Design and implementation of a signaling system for a novel light-based bioprinter

Design och implementering av ett signalsystem för en ny ljusbaserad bioprinter

Osman Abdalla

9/15/2023

Examensarbete inom
Ekonomi och elektroteknik
Grundnivå, 15 hp
Handledare på KTH: Aurora Rosato
Examinator: Elias Said
TRITA-CBH-GRU-2023:257

KTH
Skolan för kemi, bioteknologi och hälsa
141 52 Huddinge, Sverige

Abstract

A 3D bioprinter employing light-based technology has been designed and constructed in an EU-funded research initiative known as BRIGHTER (Bioprinting by Light-Sheet Lithography). This initiative is a collaborative effort between institutions and companies and aims to develop a technique for efficient and accurate production of engineered tissue.

Presently, the bioprinter's function is limited to 2D printing, with the lack of 3D printing capabilities.

The problem addressed is the integration of two separate electronic systems within the bioprinter to control the laser beam's trajectory for 3D printing. The goal of the project is to create functional software and simulation tools to control the hardware modules in a precise and synchronized manner, thereby enabling 3D printing.

The outcome manifests as a software prototype, which successfully facilitates intercommunication between the two electronic subsystems within the bioprinter, thereby enabling further progress on the bioprinter with 3D printing available. Nevertheless, the prototype requires thorough testing to determine its optimal operational efficiency in terms of timing the movements for the various hardware modules.

Keywords

System integration, 3D bioprinter, hardware communication, microcontroller, Kanban, system architecture diagram

Sammanfattning

En 3D-bioprinter som använder ljusbaserad teknik har designats och konstruerats i ett EU-finansierat forskningsinitiativ som kallas BRIGHTER (Bioprinting by Light-Sheet Lithography). Detta initiativ är ett samarbete mellan institutioner och företag och syftar till att utveckla en teknik för effektiv och korrekt produktion av konstruerad vävnad.

I dagsläget har bioprintern inte möjligheten för 3D-utskrift, utan är begränsad till 2D-utskrift.

Problemet som åtgärdas är integrationen av två separata elektroniska system inom bioprintern för att styra laserstrålens bana för 3D-utskrift. Målet med projektet är att skapa funktionell mjukvara och simuleringsverktyg för att styra hårdvarumodulerna på ett exakt och synkroniserat sätt och därigenom möjliggöra 3D-utskrift.

Resultatet av examensarbetet är en mjukvaruprototyp, som framgångsrikt möjliggör interkommunikation mellan de två elektroniska systemen inom bioprintern och därigenom öppnar möjligheten för vidare arbete med 3D-utskrift tillgängligt. Prototypen kräver dock noggranna tester för att fastställa dess optimala operativa effektivitet när det gäller koordinationen av hårdvarumodulernas rörelser.

Nyckelord

System integration, 3D bioskrivare, hårdvarukommunikation, mikrokontroller, Kanban, systemarkitekturdiagram

Acknowledgments

- I extend my sincere gratitude to my KTH supervisor, Aurora Rosato, for her invaluable guidance and unwavering patience throughout this endeavor.
- Additionally, I am deeply appreciative of the collaborative efforts of the team members at both GUF and Mycronic, especially Pontus, Jakob, Jokubas, Levin, and my supervisor Gustaf Mårtensson.
- Finally, I am profoundly thankful to my family, whose unwavering support and presence have been a constant source of strength throughout this transformative journey.

Stockholm, 09/23
Osman Abdalla

Table of contents

1 Introduction

This chapter describes the specific problem that this thesis addresses, discusses the context of the problem, outlines the goals of this thesis project, and outlines the structure of the thesis.

1.1 Background

In the scope of the EU-funded research initiative known as BRIGHTER (Bioprinting by Light-Sheet Lithography), a novel 3D bioprinter employing light-based technology has been designed and constructed. This innovative approach incorporates acousto-optic manipulation of a laser beam in combination with a light-sensitive hydrogel. Collaboratively led by the Institute for Bioengineering of Catalonia (IBEC), Goethe University of Frankfurt (GUF), Technion Israel Institute of Technology (Technion), as well as industrial entities Cellendes and Mycronic, the BRIGHTER project aims to revolutionize bioprinting.

The primary objective is to develop an advanced bioprinting technique capable of swiftly, accurately, and economically producing engineered tissue [1].

This thesis project is executed in collaboration with two key contributors, Mycronic and GUF.

1.2 Problem statement

In the rapidly evolving field of bioprinting, where innovative techniques like light-based technology hold promise for tissue engineering, there exists a challenge related to the integration and optimization of various components. As bioprinting evolves and becomes more complex, the effective coordination of hardware modules and software interfaces becomes crucial to achieving accurate, efficient, and economical tissue constructs.

This thesis seeks to address the following key issues:

- Integration of multiple hardware modules presents a challenge in terms of synchronization and communication. Developing a solution that coordinates these modules to work in harmony is essential for the successful execution of 3D bioprinting tasks.
- The development of a user-friendly graphical user interface (GUI) that facilitates precise control of the bioprinting hardware is a significant challenge. Designing an intuitive interface that enables researchers and technicians to interact seamlessly with the bioprinter for efficient experimentation and production is essential.

1.3 Goals

The goal of this project is to develop a functioning prototype software GUI to control the laser-based bioprinting of cell-laden hydrogels, including simulation software to provide optimized bioprinting strategy. This can be divided into the following two sub-goals:

1. The primary goal of the project is to design and implement a software solution to synchronise the signalling of three hardware modules (acousto-optics, galvanometer scanner, stage motor) that control the movement of a laser beam in a light-based bioprinter.

2. The secondary goal of the project is to design and implement an optimisation solution that will identify optimal optical dose parameters to create well-defined three-dimensional tissue structures in the bioprinter.

1.4 Delimitations

- Given the nature of the project, with the involved teams working in different geographical locations, the work will be done remotely without the bioprinter readily available for tests.

- When the synchronization software solution is finished, testing will be done with specific 3D models for the sake of simplicity.

- Printing an actual object will not be done unless there's time.

2 Background

This chapter provides basic background information about 3D bioprinting and the bioprinting technique used by the bioprinter in this thesis. Additionally, this chapter describes the different hardware modules that make up the bioprinter and software used by it.

2.1 3D bioprinting

3D bioprinting, a derivative of 3D printing, involves the layer-by-layer assembly of living cells and biomaterials to construct synthetic structures mimicking authentic biological tissue [2]. Various bioprinting methods exist for crafting tissue, including:

1. Extrusion-based bioprinting
2. Inkjet-based bioprinting
3. Laser-based bioprinting

Diverse options exist for bio-ink, the biomaterial employed in the fabrication process. The selection of a suitable bio-ink hinges on the intended application, given that different bio-inks excel in specific areas due to their distinct advantages and limitations [3].

There are three main steps in 3D bioprinting:

1. Pre-processing
2. Processing
3. Post-processing

The initial phase is pre-processing, which encompasses computer-based object design and 3D printer setup. The subsequent phase is processing, involving the preliminary preparation of bioink before commencing the printing procedure. The ultimate stage is focused on maturing the bio printed tissue [4].

The applications of 3D bioprinting span diverse domains including tissue engineering, regenerative medicine, 3D organ bioprinting with transplantation capabilities, drug development, and beyond [4].

Notably, the bioprinter developed within the BRIGHTER project employs laser-based stereolithography [5].

2.1.1 Stereolithography

Stereolithography is one of the available bioprinting methods, offering distinct advantages including high resolution and rapidity.

In the context of this project, the bioprinter employs stereolithography through a light-induced reaction known as photopolymerization. This process involves the resin layer within a container being cured via laser light exposure, leading to the solidification of the material. The bioprinter employs two approaches: a top-down or a bottom-up process, referring to the position of the laser in relation to the vat [6]. Notably, BRIGHTER adopts an innovative top-down lithography approach, which differs from prevailing bottom-up, layer-by-layer bioprinting practices[1].

To craft a 3D tissue utilizing this technique, a designed 3D model is converted into series of 2D image slices. Each 2D image corresponds to a cross-sectional view of the 3D model, serving as input to initiate the fabrication process [4].

2.2 Software

2.2.1 G-code

Geometric code (G-code) functions as a programming language employed to operate Computer Numerical Control (CNC) machines and 3D printers. Effectively, it takes the form of a text file comprising G-code commands [7].

Within a G-code file, individual lines encompass G-code commands. Each line initiates with a specific command followed by positional or coordinate values in the X, Y, and Z axes. Each G-code command serves a distinct purpose [7].

Common G-code commands are:

- G00: Rapid positioning – Maximum travel speed from current position to specified position.

- G01: Linear interpolation – Straight line movement at a certain speed represented by the letter F.

- M00: Program stop.

In addition to the X, Y, and Z coordinates that denote positions along different axes, there exist other letters with distinct meanings, such as S or F, associated with the G01 command. The letter S, for instance, corresponds to spindle speed in CNC machines, while in the context of a 3D printer utilizing a laser, it signifies laser power. It's important to note that these letters can carry varying definitions depending on the specific machine being used. For example, S symbolizes spindle speed in CNC machines, but it takes on the role of denoting laser power in a 3D printer equipped with a laser module [8].

2.3 Hardware

2.3.1 Acousto-optic devices

There are several different types of acousto-optic devices. Acousto-optics can be defined as the interaction between sound waves and light waves in a transparent medium. The difference between the acousto-optic modulator and the acousto-optic deflector that will be discussed is how the acoustic wave is modulated[9]. Modulation of the incoming laser beam is achieved by propagating radiofrequency (RF) signal into a transducer which converts the RF signal into acoustic waves, that are then propagated through the transparent medium which leads to changes in the refractive index of the material[10]. This process is illustrated in Fig. 1.

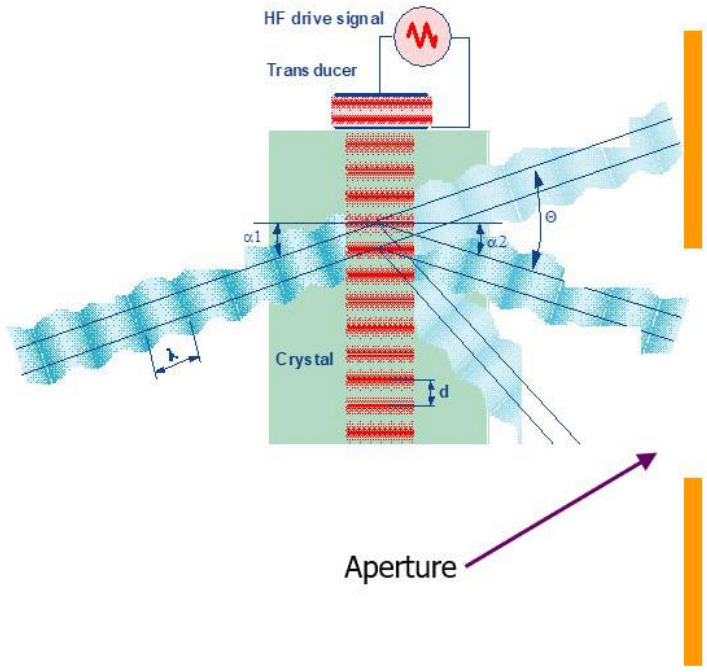

Figure 1. Acousto-optic device[5]

The Acousto-optic Modulator (AOM) allows modulation of the laser beam's intensity at a very high frequency. To do this, the acoustic wave that is propagated through the transparent material is modulated by varying the amplitude[10].

The Acousto-optic Deflector (AOD) allows the sweeping of the modulated laser beam at a very high frequency. In the case of the AOD, the modulation is done by varying the frequency of the acoustic wave, diverging the angle of the beam along one axis[11].

2.3.2 Galvanometer scanners

A galvanometer scanner is an electro-optical component that uses a rotatable mirror to position a laser beam with high precision. Galvanometer scanners have several applications such as laser marking, micromachining, 3d printing and more[12].

A galvo system consists of three main components: the motor or galvanometer, mirror or mirrors, high-precision position detector and the servo which is the driver board that controls the system[12].

They can come in different variations depending on the usage. There are single-axis scanners which can be either X-axis or Y-axis. The other option is dual-axis scanners which would be XY-axis, providing two-dimensional scanning[13].

Figure 2. Galvanometer scanner[14]

2.3.3 Teensy

The development hardware used in this project is Teensy 4.1, which is a development board that uses a powerful microcontroller. There are a total of 55 input and output signal pins available for use. There are many internal units to handle communication between the microprocessor and external devices or devices on the same circuit board, such as:

- 8 Serial ports
- 3 SPI
- 3 I2C
- 3 CAN operative
- USB
- Ethernet

Teensy is commonly called an Arduino clone because its primary programming environment is Arduino's integrated development environment (IDE) software with Teensyduino add-on.

There are other software alternatives depending on the operating system (OS) such as Visual Micro, PlatformIO and CircuitPython[15]. Other project members used and recommended PlatformIO, which is the only free IDE mentioned for use with VSCode.

2.4 Related work

This subsection briefly describes related works. Although these studies may not specifically involve the integration of pre-existing systems, the process of integrating two systems essentially mirrors the creation of a new one, albeit with reduced flexibility.

2.4.1 IOT Based Smart Garbage alert system using Arduino UNO

The scientific paper authored by N. Sathish Kumar, B. Vuayalakshmi, R. Jenifer Prarthana and A. Shankar [16], addresses the construction of a smart garbage alert system. The system's development involved the utilization of various hardware and software components, employing design methods such as block diagrams and flow charts for the systems.

2.4.2 Sun Tracking System

The scientific paper authored by Anish Sarla and Sai Charan Reddy Dandu [17], addresses the design and implementation of a sun tracking system. Same as in the work mentioned in 2.4.1, the system is portrayed using a block diagram to illustrate its components, and a flowchart is employed to explain the algorithm.

The authors successfully developed a prototype tracking system that continuously tracks light rays to maximize solar energy production.

2.4.3 A Low-Cost Portable Smart Card Based Attendance System

The scientific paper authored by Vibin Mammen Vinod, Govindasamy Murugesan, V Mekala, S Thokaiandal, M Vishnudevi and S M Siddharth [18], addresses the design and implementation of a smart attendance system. This work also utilizes block diagrams and flowcharts.

The utilization of block diagrams and flowcharts in this paper, as well as in the studies referenced in sections 2.4.1 and 2.4.2, appears to be a standard practice for the design and implementation of a system.

2.4.4 Integration och Interoperabilitet vid utveckling av IT-system

The scientific paper authored Mohammed AL-Hilfi and Emil Olovsson [19], explores how system developers can overcome complex issues when building new IT-systems. They primarily focus on the technical and organizational interoperability of the integration process.

In this paper, the authors describe the various phases of integrating an IT system, starting from its design to its actual launch. They emphasize the close relationship between integration and interoperability, which often depends on the system's specific requirements. It's crucial to have a deep understanding of the interfaces involved and how the physical components of the system are connected.

3 Methodologies and methods

This chapter offers a thorough overview of the methodologies and the tools used in this thesis. It starts with a literature study and tool introduction, including software development methods. The experimental design and GUI integration are discussed, along with hardware and software components.

3.1 Literature study and pre-study

A comprehensive review of existing literature is conducted to establish a foundational understanding of 3D bioprinting, as well as an exploration of the diverse bioprinting methods currently available. Moreover, the objective is to identify similar works and the methodologies they employed. This literature study is performed by conducting searches for articles on Google Scholar.

A substantial portion of the preliminary study involves comprehending the existing solution through discussions with various expert team members from both GUF and Mycronic, who are responsible for different aspects such as hardware, electronics, software, and optics. While some documentation is accessible, it is important to note that certain information has become outdated due to a project hiatus.

Concerning the Mycronic system, significant insights pertaining to prior work were extracted from the relevant work carried out in Ohanyan's master thesis project [5].

3.2 Software development methodologies

3.2.1 Agile

Agile is a collective name for several software development methodologies. It was based on the manifesto where the authors valued:

- Individuals and interactions over processes and tools.
- Working software over comprehensive documentation.
- Customer collaboration over contract negotiation.
- Responding to change over following a plan.

Many of these authors are also the creators of the different methodologies under the Agile name like SCRUM, Extreme programming (XP), Test-driven development(TDD), Lean software development and more[20].

3.2.1.1 Kanban

Kanban, an iterative approach rooted in Agile and Lean principles, introduces a streamlined methodology. Analogous to sticky notes on a wall, a Kanban board serves as its digital counterpart for software development. The board is sectioned horizontally into groups that denote distinct project stages, and within these, task cards are positioned. These cards, each carrying varying priority levels, can be assigned to team members. As tasks progress, cards transition from left to right across the board, ultimately reaching completion. This visual framework enhances project comprehension and clarifies each team member's contribution within the broader context, facilitating a seamless progression from inception to conclusion.

Kanban, alongside other Agile methods, contrasts with the linear waterfall model, which mandates the sequential completion of phases before progression. The Waterfall model lacks flexibility, discouraging overlap between phases. Consequently, the waterfall model is unsuitable for this thesis.

Given the project's geographical dispersion of teams, solo development, and adaptable workflow, Kanban emerged as the favored software development approach over other alternatives.

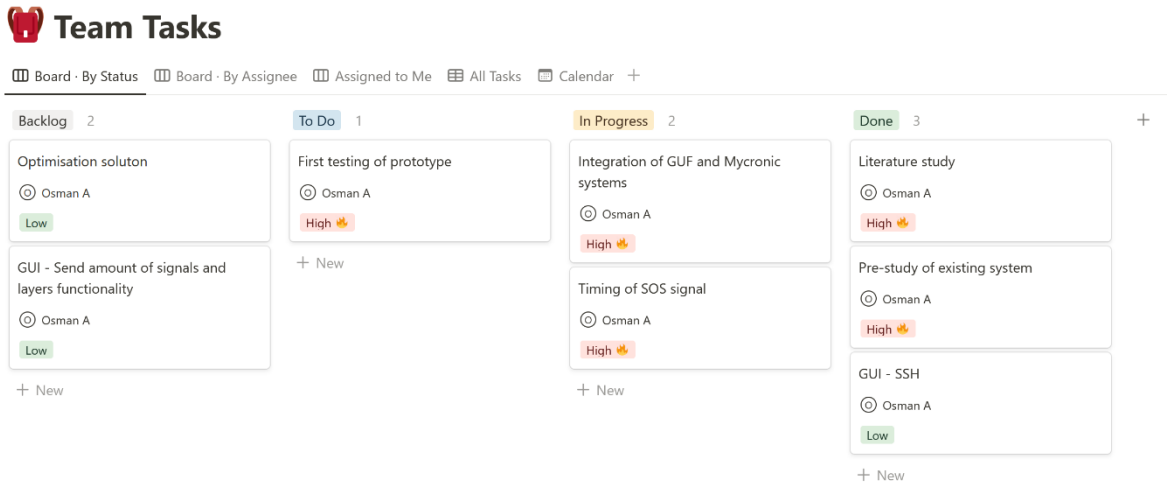

Figure 3. Kanban board at one point of the project

3.2.2 Waterfall model

The waterfall model is generally considered the traditional development method where the project is divided into several phases such as:

- Requirement analysis
- Design
- Implementation
- Testing
- Operation and maintenance

Each phase must be completed before proceeding to the next, allowing for very little flexibility and no overlap between phases, making it costly when requirements change in the later phases of a project[21].

3.3 Hardware

3.3.1 AOD/AOM rack

This module, as previously mentioned, is the optical hardware provided by Mycronic in the Brighter project and must be integrated into the setup that GUF already has with the scanners and stages.

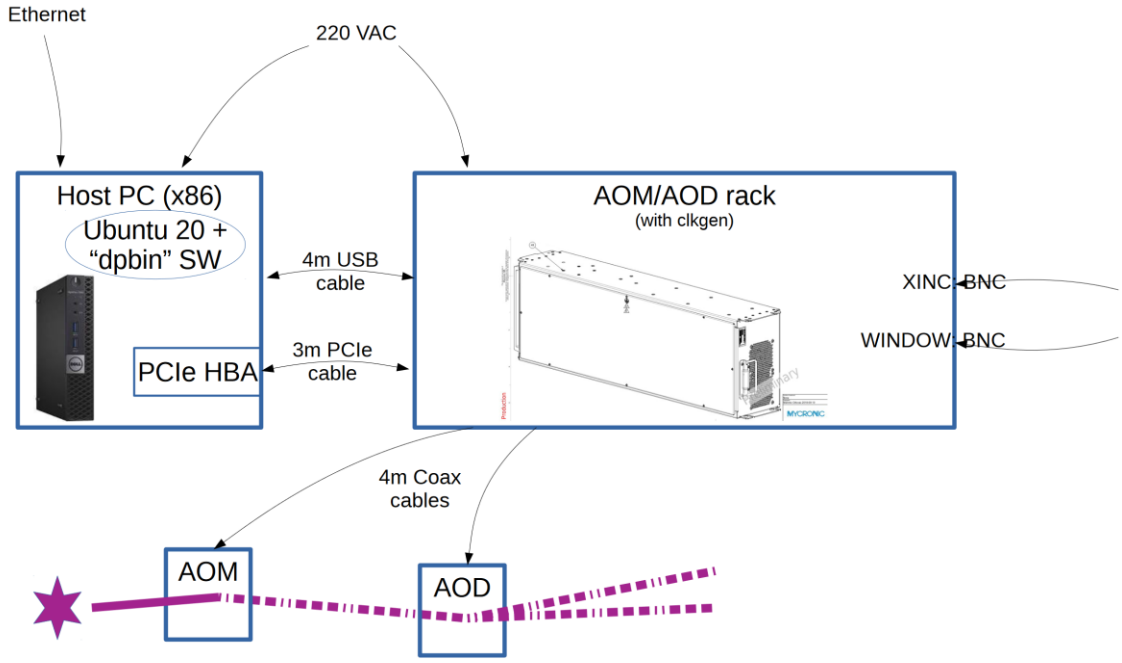

Figure 4. Hardware overview of Mycronic system (Provided by Mycronic team members)

Although AODs and AOMs are established devices within the field, the instance employed in this project is a specialized creation by Mycronic tailored specifically

for use in the bioprinter. Both the AOD and AOM have been assembled within a dedicated rack. This rack is maneuvered by 2.5V logic signals, which undergo conversion into an internal Start of Sweep (SOS) signal, subsequently steering the logic of the modulator and deflector. The XINC and WINDOW shown in figure 4, represent SOS and XWIN signals and a custom cable was made to facilitate the connection between Teensy and the rack.

The rack is complemented by an Ubuntu PC (Worksbrighter), which interfaces with various cables, as depicted in figure 4. Mycronic's provided software operates on this PC, serving the purpose of formatting a 3D model suitably and transferring pattern data that is utilized with each SOS signal. This operation is facilitated through the PCI-express connection, serving as the conduit between the hardware and the software components.

3.4 Software

3.4.1 AOD/AOM software

As mentioned in 3.3.1, Mycronic has provided accompanying software with the AOD/AOM rack, encompassing scripts for various uses.

There are two important signals needed by the AOD/AOM:

1. XWIN
2. SOS

The AOD/AOM has its own internally simulated SOS signals for initialization purposes. To change from the simulated ones to signals driven by an external source, a gating signal must be set high. A gating signal is usually used to allow or block transmission of a signal. In our case XWIN is that gating signal. When the XWIN signal is set to high by the Teensy, the AOD/AOM becomes receptive to SOS signals from an external source, in our case, the Teensy.

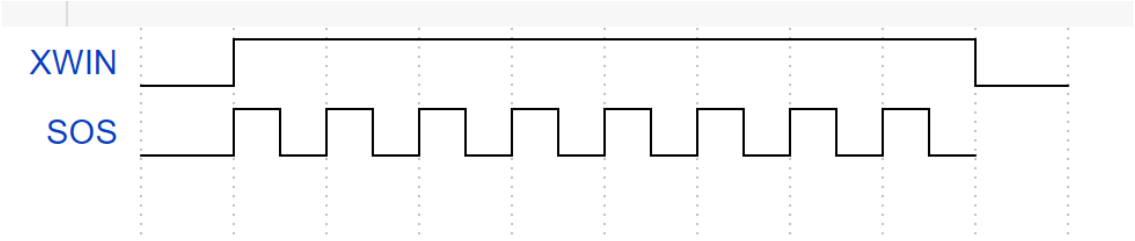

Figure 5. Timing diagram of XWIN and SOS signal

Prior to the initiation of the printing process, a sequence of conversions is required for the 3D model. This conversion starts with the original .stl format, serving as a representation of the 3D model. Subsequently, the model undergoes slicing, segmenting it into distinct layers. Ultimately, it evolves into an .st file, a unique Mycronic format tailored for pixel data representation.

Once the .st file is generated, the AOD/AOM devices can be uploaded with patterning data. This data is then “sweeped out” with the SOS signal, which activates every time the connected pin on the Teensy undergoes a transition from low to high, triggering the corresponding sweep.

3.4.2 GUI

GUF developed a GUI with several features such as controlling the stage and scanners. The communication between the GUI and Teensy was done through serial communication.

One functionality of the GUI that was relevant to the integration was the uploading and sending of the G-code file to the Teensy board. The G-code file is derived from a 3D model in a separate software application. Following its upload into the GUI, the G-code is subsequently transmitted through the serial port to the Teensy. Once this transmission is completed, the printing process can be initiated.

The printing process at the beginning of this project was a function in the Teensy firmware which parsed G-code commands and based on those commands moved the stage and scanner to certain points.

3.5 Experimental design

3.5.1 System architecture design

The system consists of a multitude of components, encompassing both hardware and software elements, with two pivotal players.

The first is the Teensy, serving as the hardware core, responsible for transmitting signals to the hardware modules. The second is the GUI, acting as the software core, initiating the printing procedure by relaying commands to both the Teensy and Worksbrighter.

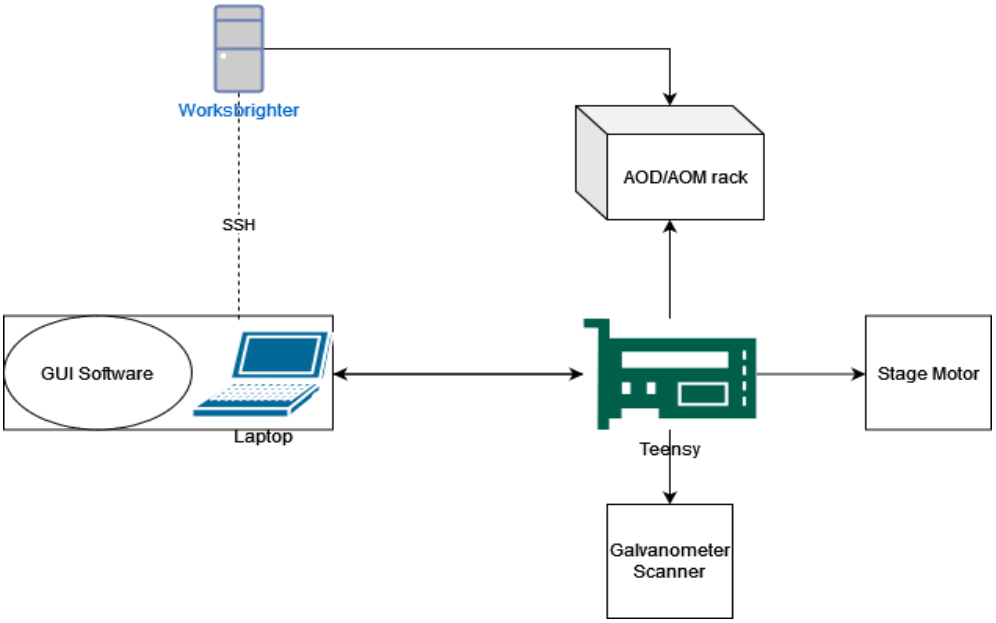

Figure 6. System architecture diagram of the two systems

As shown in figure 6, the Teensy establishes physical connections with various hardware modules. The initiation of the printing process is facilitated by the GUI on a Windows PC. It's noteworthy that the operating system of the desktop, termed Worksbrighter, has Ubuntu, which is particularly significant due to its physical linkage to the AOD/AOM rack.

The GUI cannot be run on Ubuntu and this was an issue that could be solved by two approaches:

- Secure Shell (SSH)
- Virtual Machine (VM)

The first approach is a network protocol that is used for secure remote login, transferring files, and issuing remote commands[22].

The second approach is a VM which is a virtual computer where one can allocate the amount of RAM, CPU cores and diskspace[23]. A VM can run your preferred OS which makes running Windows possible on Worksrbrighter, allowing the use of the GUI.

SSH emerged as the more pragmatic and fitting approach for this project due to the remote nature of development and constrained access to Worksbrighter throughout the thesis duration. This method additionally offers users enhanced versatility, permitting work from personal laptops, thereby not being restricted to Worksbrighter alone.

3.6 Test environment

For the test environment, the code in the firmware was written with the required layer count and SOS signals with the same 3D model in mind. This ensured the

code worked as theorized, reducing the chances of errors, and ultimately improving the efficiency of the process. The layer count was 399, with a total of 886 SOS signals required for each layer.

Timing for movement between the AOD/AOM, scanner and stage were set with a good amount of margin to make sure movements were done sequentially in X, Y and Z order.

Testing of various features was done incrementally to ensure they individually worked as intended, before moving on to the testing of the complete synchronization solution.

An oscilloscope was primarily used to measure the signals coming from the Teensy microcontroller. It also checked if the AOD/AOM received the signal correctly. This was important because the rapid patterning made it difficult to see visually.

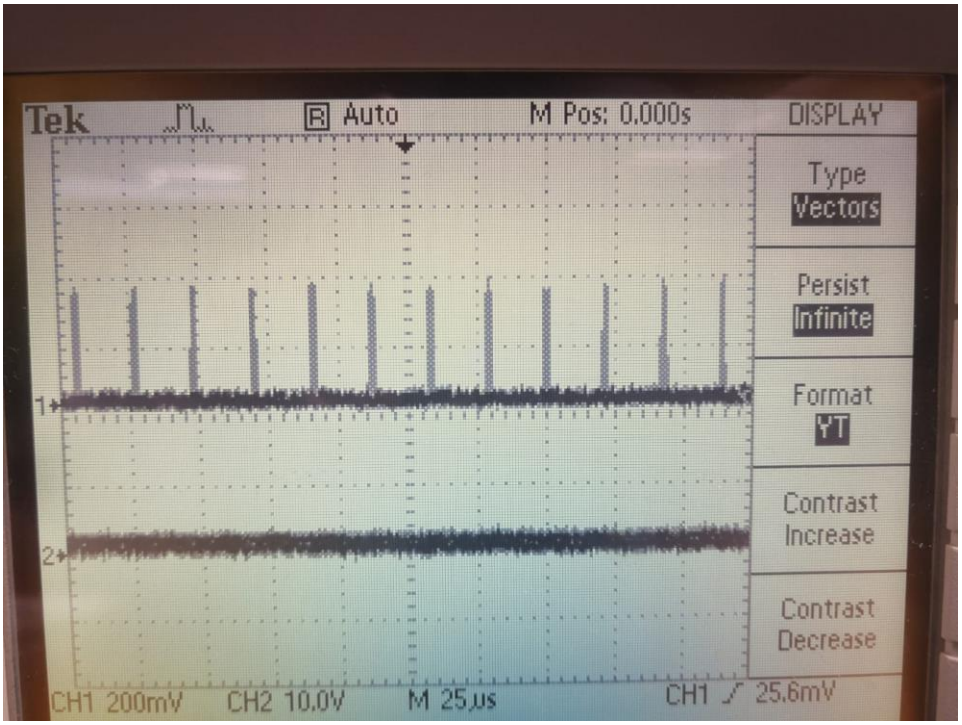

Figure 7. Oscilloscope showing the SOS signals sent by the Teensy.

In addition, a frequency counter was employed to count the SOS signals sent from the Teensy. This was done to confirm the count shown by the Mycronic software with the count from the frequency counter, which helped in the troubleshooting phase.

3.7 Tools

3.7.1 Teensy development

The primary development environment for the Teensy is Arduino IDE. This IDE is characterized by its beginner-friendly nature, featuring incorporated libraries and examples for communication with different protocols, controlling different displays, controlling different motors and more [24], [25].

Visual Studio Code (VS code) made by Microsoft was the source-code editor that was used in this project. It has a rich variety of extensions for different applications. A lot of languages come with built-in support like JavaScript, TypeScript, CSS and HTML but it also supports most major programming languages via its VS Code Marketplace where extensions can be installed for respective language[26].

The primary extension used with VS code is PlatformIO which is an IDE that supports several operating systems like Windows, Linux, and Mac. PlatformIO has extensive support for different boards, development platforms and frameworks, the board used in this project and the Arduino framework included[27].

VScode was chosen over Arduino IDE because, even though the latter is simpler to set up and work with for simpler projects. VScode has a lot of helpful features that helped with this project such as Live share and GitHub integration, making remote collaboration easier. It was also used and recommended by other teams that worked on this project.

3.7.2 GUI development

The GUI used a different environment for development called Visual Studio. Although both share the same creator and almost identical names, they are used for different purposes. Visual Studio is an actual IDE which is more complete from the beginning in contrast to VS code which is highly customizable and dependent on extensions for different goals.[28]

Development for the GUI was done with C# in Visual Studio, the native IDE for this language. Windows forms was used and it’s a UI framework used to create desktop applications. The primary extension used for this thesis was SSH.NET which was used to facilitate communication between the GUI and Worksbrighter.

4 Results and analysis

This chapter showcases the results of the preliminary study where we collected comprehensive data about the two systems and developed the system architecture diagram. Additionally, it will introduce two solutions with their corresponding outcomes.

4.1 Preliminary study results

Although the systems were physically interconnected, there was a lack of software communication between the GUF and Mycronic systems. The key questions that we addressed are:

- What hardware is used?
- What software is used?
- What communication protocols are available?
- What's the current software solution?

With these questions answered, a system architecture diagram was designed (as shown in figure 6).

The first and second question helped provide a clearer overview of the hardware and software components being utilized. The third question identified existing communication protocols and potential integration options. The fourth question involved an analysis of the current codebase, including the Teensy firmware, GUI code, and the AOD/AOM software.

4.2 G-code integration

During the implementation process, the GUI played a central role in the system. The system was designed with the GUI as the starting point for all actions, connecting all the different parts together.

To add 3D patterning capabilities using the existing G-code solution, the AOD/AOM was integrated in both Teensy and the GUI. To control the Mycronic system, SSH connection was added. This allowed for commands to be sent from the GUI to control the AOD/AOM. SSH was then used to upload each 2D layer by sending a specific character ('O') through a serial port from Teensy to the GUI. The GUI, upon receiving this character, then sent the necessary command through SSH to upload the upcoming layer using Worksbrighter.

Additionally, the Teensy microcontroller directly sent an SOS signal to control the AOD/AOM. Since the AOD/AOM managed the X-axis movement, an SOS signal was sent for each movement of the scanner, which governed the Y-axis movement.

There were two issues with this approach, the first issue is how G-code pathing works. When looking at G-code files used, the coordinates were very random. For example, starting in the middle and then jumping to the top or bottom. This is probably for optimization purposes. This type of movement is not compatible with the Mycronic system since it's only able to make a sweeping movement, basically diverging the angle of the laser beam along one axis. The other issue is how many scanner movements we get from the G-code file based on the amount of Y coordinates parsed. This is an issue because of how the Mycronic system is expecting a certain amount of SOS signals for a 2D layer, which doesn't correspond with the Y-movements in the G-code. For every 2D layer uploaded there's a specific amount of SOS signals to be sent to finish one layer.

The flowchart shown in figure 8, shows the integration with the already existing codebase. Certain modifications were done to integrate the AOD/AOM such as modification of the scanner to only use one mirror instead of two, the serial commands, the addition of SSH communication and SOS signal.

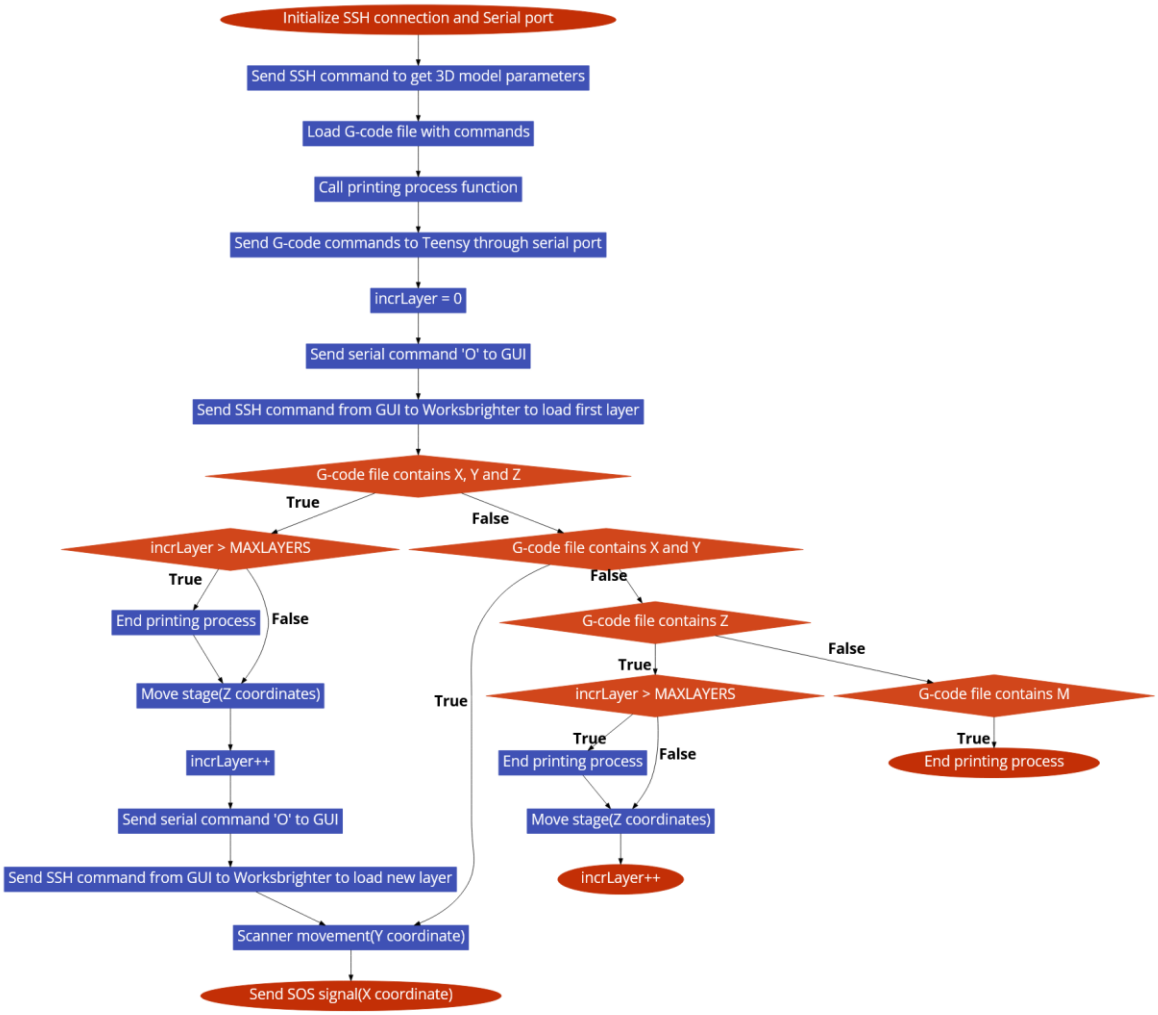

Figure 8. Flowchart of one iteration in a continous loop of the G-code integration

4.3 Custom implementation

This implementation relies on a continuous loop that correlates with the layer count of the 3D model being utilized. The primary loop essentially represents the 3D model. Nested within this primary loop is another loop for each 2D layer, based on the repetitive X and Y movement until exposure is finished and it's time for another layer.

Furthermore, this implementation offered the advantage of adjusting the number of SOS signals and synchronizing them with the appropriate scanner movements, thereby streamlining the integration process. This strategy also made testing easier by allowing simple adjustments to essential parameters like SOS signal count and layers.

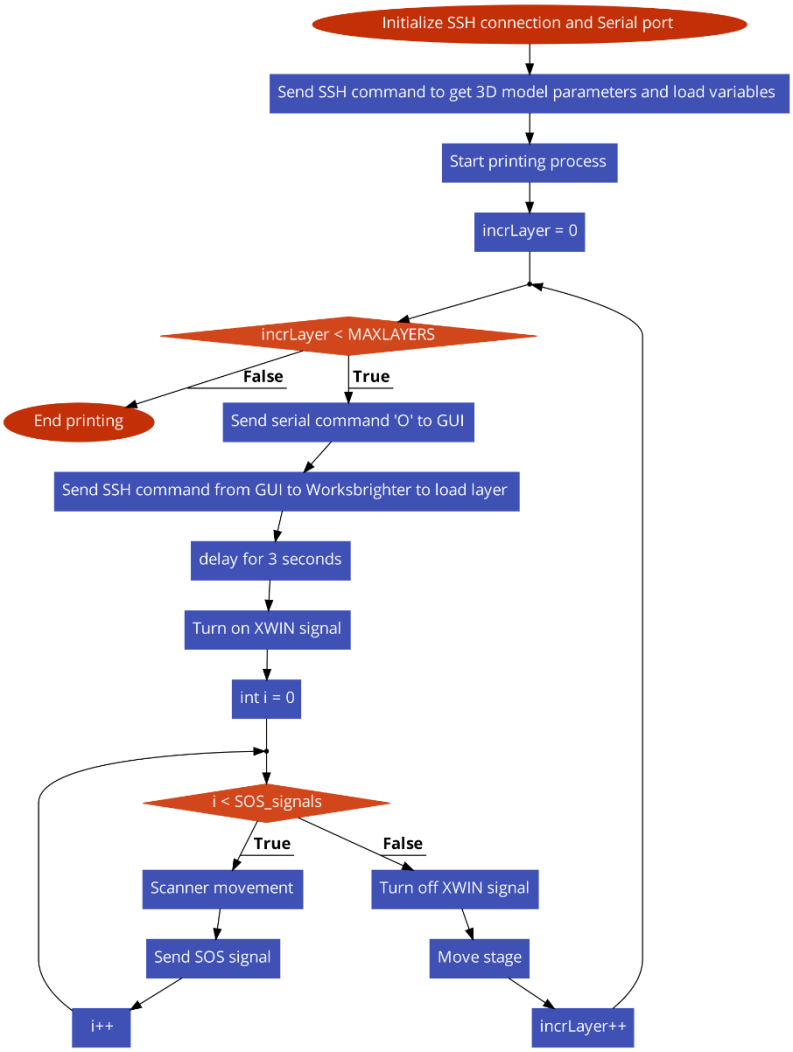

Figure 9. Flowchart of the custom implementation

The biggest difference between this implementation and the G-code integration, other than the absence of G-code coordinates, was the addition of the XWIN signal. Previously, the XWIN signal was internally set in the AOD/AOM firmware, contrary to normal practices by other Mycronic machines.

The XWIN signal must be set high before the printing process begins to allow for SOS signals to be received by the AOD/AOM, and then set back to low when printing is finished.

Apart from this, features such as SSH communication, sending SOS signal, Serial commands, and the modified scanner function were inherited from the G-code integration.

4.4 Test results

Following several rounds of testing the custom implementation, it was determined that a delay of approximately 3 seconds was essential before sending SOS signals. This delay was attributed to the time required for a pattern to be uploaded and for the AOD/AOM to be prepared to receive SOS signals.

The duration during which an SOS signal remained in the HIGH state was modified from 100 microseconds to 1 microsecond. Additionally, the delay between every SOS signal, originally set at 150 microseconds, was subsequently decreased to 20 microseconds.

It is noteworthy to highlight that the scanner's Y-axis movement and the stage's Z-axis movement were not subjected to testing during the completion of the integration process. Consequently, the timing parameters for these movements retain a considerable margin and are yet to be fine-tuned for optimization.

The conducted testing revealed the successful establishment of a communication flow between the GUF system (GUI and Teensy) and the Mycronic system (AOD/AOM and Worksbrighter). As a result of this integration, the exposure of numerous layers was successfully executed in the software.

4.5 Analysis

The primary goal of this thesis was the design and implementation of a synchronization solution between the hardware modules in the systems. Before development began, there was no communication between the GUF and Mycronic systems.

Rather than continuing with the initial G-code solution, which was considered unfeasible, the choice was made to completely redevelop the printing process function. Limited by time constraints, although there were initial attempts to explore alternative G-code generation methods compatible with the Mycronic system, it was ultimately determined that a quick, custom solution had the potential to fulfill the goals of this thesis.

It's possible to suggest that adopting an alternative agile methodology emphasizing greater teamwork and more frequent meetings could have facilitated the early detection of the incompatibility of the G-code integration. This, in turn, could have conserved valuable time in attaining the objectives outlined in section 1.3. Such an

approach might guard against tunnel vision by getting essential feedback and hearing different perspectives. However, the chosen method was deemed more suitable considering the nature of this project.

With the custom implementation, communication between the systems was established and streamlined. However, achieving the best performance requires more testing than expected. Most tests were conducted without involving the scanner and the stage responsible for Y-axis and Z-axis movement. As a result, the timing parameters are not finely tuned for optimal performance, but rather with some margin. Limited accessibility to the bioprinter at GUF hindered testing, rendering it ineffective within the time constraints of this thesis's conclusion.

Commencement of comprehensive testing occurred concurrently with the initiation of a secondary bioprinter's construction at Mycronic. However, since this secondary bioprinter was still under construction, it lacked most of the hardware modules, thereby imposing constraints on certain testing aspects. Nonetheless, the availability of Worksbrighter, AOD/AOM and Teensy facilitated substantial advancements in the testing process.

Based on the goals set in section 1.3, the results were deemed satisfactory, though not fully comprehensive. The primary goal was attained and can be further enhanced through additional testing. However, the achievement of the secondary goal hinged upon the fulfillment of the primary objective which was accomplished by the conclusion of this thesis.

4.6 Discussion

The utility of the chosen Kanban method as a solo developer lay in its ability to provide a structured approach while still offering adaptability. Task visualization was facilitated through usage of distinct cards representing diverse activities such as the addition and testing of various features, leading to distinct waypoints that led to the desired result.

In contrast to several other software development methodologies that mandate frequent meetings, the meetings in this project were irregular, influenced by participant availability and organized on an as-needed basis. Opting for Kanban was apt given its exemption from formal meeting requirements, one reason for its known flexibility.

The first software prototype, the G-code integration discussed in section 4.2, enabled communication between the two systems. However, it wasn't entirely flawless, but a general idea of the implementation was established. This approach also brought to light certain problems and disadvantages related to the parsing of G-

code itself. Despite identifying these issues later in the project, the G-code integration had components that were thoroughly tested, such as SSH communication, serial communication, and a general idea of the code flow. This made the development of the custom implementation relatively quick within the short timeframe that was available.

4.7 Social and economical aspect

The selection and implementation of the chosen design were executed with a measure of consideration for the project's economic and social implications. Notably, the central system orchestrating the interlinking of the two systems, the GUI, exhibited a prominent issue. As briefly outlined in section 3.6.2, the GUI's development was native to the Windows OS, thereby prompting the identification of a predicament and corresponding solutions, as elaborated in section 3.3.1.

Among the two presented solutions, opting for an SSH connection to Worksbrighter emerged as the more economically viable alternative. This rationale stems from the cost comparison between implementing an SSH connection and the procurement of licenses for both VM and Windows.

The SSH implementation is considered “free” in the sense that it doesn’t cost any money. However, the cost in this context pertains to the time expended for development. On the other hand, purchasing licenses for VM and Windows involves financial expenditure, although it may not demand as extensive practical effort.

Ultimately, the decision was made to proceed with the implementation of an SSH connection for a variety of reasons:

- The time required for developing the SSH connection was not long enough to justify the adoption of the alternative choice involving running the GUI directly on Worksbrighter.

- The long-term benefit of a user-friendly GUI utilization outweighed the potential convenience of running a VM with Windows on Worksbrighter from the user’s standpoint.

The second point constituted the central core in the development of the GUI. Emphasizing a user-centric approach, where critical operations occur seamlessly, resulted in a GUI accessible to all, requiring minimal guidance and expertise for its utilization. Also, potentially increasing effectivity and productivity with lesser personnel needed for its usage.

4.8 Ethical aspect

Considering the bioprinter's intended use for human tissue printing, it's essential to address privacy and data integrity concerns. Initially, the G-code method transferred data from a text file to the GUI and then to the Teensy microcontroller. While the initial solution didn’t necessarily need internet connectivity, the resulting integration uses SSH and doesn't need internet either if Worksbrighter and the GUI-connected computer share the same network. Additionally, no extensive data transfer occurs; Worksbrighter holds the complex patterning data, which are harder to decipher than a text file, and the GUI only instructs the uploading process.

4.9 Environmental aspect

The process of integration may have reached completion; however, it has not yet attained full optimization. To effectively account for the environmental aspects inherent in this integration, the attainment of a fully optimized system necessitates precise synchronization of timing intervals between the movement of the three hardware modules. As stipulated in section 4.4, additional experimentation is obligatory to expedite the printing process, thereby potentially resulting in diminished electricity consumption.

5 Conclusions and future work

5.1 Conclusions

In conclusion, this thesis delves into the realm of 3D bioprinting with the goal of integrating two electronic systems. This integration is achieved through the design and implementation of a software solution, which synchronizes the signaling of three hardware modules.

The results demonstrate the successful establishment of a signaling system, which streamlines communication between the two systems and enables 3D capabilities as a result. This achievement paves the way for further progress on the project, as the integration stands as a crucial milestone for the project.

However, the testing phase incurred greater time investment and engagement than initially anticipated. Consequently, additional testing is imperative to achieve optimal performance. This also implies that the secondary goal was not met and requires attention for future development. Efficient advancement of the project might require dedicated personnel and increased collaboration among teams.

While the primary focus of this thesis is on bioprinting, it's important to note that the methods and methodologies employed here are not necessarily restricted solely to bioprinting. Instead, they possess a broader applicability and can prove valuable in the integration of electronic systems in a more general sense.

The same set of questions outlined in section 4.1 can indeed be applied to various electronic systems, including both hardware and software components, when designing and implementing prototype solutions. These questions provide a valuable framework for considering critical aspects and requirements that are relevant across different domains of electronic system development.

5.2 Limitations

The primary constraint was the limited availability to the completed system at GUF in Germany. Given the complexity of the setup involving an array of hardware and software components, substantial testing occurred only toward the conclusion of this thesis, specifically to observe the interaction between the GUF software and the Mycronic software. During this phase, certain problems were identified that required resolution by Mycronic experts for further progress. These issues wouldn’t have been evident through used tools like the oscilloscope and frequency counter.

5.3 Future work

The clear next steps involve first optimizing the laser beam movement based on testing timing across hardware modules. Then, the secondary goal of this thesis, which is to identify optical dose parameters to create well-defined 3D tissue structures in the bioprinter, remains to be addressed and achieved.

References

[1] "Brighter – Bioprinting by Light-Sheet Lithography." https://brighterproject.eu/ (accessed Nov. 17, 2022).

[2] I. T. Ozbolat, "Bioprinting scale-up tissue and organ constructs for transplantation," *Trends Biotechnol.*, vol. 33, no. 7, pp. 395–400, Jul. 2015, doi: 10.1016/j.tibtech.2015.04.005. (accessed Sep. 04, 2023)

[3] S. Vanaei, M. S. Parizi, S. Vanaei, F. Salemizadehparizi, and H. R. Vanaei, "An Overview on Materials and Techniques in 3D Bioprinting Toward Biomedical Application," *Eng. Regen.*, vol. 2, pp. 1–18, Jan. 2021, doi: 10.1016/j.engreg.2020.12.001. (accessed Sep. 04, 2023)

[4] A. M. Bejoy *et al.*, "An insight on advances and applications of 3d bioprinting: A review," *Bioprinting*, vol. 24, p. e00176, Dec. 2021, doi: 10.1016/j.bprint.2021.e00176. (accessed Sep. 04, 2023)

[5] R. Ohanyan, "A novel dynamic acousto-optic lithographic patterning technique for bioprinting," p. 66.

[6] M. Pagac *et al.*, "A Review of Vat Photopolymerization Technology: Materials, Applications, Challenges, and Future Trends of 3D Printing," *Polymers*, vol. 13, no. 4, Art. no. 4, Jan. 2021, doi: 10.3390/polym13040598. (accessed Sep. 04, 2023)

[7] A. C. Brown and D. de Beer, "Development of a stereolithography (STL) slicing and G-code generation algorithm for an entry level 3-D printer," in *2013 Africon*, Sep. 2013, pp. 1–5. doi: 10.1109/AFRCON.2013.6757836. (accessed Sep. 04, 2023)

[8] "G-Code Formatting Reference." https://tormach.com/g-code-formatting-reference (accessed Apr. 05, 2023).

[9] "IntroductionAO.pdf." Accessed: Aug. 23, 2023. [Online]. Available: https://www.optoscience.com/maker/gooch/pdf/IntroductionAO.pdf

[10] R. Mesleh and A. AL-Olaimat, "Acousto-Optical Modulators for Free Space Optical Wireless Communication Systems," *J. Opt. Commun. Netw.*, vol. 10, no. 5, pp. 515–522, May 2018, doi: 10.1364/JOCN.10.000515. (accessed Sep. 04, 2023)

[11] G. R. B. E. Römer and P. Bechtold, "Electro-optic and Acousto-optic Laser Beam Scanners," *Phys. Procedia*, vol. 56, pp. 29–39, 2014, doi: 10.1016/j.phpro.2014.08.092. (accessed Sep. 04, 2023)

[12] "Galvanometer Scanners | Scanlab." https://www.scanlab.de/en/service/glossary/galvanometer-scanners (accessed Dec. 02, 2022).

[13] "Dual-Axis Galvanometer Scan Head Systems, ±22.5° Scan Angle." https://www.thorlabs.com (accessed Dec. 12, 2022).

[14] "01_dynAXIS_galvanometer scanner.pdf." Accessed: Dec. 02, 2022. [Online]. Available: https://www.scanlab.de/sites/default/files/2020-07/01_dynAXIS_galvanometer%20scanner.pdf

[15] "Teensy® 4.1." https://www.pjrc.com/store/teensy41.html#communication (accessed Dec. 12, 2022).

[16] N. S. Kumar, B. Vuayalakshmi, R. J. Prarthana, and A. Shankar, "IOT based smart garbage alert system using Arduino UNO," in *2016 IEEE Region 10 Conference (TENCON)*, Singapore: IEEE, Nov. 2016, pp. 1028–1034. doi: 10.1109/TENCON.2016.7848162. (accessed Sep. 04, 2023)

[17] S. C. R. Dandu and A. Sarla, *Sun Tracking System*. 2022. Accessed: Sep. 04, 2023. [Online]. Available: https://urn.kb.se/resolve?urn=urn:nbn:se:bth-23262

[18] V. M. Vinod, G. Murugesan, V. Mekala, S. Thokaiandal, M. Vishnudevi, and S. M. Siddharth, "A Low-Cost Portable Smart Card Based Attendance System," *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 1012, no. 1, p. 012046, Jan. 2021, doi: 10.1088/1757-899X/1012/1/012046. (accessed Sep. 04, 2023)

[19] M. AL-Hilfi and E. Olovsson, *Integration & Interoperabilitet vid utveckling av IT-system*. 2017. Accessed: Sep. 04, 2023. [Online]. Available: https://urn.kb.se/resolve?urn=urn:nbn:se:ltu:diva-64992

[20] "Manifesto for Agile Software Development." https://agilemanifesto.org/ (accessed Dec. 21, 2022).

[21] A. A. A. Adenowo and B. A. Adenowo, "Software Engineering Methodologies: A Review of the Waterfall Model and Object-Oriented Approach," vol. 4, no. 7, 2013. (accessed Sep. 04, 2023).

[22] "What is the Secure Shell (SSH) Protocol? | SSH Academy." https://www.ssh.com/academy/ssh/protocol (accessed May 02, 2023).

[23] "What is a Virtual Machine? | VMware Glossary," *VMware*. https://www.vmware.com/topics/glossary/content/virtual-machine.html (accessed May 02, 2023).

[24] "Built-in Examples | Arduino Documentation." https://docs.arduino.cc/built-in-examples/ (accessed Dec. 15, 2022).

[25] "Libraries - Arduino Reference." https://www.arduino.cc/reference/en/libraries/ (accessed Dec. 15, 2022).

[26] "Language Support in Visual Studio Code." https://code.visualstudio.com/docs/languages/overview (accessed Dec. 15, 2022).

[27] "PlatformIOIDE— PlatformIO v6.1 documentation." https://docs.platformio.org/en/stable/integration/ide/pioide.html (accessed Dec. 15, 2022).

[28] “Visual Studio: IDE and Code Editor for Software Developers and Teams,” *Visual Studio*. https://visualstudio.microsoft.com (accessed Aug. 18, 2023).