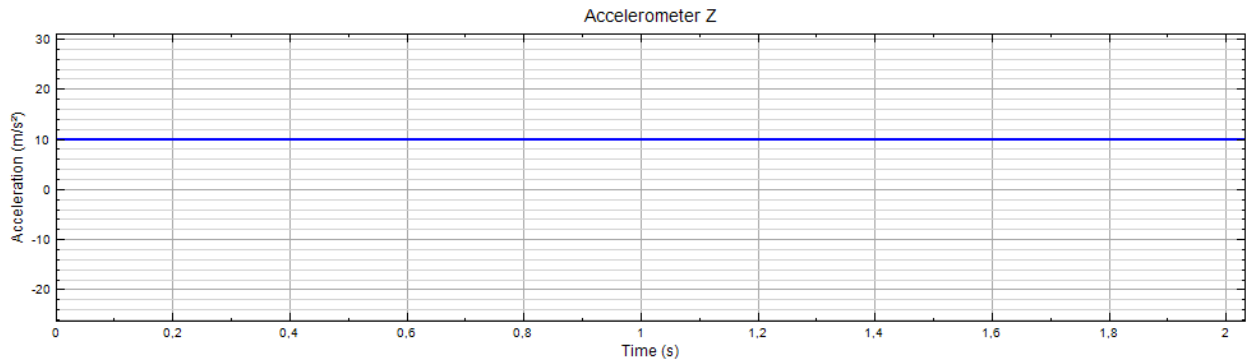
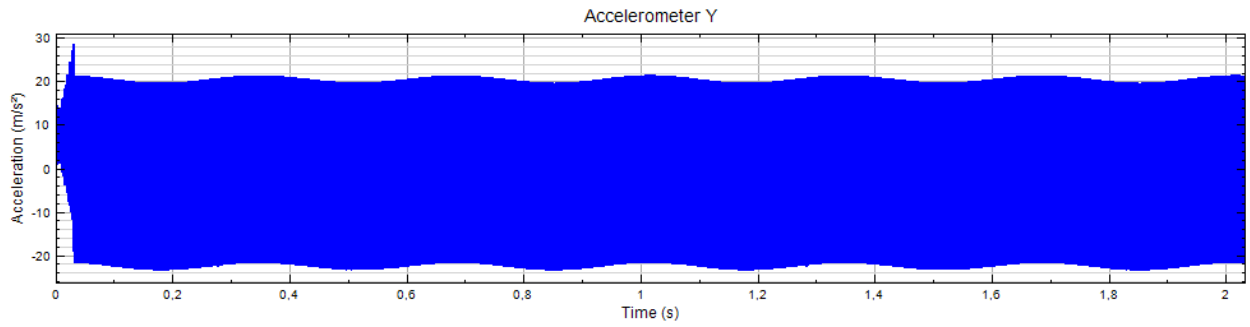
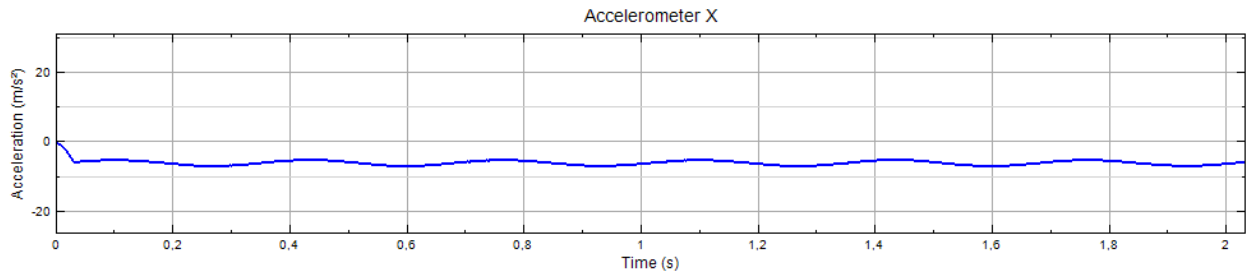
**ACCELERATIONS MEASURED WHEN ROTATING A DRILL-STRING AT 180RPM WITH A STEPPER MOTOR (10000 STEPS PER REVOLUTION) WITH AN INCLINATION OF 5°.**

Finally, if there are lateral and axial vibrations, the accelerations observed by the sensor are:

$$\begin{cases} \gamma_x = g \sin \beta \sin \varphi - r'' \dot{\theta}''^2 + (\ddot{r}' - r' \dot{\theta}'^2) \cos \theta'' + (r' \ddot{\theta}' + 2\dot{r}' \dot{\theta}') \sin \theta'' \\ \gamma_y = g \sin \beta \cos \varphi + r'' \ddot{\theta}' - (\ddot{r}' - r' \dot{\theta}'^2) \sin \theta'' + (r' \ddot{\theta}' + 2\dot{r}' \dot{\theta}') \cos \theta'' \\ \gamma_z = g \cos \beta + \gamma_a \end{cases}$$

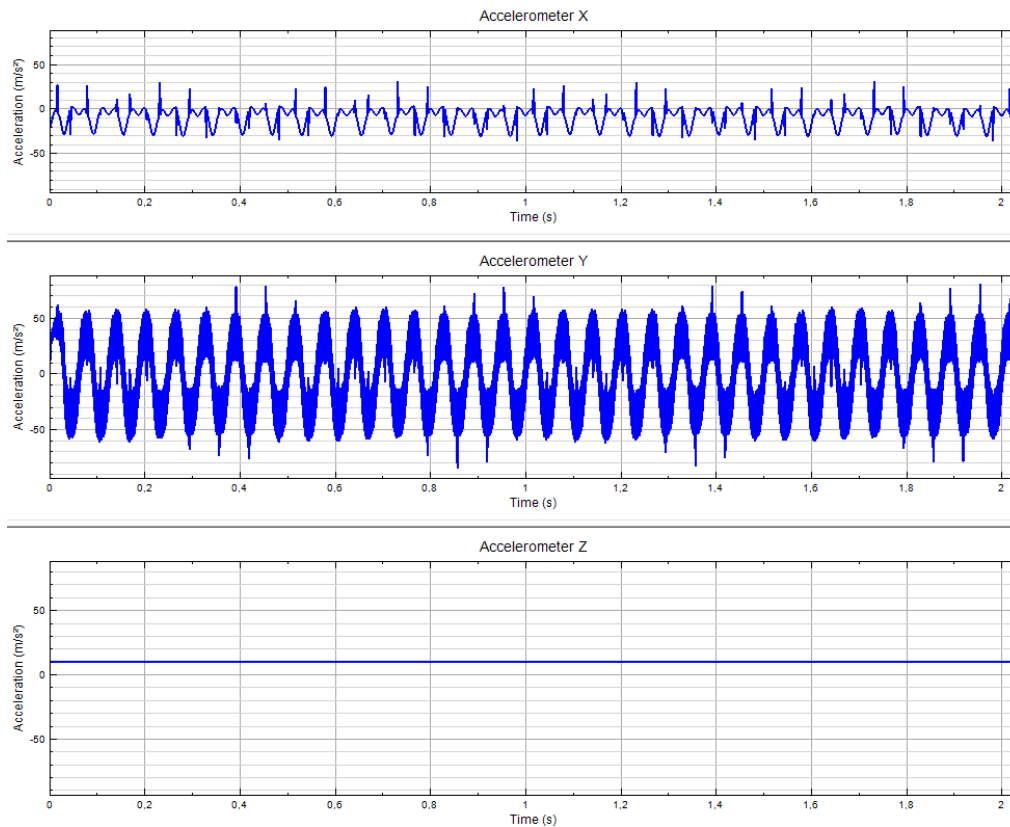
With forward whirl, the rotation of the pipe inside the borehole is dominated by friction against the wall and therefore the accelerations are relatively small (see **Error! Reference source not found.**).





**FORWARD WHIRL (60RPM WITH A 2MM OVERGAGE) WHILE ROTATING THE PIPE AT 180RPM WITH AN INCLINATION OF 5°**

However, backward whirl is predominantly a shock phenomenon and the resulting accelerations are far greater (see **Error! Reference source not found.**).



**BACKWARD WHIRL (60 RPM, 16 SHOCKS PER REVOLUTION IN A 2 MM OVERGAGED HOLE) WHILE ROTATING THE PIPE AT 180RPM IN HOLE INCLINED AT 5°**

In view of this analysis, the following strategy is used to extract the inclination and the vibration level. We record the x and y accelerations in a buffer. Then we apply a fast Fourier transform (FFT) to convert to the frequency domain. In the frequency domain, we cancel all the frequencies lower and higher than the current rotational speed and we transform back to the time domain using the inverse FFT. Finally, we calculate the root mean square (RMS) of the resulting signal which gives the amplitude of the sine function as  $MS = \frac{g \sin \beta}{\sqrt{2}} \Leftrightarrow \sin \beta = \frac{RMS}{g} \sqrt{2}$ . To estimate the vibration level, we remove the frequencies around the rotational speed of the drill-string instead. Then, the RMS value of the resulting signal after applying the inverse Fourier transform gives an estimation of the vibration level.

One Arduino (AT Mega 2560) is dedicated to do this processing. The accelerometer is connected to the board using an SPI link where the accelerometer is a slave. As there is only one master and multiple slaves in a SPI setup, and as the “coordinating” Arduino has to be the master since it is hooked up to an Ethernet shield (which is a slave) and the other Arduinos through SPI, the accelerometer has to be connected to the “coordinating” Arduino.

The range of the accelerometer is  $\pm 16g$  and the digital precision is 13 bits (i.e. resolution  $4mg$ ,  $0.03924m/s^2$ ). This Arduino needs the current rotational speed of the drill-string. This is provided by the UDP datagram sent by the “coordinating” Arduino of the drilling controller. It fills in return an estimation of the inclination and the RMS amplitude of the lateral and axial vibration levels into a UDP datagram.

There are basically two main conditions: either the drill-string does not rotate or the drill-string rotates.

If the drill-string does not rotate, then we read  $\gamma_x$  and  $\gamma_y$  and extract the inclination using  $\beta =$

$\sin^{-1} \frac{\sqrt{\gamma_x^2 + \gamma_y^2}}{g}$ . To remove possible sensor noise, we return the average over the last measurements (for instance 10 values). The lateral and axial vibrations are not estimated at all as they are unlikely to be of any significant magnitude (if any, they would correspond to the noise generated by hoisting system) and would be irrelevant as this quantity is used to determine when to stop the top-drive rotation and the top-drive is not rotating anyway.

If the drill-string rotates, then we do a Fourier transform analysis. As the Arduino has only 8kB of memory, we need to be cautious about memory consumption. The minimum rotational speed that would probably be considered is around 30rpm (0.5Hz, period of 2s). In order to extract the lowest frequencies, the time window size should be at least two times the longest period, i.e. 4s. The maximum rotational speed will probably not exceed 180rpm (3Hz, period of 0.33s). So with a 4s window size, we can hold about 12 periods when rotating at 180rpm. We expect the  $\gamma_y$  acceleration to contain a lot of high frequency noise due to the stepper motor. The minimum frequency of that noise is  $0.5 * 10000 = 5\text{kHz}$  and the maximum is  $3.0 * 10000 = 15\text{kHz}$ . As the maximum bandwidth of the sensor is 3.2 kHz, we will not be able to capture that information anyway (the Nyquist-Shannon theorem states that the sampling rate should be at least twice of the maximum frequency). Therefore, we will not use the  $\gamma_y$  information but instead we will focus on the  $\gamma_x$  readings. We need 3 arrays: 1 with the original measurements, 1 with the filtered measurements and 1 with the noise measurements. The acceleration values have a 13-bit precision and therefore can be stored on signed shorts (2 bytes). If we split the available memory in 4 (3/4 for the signal treatment and 1/4 for other information), one array is limited to 2kB, i.e. 1024 values. With a window size of 4s, the window time step is 3.90625ms and therefore the highest frequency manipulated by the FFT is 344Hz.

### Calibration

The calibration consists in a series of movement of the riser to determine the central axis and the riser overage compared to the stabilizer diameter. During the movements, if any of the proximity sensors (X-left, X-right, Y-left, Y-right) triggers then the calibration fails.

There are two possibilities at the start of the calibration:

- Either there is already a force on the riser
- Or there are no forces on the riser

If there is a force on the riser, then we move in the opposite direction to the force until the force drops below a minimum level. If the force does not decrease, then the calibration fails.

If there are no forces on the riser, then we move in any direction (for instance increasing x) until a force is detected.

When we have registered a force, then we move in the opposite direction to the force until we detect a new force. When starting, the force should decrease monotonically to value below a minimum threshold. If that is not the case, then the calibration fails. The distance between those two positions is twice the riser overage. Displacing the riser to half this distance make sure that the riser is in neutral position.

## **Appendix H**

*Unedited jury report*

### **General comments:**

#### **UIA**

Very professional and solid, well-built industrial prototype that actually worked.

HSE well covered

Well thought out in terms of construction/design, good electrical setup, and it worked, though indication of poor control during the short drilling test.

Good thinking with automatic optimization of ROP based on varying WOP and RPM

Large vibrations were indicated and the control of this seemed very coarse, and we were not sure if the rack and pinion setup also influenced the vibrations seen.

Novel to see the use of 3D printing to fast track components was good.

The use of the system for future use adding more sensors etc. had been thought through.

Data management seemed poor the impression was that data was not collected and stored in a data base for retrieval and learning/optimization.

The fluid rates seem very low..not sure how that was optimized.

HMI looked good

Kept within Budget

Had thought about mobility, and the rig showed good promise

Disappointed that they did not look at down-hole sensors, though this was seen as phase two.

#### **UIS**

UIS had a smaller team which we think limited **their** progress, presentation was rather hurried and one appeared made at the last minute, with lots of abbreviations which were not fully defined. Much more focus on the math and physics, and the use of the data to optimize.

Construction looked a lot less study and it would be interesting how it handles drilling vibration.

In terms of HSE there were too many open electrical boxes and loose wiring, and apparently no end stops, just an emergency shutdown.

Good use of heavy BHA to help avoid buckling.

Novel handling of the combination of gravity + centrifugal + Coriolis + Euler forces from downhole sensors including noise filtering.

Had good flow regimes, and seemed to have precise control

Had good alarm setups in software, though GUI /HMI is much more focused at engineers.

Well thought out time response strategy for shutdown of top drive.

Apparently had a self-calibrating system

Excellent that they have been looking at the use of downhole sensors, which allowed some form of directional control

Unfortunately over budget and they could not show the machine in operation.

It was apparent that the late change in the drill-pipe size effected UiS more because of processor choice.

#### **In Summary**

In summary we have to award it to UiA as they demonstrated the machine, as requested, though they did have some limitations and described above. UiS had good focus on the control and data management and less on the design aspects, though what they have achieved is more than good enough for the task. UiS more clearly Petroleum focused, whilst the UiA ended up with the best rig which was clearly very movable.

As we did not see the the UiS rig in operation ,we would still like to see it in motion at a future date!

One main thing is that this is an excellent initiative and should continue, as Norway should continue to have a leading position in Automation and Autonomy in Drilling, and we need the ideas and support from Academia.

Appendix H

