Technical Report

Lorenzo Daidone / Team Name

Date

Contents

[List of terms and abbreviations 2](#_Toc200787176)

[Introduction 3](#_Toc200787177)

[Internship work 3](#_Toc200787178)

[1. Technical aspects of sims 6](#_Toc200787179)

[1.1 Power distribution 6](#_Toc200787180)

[1.2 Computers of the Sims 7](#_Toc200787181)

[1.3 Motion & Control Loading 8](#_Toc200787182)

[1.3A Control loading 8](#_Toc200787183)

[1.3B Motion 8](#_Toc200787184)

[1.4 Flight compartment 9](#_Toc200787185)

[1.4A Sound System 9](#_Toc200787186)

[1.4B Visual system 9](#_Toc200787187)

[1.4C Instructor operation station(IOS) 10](#_Toc200787188)

[1.4D Avionics & flight instrument I/O 10](#_Toc200787189)

[2. Digitalization of QTG tests 11](#_Toc200787190)

[2.1 Research Questions 11](#_Toc200787191)

[2.2 Problem overview 12](#_Toc200787192)

[2.2A Recommendation/ solutions 12](#_Toc200787193)

[2.2B Scope of solution 12](#_Toc200787194)

[2.2C Risks 13](#_Toc200787195)

[2.2D Planning / execution 13](#_Toc200787196)

[2.2E Solution and validity 13](#_Toc200787197)

[3. Software 14](#_Toc200787198)

[3.1 Libraires (PI\_PDF\_PRINTER) 14](#_Toc200787199)

[3.2 Drivers for operating system: 15](#_Toc200787200)

[3.3 Operating system environment 16](#_Toc200787201)

[3.3A SCRIPT\_PATHS 16](#_Toc200787202)

[3.3B CRITICAL\_DIRECTORY\_PATHS 16](#_Toc200787203)

[3.4 PI\_PDF\_PRINTER 17](#_Toc200787204)

[3.4A How to use the software: 17](#_Toc200787205)

[4. Hardware 19](#_Toc200787206)

[4.1 Design & manufacturing introduction 20](#_Toc200787207)

[4.1A Parts list 20](#_Toc200787208)

[4.2 Design and parameters for manufacturing 21](#_Toc200787209)

[4.3 Technical Drawings 22](#_Toc200787210)

[4.3AFront Top Plate 22](#_Toc200787211)

[4.3BFront bottom plate 23](#_Toc200787212)

[4.3CMid plate 23](#_Toc200787213)

[4.3DBack plate 24](#_Toc200787214)

[4.4 Assembly 25](#_Toc200787215)

[4.4A Screw Path 25](#_Toc200787216)

[4.4B M3 nut placement 26](#_Toc200787217)

[5. Results 26](#_Toc200787218)

[5.1 Operation Testing 26](#_Toc200787219)

[5.2 Working process Testing: 27](#_Toc200787220)

[5.3 observed issue 27](#_Toc200787221)

[5.4 QTG digital Results 28](#_Toc200787222)

[Improvements 31](#_Toc200787223)

## 

## Foreword

I am very grateful to the engineering team of CAE for giving me this opportunity, they have showed a admirable amount of interest in helping me progress as a engineer and understand the sims and the workspace. It has ben a pleasure for me to work here with knowledgeable people on such complex machines.

My special thanks to:

**Jordan O’Donoghue**

Jordan as my internship supervisor which I’m very glad about, he’s given me a exceptional amount of knowledge on the sims, tips to integrate better, knowledge about commercial aviation and a better understanding of many avionics in commercial jets. Another very nice aspect of having Jordan as a supervisor is how social and charismatic he is which makes it easier for us interns to feel confident and motivated in a new workspace.

**Ronald Baden**

Ronald Baden has ben great help through shifts which we worked together, he’s taken the time to explain how the configuration files are set up and how they make changes to them when there’s a problem or updates, he’s always happy to help and takes the time to answer as many question clearly. Like Jordan he’s very charismatic and is great fun to work with and have good talks in our coffee breaks.

**Rob Glastra**

Rob has ben a great coworker to have, he’s a very skilled electrical engineer, is capable of solving very tough problems and is also very good at teaching how to approach the sims to find any faults safely. In the beginning I didn’t want to open any aircraft parts due to my fear of ruining it but Rob would put the extra effort to find issues and let me fix it and this increased my confidence a lot.

**Rene Kleindorst**

Rene has ben very helpful in terms of my project, whenever I had a change I made sure to contact him, he’s always given me any extra info needed, like when I was given to make procedures for QTGx I was introduced to everything by him and was also approved to work on the 747 QTG digitalization instead by him.

**Ijsbrand Kindt**

Ijsbrand is a very knowledgeable person whom has ben in the sim industry since the early development of the field, his knowledge in terms of regulations, electrical system, computing and mechanical systems is exceptional and has a very clear way of translating how they work to new workers here. In the beginning of my internship he explained a great deal of critical information regarding sims.

# Introduction

CAE Amsterdam B.V is a pilot training centre focused on commercial aviation. Its located at the Fokker logistics park at Schiphol, the centre serves as a European hub for pilot training which has 18 full motion flight sims in service. This technical report consists of information about the working environment for interns located in CAE Amsterdam B.V. , a brief explanation of some technical aspects of the sims, how their systems interact and my assignment for my internship project.

## Internship work

Working on solving issues and maintaining the sims are the primary goal of the technicians at CAE including interns, communication between the customer is also crucial since we need to address any issue to them and understand any issue addressed to us by the customer. This means that were required to have sufficient knowledge in in aviation to understand exactly what the problem they are describing is such as a too high ILS path or a specific problem with a FMC, we also need to have sufficient knowledge in engineering to be able to pinpoint where in the system may the problem come from.

Checking-out customers from sessions, generally we check out the customers whenever they finish with the sims, if there is a problem they can report it to us at the desk at checkout, this makes it easier for us to know what deficiencies the sims may have and allows us to report and fix it as soon as possible.

Fixing deficiencies, we log deficiencies on a software called SIMEX which is the main database for the time customers use sims, the open deficiencies and maintenance task, with this we can keep track of who did what when working on the sims. When a deficiency is fixed we put it on offer and a MPC can close it or keep it on monitor to see if it raises again.

Reloading and changeouts, when customers change some airlines use different databases and aircraft configurations so we reload it for a specific customer like changing from KLM to Cordon we need to load different software, we also need to change the cockpit configuration depending on the engine they are using like the A320 use Neo or CEO so we must change the circuit breaker panel and a few other parts of the cockpit.

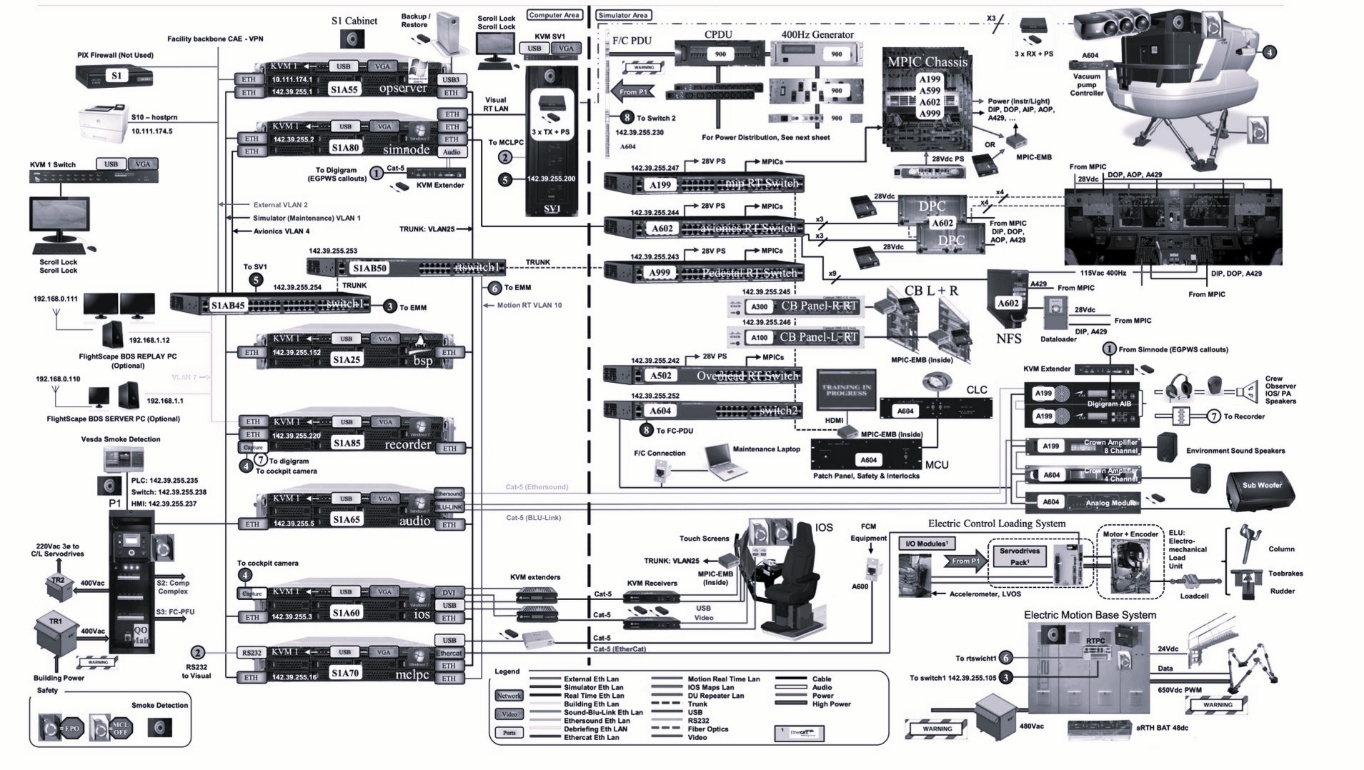
Interns have also a specified task apart of working as a technician on the sims, it can range from finding measurement solutions, creating procedures to even creating a entire product like I did, its also up to the intern to understand their set of skills and scope to pick the right project.

## List of abbreviations

|  |  |
| --- | --- |
| Abbreviation | Meaning |
| AC | Alternate current |
| A/C | Aircraft |
| AOG | Aircraft on Ground |
| CAE | Canadian Aviation Electronics |
| C/L | Control Loading |
| Db | Decibel |
| DC | Direct current |
| DOF | Degrees of freedom |
| EASA | European Aeronautical space Agency |
| EMO | Emergency off |
| EPO | Emergency power off |
| FAA | Federal Aviation Administration |
| FFS | Full-flight Simulator |
| IG | Image generator |
| IOS | Instructor operation station |
| I/O | Input output (DIGITAL) |
| LAN | Local area network |
| MCL | Motion & Control loading |
| MPC | Maintenance planner coordinator |
| OOT | Out of tolerance |
| PDU | Power delivery unit |
| PWM | Pulse Width Modulation |
| PLC | Programmable Logic Controller |
| QTG | Qualitative test guide |
| TD | Test-drive utility (QTG utility) |

# Technical aspects of sims

In this chapter the general working philosophy of how all the systems and how interact with each other on the sims, I will use the CAE 7000XR as a example since it has the best documentation and also the 747 since its the oldest, understanding these 2 gives a better idea of how the sims have also evolved since all the other sims lay between these 2 in terms of technology.



## 1.1 Power distribution

A essential part of the FFS is the power distribution and load management. The P1 cabinet is the main power control cabinet powered by the facility power grid to manage all power for a single sim, this makes sure that the power is distributed along all subsystems such as the PDU (power distribution unit) and additional sub power supplies used for visuals, motion, computers etc. P1 handles:

* 400Hz 115-120v AC current used to meet aircraft current requirements
* AC DPU distributes (400Y/230V, 60-120HZ) to other systems such as for visuals, motion and control loading which are made for standard 3 phase ac current.
* DC power supplies are also installed to provide potential difference of 28v and current load may very depending on the sim.

The P1 per simulator varies depending on the sim being used but they all have a really similar systems, what differs them mostly has to do with aspects such as how is the power management monitored. The newer sims use PLC’s (Programmable Logic Controller) which constantly monitors which systems have power and which ones don’t and can shut down the cabinet if a critical component isn’t powered. Older sims use PCB’s specifically designed for the purpose of checking power for a specific set of power supplies, a example is the 747, it uses a array of optocouplers which are connected in parallel to a set of power supplies critical, the collector and emitters are closing the gate to a signal line which connects to a mosfet base pin, if all power supplies are working then the bus line to the base of the mosfet goes high allowing current to flow between the collector and the emitter which flips up a bit on a array which specifies on a table of the Host pc that that set of power supplies are working and can be safely operated. The older sims are more complex to understand but can be elegant in my opinion since you can see the physics and logic in the computing and not get lost in endless layers of software abstraction and overly powered components for the job. (just to have a idea of the difference in how its all computed)

## 1.2 Computers of the Sims

The computer system is a collection of computer nodes located in a computer room, these computer control every aspect if the FFS system such as the simulation software, motion, control loading, sound, visuals and other systems. Computer nodes in the computer cabinet provide computing power and memory capacity for the sim, these node are loaded with commercial operating software some which are specified for the customers such as nav data or airport renderings. These computers interact with specific comm protocols which are designed to work with LAN(ethernet), RS232(USB) and optic Fiber.

**Compute nodes located in the S1 Cabinets(for XR7000)**

* **Opserver** – main computer (host), contains all simulation software which run on other nodes, doesn’t run simulations but sets the configurations.
* **SimNode** – the sim node runs real-time simulation of all FFS systems like I/O ‘s.
* **Audio** – generates and simulates all the audio related to the sims.
* **IOS** – generates the user graphical interface for in the sim, used by the instructor.
* **MCLpc** – controls the motion, control loading and the crew access way.
* **VisualPC** – generates real time visuals on the sim.

The 747 is slightly different since it uses older systems it doesn’t use sim nodes

* Opserver

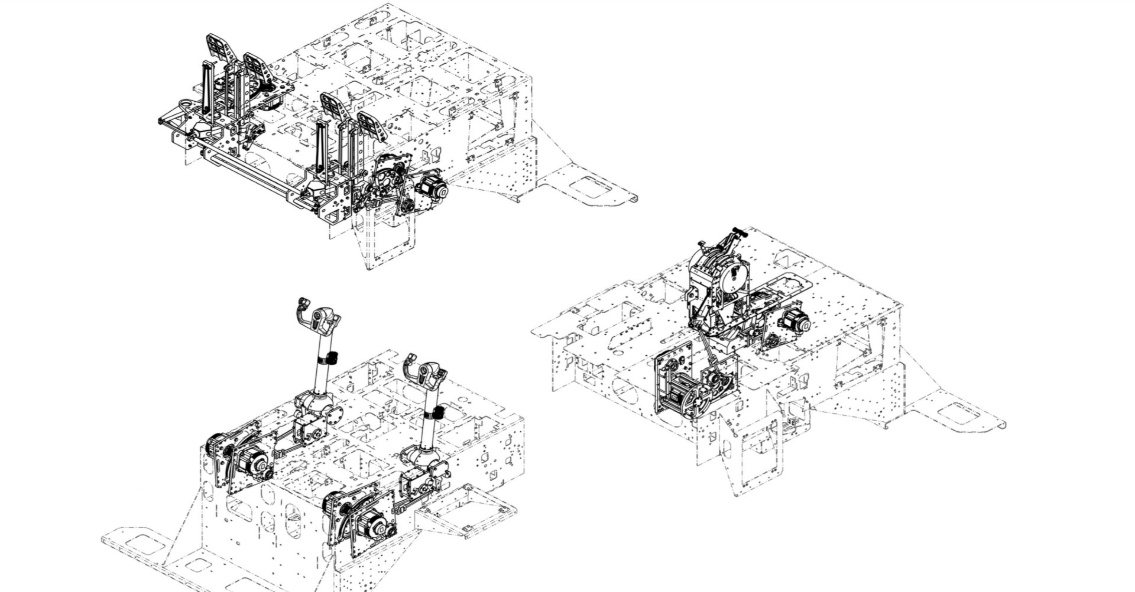


## 1.3 Motion & Control Loading

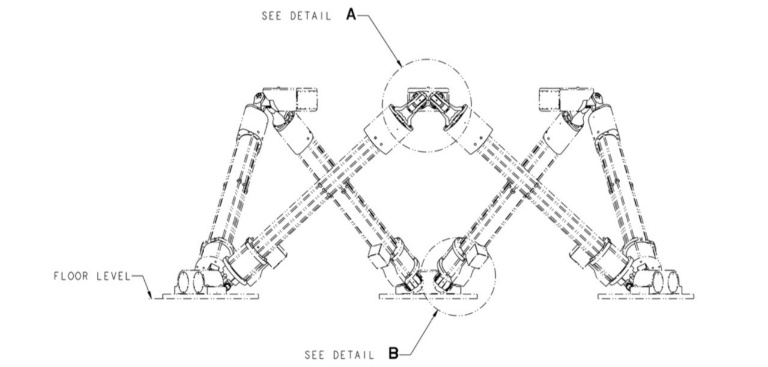
The MCLPC is divided into 2 parts the motion and the control loading, the control loading provides control surface feedback, autothrottle and autopilot.

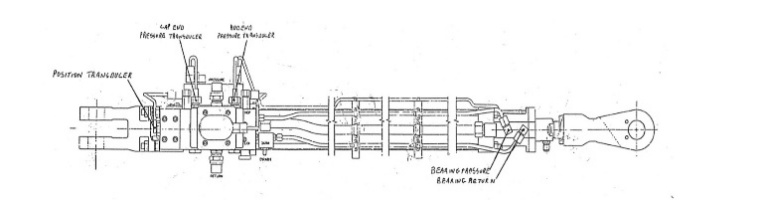
### 1.3A Control loading

* Dynamic load unit – responsible for monitoring pilot inputs and providing a forced feedback.
* Static load units – provide a predetermined feel force on their respective flight controls.

****

### 1.3B Motion

****The motion PC controls the drivers for the servo motors which actuate the legs of the Sim, the newer sim (7000XR) work using lead screw actuators and the older sims(747) use hydraulics. The main difference is the hardware used but they all fall in a very similar field when it comes to functionality. Most of these systems are modulated using PWM which is a standard for most many control systems.

****

## Flight compartment

The flight compartment is where the final output of the systems which make up the sims are, internally it consist of the visuals, sound systems, Instructor station, avionics and flight I/O for newer sims, older sim I/O are in cabinets in the compute room like with the 747 simulator. The systems explained below and figures above are for the XR7000.

### 1.4A Sound System

The sound systems of the sims simulates the acoustical environment of the real aircraft. It does this by using a multi-channel type sound amplifier, these sound systems receive the input from the sound computer, these systems very and can be quite complex, to make sure they’re are accurate QTG tests are done to be able to measure acoustic accuracy compared to real aircraft.

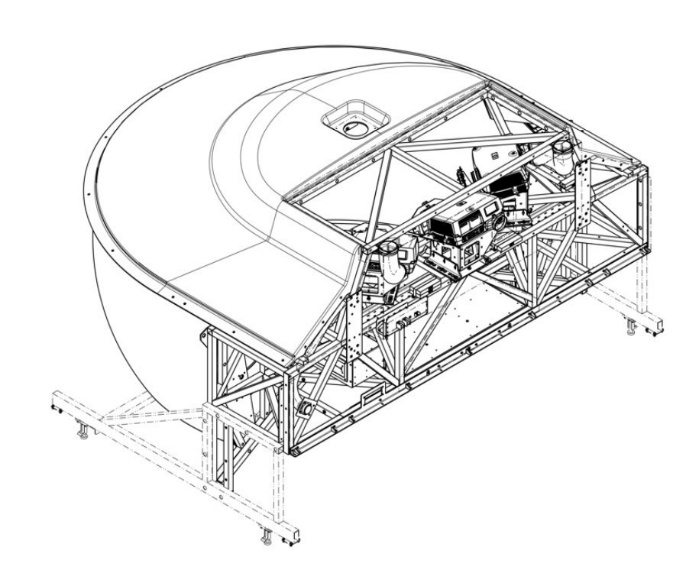
### 1.4B Visual system

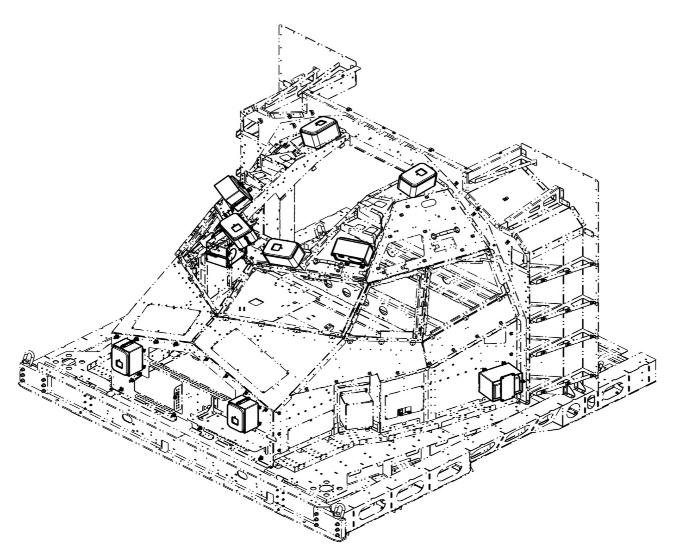
The purpose of the visual system is to generate computer images providing visual cues. These scenes simulate real time visual scenes for any weather and time of day. Customers have the option to use generic airports or custom airports made by CAE. These visuals are compromised of 2 parts, one which is in the SV1 cabinet which is the visual computer and the other is the actual visual system (projector) in the sim.

The main visual systems consist of a set of projectors mounted at a fixed location on the support structure, they consist of lenses and other electrical equipment. These projectors project to a spherically-radiused black projection screen, this screen is made by setting a aluminized polyester film which is vacuumed on one side to create a perfectly mirror dome. From the cockpit view a 3d rendered visual is presented when the systems are working.

**SV1:**

* Visual control station – controls the visual aspect if the simulation it self
* DB world server – stores all the database for the visual control station
* Image generators (IG) – converts the signal from the visual control unit to the respective projector.

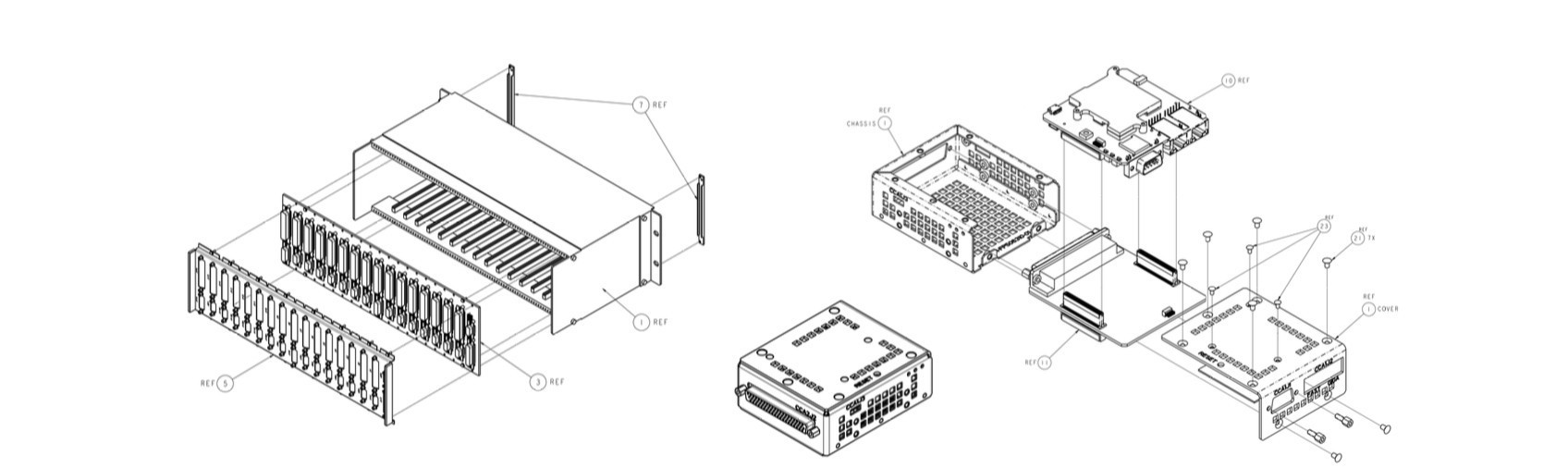




## 1.4C Instructor operation station(IOS)

The instructor operation station allows the instructor to control the simulator and monitor the crew’s performance during a session. Here is where the instructor can set scenarios such as weather, location, errors, malfunctions and is also where the instructor can act as ATC service for the pilots training. The system consist of a IOS node, a instructor seat, a observer seat and a audio interface.

## 1.4D Avionics & flight instrument I/O

****The interface system within the simulator is responsible for communication between the interface computer and the aircraft instrumentation . the system consist of I/O cards or computers depending on the sims, the 7000XR series use Multipurpose interface cards (MPIC) which are cards with preprogrammed microcontrollers to buffer data from a ethernet bus into multiple outputs for a specific avionics, some simulators use a small computer running embedded windows, I personally think using a microcontroller is more elegant since its not asynchronous meaning less can go bad but each sim is currently using the best tech at the time of it being made. Older sims like the 747 use compute modules with preprogrammed eproom’s, similar to the newer ones using the MPIC but using much older equipment which takes up cabinets of PCB boards.

# Digitalization of QTG tests

**Background information**

CAE Is a company which provides D-level flight simulator services for airline companies D-level is the highest standard for full flight sims, they have requirements which must be complied to be able to offer these services, this is set by EASA and the FAA depending on location and companies which are training. Here in CAE Schiphol EASA is the main qualifier for CAE. There’s a set of rules and issues and set of test to be able to do QTG test and can be found on the (CS-FSTD(A)) on the EASA website.

**Background information for my assignment.**

Working on QTGX to implement digital QTG’s from sims which haven’t digitalized it yet due to outdated systems. These QTG’s are logged on paper which is time consuming, they will be transferred to a digital system which allows it to be done with less labour and easier to track through a software called QTGX. The QTG test must be submitted in a format which allow efficient review and evaluation before the FSTD can gain qualification level. Where applicable, the QTG should be based on the aircraft validation data as defined by the operational suitability data (OSD) established in accordance With Part-21. So based on the requirements from EASA there’s a set of rules for procedures and compliances which need to be done so the QTGs can be validated, on the digital system this isn’t possible for many sims since they are older And the workflow isn’t well implemented yet so my job is to get the procedures to work for the QTGX and for sims which are too old to implement this to find a method of having them to convert the files and send them to the database in a reliable and expandable manner.

## 2.1 Research Questions

1. **How can XQTG be used with the existing sims to give a more reliable log history for newer and older sims?**

* **how do sims get the information into the server?**

Most modern sims use a small computer which retrieves the QTG data and stores it on a server.

* **What different operating systems do the sims use?**

Most of the sims use windows and Linux some older sims use IBM AIX.

* **Which sims can’t log the data into the server?**

The sims using IBM AIX and the L3HARRIS sims

1. **What causes the most issues in terms of documentation of QTG test on sims?**

* **how does this compare between new and old sims?**

most issues in documentation have to do with the workflow and the constant change of software used, there’s many different software for storing QTG’s and sorting them before the are reviewed and migrating from one system to another may be hard since not all sims are capable of implementing newer systems.

* **Is it more related to hardware limitations or workspace procedures?**

Its related to both, for the sims them self it has to do with the hardware. Its impossible to even plug a USB storage device into some sims so to have it match with newer systems extra hardware or software is needed. Another problem is migrating the engineers to the new system, new methods are often overlooked since it requires energy and time to learn new methods.

* **How does it affect the workspace of the engineers?**

can save a lot of labour time and increases the quality of the results since they are all fully digital and not scanned from prints.

* **What methods of can be used to implement this work flow on all sims regarding their older age?**

Creating a new set of hardware to digitalize the QTG’s and create a procedure on how to use it with the new software.

## 2.2 Problem overview

CAE have ben doing QTG tests on all sims and have for most of the companies history logged and done all QTG tests on paper, now for more modern sims the test are saved into a pdf file and stored in a database, there’s new programs being developed and the methods used are constantly changing meaning procedures are changing and older sims can’t be logged digitally without scanning a printed document, creating a larger load in the workspace.

Some of the older sims from CAE like the IBM RS6000 running IBM AIX which is the IBM Unix operating system, this PC has ben used from the late 80’s till early 2000’s for super computers with high processing power. The main issue is the PC has no simple way of saving PDF files, has no USB output. This means that older machines which are still within their working lifespan are very outdated to todays tech so either need to be modified or updated.

In Amsterdam CAE centre only one sim uses this PC (the 747-400) but they also are used in other sim centres for Fokker and 737 sims. So making a method that could work and easily added on multiple sims would be best the and most practical option for the company.

The QTG test consist of:

* Performance
* Handling qualities
* Motion system
* Visual system
* Sound system

For the sims using the IBM rs6000 the QTG’s are printed on a hp printer, the only QTG’s which aren’t are the hardware QTG’s so what has to do with control loading and the motion these are printed using a spectrum analyser and a plotter but has ben updated to be logged by a Pico scope, a oscilloscope and spectrum analyser which uses ETH and USB input to a windows computer connected to the IBM.

Its important that the method being implemented may help the engineers and their workload meaning it has to be easy to use, not require too many steps in the procedure and that it may be extremely reliable so that whenever its needed it may be ready and working but also with redundancy, such as having backups which can be swapped out in matter of minutes.

### 2.2A Recommendation/ solutions

There’s mainly 2 solutions to solve the issue with sims which only print QTG’s on paper, firstly the most obvious is changing or adding software to the sim host PC, this is the most elegant solution in general because it doesn’t add any hardware which takes space and cost money. The second solution would be to add a device which can act as a printer and buffer the serial data into a pdf format, this solution might cost more in terms of hardware but seems to be the one of least risk since it doesn’t involve modifying any current working systems. So out of the 2 solutions I decided to go for the second one since I already have hardware available for developing, for the first solution which is changing the software on the host pc its a bit tricky since its all done in c and changing any C files on a operating system can lead to crashes or corrupted data.

### 2.2B Scope of solution

In terms of my general knowledge Its enough to solve the solution I make a lot of stuff so generally its not much work for me, the main reason I chose to make a whole device is because then it can be applied to more than just one sim meaning once the design is completed as many can be made to be added to other sims and also for risk management, if something gets messed up it will not affect any working equipment.

### 2.2C Risks

For risks the main risk is messing up the system which is already working, firstly I can rule out the risk of messing up the host PC since I’m not changing any software on it, the risk fall more in messing up the way the server is configured since I’m adding a printer emulator to the server.

The second risk is the risk of creating a back door for hackers, in these types of industries hackers tend to attack often since the computers used are very expensive so its easy to put a lot at stake for the company, the computer I use for Printing is a Linux PC and it uses a few network hacking tools to trick the sim. This can cause serous issues if the ethernet is connected to the simulator and Wi-Fi to a online service, this can give unwanted access to hackers so its important to make sure that it cant be connected to the internet while connected to the sim network.

The biggest risk of all is that it messes up the current configuration and causes a halt on all current QTG’s being made on the sim, that can lead to problems with EASA, so every step I take I must make sure that I can always revert to the previous working configurations no matter what.

### 2.2D Planning / execution

Boundaries: As previously mentioned in risks, its important that I make it in such a way that it has little to no risk to the working system and that it wont create a backdoor to hackers which poses a threat to CAE.

**For this I made a set of rules**

**safety rules:**

1. No software on the sim must be changed.
2. No device connected to the server can have internet access.
3. Can only write for readback verification, all other data is only for reading.
4. Can be reverted by any technician in less than 1 minute.

**Functionality rules:**

1. Has one working interface (GUI) for all needed functionalities.
2. Can be used by any engineer in the department easily.
3. Has fast and easy backups.
4. Complete at least 100 print files with no fail (repeated prints)

**Compliance rules:**

1. File format needs to be the same as in the post script print file for EASA
2. Naming must be the same as the document header
3. Reruns can be renamed and labelled to be implemented to QTGX

This solution works perfectly as long as it complies with the rules mentioned above otherwise it needs to be modified, mostly software in terms of hardware its more than reliable enough the hardware I found available. Once I’ve checked off these set of rules I can validate it as a feasible product. Then continue to work on testing where I let technician’s use it and report feedback before end of MAY.

### 2.2E Solution and validity

My solution is to use a single board computer with a 10 inch monitor to create a printer emulator which acts as a network printer but instead of printing paper it prints a pdf file into a specific directory, once printed it needs to be able to be viewed and not modified only the naming of the test file can be changed.

For this I use a raspberry pi and a 10 inch HDMI input monitor and design a frame to put them together into a single device. The device cannot be too big and must fit easily in any computer room next to the printer being used to print the QTG’s. the frame can be 3d printed any additional electrical hardware such as PCB’s for power distribution or network switch I can also make at home or order to be made through PCB-way.

# Software

In this section the interaction of the software used is described for a better understanding of what is going on. Here I will describe the tools used to get the printer working and how I interact these tools to make a workflow for the operation of the software on the device.

1. **Cups**, is a opensource environment used to create print servers, cups stands for (common Unix printing system) which is the main type of operating system used today, like apple and Linux. Cups is mainly used for print servers to send a print job to a USB printer via WIFI or ethernet.
2. **Net-cat**, often abbreviated as NC, is a versatile networking utility that can read and write data across network connection using the TCP or UDP protocols. its used for a variety of network tasks including, testing and debugging, port scanning, file transfers, port forwarding or making a simple web server.

Main working function of net-cat here is:

* **Listen** on a specific port and wait for a connection.
* **Connect** to another machine on a specified port.

1. **PI PDF PRINTER** (PYTHON APP)this is what interacts the other tools to turn this into a virtual pdf printer, this program only opens up a shell with 5 options (start listener, system reset, Open PDF, Save PDF to, Clear data)

system reset à just stops start and resets cups.

open PDF à will open any pdf to the shell so one can check the content.

save pdf à saves the pdf so that It can be renamed and saved.

Clear data à This clears the data on the file where cups drops PDF’s.

SET\_IP/PORT à here you can set the IP and port which the PC’s printer is configured.

## 3.1 Libraires (PI\_PDF\_PRINTER)

These are the libs used for the program. They are used to make a GUI, and access other lower level tools such as cups, NC and XRANDR.

**PYSIDE6**

PySide6 is similar to QT but doesn’t require to buy any software license, its well updated and rather easy to use, the reason I’ve gone with pyside6 is because it uses the same environment as QT meaning its compatible with most renderers like XRANDR which make it easier to implement a cursor when opening a GUI in a PC which main environment is in the command line

.

**SUBPROCESS**

Subprocess allows for python to execute terminal commands, since most of the tools are mainly developed to be used in a CLI environment I decided to use subprocess to access their functionality through a GUI.

**SHUTIL**

SHUTIL I use for file management, this library allows you to control management of files using python and interacted on a higher level. I decided to use this instead of using subprocess with commands like (rm, cp, mv) since the use of subprocess stops the listener which effects the flow of the software functionality when editing files.

**TEMPFILE**

This lib allows a file to be made and viewed and deleted when not viewed, this is used to open and view the PDF.

## 3.2 Drivers for operating system:

As previously mentioned this computers operating system is running on the minimum to call it a operating system kind of.. a operating system involves hardware, software and a kernel to manage the hardware, memory and processes. For this a boot loader is needed to load the kernel into the memory so it starts at execution, kernel image which once loaded by the bootloader initializes hardware, mounts root file systems and starts the first processes which contains data the kernel needs to access the hardware and user space init processes which start services and launches shells. On the CAE\_PDF\_PRINTER all of this is completed already by flashing PI OS lite on the SD card of the PI’s local storage using Raspberry PI’s boot loader once this is all running its just files and common Linux terminal commands to navigate.

Now once these basics are set then its time for the drivers which manage graphics and USB devices and storage devices, PI OS already comes with a driver for keyboard.

**graphics:**

* libqt6pdfwidgets6
* libqt6pdf6

**window and interface**

* Xserver (window manager)
* libgl1 (GLX interface)

**USB Devices:**

* Udisk
* Udevil
* Pmount

These are all the driver used to create a graphical interface, make a cursor, have a USB storage device and have it managed in the systems file system. They are all well documented and widely used today in many computers if not most personal computers.

In most cases these are used for much more versatile computers, to give a good example of how its all set up, you can use a desktop environment Linux distro like ubuntu which uses some of these drivers to run the main desktop environment where you can see your files and run other applications but in the case of the CAE\_PDF\_PRINTER the desktop environment is the python program so to say, it wouldn’t be a very wise way to launch a common desktop environment, they are mostly programmed in more efficient ways by using C or C++ but in this case there’s more than enough computing power to run a single python script with 5 functionalities but can be laggy if it runs with a desktop environment taking ram memory also which is limited to 1gb of ram. The reason I chose this method was a trade-off, which was based on what is easier to execute, a app which runs on desktop environment efficiently or a python app which is the only application running but less efficient. I chose to do it in python, even though I’m much more familiar with C and generally find it the best language to exist but in this case I would have had to write over 1000 lines of code and have to deal with insane amount of function calls and pointer arithmetic’s to manage the graphics.

## 3.3 Operating system environment

For this project, since I’ve used a single board computer to get it running, the operating system used is also a very important aspect of the project, so I decided to go with Linux, specifically Raspbian lite which is a Debian based version of Linux with no drivers nor graphical interface installed, this gives me the freedom to choose what driver I’m using and how I’m going to use it.

I chose this for 2 reasons, first reason being processing power, I’m quire a fanatic of optimizing systems to run on less processing and electrical power so having a desktop environment for this project isn’t ideal in my opinion. Another reason is safety, having a desktop environment gives much more freedom to the people using it so what I’ve done is have everything working on the desktop first then migrate it to a CLI environment and got rid of all the back end files used to connect to the internet so that it has no possible connection to a wireless network and also created restrictions in case a technician is used to using Linux. The main downside to this is for me, every modification since the translation is tricky, I use no IDE so I program using nano a GNU text editor and if the iteration of the code is big I do it on GitHub upload it to a external drive and move it to my specified directory using the MV Linux command and run it on command line.

Here I will briefly explain how I set up the operating system to run the PI\_PDF\_PRINTER and how the directories are organized, I will also go over the drivers I use to get the graphics working and the USB drive. For to run the PI\_PDF\_PRINTER there is a few additional scrips which need to be configured to have it working properly, the first ones are the scrips used to run NC listener. Those being rawprint.sh and rawprint\_server.sh

Rawprint\_server.sh is a shell script which listens to a specific port the device IP address, the rawprint.sh is what receives the file and sends it to cups, then cups sends it to the PDF file directory. Its important to know where these directories are it gives a better idea of what does what.

### 3.3A SCRIPT\_PATHS

PI\_PDF\_PRINTER\_PATH /home/printer/CAE/myenv/ PI\_PDF\_PRINTER.py

RAWPRINT\_SERVER\_PATH /usr/local/bin/rawprint\_server.sh

RAWPRINT\_PATH /usr/local/bin/rawprint.sh

There’s many Directories involved in keeping a program running, from all the back end to the front end. Here I will only be mentioning the ones I’ve added which are used in the app it self, such as which directory gets opened when a button is pressed and the location of environments.

**Myenv** Firstly the virtual environment is where all dependencies for python are installed, this file is kept in side a directory named CAE, this is where most of the files related to the GUI is located such as libraries and python scripts.

**ANONYMOUS**, this is a file which is generated when installing CUPS, this is the driver used for network printer, its main purpose is to receive a pdf-file to send to a network printer, I use it to receive and contain PDF files, as mentioned before NC sends the Job to cups and cups converts it and stores it in anonymous now instead of printing out on a printer I keep it there, keep in mind cups is a printer driver not a pdf converter but along the process it can be extracted before being reprinted and where its extracted from is the ANONYMOUS directory.

**Media**, Media like in any other PC its where the media or storage/external devices are presented like in any normal operating system with a desktop interface. Since I’ve configured the driver for this computer I decided to keep this directory in a standard path like other computers for the sake of keeping things aligned to how most things are on the world.

## 3.3B CRITICAL\_DIRECTORY\_PATHS

myenv /home/printer/CAE/myenv/

ANONYMOUS /home/printer/var/spool/cups-pdf/ANONYMOUS/

Media /home/printer/media/

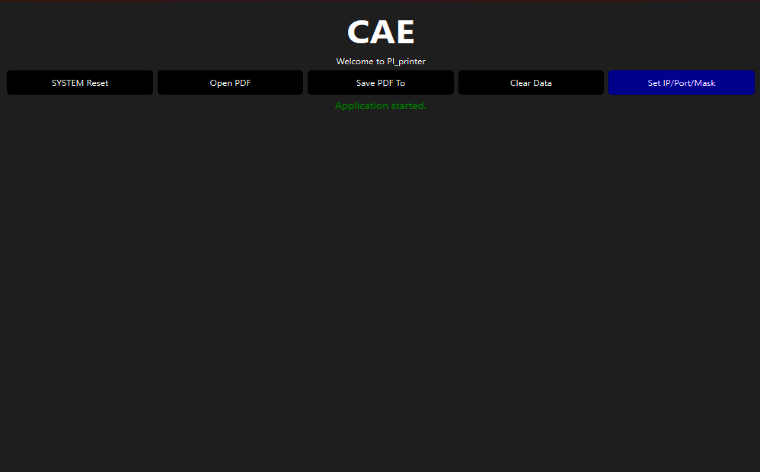
# 3.4 PI\_PDF\_PRINTER

PI\_PDF\_PRINTER is a program developed to make the ease of use for cups and netcat. This program creates a GUI which allows all the needed utilities to interact with the other tools without needing to navigate directories and constantly typing commands into the terminal. This allows less error when using the CAE\_PDF\_PRINTER and allows anyone to use it with little prior information or instruction.

### 3.4A How to use the software:

The main page on the printer has only 5 options and is used for the entire functionality of the printer it self, those being:

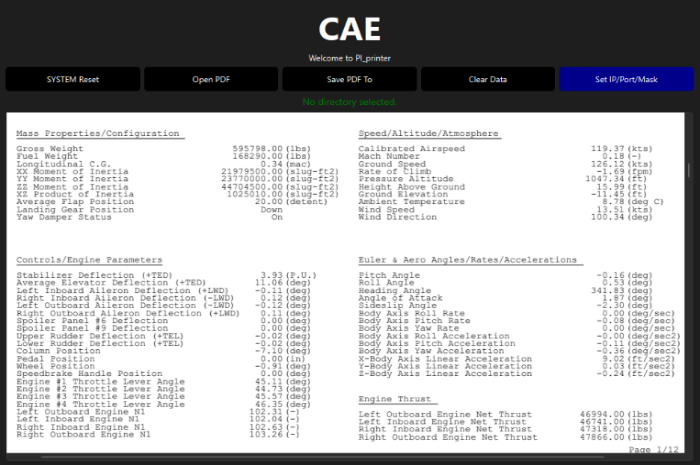
* **System reset**
* **Open PDF**
* **Save PDF To**
* **Clear Data**
* **Set IP/Port/Mask**

SYSTEM Reset

This button does 3 things in the background:

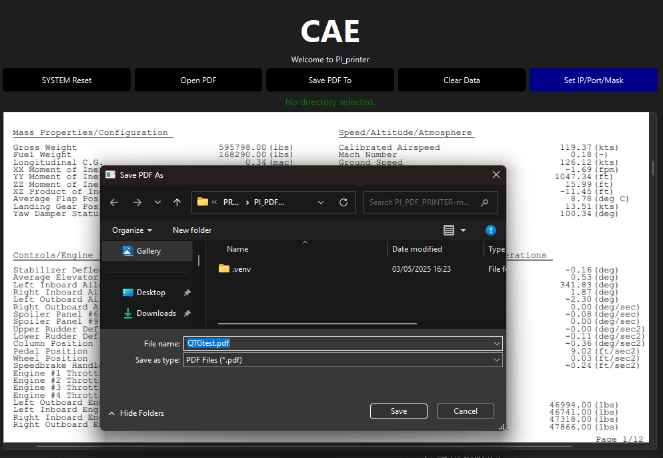
It runs multiple shell commands using a python module called subprocess, when its pressed it stops CUPS , Starts Cups and Resets CUPS in one go. It does it all just to make sure the process it completed individually. Its the equivalent of entering on the Linux **Terminal:**

* sudo systemctl stop cups
* sudo systemctl start cups
* sudo systemctl restart cups



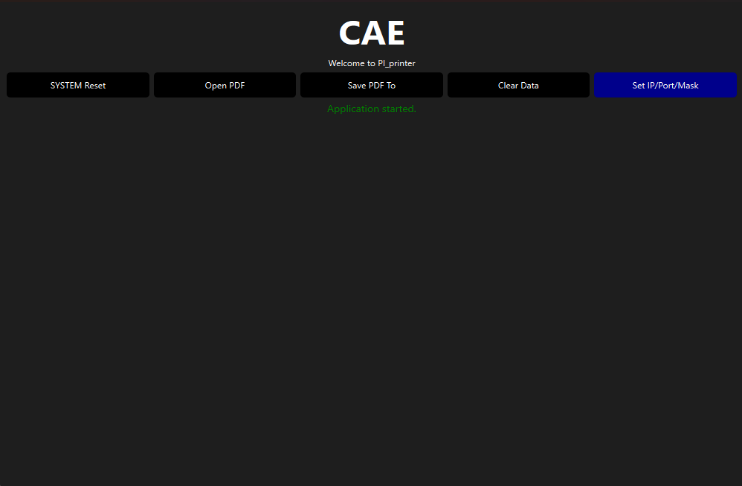
Open PDF

This specifically opens a directory with a file manager located where cups converts the postscript data into a pdf and saves the files once its selected it is opened so the content of the file can be reviewed before saving. As can be seen in the picture on the left.

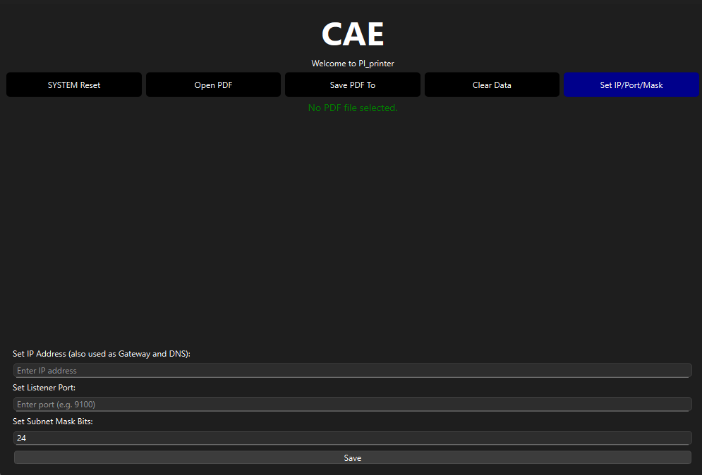
 SAVE PDF

This button allows you to select where you want the file to be saved. The file is opened inside the Media directory so that it can be directly saved onto a external USB storage device.

here is also where the file can be renamed to the actual file name since the PDF converter renames it. Its quite easy and is done exactly the same way as when you save and name a file.

Clear Data

This button only does one thing which is clear all the data within the file which the PDF files are sent to by CUPS, this is intended to be used after the batch of tests are completed and saved onto the external device. This allows a more organized set of QTG tests being done meaning that it wont be full of previous test from the last batch and so that the next batch of tests doesn’t get mixed up with the current ones.

Set IP/PORT/Mask

this button is to only be used once when setting up the printer for use on a specific sim. It sets a IP, Mask and port. For most the mask shall be set to 32 as in most printers and the port and IP is to be configured the same as network printer which is already connected to the server, this can be found either by listing the devices connected to the server on the host pc or by checking the printer settings on the host PC and entering the same ones into these inputs. Once save is pressed the listener from NC stops subprocess runs shell scrips to set the new IP and port and mask, afterwards NC starts again.

Its important to keep in mind if the computer isn’t connected to the ethernet it will give a error but wont cause any issues and will continue to work with the new settings. If the pc is connected to the ethernet it doesn’t give any errors and will display that its connected on the new configuration successfully.

# Hardware



This section contains information which is relevant to the Hardware used to make the CAE PDF printer. The structural parts are designed using CAD software and then manufactured using a 3d printer, other common parts such as LCD and Raspberry pi are sourced from a third party.

The working principle of the hardware used in this printer is to be able to run the software required for the CAE PI Printer and be ergonomically suited for the purpose of viewing and extracting digital files from rs6000 or any pc which has ethernet port access.

**Main points:**

1. Ergonomic for its function:

* The purpose of this printer is to turn a .ps from a external computer file to a .pdf file once the process is completed it should be easy to see the file and to sort it anywhere meaning easy access to eth port, having a screen, single power input.

1. Minimalist design:

* Having a minimalist design allows a easier assembly and tend to be less prone to failure.

## 4.1 Design & manufacturing introduction

Here I will explain:

* 1. Design process
  2. Components/ parts list
  3. Designs and parameters for manufacturing
  4. Assembly

For the design I started off by measuring the components I had which I found feasible for this project then just went straight to cad with a idea in mind then printed it and along the way I kept changing it depending on what I saw could be better, this is called iterative design process which can be quite time saving with less complex designs. Then for the actually making the frame of the PDF printer I used PLA printed on the bamboo labs A1 printer. This allows me to be able to make prints and fix any mistakes or iterate previous designs for a better final product. I’ve gone through about 4 designs and am quite satisfied with the cleanness of the final one. Once all components are printed and all parts in stock I assemble it, I made sure the assembly is very simple and uses the least amount of parts possible, the reason being to save in costs but also because its generally more elegant to use the minimum required parts a product needs.

For manufacturing the device there’s a few components to mention which aren’t built by me specifically but are main components of the PDF printer it self, those being:

1. Raspberry pi 4B
2. 10 inch LCD display
3. M3 screws
4. USB power supply
5. HDMI converter cables

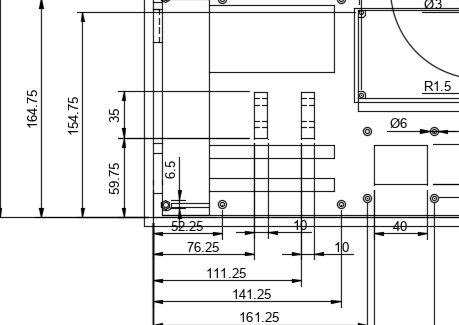
### 4.1A Parts list

|  |  |  |  |
| --- | --- | --- | --- |
| P/N | Components: | Dimensions (mm) | Quantity (n) |
|  | (Structural) | (L x W x H) |  |
| 1 | Front top plate | 247 x 16 x 8 | 1 |
| 2 | Front bottom plate | 247 x 16 x 8 | 1 |
| 3 | Mid plate |  | 1 |
| 4 | Back plate |  | 1 |
| 5 | M3 screw PI | 25 x 3 | 4 |
| 7 | M3 screw struct | 5.0 x 3 | 4 |
| 8 | M3 nut | 6.5 x 3 | 8 |
|  | (Electrical) |  |  |
| 7 | RPI 4B 2GB RAM |  | 1 |
| 8 | 10.1 LCD |  | 1 |
| 9 | HDMI | 30 x 3 x 0 |  |
| 10 | HDMI to HDMI micro | 30 x 3 x 0 | 1 |
| 11 | USBC |  |  |
| 12 | Power supply | 70 x 70 | 1 |

### 4.2 Design and parameters for manufacturing

Here you can find details on parameters such as lengths and distances between features and the parameters for screws and other parts.

Its important to keep in mind some parameters which are fitted into each other are using dimensions which are different from the final piece due to the method of manufacturing. The difference is caused due to thermal expansion of the materials when printed, this goes for screw holes and nut inserts, holes which use no nut are kept at 3mm so that when it contracts it gives material for the screw to thread in.

**example:**

If you look at the pointer in the drawing there’s a bolt insert which is labelled 6.5mm which is a insert for a m3 bolt which is 6mm from across the longest points, once printed the bolt fits just right. To find the right value I made a few calculations which are based on the parameters the values of a come from.

This method is usually used to find the elongation of a string or a cylinder in a single direction, generally for a 3d object it’s a little bit different approach than how its done with CFD but its good enough for this case since getting it perfect down to 0.01mm isn’t possible on 3d printed parts made with a FDM printer.

**EQUATION**:

(L)

|  |  |  |
| --- | --- | --- |
| parameter | Description | Value |
| L0 | Length in design for tool path | (6.5\*10^-3) m |
| a | Thermal linear expansion coefficient PLA | (3.8\*10^-4) µm C |
| T1 | Print Temp | 220 C |
| T2 | Temperature of part at room temp | 20 C |

**Inserting values to the stated equation gives;**

This method is quite good for 3d prints, generally because its not exact but that not exact helps. In these cases often, the reason its wrong is because of geometry, the equation listed is a linear 1 dimensional elongation, example is lets say we make a looped circle using a string and measure how the diameter changes as the string contracts and expands with heat, you come to realize that this stated above equation will give how much the circumference has changed not the diameter meaning the diameter change will be circumference change/ PI. For a hexagon it would be the 4xcircumference/12 and for this piece it would be relative to the pieces centre of mass which gets very complex for non geometric shapes but generally the method showed here tends to give values larger than the actual shape change but still works to fit things quite tight.

## 4.3 Technical Drawings

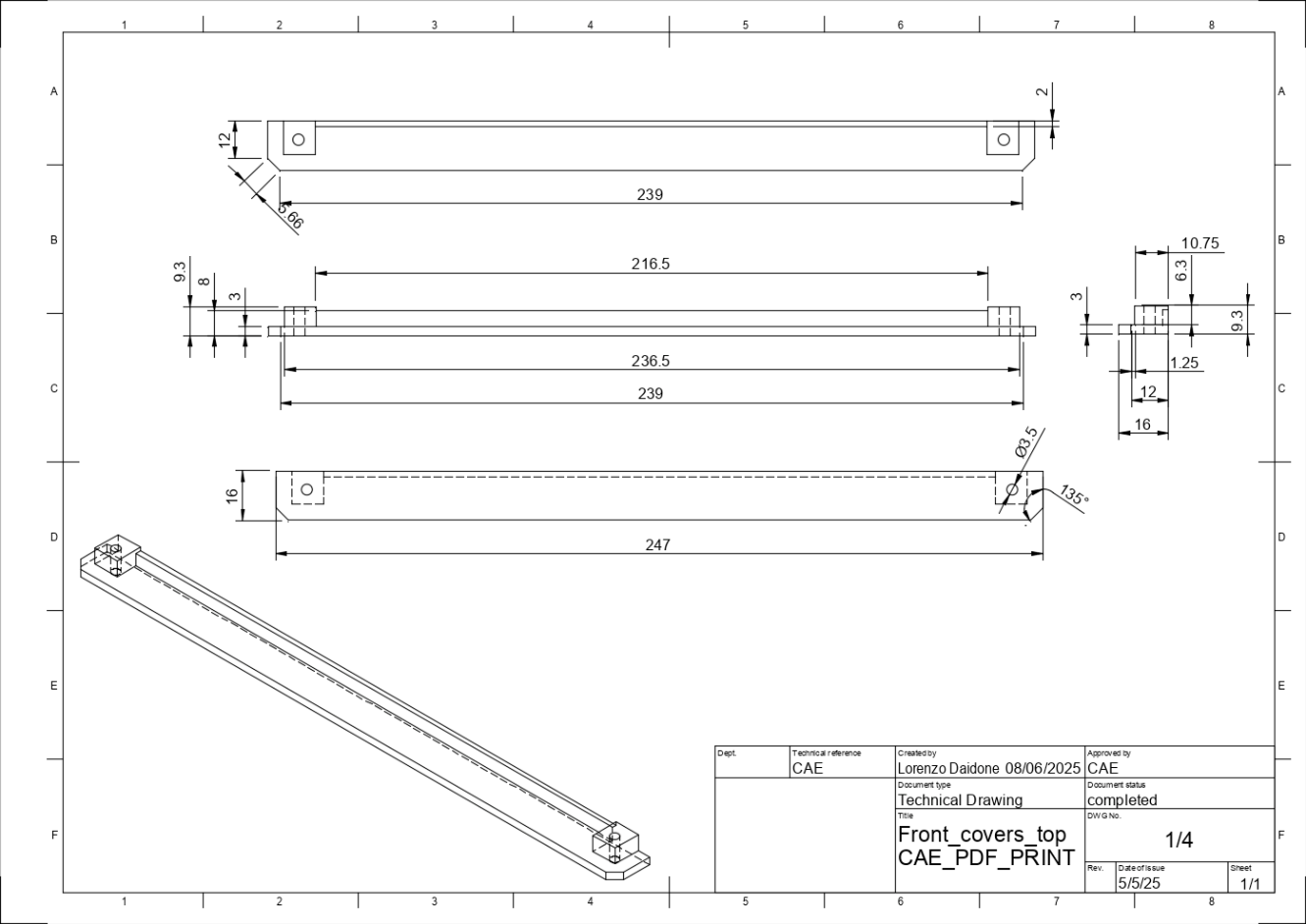
Here you can see the technical drawings of the parts which are 3D printed, these parts are all made with PLA with simplicity being the main factor in the design. Since there’s 4 total parts each labelled on the drawing.

**Those parts being**

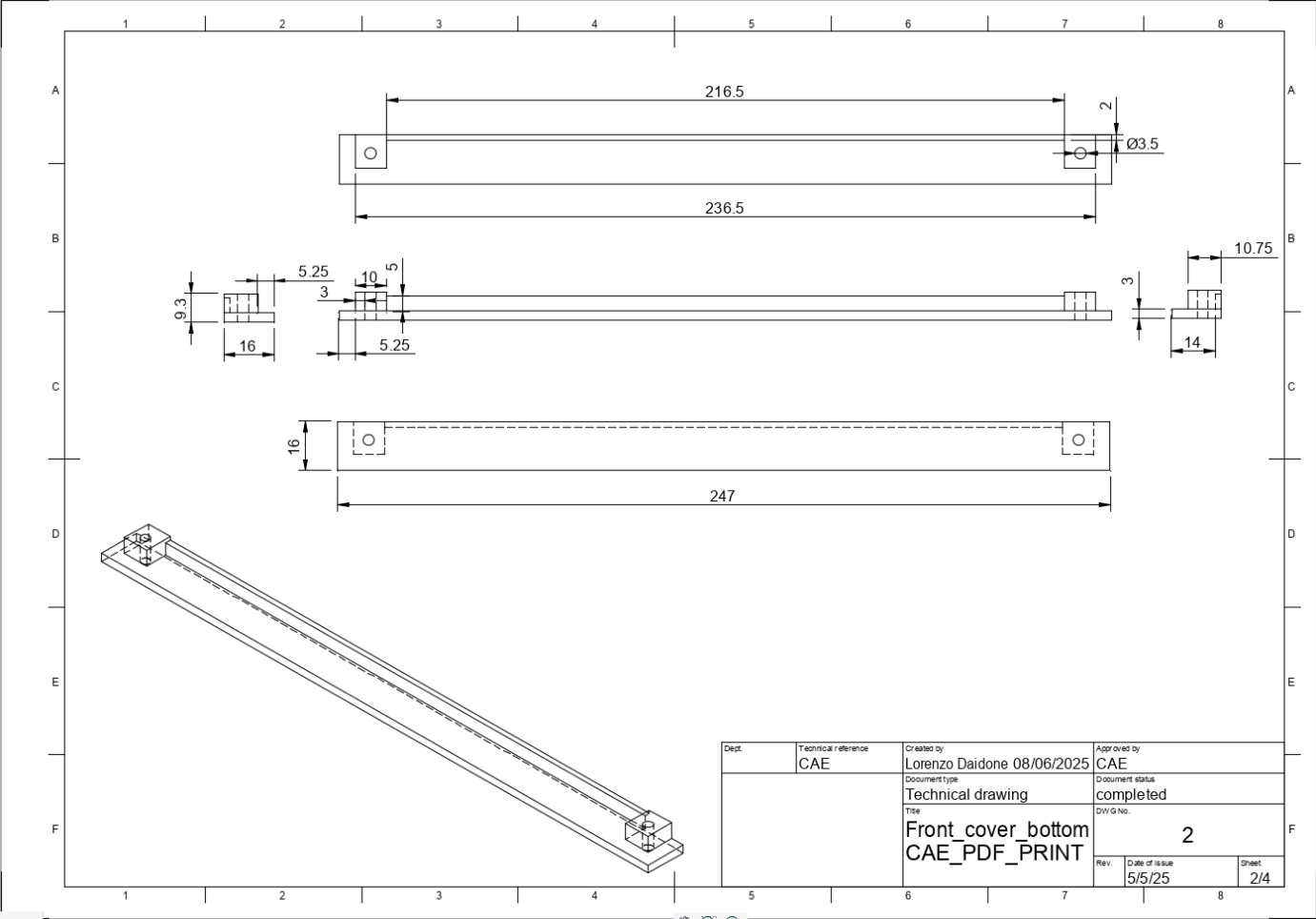
1. Front Top plate
2. Front Bottom plate
3. Mid plate
4. Back Plate

These drawings can be found in the Technical drawing folder for better details and also in the hardware datasheet for the CAE\_PDF\_PRINTER.

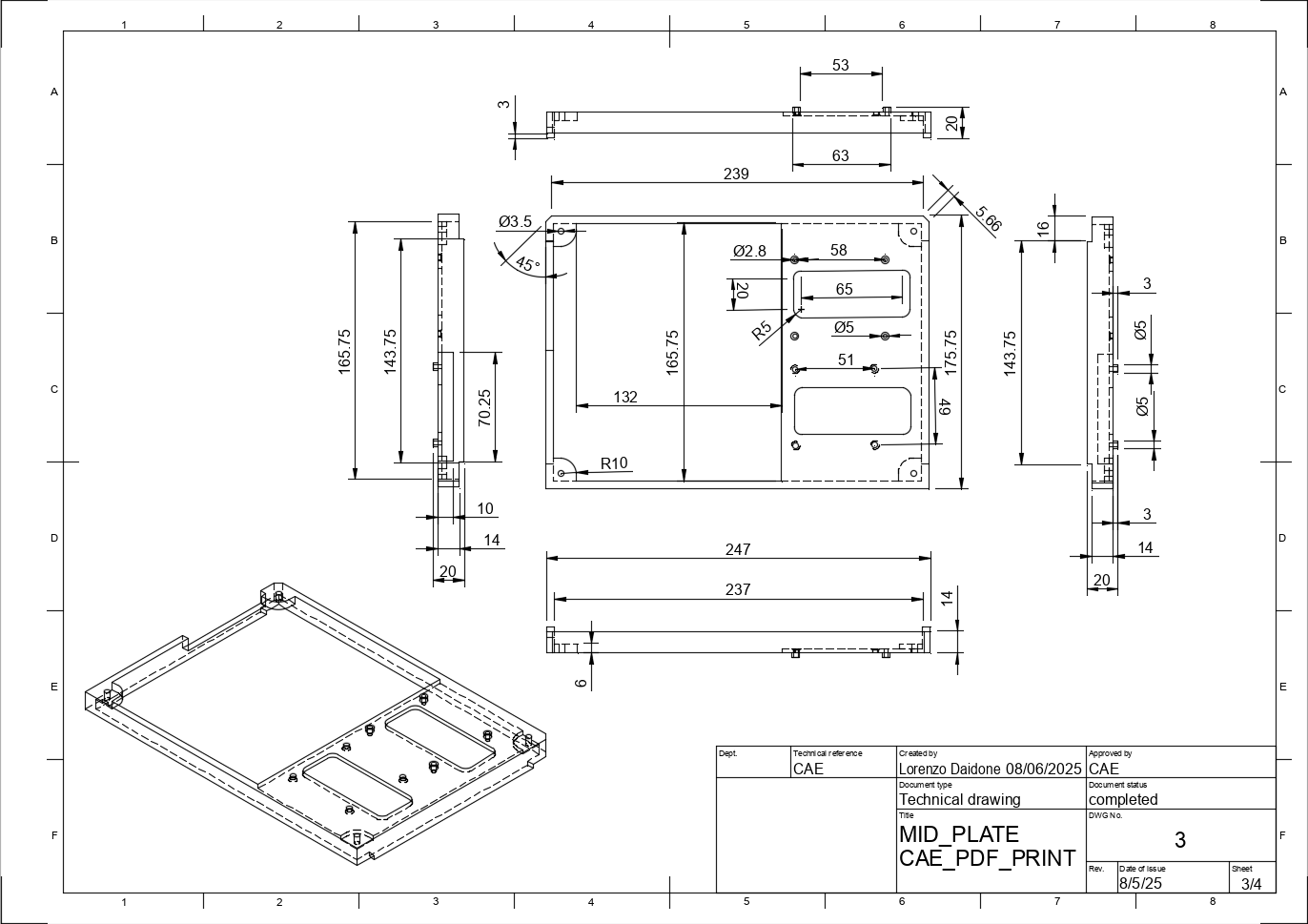
### 4.3AFront Top Plate

****

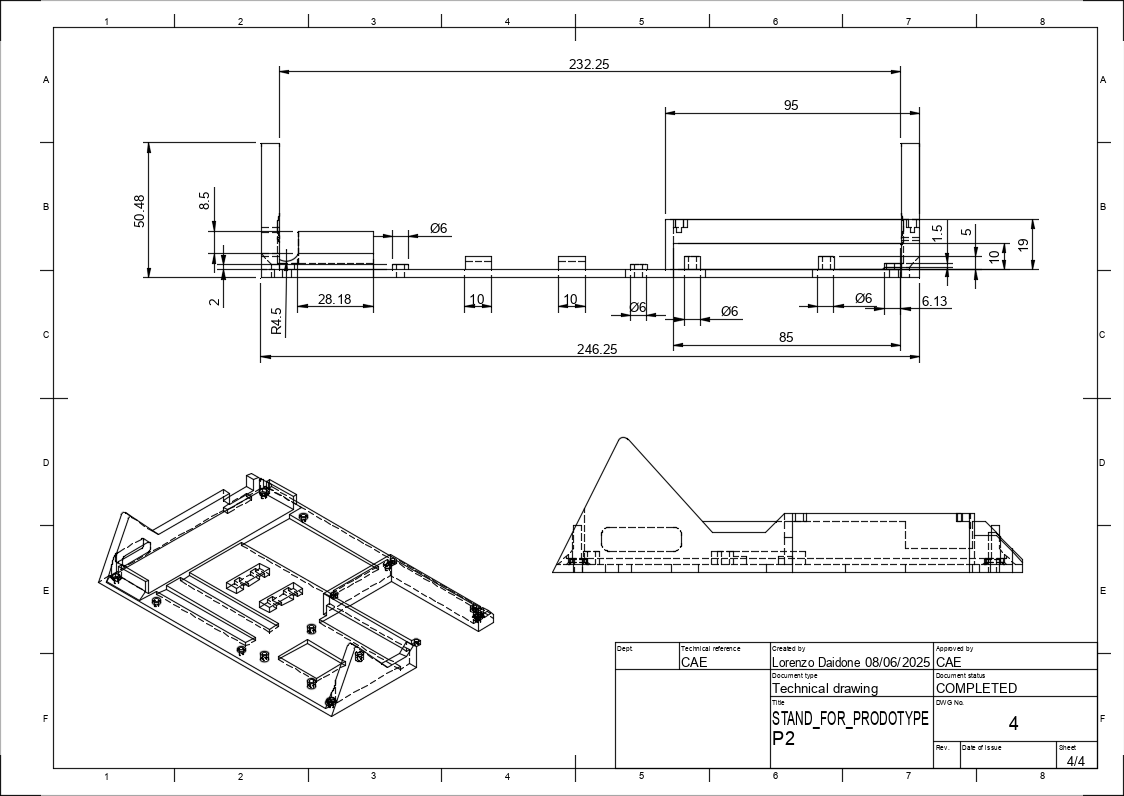
### 4.3BFront bottom plate

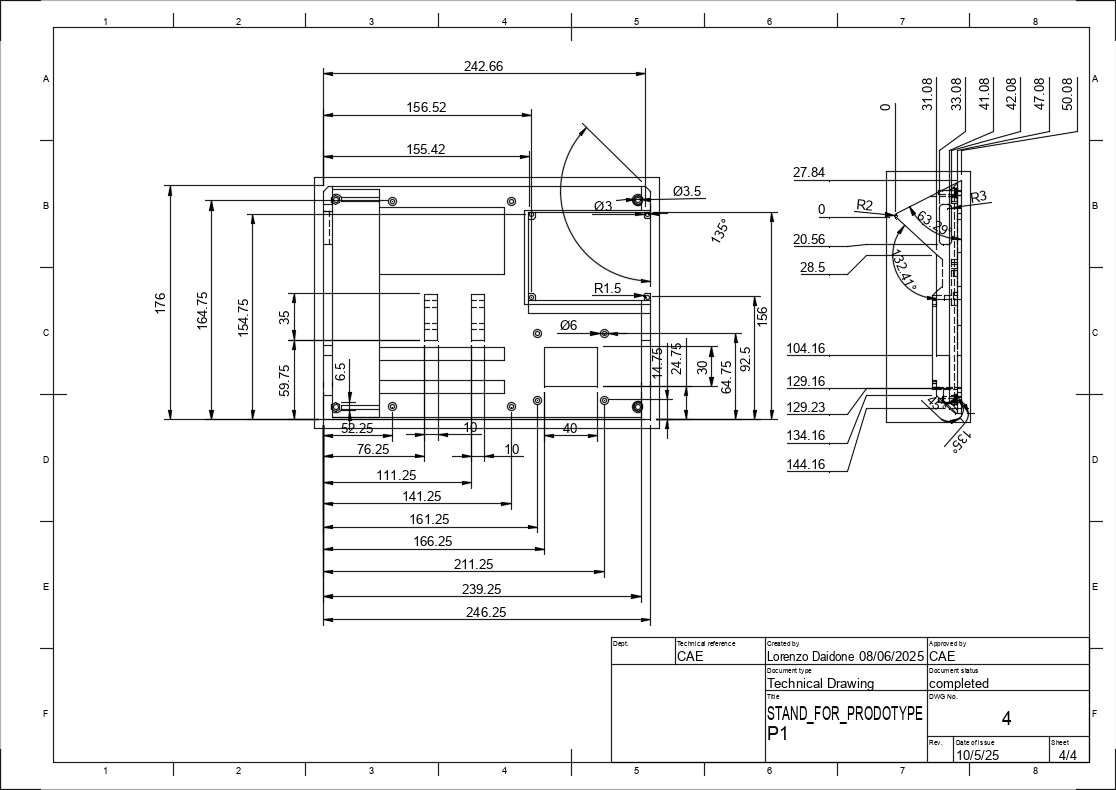


### 4.3CMid plate



### 4.3DBack plate

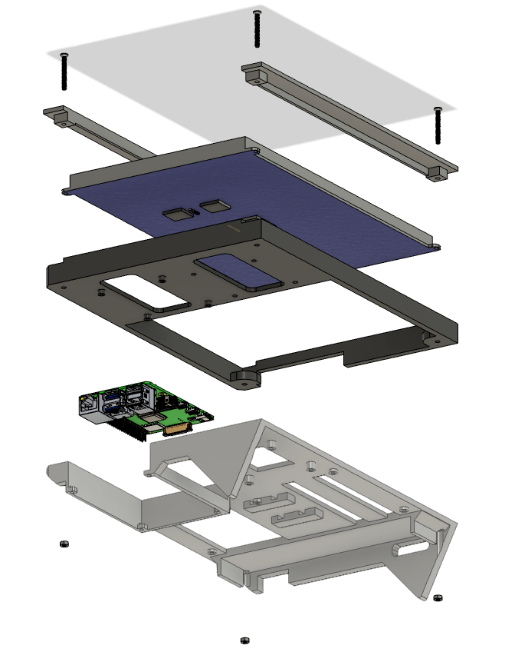
****

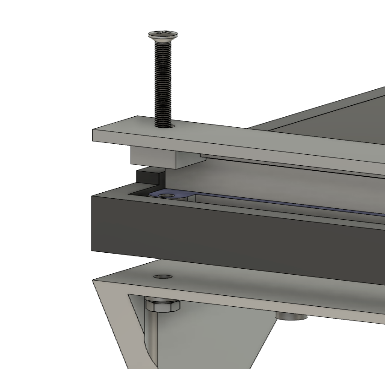
****

## 4.4 Assembly

This is the view of the main assembly for the CAE\_PDF\_PRINTER the main structural components, meaning its what keep all the electronics in place to make one device.

1. **Front Top plate**
2. **Front Bottom plate**
3. **Mid plate**
4. **Back Plate**

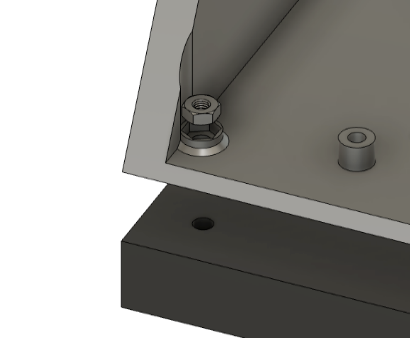


The design is rather simple it allows for everything to be held together with 8 screws and 4 bolts, the bolts pass into a hex mold in the backplate and the screws screw in from the front plate passing through the mounting holes from the 10.1 LCD and securing the LCD to the middle plate and getting fastened by the m3 bolts on the backplate.

## 4.4A Screw Path

As can be seen in the image on the left the screw M3x25 passes through the front plate (top & bottom ones are done the same picture shows bottom) , passes through the fixture holes of the lcd, then through the mid plate and lastly through the backplate.

The top plate has a small block displaced on the inner side to press on the LCD PCB for a better fixture and to be well clamped to the structure.



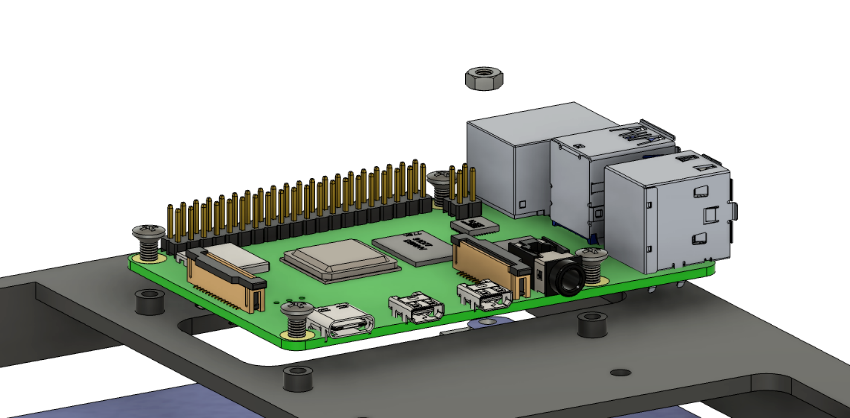
### 4.4B M3 nut placement

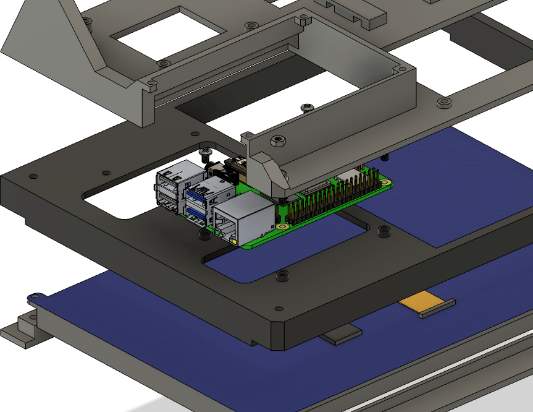
so in terms of what fastens the screw is a m3 bolt which is attached to the backplate of the PC, the is a small hex shaped indent to on the back side of the back plate, this allows a m3 bolt to slide into place. Before starting assembly its useful to pre place the bolts into the indent so when passing the screws through all the parts it allows less holding of parts and can be passed through in one go.

### 4.4C Computer installation

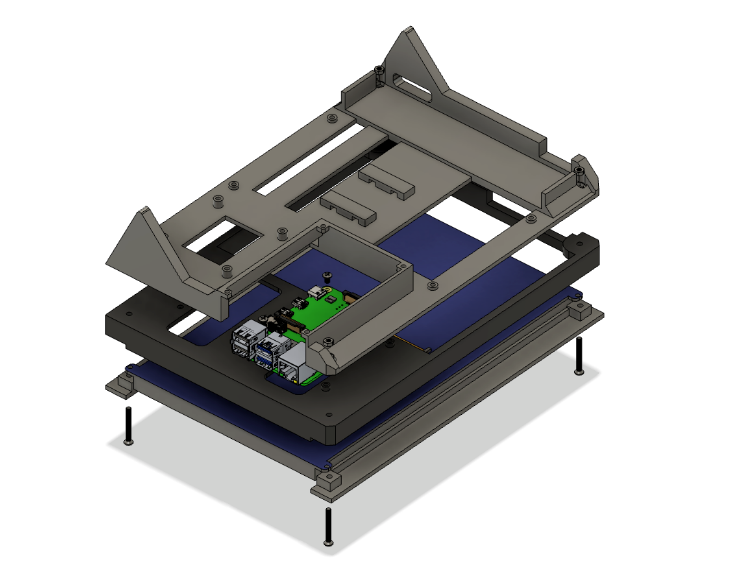
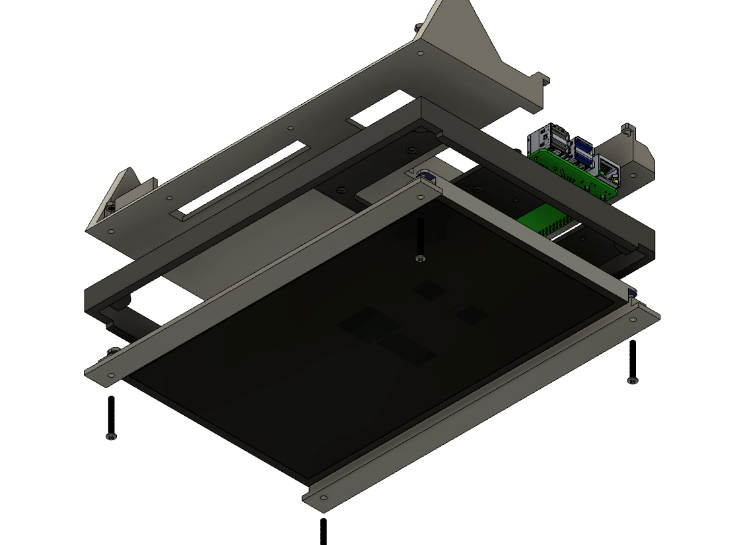
The installation of the computer consist of 4 M3x5 screws, its fixed to the Mid plate PRT(3). This allows the computer to be installed either on a fully assembled computer or directly to the plate before assembling the rest. This is important since if there is a issue with the operating system then it can be changed by removing the computer without disassembling the whole thing. and exchanging a SD card which has the operating system.

The screw holes have ben made slightly smaller and tapped so the screws don’t need a bolt to be fixed, its important to not force the screw into the thread so it doesn’t get DE threaded.





### 4.4D Exploded view



# Results

In this section I will be going over the results and how I’ve validated the results for my project, firstly I need to note the important parts of the working product, that being that its intended purpose has ben met, the workflow process is simple enough for to be operated by technicians and that it doesn’t fail in a unexpected manner.

To prove my product works I’ve put it through a series of test which are mostly based on feedback from technicians and set of tests to prove device operation in different environment cases. These test are based on how the memory is sorted, making sure it doesn’t overflow, how energy is sorted to make sure the device works at a given set of cases in terms of energy loss and operation time. Once these test are completed I can be confident that this device will work properly.

## 5.1 Operation Testing

For testing I must put the printer in many scenarios and see if it complies to the ease of use for resetting everything into a working manner. This being:

1. Un plug it then plug it and try printing to see if I can confuse the PC, once its not working trying the reset button and print again to see if the reset button works in terms of how the server is communicating with the system

**(completed)**

1. Making 100 continuous prints without fail and making sure they all get passes through as a PDF and labelled differently.

**(completed)**

1. Printing all printable QTG’s into the PDF printer to make sure all of them work other than the hardware QTG’s

**(completed)**

1. Leaving it powered for over 48 hours and printing to see if all power systems are working properly.

**(completed)**

1. Power it off through a power cut and powering back up and checking if it still prints properly.

**(completed)**

1. Resetting IP address and port to a different IP and back to the original to see if it still can print, this allows to see how well it transfers between sims.

**(completed)**

1. Setting up the download procedures on another PC with a newly flashed OS and seeing if the functionality matches the original.

**(completed)**

## 5.2 Working process Testing:

The Test of working process of the device consist of a Technician testing the device using the software datasheet in operation section as a reference. The working process of the device is, 1 GUI is opened which will also start NC listener on the selected port. Then once its all running it executes terminal commands through the press of buttons. This allows programs to run in the background the most important being CUPS.

Workflow:

1. The engineer turns on the PC
2. When PC opens it will automatically open the GUI
3. Once GUI is opened then QTG tests can be carried out and printed.
4. Once printed they can press open folder and will open the directory which file is saved and file can be opened to be viewed on the GUI it self and can the modified.
5. Then there’s a button to save the file, using this it can be renamed to the same mane of the file header and can be numbered in the specified char array as needed.

When it comes to the actual procedures for the device it self its quite simple, if you can use a phone you can use this device. So there’s not really much to mention when it comes to procedures which is a good thing because it means its rather simple and still works. That being said I’ve carried this test with the engineer doing the QTG tests for the 747 and he got it working in 10 mins. **(completed)**

## 5.3 observed issue

There was one problem encountered when testing the device, when I left it plugged on the power for 48 hours I checked the logs and found there were 2 cases of undervoltage which indicate, (1) the power supply isn’t supplying enough power, (2) the cable isn’t rated for the current draw, (3) has a faulty voltage regulator. That being said I started with changing the power supply to a spare of the same kind and left it for another 48 hr and still had a single under voltage reading so decided to run the pc at max CPU frequency to be sure I can have it working on max load to test this problem. To do this I installed Linux-cpupower tool and ran:

* Sudo cpupower frequency-set -g performance
* Sudo cpupower frequency-set -u 2300 **(max freq in MHZ)**

This sets it to run on the max CPU frequency even on low power tasks, so then I made it run the python program, the only difference is the refresh rate was very high. This takes the max amount of power the PI can use, then afterwards I used a simple USB tool which measures the energy usage passing on a USB and was showing 3.1A of current. Which is rather strange. The screen used is rated for 0.5A and the PI rated for 3.0A meaning it should be operating at 3.5A at max, the power supply is rated at 20W be the manufacturer and the devices are all working on 5v meaning it can deliver up to 4A. This gives me the idea that the power supply.

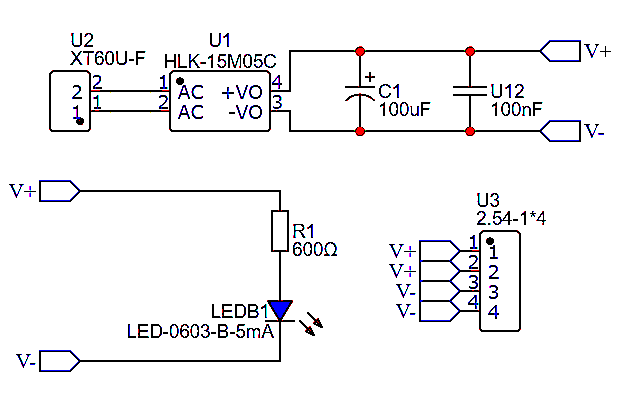
Testing power supply, I made simple circuit of a potentiometer and a 4A fuse in series with the USB output, meaning if the 4A fuse burns before the power supply than its stated electrical parameters are correct, if the supply burns before the fuse than its given values from manufacturer might not be correct. So simply I plugged it in starting at 20 OHM slowly moving down the resistance till I saw any failure, at 2.25 OHM the power supply started smoking and burned down. When I opened it up I found a burned Power Transistor, meaning its actually rated for 11.1W (at least that transistor was)and not the indicated 20W found using ohms Law & equation of power. I’m generally amazed that while running the highest load on using the CPU power tool didn’t trigger more under voltage warnings.

**( P = VI & V = IR )**

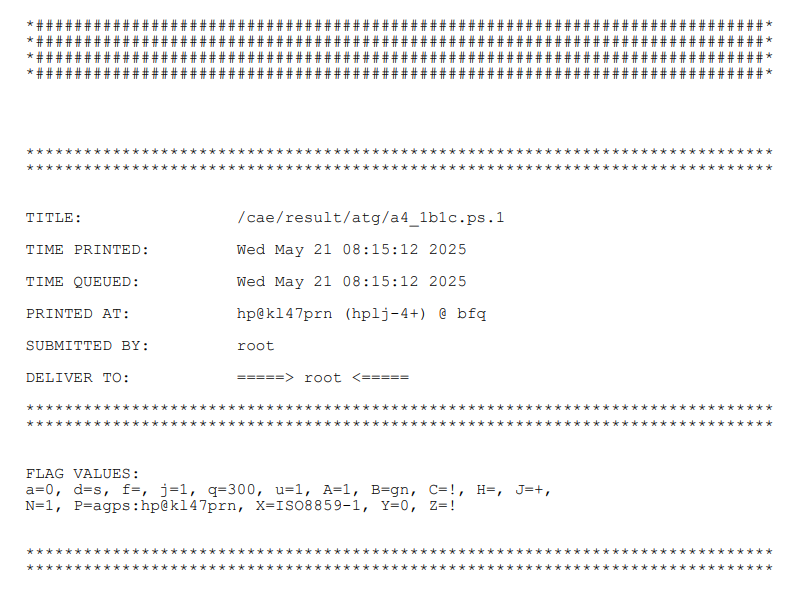
Seeing this made a lot of sense since the PI uses max of 3A and the screen 0.5 that means that it was running on the edge of the power supply capability and maybe occasionally for a microsecond pulled a slightly larger load leading to the undervoltage reading I saw on the logs, when operating normally. To counter this problem I put a new power supply but am left with this impression of not trusting the labels on Chinese power supplies so I also designed a new with parts I’ve used and tested a lot one but only have had time to make a prototype and not a tested PCB from a verified manufacturer. (Schematic attached below) its quite simple, I didn’t add a regulator since the PC and Monitor have already on their PCB’s.

**Consists of:**

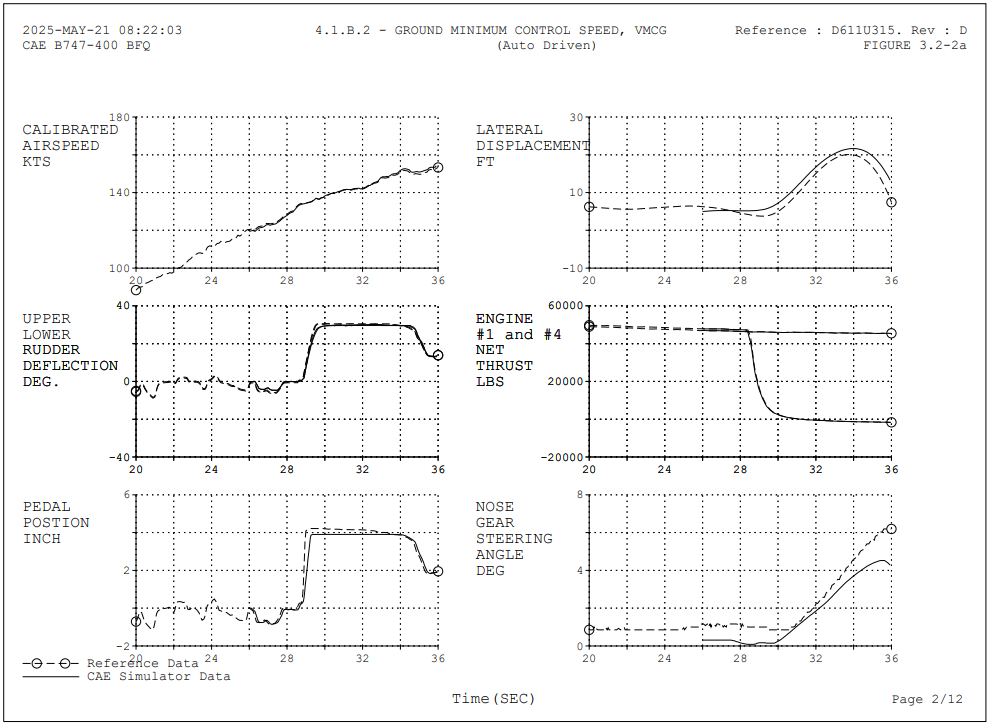
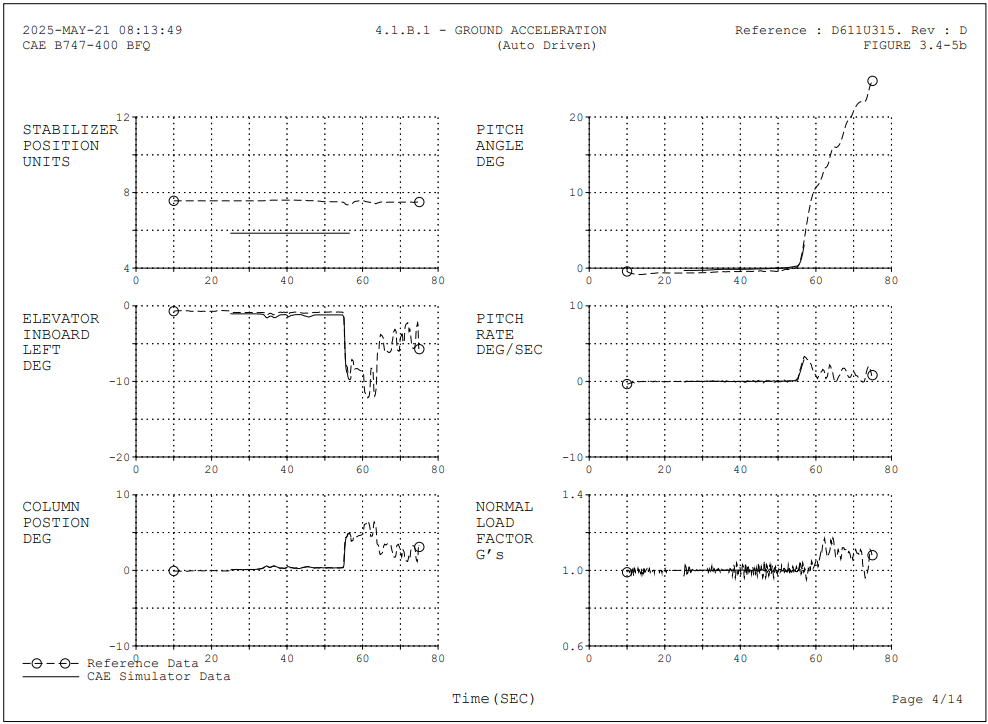
* 220v AC -> 5V 3A DC Converter to drop and rectify current
* 100uf Electrolytic & 100nf ceramic capacitors for filtering
* 600Ω resistor & LED for status



## 5.4 QTG digital Results

Here I will be describing the results of the device which I’ve made, to get these results I personally ran the QTG’s on the 747 simulator and printed them through the iOS which is the same process that has always ben there but now the print is received by the CAE\_PDF\_PRINTER . These results are attached below:

Title page: one of the main requirements is to be able to read the title page when chekcing the QTG on the GUI app, this is because the naming of the files is very important to be able to track which one is which through the logs, when the file is saved using the GUI the top where it says title has the name a4\_1b1c.ps and this is a important piece of information for CAE. The rest of the file just contains the test data as can be seen below.



# Improvements

For improvements I asked engineers to give me feedback and didn’t really get any back since they couldn’t find much to improve on, personally I do see things my device can improve on but also my self which if could have given a better outcome overall in my personal opinion.

Firstly make the device work for all QTG’s, like stated this PC can digitalize QTG’s for all QTG’s besides the hardware QTG’s regarding things like control loading and motion. These use old hardware like a plotter and a spectrum analyser, these are generally quite simple devices used to measure and record complex data. Giving it a good look I believe with a bit more time I could have implemented this also, I’ve used a ADC on a microcontroller to plot voltages against time which is a oscilloscope and could probably make it into a low sampling rate spectrum analyser and send the data via SPI or I2C to the main pc and plot it there into a pdf. Its a bit harder since most of it is c and assembly but completely possible, I could have maybe done it in the time I was here but got quite distracted by working on the sims instead so I used maybe a 1/8 of the time in work to actually work on the project the rest was working on sims, going through all the electronics of the aircraft parts and the software of the sims, which was very distracting from the project it self, I personally think I could have done a lot more just for my personal satisfaction more than for anything else since I’ve completed all required tasks given to me.

Since I’ve finished the PC 2 months early I decided to start working also on a network switch which will switch the PDF Printer from the HP printer and as a switch is toggled the PDF printer turns on and the network is switched, to keep it simple I made a circuit switch the ethernet connections using RY-5W-K relay switches which have 6 terminals 2 input and 2 output, the input come from the IBM pc and the output between the two printers, this allows it to change between the devices using just a switch but also allows it to give power to the PDF Pinter all at the same time, making it a simpler more elegant task, also cool since it’s a mechanical network switch. I will make a separate file for this also since its quite nice and I will probably use it in the future.

**Documentation:**

(software)

Cups - [Documentation - CUPS.org](https://www.cups.org/documentation.html)

netcat - <https://netcat.sourceforge.net/netcat.pdf>

**Python libs**

Pyside - <https://doc.qt.io/qtforpython-6/PySide6/QtWidgets/index.html>

Tempfile - <https://docs.python.org/2/library/tempfile.html>

Shutil - <https://pydoc-zh.readthedocs.io/en/latest/library/shutil.html>

Subprocess - <https://pydoc-zh.readthedocs.io/en/latest/library/subprocess.html>

**Drivers**

QT - <https://doc.qt.io/qt-6/qtwidgets-index.html>

Xserver - <https://docs.freebsd.org/en/books/han>

Libgl1 - <https://dri.freedesktop.org/wiki/libGL/>

Udisk - [UDisks Reference Manual: UDisks Reference Manual](https://storaged.org/doc/udisks2-api/latest/)

Udevil - <https://ignorantguru.github.io/udevil/>

Pmount - <https://linux.die.net/man/1/pmount>

**(Regulations Documentation)**

EASA CS-FSTD(A) - <https://www.easa.europa.eu/en/document-library/certification-specifications/cs-fstda-issue-2>

(Hardware)

RPI 4B - <https://datasheets.raspberrypi.com/rpi4/raspberry-pi-4-datasheet.pdf>