Welding machine 

**Final Assignment introduction software engineering**



Student: Teacher:

Datum: 13-01-2018

**Versionlog**

|  |  |  |
| --- | --- | --- |
| 29-11-2017 | V0.1 | Concept |
| 7-12-2017 | V0.2 | Definition fase |
| 29-12-2017 | V0.3 | Finalizing report |
| 13-1-2017 | V1.0 | Changes after review teacher |

**Summary**

Automated welding machines are built at AWL-Techniek in Harderwijk. The products are welded by means of welding robots. The products that are welded are, for example, car seats or fences for industrial purposes. The machines are normally controlled by a PLC. During this project, a welding machine was programmed in C language.

The project "welding machine" is programmed in the language C in QT Creator. It's a control software. The project is only a small part of a welding machine because otherwise it would become too complex. The part where the software is programmed for are a welding robot, handling robot with the associated safety devices. This part of the machine welds a fence, which is placed in parts in the machine, together by means of the welding robot. The fencing is taken from the machine by means of the handling robot.

The documentation is made by means of the program Doxygen, this documentation is automatically created by means of the C code.

The machine must be easy to operate for the operator by means of a menu from which it is possible to choose what the machine will do. The code is clearly programmed using header files and comments.

**Preface**

In front of you lies the report "welding machine" in the context of the software engineering module. Before this module started I had almost no experience with C-Programming. This was causing some problems. But with the help of the learning team and feedback from the teacher, I managed to achieve a good end result. Before starting the project, I made a plan to complete the assignment in a structured way. Thanks to this planning, I succeeded in completing the assignment on time.

I would like to thank the teacher Jos Onokiewicz for taking care of this module and for the help with creating the C code and the report for the final assignment. I also want to thank my learning team for the help.

**Inhoudsopgave**

[1. Introduction 7](#_Toc58334720)

[2. Definition fase 8](#_Toc58334721)

[2.1 Background project 8](#_Toc58334722)

[2.2 Functional specification 9](#_Toc58334723)

[2.3 Technical specification 9](#_Toc58334724)

[3. Design fase 11](#_Toc58334725)

[3.1 Layered Architecture 11](#_Toc58334726)

[3.1.1 User Interface Layer 11](#_Toc58334727)

[3.1.2 System Control Layer 11](#_Toc58334728)

[3.1.3 Subsystems Layer 11](#_Toc58334729)

[3.1.4 Hardware Abstraction Layer 11](#_Toc58334730)

[3.2 State machine 12](#_Toc58334731)

[4. Realization and testing 13](#_Toc58334732)

[4.1 Development environment 14](#_Toc58334733)

[4.2 Userinterface 15](#_Toc58334734)

[4.3 Acceptance test 15](#_Toc58334735)

[4.4 Documentation 16](#_Toc58334736)

[5. End result and recommendations 16](#_Toc58334737)

[Attachments: 17](#_Toc58334738)

[C - Source Code 17](#_Toc58334739)

[User Manual 17](#_Toc58334740)

# Introduction

The project "Welding machine" is a project that a welding machine at AWL-Techniek is programmed in the language C. At AWL-Techniek many welding machines are built for different industries. These machines are sold all over the world.

This report explains how building the software in C for a statemachine was carried out at AWL-Techniek. The state machine that has been realized is a welding machine. The welding machine consists of various parts including 2 robots, a control panel with a start and stop button, an emergency stop and a door to access the robots.

The structure of the report is as follows:

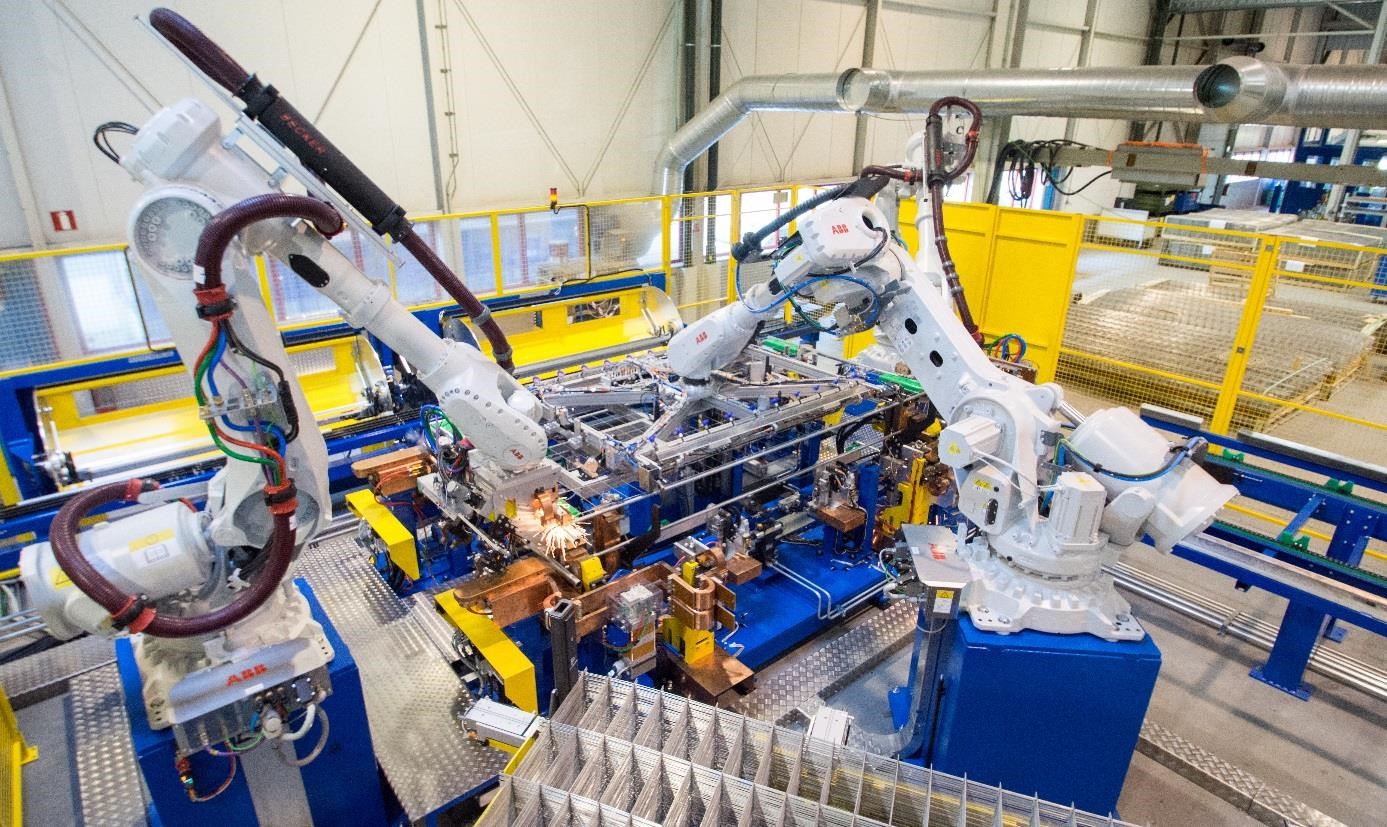
* Chapter 2 describes the definition phase. The definition phase contains the functional and technical specifications that show how the machine works and which products have been used.
* Chapter 3 describes the design phase. Here you can read how a technical solution for the welding machine was created and diagrams are showing how the machine is constructed.
* Chapter 4 contains information about the realization phase and testing. The realization phase shows how the program is structured and shows what the user environment is like. It is briefly described how the tests were carried out and where the tests can be found.
* Chapter 5 contains the final result and recommendations. The project is reviewed and a conclusion is described.
* At the end of the report is the appendix, the appendix contains the user manual of the welding machine

# Definition fase

The welding machine consists of various parts and performs all kinds of actions. What the welding machine does is described below.

## Background project

The welding machine is used for welding a metal wire mesh to a metal frame. This creates a fence. The parts are placed in the machine by an operator, welded together by a welding robot and removed from the machine by a handling robot. Only the 200 x 200 cm frame and the 200 x 200 cm mesh can be welded with the machine. The robots are not accessible through an enclosure. There is a door in the casing. At the operator position at the front is a control panel with an emergency stop, start button, stop button and a reset button. In this project, safety is programmed to receive notifications if an unsafe situation has arisen. When the hardware of the machine is developed, the machine will have to regulate safety via safety relays (special circuits for a safety circuit in a machine). Figure 1 is a depiction of the machine. The left robot is the welding robot and the right robot is the handling robot.



*Figure 1 The welding machine, left the welding robot and right the handling robot*

## Functional specification

The machine should meet all kinds of specifications. Think of starting up, stopping and switching off. All specifications of the welding machine can be read below.

1. The welding robot will weld 2 parts together to create a fence.
2. Simple and understandable user interface
3. The machine has a boot cycle where the machine is initialized.
4. For safety, the door must be closed, with the door open, the machine cannot start the process.
5. The emergency stop ensures that the machine can stop at any time in an unsafe situation, this also stops the robots.
6. After an emergency stop, the products must be removed and the reset button pressed. The products can no longer be used because the weld has been interrupted
7. The door and emergency stop are part of safety, after interrupting them, the machine must be reset. The software only listens to this, the hardware carries out the safety by means of specifications according to the machine guidelines.
8. When the parts are inserted, it is necessary to press the start button to start the process.
9. When the start button is pressed after welding, the handling robot takes the end product out of the machine.
10. The machine has a shutdown cycle where the machine de-initializes equipment before the whole machine actually shuts down.

## Technical specification

The welding machine consists of all kinds of parts and is engineered with different programs. The following programs and brands have been used for the welding machine.

1. Programming language: C
2. Development environment: QT-Creator
3. UML diagram editor: PlantUML
4. Robot brand: ABB
5. Door switch: Euchner
6. Peripherals: Siemens

Figure 2 shows the user interface of the machine. User interaction is done through keyboard buttons such as:

1: To put part 1 in the machine

2: To put part 2 in the machine

s: Emergency stop

d: Door open

x: Shut down machine



*Figure 2 Screen shot of the user interface*

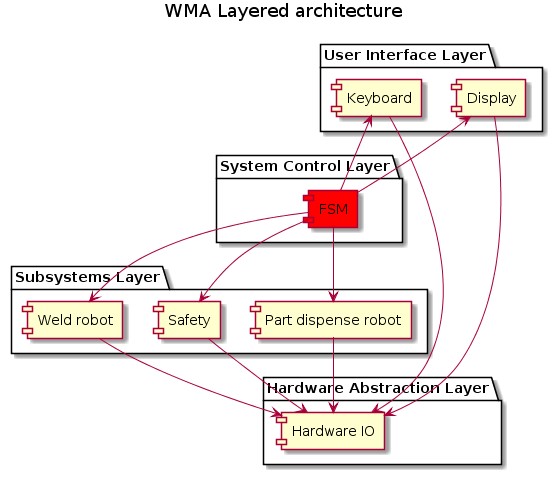
# Design fase

This chapter describes the technical design. First, the layered architecture of the welding machine. This shows the hardware / software structure within which the welding machine has been programmed and tested.

Following is the UML-diagram of the statemachine and the explanation of how the FMS is programmed.

## Layered Architecture

Figure 3 shows how the system is built up in layers.



*Figuur 3 Layered architecture of the welding machine*

### User Interface Layer

In this layer, the operator can view the status of the machine and control the machine by means of push buttons.

### System Control Layer

The FSM (finite state machine) controls the machine, the states are executed here. The machine can be in one state at a time and moves to another state by means of an event. Functions are also executed in the FMS.

### Subsystems Layer

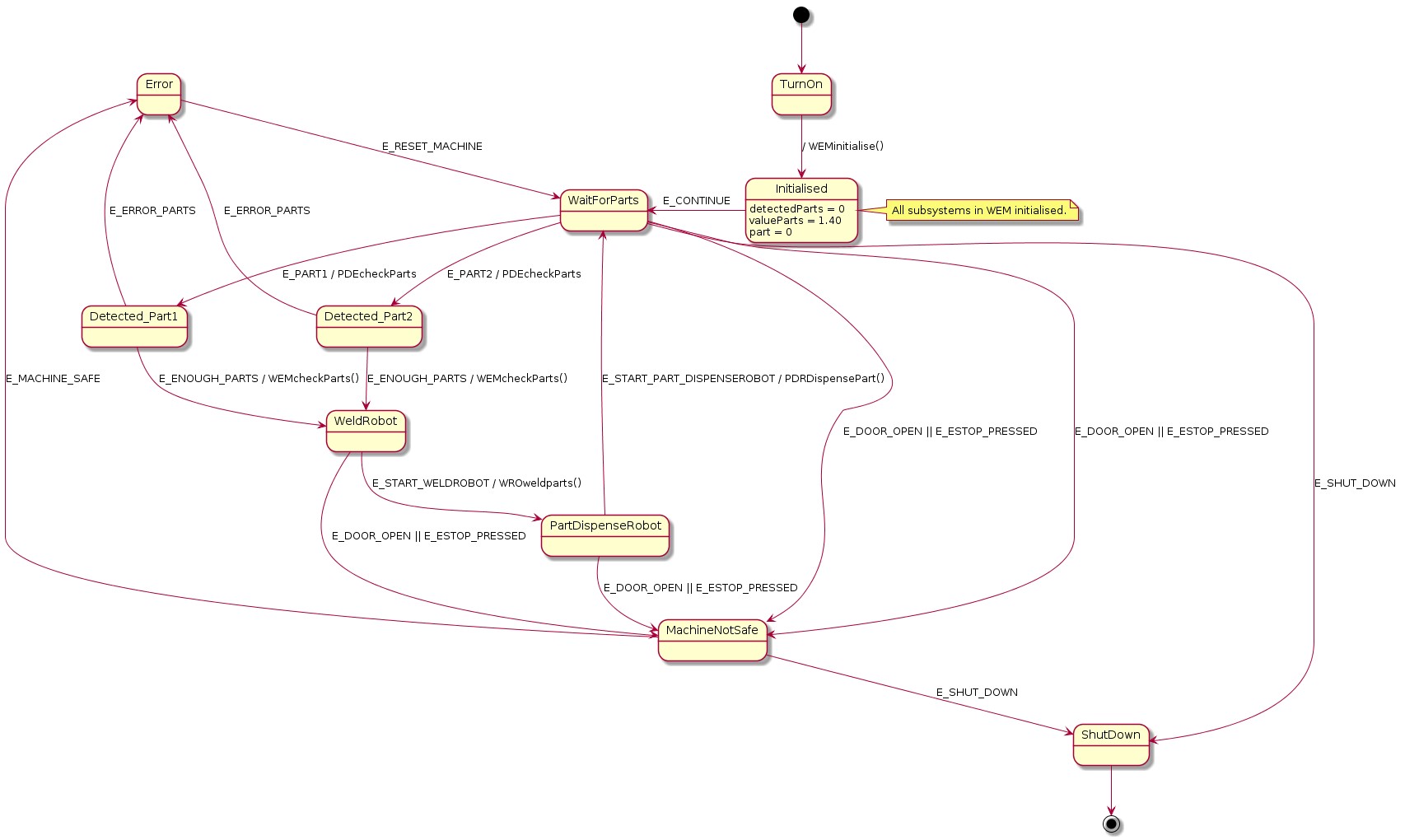
This layer contains the systems that are controlled from the FSM. For example the two robots.

### Hardware Abstraction Layer

Here the actual hardware is controlled. This is controlled from the subsystems, but can also be controlled directly via the User Interface Layer, for example for testing purposes.

## State machine

The design of the statemachine is engineered from a UML diagram. The UML state diagram of the welding machine is shown in figure 4. In the UML state diagram you can read which states the welding machine has and which events (for example a key on the keyboard) are required to enter the state..



*Figure 4 UML diagram welding machine*

The first state the machine enters after start-up is named Initialized. The welding machine is initialized in this state. The detectedParts is set to 0 because there are no parts in the machine yet and the value of the 2 parts together is set here.

After initialization, the machine enters the state WaitForParts. In this state the machine waits for a part to be loaded. The machine can be switched off from this state. It is also possible, if an unsafe situation arises, to operate the emergency stop so that the machine goes to the MachineNotSafe state. The machine also goes to the MachineNotSafe state when the door is opened.

If the machine detects parts, the WMAcheckparts function determines whether there are enough parts in the machine or whether the wrong parts are in the machine. The function works by adding the value of the parts. If the 2 parts together are the valueParts, there are enough parts in the machine. As long as the value is below the valueParts, there are not enough parts in the machine. If the value is above the valueParts then there are too many parts in the machine and the machine goes to the error state.

The states weldRobot and partDispenseRobot are simple states in which the robots can be started by means of the start button. From this state, the emergency stop can be operated in an unsafe situation. In practice, the robot will then stop moving, with this state machine this is not feasible because it does not communicate with the robot.

When the emergency stop is activated or the door is opened, the machine goes to the state MachineNotSafe, to get out of this state, it has been decided the user has to use a key on the keyboard to choose whether the machine is safe. In practice, this will be done automatically when the emergency stop is deactivated or the door is closed. After the machine is declared safe, the machine will go to the Error state because the machine needs to be reset. The machine can be switched off from the MachineNotSafe state.

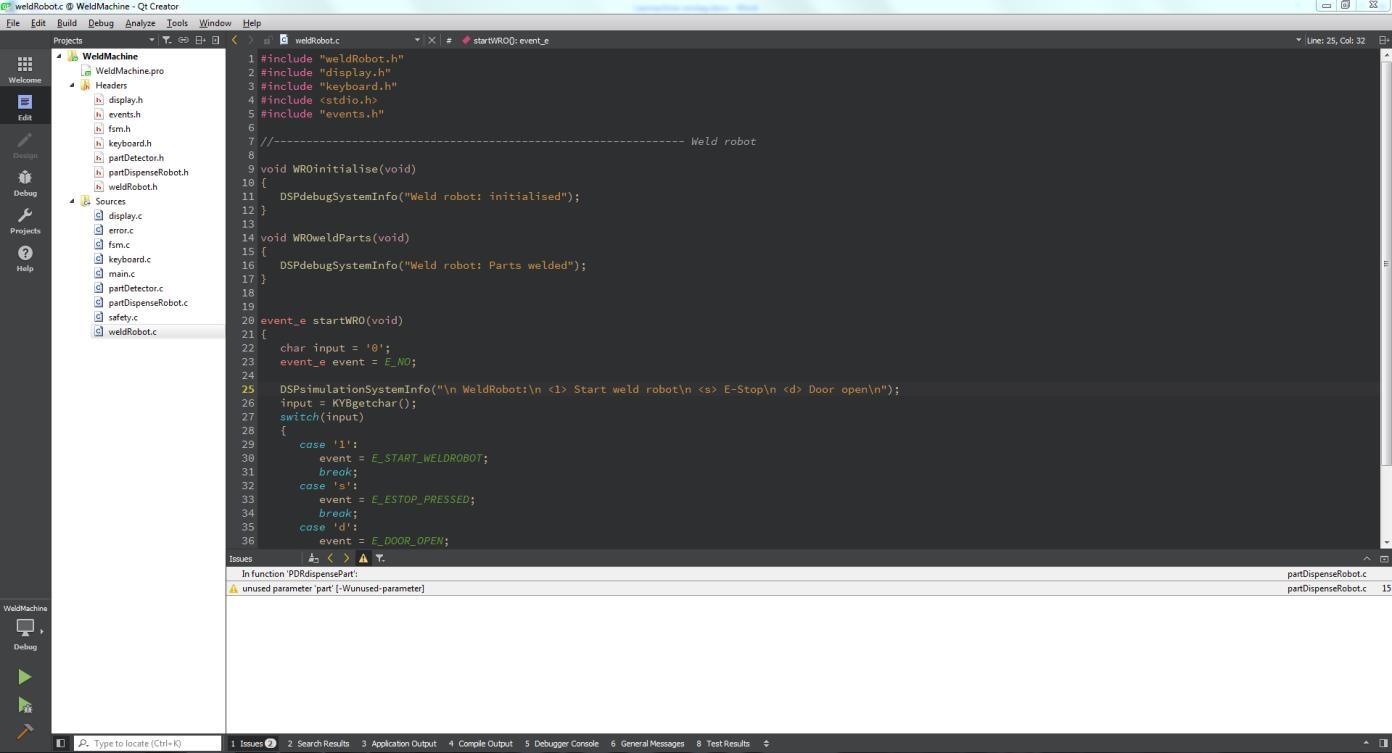
The state Error causes the machine to be reset by pressing the "r" key on the keyboard. Because the user has to remove the parts from the machine in the Error state, the value of the parts that were in the machine is set to 0.

# Realization and testing

The way in which the welding machine is realized can be read in the chapters below. A test was carried out to meet all the correct customer requirements with the welding machine. All existing scenarios have been tested and it is examined how the machine reacts to them.

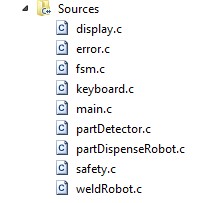
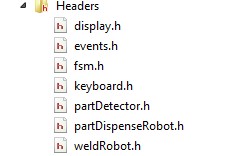
## Development environment

The project "Weldmachine" was developed with the program QT-Creator. Figure 5 shows what the development environment looks like.



*Figure 5 Development environment Qt-Creator*

In the left part is the .pro file with the header files and sources below it. On the right is the development environment where the code is programmed. At the bottom are the issues which is an important function of QT-Creator to be able to debug. Figure 6 shows the header files belonging to the welding machine and figure 7 shows the sources.



*Figuur 6 Headerfiles Figuur 7 Sources*

Below is listed the content of the .pro file:

TEMPLATE = app

CONFIG += console

CONFIG -= app\_bundle

CONFIG -= qt

CONFIG += c11

SOURCES += main.c \

fsm.c \ display.c \ keyboard.c \ partDetector.c \ partDispenseRobot.c \ weldRobot.c \ safety.c \

error.c

HEADERS += \

fsm.h \ display.h \ events.h \ keyboard.h \ partDetector.h \ partDispenseRobot.h \ weldRobot.h

## Userinterface

The user interface shown in figure 8 consists of a part that shows the state in which the machine is located. Instructions for the user are provided below this section. These instructions are, for example, pressing a button or an action on the machine such as removing parts.



*Figure 8 User interface*

## Acceptance test

In order to deliver the machine to the customer properly, it is necessary to thoroughly test the machine. Figure 9 shows a test table of the tests performed.

The sequence of testing is as follows:

* Test 1: Start application
* Test 2: Input from the user
* Test 3: Error input from the user
* Test 4: Correct parts in the machine
* Test 5: Wrong parts in the machine
* Test 6: Shut down application

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Project name: WeldMachine** | | | | **Date:** 29.12.2017 | | |
| **Test case ID: 001** | | **Name: L. v.d. Hardenberg** | |  | | |
| **Function/module/program under test:**  **WeldMachine** | | | |  | | |
| **Test** | **Action/input** | | **Expected result** |  | **Pass**  **Fail** | **Actual result if test has failed** |
| 1. | Application start properly | | Displayed main manu |  | Pass |  |
| 2. | Application accept user input (all variants) | | Application accepted all defined imputs |  | Pass |  |
| 3. | Application handle user input errors | | Error message shown on the screen |  | Pass |  |
| 4. | function calculate parts works properly | | 1.40.  Part 1 = 1  Part 2 = 0.40 |  | Pass |  |
| 5. | Application handle wrong parts | | If value > 1.40  Wrong parts in machine  displayed |  | Pass |  |
| 6. | Application ends properly | | Shutdown screen displayed, deinitialisation of subsystems shown |  | Pass |  |

*Figure 9 Test table*

The table shows that the tests meet the requirements of the machine.

## Documentation

The documentation is made with Doxygen. Doxygen creates an HTML page containing the code and the user manual. The page is automatically created by means of comments in the C-code. Doxygen can be downloaded at [www.doxygen.org.](http://www.doxygen.org/)

# End result and recommendations

The purpose of the application is to control a welding machine by means of a user. Safety is an important aspect in this welding machine. All requirements have been met.

The following requirements have been realized for the welding machine:

1. Start cycle
2. User input using a keyboard
3. A state the welding machine is waiting for parts
4. State the weldRobot can be controlled
5. State the partDispenseRobot can be controlled
6. Safety state when the machine door is open or the emergency button has been pressed
7. Errorstate when an error occured
8. Shut down cycle

The tests of the machine have all been passed correctly. The user input works fine and if the user enters a wrong input it will shown on the display. The function that adds the parts together also works correctly and ensures that there is feedback whether there are enough, insufficient or the wrong parts in the machine. The safety in the machine works correctly in the software and gives a message on the display that something is wrong with the safety. The shutdown cycle also works well and de-initializes the machine so that the machine can be shut down properly.

If development for this machine is continued in the future, communication with the robot should be investigated. The robots are programmed in the development environment of the robot brand. So a connection with this application and the robot application should be realized. The safety now works in software to receive notifications if an unsafe situation arises, but if the machine hardware is built, the safety must also interrupt the robots hardware.

# Attachments:

## C - Source Code

## User Manual