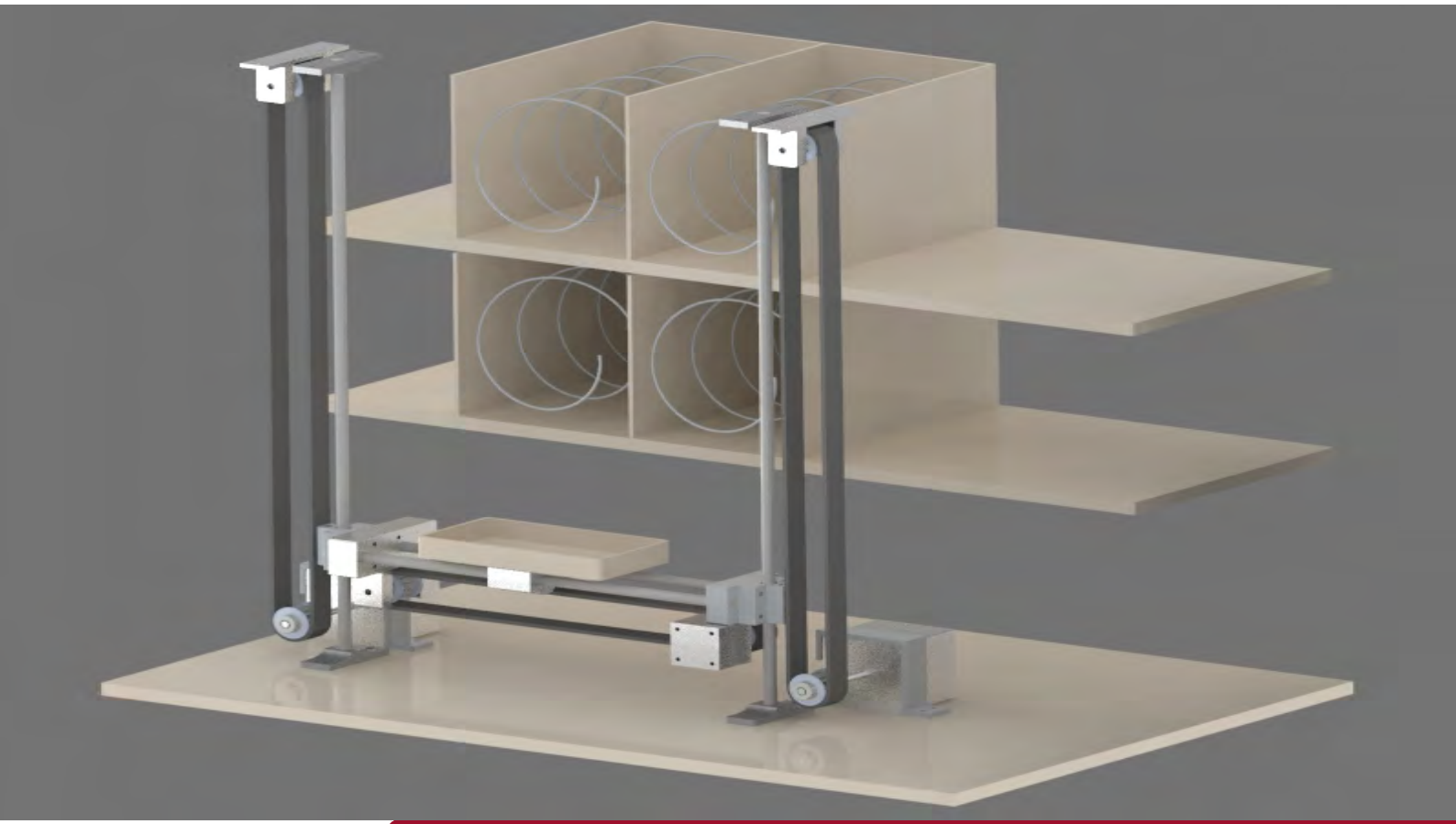
DEGREE PROJECT IN MECHANICAL ENGINEERING,
FIRST CYCLE, 15 CREDITS
STOCKHOLM, SWEDEN 2021

The Apotekomat

An Autonomous Pharmacy

EHAB ZIAD RAHEEM

JIAHAO WANG



KTH ROYAL INSTITUTE OF TECHNOLOGY
SCHOOL OF INDUSTRIAL ENGINEERING AND MANAGEMENT

The Apotekomat

An Autonomous Pharmacy

EHAB ZIAD RAHEEM
JIAHAO WANG

Bachelor's Thesis at ITM
Supervisor: Nihad Subasic
Examiner: Nihad Subasic

TRITA-ITM-EX 2021:30

Abstract

The purpose of this project was to examine the possibilities of automating pharmacies by developing a vending machine-like device. As a result, a prototype was designed, constructed and programmed. The main focus of this prototype is to inspect whether it is possible to minimize the need for staff in pharmacies.

In order for the prototype to archive it's purpose, a various amount of components were used. Those components were connected and controlled using the Arduino Mega. A screen and a keypad were used to facilitate the user interaction with the prototype. Furthermore, three different types of motors were needed to deliver medicine to the user.

The final prototype was able to deliver medicine to the user from four different boxes. Various experiments were carried out to ensure that the machine could perform the required tasks and automate medicine delivery process in an effective way.

Keywords: Mechatronics, Microcontroller, Autonomous, Pharmacy, Vending machine.

Referat

Apotekomaten
Ett automatiserat apotek

Syftet med detta projekt är att studera möjligheten till apoteksautomation genom att utveckla en enhet som liknar en varuautomat. Genom att designa, konstruera och programmera uppförs en prototyp. Huvudfokus för denna prototyp är att kontrollera om det är möjligt att minimera personalbehovet på apoteket.

För att prototypen ska uppnå sitt syfte används ett stort antal komponenter. En skärm och ett tangentbord används för att låta användare interagera med prototypen. Dessutom krävs tre olika typer av motorer för att tillhandahålla läkemedel till användarna.

Den slutliga prototypen kan förse användare med läkemedel från fyra olika rutor. Olika experiment genomfördes för att säkerställa att maskinen kan utföra de önskade uppgifterna och att den kan effektivt automatisera leveransen av olika mediciner.

Nyckelord: Mekatronik, Mikrokontroller, Automatisering, Apotek, Varuautomat.

Acknowledgements

We would like to thank:
Nihad Subasic, our supervisor and examiner, for his lectures and guidance during the project.
Staffan Qvarnström, a teacher at the Royal Institute of Technology (KTH), for his guidance and help choosing components.
Jan Stamer, a technician at KTH, for his advice regarding component design.
Amir Avdic, a teacher assistant, for his help operating machines to design mechanichal parts.
At last we would like to express our gratitude to our classmates for their help and support during the project.

Contents

List of Figures

List of Tables

List of Abbreviations

$3D$	Three dimensional
AC	Alternating current
CAD	Computer-aided design
CNC	Computer numerical control
CPU	Central processing unit
DC	Direct current
I/O	Input/Output
ID	Identification
KTH	Royal Institute of Technology
LCD	Liquid crystal display
PWM	Pulse Width Modulation
RAM	Random-access memory
ROM	Read-only memory
USB	Universal serial bus

Chapter 1

Introduction

1.1 Background

The process of delivering medicine to people in need has long been the same, it must be done by specialized pharmacists in pharmacies through direct contact with the patient. That means, getting the medicine that people need depends only on the staff in the pharmacy which raises the following questions: How much pressure can be put on pharmacists during work? How much staff is needed in pharmacies? Having too much staff can cause economic problems, on the other hand inadequate staffing will lead to stressed employees, who are more likely to make mistakes and give the wrong prescription to the wrong person. This dilemma can be resolved by automating the process which can be done by using an autonomous pharmacy that works in the same way as a vending machine, where the person only needs an identification (ID) card and a payment method to get the medicine that is needed. This could take the pressure off the pharmacists and makes it faster and easier for people to get their medicine.

1.2 Purpose

This project aims to study the possibility of automating pharmacies and delivering medicines to consumers by building a prototype that uses a combination of electrical and mechanical components. The following research questions will be discussed during the project:

- What construction performs the best in terms of storing and delivering medicine?
- How effective can The Apotekomat automate the medicine delivery process?
- How accurate is the system?

1.3 Scope

The goal of this project is to build an automated delivery mechanism that can be used to get medicine without the need for staff in pharmacies. However, due to the limited time and the small preliminary budget, the initial machine will only contain

4 to 6 different medications. Due to the technical limitations in the early stage, the machine can only complete the most basic target requirements, the purpose is to test the feasibility of the machine. Later, more types of medicine can be added on this basis to achieve the actual effect that can be used in future work.

1.4 Method

To be able to build a prototype of a vending machine that automates medicine delivery, the following steps were performed to achieve the purpose. The project was mainly divided into two main parts, construction and programming. The first step of the construction was to understand the general structure of the machine, so it was necessary to use a computer-aided design (CAD) software to simulate the construction of the machine, which increased the tolerance of errors for the real construction. Afterward, the type of motors and sensors were decided for the project to meet the corresponding requirements. Of course, the choice of components and materials was also taken into consideration. When the construction of the prototype was designed and the components were chosen, a code was written in the programming language C to make the system work as intended.

Chapter 2

Theory

2.1 Vending Device

All types of vending machines contain some sort of vending device that is used to transport products to the user. Vending devices comprise of a spiral coil connected to a rotation mechanism controlled by some type of a payment method. Once a payment is completed the coil starts to rotate, and the product is moved forward until it drops from the spiral to some type of transporting mechanism that carries the product to the purchaser [1].

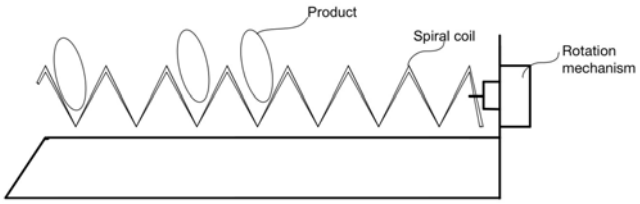


Figure 2.1: A principle figure of a vending device, made with Notability [2]

2.2 Microcontroller

Microcontrollers are devices that were initially developed to be used in embedded electronic control systems, which makes them the perfect choice when it comes to controlling electrical gadgets such as motors. Microcontrollers consist mainly of a central processing unit (CPU), Input/Output (I/O) ports and memory. The CPU can be described as the brain of the microcontroller where all the computation is performed, while the (I/O) ports provide digital communication between the external components and the microcontroller. In every microcontroller, there are two types of memory, a read-only memory (ROM) that is a memory for the program and it is nonvolatile, which means that it does not get erased when the power is removed unlike the random-access memory (RAM) which is used to store data and it is often volatile [3].

2.3 DC Motor

A direct current (DC) motor is an electric component that uses electricity to generate mechanical energy. It is mainly composed of three components, rotor, stator, and commutator. Usually, the rotor lies inside of the motor and it consists of coil windings, while the stator, which normally contains a permanent magnet that can be found outside of the motor. When the DC current powers the motor, it creates a magnetic field in the stator, repelling and attracting the magnets on the rotor which causes it to start rotating. The commutator retains the rotation of the rotor by reversing the current through the stator which reverses the magnetic field causing continuous rotation in the rotor [4].

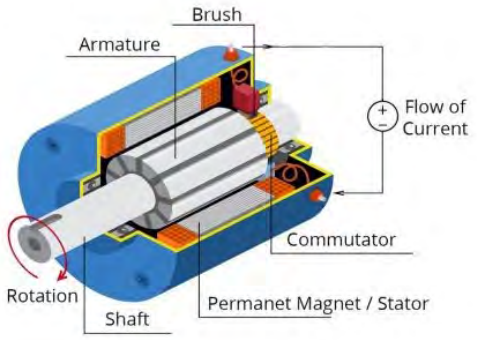
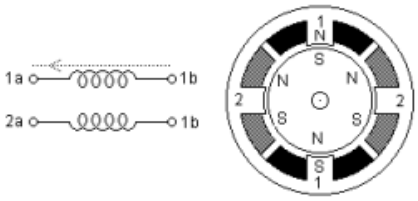


Figure 2.2: A DC motor internal construction [5]

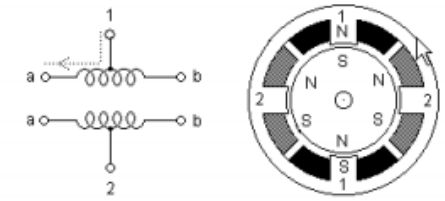
2.3.1 Stepper Motor

A stepper motor is a DC motor that rotates with discrete steps based on the electromagnet principle. Stepper motors are widely used in three dimensional (3D) printers and computer numerical control (CNC) machines that require exact controls on speed and position. A stepper motor is controlled by applying pulses of DC electricity to its internal coils. The stepper motor has a magnetized gear core surrounded by coils, and by precisely controlling the electric current on the coil, the motor shaft can be made to move. Depending on the methods used to control the magnetic field inside the motor, there are two types of stepper motors, bipolar and unipolar [6].

The bipolar stepper motor, as shown in Figure 2.3a, consists of a shaft and two coils where each coil has two connections. The shaft, which is a permanent magnet, is attracted to the energized coil. This means that activating the coils alternately by applying current to one coil at a time makes the shaft rotate. During operation, bipolar stepper motors have higher efficiency and generate more torque than unipolar motors of comparable scale, but require more complex controller to control it's speed and direction. Simultaneously, the unipolar stepper motor also includes two coils, but it is built differently. In addition to two connections, each coil also



(a) Bipolar stepper motor



(b) Unipolar stepper motor

Figure 2.3: Stepper motors winding connections [7]

consists of a center tap, see Figure 2.3b. These two center taps can either be taken outside the motor as two separate wires or connected internally, which means that the unipolar stepper motor can have five or six connections. The advantage of unipolar stepper motors is that they are easier to control but less efficient than bipolar motors [7].

2.3.2 Servo Motor

A servo motor is a DC motor that uses feedback based on information from a sensor or sensors and external circuitry to correct the motor's output. The servo motor consists of a DC motor, a potentiometer, and a control circuit. The servo motor is a very energy-efficient device controlled by electrical impulses, as shown in Figure 2.4. The motor is connected to gears, which in turn control the shaft. The operating principle of the servo motor can be briefly summarized as follows. When the motor's shaft is not in the required position, it will move in the appropriate direction; the opposite, if the shaft reaches the specified position, the power supply will stop. Servo motors can be divided into two types: alternating current (AC) and DC servo motors. AC servo motor can work at relatively high currents and is usually used in the industry. On the other hand, the DC servo motor is suitable for working under a small current and is generally used in small applications. Apart from this, there are also servos which work for continuous rotations [8].

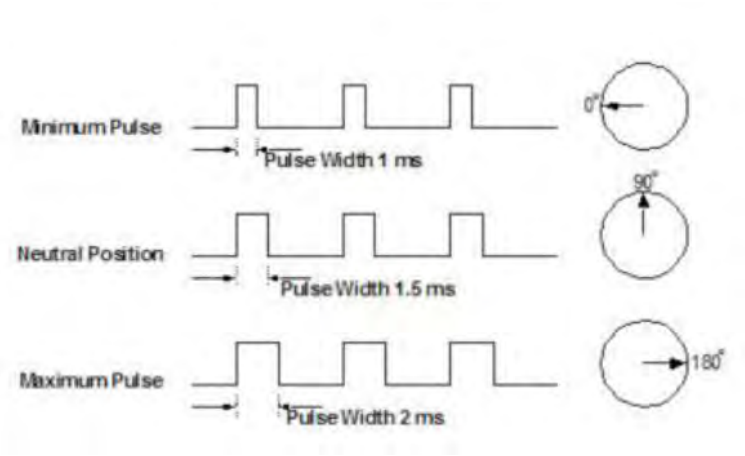


Figure 2.4: Servo motor's controlling pulses [8]

2.4 H-bridge

To control the stepper motor and reverse the direction of rotation, an H-bridge needs to be used. The H-bridge is a simple electric circuit that contains transistors which are used as switches to control the polarity of a voltage. Controlled by an input signal from a microcontroller, these switches can be used to change the direction of a current through a stepper motor which changes the direction of the rotation. As shown in Figure 2.5, the idea is to connect two switches to each side of the motor, one switches the ground connection and the other switches the positive voltage to the motor. By switching on S1 and S4 the motor starts to spin in one direction, on the other hand switching on S2 and S3 will make the motor spin in the opposite direction [9].

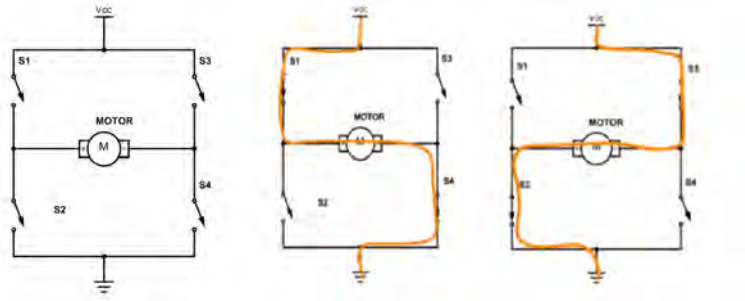


Figure 2.5: Schematic drawing of an H-bridge [10]

2.4.1 DRV8825 Stepper Motor Driver

For printers, scanners, and other electronic machinery applications, the DRV8825 driver, shown in Figure 2.6, offers an optimized motor driver solution. The gadget is designed to drive a bipolar stepper motor and includes two H-bridge drivers, a

micro-stepping indexer, an output current up to 2.4 A and it can control bipolar stepper motors in full 1/2, 1/4, 1/8, 1/16 and 1/32-step modes [11].

The stepper motor's speed is regulated by the driver, this can be done by changing the length of time each coil is switched on and off, this indicates that short periods leads to faster motor rotation, because then the magnet will be aligned with the coil for a shorter amount of time. Since the driver processor, not the Arduino, is in charge of this, the microcontroller can be freed up to perform other tasks [12].

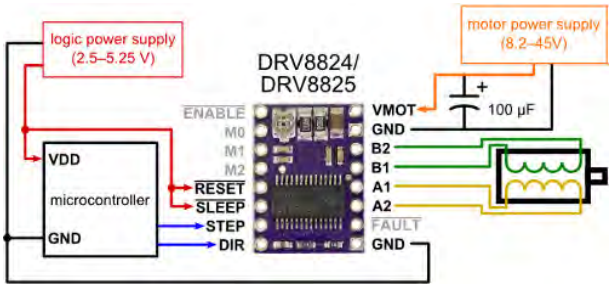


Figure 2.6: DRV8825 stepper motor driver [13]

Chapter 3

Demonstrator

This chapter introduces the problem and describes how the components were connected and built together to demonstrate a fully functional prototype.

3.1 Problem Formulation

The main difference between The Apotekomat and a vending machine is that this project was designed to store medicine and deliver it only to people who are authorized to possess them. Meaning that in addition to having a transporting mechanism the prototype should also contain an input technique that lets it know if the user is authorized to have this kind of medicine. Primarily the development of the prototype aims to fulfill three essential tasks:

- Asking the user to input the right code for the medicine.
- Checking if the code fits the medicines in storage.
- Transporting the medicine to the user.

3.2 Hardware and Mechanical Parts

The shape of The Apotekomat was chosen as a cuboid box which was made of an eight millimeters thick wooden board as the shell material. The dimensions of the box are 60 centimeters in length, 40 centimeters in width, and 50 centimeters in height. The box was not sealed from behind to make installation and reparation of the interior hardware easier. The other five sides of the cuboid were covered with a wooden board, as shown in Figure 3.1 and Figure 3.2.

Figure 3.1: Overall picture of The Apotekomat, made with Solid Edge [14]

Figure 3.2: Overall picture of The Apotekomat from the backside, made with Solid Edge [14]

3.2.1 The Internal Frame

The internal frame of the box was also mainly composed of wooden boards and it was divided into three levels. Each level was separated by an eight millimeters thick shelf, as shown in Figure 3.3. The shelves have four shelf pins to hold them in place, two on each side. The top and middle levels were used to store the product boxes

which were placed closer to the left wall. The screen and keypad were positioned on the right side of the cuboid. The stepper motor and pick-up plate were placed at the bottom level.

There are four product boxes inside the shell, arranged in two rows displayed as number one in Figure 3.3. Opposite to each box is a vertical iron rod with a rubber band parallel to it, shown in Figure 3.3 as number two. The length of the iron rod is the same as the height of the shell. The rods were held in position by 3D-printed attachments which screwed into the top and bottom of the shell, see number three in Figure 3.3. One end of the rubber band was wound to a gear that was attached to the top of the box by a 3D-printed attachment. The other end of the rubber band was wound to a gear that was attached to a stepper motor at the bottom of the box, shown in Figure 3.3 as number four. Each side of the metal rods was connected to a bearing with a 3D-printed attachment on it. A 3D-printed attachment was also placed on the opposite side of the bearing to connect it to the rubber band so that motion of the rubber band triggers motion of the bearing, as shown in Figure 3.4.

Two horizontal hollow rods were placed between the vertical iron rods. These hollow rods were connected to the vertical rods with a 3D-printed attachment and a bearing. A plate was placed on top of the horizontal rods to catch the falling products. Two bearings were used to hold the plate onto the horizontal rods.

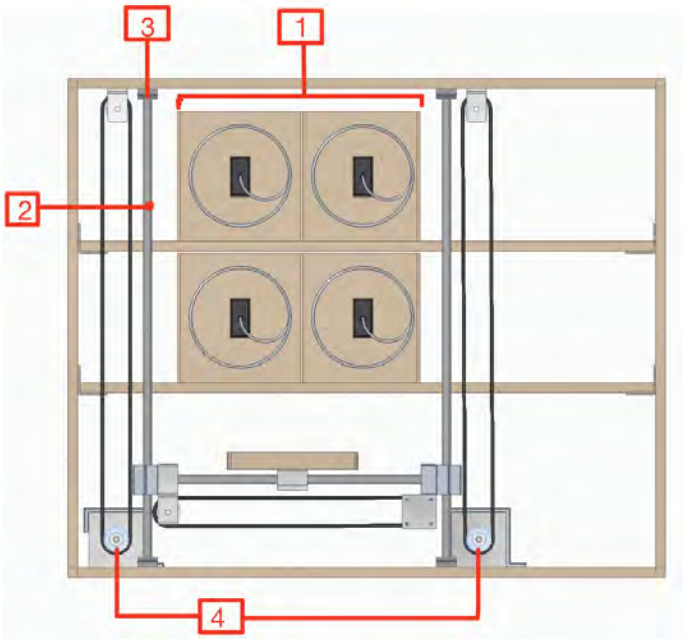


Figure 3.3: The internal frame of The Apotekomat, made with Solid Edge [14] and edited using Notability [2]

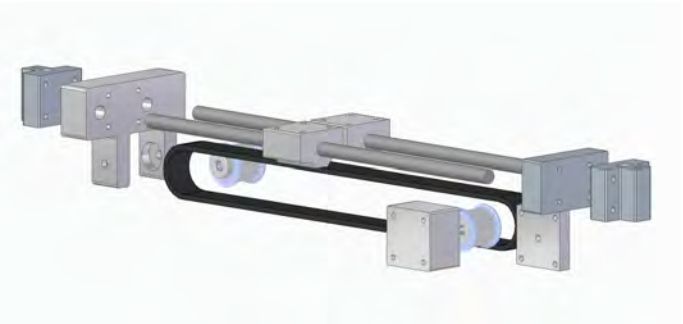


Figure 3.4: Exploded view of the delivering system, made with Solid Edge[14]

3.2.2 Products Boxes

The product box consists of 13x13x26 centimeters wooden boards with a rectangular hole at the end of the box for the servo motor which was connected to the spiral, as shown in Figure 3.5. The spiral was made of a steel wire with a thickness of two millimeters and the diameter of the spiral was chosen as twelve centimeters.

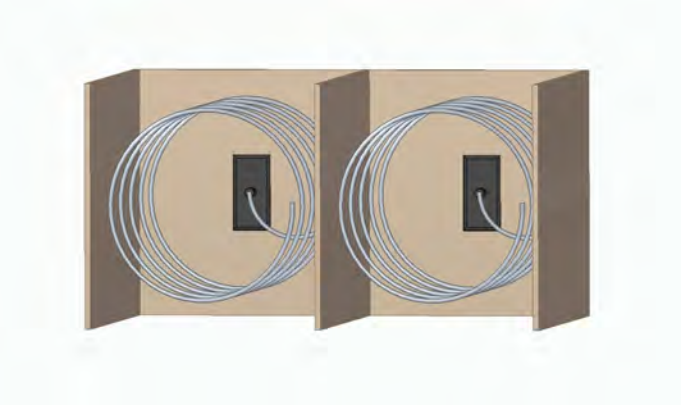


Figure 3.5: Products boxes arranged in row, made with Solid Edge [14]

3.2.3 The Operation Principle

The operation principle of The Apotekomat was mainly divided into two parts, vertical movement and horizontal movement. The vertical movement relies on two large stepper motors located at the bottom of the box, see Figure 3.6, to provide torque. The rotation of the motors leads to the rotation of the rubber band hence up and down motion.

The principle of horizontal movement was the same as vertical movement. The difference was the use of a small stepper motor that rotates the rubber band hence the left and right motion.

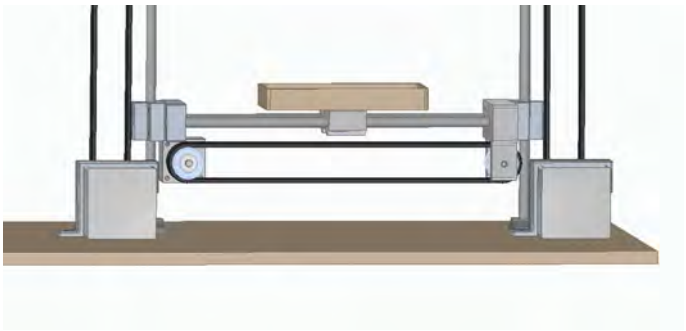


Figure 3.6: The construction of the delivering mechanism, made with Solid Edge [14]

3.3 Electronics

The electrical components that were used in this project are shown in Figure 3.7. Stepper motors were used to transport the medicine horizontally and vertically, while servo motors were used to rotate the spiral coil in the vending device. To display the instructions to the user a liquid crystal display (LCD) screen was added to the prototype, in addition to that a keypad was used as an input mechanism. All those components were connected to the Arduino Mega that was programmed to accomplish different tasks.

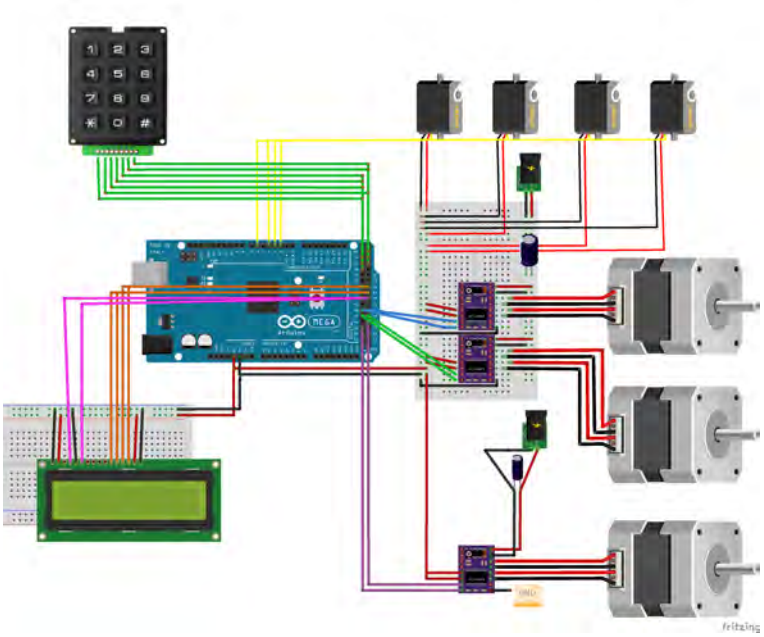


Figure 3.7: Circuit schematic, made in Fritzing [15] and edited using Paint [16]

3.3.1 Arduino Mega 2560 REV3

The microcontroller board used in this project is ARDUINO MEGA 2560 REV3, see Figure 3.8. The Arduino Mega is based on the ATmega2560 which is an 8-bit microcontroller with 16/32/64KB In-System Programmable Flash. It has an input limit of voltages in the range of 6-20V. The board contains 54 digital inputs/outputs ports of which 15 can provide pulse width modulation (PWM) outputs, in addition to that it has 16 analog input pins and can easily connect to a computer with a universal serial bus (USB) cable [17]. As a result of using a lot of components for this project, a significant amount of (I/O) pins were needed which made the Arduino Mega a perfect choice in this project.

Figure 3.8: Arduino Mega 2560 REV3 [17]

3.3.2 Parallax Continuous Rotation Servo Motor

To rotate the spiral coil in the vending device a continuous rotation servo motor, shown in Figure 3.9 was used. Both speed and direction of the servo motor were adjusted by using PWM. The motor has the ability to rotate continuously in both directions, moreover it has a maximum output torque of 27×10^{-2} Nm which is enough to rotate the spiral coil. The motor usually needs to be connected to an operational amplifier to be able to handle bigger loads. However, due to the small load tested in this project the motor was directly powered by the Arduino's 5V power supply [18].

Figure 3.9: Parallax Continuous Rotation Servo Motor [18]

3.3.3 KH56-801 Stepper Motor

To move the plate that delivers medicines in the vertical direction two KH56-801 stepper motors were used, one on each side. These are unipolar stepper motor but were used as bipolar by leaving the center tap in each phase disconnected.
The KH56-801 motor, see Figure 3.10, is a two-phase 2.4V stepper motor that has a holding torque of 833×10^{-3} Nm which makes it able to overcome the gravitation force and the friction force between the linear bearings and the vertical bars. The full specification of the motor can be found in Appendix A.

Figure 3.10: KH56-801 Stepper Motor [19]

3.3.4 TS3214 N61 Stepper Motor

Due to the minor friction force in the horizontal direction there was no need to use a powerful stepper motor in that direction, instead the 5V TS3214 stepper motor was used. In this motor a full rotation is accomplished by 200 steps, meaning that the motor's axis rotates 1,8 degrees per step. This information can then be used to estimate the number of steps required to cover the target distance [20].

3.3.5 LCD Screen

The LCD screen, shown in Figure 3.11, was used to display information to the user. The ability to show 16 characters made this screen a good choice to show simple instructions such as telling the user to enter a four-digit code and displaying the name of the medicine that is being delivered.

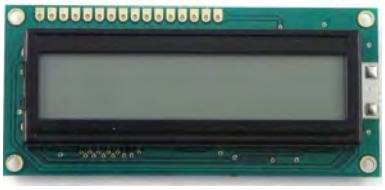


Figure 3.11: LCD 2×16 characters TN LED [21]

3.3.6 MCU Membrane Switch Keypad

An input method was needed to allow the user to interact with the microcontroller. For this purpose a 4x3 matrix keypad, see Figure 3.12, was added to the project. It offers the numbers 0 through 9, as well as the regular star(*) and hash symbols.

Figure 3.12: 12 Key MCU Membrane Switch Keypad [22]

3.4 Software and User Interface

In order to perform tasks that were mentioned in section 3.1 some type of input from the user was needed in the program. The program starts by showing a message on the LCD screen which tells the user to write a four-digit code using the keypad. When the code is entered the program will go through a hashtable which is a Key-Value store method where the four digit codes are included as keys and the medicines' names are stored as values.

If the code that was inserted by the user is available in the hashtable, the user

will be asked to enter a password and if the password matches the code entered by the user a sequence of actions will be performed:

- The LCD screen will be showing the medicine's name.
- KH56 stepper motors will start rotating clockwise moving the plate vertically to the same height as the product box.
- The TS3214 motor will move the plate horizontally until it is located under the product box.
- The servo motor that is attached to the product box will rotate the spiral coil to move the medicine forward until it reaches the plate.
- Motors will rotate counter clockwise to return the plate with the medicine to the home position where the user can pick it up.

The complete code for this process was written in Arduino IDE [23] and can be found in Appendix B. A visual representation of the process and the code is shown by the flowchart in Figure 3.13. Furthermore, Acumen program [24] was used to simulate the delivering mechanism and the code for the simulation is shown in Appendix C.

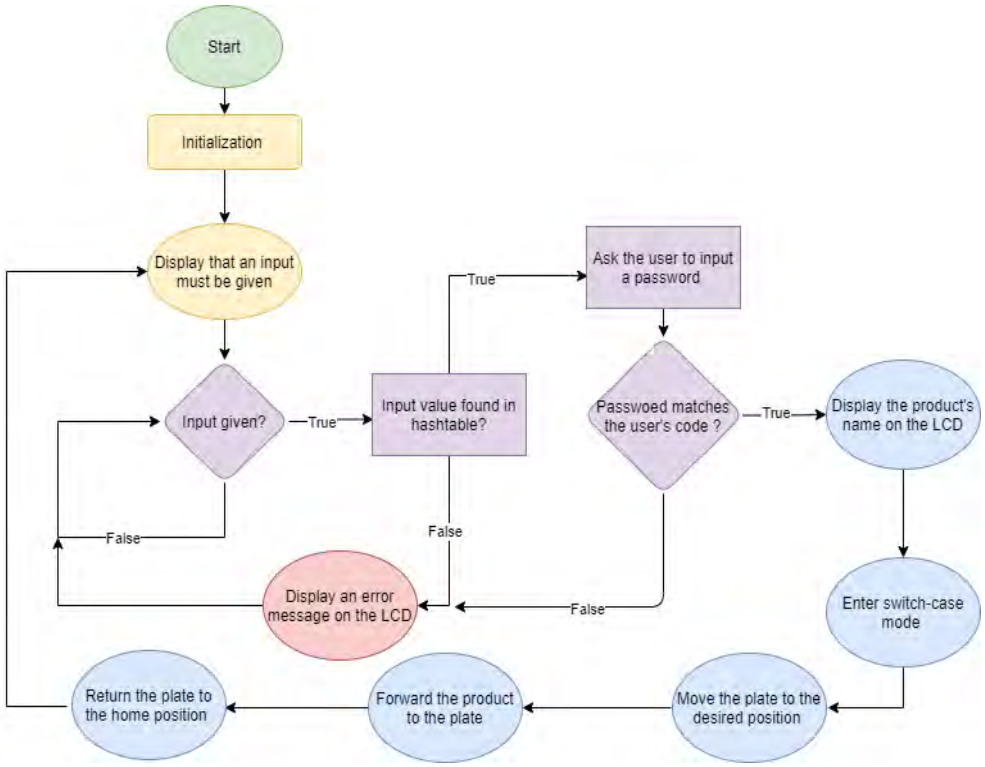


Figure 3.13: Flowchart to visualize the code, made with draw.io [25]

Chapter 4

Testing and Results

This chapter addresses the project's final results which were used to answer the research questions that were mentioned in the Purpose section.

4.1 Final Construction

The prototype's final design and construction can be seen in Figure 4.1, using this prototype, tests were performed to determine how much accuracy and effectiveness can be achieved.

Figure 4.1: The final construction of The Apotekomat, captured by authors and edited using Paint [16]

4.2 Accuracy

One of the most important aspects of this project is to make sure that the right medicine will be delivered to the right person. A total of 30 tests were conducted in order to ensure that the system is able to determine which medicine should be delivered and that the motors operate in the proper sequence to deliver the right medicine to the user. The prototype operated successfully in 25 tests, however, in

the remaining five tests, the two KH56 motors were unable to raise the plate. This problem was solved by decreasing the rotation speed of the motors, resulting in a much smoother vertical movement. This issue was found to be resolved after ten retests that were done after changing the rotation speed.

4.3 Delivering Time

Several tests were made to calculate the time that the prototype needs to move the plate to different positions. Tests showed that the time required to rotate the spiral coil such that the product moves forward to the plate is 1,4 seconds.

Table 4.1: Delivering time test results

Movement	Horizontally	Vertically
From home position to first box	1 sec	1 sec
Pass one product box	2 sec	2 sec

Based on the test results, shown in Table 4.1, the time required to deliver a medicine to the user can be determined using equation 4.1. Note that the test results can vary depending on the values specified by the arduino code's delay functions.

$$t = 4y + 4x + 5,4 \tag{4.1}$$

Where t is time in seconds, y is the number of product boxes passed by the plate in the horizontal direction and x is the number of product boxes passed by the plate in the vertical direction. To ensure that the equation that we concluded is realistic, several tests were carried out to calculate the time required to deliver medicine from our four product boxes. Test results compared to the time estimated by equation 4.1 are presented in Table 4.2, where it can be seen that the results are almost identical. Meaning that the equation can be used to estimate the time in a larger project with more boxes.

Table 4.2: Comparing test results to equation 4.1

Movement	Test result	Estimated time with equation 4.1
First box	6,5 sec	5,4 sec
Second box	10 sec	9,4 sec
Third box	10,5 sec	9,4 sec
Fourth box	14 sec	13,4 sec

4.4 Maximum Load Test

Different medications have variable weights, it is important to determine whether the stepper motors can handle the weight of the medicine for transport. Transporting the medicine only takes place on the return trip, so the test only calculates the weight of the downward movement. In the test, 10 different items were selected to simulate the weight of the medicine, ranging from 100 grams to 1000 grams with a difference of 100 grams per item. When the weight of the medication reached 600 grams, the stepper motors could no longer function normally.

The thrust of the spirals and servo motors were also tested to determine the maximum weight of the medicine could have. The tests were generally consistent with the previous tests, where items of different weights were selected, and tested in increasing order of weight, from small to large until the spirals could not push out the medicine. The final test result shows that the maximum thrust of the spirals is 300 grams.

Chapter 5

Discussion and Conclusions

This chapter discuss the project's research questions by using the data from the test results.

5.1 Discussion

Most of the mechanical and electrical components functioned well and gave the desired results during tests. Throughout the construction process servo motors, LCD screen, the keypad and most of the 3D printed parts were reliable. Stepper motors, on the other hand, needed to be replaced and reprogrammed several times during the development of the prototype. At first, only TS3214 motors were used in this project, but the plate and the attachments used to hold the horizontal rods were heavier than we estimated. As a result, the TS3214 motor stalled and were not able to transport the plate vertically, necessitating the use of two KH56 motors. But, as it was presented in section 4.2, these two motors failed in 5 out of 30 tests most likely because of the minor defects in the 3D printed parts and the binding in the wood used to build the shell. In order to overcome this complication we needed to reduce the speed of rotation which increased the time needed to deliver medicines.

Even after slowing motors down, the prototype performed well in the delivery time test. Test results, presented in section 4.3, showed that the prototype was able to deliver medicine in a reasonable time. Most of the vending machines used nowadays contain around 40 to 60 product boxes [26]. Equation 4.1 was implemented to estimate the worst case scenario time to 62 seconds which is comparable to the time it takes a pharmacist to obtain a prescription from the respiratory.

According to the results of the weight test, the maximum weight of the medicine that the stepper motor can withstand is 600 grams, and the maximum weight of the medicine that can be pushed by the spiral is 300 grams, so to meet the above two conditions, the maximum mass of the medicine cannot be higher than 300 grams. As an example, the weight of each tablet in a variety of tablet-type drugs currently on the market is approximately 500 milligrams, and the weight of the whole pack-

age is roughly 200 to 300 grams. Therefore the current prototype can be equipped with most of the tablet-type medicine. But other types of medicine, such as liquid medicines, often have higher weights and larger volumes. Meaning that medicine other than tablet formulations cannot be loaded into the current system. However, this problem can be solved by using more powerful motors and thicker spirals which makes it possible to deliver medicine that weighs more than 300 grams.

5.2 Conclusions

Various methods to store and deliver medicine were taken into consideration during the project's preliminary design process, including using a robot arm to distribute the medicines directly from the pharmacy's repository to the consumer, or transporting the medicines using a system similar to a vending machine. After discussions, it was decided that a vending machine alike design would be the most optimal choice for this project. Because it can be placed anywhere it is needed, has its own repository which will eliminate the need for special equipment to place the machine in pharmacies. Making the vending machine-like construction better suited to achieve the project's goals.

To answer the question about how effective can this prototype automate the medicine delivery process, there are some important factors that should be considered. First and foremost, the time factor, which was estimated to be approximately one minute under the assumption that only 60 different types of medicines are stored in The Apotekomat. Another factor is the medicine's mass and form, as it was stated in section 5.1, The Apotekomat can only store medicine with a specific shape and a maximum weight of 300 grams. Making this prototype an effective tool to automate drug delivery, but it is limited to certain types and quantities of medications. Lastly, an important aspect of the chosen design is that the product boxes would need to be filled with medicines by a specialized pharmacist to ensure that the correct drug is put into the correct product box. This makes the prototype a partially autonomous unit, but still requires a human operator to ensure that the right user receives the correct drug.

Based on the results presented in section 4.2, the program did not show any problems identifying the correct medicine for the user. However, the results revealed that in order to achieve the necessary precision in the delivering mechanism, the stepper motors should be slowed down. Meaning that the system will be accurate as long as the motors are operated at an acceptable speed.

Chapter 6

Recommendations And Future Work

This chapter summarizes what can be improved or enhanced about the project and provides a foundation for future releases.

6.1 Selection Of Materials

Wooden boards were selected as the main material for the prototype because they could be cut and sanded more easily than other materials to achieve the required size and dimensions. However, considering that some medications may need to be stored at low temperatures and the wooden board does not have insulation properties, the choice of wooden board may not be the best option. Therefore, this factor needs to be taken into consideration in future improvements. Furthermore, in subsequent improvements, it is possible to increase the maximum medicine weight that the machine can withstand by using harder material to construct spiral coils. In this way, more types and heavier medicine can be made available for patients to obtain directly from the machine.

6.2 Dimensional Errors

Most of the construction parts, such as box shells, delivery boxes, and rods are cut by auxiliary machines or completely manually. As a result, there is a certain error in the dimensions of the parts compared to fully machine-cut parts, which results in a certain gap between the different parts in the subsequent assembly. Therefore, in future improvements, all parts that require precise measurements should be constructed by machines.

6.3 More Comprehensive Functions

This prototype has only the most basic functions and can only contain four different medicines, so in future practical use, more product boxes with medicine could be added to supply more people in need. In addition, the machine could be fitted with a larger screen to display more detailed information needed and the 12-key keyboard could be replaced with an alphabetic keyboard or an ID-card reader which makes the system more secure and faster at identifying the user.

Bibliography

[1] Arlan J Hoffman and Charles P Crawley. *Vending machine operating mechanism*. US Patent 5,070,986. Dec. 1991.

[2] Ginger Labs. *Notability*. Version 10.4. Feb. 15, 2021. URL: https://apps.apple.com/se/app/notability/id360593530?mt=8&ign-mpt=uo%3D4.

[3] John H. Davies. "Chapter 1 - Embedded Electronic Systems and Microcontrollers". In: *MSP430 Microcontroller Basics*. Ed. by John H. Davies. Burlington: Newnes, 2008, pp. 1–20. ISBN: 978-0-7506-8276-3. DOI: https://doi.org/10.1016/B978-075068276-3.50002-X. URL: https://www.sciencedirect.com/science/article/pii/B978075068276350002X.

[4] Magnetic Innovations. *DC MOTOR, HOW IT WORKS?* 2021. URL: https://www.magneticinnovations.com/faq/dc-motor-how-it-works/ (visited on 02/15/2021).

[5] Jeff Shepard. *Motor fundamentals and DC motors*. 2020. URL: https://www.powerelectronictips.com/motor-fundamentals-dc-motors-faq/ (visited on 04/07/2021).

[6] DroneBot Workshop. *Stepper Motors with Arduino - Getting Started with Stepper Motors*. 2018. URL: https://dronebotworkshop.com/stepper-motors-with-arduino/ (visited on 02/13/2021).

[7] Reston Condit and Douglas W Jones. "Stepping motors fundamentals". In: *Microchip Inc. Publication AN907* (2004), pp. 1–22.

[8] Sustek, Michal et al. "DC motors and servo-motors controlled by Raspberry Pi 2B". In: *MATEC Web Conf.* Vol. 125 (2017), p. 02025. DOI: 10.1051/matecconf/201712502025. URL: https://doi.org/10.1051/matecconf/201712502025.

[9] A. Williams. *Microcontroller Projects Using the Basic Stamp*. CRC Press, 2002, p. 344. ISBN: 9781482280777. URL: https://books.google.se/books?id=j0hZDwAAQBAJ.

[10] Øyvind Nydal Dahl. *What Is an H-Bridge?* 2018. URL: https://www.build-electronic-circuits.com/h-bridge/ (visited on 02/14/2021).

[11] Arief Wisnu Wardhana and Daru Tri Nugroho. “Stepper motor control with DRV 8825 driver based on square wave signal from AVR microcontroller timer”. In: *AIP Conference Proceedings*. Vol. 2094. 1. AIP Publishing LLC. 2019, p. 020015.

[12] Mikael Sjöberg and Jonas Xu. “A.D.D”. Degree project in technology, first cycle, KTH, Stockholm 2019. URL: http://www.diva-portal.org/smash/get/diva2:1373888/FULLTEXT01.pdf (visited on 04/20/2021).

[13] Diigiit Robotics Europe. *Pololu DRV8825 High-Current Stepper Motor Controller*. URL: http://www.eu.diigiit.com/pololu-drv8825-high-current-stepper-motor-controller (visited on 04/27/2021).

[14] Siemens. *Solid Edge 2020*. Version 220.00.00.104 x64. Mar. 4, 2021. URL: http://solidedge.siemens.com/.

[15] I. D. L. Potsdam. *Fritzing*. Version 0.9.6. Apr. 8, 2021. URL: https://fritzing.org/.

[16] Microsoft. *Paint*. Version 10.0.19042.964. Apr. 8, 2021. URL: https://support.microsoft.com/en-us/windows/get-microsoft-paint-a6b9578c-ed1c-5b09-0699-4ed8115f9aa9.

[17] Arduino. *Arduino MEGA 2560 and Genuino MEGA 2560*. 2021. URL: https://store.arduino.cc/arduino-mega-2560-rev3 (visited on 04/06/2021).

[18] Parallax Inc. *Parallax Continuous Rotation Servo*. 2021. URL: https://www.parallax.com/product/parallax-continuous-rotation-servo/ (visited on 04/07/2021).

[19] Premiumplc. *Japan Servo Co. Kh56Qm2-801 Stepper Motor 1.8 Deg*. URL: https://picclick.com/Japan-Servo-Co-Kh56Qm2-801-Stepper-Motor-18-Deg-312217122386.html (visited on 04/07/2021).

[20] Fredrika Ardestam and Sara Soltaniah. “Dot Master”. Degree project in electrical engineering, first cycle, KTH, Stockholm 2018. URL: http://www.diva-portal.org/smash/get/diva2:1217292/FULLTEXT01.pdf (visited on 04/15/2021).

[21] Electrokit. *LCD 2×16 char TN LED*. 2021. URL: https://www.electrokit.com/en/product/lcd-2x16-char-tn-led/ (visited on 04/27/2021).

[22] Banggod. *5pcs 12 Key MCU Membrane Switch Keypad 4 x 3 Matrix Array Matrix Keyboard Module Geekcreit for Arduino - products that work with official Arduino boards*. 2021. URL: https://banggood.app.link/NItbNaSWXfb (visited on 04/27/2021).

[23] Arduino. *Arduino IDE*. Version 1.8.13. Feb. 1, 2021. URL: https://www.arduino.cc/en/software.

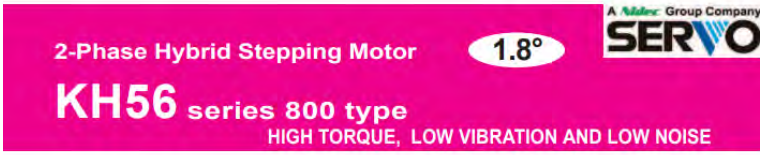
[24] Effective Modeling Group (EMG). *Acumen*. Mar. 19, 2021. URL: http://www.acumen-language.org/.

[25] JGraph ltd. *draw.io*. Version 14.5.1. Apr. 28, 2021. URL: https://www.diagrams.net/.

[26] Fantastic Offense. *Snack Vending Machine - Large*. 2021. URL: https://www.dimensions.com/element/snack-vending-machine-large#:~:text=Featuring%20a%20robust%20selection%20of,617%20items%20at%20a%20time. (visited on 04/29/2021).

Appendix A

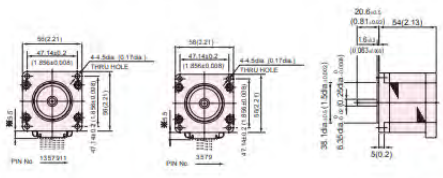
Data sheet for the KH56 stepper motor



■ STANDARD SPECIFICATIONS

MODEL		KH56KM2			
		-801	-802	-803	-851
DRIVE METHOD	———	UNI-POLAR			BI-POLAR
NUMBER OF PHASES	———	2			2
STEP ANGLE	deg./step	1.8			1.8
VOLTAGE	V	2.4	3.7	6.8	4.35
CURRENT	A/PHASE	3.0	2.0	1.0	1.5
WINDING RESISTANCE	Ω/PHASE	0.8	1.85	6.8	2.9
INDUCTANCE	mH/PHASE	1.1	3.3	13.5	10.7
HOLDING TORQUE	mN • m	833	833	833	981
	oz • in	118	118	118	132
DETENT TORQUE	mN • m	37	37	37	37
	oz • in	5.6	5.6	5.6	5.6
ROTOR INERTIA	g • cm2	270	270	270	270
	oz • in2	1.48	1.48	1.48	1.48
WEIGHTS	g	650	650	650	650
	lb	1.4	1.4	1.4	1.4
INSULATION CLASS	———	JIS Class E (120°C 248°F) (UL VALUE : CLASS B 130°C 266°F)			
INSULATION RESISTANCE	———	500VDC 100MΩmin.			
DIELECTRIC STRENGTH	———	500VAC 50HZ 1min.			
OPERATING TEMP. RANGE	°C	0 to 50			
ALLOWABLE TEMP. RISE	deg.	70			

■ DIMENSIONS unit = mm (inch)



UNI-POLLAR Bi-POLAR SINGLE SHAFT

18

Appendix B

Arduino Code

```
/*///////////////// INTRODUCTION ////////////////
University : Royal Institute of Technology, KTH
Project: The Apotekomat
Course : MF133X ,Degree Project in Mechatronics
Students : Ehab Ziad Raheem & Jiahao wang (Grupp 12)
Assignment : 100% assignment Arduino program
Last modification : 2021/05/02
////////////////////////////////////////////*/

/************************INITIERING AND DEFINE PINS
   ********************/

/*Including libraries*/
#include <Dictionary.h>                //Including a library
   to use dictionary and Hashtabels in Arduino
#include <NodeArray.h>
#include <LiquidCrystal.h>             //Including a library
   to use the LCD screen
#include <Keypad.h>                    //Include a library to
    use the  Keypad
#include <Servo.h>                     //Include a library to
    use servo motors


/*Define stepper motor connections:*/
#define direction_pin 42
#define stepping_pin 43
#define direction_pintwo 44
#define stepping_pintwo 45
#define direction_pinTHREE 47
```

```
26 #define stepping_pinTHREE 48
27
28 /*Define the maximum allowable length of both keys and
      values */
29 #define _DICT_KEYLEN 64
30 #define _DICT_VALLEN 254
31 Dictionary &dict = *(new Dictionary(5));   //Creating a
      dictionary with the size 5
32 Dictionary &dictpass = *(new Dictionary(5)); //Adding a
      dictionary for the passwords
33 /*Introducing servo motors*/
34 Servo servo1;
35 Servo servo2;
36 Servo servo3;
37 Servo servo4;
38
39 /*Define servo motors connections:*/
40 int servo_1_Pin = 6;                    //define pins for the
      first servo
41 int servo_2_Pin = 2;                    //define pins for the
      second servo
42 int servo_3_Pin = 3;                    //define pins for the
      third servo
43 int servo_4_Pin = 4;                    //define pins for the
      fourth servo
44
45 /*Define the keypad rows and columns connections:*/
46 const byte ROWS = 4;                    // four rows in the
      Keypad that is used
47 const byte COLS = 3;                    // three columns in
      the Keypad that is used
48 int chosenBox;
49 byte rowPins[ROWS] = {25,24,23, 22};  // connect to the row
      pinouts of the keypad
50 byte colPins[COLS] = {28, 27, 26};    // connect to the
      column pinouts of the keypad
51 char keys[ROWS][COLS] = {
52   {'1','2','3'},
53   {'4','5','6'},
54   {'7','8','9'},
55   {'*','0','#'}
56 };
57 Keypad customKeypad = Keypad(makeKeymap(keys), rowPins,
      colPins, ROWS, COLS);
```

```
58
59 LiquidCrystal lcd(39, 38, 37,36, 34, 35);           //Assign
       the LCD screen to the pins in Arduino
60
61
62 String inputString;                                  //The
       input varibel that will contain the four digit code
       entered by the user
63 String PassString;                                  //The
       input varibel that will contain the four digit password
       entered by the user
64
65 /*************************************/
66
67 void setup() {
68   //Creating a Hashtable that includes the code of the
       medicines as a key and the name as a value
69   dict("1111", "Alvedon");
70   dict("2222", "Paracetamol");
71   dict("3333", "ViaminC");
72   dict("4444", "VitaminD");
73   dict("5555", "VitaminB");
74   //Craeting a Hashtable that includes passwords
75   dictpass("1111", "1616");
76   dictpass("2222", "2626");
77   dictpass("3333", "3636");
78   dictpass("4444", "4646");
79   dictpass("5555", "5656");
80
81   // Attaching servo motors to Digital PWM in Arduino:
82   servo1.attach(servo_1_Pin);
83   servo2.attach(servo_2_Pin);
84   servo3.attach(servo_3_Pin);
85   servo4.attach(servo_4_Pin);
86   screen_start();
87   inputString.reserve(5); // maximum number of digit the
       code entered is 5
88
89 }
90
91 void loop() {
92   char nyckel = customKeypad.getKey();                //
       Get the key value from the key pad
93   if (nyckel) {
```

```
94      lcd.clear();
95      if (nyckel >= '0' && nyckel <= '9' && inputString.length
            () <= 4) {
96        inputString += nyckel;                                  //
              to change the key value to a string instead of a
              char so that it can be used in the dictionary
              library
97        screen(inputString);                                    //
              call screen function
98    }
99      if (inputString.length() == 5 ) {                       //
           if the users enters more than 4 digits
100       errorscreen();                                        //
              the screen will display an error and be restarted
101       delay(3000);
102       screen_start();
103       inputString = "";
104     }
105     if (nyckel == '#'&& inputString.length() < 4 ){       //
           if the user input an "#" by mistake
106       inputString = "";                                     //
              the screen should be restarted
107       screen_start();
108     }
109     if (nyckel == '#'&& inputString.length() == 4 ) {     //
           When the user is done writing the code
110       check_dictionary(inputString);                        //
              calling the function to check if the code is vaild
111       delay(3000);
112       inputString = "";
113
114
115       }
116
117     if (nyckel == '*') {                                    //
           to reset the code
118       inputString = "";                                     //
              Cleaning the string
119       screen(inputString);                                  //
              restart the screen to input a new code
120
121     }
122   }
123
```

```
124 }
125
126 /**********************CHECKING THE USER'S CODE AND PASSWORD
       ***********************/
127 void check_dictionary(String medicin_number) {    // go
       through dictionary and check if the medicin is available
128    if (dict(medicin_number)){
129     lcd.clear();                                    //if the
          code is avilable in the dictionary, show the medicin
          name on the screen and move to the check password
          loop
130     lcd.setCursor(0, 0);
131     lcd.print("Din␣medicin:");
132     lcd.setCursor(0, 1);
133     lcd.print(dict[medicin_number]);
134     delay(2000);
135     check_password();
136
137    }
138    else {                                    //If the medicin
          is not available or if the entered code was inaccurate
139     // then inform the costumer and return to the main
          screen "screen_start"
140     lcd.clear();
141     lcd.setCursor(0, 0);
142     lcd.print("Medicinen␣finns");
143     lcd.setCursor(0, 1);
144     lcd.print("inte␣i␣lager");
145     delay(2000);
146     screen_start();
147    }
148
149 }
150
151 /*This function is similar to the loop function that cheks
       the code*/
152 /*it reacts with the password entered by the user*/
153 void check_password(){
154   password_screen();
155 while (true){
156   char pass = customKeypad.getKey();                    //
        Get the key value from the key pad
157   if (pass) {
158     lcd.clear();
```

```
159     if (pass >= '0' && pass <= '9' && PassString.length() <=
            4) {
160       PassString += pass;
161       pscreen(PassString);
162       }
163     if (PassString.length() == 5 ) {
164       errorscreen();
165       delay(3000);
166       PassString = "";
167       check_password();
168       }
169     if (pass == '#'&& PassString.length() < 4 ){
170       errorscreen();
171       delay(3000);
172       PassString = "";
173       check_password();
174       }
175     if (pass == '#'&& PassString.length() == 4 ) {
176       pscreen(PassString);
177       go_through_passwords(PassString);     //if the
              password entered is 4 digit then this function will
               be called
178       PassString = "";
179       break;
180       }
181     if (pass == '*') {
182       PassString = "";
183       password_screen();
184       }
185     }
186   }
187 }
188
189 void go_through_passwords(String password){   //A function
    To check if the entered password is aligned with the user
    's code
190     if (dictpass[inputString]== password){
191     lcd.clear();                                  //if the
          code is avilable in the dictionary, show the medicin
          name on the screen
192     lcd.setCursor(0, 0);
193     lcd.print(dict[inputString]);
194     lcd.setCursor(0, 1);
195     lcd.print("levereras...");
```

```
196     deliver_medicin(inputString);              //start
           delivering the medicin
197   }
198   else {
199     lcd.clear();                                 //if the
           code is avilable in the dictionary, show the medicin
           name on the screen
200     lcd.setCursor(0, 0);
201     lcd.print("Fel password");
202     delay(2000);
203     screen_start();
204   }
205 }
206
207 /*Depending on which code is entered motors will rotate
208 to move the plate from home positionto reach the goal
       position */
209
210 void deliver_medicin(String medicin_number){
211   if (medicin_number == "1111"){
212       chosenBox = 1;
213     }
214      if (medicin_number == "2222"){
215       chosenBox = 2;
216     }
217      if (medicin_number == "3333"){
218       chosenBox = 3;
219     }
220      if (medicin_number == "4444"){
221       chosenBox = 4;
222     }
223     switch (chosenBox){
224
225       case 1:
226        Moveup(200);
227        delay(1000);
228        Moveleft(350);
229        delay(1000);
230        servo1.writeMicroseconds(1200);  // rotate the first
              servo motor
231        delay(1400);
232        servo1.writeMicroseconds(1508); //stop
233        delay(1000);
234        Moveright(350);
```

```
235         delay(1000);
236         Movedown(200);
237         break;
238        case 2:
239         Moveup(200);
240         delay(1000);
241         Moveleft(50);
242         delay(1000);
243         servo2.writeMicroseconds(1200);  // rotate the second
                servo motor
244         delay(1400);
245         servo2.writeMicroseconds(1508); //stop
246         delay(1000);
247         Moveright(50);
248         delay(1000);
249         Movedown(200);
250         break;
251        case 3:
252         Moveup(680);
253         delay(1000);
254         Moveleft(350);
255         delay (1000);
256         servo3.writeMicroseconds(1200);  // rotate the third
               servo
257         delay(1400);
258         servo3.writeMicroseconds(1508); //stop
259         delay(1000);
260         Moveright(350);
261         delay(1000);
262         Movedown(680);
263         break;
264        case 4:
265         Moveup(680);
266         delay(1000);
267         Moveleft(50);
268         delay (1000);
269         servo4.writeMicroseconds(1200);  // rotate the fourth
                servo
270         delay(1400);
271         servo4.writeMicroseconds(1508); //stop
272         delay(1000);
273         Moveright(50);
274         delay(1000);
275         Movedown(680);
```

```
276         break;
277       }
278         delay(3000);                              // After
              delivering the medicine
279         inputString = "";                         //clean the "
              inputString" and return to the main screen to enter
               a new code by new costumer
280         screen_start();
281 }
282
283 /**********************TRANSPORTING THE PLATE
      ******************/
284 /* functios for the stepper motor */
285 void Moveup(int stepsperrevol){         //To move the plate
      up by rotating two KH56-801 motors
286   // Set the spinning direction :
287   digitalWrite(direction_pin, HIGH);
288   digitalWrite(direction_pintwo, LOW);
289   for (int i = 0; i < stepsperrevol; i++) {
290   // These four lines result in 1 step:
291     digitalWrite(stepping_pin, HIGH);
292     digitalWrite(stepping_pintwo, HIGH);
293     delayMicroseconds(2000);
294     digitalWrite(stepping_pin, LOW);
295     digitalWrite(stepping_pintwo, LOW);
296     delayMicroseconds(2000);
297   }
298 }
299
300 void Movedown(int stepsperrevol){      //To move the plate
      down by rotating two KH56-801 motors
301   // Set the spinning direction:
302   digitalWrite(direction_pin, LOW);
303   digitalWrite(direction_pintwo, HIGH);
304   for (int i = 0; i < stepsperrevol; i++) {
305   // These four lines result in 1 step:
306     digitalWrite(stepping_pin, HIGH);
307     digitalWrite(stepping_pintwo, HIGH);
308     delayMicroseconds(2000);
309     digitalWrite(stepping_pin, LOW);
310     digitalWrite(stepping_pintwo, LOW);
311     delayMicroseconds(2000);
312   }
313 }
```

```
314 void Moveleft(int stepservohorisontell){  //To move the
       plate left by rotating the KH56-801 stepper motor
       clockwise
315   digitalWrite(direction_pinTHREE, HIGH);
316   for (int i = 0; i < stepservohorisontell; i++) {
317     digitalWrite(stepping_pinTHREE, HIGH);
318     delayMicroseconds(2000);
319     digitalWrite(stepping_pinTHREE, LOW);
320     delayMicroseconds(2000);
321   }
322 }
323
324 void Moveright(int stepservohorisontell){  //To move the
       plate right by rotating the KH56-801 stepper motor
       counter clockwise
325   digitalWrite(direction_pinTHREE, LOW);
326   for (int i = 0; i < stepservohorisontell; i++) {
327     digitalWrite(stepping_pinTHREE, HIGH);
328     delayMicroseconds(2000);
329     digitalWrite(stepping_pinTHREE, LOW);
330     delayMicroseconds(2000);
331   }
332 }
333
334 /**********************SCREEN CODE**********************/
335 void screen_start(){                //A function to display
       the "start" screen
336   // set up the LCD's number of columns and rows:
337   lcd.begin(16, 2);
338   // Clears the LCD screen
339   lcd.clear();
340   // Print a message to the LCD.
341   lcd.print("Skriv␣in␣din␣kod");
342   lcd.setCursor(0, 1);
343   lcd.print("Avsluta␣med␣'#'␣");
344 }
345
346 void screen(String show_on_screen){  //A function to display
       the "start" screen after statring inputing the code
347     lcd.setCursor(0, 0);
348     lcd.print("Vid␣fel␣tryck␣'*'");
349     lcd.setCursor(0, 1);
350     lcd.print("Koden:␣");
```

```
351     lcd.print(show_on_screen);       // append new character
          to input string
352 }
353
354 void errorscreen(){                  //A function to display
      an error messege
355     lcd.clear();
356     lcd.setCursor(0, 0);
357     lcd.print("Endast␣4␣siffrig");
358     lcd.setCursor(0, 1);
359     lcd.print("kod,testa␣igen");
360 }
361
362 void password_screen(){          //screen #1 Displaying
      instructions about the password
363     lcd.clear();
364     lcd.setCursor(0, 0);
365     lcd.print("Skriv␣password");
366     lcd.setCursor(0, 1);
367     lcd.print("Avsluta␣med␣#");
368
369 }
370
371 void pscreen(String show_on_screen){     //screen #2
      Displaying instructions about the password
372     lcd.setCursor(0, 0);
373     lcd.print("Vid␣fel␣tryck␣'*'");
374     lcd.setCursor(0, 1);
375     lcd.print("Password:␣");
376     lcd.print(show_on_screen);
377 }
```

Appendix C

Acumen Code

```
////////Introduction/////////
////GROUP: PROJECT GROUP 12
////STUDENTS: Ehab Ziad Raheem && Jiahao Wang
////Assignment: The Apotekomat simulation in Acumen
////LAST MODIFICATION: 2021/03/26
////////////////////////////
model Main(simulator) =     //main part for simulator
 initially
    _3D=(),x=0,x'=1,          //the variables for
    //different cases
    y=0, y'=0, y''=0,             //variable x is used
    //for spirals that push products,
    //in our case it is a cylinder
    z=0,z'=0,z''=0,               //variable y is used
    //for their situation where the plate must come under
    //the box to able to catch products
    platta=create p((0,0,0),black),   //variable z is
    //for products to fall down when cylinder pushed forward
    medicin=create m((0,0,0),green)  //creates a plate and
    //a product (medicine)with "model" with a specific size
     /////if state for different parts how to move
      if x<4    // when x is less than 4, the
      //cylinder will increase its length
      //otherwise will it stop
        then x'=1
        else x'=0,
       if y<10      //when y is less than 10 then
       //the plate will be moved in minus y-direction until
       // it reaches the expected location
        then y''=-(y'-11)
```

```
    else if y'>0
       then y''=-10
       else y''=0,        //otherwise will it stop
       platta.pos=(0,-y,0),
  if z<2                  //when  z is less than 2,
  //the product will be moved in the x- and z-directions
  //to be able to fall down
    then z''=-(z'-1)
    else if z'>0
       then z''=-10
       else z''=0,     // otherwise will it stop
       medicin.pos=(z,0,-z),
//appearance of how everything looks
 _3D = ((Box     //box one with length 20,
 //width 5, height 5 and thickness 0.2
             center=(0,0,0)
             size=(0.2,5,5)
             color=red
             rotation=(0,0,0))
            (Box
             center=(5,2.5,0)
             size=(10,0.2,5)
             color=red
             rotation=(0,0,0))
            (Box
             center=(5,-2.5,0)
             size=(10,0.2,5)
             color=red
             rotation=(0,0,0))
            (Box
             center=(5,0,2.5)
             size=(10,5,0.2)
             color=red
             rotation=(0,0,0))
            (Box
             center=(5,0,-2.5)
             size=(10,5,0.2)
             color=red
             rotation=(0,0,0))
             (Cylinder              //"spiral"
             center=(5,0,0)
             size=(2*x,1.5)
             color=black
             rotation=(1.5,1.5,0))
```

```
                //box two, same dimensions for box two
                (Box
                  center=(0,5,0)
                  size=(0.2,5,5)
                  color=blue
                  rotation=(0,0,0))
                (Box
                  center=(5,7.5,0)
                  size=(10,0.2,5)
                  color=blue
                  rotation=(0,0,0))
                (Box
                  center=(5,5,2.5)
                  size=(10,5,0.2)
                  color=blue
                  rotation=(0,0,0))
                (Box
                  center=(5,5,-2.5)
                  size=(10,5,0.2)
                  color=blue
                  rotation=(0,0,0))
                 (Cylinder              // "spirals"
                  center=(5,5,0)
                  size=(1.2*x,1.5)
                  color=black
                  rotation=(1.5,1.5,0)))

model p(pos,col)=                       // model for plate
  initially
  _3D=(), _plot=()
  always
  _3D = ((Box                          //dimension for plate
                  center=pos+(15,15,-3)
                  size=(5,5,0.2)
                  color=black))
model m(pos,col)=                      //model for products
 initially
   _3D=(), _plot=()
 always
    _3D = ((Sphere                    //diemension for product
                  center=pos+(13,0.3,0)
                  size=1
                  color=green
    ))
```

TRITA TRITA-ITM-EX 2021:30