Master Thesis

Developing a knock-out code for production purposes

Rickard Johansson

Division of Machine Design • Department of Design Sciences

Faculty of Engineering LTH • Lund University • 2014

LUND UNIVERSITY

Wayne

A GE Energy Business

Developing a knock-out code for production purposes

Rickard Johansson

Division of Machine Design • Department of Design Sciences

Faculty of Engineering LTH • Lund University • 2014

Division of Machine Design, Department of Design Sciences
Faculty of Engineering LTH, Lund University
P.O. Box 118
SE-221 00 Lund
Sweden

ISRN LUTMDN/TMKT 15/5514 SE

Preface

I want to thank all the people who helped out with this project, especially my two supervisors Mattias Mårtensson at Wayne and Damien Motte at the Division of Machine Design. I also want to thank Wayne for giving me the opportunity to work on this project.

Finally I want to thank my family for their moral support and for always being encouraging during my studies and for keeping on bothering me with questions of how my work is going.

Lund, June 2014

Rickard Johansson

Abstract

This report is the product of my master thesis in technical design with a focus on product development. The project was done in co-operation with Wayne, who produces and sells fuel dispensers and are a part of General Electrics, at their Malmö office in Sweden. For their latest line of fuel dispensers, Helix, they were in need of a system for describing different configurations for the barrier, an essential part of the product in that it separates the mechanical and electrical components. This was to be done with consideration to a knock-out machine that is currently in development and that will automate the production of the barrier. The knock-out machine punches out holes on the metal barrier and which holes are to be punched out depends on the configuration of the fuel dispenser in question.

The project followed a linear path that started off with studying the systems that directs the configuration of the fuel dispenser and the barrier. With this base, the final concept a binary based code string that contains information of which holes are to be open on the barrier was developed and presented to Wayne.

The chosen concept has not yet been implemented in their system and testing may determine that changes need to be made to the code.

Keywords:

Wayne, product development, fuel dispenser, barrier, encryption

Sammanfattning

Denna examensarbetsuppsats beskriver processen som användes för att åt företaget Wayne skapa en kod som kan användas vid produktionen av deras senaste produktserie, Helix. Koden beskriver vilken konfiguration det är på barriären, en specifik del av bränsle-pumparna som Wayne producerar. Barriären har till uppgift att separera de mekaniska och de elektroniska delarna från varandra och beroende på vilken konfiguration som bränsle-pumpen har så skall olika slangar och kablar kunna dras igenom den.

Som riktlinje för har jag använt *Product design and development* av Karl T. Ulrich och Steven D. Eppinger. Projektet inleddes med att gå in djupare i hur konfigurationen för barriären är kontrollerad och även att lära mig tolka den information som ges från försäljningsavdelningen vid en inkommen order. Företagets nomenklatur gicks igenom och riktlinjer hämtades härifrån för att kunna sätta upp en tolkning av den modell-sträng som kommer med varje order.

Detta ledde till att vi kunde skapa en bas från vilken ett antal koncept genererades och vägdes mot varandra. Koncepten skapades för att hantera den mängd av information som behövs för att beskriva barriären på ett behändigt och lätt hanterligt sätt. Koncepten vägdes sedan mott varandra med hjälp av de riktlinjer som Wayne satt upp och det koncept som fick högst valdes ut.

Det koncept som valdes att gå vidare med använder en binär bas där varje hål på barriären ges en indikator på om den är öppen eller stängd. Detta ger en lång sträng av ettor och nollor som sedan delas upp i grupper om fem. Varje grupp omvandlas sedan till decimal-form och därifrån med en 32-bitars kryptering fås de slutgiltiga tecken bestående av de siffror och bokstäver som utgör koden.

De system som relaterar till projektet men som inte har kunnat utvecklas i detta examensarbete har dock gåtts igenom och beslutsunderlag har lagts fram för att Wayne lättare skall kunna ta beslut angående dessa. Olika exempel visas på både algoritmer för kontroll-siffror och avläsnings-anordningar. Deras för- och nackdelar listas upp mot varandra och synpunkter ges om vilket system som är mest lämpligt.

Table of Contents

1 Introduction

1.1 Background

This project was done on behalf of Wayne (formerly known as Dresser Wayne), a part of General Electrics, Wayne [1] is an American company with one of its offices situated in Malmö, Sweden. Wayne has since 1891 developed and produced fuel dispensers for gas stations and is one of the world leaders in its field.

In 2012 Wayne developed a new line of dispensers called Helix [2], as can be seen in figure 1.1, which are developed to be modular so as to reduce the effect of discrepancies between regions. The Helix line is the platform to which this thesis is centered around and the project is based on solving a problem in regards to the production of this product.

Figure 1.1 The Wayne Helix family

1.2 Problem description

The main body of the fuel dispenser is comprised of an upper and a lower half, in the upper half all the electronic systems are housed and in the lower part we find the mechanical systems (such as pumps and motors). Separating the two compartements is a barrier and this barrier is the focus of the project. The barrier can be seen in figure 1.2.

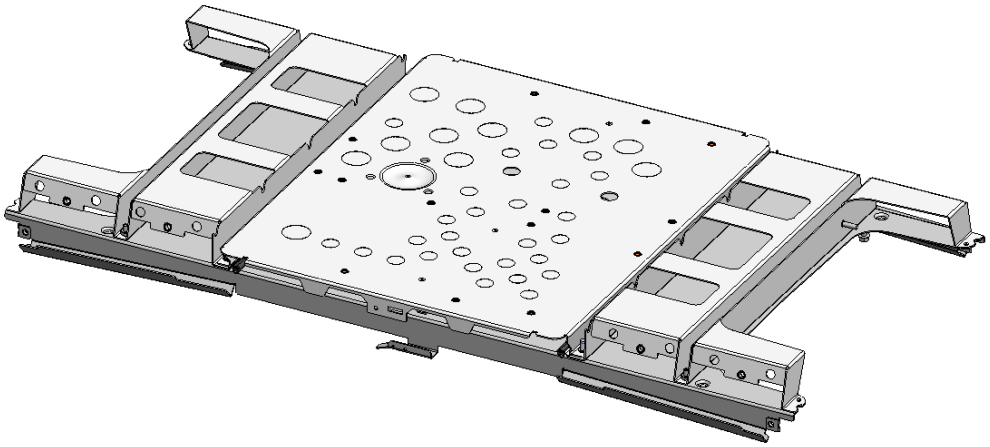


Figure 1.2 CAD drawing of the barrier

The barrier needs to isolate the two compartments from each other to not let fumes or other hazards through which could cause an explosion or a malfunction in the electrical system. At the same time the two compartments needs to be connected with hoses and cables to operate the fuel dispenser. This causes each individual configuration of the dispenser to require different configurations on the barrier. There are 2 pre-determined layouts of the barrier labeled IEC (International Electro-technical Commission) and UL (Underwriters Laboratory) and which one is used depends on the sales region. The IEC barrier has 41 pre-punched holes and UL has 33, these holes are all re-sealed until the production of each individual product, where the required ones are opened up again for usage. The blueprints for both barriers can be seen in Appendix B.

In the current production environment a worker on the line punches the required holes for each individual barrier by hand following an attached schematic accompanying each needed configuration. The company is currently working on a machine that will be able to knock-out the required holes in the production line, but the machine will require a different means of receiving instructions to the one that is used currently.

1.3 Aim of the project

Wayne is looking to create a code consisting of numerals and/or letters that when translated will show which holes need to be open for an individual barrier. Several concepts need to be generated and evaluated against each other to find the most suitable solution. Furthermore the code needs to be able to be used with some form of reading device like a barcode or RFID and not just having to be typed in manually, therefore some alternatives should be suggested, along with their pros and cons.

1.4 Approach

For the most part the development methodology follows what was written by Karl T. Ulrich and Steven D. Eppinger in *"Product design and development"* [3]. This methodology has been the foundation for how the work was disposed and the how the decisions in the project where made.

Before we could start considering most of our concepts we needed to get to know how the configurations for the barriers are determined. We therefore need to learn which parts of the nomenclature that are coupled with opening holes.

The next part of the project required creating several possible concepts for the code and weighing them against each other. The chosen concept will then be finalized into a solution that can be presented to Wayne.

Furthermore groundwork needs to be set up for Wayne to be able to make a decision on a system for control numbers for the code and also for what kind of reading device that should be used.

1.5 Specifications

There are several guidelines set up by Wayne as to how they want the code to be.

- The code can only consist of numerals and/or letters which should either be upper- or lower case.
- The code should be short so to reduce the difficulty of handling it manually.
- The code should be flexible in case Wayne wants to make changes to the barrier or the product in general.
- The preparation needed before the code can be used should be minimal.
- The code should be readable both ways so you can both translate your hole-configuration to the code and read the hole-configuration from the code.

These guidelines will be our baseline for evaluating the different concept against each other in decision process.

1.6 Disposition

1. Introduction
2. Getting to know the system
3. Concept generation
4. Choosing our concept
5. Moving forward with the concept
6. Check digit
7. Reading device
8. Reflections of the project

2 Getting to know the system

In order to understand what causes the individual configurations of the barrier, the nomenclature and hydraulic configurations needed to be interpreted.

2.1 Incoming order

When a costumer of Wayne wants to order one of their products they use a template setup by Wayne's sales department using a program called Combinum (In the near future General Electrics will change this to another program called Big machines). Each choice that the costumer makes opens up new alternatives to choose from depicting how they want their fuel pump. When all choices are made the program generates a lengthy model string (figure 2.1) that contains all the information needed to produce the order, the issue for this project is that the information of which holes needs to be open on the barrier is spread throughout the string.

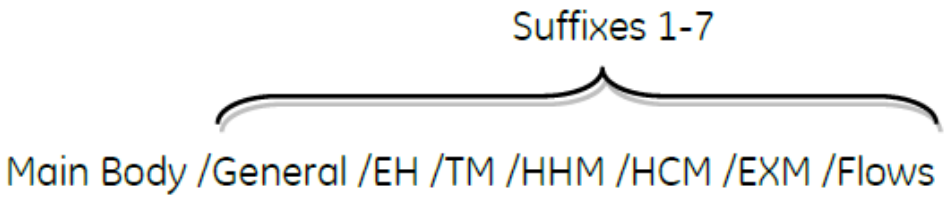


Figure 2.1 The sales generated model string's format

The main body of the string contains information about most of the holes as it dictates how many motors are needed and which valves to use.

2.2 How to interpret the string

There are two documents that where needed to understand the model string, first off is the nomenclature that tells us what all the characters in the string mean and the second one is the hydraulic configuration which is needed to interpret which valves are open and it helps us understand how configurations for some of the holes are linked. The second one also shows how the flows can change the configuration. There will be examples from the documents, but the documents themselves will not be a part of this report as Wayne does not want them made public.
It should be noted that a few of the holes are always open.

For IEC these are:

#15, 16, 21 and 27

For UL they are:

#19 and 21

Also most of the features available on IEC are not yet available on UL; here we only have the motors, valves, WIP and Automatic Temperature Compensation (ATC).

2.2.1 Main Body

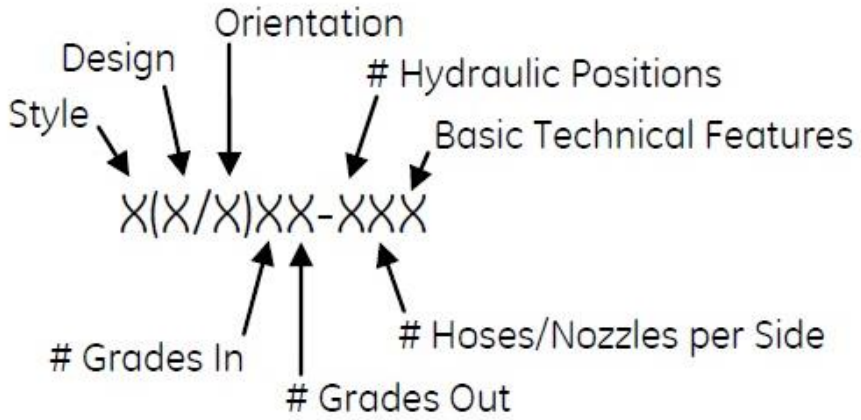


Figure 2.2 Layout of the main body of the model string

Figure 2.2 shows the main body part of the model string and the 4th to 7th character in the main body is the most essential in all of the model string for this project and is most often how the employees at Wayne refer to an order, they might talk about the "22-21" or "33-11". This tells them most of what they need to know about the fuel pumps layout. The "#Hydraulic positions" tells them how many motors and WIPs are needed, except for if there is a R (remote) in the basic technical features. This indicates that the motors are separate from the fuel pump and we have a dispenser that only require hole #4 (IEC) to be open, the amount of WIPs are still the same as the "#Hydraulic positions".

Each hydraulic has two valves except for when we have simultaneously filling (SIM), satellite (SAT), duplex or a non-mirrored setup. SAT which can be seen in figure 2.3 is triggered by a Z in the basic technical feature and adds four extra valves (two each on the first two hydraulics). On these figures the big box is the hydraulics while the smaller numbered ones that are attached to the hydraulics are the valves.

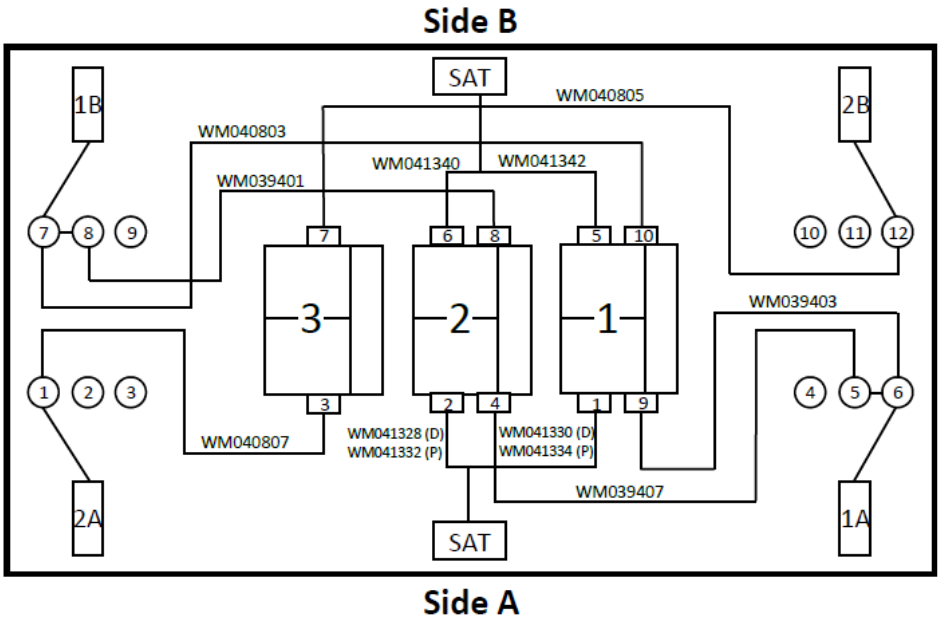


Figure 2.3 SAT hydraulic configuration

If "#Hoses/Nozzles per side" is greater than "#Grades out" or if "#Grades out" is equal to "#Hoses/Nozzles per side" plus one then we get a duplex (figure 2.4) which adds two valves to one of the hydraulics.

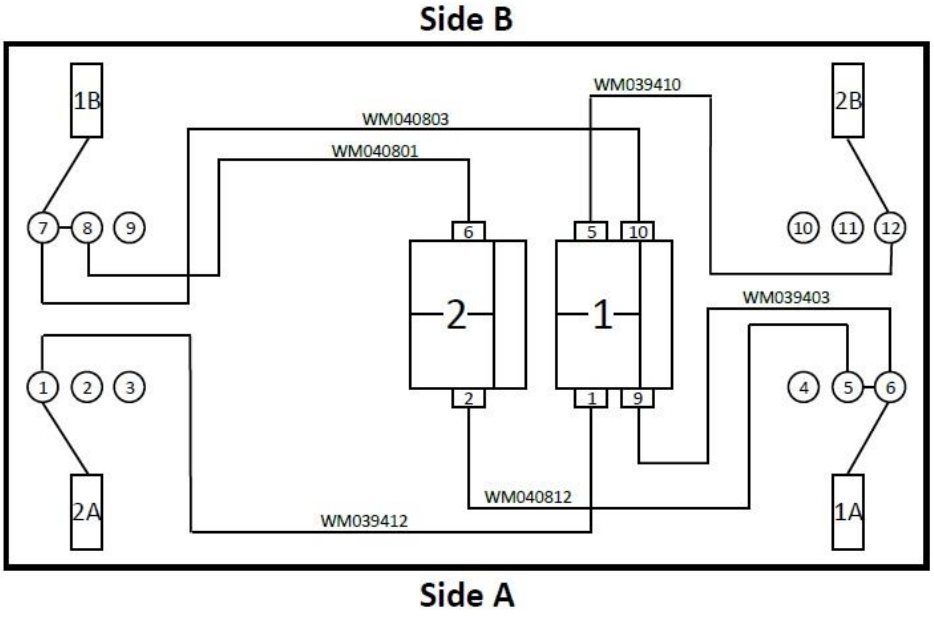


Figure 2.4 Duplex hydraulic configuration

Further if "#Hoses/Nozzles per side" is greater than "#Hydraulic positions" then we have a double duplex which adds two valves to two of the hydraulics as can be seen in figure 2.5.

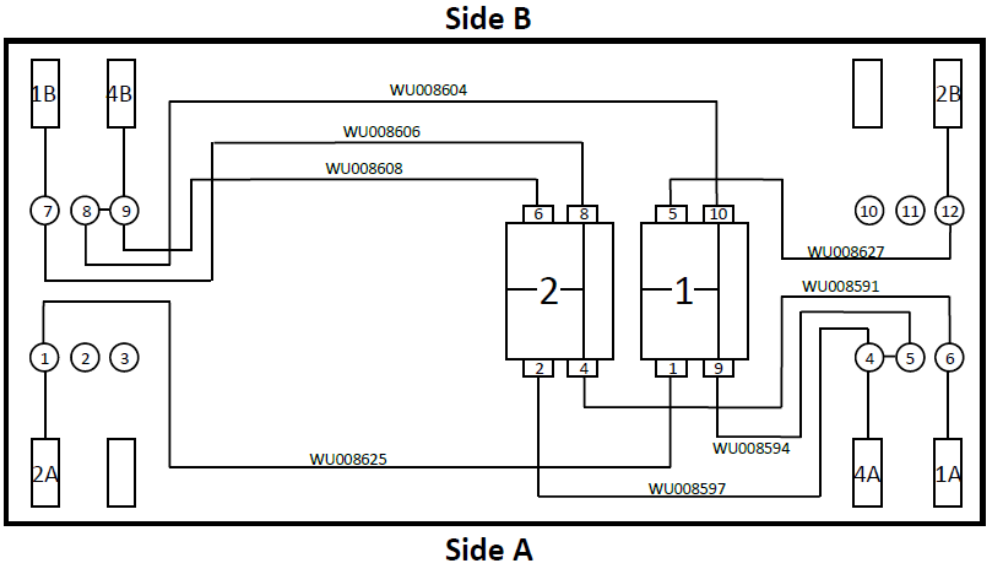


Figure 2.5 Double duplex hydraulic configuration

SIM (figure 2.6) is caused by a need for excess pressure that requires an extra hydraulic which does not have any valves, SIM can be combined with both SAT and duplex.

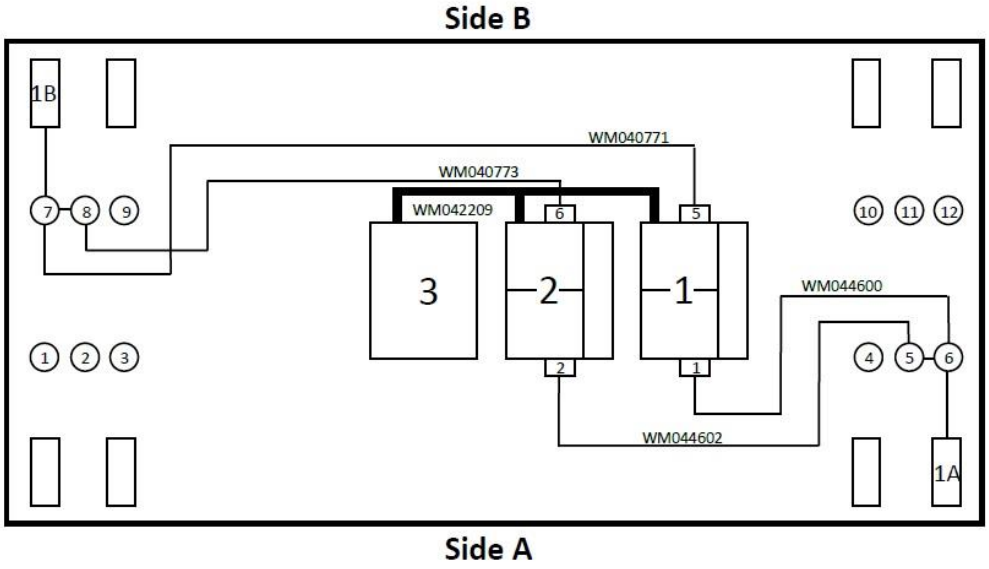


Figure 2.6 SIM hydraulic configuration

The extra valves needed for SAT and duplex uses the assigned duplex holes, except for double duplex and the second pair of SAT valves that use valve 4's hole on side A and B on the IEC barriers and on the UL we can have several conduits going through the same holes which is noted on the drawings for these.

The only other thing that can change the number of valves from being two per hydraulic is if "# Hoses/Nozzles per side" is a double-digit number which indicates that side A and B are not mirrored to each other which can be seen in figure 2.7, in these cases one side will have number of valves equal to the number hydraulics and the other side will depend on the flows in the last part of the model string.

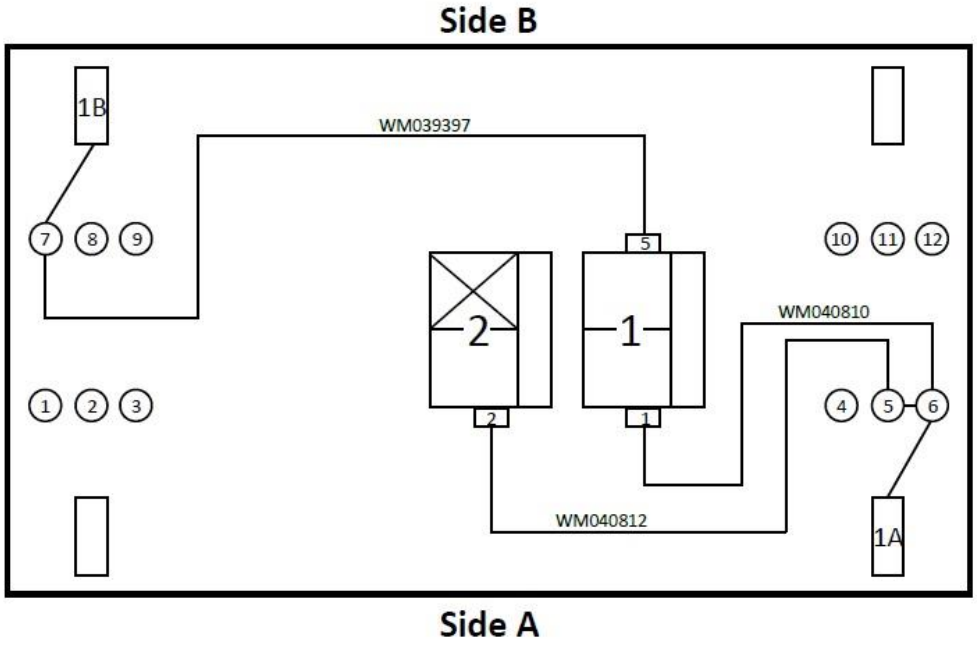


Figure 2.7 Non-mirrored hydraulic configuration

In the last part of the hydraulic configuration document we see the setup for the small style model which only utilizes at most two of the hydraulics. Here we see another special case which we get when "#Hydraulic positions" is two and we have double-digits for "#Hoses/Nozzles per side" but we still have a normal flow, this leads to one of the hydraulics not having any valves as can be seen in figure 2.8.

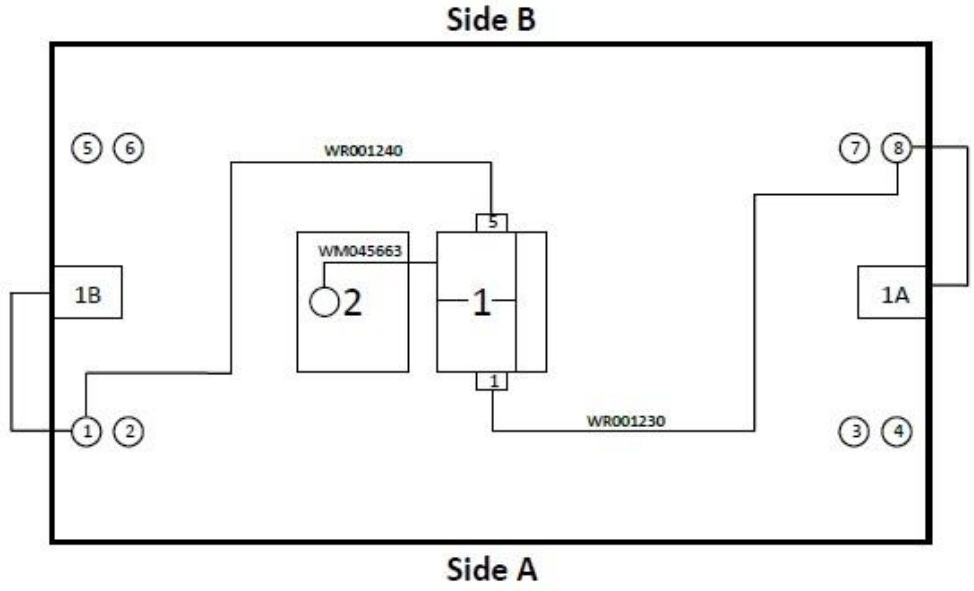


Figure 2.8 Small pump with extra hydraulic

The last part of the main body (basic technical features) governs over a few of our holes as well, as already mentioned that a R determines that only motor 4's hole is open for the motors. Also a V will indicate that holes #33 and #36 are open.

2.2.2 Electronic Head (EH)

This part of the model string handles most of the electronics in the fuel dispenser, here we have several of our holes being handled.

H3 gives #28

P gives #37

U gives #38

N gives #39

F6 gives #40

XA gives #41

2.2.3 Terminal Module (TM)

Here most of the input- and output systems are registered, we only have one hole which is decided in this section and that is #34 which is triggered by P, T or U.

2.2.4 Hydraulic Cabinet Module (HCM)

This is the last part that controls any of our holes except for flows, but flows are still linked to the main body of the model string and do not make any solo decisions.

T gives #20 (this one also applies for UL and gives #24).

P 2 or 5 gives #22

R gives #31 and/or 32 which ones depends on if the sides are mirrored or not.

R 3 or 5 gives #25 and/or 26 this also depends on if the sides are mirrored or not.

Q2 gives #23 and/or 24 also dependent of which sides are used.

W 2 or 3 gives #29

W4 gives both #29 and 30

2.2.5 Exceptions

There are several cases that are not properly translatable with the nomenclature and these needs to be handled by personnel on a case to case basis. For example one of the examples given to me by Wayne was a 33-333 where it showed the same number of flows, the only difference was that one of the flows was 40-70 instead of the normal 40. This led me to believe that it might be a duplex on just one side, the actual case was however that one of the sides had an option to change the flow between 40 and 70.

2.3 Testing the interpretations

When this part of the project was done Wayne sent me several actual sales orders to test the interpretation on. These consist of a number of details concerning the whole dispenser and at the bottom the model string is written. The interpretation was now used to translate the string to find out which holes where open for each order. This was mostly done to find flaws in the interpretation process such as the case with 33-333 that was mentioned in the previous paragraph. More extensive testing will have to be made further into the process when the code is implemented into Wayne's systems.

3 Concept generation

Here I will go through the concept generation and the ideas that lie behind the concepts.

3.1 Boundaries

When generating concepts I first wanted to find out what would limit my options, what would set up boundaries and what would be the extreme cases. Also I wanted to see in how many dimensions would I be able to work, for example would it be a line with one limiting variable or a plane with two etc.

I came to the conclusion that the only scalable limitation was how one could define a marker for the holes. The outer cases for this would be a micro or macro definition, what this means for the outer cases is that on a micro level each individual hole is assigned an indicator and on a macro level the whole configuration of holes is given the indicator. Anything in between would be a compromise or an alteration of the micro and macro which would consist of larger- or smaller groups of holes.

3.2 Concepts

I will here list the concepts that where created and their pros and cons. For all the examples I use the same configuration given by one of the example model strings that was provided to me by Wayne.

Model string: H(W/LU)33-
330BGPSV/CZVE/A2C3D2F5G6H2KMXJXL2/B4C7D8E3P2T2U2/A3B2C2E2F2
KL4N2/A2B2C2F2HM3P2Q2/F2/40',40,40-70:B,B,B

This is an IEC barrier (figure 3.1) and the open holes are:
1,2,3,5,6,7,15,16,17,18,21,22,23,27,33,34,36

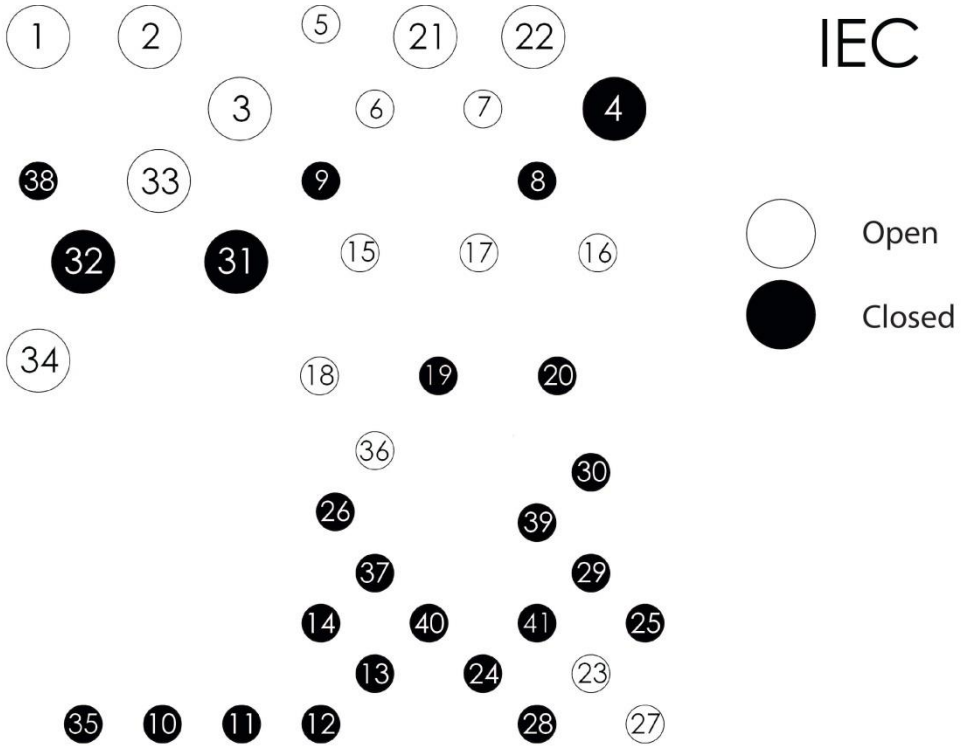


Figure 3.1 Visual representation of our example. White holes are open and black holes are closed.

3.2.1 The list

We start with the macro extreme, in this concept each configuration is given a number and placed in a list. This concept would be very good if there weren't "dead" configurations, when going through the nomenclature it became clear that not all possible configurations are in use instead we have ca. 20000 active configurations (compared to about $4*10^{12}$ possible configurations). To be able to use this concept someone would therefore have to input each active configuration into the list. Also the only way to find missing list objects would have to be the specific configuration being sought after.

On the plus side this is the shortest code in our concept consisting of 5 numbers and even if we added an initial indicator for IEC or UL this would still hold true with a decent margin. This concept is also very flexible and new configurations can easily be added to the list.

3.2.2 Binary foundation code

On the other side of our boundary line we have the micro configuration, we give each hole a binary indicator (figure 3.2) that states if the individual hole is open or closed. We also need to start the code of with an indicator that shows if the barrier is UL or IEC. However if we leave it as a binary code it would become very long (42 characters) and we therefore want to transform it into decimal form. The code should be equally long for both UL and IEC, so for UL we would have to add closed "holes" as fillers.

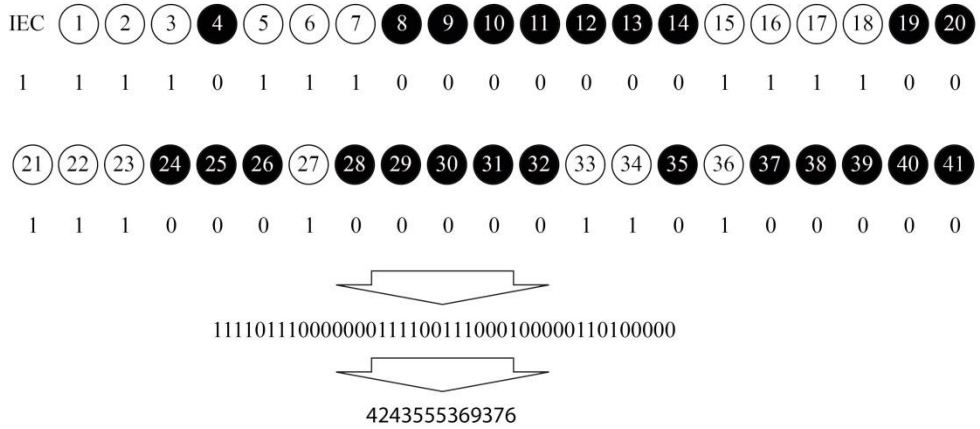


Figure 3.2 Binary transformation of the example

This is the code that is the easiest to implement and it is also very flexible, since it easily changes on an individual level. However this is also the longest code with 13 characters and the likelihood of a user error is very high.

3.2.3 Refined binary foundation code

The usage of the binary foundation has a logical feeling and in this concept we make a few alterations to the creation of the code to allow for a more manageable code. We start the same way as the previous concept, we give a binary indicator for each hole and that states if it's an IEC or UL barrier. Now we divide the long string into groups of 5 (figure 3.3), filling out the last ones with closed "holes" to make the groups even. We then transform the groups into decimal form and use a 32-bit encryption system (figure 3.4) to receive a 9 character code consisting of numerals and letters.

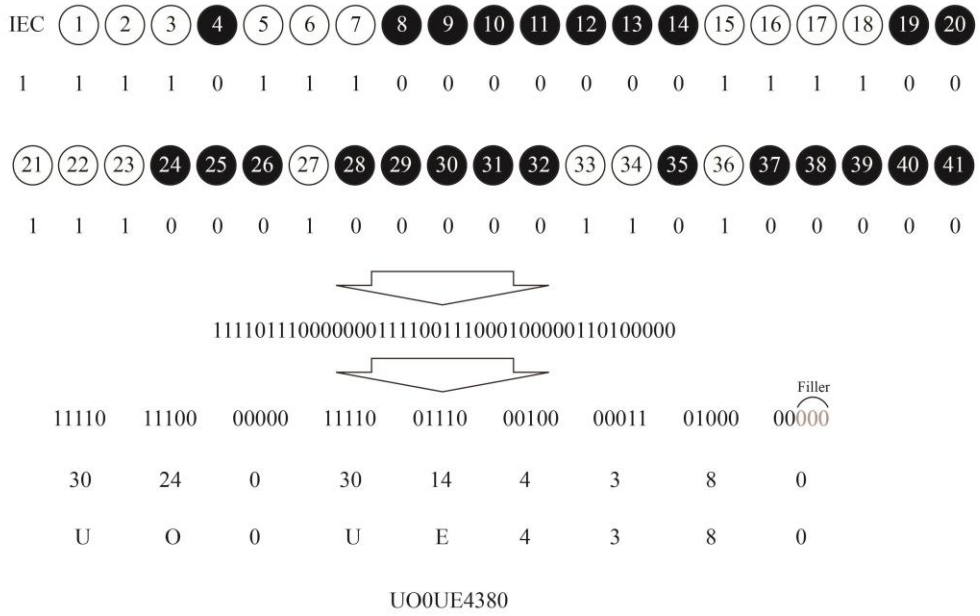


Figure 3.3 Binary transformation of the example for the refined code

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V

Figure 3.4 The 32-bit encryption system

This code still contains the advantages of the previous binary foundation code, in this case however we have a shorter and more manageable code.

3.2.4 Grouping based code

This code is based on the fact that most of the holes are linked to each other due to the nomenclature. We can therefore form groups (table 3.1) that will be consisting of three or more holes with a maximum of ten alternatives. This allows us to not have to take into account a lot of the configurations which are not active. Those holes that are not linked to any other holes can still be placed into groups of 3 that will have eight different alternatives.

We need to start off with a group that indicates whether it is an IEC or UL barrier and since this is only three alternatives (here we need to take into account if it's UL wide or UL narrow) we can add in some of the holes that did not fit into our groups.

Table 3.1 Suggestion for how the groups could be arranged for the IEC barrier

Group	Alternatives	Configurations
IEC, UL Wide/Narrow, 41	4	(IEC) (IEC,41) (UL Wide) (UL Narrow)
1-4, 17-19	8	(1) (1,2,17) (1-3,17,18) (1-4,17-19) (4) (4,17) (4,17,18) (4,17-19)
5-9	10	(-) (5) (5,6) (5-7) (5-8) (5-9) (5,9) (5,6,9) (5,6,8,9) (5-7,9)
10-14	10	(-) (10) (10,11) (10-12) (10-13) (10-14) (10,14) (10,11,14) (10,11,13,14) (10-12,14)
23-26,31-33	10	(-) (23,33) (24,33) (23,24,33) (23,31,33) (24,32,33) (23,24,31,32,33) (23,25,31,33) (24,26,32,33) (23,24,25,26,31,32,33)
20,22,28	8	(-) (20) (22) (28) (20,22) (20,28) (22,28) (20,22,28)
29,30,34	8	(-) (29) (30) (34) (29,30) (29,34) (30,34) (29,30,34)
35-37	8	(-) (35) (36) (37) (35,36) (35,37) (36,37) (35-37)
38-40	8	(-) (38) (39) (40) (38,39) (38,40) (39,40) (38-40)

We use the grouping in table 3.1 to display our example in table 3.2.

Table 3.2 Code example using group based code

Group	Configuration	Code indicator
1	IEC	0
2	1,2,3,17,18	2
3	5,6,7	3
4	N/A	0
5	23,33	1
6	22	2
7	34	3
8	N/A	0
9	N/A	0

This means the code in this example is: 023012300

This code is 9 characters long, but it only consists of numbers and it is possible to gain knowledge of the configuration by looking at the code. However this code requires a bit of work to setup and it is very susceptible to changes in the barrier, if any changes where to be made most of the code would probably have to be reworked.

3.3 Reflection on the process

We ended up with four concepts which might feel like quiet few, however when working on our boundaries it became clear that all the possible solutions between the extremes are mere alterations of the same thing and all suffer from the same flaw which is the lack of flexibility. I do however feel that the group solution that was presented utilized the grouping in a way that is very close to being optimized.

4 Choosing concept

In this chapter we will go through the decision process that was made to decide which concept we would follow up with.

4.1 Setting up selection matrix

We look back to our specifications and use these to setup our matrix (table 4.1) that will be used to score our concept against each other. We also give them each a weight that determines how important each criterion is in our scoring process.

Table 4.1 Selection matrix

Criteria	Explanation	Weight
Complexity	How long/short is the code and does it use letters as well as numerals?	30%
Flexibility	How difficult is it to make changes to the code if the barrier changes?	20%
Overview	How easy/hard is to find error in the code or in anything related to it?	15%
Work load	Amount of prep-work needed to implement the code.	20%
User-friendliness	How easy is the code to work with? How probable is someone to make an error when using it?	15%
Backwards compatibility	Is the code interpretable in both ways	

Except for the last one each criteria is graded on a scale from 1-10. The backwards compatibility is a simple yes or no and any concept that would get a no would automatically be removed as a usable concept.

4.2 Scoring the concepts

When all the concepts where created a presentation was held at Wayne and the employees that attended the presentation where used to get a basis for scoring our concept against each other. This information for our selection matrix can be seen in table 4.2.

Table 4.2 Scoring of our concepts, weighted scores in the brackets

Criteria	Weight	List	Group	Binary (refined)	Binary
Complexity	30%	10 (3)	7 (2.1)	6 (1.8)	4 (1.2)
Flexibility	20%	10 (2)	5 (1)	9 (1.8)	10 (2)
Overview	15%	3 (0.45)	8 (1.2)	8 (1.2)	8 (1.2)
Work load	20%	2 (0.4)	7 (1.4)	9 (1.8)	9 (1.8)
User-friendliness	15%	9 (1.35)	8 (1.2)	8 (1.2)	5 (0.75)
Backwards compatibility		Yes	Yes	Yes	Yes
Score		7.2	6.9	7.8	6.95

As we can see the concept that scored the highest is the refined binary foundation code and this is also the one that we decided to go forward with. As can be seen the group concept suffered heavily from its lack of flexibility, the lists low scores on work load and overview severely weighed down it's excellent scores on the other three criteria and the original binary foundation concept suffered heavily on complexity and user-friendliness. What made the refined binary concept shine was it overall high scores, it was fairly low on complexity but scored well on all the other criteria.

4.3 Reflecting on the process

The concept that we were feeling most comfortable with was the one that scored highest and is the one that was chosen. The issue that was most noticeable during this process was that even though the involved employees at Wayne gave input on the flaws they saw in the concepts the scoring and choice was up to me. This was due to fact that I was the only one who had knowledge in how exactly the concepts would function, for them it was more about whether they could actually make it function in their systems. We do however fell that the right concept was chosen in the end and when presenting it to people working on related projects and departments they have also shown enthusiasm towards it.

In the decision process additional scoring steps could have been used to give a more accurate picture over which is the best concept. However since there where only four concepts there was no use to start off with cutting some of them or even combine several ones to create a better solution.

5 Moving forward with the concept

In this chapter I will talk about what steps need to be taken to implement the chosen concept into Wayne's systems and testing its functionality.

5.1 Current state

We now have a concept that has been approved by the people at Wayne who are in some way involved in the larger process of improving the production of the Helix dispensers. However there are still things that need to be done before the concept can be implemented into the system. There are also some choices that need to be done by Wayne to finalize the project.

First of Wayne needs to decide on one of their systems that will be used to translate the barrier- or dispenser configuration into our code, then they need to make this system output the code so it can be tested and see what changes need to be made in the interpretation of the nomenclature.

5.2 Testing of the concept

There are several tests that Wayne needs to do before they are fully able to rely on the code. First they need to implement a system for generating the code and test that the output is correct. This will require first testing how the configuration interpreter handles standard model strings and then move forward to analyze to what extent it can handle special cases.

Tests will also have to be made to see how employees interact with the code and possibly make changes from their feedback when handling it.

Lastly test should be made once the knock-out machine is finished to see how it handles interpreting the code and to make sure that the correct hole configuration is being outputted by the machine.

5.3 What needs to be added

There are still several things that need to be done before the entirety of this project is done, these things are however outside the scope of my part of this project but I will however give some input that can be used as a foundation for these decisions.

There is a request from several employees at Wayne that the code should be able to give a visual representation of the hole configuration. They want to input the code

into a system and get a picture in response that shows the barrier and which of the holes are open.

For easier usage of the code it is desirable to implement some sort of reading device, this will not only help reduce the human factor, it will also make the interaction with the knock-out machine easier.

To increase the security when using the code a check digit should be implemented at the end of the code, most codes that use a reading device also have check digit even though the human factor is heavily reduced.

As have been mentioned in chapter 2 the employees at Wayne often refer to different configurations with the hydraulic part of the model string and it has been suggested that this could in some way be connected to the code. My suggestion here would be that this be added to for example a barcode. This would give an employee a quick idea of how a big part of the barrier should look like.

6 Check digit

In this chapter I will go over the function of a check digit and go over a few possible alternatives that could be used for our code.

6.1 Function

A check digit is an algorithm that takes your code as an input and ads an extra character at the end. If the code is changed the check digit will be incorrect and the algorithm is designed in such a way that the check digit will detect most of the more common mistakes such as transpositions (switching 09 to 90), single digit (1 => 2) and several others.

Most of these errors are caused by human error and even though a reading device would remove most human interaction a check digit is still recommended. If you look at most common uses of bar codes such as in supermarkets they still implement a check digit in case of a technological malfunction.

6.2 Suggested algorithms

When going over check digits there was two algorithms that appeared to work with our code. Most of the algorithms can't handle both numerals and letters, also our 32-bit encryption causes them some issues.

6.2.1 Luhn mod N algorithm

The Luhn algorithm [4] was first introduced in 1954 and is still extensively used today in for example credit card numbers. The algorithm (table 6.1) takes every second character (starting with the second from the left) and doubles it, if the digit is higher than 31 (since we use 32 as our base) you sum the digits of the products (45 would give 4+5=9). Now add all the numbers in the code and use modulus (32), the check digit becomes 32 minus our answer. To check the validity of the check digit we redo the procedure and the modulus (32) should equal zero for the check digit to be correct, however since our code has an uneven number of characters we need to double every second except for the check digit.

Table 6.1 Computing a check digit with the Luhn algorithm

Code	U	O	0	U	E	4	3	8	0
Encryption	30	24	0	30	14	4	3	8	0
Double every second character	30	48 4+8=12	0	60 6+0=6	14	8	3	16	0
Sum	89								
Modulus(32)	25								
32-25	7								
Check digit	7								

This code is fairly simple to implement and there are written up codes for most programing languages available online. The flaws lie in that it has a slightly lower security level then our alternative, it cannot detect many of the adjacent transposition errors such as 09 to 90. However the large amount of valid input characters reduces the impact of this weakness.

6.2.2 Damm algorithm

The Damm algorithm [5] was introduced in 2004 and uses a matrix to determine the check digit. The matrix is an anti-symmetric quasigroup (figure 6.1), this means that each character only exists once in every row and column.

	0	1	2	3	4	5	6	7	8	9
0	0	3	1	7	5	9	8	6	4	2
1	7	0	9	2	1	5	4	8	6	3
2	4	2	0	6	8	7	1	3	5	9
3	1	7	5	0	9	8	3	4	2	6
4	6	1	2	3	0	4	5	9	7	8
5	3	6	7	4	2	0	9	5	8	1
6	5	8	6	9	7	2	0	1	3	4
7	8	9	4	5	3	6	2	0	1	7
8	9	4	3	8	6	1	7	2	0	5
9	2	5	8	1	4	3	6	7	9	0

Figure 6.1 Anti-symmetric quasigroup

To use this algorithm (figure 6.2) we setup an interim digit and set it to 0, we then go through the code character by character. We use the current character to determine our column and the interim digit to find the row. The number in this position is set to our new interim digit and we continue to the next character in the code. When we have gone through the entire code the final interim digit becomes our check digit. To check that it is correct we go through the same process again and include the check digit at the end, if the final interim is zero our check digit is correct.

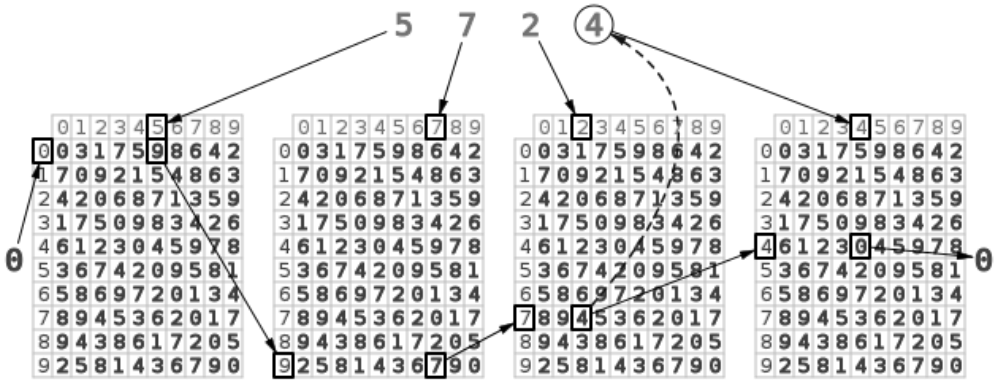


Figure 6.2 This is an example using a base of 10 where the code is 572 where the check digit becomes 4.

This algorithm is safer than the Luhn algorithm since it detects all transposition errors. It is however a bit more work to setup since one first need to determine an anti-symmetric quasigroup in an order of 32, the method for this can be found in Damm's doctoral dissertation. The main downside however is that the algorithm is fairly unknown and has hardly any users.

6.3 Reflections and suggestions

Both of these algorithms are viable for our code and I do believe that a check digit is a necessity. My suggestion would be to go with the Luhn algorithm since there are plenty of available pre-written codes online. Also while the Damm algorithm is more secure this should get negated by the implementation of a reading device and the Damm algorithm might also be trickier to make into code. Finally since the Luhn algorithm is heavily used and the Damm algorithm is very scarcely used it might be easier to find support for the Luhn algorithm if a problem arises.

7 Reading device

I'll in this chapter go over a few different options for reading devices and make arguments for which one is the best option for this project.

7.1 Barcode

The barcode was first introduced in 1948 and is still heavily used today due to its cheap nature and its massive variety. You most likely encounter barcodes every day on consumer items you buy or use and today it is easy and cheap to get your own readers and codes.

There are plenty of different variations of barcodes and for our code we could use either a linear- or matrix barcode. Some of the options include:

- Code 93 [6] (figure 7.1), a linear barcode that can write all letters and numerals.

Figure 7.1 An example of code 93

- Aztec Code [7], matrix barcode, it is public domain.
- Code 128 [6], linear barcode also utilizes alphanumeric codes.
- Data Matrix [7], matrix barcode, public domain.
- QR-code [7] (figure 7.2), one of the more well-known matrix barcodes, not licensed

Figure 7.2 QR-code

The choice of which system to use would be up to Wayne and can depends on what other information they may want to display with the barcode. Extra information could also be added on the barcode sticker for user to get a quick idea of the general configuration of the barrier in question.

7.2 Radio-frequency identification (RFID)

The first RFID [8] was invented in 1945 and it's a non-contact reading device. The RFID consist of a tag and a reader (figure 7.3), the tag is attached to the object you want to identify and when it comes in range of the reader information is transmitted to the reader. You probably run into RFIDs fairly often in for example door tags, security cards or public transportation payment.

Figure 7.3 RFID tag and reader

There are three different systems and these are classified by the tag and the reader relationship. These are; passive reader active tag, active reader passive tag and active reader active tag. This depends on which part of the RFID sends out signals to find the other part of the RFID, this also determines which part of the RFID that requires a power source. For Wayne's purposes an active reader passive tag would be the better option since a tag can basically be a sticker that would be put onto the barrier.

There are well prized options when it comes to RFID and it can allow for a lot of information being connected to the barrier.

7.3 Optical reader

An optical reader is a device that can scan text and interpret it, devices that read 2-dimensional barcodes are also optical readers and it is also very common in scanners. The device is basically a camera and a program that recognizes text or patterns. The price for these are similar to the other features, the question is how it would handle factory environments.

7.4 Reflections

Any of these suggestions would be viable for Wayne's needs; it would be up to them to go through a process of doing in depth comparisons on the alternatives that they could utilize. The strongest suggestion would probably be a simple linear barcode,

since after the implementation of these the printing of all barcodes could easily be made in-house.

8 Reflections of the project

In this chapter I will reflect upon what was done well and what could have been done differently with the project. I will also analyze the different steps of the process.

8.1 The process

This process has followed a good thread, it have lead me to learning more and more of the system until I have been able to feel comfortable with my understanding of most things related to the project. If possible I would have wanted to generate some additional concepts, however after the initial ground work it seems like there is no additional concept that would add anything new.

Ideally the project would have gotten so far as to be implemented into their systems and tested, the initial delay in the project and the decisions having to be made at a higher level made it so that this did not happen. It does however seem like Wayne are positive to the results and we will see the chosen concept being used soon.

8.2 Time consumption

As can be seen in Appendix A the expected time table did not hold up, this was because almost nothing happened in the first two months due to some personal issues on my side. There was some communication going on with Wayne at the time so that the project at least got started even though the general speed of the project has been slow. However the project was finished before summer vacations, not causing it to get further delayed.

8.3 Aftermath

This project has given me insight into the workings of a company's dealings with a project and how bureaucracy interacts with the dealings of everyday tasks. I'll also take with me the experience of working with a project that a company is looking to implement.

This project has also taught me that there are always events that will spoil your time plan no matter how flexible it may seem.

References

[1] Information about Wayne (Electronic). www.wayne.com, 2014-06-01

[2] Information about the Helix dispensers. www.wayneHelix.com, 2014-06-01

[3] Ulrich, K och Eppinger, S (2008). Product Design and Development. McGraw Hill, USA.

[4] Patent for Luhn computer for verifying numbers (Electronic). http://www.google.com/patents/US2950048, 2014-06-01

[5] Damm, M.H. (2007). Totally anti-symmetric quasigroups for all orders $n \neq 2,6$ (Electronic), Doctoral thesis, Discrete mathematics, Phillips-Universität, Marburg, Germany, www.sciencedirect.com, 2014-06-01

[6] Information about code 93 and 128, (Electronic). http://www.barcodeisland.com/symbolgy.phtml, 2014-06-01

[7] Information about 2-dimensional barcodes (Electronic). http://www.adams1.com/stack.html, 2014-06-01

[8] Sanghera, P (2007). RFID+ Study Guide and Practice Exams, Syngress Media, USA

Appendix A: Timetable

Expected timetable

11th November – 21st December: Collecting information that will lay the foundation for the concepts and their evaluation.

21stDecember – 20thFebruary: Creating concepts and evaluating them.

~5thFebruary: Halftime presentation with handler.

20thFebruary– 14thApril: Testing and implementing concept into the systems

14thMarch – 14thApril: Compiling of the work and creating the presentation.

14th-29thApril: Final adjustments with handler.

~29thApril: Presentation of Master thesis.

Actual timetable

11thNovember – 1stMarch: Collecting and interpreting information that lays foundation for the concepts.

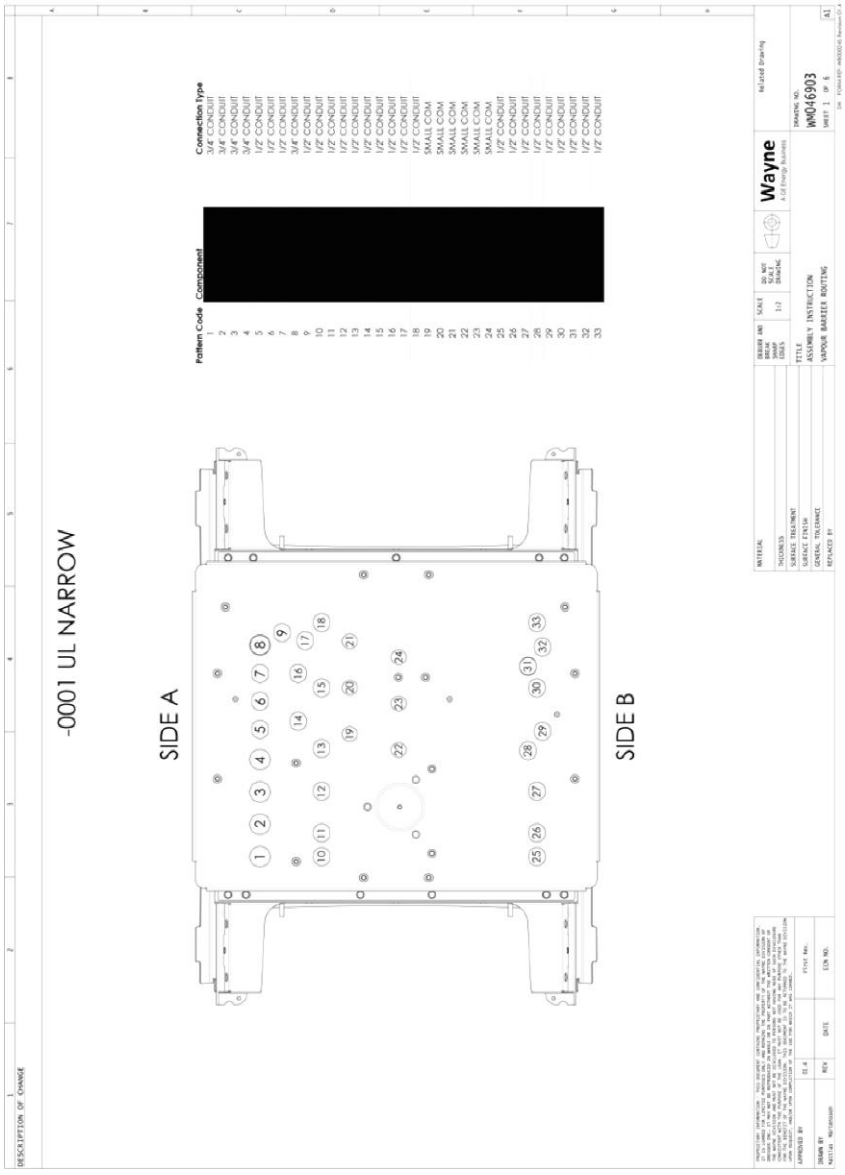
1stMarch – 11thApril: Creation and evaluation of concepts.

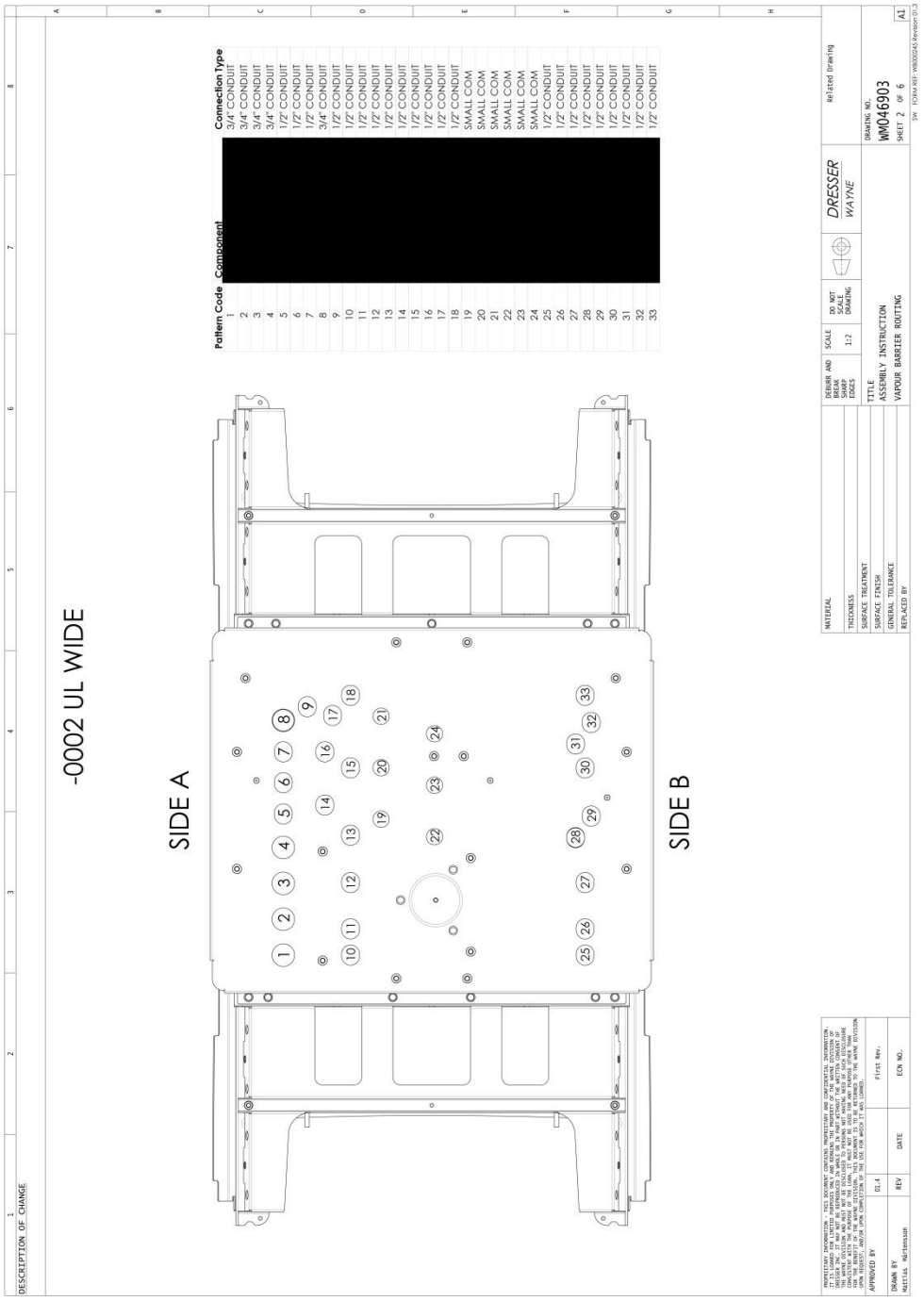
11thApril – 30th May: Gathering and implementation of information concerning systems that will should be implemented with the chosen concept.

8thMay-4thJune: Compiling the work into the report

4thJune-16thJune : Final adjustments with handlers.

Appendix B: Barrier blueprint

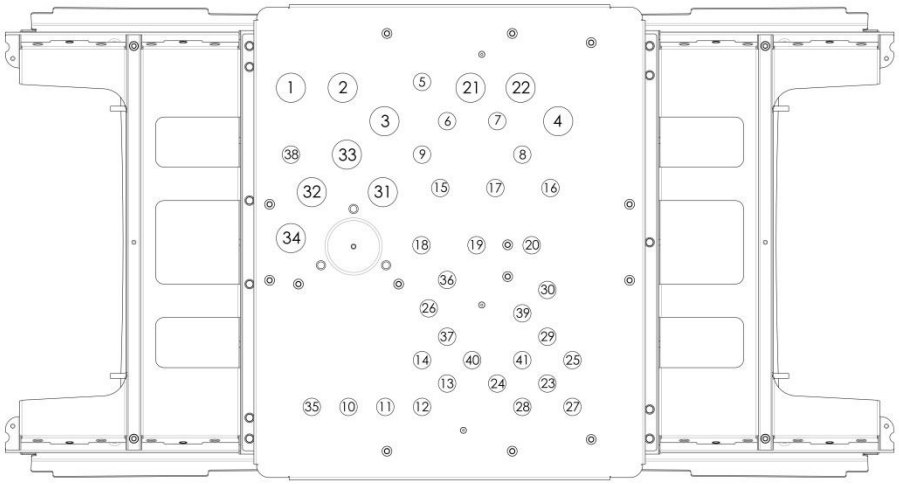



-0002 UL WIDE
SIDE A
SIDE B
Pattern Code Component Connection Type
1 3/4" CONDUIT
2 3/4" CONDUIT
3 3/4" CONDUIT
4 1/2" CONDUIT
5 1/2" CONDUIT
6 1/2" CONDUIT
7 3/4" CONDUIT
8 1/2" CONDUIT
9 1/2" CONDUIT
10 1/2" CONDUIT
11 1/2" CONDUIT
12 1/2" CONDUIT
13 1/2" CONDUIT
14 1/2" CONDUIT
15 1/2" CONDUIT
16 1/2" CONDUIT
17 1/2" CONDUIT
18 SMALL COM
19 SMALL COM
20 SMALL COM
21 SMALL COM
22 SMALL COM
23 1/2" CONDUIT
24 1/2" CONDUIT
25 1/2" CONDUIT
26 1/2" CONDUIT
27 1/2" CONDUIT
28 1/2" CONDUIT
29 1/2" CONDUIT
30 1/2" CONDUIT
31 1/2" CONDUIT
32 1/2" CONDUIT
33 1/2" CONDUIT
DESCRIPTION OF CHANGE
MATERIAL
THICKNESS
SURFACE TREATMENT
SURFACE FINISH
GENERAL TOLERANCE
REPLACED BY
DEBUR AND BREAK SHARP EDGES
SCALE 1:2
DO NOT SCALE DRAWING
DRESSER WAYNE
TITLE ASSEMBLY INSTRUCTION VAPOUR BARRIER ROUTING
DRAWING NO. WM046903
SHEET 2 OF 6
Related Drawing
APPROVED BY
DRAWN BY Mattias Mårtensson
DATE
ECN NO.
First Rev.
A1

DESCRIPTION OF CHANGE

-0003 IEC

SIDE A

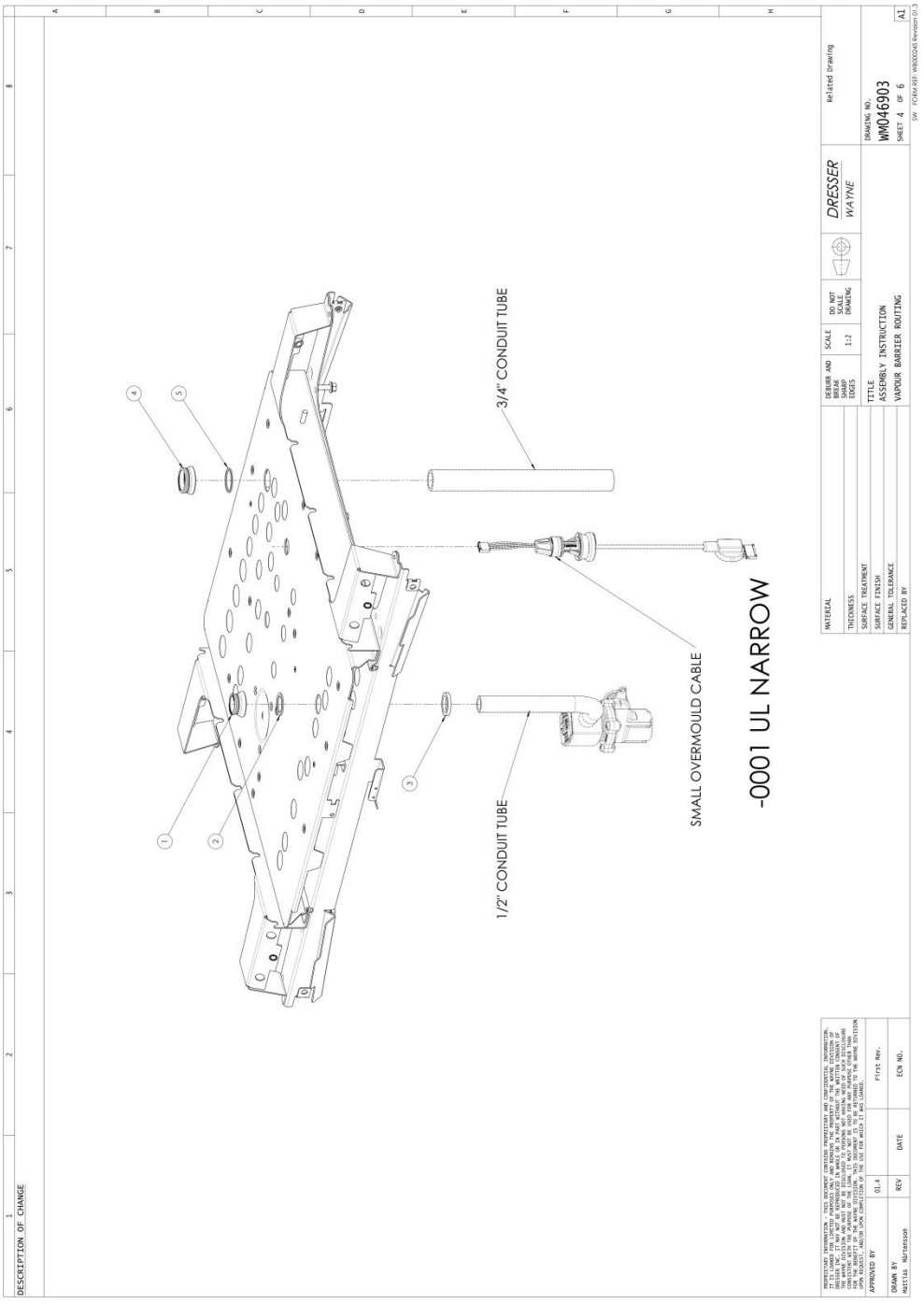


SIDE B

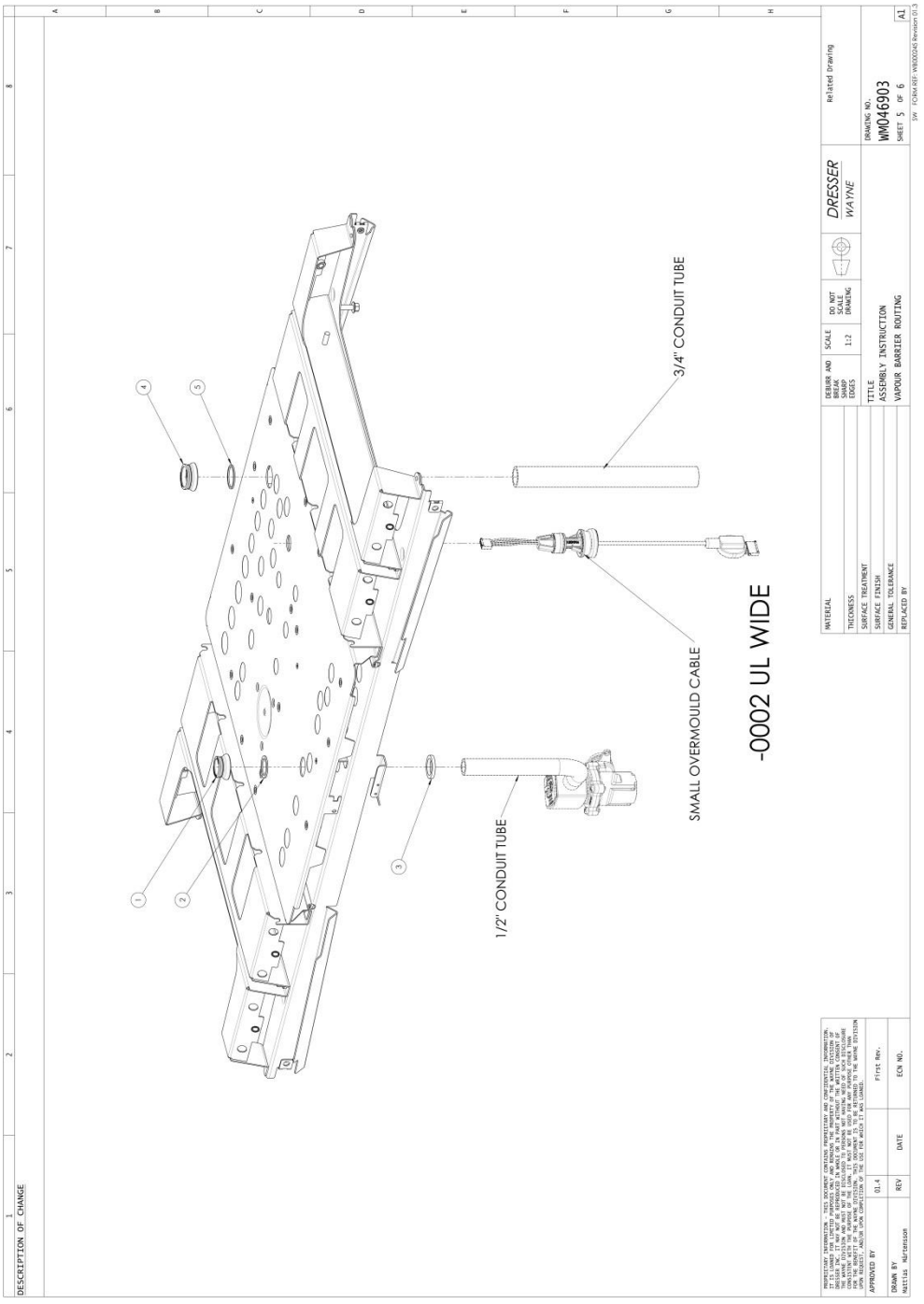
2	[redacted]	LARGE COM
3		LARGE COM
4		LARGE COM
5		SMALL COM
6		SMALL COM
7		SMALL COM
8		SMALL COM
9		SMALL COM
10		SMALL COM
11		SMALL COM
12		SMALL COM
13		SMALL COM
14		SMALL COM
15		SMALL COM
16		SMALL COM
17		SMALL COM
18		SMALL COM
19		SMALL COM
20		SMALL COM
21		LARGE COM
22		LARGE COM
23		SMALL COM
24		SMALL COM
25		SMALL COM
26		SMALL COM
27		SMALL COM
28		SMALL COM
29		SMALL COM
30		SMALL COM
31		LARGE COM
32		LARGE COM
33		LARGE COM
34		LARGE COM
35		SMALL COM
36		SMALL COM
37		SMALL COM
38		SMALL COM
39		SMALL COM
40		SMALL COM
41		SMALL COM
41		SMALL COM

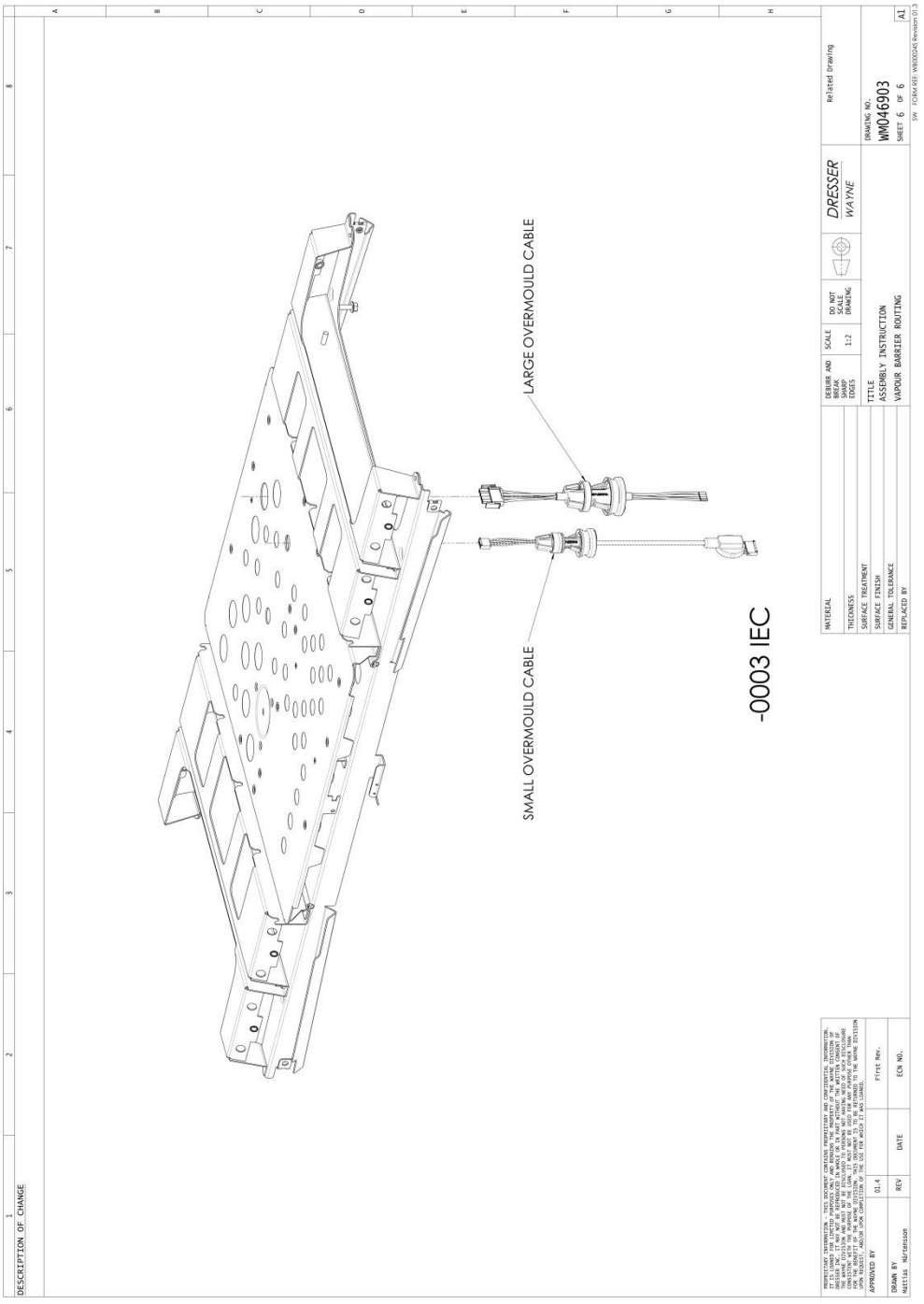
APPROVED BY	01.4		First Rev.
DRAWN BY Mattias Mårtensson	REV	DATE	ECN NO.

MATERIAL	DEBURR AND BREAK SHARP EDGES	SCALE 1:2	DO NOT SCALE DRAWING	DRESSER WAYNE	Related Drawing
THICKNESS					
SURFACE TREATMENT					
SURFACE FINISH	TITLE ASSEMBLY INSTRUCTION				DRAWING NO. WM046903
GENERAL TOLERANCE	VAPOUR BARRIER ROUTING				SHEET 3 OF 6 A1
REPLACED BY					


3/4" CONDUIT TUBE
SMALL OVERMOULD CABLE
1/2" CONDUIT TUBE
-0001 UL NARROW
DRESSER WAYNE
DRAWING NO.
WM046903
SHEET 4 OF 6
TITLE
ASSEMBLY INSTRUCTION
VAPOUR BARRIER ROUTING
SCALE
1:2
DO NOT SCALE DRAWING
Related Drawing
DESCRIPTION OF CHANGE
APPROVED BY
DRAWN BY
Matthäs Mårtensson
DATE
ECN NO.
First Rev.

Appendix B: Barrier blueprint




DESCRIPTION OF CHANGE
LARGE OVERMOULD CABLE
SMALL OVERMOULD CABLE
-0003 IEC
DRESSER WAYNE
DRAWING NO.
WM046903
SHEET 6 OF 6
VAPOUR BARRIER ROUTING
ASSEMBLY INSTRUCTION
Related Drawing