

FM4017 Project 2021

# **Microcontroller based Automatic Optical Inspection Module for Fault Detection in Printed Circuit Board Assembly**

MP-32-21

**Course:** FM4017 Project, 2021

**Title:** Microcontroller based Automatic Optical Inspection Module for Fault Detection in Printed Circuit Board Assembly

This report forms part of the basis for assessing the student's performance in the course.

**Project group:** MP-32-21

**Group participants:** Jon Eilert Liane

**Supervisor:** Hans-Petter Halvorsen, Saba Mylvaganam

**Project partner:** **Hapro Electronics**

**Summary:**

A major part of the production at Hapro Electronics is manual assembling of PCBs. Due to the presence of human errors in these assemblies, an AOI has been developed.

The Goal of this project is to solve two challenges. Implement a communication interface between the AOI and the soldering machine and create and implement a user interface to allow an operator to interact with the AOI.

For communicating between the AOI and the Soldering machine, several communication standards were researched. For the user interface both type of interface and the programming language for implementation have been researched.

A graphical user interface has been created to allow the operators to interact with the AOI, the features of this interface have been tested and a manual for operators have been created. The communication between the AOI and soldering machine could not be completed but the method and approach have been planned. This communication will be performed by creating an Interface board that can communicate with the AOI and the soldering machine by utilizing the SMEMA-9851 interface standard as this is natively supported by the soldering machine.

# Preface

This project was started, due to increased manual assembly of PCB at Hapro Electronics. Due to this increase an AOI was developed. The AOI is required to be able to communicate the result of the inspection to a soldering machine, as well as being able to receive input from an operator. This project and the work described in this report is part of the FM1410 Project course in the Industrial IT and automation industry master course at the University of South-Eastern Norway.

Porsgrunn, 19.11.2021, Jon Eilert Liane

# Contents

<b>1</b>	<b>Introduction</b>	<b>7</b>
1.1	Background	7
1.2	The purpose of the project	8
<b>2</b>	<b>PCBA</b>	<b>9</b>
2.1	Manual assembly	9
2.2	Automatic assembly	9
<b>3</b>	<b>Communication standards</b>	<b>10</b>
3.1	Communication alternatives	10
3.1.1	<i>OPC UA</i>	10
3.1.2	<i>EtherCAT</i>	10
3.1.3	<i>SMEMA-9851</i>	10
3.2	Soldering machine	11
3.2.1	<i>Interfaces compatible with the soldering machine</i>	11
3.3	AOI	12
3.3.1	<i>Interfaces compatible with the Raspberry pi</i>	13
3.4	Communication Between the AOI and soldering machine	14
<b>4</b>	<b>User interface for the AOI</b>	<b>15</b>
4.1	Introduction	15
4.2	Requirements	15
4.3	Alternatives for user interfaces	16
4.3.1	<i>Graphical user interface</i>	16
4.3.2	<i>Command line interface</i>	16
4.3.3	<i>User interface for the AOI</i>	17
4.4	Selecting Programming language for the Interface	18
4.4.1	<i>Python</i>	18
4.4.2	<i>C#</i>	18
4.4.3	<i>Programming language used for the interface</i>	18
4.5	Developing the GUI	19
4.5.1	<i>Creating a new template</i>	20
4.5.2	<i>Main window of the program</i>	22
4.5.3	<i>Load product</i>	23
4.5.4	<i>Creating a new product</i>	24
4.5.5	<i>View templates</i>	26
<b>5</b>	<b>Communication System</b>	<b>27</b>
5.1	Creating an interface board	27
5.1.1	<i>Designing the interface board</i>	28
5.1.2	<i>Communication standard used for communicating between the AOI ant the soldering machine</i>	29
<b>6</b>	<b>Testing the SMEMA communication</b>	<b>30</b>
6.1	SMEMA communication	30
6.1.1	<i>Features to be tested</i>	30
6.1.2	<i>Deploying test</i>	30
6.2	GUI	30
6.2.1	<i>Features to be tested</i>	31
6.2.2	<i>Deploying tests</i>	31
<b>7</b>	<b>Documentation</b>	<b>33</b>

7.1 GUI operator manual and SMEMA circuit.....	33
<b>8 Discussion.....</b>	<b>34</b>
8.1 The GUI challenges and improvements .....	34
8.2 The SMEMA interface board .....	34
<b>9 Conclusion .....</b>	<b>35</b>
<b>10References.....</b>	<b>36</b>

# Nomenclature

## Abbreviations:

AOI – Automatic optical inspection, name of the machine and software that inspects the PCB

EMS – Electronic Manufacturing Service

ESD – Electro Static Discharge

GPIO – General Purpose Input/Output

GUI – Graphical user interface.

PCB – Printed circuit boards.

PCBA – Printed circuit boards assembly.

SMT – Surface Mount Technology.

## Terms:

AMP 206043 – Contacts used with the SMEMA machine interface

# 1 Introduction

Hapro Electronics provides Electronic Manufacturing Service (EMS) for the electronics and data industry and produces prototypes and deals with industrialization, automatic and manual assembly, box building and testing. The Automatic optical inspection (AOI) is an inspection station developed at Hapro Electronics using off the shelf components to minimize costs and increase scalability.

## 1.1 Background

Hapro electronics produce products from several different companies. Most of these products contain one or more Printed Circuit Board (PCB), that has been assembled and soldered at Hapro Electronics.

PCB consists of a network of copper traces and insulators which connects the different components based on the circuit diagram from customers. Printed Circuit Boards Assembly (PCBA) is one of the major activities in Hapro Electronics. This process is often automatic, however in cases where the automatic equipment cannot assemble the PCB, the PCBA must be done manually. The reasons for this can be low-volume production runs, thru-hole components on low-volume runs, custom parts and sub-assemblies attached to a board, “stand-offs” and press-fit connectors.

As the PCBA that’s done manually is vulnerable to human errors, Hapro electronics have developed an AOI to decrease the number of faulty PCBs to go through the soldering process, as fixing the faulty PCB after soldering is a time consuming and therefore expensive process.

## 1.2 The purpose of the project

The main goal of this project is the development of a versatile communication system between the AOI, and the soldering machine based on IPC-SMEMA-9851. The system for AOI will be based on Raspberry Pi and a Raspberry Pi camera module. The signals for the operator and the soldering machine will be based on the results of the AOI inspection. Figure 1.1 shows a sketch of the intended setup of the system.

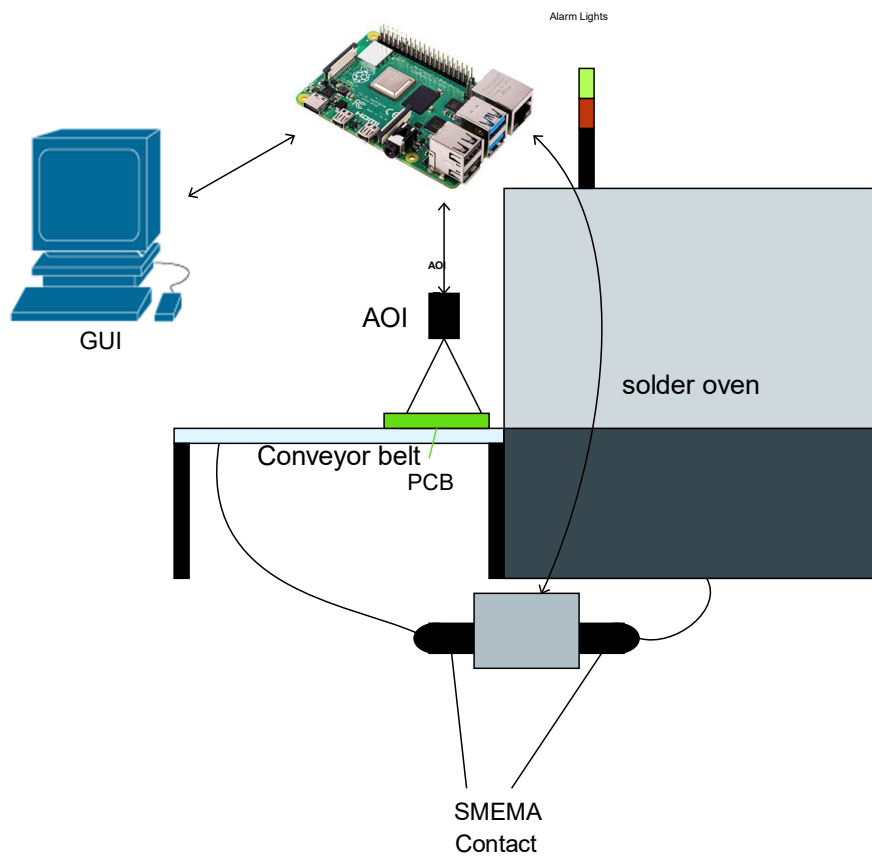


Figure 1.1 System sketch of the AOI, soldering machine and the conveyor belt for moving the PCB.



## 2 PCBA

### 2.1 Manual assembly

While, at Hapro Electronics there are several products being manufactured without humans assembling the PCB, there are still some products that require partial manual assembly. This can be due to some complexity of the assembly, like contacts that need to be connected to pins. Other components may be too large to be compatible with the equipment at the factory.

Some recurring challenges are:

- Interaction with the circuit board can cause ESD damage if caution is not taken
- Humans may forget components
- Humans may assemble components wrong
- Precautions needs to be taken to prevent grease and dirt coming onto the solder pins

The Goal of the AOI system is to reduce the significance of human errors that result in re-soldering of the circuit board.

### 2.2 Automatic assembly

The most efficient way to create a circuit board is to use standardized parts in an SMT machine, Figure 2.1 shows an SMT assembly machine at Hapro Electronics.



Figure 2.1 SMT assembly machine at Hapro Electronics, this is a FUJI NXT II machine

These machines are capable of assembling several thousand components per hour[1]. This allows for high product output with high accuracy. The FUJI NXT II machine Hapro electronic use also have a built in AOI system for quality control. Since these machines are modular and can be fitted with a soldering station, they are highly automated and is safer for workers than the manual assembly lines. However, these machines also require the PCB to only use standardized parts that can be fitted on the component wheels shown on the right in Figure 2.1. This means that PCBs that requires larger components cannot be used with these machines.

## 3 Communication standards

This chapter will discuss the communication between the different machines in the project. The following sub chapters will focus on alternatives for communication between the AOI and the soldering machine as well as the communication standards they support.

### 3.1 Communication alternatives

There are several different machine-to-machine communication standards available. This sub chapter will provide some background into some alternatives for communication between the AOI and the soldering station

#### 3.1.1 OPC UA

OPC unified architecture is a machine-to-machine communication protocol that is designed for industrial automation[2]. The protocol has wide support in automation equipment and enables scalability through cross platform support[2]. There is also built-in security, like support for encryption and server authentication.

#### 3.1.2 EtherCAT

EtherCAT is a wired machine-to-machine communication protocol. EtherCAT is based on ethernet and requires the devices to be connected through the RJ45 ports[3]. The EtherCAT is a popular communication standard in industrial automation for enabling real time applications.

#### 3.1.3 SMEMA-9851

The IPC SMEMA-9851 mechanical equipment interface standard is a communication standard designed for mechanical equipment in the electronics manufacturing industry. The standard uses a set of signals to communicate between the different machines[4]. Dedicated signals include a machine is ready signal and a board available signal. In addition, the standard has dedicated pins for auxiliary equipment and an optional signal for failed boards.

## 3.2 Soldering machine

The soldering machine for the project is a VERSAFLOW 3/66 selective soldering system as seen in Figure 3.1.



Figure 3.1 Versaflow3/66 selective soldering system[5]

### 3.2.1 Interfaces compatible with the soldering machine

The Versaflow system has support for 4 interfaces[6]. The IPC-SMEMA-9851 which is used to communicate with the conveyor belts on each side of the soldering machine, a scanner interface, a traceability interface, and a transponder reading devices interfaces[6].

Since the purpose of the project is to create a form of control before the components are soldered onto the PCB, a device will have to be connected between the conveyor belt and the soldering machine to control the machines as shown in Figure 3.2.

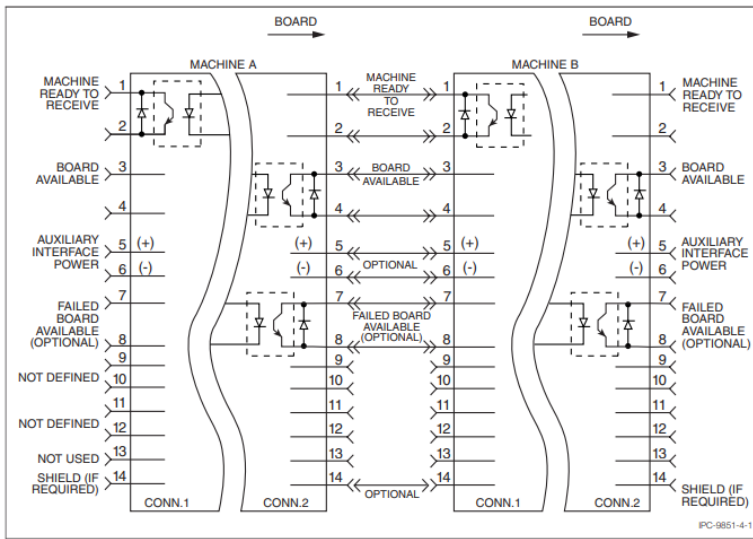


Figure 3.2 General schematic for the SMEMA-9851 interface[4]

The interfacing device will be required to utilize the “machine ready to receive signal”, the “board available” signal and the “failed board” signal shown in Figure 3.2.

### 3.3 AOI

The AOI is a python program that runs on a Raspberry Pi, see Figure 3.3. This enables the AOI to be flexible in terms of communication as there are several options available. The AOI used for this project is a solution that has been developed at Hapro Electronics.



Figure 3.3 The Raspberry pi 4 model B is the core computer of the AOI

### 3.3.1 Interfaces compatible with the Raspberry pi

the raspberry pi supports several interfaces for transferring information. Table 3.1 shows some of the supported communication standards for raspberry pi

Table 3.1 Interfaces for communication supported by Raspberry Pi

<b>Protocol</b>	<b>Function</b>	<b>description</b>
<b>I2C</b>	Single wire serial bus communication	Single wire synchronous data communication. Typically used for communicating between microcontrollers or from supported sensors to microcontrollers[7].
<b>SPI</b>	Serial interface bus	Serial peripheral interface. SPI is a synchronous bus used for communication between the microcontrollers or other peripheral [8].
<b>Bluetooth</b>	Short range wireless communication	2.4 GHz communication used for transferring data between a computer and peripheral equipment[9]. The protocol is capable of 2Mbit/s data transfer
<b>802.11n</b>	Wireless communication	Ethernet based 2.4 or 5 GHz wireless data communication[10]. Most used for connecting computers to the internet.

In addition to the communication protocols showed in Table 3.1, the Raspberry Pi also have general purpose inputs and outputs (GPIO), these can be used to send and receive signals to devices that do not support other communication standards.

### **3.4 Communication Between the AOI and soldering machine**

The communication for the system will be the SMEMA-9851 standard. The reason for this is that using any other way of communication between the two systems would require intrusive modification of the soldering machine and the conveyor belt. This kind of modification could potentially void the warranty of the machines.

Using the SMEMA standard allows for communication with the soldering machine without any modification to the machine. To use the AOI, there will be a need to create a circuit that can interrupt the signal from the conveyor belt and send it once the inspection is done.

# 4 User interface for the AOI

This chapter is concerned with the methods and solutions for the user interface. Subchapters will establish the requirements for the user interface and give some background into the alternatives.

## 4.1 Introduction

Any software that requires humans to make decisions need a user interface. This interface can be a simple terminal program, physical buttons that is connected to a machine, like a keyboard or a mouse or a graphical interface with digital buttons[11].

## 4.2 Requirements

This subchapter will list the required functionality for the interface. The requirements define how the interface will behave with the operator and the AOI. The main features of the GUI will be:

- The interface needs to alert the operator when an error occurs in the assembly.
- An operator should be able to create a new program for a new assembly.
- Load programs for PCB.
- Create new templates for finding components in an existing program.
- Pass or fail PCBs that the AOI have deemed faulty.
- View templates for the selected program.

## 4.3 Alternatives for user interfaces

This subchapter will explore the different alternatives for the user interface and the opportunities and challenges to these options.

### 4.3.1 Graphical user interface

The graphical user interface is a graphical control panel for a program. This is the type of interface seen when using most commercial programs. It may include windows with buttons, text boxes and other graphical features.

Since it is a common user interface, there are several tools available for creating user interfaces in almost every programming language. A mind map for alternatives in a graphical interface is found in Figure 4.1.

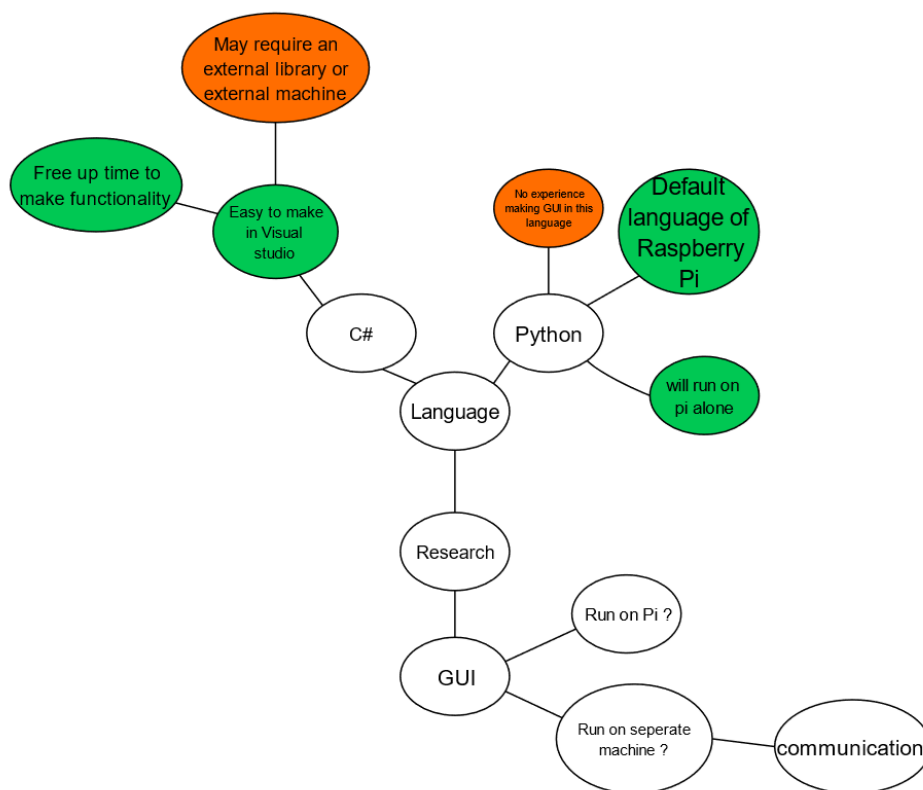
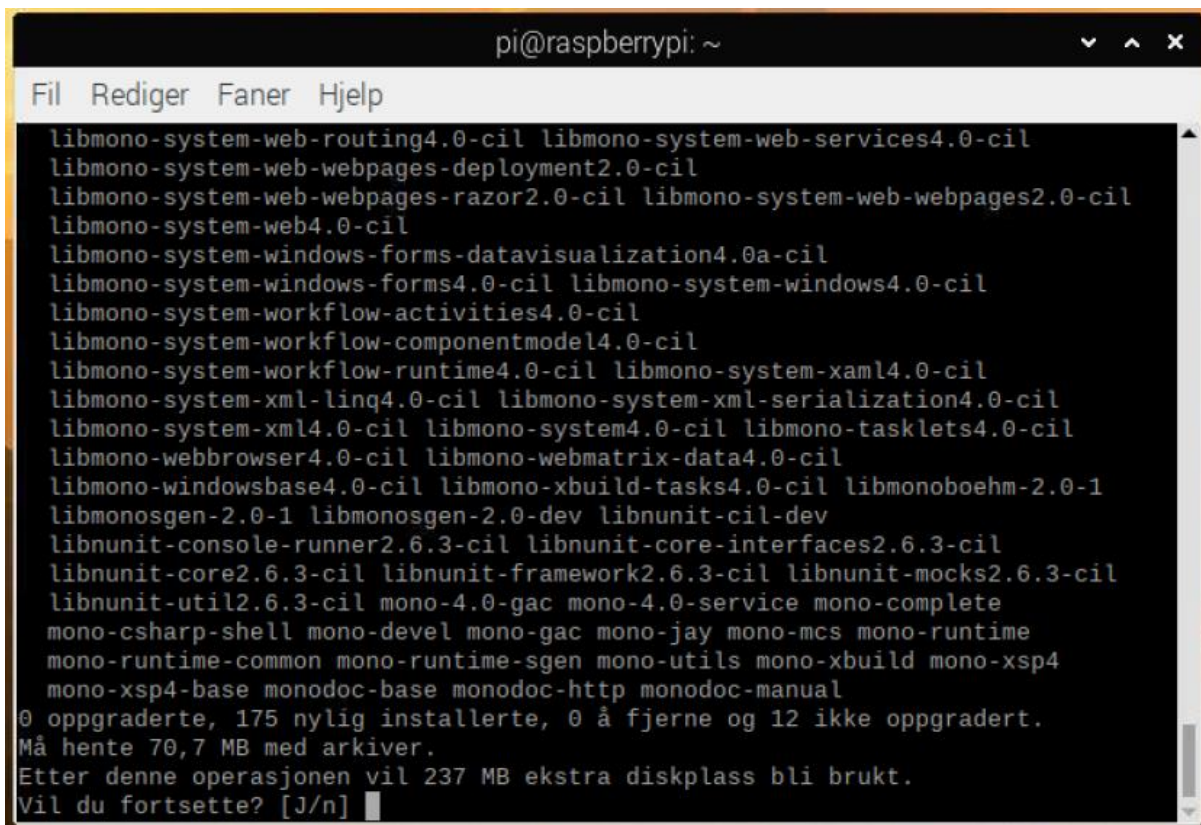


Figure 4.1 Mind map for the GUI, the green and orange indicate positive and negative consequences for each choice.

### 4.3.2 Command line interface

A command line interface consists of a window in which the operator can type commands for the program. See Figure 4.2





```
pi@raspberrypi: ~
Fil Rediger Faner Hjelp
libmono-system-web-routing4.0-cil libmono-system-web-services4.0-cil
libmono-system-web-webpages-deployment2.0-cil
libmono-system-web-webpages-razor2.0-cil libmono-system-web-webpages2.0-cil
libmono-system-web4.0-cil
libmono-system-windows-forms-datavisualization4.0a-cil
libmono-system-windows-forms4.0-cil libmono-system-windows4.0-cil
libmono-system-workflow-activities4.0-cil
libmono-system-workflow-componentmodel4.0-cil
libmono-system-workflow-runtime4.0-cil libmono-system-xaml4.0-cil
libmono-system-xml-linq4.0-cil libmono-system-xml-serialization4.0-cil
libmono-system-xml4.0-cil libmono-system4.0-cil libmono-tasklets4.0-cil
libmono-webbrowser4.0-cil libmono-webmatrix-data4.0-cil
libmono-windowsbase4.0-cil libmono-xbuild-tasks4.0-cil libmonoboehm-2.0-1
libmonosgen-2.0-1 libmonosgen-2.0-dev libnunit-cil-dev
libnunit-console-runner2.6.3-cil libnunit-core-interfaces2.6.3-cil
libnunit-core2.6.3-cil libnunit-framework2.6.3-cil libnunit-mocks2.6.3-cil
libnunit-util2.6.3-cil mono-4.0-gac mono-4.0-service mono-complete
mono-csharp-shell mono-devel mono-gac mono-jay mono-mcs mono-runtime
mono-runtime-common mono-runtime-sgen mono-utils mono-xbuild mono-xsp4
mono-xsp4-base monodoc-base monodoc-http monodoc-manual
0 oppgraderte, 175 nylig installerte, 0 å fjerne og 12 ikke oppgradert.
Må hente 70,7 MB med arkiver.
Etter denne operasjonen vil 237 MB ekstra diskplass bli brukt.
Vil du fortsette? [J/n]
```

Figure 4.2 Command window in in Raspberry Pi.

This type of interface requires the operator to know the commands for the program and to type them correctly. The command line interface is simple to make relative to the GUI, the challenge with this form of user interface is to fulfil the requirements in a user-friendly way.

### 4.3.3 User interface for the AOI

The interface for the AOI will be created as a graphical interface as it is the most operator friendly solution. Using programming tools for designing the interface will simplify the development of the interface and fulfilling the requirements in the previous chapter.

## 4.4 Selecting Programming language for the Interface

This chapter will focus on alternatives for programming the interface. There are several different languages for programming. The main limiting factor in this project is the compatibility with the raspberry pi and the tools available for creating a user interface.

### 4.4.1 Python

Python is the most common language to use with Raspberry Pi. Python is an object-oriented language that is popular for computational tasks and neural networks. This language is less common for creating user interfaces, but there is some tools available like pyQT and Qt Designer by the Riverbank Computing[12]. Qt Designer offers several python classes and creating an interface uses a graphical tool for designing interfaces[12].

### 4.4.2 C#

Like python C# is an object-oriented language. C# is not natively supported in raspberry pi, but the mono-project have created a software platform for running applications that uses the .net framework on other operating systems than windows[13]. Mono is sponsored by Microsoft and enables a GUI created as a windows forms application to run on a Raspberry Pi[13].

A C# GUI can be created in the graphical designer in Microsoft visual studio, which has all the features needed for the GUI in this project, particularly C# has graphical features for displaying and editing images. Creating the GUI in C# also has the advantage of creating an .exe file that can store some of the data between each run without the need for creating an external file.

### 4.4.3 Programming language used for the interface

The interface will be programmed in C# as the designer in visual studio enables easy image editing which is a key feature for the GUI. Any GUI in python would either need to be designed entirely in code or use a third-party designer as mentioned before.

## 4.5 Developing the GUI

This sub chapter will follow the development of the GUI. The following subchapters will go through the creation of each of the feature described in the requirements chapter. The features will be created and presented in the following subchapters in order of complexity. The features can be seen in the use case diagram in Figure 4.3.

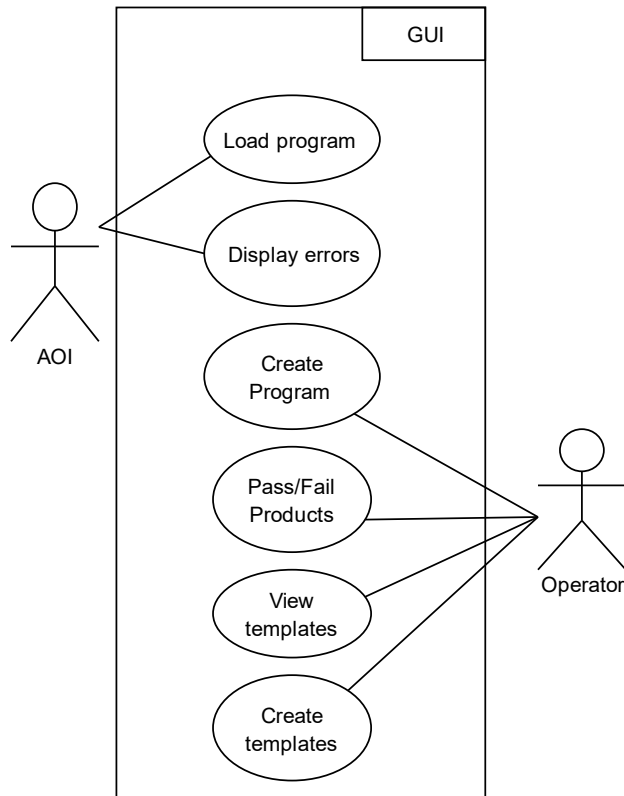


Figure 4.3 Use-case diagram for the user interface with AOI and the operator

### 4.5.1 Creating a new template

The first feature of the program to be created is the create template feature. As shown in Figure 4.4, the purpose of this feature is to allow the operator to add a search for a new component to an existing product. This feature can be useful if a product gets a new revision, or a component is added to the PCB.

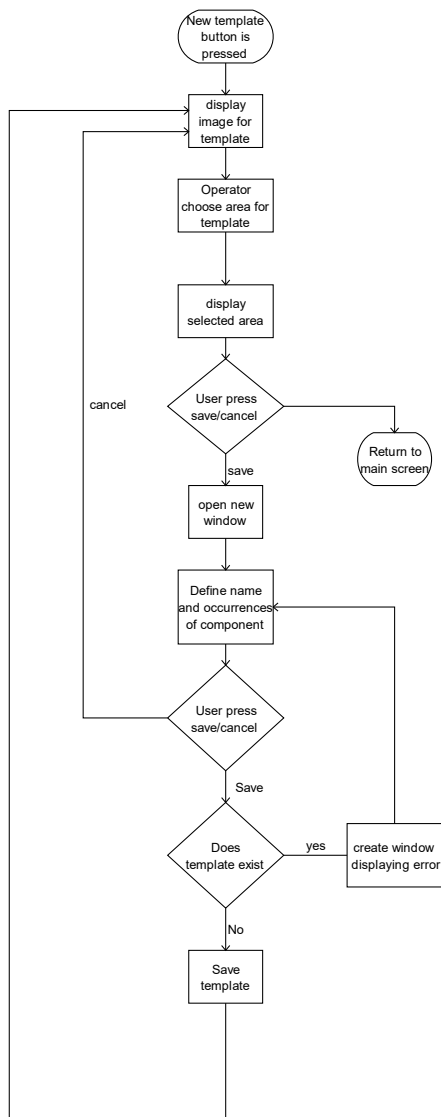


Figure 4.4 Flow chart for creating a new template for product



Figure 4.5 New template window using a Raspberry Pi as an example

The function can be accessed by the operator from the main screen of the program. When the feature is opened, a new window is displayed see Figure 4.5. In this window shows an image of the product and allows for creating and saving new templates for the AOI. There are also two buttons in this window. The cancel button returns the program to the main window. The save button open a new window, see Figure 4.6. In this window, the operator can input a name and the number of occurrences for the template. The number of occurrences dictates how many times the AOI will search for a unique occurrence of that component. This number will be added on the end of the filename before the file extension as this is how the AOI reads the occurrences.

Figure 4.6 Window for saving the created template to the AOI

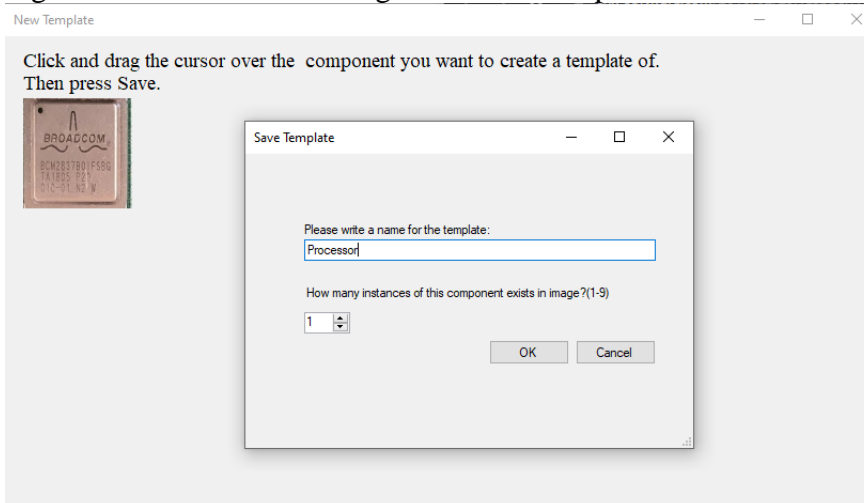


Figure 4.6 Window for saving the created template to the AOI

## 4.5.2 Main window of the program

The first window to be displayed when opening the GUI is the main window. This window features a picture box, with two buttons on the right as well as controls for accessing the other features of the GUI. This window will automatically update the picture box with an image if the AOI detects an error on the product. When a faulty product is detected, the operator will be able to select pass or fail for the product.

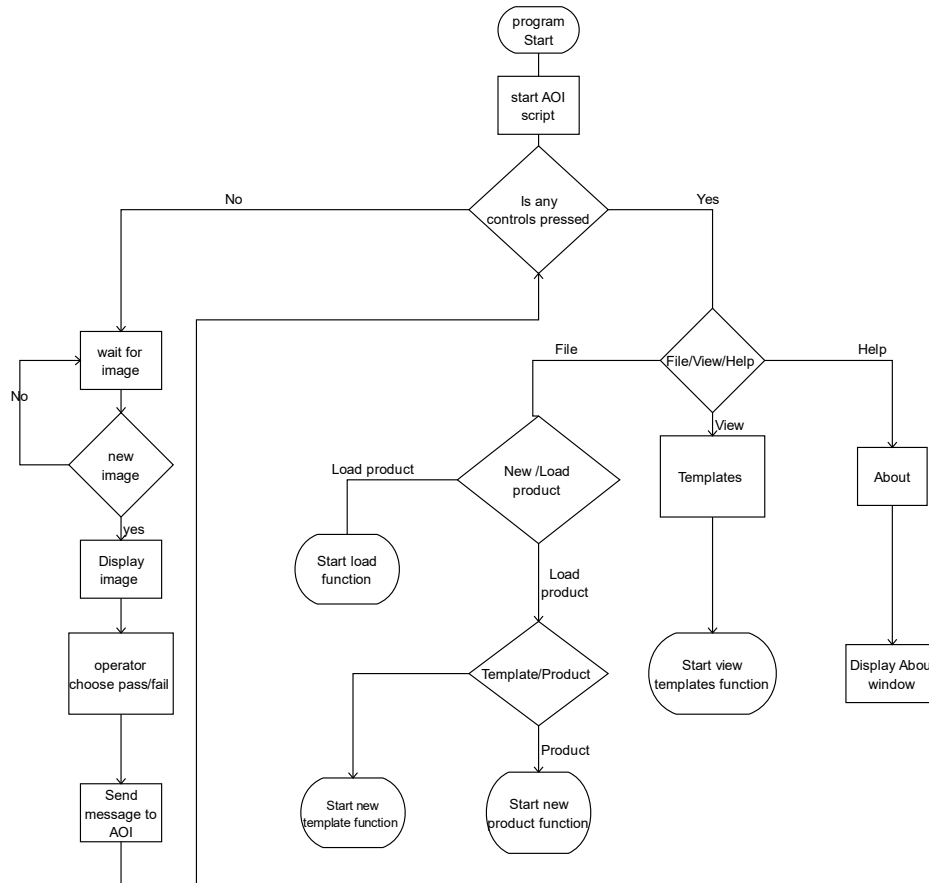


Figure 4.7 Flow chart for the main window of the GUI

On the backend of the main window, the GUI starts by loading a new process for running the control script for the AOI, see flow chart for the main screen in Figure 4.7. On the left side of the flowchart, the main loop of the program is illustrated. If no user input is given, the GUI will wait for the AOI script to flag an image for inspection. When an image is flagged the GUI will load the image to the main window and the program will then wait for a pass or fail response from the operator. one of the other features of the GUI is accessed, the main window will pause until the feature is closed. This means that no product can be passed while a different window than the main window is present.

### 4.5.3 Load product

The load product feature allows the operator to select the product the AOI will control. when the feature is selected, the GUI loads a window displaying the available products. See Figure 4.8.

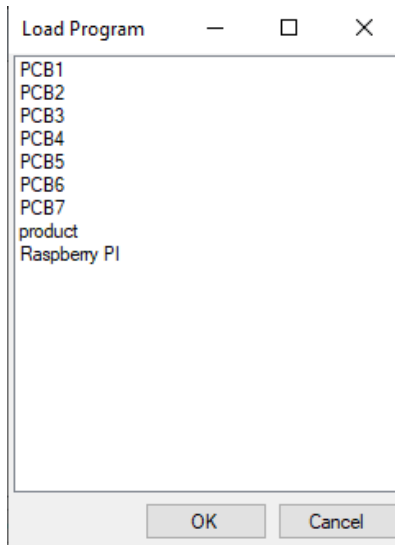


Figure 4.8 Window for selecting the product for the AOI

On the backend for this window, the GUI takes the product the user selected and writes the selection to a file. The AOI uses the value from this file as part of the file path to find the templates and parameters for the products. The flow chart for loading programs is shown in Figure 4.9

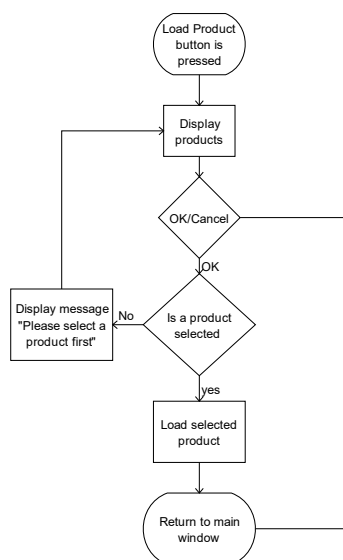


Figure 4.9 Flow chart for loading a program for a product into the AOI.

#### 4.5.4 Creating a new product

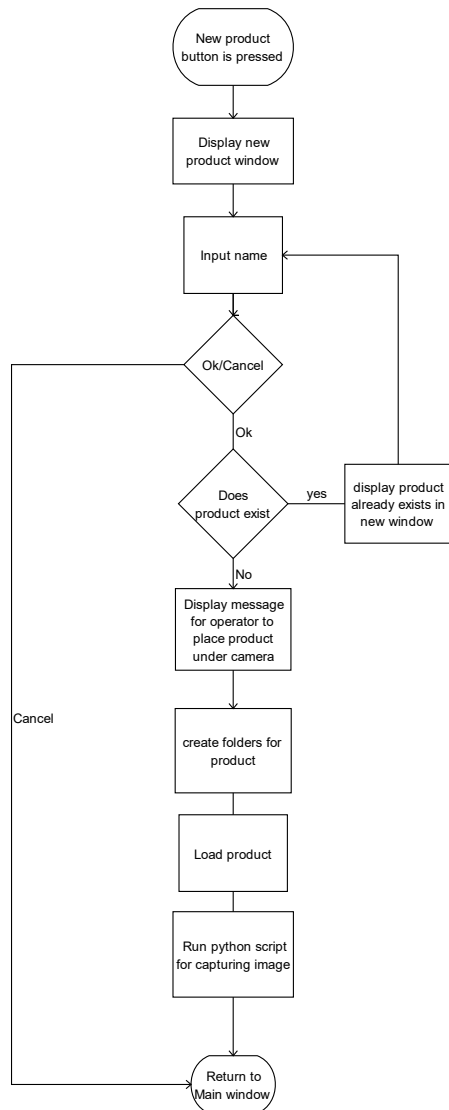


Figure 4.10 Flow chart displaying how a program for a new product is displayed

This feature allows the Operator to create a new product for the AOI. On the backend of this feature the GUI creates new folders for the AOI. on the frontend the GUI displays a window allowing the operator to input a name for the product. This window is shown in Figure 4.11.

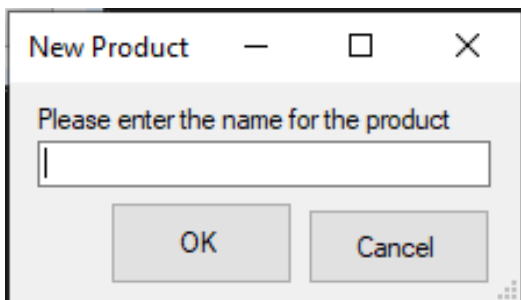


Figure 4.11 Window for naming the new product.



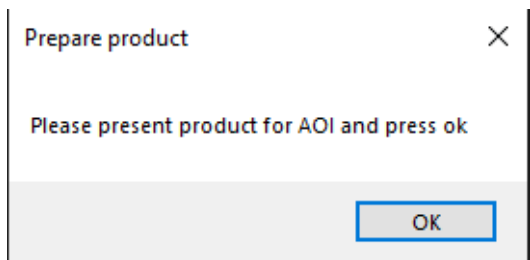


Figure 4.12 Waiting for operator to present the product for the AOI.

The product will then need to be presented to the AOI for capturing an image to use as a base for the templates, see Figure 4.12. When this window is closed the image is taken and saved. The new product is then loaded, and the main window is displayed.

### 4.5.5 View templates

This feature enables viewing and deleting existing templates for the products. When the feature is accessed, a new window appears see Figure 4.13. The window features three buttons, two for navigating the templates and one for deleting the current template. A flow chart displaying the behaviour of this feature can be seen in Figure 4.14.

When the operator accesses this feature, the GUI will look up all the files in the templates folder for the selected program. If any templates are available, the GUI will load the first template in the picture box in the window as shown in Figure 4.13. The text above the template shows the index of the current template as well as the total number of templates available for the product. If no templates have been made for the product the window will appear empty and the text at the top will read template nr 0 of 0.



Figure 4.13 View templates feature of the AOI

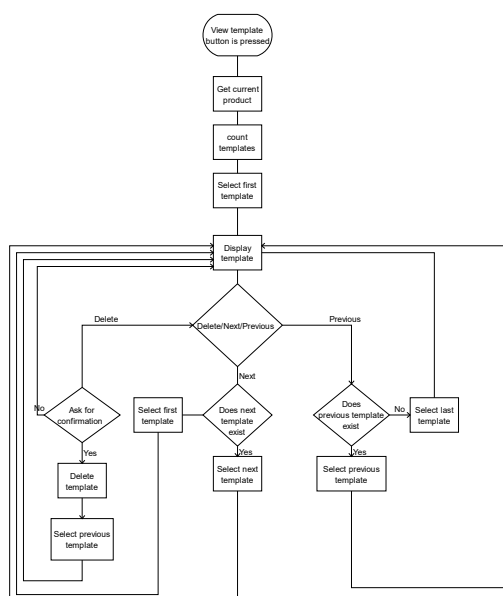


Figure 4.14 Flow chart for the View templates feature

# 5 Communication System

This chapter will go through the process of communication between the AOI and the Soldering machine and the conveyor belt. The following sub chapters will cover how the communication is achieved

## 5.1 Creating an interface board

The AOI requires a separate circuit for communicating with the soldering machine. This circuit will act as a connection point between the AOI, the conveyor belt and the Versaflow soldering machine. This setup is shown in the system sketch in Figure 1.1. The relevant connection pairs are shown in Table 5.1.

Table 5.1 connection pair used in the SMEMA connection board. The table is abbreviated from the IPC-SMEMA-9851 standard[4]

Connector	Function	Condition	Description
Pair 1-2	Machine Ready to Receive	Contacts Closed	Machine is ready to receive next board.
Pair 3-4	Board available	Contacts Closed	A board has been approved and is ready for the machine.
Pair 7-8	Failed board available	Contacts Closed	The current board has not passed.

Additionally, the board will be fitted with optocouplers for each signal as it is required by the SMEMA standard[4]. These are simple components consisting of a LED and a photoresistor and will be used to isolate the different circuits for the machines and the interface board.

### 5.1.1 Designing the interface board

The circuit for the board is shown in Figure 5.1. The circuit will be accessed by the conveyor and the soldering machine by the AMP 206043 contacts, shown in figure as J1 and J3 on each side of the circuit. The raspberry pi from the AOI will be connected through the RJ45 port J2. The AOI can then send ready to receive signal to the conveyor belt using the Q1 MOSFET transistor to drive current through the LED of the optocoupler, this will short the connection on the other side of the optocoupler signalling to the conveyor belt that the soldering machine is ready for a new PCB. The same solution is created for the other outputs at the downstream, with the board available signal and the failed board available signal on the downstream side of the circuit. The signals that are not defined or unused will be connected upstream and downstream directly, so that the SMEMA standard is upheld.

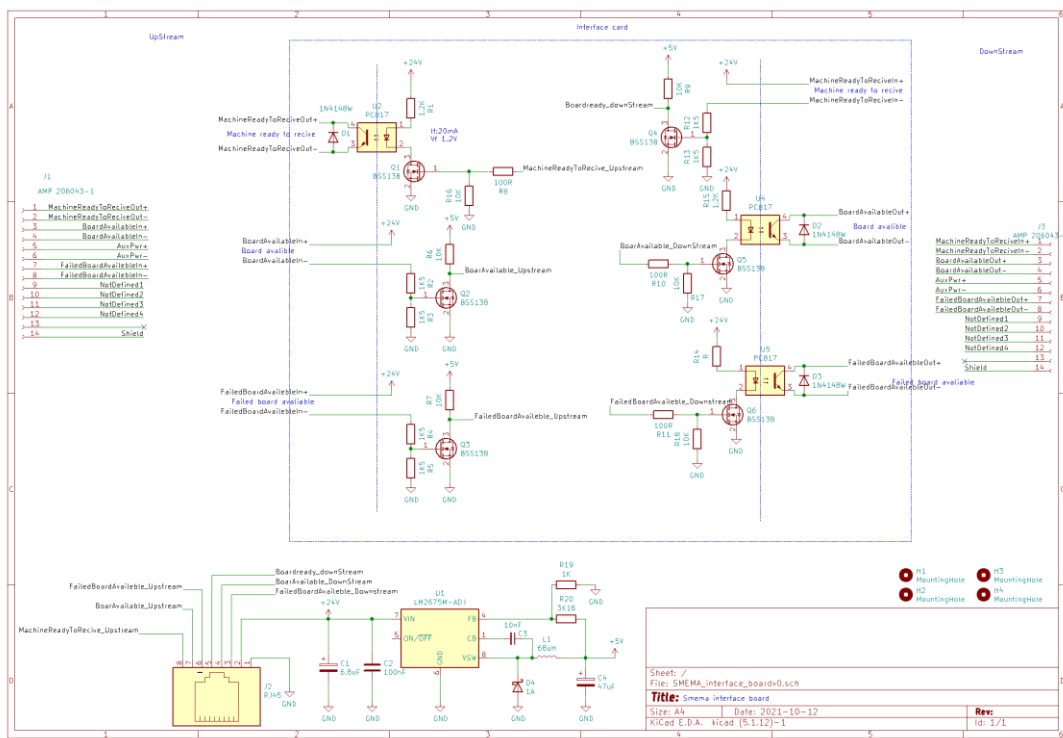


Figure 5.1 Circuit for the interface board.

### 5.1.2 Communication standard used for communicating between the AOI and the soldering machine

Since the Raspberry pi is already part of the AOI, the communication will be achieved by extending the python program to handle the inputs and outputs. The program flow is shown in Figure 5.2.

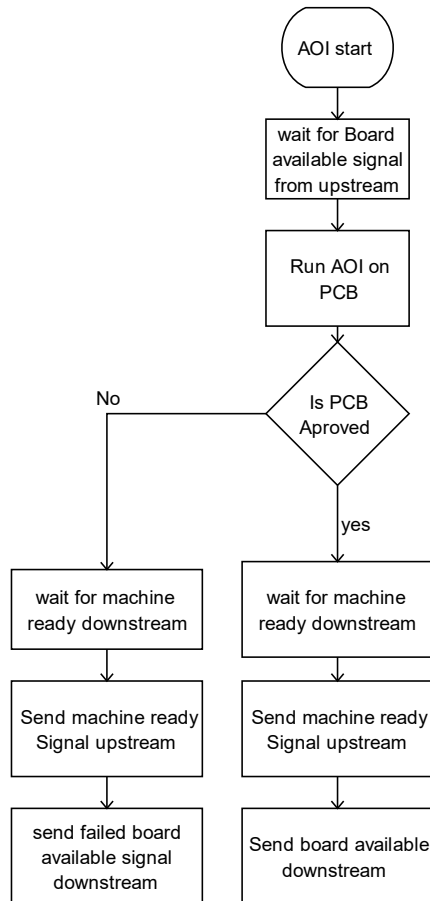


Figure 5.2 flow chart for the SMEMA communication

In the case of an PCB not being approved after the inspection, the AOI will send a failed board available signal to the soldering machine. The conveyor belt will start to send the PCB as soon as the machine ready signal is sent. The transfer of the board may only start once the conveyor belt have received the machine ready signal and the soldering machine have received the Board available signal.

# 6 Testing the SMEMA communication

This chapter focuses on the functionality achieved from the GUI and the SMEMA interface through testing the systems. Note that the tests could not be fully executed due to component shortage at the supplier of the interface board.

## 6.1 SMEMA communication

Due to component shortage at the supplier for the interface board, the supplier could not deliver the board in time for this project. Therefore, the program and the circuit could not be tested. The strategy for testing the communication is described below.

### 6.1.1 Features to be tested

The following features from the SMEMA communication will require testing:

- The voltage levels for all for all connections to the raspberry pi.
- The logic controlling the signals out of the board

The voltages into the board are assumed to be correct as they are created by the Conveyor belt and the soldering machine. Any error that occurs in the SMEMA communication must be considered as a severe fault as it could damage the Raspberry Pi or produce faulty products.

### 6.1.2 Deploying test

The voltages should be tested first. The 24V power for the

To test the SMEMA communication, the interface board would first be tested without being connected to the machines. By manually triggering the board available signal on the upstream side and the machine ready signal on the downstream side while measuring the load over the board available signal on the downstream side, it is possible to test the communication without needing to connect the board to the machines. This is necessary to prevent any damage to the machines or the PCB in case of a faulty circuit. The testing will still require the AOI to be set up as it is necessary to test that the logic for the signals is correct.

## 6.2 GUI

The GUI have been completed as intended, however since the AOI cannot be properly setup without the interface board, the testing must be carried out using special python scripts in place of the AOI, since the AOI cannot be properly set up without the interface board.

## 6.2.1 Features to be tested

The following features from the GUI communication will require testing:

- Main window image updates
- Pass and fail buttons work
- New template function saves image to correct location
- New template function display error when no name is defined
- Setting new template occurrences to 0 is not possible
- New Product function create and saves correct folders.
- New Product function returns error when product already exists
- New Product function starts capture image script
- View templates function shows all templates in folder
- View templates function can delete a user selected template

## 6.2.2 Deploying tests

Since the AOI is not properly set up, the test of the program will be performed using a computer running windows. For the tests that require an image, the GUI program will be executed with a python script that returns a known image. The tests carried out and the results is shown in Table 6.1. Note that the error message described here refers to a small window containing an error message

Table 6.1 Tests executed on the GUI program and results.

<b>Test feature</b>	<b>Result</b>	<b>Description</b>
<b>Main window image updates</b>	Approved	The test is conducted by using a python script that flags an image for inspection in place of the AOI. The test was conducted three times to test that the variables in the GUI resets properly.
<b>Pass and fail buttons work</b>	Approved	The test was completed by supervising the file that the GUI writes the pass/fail input to. The test was run three times to ensure the result was repeatable.
<b>New template function saves image to correct location</b>	Approved	This test was completed by creating a new program in the GUI and placing an image as a placeholder for an image taken by the camera. The template folder was supervised to ensure that the GUI saved the files as expected.
<b>New template function saves image to correct location</b>	Approved	Blank name was entered in the feature. Error message appeared as expected. The GUI returned to the name input window.

<b>Setting new template occurrences to 0 is not possible</b>	Approved	The test was carried out by creating a new template and attempting to set the occurrence value to zero. The input field for occurrences changed the input back to one when this was attempted as expected.
<b>New Product function create and saves correct folders.</b>	Approved	Three new products were created all with correct folders.
<b>New Product function returns error when product already exists</b>	Approved	A new product was created and given the name "PCB", creating a new product with the same name resulted in an error message appearing as expected.
<b>New Product function starts capture image script</b>	Partly Approved	The capture image python script could not be tested at this time. The GUI was tested with a script that returns an image. The test was considered a success as the GUI started the script. The image capture script must be tested once the AOI is properly set up
<b>View templates function shows all templates in folder</b>	Approved	The feature was tested with three different products, all templates were shown in each case
<b>View templates function can delete a user selected template</b>	Approved	The feature was tested by creating three new templates for three products and attempting to delete them with the delete button in the view templates function. When the delete input was made, the templates were deleted as expected.



# 7 Documentation

This chapter will focus on the Documentation created for the GUI and SMEMA communication.

## 7.1 GUI operator manual and SMEMA circuit

An operator's manual has been written and is included in appendix A. This manual is Targeted at the operators at Hapro Electronics and covers Basic operations for how to navigate the GUI. The circuit schematic for the SMEMA interface can be found in Figure 5.1.

## 8 Discussion

This chapter will discuss the results from the GUI and the SMEMA interface board.

### 8.1 The GUI challenges and improvements

Creating the GUI proved a more challenging task than first anticipated for this project. As stated in chapter 4, the features were created in order of perceived complexity. Therefore the “new template” function was created first. In reality the functions of the main page were the most challenging. This was because the GUI and AOI program required a way to interface between each other. A few different solutions were attempted, but most ended with a roadblock due to the limitations of the raspberry pi and the mono platform required to run .exe files on the raspberry pi[13]. In the final solution the communication between the AOI and GUI is done by using a file containing the parameters, which is constantly supervised by the GUI in an individual thread. A future improvement to this solution could be to implement a way for the GUI to access the returned values from the AOI program directly rather than depending on a common file.

Other useful features for future development could be implementing a form of user login. This could improve traceability of the product by having the operator login, as well as making it possible to create different features for administrators and operators of the AOI.

### 8.2 The SMEMA interface board results and improvements

The SMEMA interface board could not be completed due to a lack of components at the supplier. The challenges connected to creating the schematic for the board were usually solved by using resources from Hapro Electronics.

Improvements to the SMEMA board could be to create extension board to raspberry pi instead of having a separate circuit board for the communication. This will free up some wires and make the entire system look more professional. Another improvement could be to add a circuit for powering LED lights, allowing the operator to not be required to always wait for updates in front of the computer screen.

## 9 Conclusion

The goal of the project where to create a user interface to the AOI as well as establishing communication between the AOI and the soldering machine. The user interface has been fully created and documented in the form of a user manual. This manual can be found in the appendix. The creation of the GUI was challenging as it requires a C# program to run and interface with a python script.

The communication between the AOI and soldering machine have been planned. However, it could not be completed due to component shortage. The planned communication interface would have been based on the SMEMA-9851 machine interface standard. This communication standard was selected as this allows satisfactory communication with the soldering machine without modifying the machine in any way, which could void the warranty of the machine.

# 10 References

- [1] **Fuji.** "NXT-II-Brochure." islandsmt.com. <https://islandsmt.com/wp-content/uploads/2017/04/NXT-II-Brochure.pdf> (accessed 10.11, 2021).
- [2] O. foundation. "Unified Architecture." OPC foundation. <https://opcfoundation.org/about/opc-technologies/opc-ua/> (accessed 12.11, 2021).
- [3] G. M. Smith. "What Is EtherCAT Protocol and How Does It Work?" <https://dewesoft.com/>. <https://dewesoft.com/daq/what-is-ethercat-protocol> (accessed 15.11, 2021).
- [4] *IPC-SMEMA-9851*, IPC, ocmmanufacturing.com, February 2007 2007. [Online]. Available: <https://ocmmanufacturing.com/wp-content/uploads/2016/01/IPC-SMEMA-9851.pdf>
- [5] k. erza. "Highlights Selective Soldering System VERSAFLOW 3/66:." kurtzrsa. <https://www.kurtzrsa.com/electronics-production-equipment/soldering-machines/selective-soldering/produkt-details/versaflow-366-2.html> (accessed 28.10, 2021).
- [6] k. ersa, "Ersa Selective Soldering Systems  
In a class of its own!," in "Ersa Selective Soldering Systems," kurtzrsa.com, Datasheet 07.2020 2020. [Online]. Available: [https://www.kurtzrsa.com/members/electronics/9-media/95-catalogues/952-ersa-machines/9524-selective-soldering-systems/95241-interested-client/mbdownload/6616/go.html?no\\_cache=1&cHash=bcf6d1b7c9bfa6bc9c2a588bcec2c96](https://www.kurtzrsa.com/members/electronics/9-media/95-catalogues/952-ersa-machines/9524-selective-soldering-systems/95241-interested-client/mbdownload/6616/go.html?no_cache=1&cHash=bcf6d1b7c9bfa6bc9c2a588bcec2c96)
- [7] S. Campbel. "BASICS OF THE I2C COMMUNICATION PROTOCOL." circuitbasics.com. (accessed 15.11, 2021).
- [8] MIKEGRUSIN. "Serial Peripheral Interface (SPI)." <https://learn.sparkfun.com/tutorials/serial-peripheral-interface-spi/all> (accessed 15.11, 2021).
- [9] H. Øverby. "Bluetooth." snl.no. <https://snl.no/Bluetooth> (accessed 15.11, 2021).
- [10] H. Øverby. "Wi-Fi." snl.no. (accessed 15.11, 2021).
- [11] F. Churchville. "user interface (UI)." TechTarget.com. <https://searcharchitecture.techtarget.com/definition/user-interface-UI> (accessed 14.10, 2021).
- [12] R. Computing. "Introduction." <https://riverbankcomputing.com/software/pyqt/intro> (accessed 28.10, 2021).
- [13] mono-project. "Cross platform, open source .NET framework." <https://www.mono-project.com/> (accessed 28.10, 2021).

## Appendices

Appendix A Operators manual

Appendix B Project description

Appendix A

# **Operators MANUAL**

**For the Automatic optical inspection**

Prepared by: Jon Eilert Liane

November 2021

---

---

## Revision Sheet

Release No.	Date	Revision Description
Rev. 0	17/11/21	Operator's Manual Created

# OPERATOR'S MANUAL

## TABLE OF CONTENTS

Page #

<b>GENERAL INFORMATION .....</b>	<b>i</b>
System Overview .....	i
Acronyms and Abbreviations.....	i
<b>Using the Graphical user interface .....</b>	<b>ii</b>
Starting the AOI and GUI.....	ii
Load a program for a product .....	iii
Create a program for a new product.....	iv
Creating new templates .....	v



# GENERAL INFORMATION

## System Overview

The AOI is an inspection system that relies on a microcontroller in the form of a Raspberry Pi and a camera to inspect PCBAs before components are soldered onto the board. The AOI software utilize a GUI to allow an operator to configure an interact with the AOI.

## Acronyms and Abbreviations

AOI - Automatic Optical Inspection, name of the machine and software that inspects the PCB

GUI - Graphical User Interface. The program that allows the operator to interact with the AOI

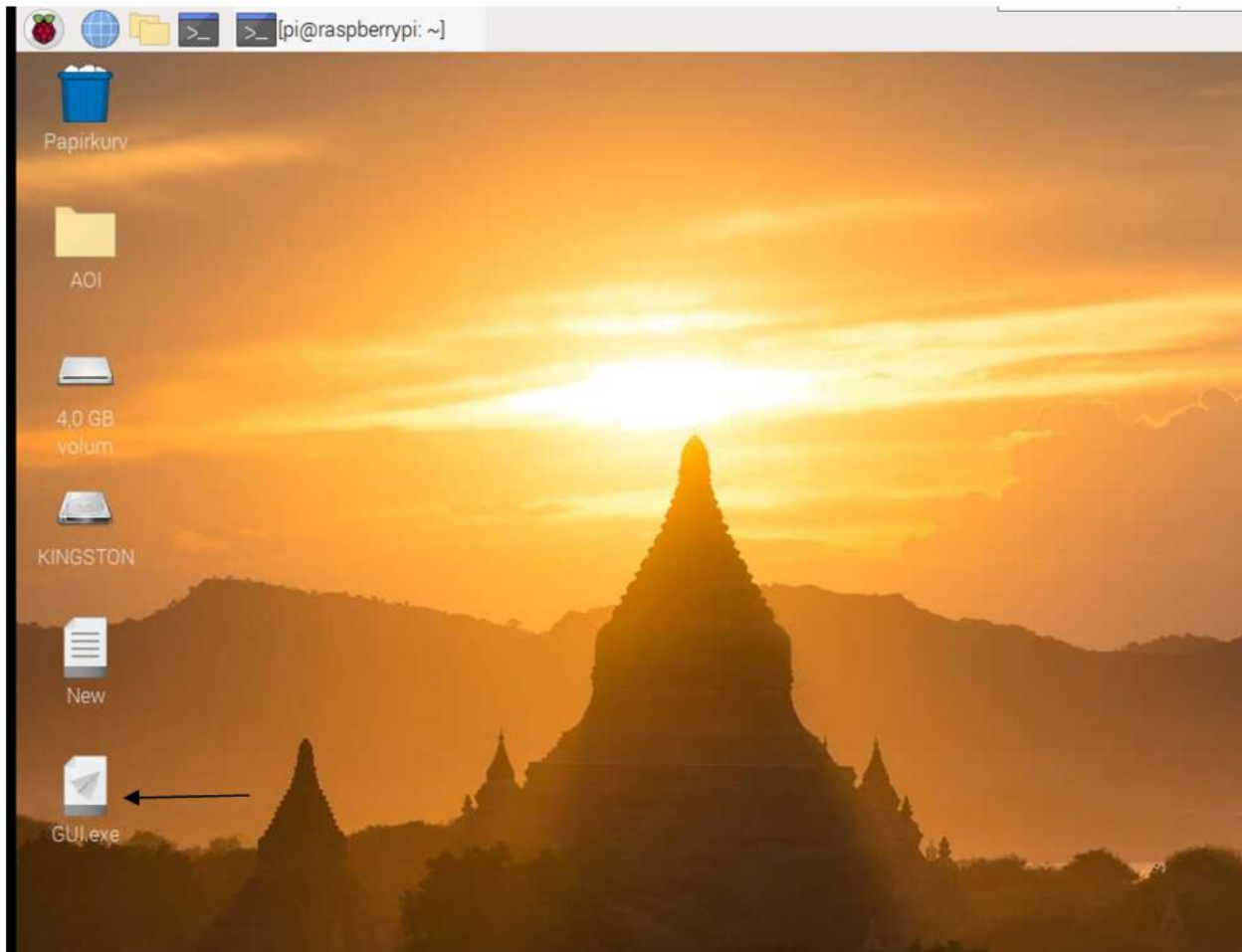
PCB - Printed Circuit Boards

PCBA - Printed Circuit Boards Assembly

# Using the Graphical user interface

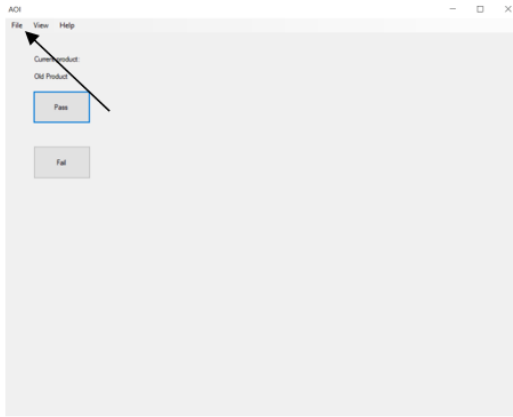
## Starting the AOI and GUI

To start the GUI, double-click on the “GUI.exe” file on the desktop. This will start the graphical interface as well as the AOI program. The main window is displayed when the AOI and the user interface have started.

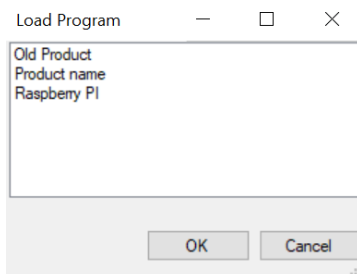


## Load a program for a product

To load a product press “file” and “load product”. This will open a list of all available products.

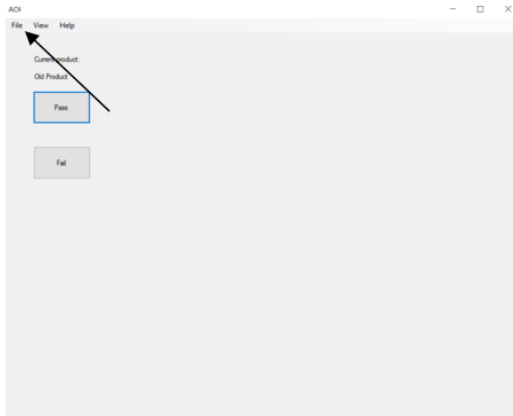


Select the desired product and press ok. The current product can be viewed in the main window of the program.

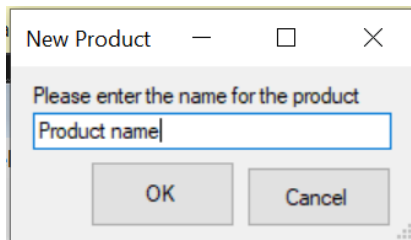


## Create a program for a new product

If a new product is introduced, a new program will have to be created. To do this, press file, new and program.



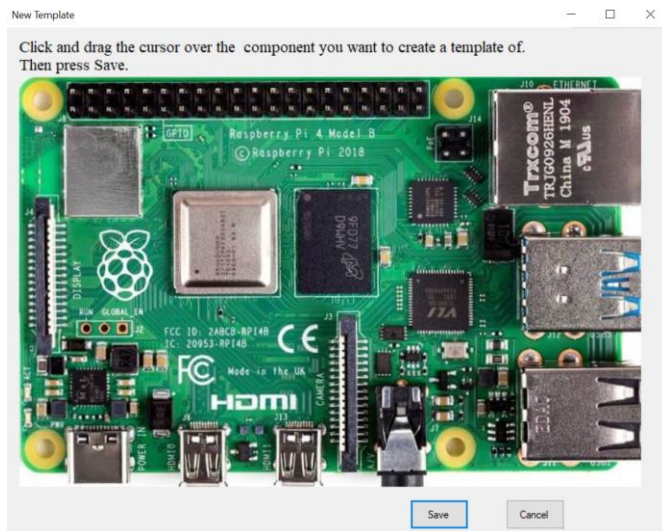
A new window will appear. Type the name of the product into the box and press ok.



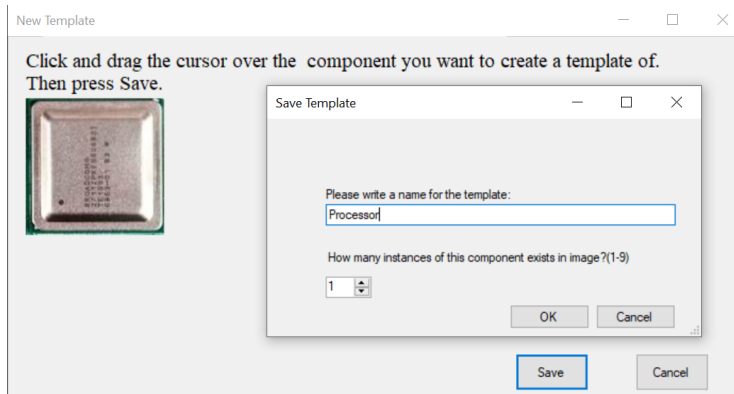
Once the ok button is pressed, a new window appears instructing to place the product under the AOI. When the product is in position press ok. The AOI now takes a picture and creates the folders for the new product. Please note that the new product is automatically loaded into the main window.

## Creating new templates

To create new templates, first load the program that you wish to create templates for, then press file, new and template. A new window appears with the image of the product.



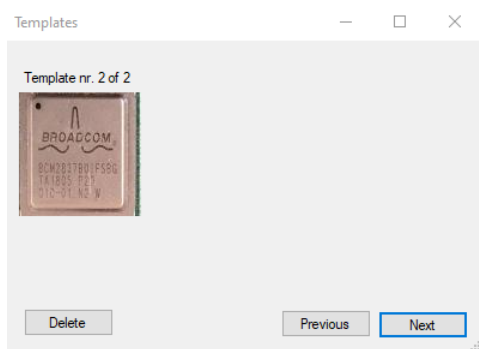
Click and drag the mouse over the area that you wish to create a template for, and press save. A new window appears.



Enter the name and the number of occurrences the template and press ok. The template is now saved to the product.

## Viewing the templates

To view all the templates for the product, first load the desired product, press view and templates. A new window appears. The templates can be viewed by using the buttons on the bottom right of the screen. If a template needs to be deleted, press the delete button on the bottom right. This will remove the selected template from the product.



## Passing or failing an inspection

When the AOI finds a product with one or more faults, the main window will be updated with an image. When this happens, inspect the presented image. If there are faults to the image, press Fail, otherwise press pass. If a product fails, the AOI will not send the product into the soldering machine.



This product needs to be removed. If Pass is selected, the product will be sent into the soldering machine.

# Appendix B

## FM4017 Project

### **Title:**

Microcontroller based Automatic Optical Inspection (AOI) Module for Fault Detection in Printed Circuit Board Assembly (PCBA)

### **USN supervisor:**

Hans-Petter Halvorsen and Saba Mylvaganam

### **External partner:**

Hapro Electronics

### **Task background:**

Hapro Electronics provides Electronic Manufacturing Service (EMS) for the electronics and data industry and produces prototypes and deals with industrialization, automatic and manual assembly, box building, testing.

PCB consists of a network of copper traces and insulators which connect clusters of components based on different of circuit diagrams from customers. Printed Circuit Boards Assembly (PCBA) based on some considerations is done manually, e.g., low-volume production runs, thru-hole components on low-volume runs, custom parts and sub-assemblies attached to a board, “stand-offs” and press-fit connectors. EMS in the manual and automatic PCBA of components is one of the major activities in Hapro Electronics.

To prevent critical “issues” cropping up in PCBs and as a measure of guaranteeing function and quality, PCBA is continuously tested using different techniques including Automatic Optical Inspection (AOI). Soldering machine used in PCBA and the AOI system communicate with each other based on the standard for communication between mechanical equipment, IPC-SMEMA-9851.

### **Task description:**

The main goal of this project is the development of a versatile communication system between the AOI, and the soldering machine based on IPC-SMEMA-9851. At Hapro Electronics, the next generation AOI is under development. The system for AOI will be based on Raspberry Pi and at least one camera. Based on camera vision, different signals are transmitted to the Soldering machine and the operator.

In this project the tasks are:

- Give a list of reasons for manual PCBA and possible issues arising during this process with a good overview of the status of technology.
- Give an overview of the system for PCBA in Hapro Electronics
- Give an overview of different standards like IPC-SMEMA-9851 with a short literature survey of its applications.
- Describe scenarios of camera vision and the image characteristics and features useful in generating alert or alarm signals
- Design a communication system using camera vision, Raspberry Pi, and the Soldering Machine. Discuss different solutions.
- Development of a communication interface with AOI.
- Develop a GUI for the communication system with AOI for alerting the operator of missing components and of new components in PCBA
- Test the complete AOI communication system, with documentation of the plan for testing, and guidelines for executing these tests.
- Organize the codes and procedures in the form of a user manual.
- Submit a report with the details covering all the tasks for this master project following the guidelines of USN.

**Student category:** IIA

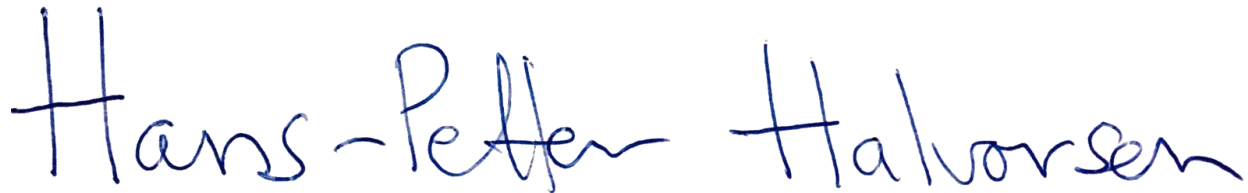
**The task is suitable for students not present at the campus (e.g. online students):** No

**Practical arrangements:** Hapro Electronics will provide necessary support for the candidate. USN, Faculty TNM, Dept. EIK will assist in providing software necessary from the existing software packages.

**Signatures:**

Supervisors (date and signature):

Hans-Petter Halvorsen – 2021.09.27



Students (write clearly in all capitalized letters + date and signature):

Jon Eilert Liane – 2021.10.13



Jan Eilert Liane