CHAPTER ONE

INTRODUCTION

1.1 Overview

Hospitals are become very large and most of them have more than five floors, so they have problem with transfer their medications, drugs, medical consumables between the pharmacy and the nurse stations in deferent floors. Sometimes there are important dangerous medications that had to be delivered, and some critical situations need faster response.

Now a days the world faced by epidemic disaster, Covied-19 which is infectious disease separates primarily by direct contact between people and the infection rate Is higher in crowdie areas the elevator operates by vertical movement, it has a motor settled on a box at the back, and there is a motor driver to control the main motor and the movement of the cables.

The control of the elevator will be by control circuit consist of a group of push buttons to make the order, this orders is sent to the Arduino directly which is connected with the main motor through the motor driver. Arduino is programmed to receive signals from push buttons which are the user interface in the project and the ultrasonic sensor to detect the floor to control the main motor.

1.2 Problem Statement

Nurse assistant is needed to provide the medical needs (medicine and medical consumables) from the hospital pharmacy. The process of fetching medicine will be delayed due to crowding inside the hospital. Covied-19 infection rate increases due to direct contact and interaction between people.

1.3 Objectives

The main aims of this project are:

- To implement and design an elevator that connects the pharmacy and the nurse stations
- To decrease the infection rate of coveid-19.
- Reduce crowding in the pharmacy and the hospital will need fewer employers. `

1.4 Methodology

The necessary data to implement the project was collected through field tours in the hospitals, interviews and discussions with the hospital staffs and co-picante. Data was recorded, studied and analyzed, and according to the result, the project was applied.

To design an elevator which convey the medicines and moves in vertical direction only:

The box, which carries the medicine, is designed inside the building, which is attached to weir causing the movement. The weir was fixed to the pulley that was mounted to the motor shaft.

Electrical circuit was designed to connect the motor, Arduino, push buttons and ultrasonic sensors.

1.5 Layout

Chapter one shows the introduction of the research, the problem statement, objectives and the methodology.

Chapter two shows the general information about the project and the basic topics that contain theoretical side of the project and the related study.

Chapter three shows the modeling and circuit components with details, the first section gives brief definition to each component and the other section discuss all the components in details.

Chapter four represents the simulation and the practical part of the project, the simulation program, how it's used and discuss basic concepts about the program, then shows simulation modules and explain for each modules. Finally detailed explanation for the practical part.

Chapter five represents a summary of the theoretical study, circuit component, simulation, and practical implementation that has been done in the project.

CHAPTER TWO

GENERAL INFORMATION

2.1 Introduction

The project aims to deliver medications and medical consumables in the hospitals between the pharmacy and the nurse stations by using a control circuit consists of push buttons that send signals to the Arduino which is programmed to control the movement of the motor which is connected to the motor driver, and use the ultrasonic sensor to detect the elevator position, also talk about the related studies to the project.

2.2 Hospital System

Hospitals deal with cases in specific sequence, once the patient sees the doctor either in the clinic or in the emergence department, if the doctor decided that patient needs further care and follow up, a file will be created for the patient, this file contain the patient ID, patient payment information, Prescription medication and the full details about the patient case and the file will be saved in HIS and attached to specific nurse station according to the patient case, this file should be accessed only by doctors, nurses and nurse assistants due to its importance, the procedure of fetching medications should be completed through nurse assistant. Also HIS can be used to order the medications but the co-patient must take them in person.

2.3 Control System

A control system is a collection of components working together under the direction of some machine intelligence. In most cases, electronic circuits provide the intelligence, and electromechanical components such as sensors and motors provide the interface to the physical world. The following figure shows the simple block diagram of a control system. Figure 2.1 shows the open loop control system block diagram [1].

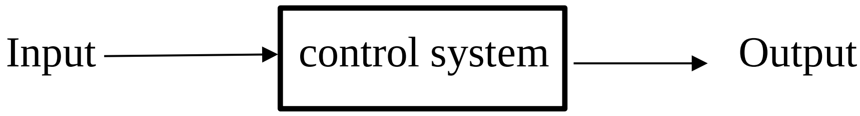

Figure 2.1: Block diagram of the open loop control system.

Every control system has (at least) a control and an actuator (also called a final control element), the controller is the intelligence of the system and is usually electronic. Based on some parameters, will classify the control systems into the following ways.

(a) Continuous time and discrete-time control systems

Control Systems can be classified as continuous time control systems and discrete time control systems based on the type of the signal used. In continuous time control systems, all the signals are continuous in time. But, in discrete time control systems, there exists one or more discrete time signals.

(b) SISO and MIMO control systems

Control Systems can be classified as Single-Input-Single-Output (SISO) control systems and Multiple-Input-Multiple-Output (MIMO) control systems based on the number of inputs and outputs present. SISO control systems have one input and one output. Whereas, MIMO control systems have more than one input and more than one output.

(c) Open loop and closed loop control systems

Control Systems can be classified as open loop control systems and closed loop control systems based on the feedback path. In open loop control systems, output is not fed-back to the input. So, the control action is independent of the desired output. In closed loop control systems, output is fed back to the input. So, the control action is dependent on the desired output [1].

2.4 Microcontroller

A microcontroller is a device, which contains all the devices necessary to form a working system, in a single chip. It consists of a Central Processing Unit (CPU), data and program memory, serial and parallel Input/ Output (I/O), timers, external and internal interrupts. And communicate with the outside world through different ports, these ports can be used as address and data bus depending upon control signals [2].

Microcontroller Random Access Memory (RAM) is small, but it is enough for small applications. If it is not sufficient, then external memory may be added in the microcontroller-based system. Microcontrollers are available from 4-bit to 32-bit. 4-bit microcontrollers are used for simple applications. 16-bit and 32-bit and 64-bit microcontrollers are used for high speed applications. 8-bit microcontrollers are most commonly used in different applications. Microcontrollers are used in small embedded system products to perform control-oriented functions. Modern microcontrollers are available in different packages. This packaging forms the outer shape of the IC with the silicon wafer encapsulated inside it [3].

Intel created many significant microcontrollers besides producing the world's first ever microprocessor. The important ones produced by Intel are the 8048 and the 8051 microcontrollers. 8048 was introduced in 1976 and was the first of Intel's microcontrollers. It was used as the processor in the PC keyboard of IBM. The 8051 microcontroller was introduced in 1980 and is one of the most popular microcontrollers. It is even used now and is considered to be one of the most long-lived microcontrollers.

Today, in addition to the general purpose gadgets, unique microcontrollers are being created for areas like lighting, automotive, communications, and low-power driven consumer goods. The present day microcontrollers like AVR, and PIC have become smaller and sleeker yet more and more powerful. For instance, there are so tiny microcontrollers available, small and cheap enough to be used in simple products like toothbrushes and toys [3].

2.5 Related Study

Pneumatic tube transport (PTT) is system that propel cylindrical containers through networks of tubes by compressed air or by partial vacuum, pneumatic conveying systems are basically quite simple and are eminently suitable for the transport items from specific place to another. The system requirements are a source of compressed gas, usually air, a feed device, a conveying pipeline (a pipe made of something like PVC plastic or a strong lightweight metal such as aluminum) and a receiver. The system is totally enclosed, and if it is required, the system can operate entirely without moving parts coming into contact with the conveyed material [4].

A pneumatic tube transport system, which may be either a 'point-to-point' or a multi-point system, is a distribution network of tubes through which capsule of various sizes containing small items are driven by air flow. The prime mover is an exhauster or blower with a changeover valve which can alter the direction of the air flow in the tube as required moving the carrier through the system. The destination of the capsule may be controlled by diverters which switch the capsule from one branch to another. Often, the sending and receiving stations have chimes, ringers, or flashing lights to signal when a package has just been received. A central controller ensures that capsules are routed through the network [4].

These systems provide an efficient, rapid and secure means of transporting different items from one department to another. The system take up little floor space and the pipeline can be easily routed up walls, across roofs or even underground to avoid any existing equipment or structures. Pipe bends in the conveying line provide this flexibility, but they will add to the overall resistance of the pipeline. The speed of the system (up to 5 meters per second would be regarded as an effective speed considerably reduces the wait time for the process.

The material to be transported is loaded into a capsule which is then placed in the tube at the sending station. A destination station address, usually a three-digit numerical code is entered via a keypad and the capsule is then sent automatically through the network of transmission tubing to the destination. A large pneumatic system might have up to 500 sending and receiving stations, dozens of transfer units where packages can be routed between senders and receivers in complex ways, and dozens of compressor/blower units to provide the pneumatic power [5].

.2.6 Motor

Motors are electrical devices that convert electrical energy into mechanical energy. When a current-carrying conductor is placed in a magnetic field, it experiences a force. Experiment shows that the magnitude of the force depends directly on the current in the wire and the strength of the magnetic field, and that the force is greatest when the magnetic field is perpendicular to the conductor.

Electrical motors are used to produce linear or rotary force (torque), and should be distinguished from devices such as magnetic solenoids and loudspeakers that convert electricity into motion but do not generate usable mechanical powers, which are respectively referred to as actuators and transducers.

Electrical motors are found in applications as diverse as industrial fans, blowers and pumps, machine tools, household appliances, power tools, and disk drives. Motors can be powered by direct current sources, such as from batteries, motor vehicles or rectifiers, or by alternating current sources, such as from the power grid, inverters or generators [6].

(a) AC motor

Electrical device that convert the AC power into mechanical energy and it's consists of stator which is consists of a steel frame that encloses a hollow, cylindrical core made up of thin laminations of silicon steel which have coils supplied with alternating current to produce a rotating magnetic field. And rotor mounted on a shaft, it's a hollow laminated core having slots on its outer periphery. The winding placed in these slots (called rotor winding) may be either Squirrel cage type or Wound type [6].

(b) DC motor

Are electrical devices that convert direct electrical current power into mechanical energy. DC motors have some internal mechanism, either electromechanical or electronic, to provide the change in the direction of current flow in part of the motor.

The motor has two separate electrical circuits. The smaller pair of terminals connect to the field windings, which surround each pole and are normally in series, these windings provide the m.m.f to set up the flux in the air-gap under the poles. The main terminals convey the ‘torque-producing’ or ‘work’ current to the brushes which make sliding contact to the armature winding on the rotor.

The supply to the field (the flux-producing part of the motor) is separate from that for the armature, hence the description ‘separately excited’. Small DC motors are used in tools, toys, and appliances [6].

(c) Rotary motor

Nothing in the principle of any of the motors described above requires that the iron (steel) portions of the rotor actually rotate. If the soft magnetic material of the rotor is made in the form of a cylinder, then (except for the effect of hysteresis) torque is exerted only on the windings of the electromagnets [7].

(d) Servo motor

A servo motor is a motor, very often sold as a complete module, which is used within a position-control or speed-control feedback control system mainly control valves, such as motor-operated control valves. Servomotors are used in applications such as machine tools, pen plotters, and other process systems [7].

(e) Stepper motor

The stepper or stepping motor is a motor that produces rotation through equal angles, the so-called steps, for each digital pulse supplied to its input they are frequently used when precise rotations are required. In a stepper motor an internal rotor containing PMs or a magnetically soft rotor with salient poles is controlled by a set of external magnets that are switched electronically. A stepper motor may also be thought of as a cross between a DC electric motor and a rotary solenoid. As each coil is energized in turn, the rotor aligns it self with the magnetic field produced by the energized field winding [7].

(f) Linear motor

A linear motor is essentially any electric motor that has been "unrolled" so that, instead of producing a torque (rotation), it produces a straight-line force along its length. Linear motors are most commonly induction motors or stepper motors. Linear motors are commonly found in many roller-coasters where the rapid motion of the motor less railcar is controlled by the rail [7].

2.7 Sensor

Sensors are devices that receive and respond to signals or stimulus. Any sensor is an energy converter. The process of sensing is a particular case of information transfer, and any transmission of information requires transmission of energy. The purpose of a sensor is to respond to some kind of an input physical property (stimulus) and to convert it into an electrical signal which is compatible with electronic circuits. Sensors generally translate nonelectrical value into an electrical value. The electrical value is a signal which can be channeled, amplified, and modified by electronic devices. The sensor's output signal may be in the form of voltage or current [8]. Sensors Classifications are:

(a) **Passive sensor** which does not need any additional energy source and directly generates an electric signal in response to an external stimulus; that is, the input stimulus energy is converted by the sensor into the output signal. Such as thermocouple, photodiode, and piezoelectric sensors [9].

(b) **Active sensor** requires external power for their operation, which is called an excitation signal. That signal is modified by the sensor to produce the output signal. Such as thermistor and resistive strain gauge Some physical principles for Sensing are electric charges, fields, and potentials, capacitance, magnetism, induction, resistance, piezoelectric effect, hall effect and sound waves [9].

CHAPTER THREE

MODELING AND CIRCUIT COMPONENTS

3.1 Introduction

According to the collected data and the Suggested recommendations, the necessary devices and components were selected. And connected together in order to implement the project aims.

3.2 Components

The selected devices, there specifications and the way the components where connected based on the objectives of the project:

3.2.1 Arduino board

Arduino is an open-source prototyping platform in electronics based on easy-to-use hardware and software. It is a microcontroller based prototyping board which can be used in developing digital devices that can read inputs like finger on a button, touch on a screen, light on a sensor etc. and turning it in to output like switching on an LED, rotating a motor and make alarm through a speaker.

Arduino boards generally based on microcontrollers from Atmel Corporation like 8, 16 or 32 bit AVR architecture based microcontrollers. These boards can be programmed to perform different kind of operations by simply programming the microcontroller on board using a set of instructions for which, the Arduino board consists of a USB plug to communicate with your computer and a bunch of connection sockets that can be wired to external devices like motors, LEDs and sensors [10].

In order to program the Arduino board, we need to use (IDE) provided by Arduino. The Arduino IDE is based on processing programming language and supports C and C++. There are many types of Arduino boards available but all the boards have one thing in common, they can be programmed using the Arduino IDE. The reasons for different types of boards are different power supply requirements, connectivity options and their applications.

The Arduino Uno is a microcontroller board based on the ATmega328. It has 14 digital input/output pins (of which 6 can be used as Pulse Width Modulation (PWM) outputs), 6 analog inputs, a 16/ MHz crystal oscillator, a Universal Serial Bus(USB) connection, a power jack, an ICSP header, and a reset button. It can be powered by USB cable connected to the computer or AC-to-DC adapter or battery to get started. The board contents are:

Power USB Arduino board can be powered by using the USB cable from your computer. All you need to do is connect the USB cable to the USB connection.

Power (Barrel Jack) Arduino boards can be powered directly from the AC mains power supply by connecting it to the Barrel Jack.

Voltage Regulator The function of the voltage regulator is to control the voltage given to the Arduino board and stabilize the DC voltages used by the processor and other elements.

Crystal Oscillator The Arduino calculates time by using the crystal oscillator. The frequency of it is 16 MHz

Arduino Reset A button used to reset the program,

Pins (3.3, 5, GND, VIN) 3.3V − Supply 3.3 output volt, 5V − Supply 5 output volt, GND (Ground) − There are several GND pins on the Arduino, any of which can be used to ground your circuit, VIN − this pin also can be used to power the Arduino board from an external power source.

Analog pins The Arduino UNO board has six analog input pins A0 through A5. These pins can read the signal from an analog sensor and convert it into a digital value that can be read by the microprocessor.

Main microcontroller Each Arduino board has its own microcontroller. The main IC on the Arduino is slightly different from board to board. The microcontrollers are usually of the ATMEL Company.

ICSP pin Mostly, ICSP is an AVR, a tiny programming header for the Arduino consisting of MOSI, MISO, SCK, RESET, VCC, and GND. It is often referred to as an SPI, which could be considered as an "expansion" of the output. Actually, you are slaving the output device to the master of the SPI bus.

Power LED indicator This LED should light up when you plug your Arduino into a power source to indicate that your board is powered up correctly.

TX and RX LEDs They appear in two places on the Arduino UNO board. First, at the digital pins 0 and 1, to indicate the pins responsible for serial communication. Second, the TX and RX led. The TX led flashes with different speed while sending the serial data. The speed of flashing depends on the baud rate used by the board. RX flashes during the receiving process.

Digital I/O The Arduino UNO board has 14 digital I/O pins (of which 6 provide PWM output). These pins can be configured to work as input digital pins to read logic values (0 or 1) or as digital output pins to drive different modules like LEDs, relays, etc. The pins labeled “~” can be used to generate PWM.

AREF Stands for Analog Reference. It is sometimes, used to set an external reference voltage (between 0 and 5 Volts) as the upper limit for the analog input pins [10]. Arduino is programmed by the required instructions which is attached at the appendix to complete the specified tasks through the USB capsule. It's connected to the control circuit (push buttons), ultrasonic sensor, the stepper motor and the LEDs.

The Arduino receives the control signal from both the push buttons and the ultrasonic sensor, then processes the data and based on the processed data it sends the control signal to the motor through the motor driver and the LEDs. Table 3.1shows Arduino Uno data sheet, and figure 3.1 shows the Arduino Uno.

Table 3.1: Arduino Uno data sheet

Microcontroller	ATmega328
Operating Voltage	5V
Input Voltage (recommended)	7-9V
Input Voltage (limits)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	6
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328) (0.5 KB used by bootloader)
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Clock Speed	16 MHz

Figure 3.1 : The Arduino Uno

3.2.2 Stepper motor

A stepper motor is an electric motor whose its shaft rotates by performing steps by moving by a fixed amount of degrees. Stepper motors have a stationary part (the stator) and a moving part (the rotor). On the stator, there are teeth on which coils are wired, while the rotor is either a permanent magnet or a variable reluctance iron core [7]. The motor is connected to the Arduino board through the motor driver. And it provides the necessary mechanical power to lift the box. Table 3.2 shows stepper motor 28BYJ-48 specifications. And figure 3.2 shows 28BYJ-48 stepper motor.

Table 3.2: Stepper motor 28BYJ-48 specifications

Rated voltage	5VDC
Number of Phase	4
Speed Variation Ratio	1/16
Stride Angle	5.625°/64
Frequency	100

DC resistance	50Ω±7%(25℃)
Idle In-traction Frequency	> 600Hz
Idle Out-traction Frequency	> 1000Hz
In-traction Torque	>34.3mN.m(120Hz)
Self-positioning Torque	>34.3mN.m
Friction torque	600-1200 gf.cm
Pull in torque	300 gf.cm
Insulated resistance	>10MΩ(500V)
Insulated electricity power	600VAC/1mA/1s
Insulation grade	A
Noise	<35dB(120Hz,No load,10cm)
Rise in Temperature	<40K(120Hz)
Mode	28BYJ-48 – 5V

Figure 3.2: Stepper motor 28BYJ-48

3.2.3 Motor driver

Acts as an interface between the motors and the control circuits. Motor require high amount of current whereas the controller circuit works on low current signals. ULN2003 is used in this project, it Contains 7 high-voltage and high current Darlington pairs, each pair is rated for 50V and 500mA ,Input pins can be triggered by +5V, all seven Output pins can be connected to gather to drive loads up to (7×500mA) ~3.5A and can be directly controlled by logic devices. The driver is connected to the Arduino analog pins from A2 to A6, VCC, and ground and connected to the motor terminals. The function of motor drivers is to take a low-current control signal and then turn it into a higher-current signal that can drive a motor. Figure 3.3 shows the ULN2003 motor driver.

Figure 3.3: ULN2003 motor driver

3.2.4 Ultrasonic sensor

An ultrasonic sensor is an electronic device that measures the distance of a target object. They work by exciting an acoustic transducer with voltage pulses, causing the transducer to vibrate ultrasonically. These oscillations are directed at a target and by measuring the time for the echo to return to the transducer, the target’s distance is calculated. Ultrasonic sensors generally provide an accuracy of 1 mm a distances ranging from 100 mm to 6,000 mm (over 19 feet). Ultrasonic sensors provide precise no-touch presence/absence sensing and distance sensing or tracking. These sensors are used to solve the complex tasks such as object detection or leveling even most measurement with millimeter precision, because their measuring method works reliably under almost all conditions. They have no difficulty working with round objects, moving targets [9].

The sensor consists of three parts transmitters, receivers and transceivers. Transmitters convert electrical signals into ultrasound, receivers convert ultrasound into electrical signals, and transceivers can both transmit and receive ultrasound [9].

Ultrasonic sensor HC-SR04 is used to determine the distance between the box and the floor and sends a signal to the Arduino board. And it is connected to the Arduino analog pins directly. Table 3.3 shows the Ultrasonic sensor HC-SR04 data sheet. And Figure 3.4 shows the HC-SR04 ultrasonic sensor.

Table 3.3: Ultrasonic sensor HC-SR04 specifications

Working Voltage	DC 5V
Working Current	15mA
Working Frequency	40Hz
Max Range	4m
Min Range	2cm
Measuring Angl**e**	15 degree
Trigger Input Signal	10uS TTL pulse
Echo Output Signal Input	TTL lever signal and the range in proportion
Dimension	45*20*15mm

Figure 3.4: Ultrasonic sensor

3.2.5 Push button

Push button is a simple type of switch that controls an action in a machine or some type of process. Most of the time, the buttons are plastic or metal. They can be shown as normally open or normally closed, most are momentary contact devices in that they make or break connection only as long as pressure is applied to them. When the pressure is removed, they return to their normal position.

Push button switches have three parts. The actuator, stationary contacts, and the grooves. The actuator will go all the way through the switch and into a thin cylinder at the bottom. Inside is a movable contact and spring. When someone presses the button, it touches with the stationary contacts, causing the action to take place. In some cases, the user needs to keep holding the button, or to press it repeatedly, for an action to take place. With other push buttons, a latch connects and keeps the switch on until the user presses the button again.

Mini push button is used in this project with 6mm × 6mm × 6mm Dimensions, Rated Load DC 12V 50mA, Mounting Type is through hole, push buttons were connected to the 5v pin and protected by resistors then to the Arduino digital pins from D4 to D12. And they provide the control signal from the users to the Arduino board in the form of digital input. Figure 3.5 shows mini push button.

Figure 3.5: Mini push button

3.2.6 LEDs

Light emitting diode is an electric component that emits light when the electric current flows through it. It is a light source based on semiconductors. A Light Emitting Diode consists of three layers: p-type semiconductor, n-type semiconductor and depletion layer. The p-type semiconductor and the n-type semiconductor are separated by a depletion region or depletion layer. Stander LEDs works on 80mW Power Dissipation, 100mA peak forward current, 20mA continuous forward current and 5V reverse voltage. They are connected to the board analog pins (A0 and A1) then to the ground and they are protected by resistors. They were used as indicators for the operation. Figure 3.6 shows red and green LEDs.

Figure 3.6: Red and green LEDs

3.2.7 Resistors

Is an electrical component that limits or regulates the flow of electrical current in an electronic circuit? Resistors can also be used to provide a specific voltage for an active device.
The resistors are connected between the 5V pin in the board the push button also connected between the board digital pins and the LEDs. And there is to

protect the components of the circuit from high current. Figure 3.7 shows the electrical resistor.

Figure 3.7: Electrical resistor

3.4 Block Diagram

This block diagram shows how the operation was done:

Step 1: send the signal from the push buttons, **step 2:** ultrasonic measure then distance of the box to detect On which floor it, **step 3:** Arduino receives the data to make a decision and send control signal to the motor through motor driver, **step4:** the motor moves the box according to the Arduino instructions. Figure 3.8 shows the block diagram of project.

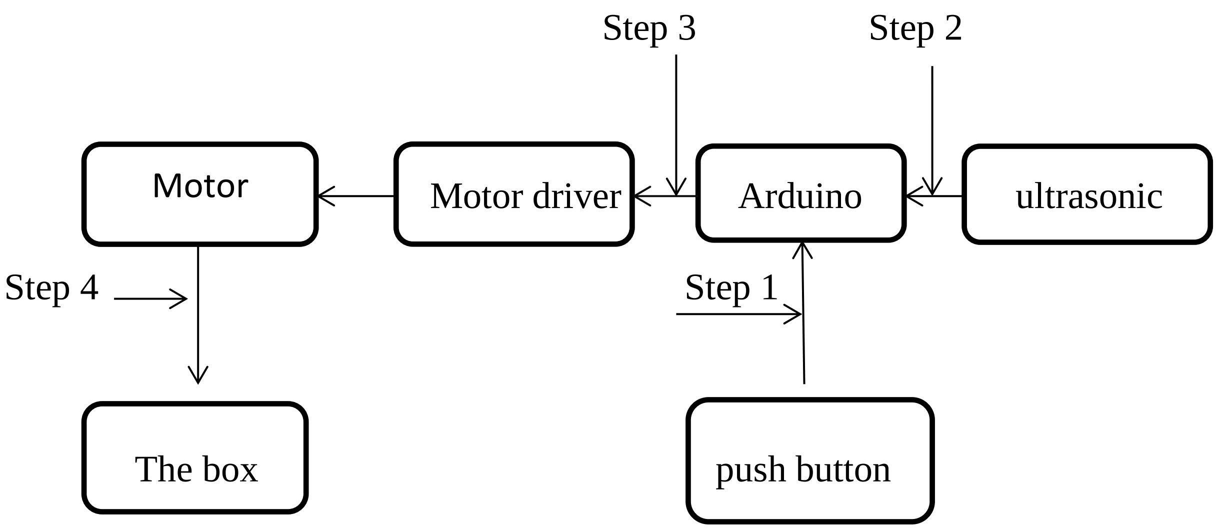

Figure 3.8: The block diagram of project

CHAPTER FOUR

SIMULATION AND OPERATION

4.1 Introduction

The simulation which is an approximate imitation of the operation of a process or system; that represents its operation over time and the practical implementation which is the prototype of the project shows how the project is operated. Showing the how the devices are connected and the sequence of the process, the program and how it was implemented.

4.2 Simulation

Software program was used to simulate the project; all the devices and connections were setup in the program to show how the project will operate.

4.2.1 Simulation software

Proteus is a simulation and design software tool developed by Labcenter Electronics for Electrical It is a software suite containing schematic, simulation as well as PCB designing. ISIS is the software used to draw schematics and simulate the circuits in real time. The simulation allows human access during run time, thus providing real time simulation. ARES is used for PCB designing. It has the feature of viewing output in 3D view of the designed PCB along with components. The designer can also develop 2D drawings for the product. The steppes to simulate the project:

Step 1: Open ISIS software and select New design in File menu. Figure 4.1 shows opening the ISIS.

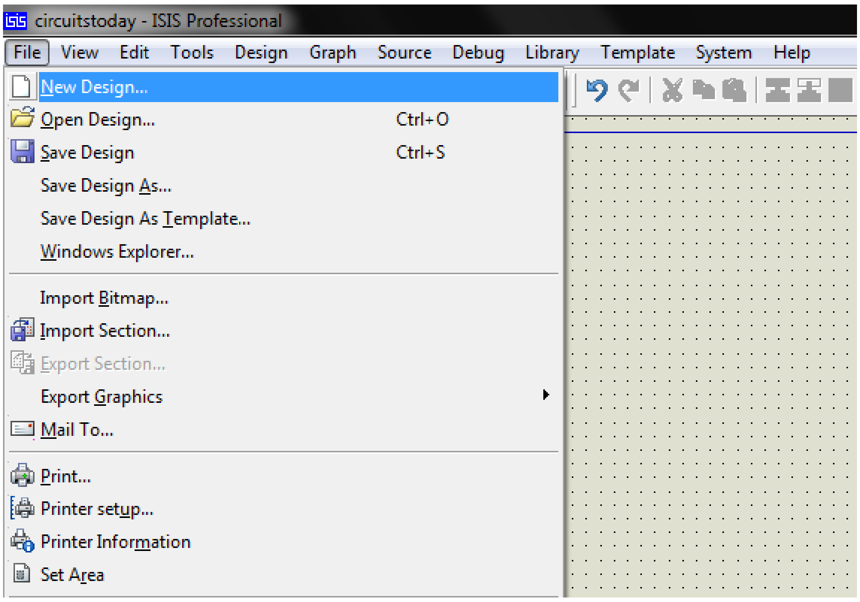

Figure 4.1: Opening the ISIS

Step 2: a Pop-Up appears asking to select the template. select default or according to the layout size of the circuit. Figure 4.2 shows selection of a templet

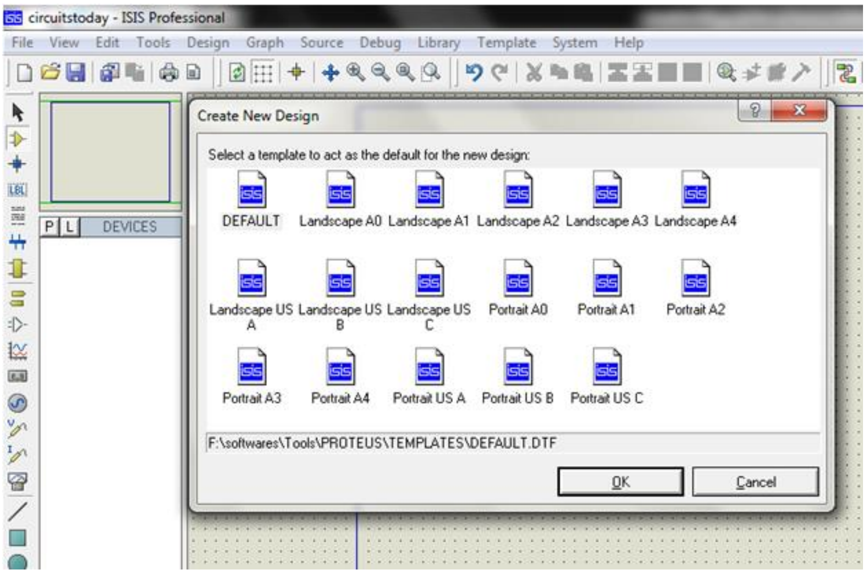

Figure 4.2: Selection of a templets

Step 3: an untitled design sheet will be opened. Figure 4.3 shows the main sheet

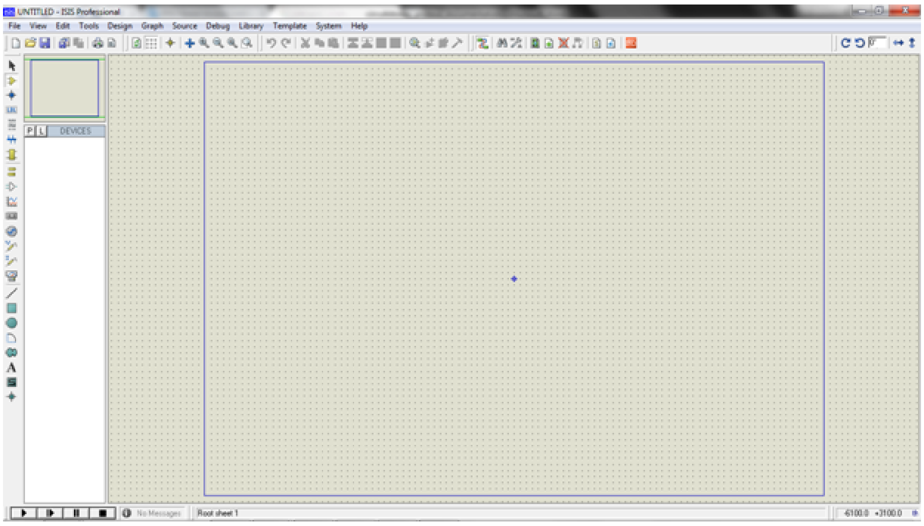

Figure 4.3: The main sheet

4.2.2 Simulation steps

Step 1: a circuit of push buttons and resistors was connected. Figure 4.4 shows how push buttons and resistors are connected.

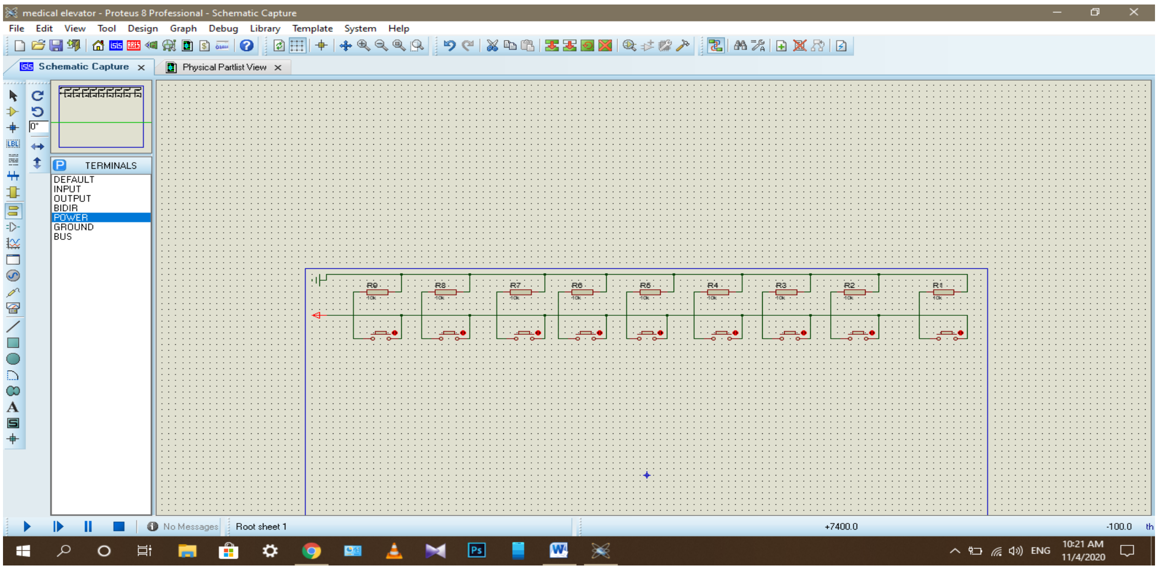

Figure 4.4: Push buttons and resistors connection

Step 2: push buttons were connected to the Arduino digital input pins. Figure 4.5 shows the connection of push buttons with the Arduino.

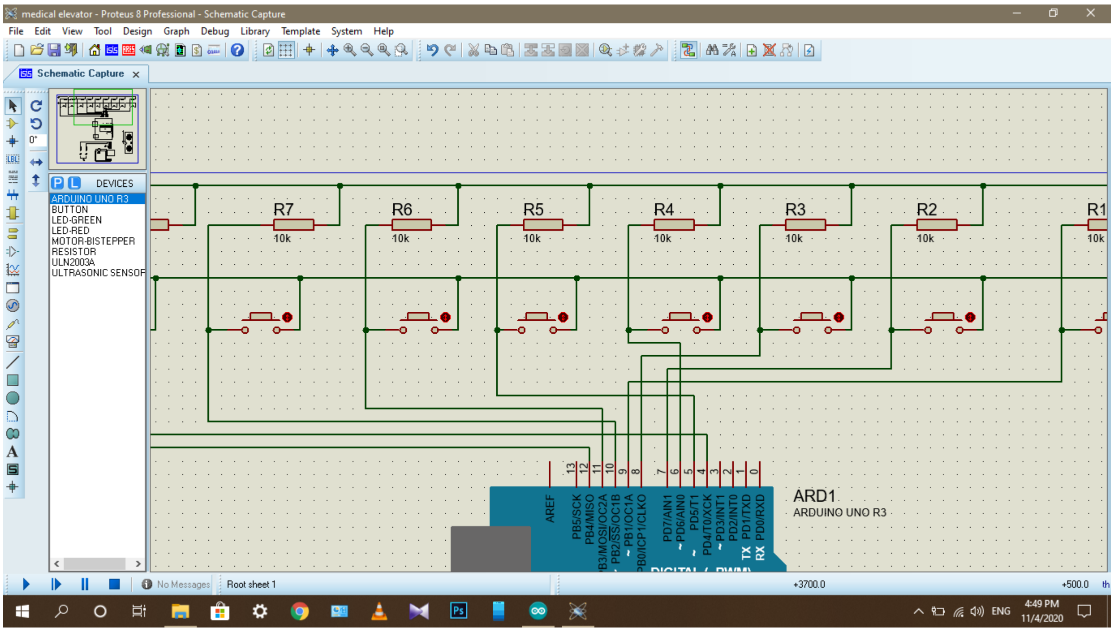

Figure 4.5: Connection of push buttons to the Arduino

Step 3: ultrasonic sensor to the Arduino digital pins. Figure 4.6 shows the connection between ultrasonic sensor and the Arduino.

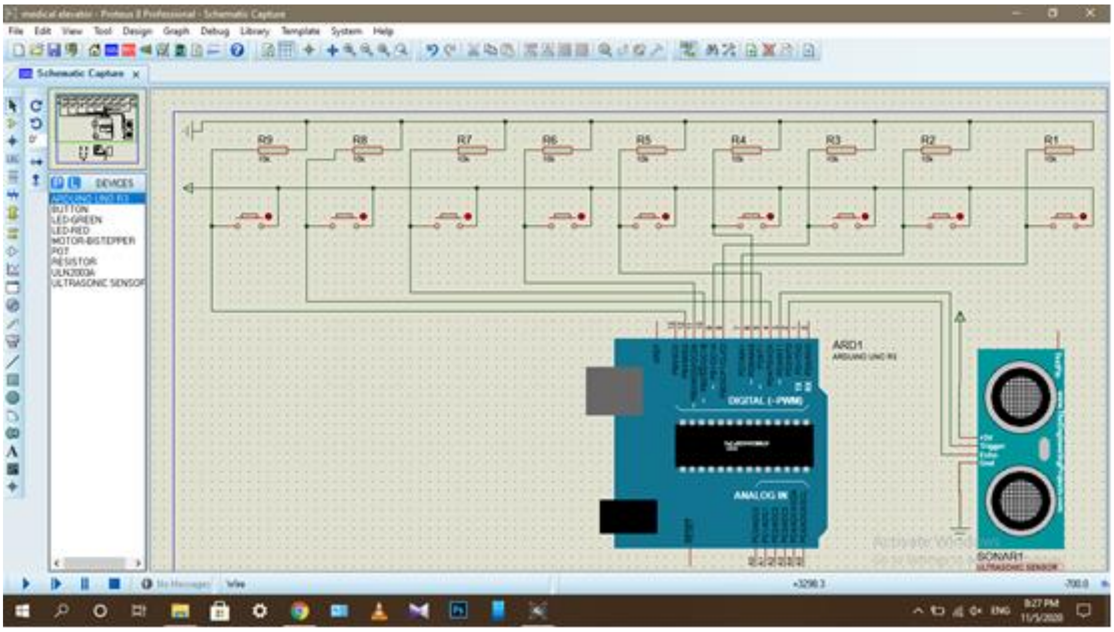

Figure 4.6: Connecting ultrasonic sensor to Arduino

Step 4: motor connected to the Arduino through motor driver to the analog pins of the board also the LEDs where connected. Figure 4.7 shows how the motor is connected to the Arduino.

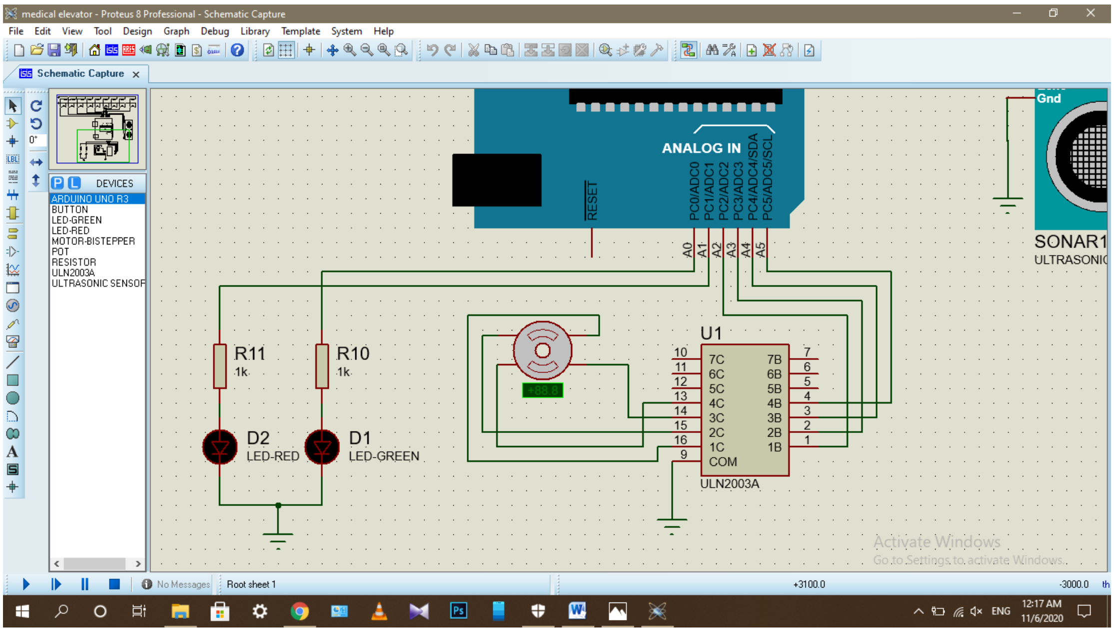

Figure 4.7: Motor connection to the Arduino

4.2.4 Simulation model

The devices were compiled together, and the process was simulated by loading the program to the Arduino board and the components were tested. Figure 4.8 shows the complete simulation sample.

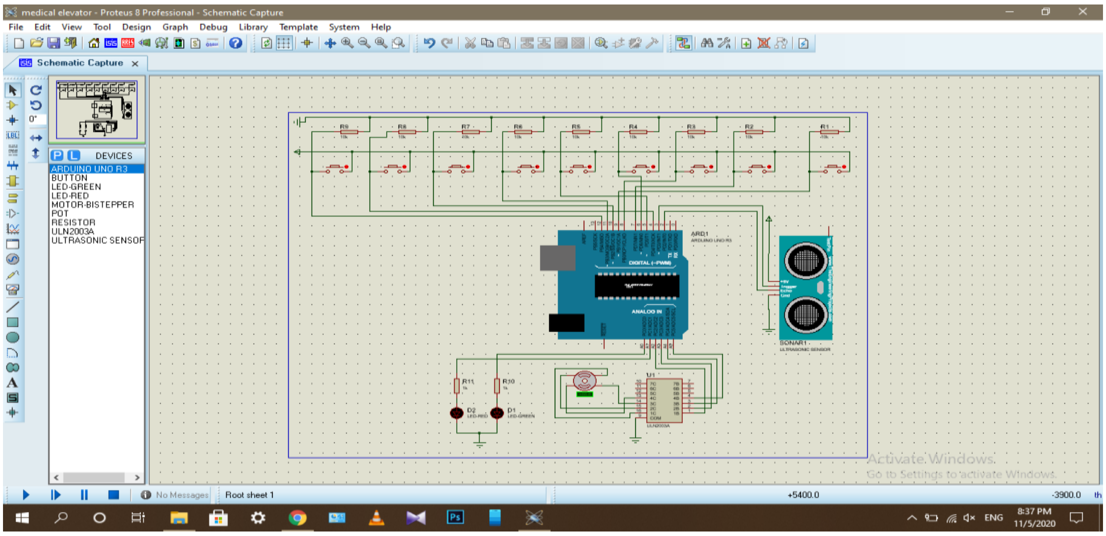

Figure 4.8: The complete simulation sample

4.3 Operation

The medication request passes through the HIS which links the nurse stations and the pharmacy. And to start the process of medication delivery for the each station a button is pressed, each station have two types of buttons one for normal request button and another for critical request button.

The pharmacist puts the orders into the cabin and then starts the process, the pharmacy have three buttons to start the process, normal request and critical request buttons to choose the type of the request and the start button which start the operation.

Normal buttons makes normal requests, and if there is multiple normal requests the operation priority will be closest first

Critical buttons makes critical requests and if there is normal request and critical request the priority will be the critical first, or if there are multiple critical requests the priority will be the closest first.

The red LED will indicate that an operation is under process and the green LED indicate that the elevator has reached the desired floor.

After request button is pressed the Arduino will save in which floor the request was made. Then after start button is pressed the Arduino will send control signal to move the motor through the motor driver which drive the box to the saved floors and the red LED indicator will be turned on, the Arduino uses ultrasonic sensor to detect where the box is and its destination from the saved floor.

Each time the box arrive to saved floor the motor will stop also green LED will be turned on and the red LED indicator will be turned off for specified time programmed in the Arduino, then checks if there is still saved uncomplete requests, if not, the box will return to the pharmacy.

The program which implements the process is attached in appendix A. The operation was implemented using the project model. Figure 4.9 shows the project model.

Figure 4.9: The project model

After the process of making the order the elevator completes the task and wait for specific time on the floor and the green LED indicates the box arrival. Figure 4.10 shows the elevator delivering an order.

Figure 4.10: The elevator delivering an order

After that the system checks if there is still uncompleted orders. If not the elevator will return to the ground floor. Figure 4.11 shows the elevator returning to the ground floor.

Figure 4.11: the elevator returning to the ground floor

CHAPTER FIVE

CONCLUTION AND RECCOMENTATIOS

6.1 Conclusion

In accordance to the objectives that lead to conduct this study, which was simulating, implementing and determined the necessary devices to realize an elevator which transmits the Medications or Medical consumables in the Hospital from The pharmacy to the nurse station.

According to the problem of transmitting medications in hospitals in conclusion, this objectives and problems resulted in the creation of this project which will ease and rapid the process of delivering the Medical consumables, the hospital employs fewer employees, reduce the crowding in the hospitals also limits the direct contact and interact between people which decreases the infection rate of covied-19.

6.2 Recommendations

Recommendation for implementation and enhancement are:

- Adding away to lock the system for only authorized people.
- Improving the software programming to receive the orders directly from the HIS.
- Design the project with high quality materials.
- Expanding the system by adding another box in new shaft to serve heavy duty in large hospitals.

REFERECES

[1] Norman S. Nise, Control systems engineering. Seventh edition. John Wiley & Sons. Inc 2015.

[2] S.K Mandalx, Microprocessors and Microcontrollers Architecture Programming Interfacing using 8085, 8086, and 8051. Tata Mcgraw Hill, 2011 .

[3] Ghosal Subrata, 8051 Microcontroller: Internals, Instructions, Programming and Interfacing. 1st Edition. Pearson, 2010.

[4] EurIng Ian C. Thrner, Engineering Applications of Pneumatics and Hydraulics. Butterworth-Heinemann 2002.

[5] Health Facilities Scotland, Pneumatic tube transport systems Part A: Overview and management responsibilities. A Division of NHS National Services Scotland, 2011.

[6] Theodore Wildi, Electrical Machines, Drives, and Power Systems. 5th edition. Hall College Div. 2002.

[7] V.K. Mehta & Mehta Rohit. S Chand, principles of electrical machines. Reprint Edn. 2006.

[8] Jon S. Wilson, sensor technology handbook. Newnes. 2004.

[9] Jacob Fraden, handbook of modern sensors: physics, designs, and applications. 3rd edition. Springer. 2003.

[10] Alan Trevennor, practical AVR microcontrollers games, gadgets, and home automation with the microcontroller used in Arduino. 1st Edition. Apress . 2012.

APPENDIX

THE PROGRAM CODE

```
#include <Ultrasonic.h>

#include <Stepper.h>

Ultrasonic ultrasonic(2, 3);

int an1 = 7;

int ae1 = 8;

int an2 = 6;

int ae2 = 5;

int an3 = 11;

int ae3 = 10;

int an4 = 4;

int ae4 = 12;

int start = 9;

int gr = A0;

int red = A1;

int n2=0,e2=0,n3=0,e3=0,n4=0,e4=0,s1=0,n1=0,e1=0;

const int stepsPerRevolution = 400;

int cd=0;

Stepper myStepper(stepsPerRevolution, A2, A4, A3, A5);


void setup()
```

```
{
  Serial.begin(9600);
   pinMode(an1, INPUT);
  pinMode(ae1, INPUT);
 pinMode(an2, INPUT);
  pinMode(ae2, INPUT);
  pinMode(an3, INPUT);
  pinMode(ae3, INPUT);
  pinMode(an4, INPUT);
  pinMode(ae4, INPUT);
  pinMode(start, INPUT);
  pinMode(gr, INPUT);
  pinMode(red, INPUT);
  analogWrite(gr,0);
  analogWrite(red,250);
while (  ultrasonic.distanceRead() <=35) {
Serial.println(ultrasonic.distanceRead());
  myStepper.setSpeed(70);
  myStepper.step(stepsPerRevolution);
}
}

void loop()
```

```
{
  int xx = ultrasonic.distanceRead();
 // Serial.println(xx);


 int nn1 = digitalRead(an1);
 int ee1 = digitalRead(ae1);
 int nn2 = digitalRead(an2);
 int ee2 = digitalRead(ae2);
 int nn3 = digitalRead(an3);
 int ee3 = digitalRead(ae3);
 int nn4 = digitalRead(an4);
 int ee4 = digitalRead(ae4);
 int s = digitalRead(start);


if (nn1 ==1)
n1=1;
if (ee1 ==1)
e1=1;
if (nn2 ==1)
n2=1;
if (ee2 ==1)
e2=1;
if (nn3 ==1)
```

```
n3=1;

if (ee3 ==1)

e3=1;

if (nn4 ==1)

n4=1;

if (ee4 ==1)

e4=1;

if (s==1)

s1 =1;



if (s1==1 )

{


  //*********n2,e3,e4*********************


if (n2==1&n4==0&e4==1&e2==0&e3==1&n3==0&e1==1&n1==0)

  {

    //cd=1;

    while (  ultrasonic.distanceRead() >14) {

  myStepper.setSpeed(70);

  myStepper.step(-stepsPerRevolution);

      analogWrite(red,250);
```

```
    }
   analogWrite(gr,250);
   analogWrite(red,0);
    delay(3000);
    analogWrite(gr,0);
while (  ultrasonic.distanceRead() >5) {
  myStepper.setSpeed(70);
  myStepper.step(-stepsPerRevolution);
  analogWrite(red,250);
  }
   analogWrite(gr,250);
   analogWrite(red,0);
    delay(3000);
   analogWrite(gr,0);
  while (  ultrasonic.distanceRead() <=23) {
  myStepper.setSpeed(70);
  myStepper.step(stepsPerRevolution);
  analogWrite(red,250);
    }
  analogWrite(gr,250);
  analogWrite(red,0);
   delay(3000);
   analogWrite(gr,0);
```

```
    while (  ultrasonic.distanceRead() <35) {
  myStepper.setSpeed(70);
  myStepper.step(stepsPerRevolution);
  analogWrite(red,250);
    }
   n2=0;n3=0;s1=0;e4=0;n4=0;cd=0;s1=0;e1=0;

}

  //*********e2,e3,e4,n2,n3,n4********************

if (((e4==1&e3==1&e2==1)||(n4==1&n3==1&n2==1))&(n1==1||e1==1))
  {
    while (  ultrasonic.distanceRead() >23) {
  myStepper.setSpeed(70);
  myStepper.step(-stepsPerRevolution);
  analogWrite(red,250);
    }
    analogWrite(gr,250);
    analogWrite(red,0);
    delay(3000);
    analogWrite(gr,0);
while (  ultrasonic.distanceRead() >14) {
```

```
  myStepper.setSpeed(70);
  myStepper.step(-stepsPerRevolution);
  analogWrite(red,250);
    }
    analogWrite(gr,250);
    analogWrite(red,0);
    delay(3000);
    analogWrite(gr,0);
while (  ultrasonic.distanceRead() >5) {
  myStepper.setSpeed(70);
  myStepper.step(-stepsPerRevolution);
  analogWrite(red,250);
    }
    analogWrite(gr,250);
    analogWrite(red,0);
    delay(3000);
    analogWrite(gr,0);
    while (  ultrasonic.distanceRead() <35) {
  myStepper.setSpeed(70);
  myStepper.step(stepsPerRevolution);
  analogWrite(red,250);
    }
   n2=0;n4=0;s1=0;e2=0;e4=0,n2=0;e3=0;n1=0;
```

```
  // n2=0;n3=0;
}

  //*********n2,n3,e2,e3*********************

if (((e3==1&e2==1)||(n3==1&n2==1))&(n1==1||e1==1))
  {
    while (  ultrasonic.distanceRead() >23) {
  myStepper.setSpeed(70);
  myStepper.step(-stepsPerRevolution);
  analogWrite(red,250);
    }
    analogWrite(gr,250);
    analogWrite(red,0);
    delay(3000);
    analogWrite(gr,0);
while (  ultrasonic.distanceRead() >14) {
  myStepper.setSpeed(70);
  myStepper.step(-stepsPerRevolution);
  analogWrite(red,250);
    }
    analogWrite(gr,250);
    analogWrite(red,0);
```

```
    delay(3000);
    analogWrite(gr,0);
    while (  ultrasonic.distanceRead() <=35) {
  myStepper.setSpeed(70);
  myStepper.step(stepsPerRevolution);
  analogWrite(red,250);
    }
   n2=0;n3=0;s1=0;e2=0;e3=0;n1=0;
  // n2=0;n3=0;
}

 //*********n2,n4,e2,e4********************

if (((e4==1&e2==1)||(n4==1&n2==1))&(n1==1||e1==1))
  {
    while (  ultrasonic.distanceRead() >23) {
  myStepper.setSpeed(70);
  myStepper.step(-stepsPerRevolution);
  analogWrite(red,250);
    }
    analogWrite(gr,250);
    analogWrite(red,0);
    delay(3000);
```

```
    analogWrite(gr,0);
while (  ultrasonic.distanceRead() >5) {
  myStepper.setSpeed(70);
  myStepper.step(-stepsPerRevolution);
      analogWrite(red,250);
    }
    analogWrite(gr,250);
    analogWrite(red,0);
    delay(3000);
    analogWrite(gr,0);
    while (  ultrasonic.distanceRead() <35) {
  myStepper.setSpeed(70);
  myStepper.step(stepsPerRevolution);
  analogWrite(red,250);
    }
   n2=0;n4=0;s1=0;e2=0;e4=0;n1=0;
      // n2=0;n3=0;
}

 //*********n3,n4,e3,e4********************

if (((e4==1&e3==1)||(n4==1&n3==1))&(n1==1||e1==1))
  {
```

```
  while (  ultrasonic.distanceRead() >14) {
 myStepper.setSpeed(70);
 myStepper.step(-stepsPerRevolution);
 analogWrite(red,250);
  }
  analogWrite(gr,250);
  analogWrite(red,0);
  delay(3000);
  analogWrite(gr,0);
while (  ultrasonic.distanceRead() >5) {
 myStepper.setSpeed(70);
 myStepper.step(-stepsPerRevolution);
 analogWrite(red,250);
  }
  analogWrite(gr,250);
  analogWrite(red,0);
  delay(3000);
  analogWrite(gr,0);
  while (  ultrasonic.distanceRead() <35) {
 myStepper.setSpeed(70);
 myStepper.step(stepsPerRevolution);
 analogWrite(red,250);
  }
```

```
   n3=0;n4=0;s1=0;e3=0;e4=0;n1=0;

  // n2=0;n3=0;
}

  //*********n2,e3********************

if (n2==1&n4==0&e3==1&e2==0&e4==0&n3==0&e1==1&n1==0)
  {
    //cd=1;
    while (  ultrasonic.distanceRead() >14) {
  myStepper.setSpeed(70);
  myStepper.step(-stepsPerRevolution);
  analogWrite(red,250);
    }
    analogWrite(gr,250);
    analogWrite(red,0);
    delay(3000);
    analogWrite(gr,0);
while (  ultrasonic.distanceRead() <=23) {
  myStepper.setSpeed(70);
  myStepper.step(stepsPerRevolution);
  analogWrite(red,250);
```

```
    }
    analogWrite(gr,250);
    analogWrite(red,0);
    delay(3000);
    analogWrite(gr,0);
    while (  ultrasonic.distanceRead() <35) {
  myStepper.setSpeed(70);
  myStepper.step(stepsPerRevolution);
  analogWrite(red,250);
    }
   n2=0;n3=0;s1=0;e3=0;n4=0;cd=0;s1=0;e1=0;
  // n2=0;n3=0;
}

  //*********n2,e4********************

if (n2==1&n4==0&e3==0&e2==0&e4==1&n3==0&e1==1&n1==0)
  {
    //cd=1;
    while (  ultrasonic.distanceRead() >5) {
  myStepper.setSpeed(70);
  myStepper.step(-stepsPerRevolution);
  analogWrite(red,250);
```

```
    }
    analogWrite(gr,250);
    analogWrite(red,0);
    delay(3000);
    analogWrite(gr,0);
while (  ultrasonic.distanceRead() <23) {
  myStepper.setSpeed(70);
  myStepper.step(stepsPerRevolution);
  analogWrite(red,250);
    }
    analogWrite(gr,250);
    analogWrite(red,0);
    delay(3000);
    analogWrite(gr,0);
    while (  ultrasonic.distanceRead() <35) {
  myStepper.setSpeed(70);
  myStepper.step(stepsPerRevolution);
  analogWrite(red,250);
    }
   n2=0;n3=0;s1=0;e3=0;n4=0;cd=0;e4=0;s1=0;e1=0;
  // n2=0;n3=0;
}
```

```
 //*********n3,e4*********************

if (n2==0&n4==0&e4==1&e2==0&e3==0&n3==1&e1==1&n1==0)
  {
    //cd=1;
    while (  ultrasonic.distanceRead() >5) {
  myStepper.setSpeed(70);
  myStepper.step(-stepsPerRevolution);
  analogWrite(red,250);
    }
     analogWrite(gr,250);
     analogWrite(red,0);
     delay(3000);
     analogWrite(gr,0);
while (  ultrasonic.distanceRead() <14) {
  myStepper.setSpeed(70);
  myStepper.step(stepsPerRevolution);
  analogWrite(red,250);
    }
    analogWrite(gr,250);
    analogWrite(red,0);
    delay(3000);
```

```
    analogWrite(gr,0);
    while (  ultrasonic.distanceRead() <35) {
  myStepper.setSpeed(70);
  myStepper.step(stepsPerRevolution);
  analogWrite(red,250);


    }
   n2=0;n3=0;s1=0;e4=0;n4=0;cd=0;s1=0;e1=0;


}


  //*********n2,e2*********************


  if ((n2==1||e2==1)&(n1==1||e1==1))
  {
    while (  ultrasonic.distanceRead() >23) {
  myStepper.setSpeed(70);
  myStepper.step(-stepsPerRevolution);
  analogWrite(red,250);
    }
    analogWrite(gr,250);
    analogWrite(red,0);
    delay(3000);
```

```
    analogWrite(gr,0);
    while (  ultrasonic.distanceRead() <=35) {
  myStepper.setSpeed(70);
  myStepper.step(stepsPerRevolution);
  analogWrite(red,250);
    }
    n2=0;e2=0;s1=0;cd=1;n1=0;
}

  //*********n3,e3********************



 if ((n3==1||e3==1)&(n1==1||e1==1))
  {
    while (  ultrasonic.distanceRead() >14) {
  myStepper.setSpeed(70);
  myStepper.step(-stepsPerRevolution);
  analogWrite(red,250);
    }
    analogWrite(gr,250);
    analogWrite(red,0);
    delay(3000);
    analogWrite(gr,0);
```

```
    while (  ultrasonic.distanceRead() <=35) {
  myStepper.setSpeed(70);
  myStepper.step(stepsPerRevolution);
  analogWrite(red,250);
    }
   n3=0;e3=0; s1=0;n1=0;
}
 //*********n4,e4********************



if ((n4==1||e4==1)&(n1==1||e1==1))
  {
    while (  ultrasonic.distanceRead() >5) {
  myStepper.setSpeed(70);
  myStepper.step(-stepsPerRevolution);
  analogWrite(red,250);
    }
    analogWrite(gr,250);
    analogWrite(red,0);
    delay(3000);
    analogWrite(gr,0);
    while (  ultrasonic.distanceRead() <=35) {
  myStepper.setSpeed(70);
```

```
  myStepper.step(stepsPerRevolution);

  analogWrite(red,250);

    }

   n4=0;e4=0; s1=0;n1=0;

}

}

 analogWrite(red,0);

  }
```