UNIVERSITY OF THESSALY

SCHOOL OF ENGINEERING

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

IoT management system for industrial production lines

Diploma Thesis

Alexandros Nikolaidis

Supervisor: Christos Sotiriou

Volos, September 2023

UNIVERSITY OF THESSALY

SCHOOL OF ENGINEERING

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

IoT management system for industrial production lines

Diploma Thesis

Alexandros Nikolaidis

Supervisor: Christos Sotiriou

Volos, September 2023

ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

IoT σύστημα διαχείρισης για βιομηχανικές γραμμές παραγωγής

Διπλωματική Εργασία

Αλέξανδρος Νικολαΐδης

Επιβλέπων: Χρήστος Σωτηρίου

Βόλος, Σεπτέμβριος 2023

Approved by the Examination Committee:

Supervisor **Christos Sotiriou**
Professor, Department of Electrical and Computer Engineering, University of Thessaly

Member **Fotios Plessas**
Professor, Department of Electrical and Computer Engineering, University of Thessaly

Member **Spyridon Loutridis**
Professor, Department of Electrical and Computer Engineering, University of Thessaly

Acknowledgements

I would like to express my gratitude to my supervisor, Professor Christos Sotiriou, for his invaluable guidance and support throughout the entire duration of my thesis. His expertise and dedication have been instrumental in shaping the outcome of this research. I would also like to thank the the dedicated team at DNA Filters LTD, who during my time there, helped me bring this project to fruition. Lastly I would like to thank my family and friends for standing by and supporting me all these years in all of my academic and personal pursuits.

DISCLAIMER ON ACADEMIC ETHICS AND INTELLECTUAL PROPERTY RIGHTS

«Being fully aware of the implications of copyright laws, I expressly state that this diploma thesis, as well as the electronic files and source codes developed or modified in the course of this thesis, are solely the product of my personal work and do not infringe any rights of intellectual property, personality and personal data of third parties, do not contain work / contributions of third parties for which the permission of the authors / beneficiaries is required and are not a product of partial or complete plagiarism, while the sources used are limited to the bibliographic references only and meet the rules of scientific citing. The points where I have used ideas, text, files and / or sources of other authors are clearly mentioned in the text with the appropriate citation and the relevant complete reference is included in the bibliographic references section. I also declare that the results of the work have not been used to obtain another degree. I fully, individually and personally undertake all legal and administrative consequences that may arise in the event that it is proven, in the course of time, that this thesis or part of it does not belong to me because it is a product of plagiarism».

The declarant

Alexandros Nikolaidis

Diploma Thesis

IoT management system for industrial production lines

Alexandros Nikolaidis

Abstract

This thesis seeks to design, develop, and evaluate an Internet of Things (IoT) production line automation system based on two microcontrollers, RFID technology, 2.4GHz wireless communication, and a relational database for data storage. When there is a need to scale up their production, many small-scale production lines are hampered by the fact that they rely significantly on manual labor. The project presented in this Thesis is an endeavor to automate some manufacturing processes and increase the production line's overall efficiency. Our ultimate objective was to take the first step toward full automation by providing the fundamental components of a system that can control the various manufacturing parameters, provide detailed logging of production batches, and assist engineers in identifying system failures and deficiencies. Numerous requirements for this project derive from the application's specific characteristics, such as the chemical properties of the injection host material, the already-established working principles of the operators, and the proof-of-concept concept at that time. The implementation decisions were made with development simplicity in mind and were tailored to the requirements of this particular application. Using 13.56 MHz RFID technology, the system identifies each mold and retrieves the necessary data from a database operating on a local server. The processed data is then communicated to the PLC in order to establish the parameters according to the specific mold characteristics. Simultaneously, the system maintains a comprehensive log of all processes, allowing the end-user to extract and analyze the data retrospectively. The system was evaluated to test the hypothesis and demonstrate its effect on the overall process, resulting in a 64.8% improvement in productivity.

Διπλωματική Εργασία

IoT σύστημα διαχείρισης για βιομηχανικές γραμμές παραγωγής

Αλέξανδρος Νικολαΐδης

Περίληψη

Η παρούσα διπλωματική εργασία επιδιώκει να σχεδιάσει, να αναπτύξει και να αξιολογήσει ένα σύστημα αυτοματισμού γραμμής παραγωγής Internet of Things (IoT) που βασίζεται σε δύο μικροελεγκτές, τεχνολογία RFID, ασύρματη επικοινωνία 2.4 GHz και μια σχεσιακή βάση δεδομένων για αποθήκευση δεδομένων. Όταν υπάρχει ανάγκη να αυξηθεί η παραγωγή τους, πολλές γραμμές παραγωγής μικρής κλίμακας παρεμποδίζονται από το γεγονός ότι βασίζονται σημαντικά στη χειρωνακτική εργασία. Η έρευνα που παρουσιάζεται σε αυτή τη Διατριβή είναι μια προσπάθεια αυτοματοποίησης ορισμένων παραγωγικών διαδικασιών και αύξησης της συνολικής απόδοσης της γραμμής παραγωγής. Ο απώτερος στόχος μας ήταν να κάνουμε το πρώτο βήμα προς την πλήρη αυτοματοποίηση παρέχοντας τα θεμελιώδη στοιχεία ενός συστήματος που μπορεί να ελέγξει τις διάφορες παραμέτρους κατασκευής, να παρέχει λεπτομερή καταγραφή των παρτίδων παραγωγής και να βοηθήσει τους μηχανικούς στον εντοπισμό αστοχιών και ελλείψεων του συστήματος. Οι πολυάριθμες απαιτήσεις για αυτό το έργο απορρέουν από τα ειδικά χαρακτηριστικά της εφαρμογής, όπως οι χημικές ιδιότητες του υλικού έγχυσης, οι ήδη καθιερωμένες αρχές λειτουργίας των χειριστών καθώς και η ανάπτυξη του συστήματος σε λογική πρωτοτύπου απόδειξης της ιδέας. Οι αποφάσεις εφαρμογής λήφθηκαν με γνώμονα την αναπτυξιακή απλότητα και ήταν προσαρμοσμένες στις απαιτήσεις της συγκεκριμένης εφαρμογής. Χρησιμοποιώντας τεχνολογία RFID 13.56 MHz, το σύστημα αναγνωρίζει κάθε καλούπι και ανακτά τα απαραίτητα δεδομένα από μια βάση δεδομένων που λειτουργεί σε έναν τοπικό διακομιστή. Στη συνέχεια, τα επεξεργασμένα δεδομένα κοινοποιούνται στο PLC προκειμένου να καθοριστούν οι παράμετροι σύμφωνα με τα συγκεκριμένα χαρακτηριστικά του καλουπιού. Ταυτόχρονα, το σύστημα διατηρεί ένα ολοκληρωμένο αρχείο καταγραφής όλων των διαδικασιών, επιτρέποντας στον τελικό χρήστη να εξάγει και να αναλύει τα δεδομένα αναδρομικά. Το σύστημα αξιολογήθηκε για να ελεγχθεί η υπόθεση και να αποδειχθεί η επίδρασή του στη συνολική διαδικασία, με αποτέλεσμα τη βελτίωση της παραγωγικότητας κατά 64.8%.

Table of contents

List of figures

List of tables

Abbreviations

IDE	Integrated Development Environment
MCU	MicroController Unit
PLC	Programmable Logic Controller
RFID	Radio Frequency Identification
NFC	Near Field Communication
UI	User Interface
IoT	Internet of Things
API	Application Programming Interface
OTA	Over The Air
LCD	Liquid Crystal Display
LED	Light Emmitting Diode
GPIO	General Purpose Input Output
PCB	Printed Circuit Board
CAD	Computer Aided Design

Chapter 1

Introduction

1.1 Production process overview

A company specializes in the design and production of polyurethane automotive parts. These parts are produced through a process called polyurethane Injection Molding. This process is traditionally used for metal parts (die-casting) but can also be used with thermoplastics as host materials. The material is injected at a high pressure into a specially designed mold. The resulting product is net shape, or near net shape, directly out of the mold. There are 2 types of molds, open and closed.

Each mold requires different conditions and different volume of host material, which in reality translates to different material dispensing duration. This certain duration is calculated with the theoretical rates of the injection molding machine, then gets refined through production testing trials. When the development stage is passed the mold (and therefore the part) are available to go to the production stage, the production formula is documented for the operators to follow every time the part needs to be produced. The 2 main unique parameters of each mold are the mold temperature and the volume of the host material, these parameters define the mold formula.

The operators have been given a backorder list, these are products that have been ordered by customers but are not in sufficient quantities in the warehouse. Before commencing on a production batch the operators have to make a production manifest, a document that states which (and how many) products will be included in the batch.

In a single batch there may be more than 40 different products, which means that there need to be more than 40 different formulas followed by the operators. This process works for

small-scale production lines, but when the need arises for the production line to scale up to large-scale manufacturing the process described above quickly becomes a serious restriction.

1.2 Problem description

When dealing with a high number of product variations, manual operations in polyurethane injection molding might in fact provide substantial difficulties and possible problems, principally because human error is a possibility. Here are several issues with manual procedures and how they can be severely impacted by human error:

- Formula Accuracy: Since operators must adhere to over 40 different formulas, there is a significant risk of making mistakes when recording and carrying out the right formula for each product. Even a small error in the temperature or volume of the material used to make the mold can produce components or goods that are not up to par.
- Production Manifest Mistakes: It takes meticulous attention to detail to create a production manifest that includes information on the items and quantities that should be included in the batch. This document may have been prepared manually, which may have resulted in inaccurate production quantities, unfulfilled backorders, or excess inventories of some parts.
- Issues with quality control: Variations in product quality might be caused by human mistake in material application, mold setup, or production parameter modifications. Dimensional, aesthetic, or mechanical variations in the product could have an effect on the general dependability and safety of the vehicle parts.
- An increase in waste might be attributed to setup mistakes or incorrect material dispensing, which can lead to the manufacture of defective products that cannot be sold to customers. This results in more material waste and more expensive production.
- Downtime and Rework: When human mistake results in flaws or production problems, production may be interrupted while the issues are found and fixed. Rework may be necessary as well, increasing both production time and expenses.
- Training and Skill Dependence: Operators must receive significant training to handle each product variation correctly in a manual process with several unique formulations.

There may be a shortage of skilled operators, and as a result, the process becomes dependent on their knowledge, creating problems with scalability and workforce management.

- Scaling is difficult because there are more formulas and product variations as production volume rises, which raises the possibility of human mistake. Managing and controlling manual operations becomes more and more challenging, making it more difficult for the business to scale up production effectively.

- Traceability and Documentation: Manual methods may lead to erroneous or inadequate production data documentation, making it difficult to identify the underlying causes of flaws or deviations from quality standards.

- Safety worries: Operator safety can be put at risk by human mistake while using high-pressure injection molding techniques, which could result in mishaps or injuries.

Automation can lessen the need for human involvement while enhancing uniformity, precision, and repeatability[1]. Implementing cutting-edge control systems and process monitoring in current injection molding machines can help improve quality assurance and overall effectiveness.

1.3 Practical restrictions

Due to the nature of the host material, the injection mold machine has to mix the 2 polymers in real-time as the injection is taking place. This is a necessary process to create polyurethane[2], being the most widely used elastomeric polymer for these kind of applications due to it's ability to withstand chemical abrasion and high temperatures. This restriction stems from a material property known as gel time. Gel time is the duration that the host material requires to go through a phase change and transform into gel consistency. At that time the material should not be circulating through the system and the injection process should be finished. At the event that the host material undergoes a phase change while still in the system, the machine needs to go through a cleansing cycle which in our case translates to several hours of downtime and possibly the need for mechanical parts' replacement.

As far the production processes are concerned, this poses a restriction on the idle time between part injection. The maximum duration the machine can be idle is 30 seconds, that

means that if the idle time passes the 30-second mark the operator may take one of the two courses of action:

- He/She can abort the rest of the injection manifest and commence with the material purging sequence. This protocol essentially uses special cleaning chemicals to purge all the host material from the system, when the purging sequence is done the machine is again ready to use. The downside is that this process is costly both on time (as the production batch is halted, time is spent on purging the machine) and on material (as there is a significant volume of material wasted).

- The second option is to do a ”dummy” material injection in order to acquire another 30 seconds of idle time, hopefully enough to change the injection program (or overcome any difficulty faced) and continue the production batch normal.

1.4 Proposed solution & goals

After we did all the research concerning the injection difficulties and restrictions, we proposed a solution that aims to reduce the human error variability and increase the production line’s productivity through automation;

The system aims to interface with the PLC directly in order to program the injection duration for each part automatically. This requires a microcontroller and assisting hardware connected to the PLC. It also requires a system that can identify each part’s mold and communicate this information back to the microcontroller connected to the PLC.

1.4.1 Identification of molds

The basic architecture for the part of the system that is responsible for identification the molds was decided early-on. There would be a microcontroller that would need to identify each mold and send back the information either via a wireless or wired interface. After testing various technologies of part identification like optical barcode scanners and RFID. The choice was made to go with an RFID antenna-tag system: The microcontroller would be connected to an 13.56MHz RFID antenna and be able to scan 13.56MHz NFC tags that would be placed onto the molds. Then the data will be transmitted back to the other microcontroller

via a 2.4GHz wireless interface. These implementation choices gave us the ability to create a compact electronics case for the microcontroller and RFID antenna.

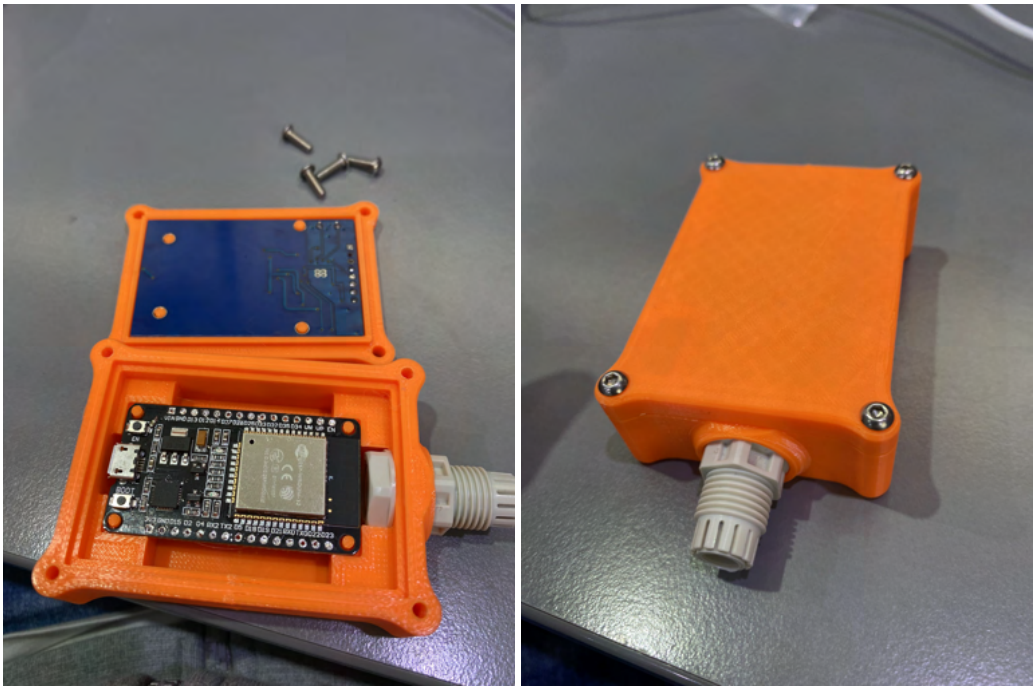


Figure 1.1: Electronics case open (Left) & closed (Right)

1.4.2 Production traceability

Another problem our proposal aims to solve is the traceability and documentation of the production batches (see section 1.2). This can be accomplished by using software and tools such as SQL database, PHP scripts and HTML/CSS/ front-end.

The microcontrollers can now identify each separate mold and wether it the mold injection was successful or not. This information is send by the microcontroller via HTTP requests in a Synology server, PHP scripts handle the requests and perform the needed reads/writes on the SQL logging database, essentially providing a production manifest for each batch.

To make this information easily accessible by the end-user, we created a basic front-end web UI, providing the end-user the ability to read, write and export production logs. In this way we managed to eliminate another point of failure in the system caused by the probable human error.

Chapter 2

Background

2.1 System specifications

2.2 Block diagram

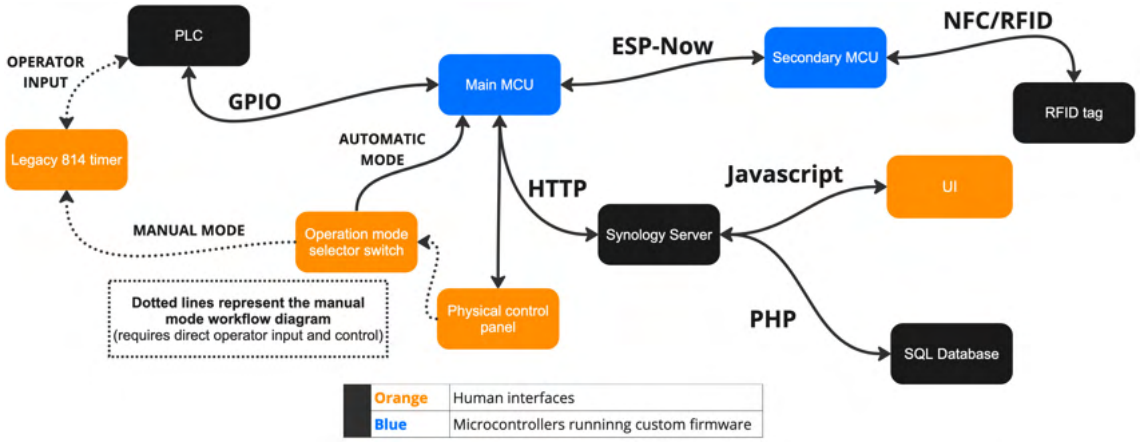


Figure 2.1: Graphical system description

2.3 Technical characteristics & implementation choices

2.3.1 Technologies used

2.3.1.1 Arduino IDE

Tools like the Arduino framework and IDE are necessary for creating firmware for micro-controllers like the ESP32. Programming embedded devices is made easier and more user-

friendly by the Arduino platform. It has a strong ecosystem of hardware support and libraries, making it the best option for the job at hand.

For Arduino-compatible boards, the Arduino IDE is a user-friendly development environment that offers a code editor, compiler, and uploader. Even individuals with little programming knowledge can use it thanks to its clear and user-friendly interface. The Arduino programming language, a C/C++ variation with extra libraries and functions created expressly for microcontroller development, is supported by the IDE.[3]

The GPIO (General Purpose Input/Output) control, analog and digital sensor interfacing, communication protocols, and other complicated activities are made simpler by the Arduino framework's prewritten libraries and functions. By abstracting away low-level hardware operations, these libraries free up developers to concentrate on their projects' higher-level logic. The large library selection and the helpful Arduino community make it possible for developers to obtain materials, examples, and solutions for a variety of applications.

For the project utilizing ESP32 microcontrollers, the Arduino framework makes sense for a number of reasons. First off, it offers first-rate hardware support for ESP32 boards, including pre-built libraries for controlling Bluetooth, Wi-Fi, and other features. This facilitates the use of ESP-NOW to implement wireless communication between the microcontrollers.

Second, the ease of use and beginner-friendliness of the Arduino framework enable developers to quickly prototype and iterate their firmware. The development process is accelerated by the straightforward syntax and extensive library support, which requires less time and work to implement.

Last but not least, there is a lot of documentation, tutorials, and community support due to the widespread usage of Arduino among the maker and IoT communities. Developers may take advantage of available information and tools, successfully debug problems, and cooperate with others working on related projects thanks to this accessibility.

Overall, the ESP32 compatibility, wide library support, and user-friendly development environment of the Arduino IDE and framework make them the ideal tools for this task. Because of its speedy development, straightforward implementation, and friendly community, Arduino is a great choice for this project's firmware development.

For more information and details regarding the development process of the firmware for the microcontrollers please refer to 3.1

2.3.1.2 ESP-Now

Espressif Systems created the ESP-Now communication protocol especially for ESP8266 and ESP32 microcontrollers[4]. It utilizes the same 2.4 GHz frequency spectrum that is frequently used for Wi-Fi communication. It's important to remember that ESP-Now may function without a Wi-Fi network and is independent of the conventional Wi-Fi infrastructure.

ESP-Now makes use of the ESP8266 and ESP32 microcontrollers' integrated Wi-Fi radios to enable direct connection between these gadgets. Peer-to-peer architecture is used, so there is no need for a router or centralized access point for communication between devices.

Low latency, low battery usage, and great reliability are just a few benefits the protocol offers for IoT applications. It is suited for applications requiring real-time or nearly real-time communication since it is optimized for quick and effective data transmission.

Both unicast and multicast communication are supported by ESP-Now. While multicast offers one-to-many communication, allowing a single device to send data to numerous recipients at once, unicast permits one-to-one communication between two devices.

The capacity of ESP-Now to create and maintain secure communication channels utilizing encryption is a noteworthy feature. ESP-Now uses encryption methods to make sure that data sent between devices is safe and difficult to intercept or tamper with.

Devices must be paired or registered with one another in order to establish an ESP-Now connection. Devices exchange security credentials during the pairing process in order to authenticate and build trust. After successful pairing, devices can communicate with one another directly by sending ESP-Now data packets.

The range between ESP-Now devices is determined by a number of variables, such as transmit power, environmental conditions, and potential interference. In open areas, ESP-Now often achieves stable connection within a few hundred meter range. However, physical barriers like walls or structures might degrade the signal's range and quality.

Overall, the ESP-Now protocol is a strong one that permits effective and direct communication between the ESP8266 and ESP32 microcontrollers. It uses the 2.4 GHz frequency spectrum, has minimal latency and power requirements, supports encryption for secure connection, and enables both unicast and multicast transmission. We can easily transmit RFID tag data and enable seamless integration with our IoT system by utilizing ESP-Now in our project to provide a strong and dependable communication channel between the two ESP32 microcontrollers.

2.3.1.3 REST API - HTTP Requests

In our project, we establish communication between an ESP32 MCU running Arduino firmware and a Synology Linux server running a MariaDB SQL database using HTTP requests. In order to streamline communication between the client and the server, we also use a RESTful API (Representational State Transfer).

HTTP requests, as mentioned earlier, form the foundation of data communication on the World Wide Web. They involve a client delivering a request to the server, which responds by processing the request. GET, POST, PUT, and DELETE are some of the several methods used by HTTP requests to carry out distinct server-side operations.

[5]An architectural design known as a RESTful API makes use of the fundamentals of HTTP to enable scalable and effective web services. Its foundation is the idea of resources, each of which is given a distinct URL (Uniform Resource Locator). Each resource in the REST architecture can be modified using conventional HTTP protocols, offering a consistent and standardized manner of communication.

In order to expose endpoints on the Synology Linux server, our project uses a RESTful API. [6]These endpoints stand in for particular system resources like sensor data, user data, or configuration settings. We guarantee that our API is user-friendly, stateless, and uses a client-server communication paradigm by adhering to the REST principles.

In order to communicate with the RESTful API, the ESP32 MCU sends HTTP requests to the designated endpoints. For instance, the ESP32 MCU can send a GET request to the appropriate endpoint to retrieve sensor data. Similar to this, a POST request can be used to provide fresh sensor values to the server. These requests are translated by the REST API, which then utilizes the MariaDB SQL database to carry out the required actions.

Our product delivers a high level of interoperability and scalability by employing a RESTful API. The REST architecture enables autonomous development and modification of each component by permitting the decoupling of the client and server. As the client, the ESP32 MCU has access to the server's resources via well defined and self-descriptive endpoints.

Additionally, RESTful APIs uphold a stateless nature, which means that each client request includes all the data needed for the server to process it. This strategy makes server-side implementation easier and makes scaling and load balancing possible.

Our project enables effective and standardized communication between the ESP32 MCU and the Synology Linux server through the use of HTTP requests and a RESTful API. The

ESP32 MCU can retrieve or send data to the server via a variety of HTTP requests. The HTTP-based RESTful API offers a consistent and standardized interface for navigating and interacting with the server's resources.

In conclusion, HTTP requests and a RESTful API are essential to the success of our project. Our ESP32 MCU and the Synology Linux server can communicate using HTTP queries, and the server's resources may be accessed and changed using a standardized and scalable RESTful API. Together, they make our IoT system's data interchange and control more effective.

2.3.1.4 PHP Scripting

Popular server-side scripting language PHP (Hypertext Preprocessor) is mostly used for web development. It is run on the server while embedded within HTML code to provide dynamic web pages. The ESP32 MCU's HTTP requests are handled by a PHP script in your project, which also communicates with the MariaDB SQL database.

[7]For web development, PHP offers a wide range of features, such as database access, file handling, form processing, and more. The ESP32 MCU sends an HTTP request, which the PHP script receives, parses out the necessary information or parameters, and then uses those to interface with the database.

The PHP script serves as a link between the MariaDB SQL database and the ESP32 MCU in the context of your project. It receives the HTTP request, decodes it, and then uses the database to carry out the required activities. Depending on the information received, this can entail writing data to the database or reading data from it in response to a request.

Numerous database management systems, notably MariaDB (a derivative of MySQL), are supported by PHP. It offers libraries and functions to connect to the database server, run SQL statements, get results, and carry out essential database activities. These functions are probably used by the PHP script in your project to read or write data to the MariaDB SQL database in response to HTTP requests.

2.3.1.5 SQL Database

Relational database management is done using SQL (Structured Query Language). It offers a uniform method for defining, manipulating, and retrieving database data. The data storage backend for your project is the MariaDB SQL database.[8]

Tables are used by SQL databases to organize data, with each table having rows (records) and columns (fields). Through clearly specified relationships, tables can be connected to one another, allowing for effective data storage and retrieval. Using statements like SELECT, INSERT, UPDATE, and DELETE, SQL enables the creation, modification, and querying of databases and tables.

Sensor readings, user data, and any other data that is pertinent to your project can all be stored in the database. For storing and managing the data necessary for your project, a dependable and scalable environment is provided by the MariaDB SQL database operating on the Synology Linux server.

The data that the ESP32 MCU needs to read or modify in the context of your project is stored in the SQL database. By running SQL queries, the PHP script communicates with the database on behalf of the user. Based on the received HTTP requests, these queries either modify or obtain data from the database. The SQL database offers an organized and effective method of managing and storing the data, guaranteeing its availability and integrity.

Overall, the ESP32 MCU and the Synology Linux server can communicate without any issues thanks to a software combination of HTTP requests, PHP programming, and the MariaDB SQL database. The SQL database offers a dependable and structured environment for storing and maintaining the data needed for your project. The HTTP requests permit the interchange of data, the PHP script serves as the intermediary for processing the requests and interacting with the database.

2.3.1.6 NFC/RFID

A RFID antenna on the second microcontroller enables it to scan RFID tags. The data from the tags is collected by this microcontroller, which then transmits it to the other ESP32 using the ESP-NOW communication protocol. RFID tags are employed in our system, which is attached to the injection molds. The RFID tag that is attached to each mold allows for unique identification, allowing the system to recognize the unique characteristics of each mold and adjust the injection time as necessary.[9]

The wireless communication technologies known as NFC and RFID (Radio Frequency Identification) allow for the interchange of data between devices. An integrated circuit and an antenna are the only components of RFID tags, which are tiny electrical gadgets. When in close proximity to an RFID reader or antenna, these tags can wirelessly transmit and store

data.

There are several RFID frequencies available, each with unique benefits and uses[10]. We intentionally selected the 13.56MHz frequency for our project's RFID tags. Applications requiring close-range communication, such NFC-enabled gadgets, frequently use this frequency. It is a fantastic fit for our mold identification system since it strikes a balance between communication range and power usage. Additionally, a variety of RFID/NFC readers and antennas are compatible with the 13.56MHz frequency, giving us versatility and making integration simple.

In our thermoplastic injection process, we have developed a dependable and effective method for mold identification by utilizing the RFID/NFC protocol with the 13.56MHz frequency. We can accurately change the injection duration based on the unique characteristics of each mold thanks to the ESP32 microcontrollers, RFID tags, and antennae, ensuring optimal production outcomes.

	LF	HF	UHF	Active
Frequency	125-134.2 KHz	13.56 MHz	850-960 MHz	100 KHz-2.45GHz
Range	0.2-2m	Up to 1m	Up to 3m	Up to 100m
Cost	3 Euro	0.5 Euro	0.3 Euro	20 Euro
Memory	Typ. 64 bits	Typ. 2048 bits	Typ. 96 bits	Typ. 32 bits
Penetration of Materials	V. Good	Good	Poor	V. Good
Data Rate	Slow	Fast	Fast	Fast
Reader Cost	50-500 Euro	50-3000 Euro	1000-3000 Euro	200-600 Euro
Read Multiple Tags	Poor	Good	Very Good	Good
Applications	Animal Tags, Vehicle Immobilisers, Industrial Applications	Item Tracking, Access Control, Smart Labels	Box and Pallet tracking Some Item Tracking	Industrial Applications. Asset Tagging Location Systems

Table 2.1: RFID Frequencies

2.3.2 Microcontroller

The choice of microcontroller for a project depends on several factors such as performance requirements, power consumption, cost, available resources, and ease of use. In this case, an ESP32 microcontroller has been chosen as the “brains” of the operation for the automation system[11].
The are multiple reasons why this microcontroller is the most suitable choice for the task at hand:

- WiFi connectivity:
  One of the key features of the ESP32 is its built-in WiFi capability. This feature allows the ESP32 to communicate with other devices on the network or access the internet. In this case, the MCU needs to access an SQL database via http requests to obtain the requested info from the server. Furthermore the ability to connect to WiFi is crucial for the future-expandability of the system via OTA firmware updates, thus negating the need for a field technician.

- Dual-core processor:
  The ESP32 is equipped with a dual-core processor, which can help improve performance in multitasking applications. This feature can be beneficial if your project requires running multiple tasks simultaneously, such as accessing the database while also controlling the machine.[12]

- GPIO pins:
  The ESP32 has a large number of General Purpose Input/Output (GPIO) pins, which can be used for various purposes, including driving relays to control the PLC, human interfacing devices such as LCD screens. The number of available pins on the ESP32 eases the burden on the PCB designer as it reduces the complexity of the circuit design.

- Cost:
  Compared to some other microcontrollers, such as the STM32 or Nordic nRF54, the ESP32 is relatively inexpensive.

- Familiarity:
  Being already familiar with the Arduino development environment, using the ESP32 was the obvious choice as the development and testing time is significantly shorter

compared to getting accustomed to a new development platform and using a different MCU development board.

Ultimately, the choice of microcontroller depends on the specific requirements of the project. The ESP32 may be a suitable choice for this automation system based on its features, cost, and ease of use. However, other microcontrollers may also be suitable depending on the project's specific needs.

Chapter 3

Software implementation

3.1 Microcontroller firmware

The automated mold injection production machine's main controller is the first ESP32 code. Its functions include managing the injection process using information from a SQL database and simulating the operation of an 814 PLC timer. The code is written with the Arduino IDE and makes use of a number of libraries and protocols to accomplish its goals.

- Setting up the Required Components and Configurations: The first section of the code consists of initializing the setup. In order to interface with the I2C-based LCD display, it initializes both the Serial communication for debugging and the LiquidCrystal-I2C library. Additionally, it configures the GPIO pins for the relay outputs that control the injection mechanism (RELAY-PIN-A, RELAY-PIN-B, and RELAY-PIN-C) and the PLC signal pin (PLC-SIGNAL-PIN) that detects the trigger from the PLC. It also creates a Wi-Fi connection using the supplied SSID and password, allowing communication with the SQL database.

- The next step in the code is to initialize the ESP-NOW protocol in order to communicate with the second ESP32 board, which is in charge of reading RFID tags. Peer-to-peer communication between ESP32 devices is made possible through the low-power ESP-NOW protocol, which offers quick and dependable data transfer.

- Processing RFID Data: The main loop of the code is constantly monitoring the second ESP32 board for RFID data. The first ESP32 retrieves the NUID (Unique Identifier) of the RFID tag when RFID data is received (onDataRecv function), which serves as

a unique identifier for each mold portion. The NUID and API key are then passed as parameters in an HTTP POST request to the SQL database server in order to retrieve details about the particular mold portion.

- SQL Database Interaction: The code uses the HTTPClient library to send an HTTP POST request to the server in order to communicate with the SQL database. The NUID and API key are sent as request parameters. The request is processed by the server, which then replies in JSON format with the pertinent data regarding the mold part.

- Parsing JSON Data[13]: The Arduino-JSON library is used to parse the JSON data that has been received. The code takes the JSON object's part number, dispensing time, and dispense cycles. The necessary error messages are presented on the LCD panel if the database answer has any issues, such as an invalid API key or a missing entry.

- Process of Injection: The first ESP32 starts the injection process after obtaining the necessary data. It activates the injection mechanism by pressing the relay output buttons (RELAY-PIN-B and RELAY-PIN-C). The LCD panel shows the cycles and duration of the dispensing process for real-time monitoring.

- PLC Interaction[14]: The code communicates with the PLC-SIGNAL-PIN in order to synchronize with the PLC. The first ESP32 sets the signal pin low to mark the start of injection when the injection process starts. The first ESP32 waits for the PLC to set the signal pin high, signaling the end of injection, when the predetermined amount of time has passed. The injection operation is finished after the signal is received and the relay output (RELAY-PIN-A) is turned off.

- Error addressing: The code also has procedures for addressing errors, such as missing database records or erroneous API keys. If these mistakes happen, the code activates the relay output (RELAY-PIN-B), setting off an alert and a buzzer attached to RELAY-PIN-B that beeps.

A comprehensive control system for automating the mold injection production process is shown in the first ESP32 code, which brings up this discussion's conclusion. The code successfully connects with the second ESP32 and the SQL database, retrieves the necessary data, and handles the injection process with accuracy and efficiency by utilizing ESP-NOW, Wi-Fi, and libraries like Arduino-JSON and LiquidCrystal-I2C. It is a reliable and useful

option for automation in mold injection production because it interacts with the PLC and relay outputs to enable smooth integration with current industrial systems.

The second ESP32 code serves as the RFID controller in the automated mold injection production machine. Its primary functionality is to scan RFID tags placed on molds, retrieve the NUID (Unique Identifier) from the tags, and communicate this data to the first ESP32 using the ESP-NOW protocol. The code is written in the Arduino IDE and utilizes the MFRC522 library to interact with the RFID reader.

- Initializing the Setup: The code begins by setting up the necessary components and configurations. It initializes the Serial communication for debugging purposes and sets up the GPIO pins for the RFID reader (SS-PIN and RST-PIN). The ESP32 is then set to WiFi Station mode to establish a connection with the same SSID used by the first ESP32, ensuring they can communicate with each other.

- RFID Scanning: In the main loop, the code constantly scans for RFID tags placed on the molds. When it detects a new RFID tag (rfid.PICC-IsNewCardPresent()), it reads the NUID (rfid.uid.uidByte) from the tag (rfid.PICC-ReadCardSerial()). The NUID serves as a unique identifier for each mold part.

- ESP-NOW Communication: Once the NUID is obtained, the second ESP32 communicates with the first ESP32 using the ESP-NOW protocol. It sends the NUID data to the first ESP32 via esp-now-send(), ensuring fast and reliable data transfer between the two ESP32 boards.

- Error Handling: The code includes error handling routines to handle scenarios where communication with the first ESP32 fails. If the data is not sent successfully (OnDataSent function), the code triggers an alarm by producing a beeping sound using the buzzer connected to BUZZER-PIN.

The two ESP32 controllers cooperate together to automate the mold injection production process. The second ESP32 scans RFID tags on molds and retrieves the NUID of the specific part. It then sends this NUID to the first ESP32 using ESP-NOW communication. The first ESP32, acting as the main controller, receives the NUID and sends an HTTP request to the SQL database to retrieve the required injection parameters for that mold part. Once the necessary data is obtained, the first ESP32 triggers the injection process, controlling the

relay outputs to activate the injection mechanism. The first ESP32 also communicates with the PLC through the PLC-SIGNAL-PIN, ensuring precise synchronization and signaling the end of the injection process. If any errors occur during the process, both ESP32 controllers are equipped with error handling routines to handle and notify the user. Together, these two ESP32 controllers form a seamless and efficient automation system for the mold injection production machine, reducing manual intervention and improving overall production efficiency.

3.2 Front-end UI

The requirements of the project dictated that an interface should be created for the end-user to browse the mold database.

We decided to implement a simple but responsive front-end interface using HTML/CSS. This meant that the development would be relatively simple and the front-end would be able to efficiently communicate with the back-end.

There are separate scripts for each of the actions that the user might request:

- Creating new records
- Editing already-existing records' attributes
- Deleting already-existing records

All the scripts all called from the main index.php script and are exchanging data via POST HTTP requests. The link qith the database is made via the mysqli PHP extension interface[15].

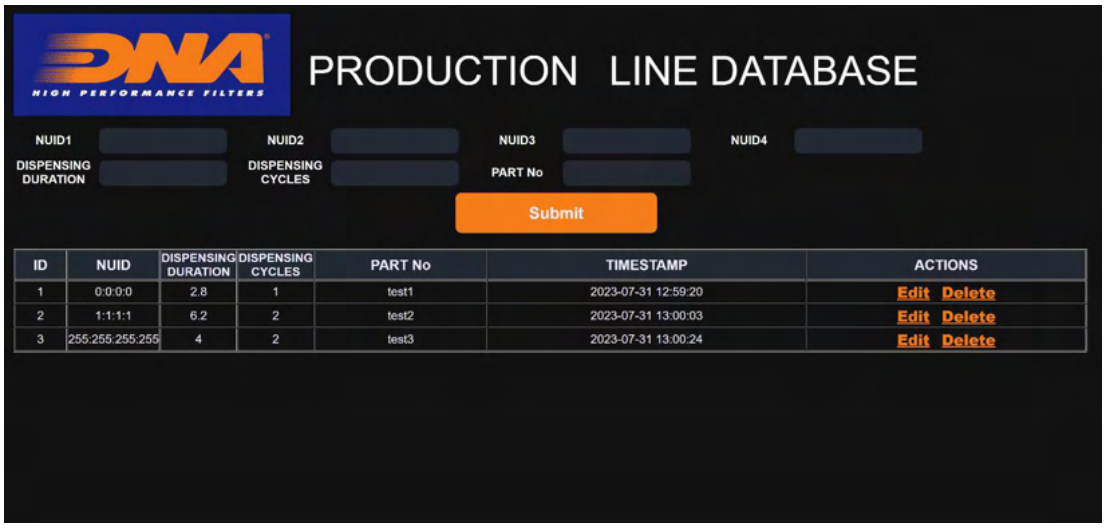


Figure 3.1: Browsing records & creating new records

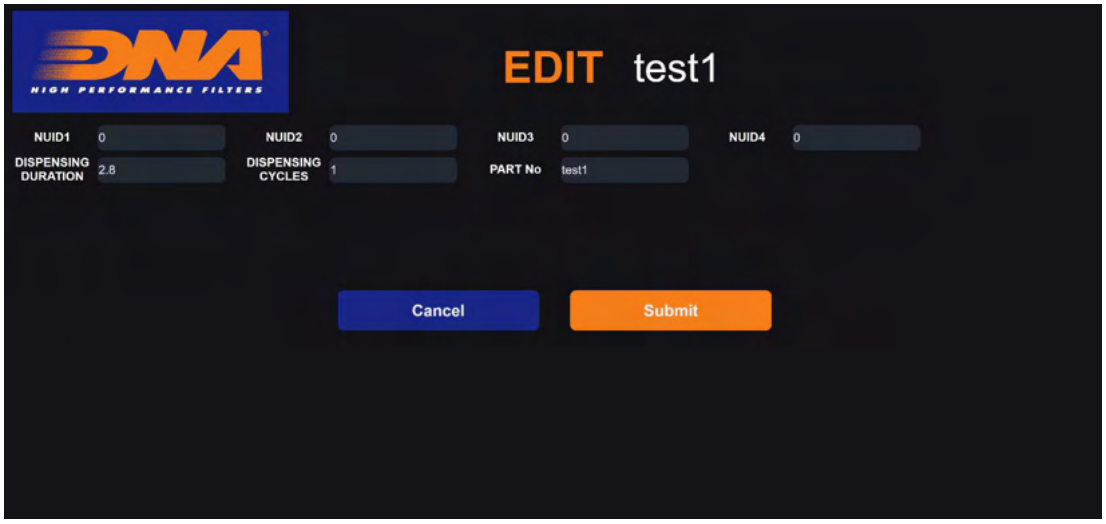


Figure 3.2: Editing existing record's attributes

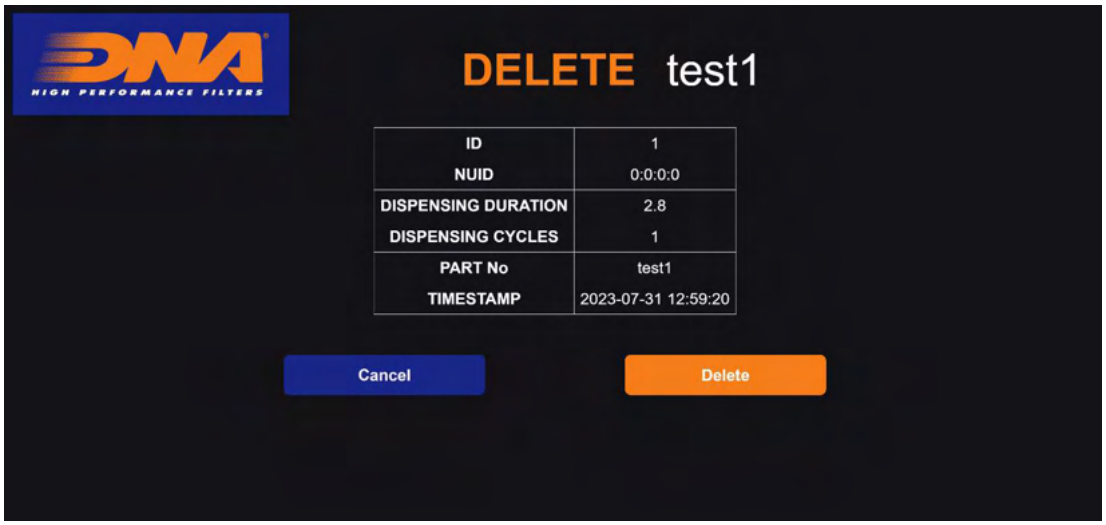


Figure 3.3: Deleting records

Chapter 4

Hardware implementation

4.1 Control panel

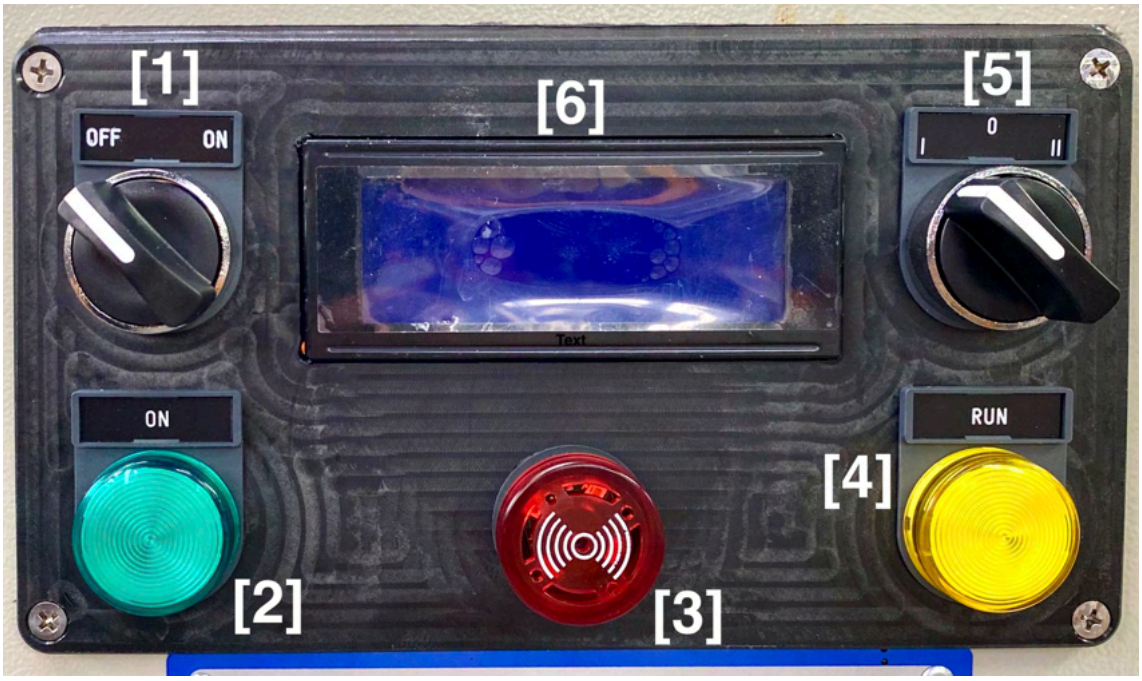


Figure 4.1: The control panel

The main function of the system is to simplify and automate the data input for each operation cycle of the machine. In order to achieve this the system ”mimicks” the functionality of the 814 timer as described in the previous sections. Following the ”automate as-much-as-possible” philosophy of the system, a minimal control panel was created for the user to interact with the system.

Figure 4.2: The control panel (rear view)

1. Main power switch

2. Power on indicator LED

→ Signals that the machine is currently powered on

3. Siren / Alarm indicator LED

→ Informs the user of important events such as start/end of program, successful/unsuccessful scan of RFID tag and any errors that occur.

4. Routine run indicator LED

→ Signals that the machine is currently in operation

5. Operation mode selector (Manual | Off | Automatic)

→ Used by the user to select in which mode will the machine operate.
Manual mode uses the 814 timer and manual data input.
Automatic mode uses the ESP32 microcontroller and the data obtained from the RFID tag - Database routine described in the previous sections. Also used as a fail-safe switch in the occasion of fatal error in the automation system (Network connection outage, hardware failure etc.)

6. LCD display

→ Allows the user to monitor the status of the machine, in conjunction with the alarm siren informs the user of any errors that might have occured.

In normal operation the user has to switch on the power and select the automatic operation mode. The system is designed to initialize in a default startup program state. When the user scans a RFID tag the system retrieves the appropriate information from the database. Upon successful data retrieval from the server, the data is displayed in the control panel LCD display. If the mold injection program is executed correctly the microcontroller logs the production cycle as successful in the database. If the program is not executed as expected the production cycle is flagged for monitoring purposes in the database.

Figure 4.3: Default startup program state

4.2 814 PLC timer operation

In this industrial production line, the PLC is used to control the mixing and dispensing of two chemicals to produce polyurethane. The duration of dispensing of these chemicals is controlled by an 814 timer.

The 814 timer is an electronic device that operates based on a predetermined time interval. In this specific situation, the 814 timer is used to select the duration of the dispensing of the material. The timer is triggered by a signal from the PLC and sends a signal back to the PLC after the set duration has elapsed.

The timer is connected to the PLC through its input/output (I/O) ports. The PLC sends a signal to the timer through one of its output ports to start the timer when the dispensing process is initiated. The timer then starts counting down the predetermined time interval.

When the set duration has elapsed, the timer sends a signal back to the PLC through its input port. This signal indicates that the dispensing process should be stopped, and the PLC then sends a signal to the dispensing system to shut off the flow of the chemicals.

The 814 timer may also have additional features such as the ability to adjust the time interval based on external signals or to trigger other events in the production process. However, in

this specific situation, it is mainly used to control the dispensing of the chemicals based on the weight or volume needed for the specific mold being produced.

4.3 Electronics

For very early testing a breadboard was used. In this stage, research was still ongoing about the most suitable hardware for this use case.

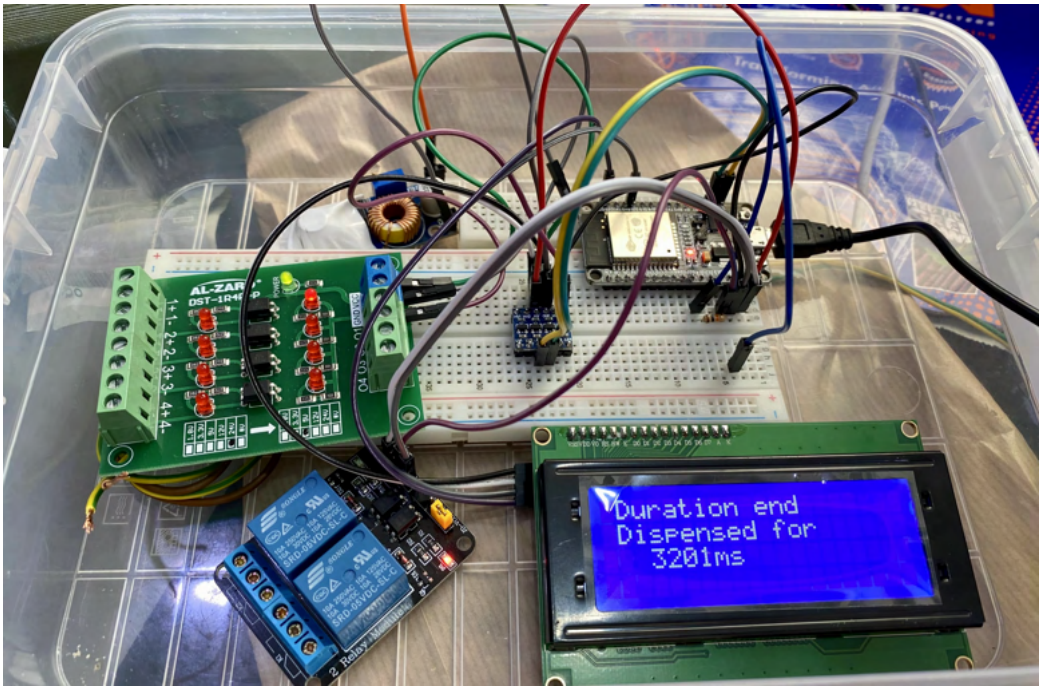


Figure 4.4: Prototype electronics on breadboard

In order to minimize the development time, the first prototypes where constructed using prototyping perfboards.

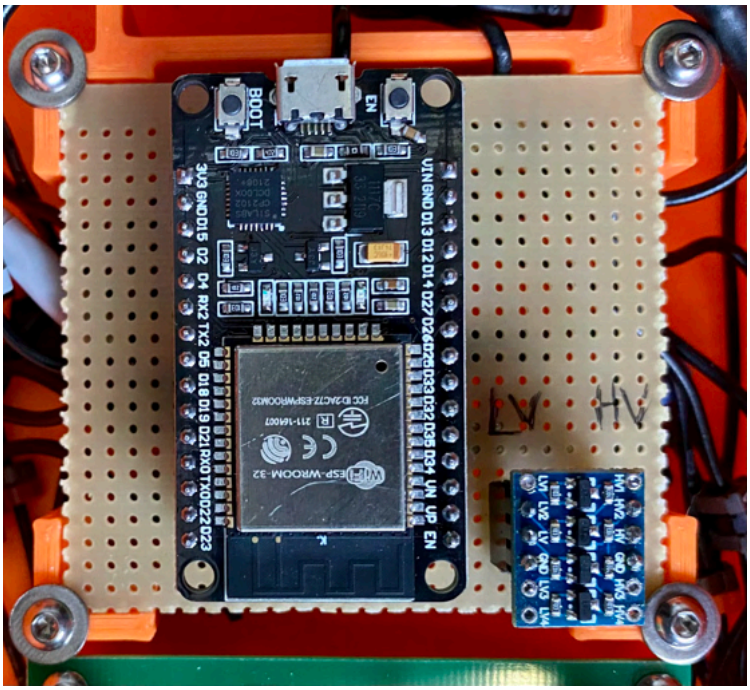
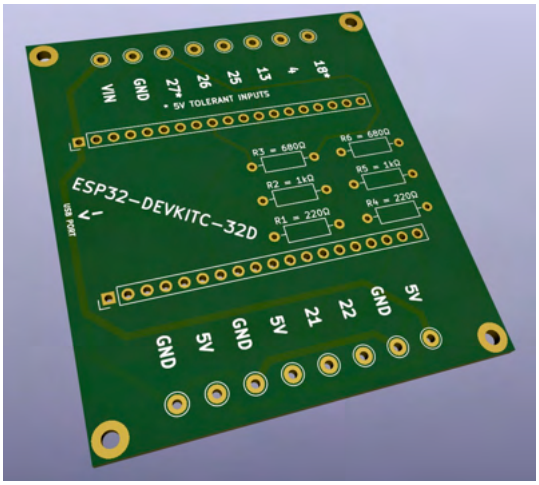


Figure 4.5: Prototype electronics on perfboard

Following the successful tests and pilot installation, the perfboards were upgraded to custom designed PCBs. The PCBs were designed using KiCAD software, chosen due to its ease of use and small learning curve compared to other PCB design software.

In total there are 2 custom PCBs in this system:

1. The first one is designed as a breakout board specifically designed for this application. The PCB is designed to expose the necessary GPIO pins and in some cases use step-down logic (5V to 3.3v) to allow the microcontroller o communicate with the PLC using control signals.



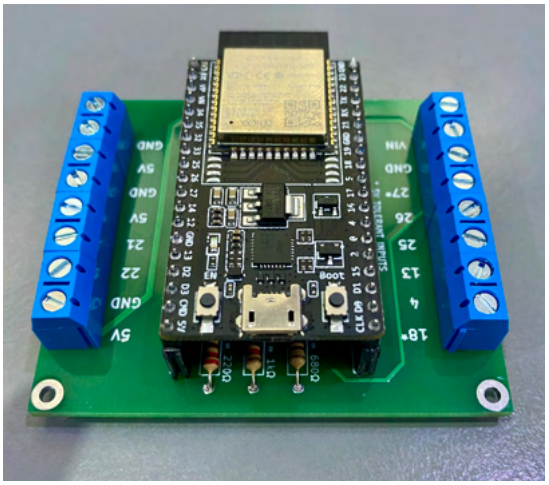


Figure 4.6: (1) Rendered PCB (Left) & Assembled PCB (Right)

2. The second one was created mainly due to space constraints. Allowing the microcontroller and RFID antenna to be placed on opposite sides of the same PCB, the footprint of the assembly is significantly reduced. This results in the final part being more compact and thus less likely to be damaged.

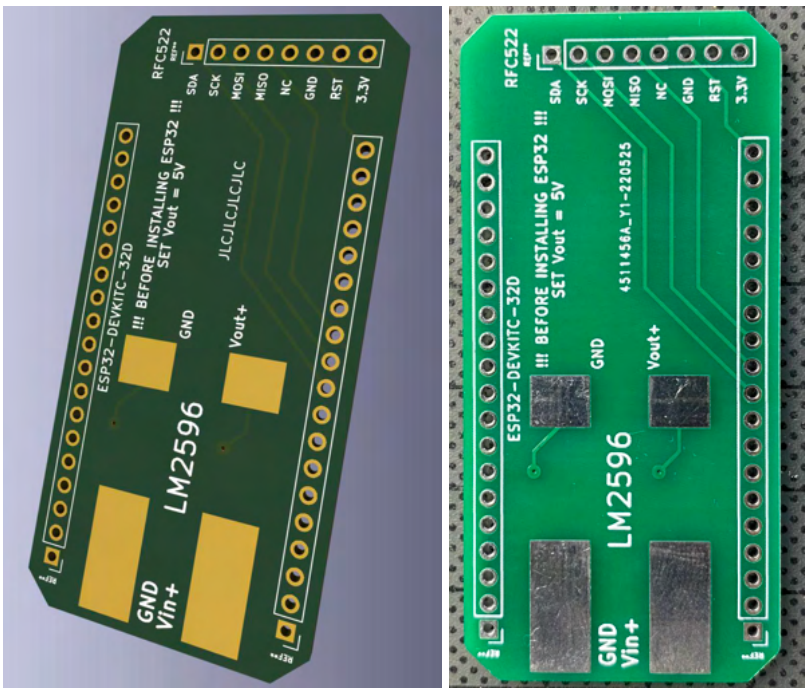


Figure 4.7: (2) Rendered PCB (Left) & Assembled PCB (Right)

For more information regarding tha hardware used in this project and the functionality behind the control panel, please refer to the schematics in appendix B

Chapter 5

Experimental evaluation

5.1 Experimental methodology

5.1.1 Testing theory

The first testing methodology is designed to test the system's improvement as far as the "idle" time is concerned. When the mold preparation procedures have been completed and the main machine operator is ready to commence the injection of a new "batch" of products, the operator begins the machine priming procedure and then starts injecting the first mold, then the second etc. Between each material injection the main operator (or a secondary operator/helper) has to program the specific injection duration (which translates to a specified weight/volume of material calculated at the product research & development stage) in the PLC interface. The time between each injection cycle (completion of 1 mold) is referred to as "idle" time.

To correctly and precisely assess the system's increased productivity we have to control as many variables as possible. The controlled variables are the following:

- Environmental variables
  - Air temperature
  - Air humidity
- Specific mold properties
  - mold temperature (measured with IR thermometer from 50 cm away)
  - release agent application (uniform application film)

  - demolding time

- Specific batch properties
  - Same number of molds
  - Same product part numbers
  - Same sequence of mold injections

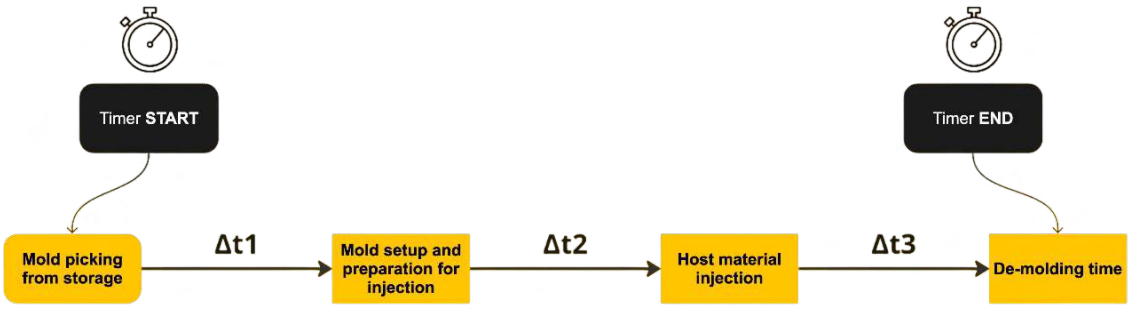
So by carefully selecting a wide array of mold geometries and sizes we can obtain a representative sample of the average batch that goes through the production line every day.

5.1.2 Testing protocol

The timer begins as the operators begin to collect the molds from storage and ends when the last product is removed from the mold.

Included in the duration that is monitored are the below procedures:

- Mold picking from storage
- Mold setup and preparation for injection (the molds need to reach the ideal injection temperature, any complementary parts need to be placed and the release agent spray needs to be applied)
- Material injection
- Demolding time



To accurately assess productivity both with and without the automation we have to repeat the process at least 10 times successfully and without any inconsistencies or interruptions.

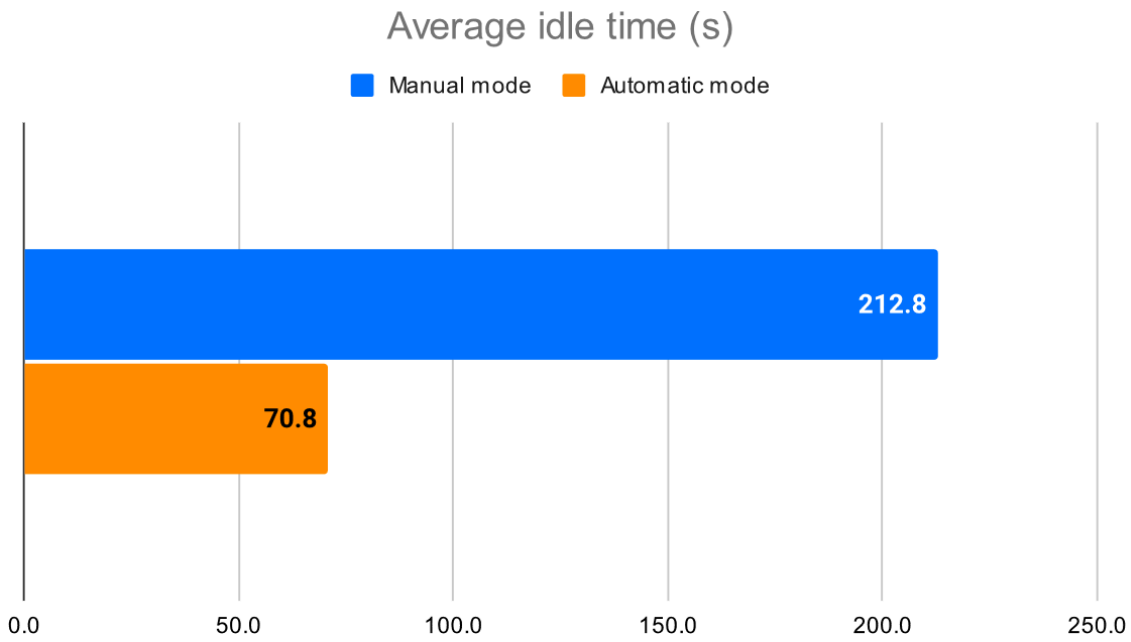
On the production line there are 2 operators: the main machine operator and the secondary operator/helper. In reality, while using manual mode the injection duration is being input in the PLC by either the main or the secondary operator, depending on various factors mainly how the work shifts are created on that certain day (but that is beyond the scope of this project).

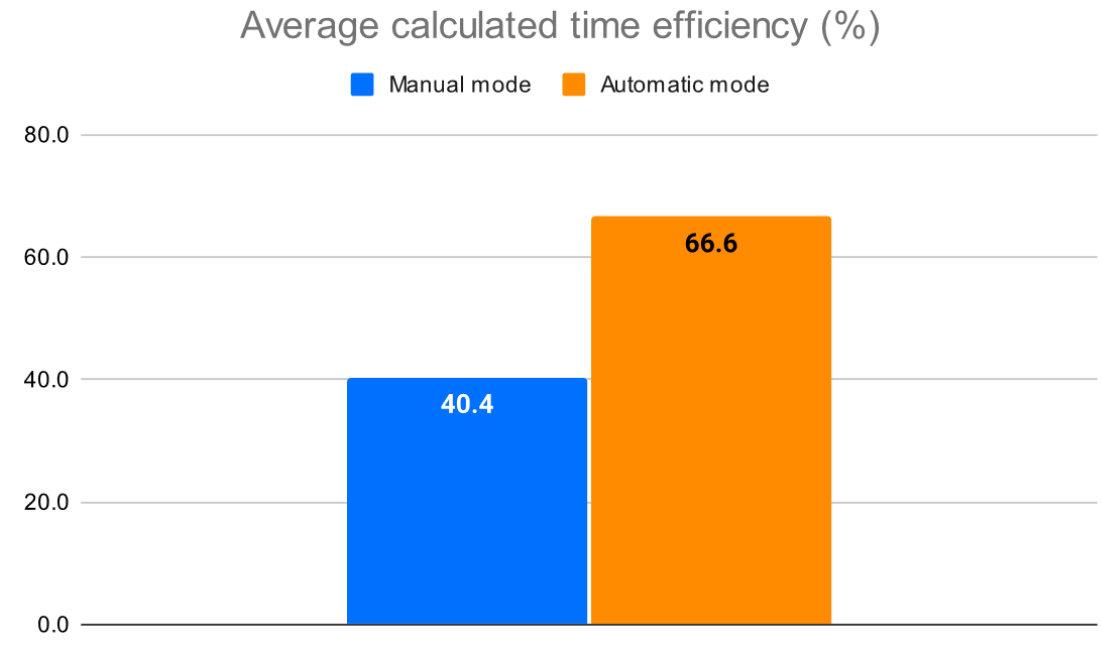
In order to obtain a sample representative of reality, measurements will be taken with the main operator selecting the injection program half the time and the secondary operator the other half. More specifically, 5 batches will be completed with the main operator selecting the program and a further 5 batches will be completed with the secondary operator doing so.

5.1.3 Results and Findings

Mode	Batch	Operator	Injection time (s)	Idle time (s)	Total (s)	Efficiency (%)
M	1	A	141.0	236.7	377.7	37.3
M	2	A	141.0	271.5	412.5	34.2
M	3	A	141.0	234.0	375.0	37.6
M	4	A	141.0	288.2	429.2	32.8
M	5	A	141.0	199.1	340.1	41.5
M	6	B	141.0	161.7	302.7	46.6
M	7	B	141.0	199.5	340.5	41.4
M	8	B	141.0	186.3	327.3	43.1
M	9	B	141.0	184.2	325.2	43.4
M	10	B	141.0	166.5	307.5	45.8
A	11	A	141.0	68.4	209.4	67.3
A	12	A	141.0	73.7	214.7	65.7
A	13	A	141.0	70.2	211.2	66.8
A	14	A	141.0	71.1	212.1	66.5
A	15	A	141.0	69.8	210.8	66.9
A	16	A	141.0	68.9	209.9	67.2
A	17	A	141.0	73.8	214.8	65.6
A	18	A	141.0	67.9	208.9	67.5
A	19	A	141.0	67.5	208.5	67.6
A	20	A	141.0	76.3	217.3	64.9

Average idle time (Manual) (s)	**212.8**
Average idle time (Auto) (s)	**70.8**
Average calculated efficiency (Manual) (%)	**40.4**
Average calculated efficiency (Auto) (%)	**66.6**
Calculated time efficiency improvement (%)	**65.0**





Summing up the above findings:

- We can see a drastic reduction to the idle time of the machine, from a mean value of 212.8 seconds (per batch) to 70.8 seconds (per batch). This in turn translate to massive gains in time efficiency, from 40.4% to 66.6%. An improvement of **64.8%**

Chapter 6

Conclusion

In summary, this Thesis has explored and demonstrated the advantages of automation in a industrial production environment. By utilizing problem-solving techniques and implementing systems that manage simple processes in efficient ways, we have drastically improved the manufacturing process of injection molded parts. We implemented a microcontroller system that is able to directly communicate with a PLC, through identification of each mold we are able to adapt the injection process to each part's unique requirements and thus improve upon the manufacturing process. On the other side, we have implemented a database that holds all the mold profiles and is able to efficiently log the production batches and thus improve the traceability of the products through the production line. All these subsystems can interface with each other seamlessly and effortlessly through advanced protocols and techniques.

6.1 Lessons learned

Although the systems developed proved to have a profound effect on the way the production line operates, there were also many lessons learned. Through extensive testing and research of the already-existing infrastructure we got to the conclusion that the process still largely operates on an human operator-based logic. So apart from implementing automation systems and processes, there is the greater need of better training of the workers and improvement of the workplace environment and culture in order to promote healthier habits among the workers and thus improve the overall efficiency of the workforce.

Another aspect of this project that needs further examining is the expansion possibilities and the requirements of different automation systems in order to be compatible with each

other. The manufacturing process includes many stages both before and after the injection molding of the parts in question, this means that the techniques described in this Thesis may have applications in other stages of the production line as well. Targeting towards the end-goal of a fully automated production line where the operator's skill would have minimal effect in the quality of the end product and thus giving the business the ability to explore new possibilities regarding manufacturing, quality control, packaging and logistics.

6.2 Knowledge acquired during the development

Extensive and invaluable knowledge was acquired during the process of developing and evaluating the IoT production line automation system. Initially, a deep understanding of microcontroller architecture in combination with PLC automation and its benefits was acquired. The development process provided insights into the challenges and considerations involved in wireless data transfer and synchronisation for IoT applications that requires a focus on limiting resource utilization, security, and reliability aspects. Furthermore, by designing and developing specialized PCBs which required various spacial constraints, I learned to operated CAD programs to design PCBs and create schematics for electronics' systems. Also on the UI side, there was a need for developing an interface for the end-user, various WEB technologies and scripting languages were used to acheive this functionality.

The systems needed in each test case entailed lengthy debugging and quality assurance before proceeding to the final evaluation, in order to ensure the validity of the produced results and avoid repetition of time-consuming field testing.

The insights gained from this experience not only contributed to the successful completion of this Thesis, but will also serve as a valuable asset in the pursuit of future endeavors in embedded software and automation engineering.

Bibliography

[1] Jörgen Frohm. *Levels of Automation in production systems*. Chalmers University of Technology Göteborg, 2008.

[2] Abhijit Das and Prakash Mahanwar. A brief discussion on advances in polyurethane applications. *Advanced Industrial and Engineering Polymer Research*, 3(3):93–101, 2020.

[3] Yusuf Abdullahi Badamasi. The working principle of an arduino. In *2014 11th international conference on electronics, computer and computation (ICECCO)*, pages 1–4. IEEE, 2014.

[4] Espressif esp-now: A wireless communication protocol for quick responses and low-power control. `https://www.espressif.com/en/solutions/low-power-solutions/esp-now`.

[5] Redhat enterprise linux: Restful apis. `https://www.redhat.com/en/topics/api/what-is-a-rest-api`.

[6] Amazon web services: Restful apis. `https://aws.amazon.com/what-is/restful-api/`.

[7] Helen J. Burgess. *< ?php > : "Invisible" Code and the Mystique of Web Writing*, pages 167–185. University of Minnesota Press, ned - new edition edition, 2010.

[8] Jay D. Jurie. Structured query language: An instructional tool for public administration. *Public Productivity Management Review*, 15(3):371–380, 1992.

[9] R. Weinstein. Rfid: a technical overview and its application to the enterprise. *IT Professional*, 7(3):27–33, 2005.

[10] Different rfid frequencies and their respective applications. `https://www.advancedmobilegroup.com/blog/whats-the-deal-with-rfid-frequencies`.

[11] Dimosthenis E Bolanakis. A survey of research in microcontroller education. *IEEE Revista Iberoamericana de Tecnologias del Aprendizaje*, 14(2):50–57, 2019.

[12] Marek Babiuch, Petr Foltỳnek, and Pavel Smutnỳ. Using the esp32 microcontroller for data processing. In *2019 20th International Carpathian Control Conference (ICCC)*, pages 1–6. IEEE, 2019.

[13] Dunlu Peng, Lidong Cao, and Wenjie Xu. Using json for data exchanging in web service applications. *Journal of Computational Information Systems*, 7(16):5883–5890, 2011.

[14] Mallikarjun G Hudedmani, RM Umayal, Shiva Kumar Kabberalli, and Raghavendra Hittalamani. Programmable logic controller (plc) in automation. *Advanced Journal of Graduate Research*, 2(1):37–45, 2017.

[15] Mysqli php extension. `https://www.php.net/manual/en/book.mysqli.php`.

APPENDICES

Appendix A

Open & closed molds

- Closed molds

The specific kind of mold utilized in the production process is referred to as a ”closed mold” in the context of thermoplastic injection molding. During the injection molding process, these molds are made to completely envelop and hold the molten thermoplastic host material. They are made of two or more precisely fitted steel or aluminum sections that are divided into a hollow into which plastic is injected.The closed mold design is essential to the efficiency and precision of the injection molding process. It gives the plastic substance a controlled atmosphere, preventing any leaks or spills. The exact alignment and snug fit of the mold halves, which create a sealed hole where the plastic may be injected under high pressure, enable this containment.The closed mold method in thermoplastic injection molding has a number of benefits. It primarily makes it possible to produce elaborate and complex pieces with high precision. Closed molds may produce detailed details, sharp edges, and fine features in the end product because of their precise cavity form and near tolerances. This makes it perfect for producing items that need intricate designs, like consumer goods, electronic components, medical equipment, and automobile parts.Additionally, closed molds guarantee reliable and reproducible production. Since the mold is closed, there won’t be any variations in the end product’s size and shape, guaranteeing constant quality over several manufacturing runs. For sectors that demand exact dimensional accuracy and respect for specifications, this reliability is crucial.Faster production cycles are also made possible by closed molds. The molten plastic can cool quickly in the

enclosed space, hastening the solidification process. As a result, each injection molding cycle's cycle time is shortened, increasing production and reducing lead times.Closed molds also give you more control over the molding process. The mold enables exact control of variables like temperature, pressure, and cooling rates since it encloses the liquid plastic. With the use of this control, manufacturers can tailor their production processes to each unique thermoplastic material, maintaining constant part quality and reducing faults.In conclusion, closed molds are an important part of thermoplastic injection molding. They are built with the precision and containment needed to produce delicate plastic parts, thanks to their design and manufacture. Closed molds are frequently employed in many different industries that depend on high-quality thermoplastic components because of their capacity to produce complex shapes, maintain consistency, and offer control over the manufacturing process.

- Open molds

In thermoplastic injection molding, the term "open molds" refers to a particular kind of mold that is utilized in the production process and differs from closed molds. Open molds, in contrast to closed molds, which entirely encase the molten plastic, have one or more vents or apertures that allow the substance to flow through and fill the mold cavity.When smaller, less intricate thermoplastic parts are needed that do not require the level of precision provided by closed molds, open molds are often used. They are frequently employed in applications where emphasis is placed on effectiveness, economy, and quicker production cycles.Comparatively speaking, open molds have a simpler design than closed molds. They are made up of two or more joined mold parts, which are often constructed of steel or aluminum. Open molds, in contrast to closed molds, have particular channels or runners that permit the plastic substance to flow into the mold cavity. These tubes are intended to direct the flowing plastic melt from the injection unit to the desired areas of the mold.The simplicity and adaptability of open molds are two of their benefits. The presence of vents and apertures makes it simpler to fill the mold with plastic. Additionally, it makes it easier for air or gases to escape during the injection process, which lowers the likelihood of flaws like air bubbles or voids in the finished product. The mold design's simplicity also makes

manufacturing, maintaining, and making any modifications easier and more cost-effective.When speed and cost-efficiency are crucial factors in high-volume production runs, open molds are frequently used. Shorter cycle durations and higher output are produced as a result of the open design's ability to facilitate quicker filling and cooling times. As a result, open molds are appropriate for uses like packaging, throwaway items, and other goods that call for quick production and a low degree of part complexity.It is crucial to remember that, compared to closed molds, open molds may be less able to produce precise details, strict tolerances, and dimensional precision. Vents and apertures may restrict the design of the mold and increase the risk of flash development (extra material seeping out of the mold). Therefore, open molds might not be appropriate for components that need fine details or extremely accurate proportions.In conclusion, open molds in thermoplastic injection molding offer a more straightforward and affordable method for creating less complicated parts. They have benefits like simplicity of usage, adaptability, and quicker manufacturing cycles. Open molds are ideal for applications where efficiency, high volume production, and cost effectiveness are valued more than the ability to produce complex details and precision.

Appendix B

Schematics

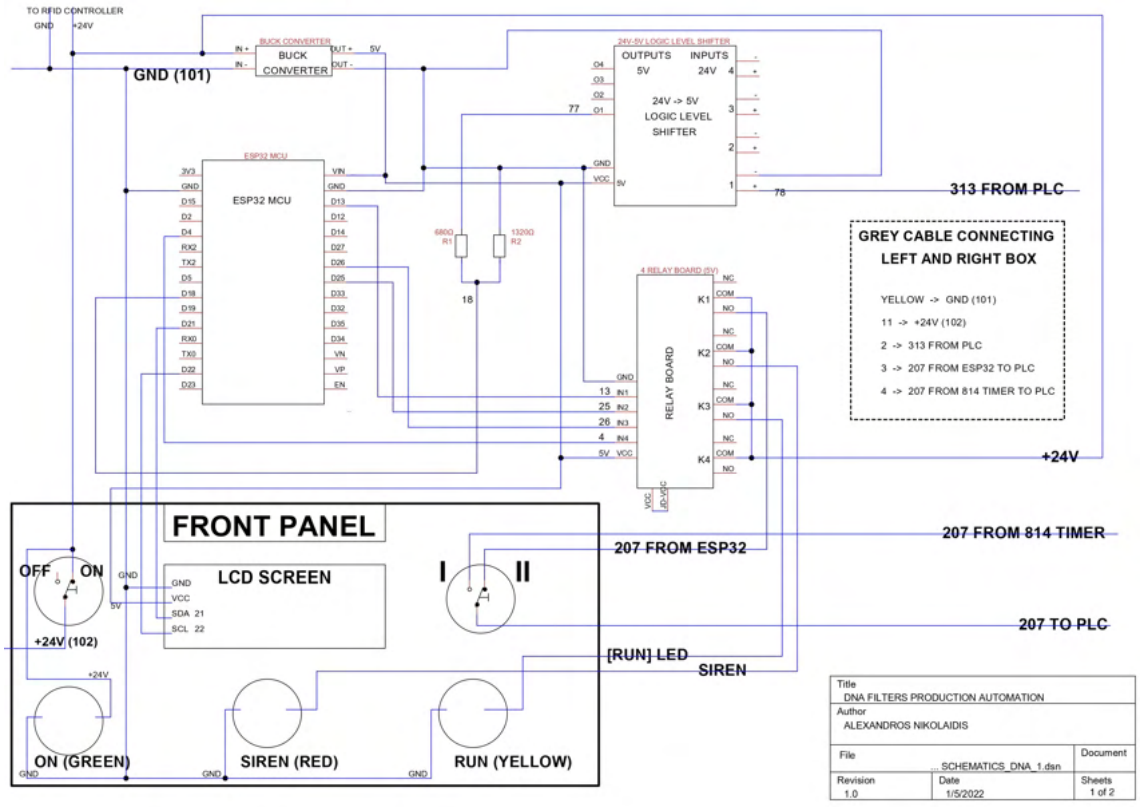


Figure B.1: Schematic 1 of 2

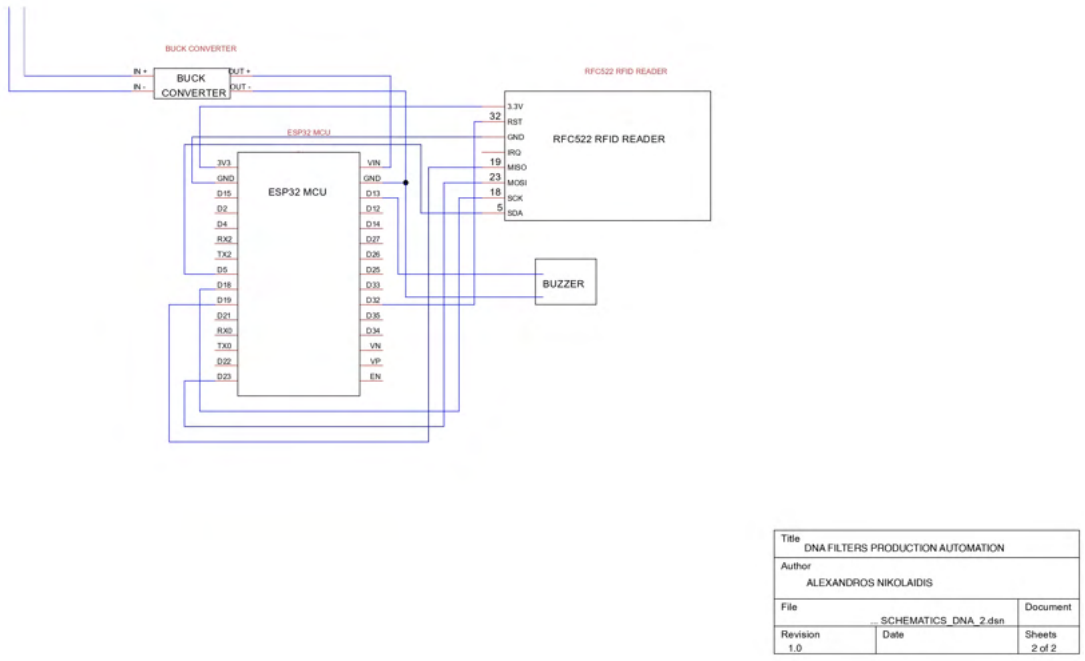


Figure B.2: Schematic 2 of 2

Appendix C

Experimental evaluation tables

Mode	Batch	Operator	Injection time(s)	Idle time(s)	Total(s)	Time efficiency(%)
M	1	A	141.0	236.7	377.7	37.3
M	2	A	141.0	271.5	412.5	34.2
M	3	A	141.0	234.0	375.0	37.6
M	4	A	141.0	288.2	429.2	32.8
M	5	A	141.0	199.1	340.1	41.5
M	6	B	141.0	161.7	302.7	46.6
M	7	B	141.0	199.5	340.5	41.4
M	8	B	141.0	186.3	327.3	43.1
M	9	B	141.0	184.2	325.2	43.4
M	10	B	141.0	166.5	307.5	45.8
A	11	A	141.0	68.4	209.4	67.3
A	12	A	141.0	73.7	214.7	65.7
A	13	A	141.0	70.2	211.2	66.8
A	14	A	141.0	71.1	212.1	66.5
A	15	A	141.0	69.8	210.8	66.9
A	16	A	141.0	68.9	209.9	67.2
A	17	A	141.0	73.8	214.8	65.6
A	18	A	141.0	67.9	208.9	67.5
A	19	A	141.0	67.5	208.5	67.6
A	20	A	141.0	76.3	217.3	64.9

Table C.1: Experimental evaluation

Average idle time (Manual) (s)	212.8
Average idle time (Auto) (s)	70.8
Average calculated efficiency (Manual) (%)	40.4
Average calculated efficiency (Auto) (%)	66.6
Calculated efficiency improvement (%)	65.0

Table C.2: Evaluation results