BAR-CODE SCANNER SYSTEM ROBOTIC ARM PRODUCT PICK AND PLACE

A thesis
by

Md Asaduzzaman;	ID: BME 1801014420
Md Rabiul Islam;	ID: BME 1901017364
Md. Kabir Mollah;	ID: BME 1901017436
Zamilur Rahaman;	ID: BME 1901017444
Shirajul Islam;	ID : BME 1901017531

Department Of Mechanical Engineering
SONARGAON UNIVERSITY (SU)

JANUARY, 2023

Bar-code Scanner System Robotic Arm Product Pick and Place

A Thesis

by

Md Asaduzzaman
Student No.: BME1801014420

In Cooperation With

Md Rabiul Islam
Student No.: BME1901017364

Md.Kabir Mollah
Student No.:BME 1901017436

Zamilur Rahaman
Student No.:BME1901017444

Shirajul Islam
Student No.:BME1901017531

Supervisor: **Shahinur Rahman**

Lecturer

Submitted to the

DEPARTMENT OF MECHANICAL ENGINEERING

SONARGAON UNIVERSITY (SU)

In partial fulfillment of the requirements for the award of the degree

of

Bachelor of Science in Mechanical Engineering

JANUARY 2023

Acknowledgement

First, we started in the name of almighty Allah. This thesis is accomplished under the supervision of **Shahinur Rahman**, Lecturer, Department of Mechanical, Sonargaon University. It is a great pleasure to acknowledge our profound gratitude and respect to our supervisor for this consistent guidance, encouragement, helpful suggestion, constructive criticism and endless patience through the progress of this work. The successful completion of this thesis would not have been possible without his persistent motivation and continuous guidance.

The authors are also grateful to Md. Mostofa Hossain. Professor, Head of the Department of Mechanical Engineering and all respect teachers of the Mechanical Engineering Department for their co-operation and significant help for completing the thesis work successfully.

The authors are also grateful to Prof. Dr. Md. Alamgir Hossain, Departrment of Mechnical Engineering, Professor and Dean of Sonargaon University.

The authors are also grateful to Prof. Dr. Md. Abul Bashar. Vice-Chancellor of our university.

Abstract

Automation is creating revolution in the present industrial sector, as it reduces manpower and time of production. The pick and place robot is one of the technologies in manufacturing industries which is designed to perform pick and place operations. This paper mainly faces other control system design for the implementation of pick and place robot by using Arduino for any pick and placed function with bar-code based package sorting system, contains a belt conveyor system and an articulated robotic arm. The main objective of the project is to scan the bar-code of an object and placed it in a specific location. For Pick and Placed purposes, Robotic Arm are used. Robotic Arms are one of the most important parts of today's world. The Robotic Arm is controlled by an ARDUINO Nano and using Servo Motor it is created. With the help of the bar-code scanner, the Robotic Arm detect the object's bar-code and placed it into the equivalent position. One of the most important advantages of using a Robotic Arm is to increase the efficiency, productivity, and precision of the operations taking place.

Table of Contents

List of Figures

Chapter 1

Introduction

1.1 Introduction

Today, technology is developing in the same direction in line with rapidly increasing human needs. Automation is creating revolution in the present industrial sector, as it reduces manpower and time of production. The purpose of using robotic arm is to reduce errors and human efforts. As the robotic field is having an application in various department of engineering such as production, inspection, material handling etc. Barcode-based package sortation system mainly comprises with conveyor system, Automatic Identification and Data Capture (AIDC), and Industrial Robot. A complete warehouse automation system can drastically reduce the workforce required to run a facility, with human input required only for a few tasks, such as picking units of product from a bulk packed case. Even here, assistance can be provided with equipment such as pick-to-light units.

Smaller systems may only be required to handle part of the process. Sortation systems offer a highly accurate and efficient means of sorting, routing, consolidating, and diverting a wide range of package types. These systems are used for the purposes of order selection, processing, packaging, palletizing, storing, and shipping. Sortation systems can transport packages ranging in shape, weight, and size from sheets of paper to fully loaded packages. This systems generally fall into two groupings: linear sortation systems and looped sortation systems. Linear sortation system or line sorters sort in a straight line. An industrial robot is a reprogrammable, multifunctional manipulator designed to move materials, parts, tools, or specialized devices through various programmed motions for the performance of a variety of tasks. A robotic arm is a type of mechanical arm, usually programmable, with similar functions to a human arm. The basic components of an industrial robot are the manipulator, the end effector, the power supply, and the controller.

In any industrial processing plant, the ultimatum objective is to produce standard and high quality products and sell them at prices which make profits. These objectives can be achieved by efficiently designed and optimally controlled processing plants. A

process is set of equipment and materials specifically related to some manufacturing operations or system mechanism. The process control is the system of gathering the system information, quantifying the state of the process, standardizing the rules and guidelines to actuate the control elements accordingly [1]. For any process control system, such actions can be executed manually where only humans are involved or in a semi-automatic way where both machines and humans both are mutually involved or even without any involvement of human to perform such actions in automatic way by only using machines [2].

Different types of the products being manufactured by an automation process need to be identified, transport and store at ultimate destination locations. For such a process requires an object identifier and sorting system to recognize, classify and sort the different types of manufactured products. Different types of conveyor systems are in practice for multi objectives and applications specifically for transportation of objects and luggage's from last 50 years. Such conveyor systems are mostly driven by variable speed electric motors or similar type drives to move the parts and objects. The use of universal conveyor belts for moving, loading or unloading and transportation of objects is in practice for many years. The Vanderlande Industries of Belgian city uses the conveyor belts for object sorting applications especially for transportation and courier services [3].

The objects are sorted through barcode reading and according to weight dimensions [4]. Similarly the Track & Trace service provided by De Post -La Poste also employs barcode and weight mechanism to generate the item based billing for certain customers [5]. All these Industry uses weight and barcode mechanism to track and sort the objects, they does not possess any colour sensing mechanism to identify and sort the objects. The LEGO kit made by United States of America has the capacity to identify the object boxes by its colour coding [8], but it is an off shelf board kit. Apart from industries, food bakeries and pizza shops also frequently uses a slow-moving chain based conveyor belt to transport food products through an oven within limited defined distance. At the airports the conveyor belts are main tool to transfer passenger's luggage. Traditionally, Bar Code readers are used to classify and sort the objects, according to size, shape, weight, and so on [9]. The data acquisition through

sensors involves measurement, adjustment and control of different parameters such as direction, speed, angle etc.

1.2 Robotic Arm Definition

A Robotic arm is basically a machine which is very similar to a human hand, it consists of a combination of links attached in series or parallel. It can be controlled by programming it to perform a specific task.Joints of the manipulator connect the links that leads to the displacement which is dither translational or rotational. A kinematic chain is formed by the links of the arm. End Effector is the terminating part of this kinematic chain and it can be considered as the hand of a human.

1.3 Objective

The objectives of this project are:

- To study about **Barcode Scanner System Robotic Arm Product Pick and Place.**
- To implement detect an object in perfectly and take exact box.
- To test the performance of **Barcode Scanner System Robotic Arm Product Pick and Place**

1.4 Methodology

Our used methodology for the project:

- Design and construction of an **Barcode Scanner System Robotic Arm Product Pick and Place** and designing a block diagram & circuit diagram to know which components need to construct it.
- Collecting the all components and programming for the micro-controller to controlled the system.
- Setting all components in a PCB board & soldering. Then assembling the whole block in a board and finally run the system & checking.

1.5 Structure of the Project

This project book consists of five chapter. The first chapter contains the statement of the introduction, our background study for the project, Literature Review, objectives of the study in the project, Methodology and the project organization. Chapter two contains System design and simulation. Here we describe the design of this project, block diagram and circuit diagram, working principle, background and real project, details of component and instrument details of the whole project. Chapter three contains the Hardware implementation. Chapter four deals with the experiment result, calculation, advantages and application of this project. In the final chapter, we discuss about future scope and conclusion of our project.

Chapter 2

Literature Review

2.1 Literature Review:

S. V. Rautu, A. P. Shinde, N. R. Darda,A. V.Vaghule, C. B.Meshram, S.S.Sarawade proposed a Low Cost Automation System for sorting colored objects on the basis of their color, weight variation and type. The project mainly focuses on sorting 2different weighing non-metallic objects which are available in 3 different color sensing load cell, TCS230 Color

Sensor, inductive sensor and DC Geared Motors. The system consists of Conveyor Belt which takes the objects[11]. In Aji Joy reviews there are many color sensing ICs available today. In different ICs the properties vary such as color differentiating ability, output format, price, speed, resolution etc. In this project TCS3200 is selected. The TCS3200 is a programmable light-to-frequency converter that combines configurable silicon photodiodes and acurrent to frequency converter on a single monolithic CMOS integrated circuit.

The output is a square wave (50% dutycycle) with frequency directly proportional to light intensity[12]. The Aung Thike, Zin Zin Moe San, Dr. Zaw Min Oo have to consider separating and placing the blocks according to their colors. The TCS230 color detector can measure three primary colors Red, Green and Blue and it also has a separate white light detector. Since any color can be created from different levels of these primary colors, the unit can tell you the color composition of a light source. Color blocks are used to test the project. The authors have to consider separating and placing the blocks according to their colors[13]. Vishnu R. Kale, V. A. Kulkarni proposed the servo turn rate, or transit time, is used for determining servo rotational velocity. This is the amount of time it takes for the servo to move a set amount, usually 60 degrees. For example, suppose you have a servo with a transit time of 0.17sec/60 degrees at no load. This means it would take nearly half a second to rotate an entire 180 degrees. More if the servo were under a load. This information is very important if high servo response speed is a requirement of your robot application[14].

Survey on Design and Development of competitive low-cost Robot Arm with Four Degrees of Freedom by Ashraf Elfasahany. In this paper the representation of the design, development and implementation of robot arm is done, which has the ability to perform simple tasks, such as lightmaterial handling. The robotic arm is created and made from high quality acrylic[15]. Sharath Surati, Shaunak Hedaoo, Tushar Rotti, Vaibhav Ahuja, Nishigandha Patel have used 4 servo motors to make joints of the robotic arm and the movement will be controlled with the help of potentiometer. The arm has been built by the Cardboard and individual parts are attached to the respective servo motors. The arm is specifically created to pick and place light weight objects. So low torque servos, with a rotation of 0 to 180 degrees have been used. Thus the paper basically focuses on creating a robotic arm with non useful materials and its application on small purposes[16]. In Muhammad Naufal Bin A Rahman, Zol Bahri Razali
reviews the structure of the Conveyor and Robotic Arm was formed by three sub-assemblies for Robotic Arm namely gripper assembly, elbow assembly and arm assembly. This type of modular design is efficient in the sense that it enables easy replacement of parts and components during maintenance of device. Also, it helps in providing comprehensible design concept and thus, machines and devices with modular design are easier to be assembled and
managed because of this advantage[17]. Dr. Reg Pecen proposed Conveyor Belt prototype is heavily constrained to maintain the cost within the limited budget provided to students. For an industrial equivalent of a real system, an ideal motor for a similar project would be a 3- phase induction motor to provide a high starting torque, good speed regulation, and reasonable overload capacity.

The proposed modular Conveyor Belt system starts up with both Robotic Arms actively looking for a specific universal product code (UPC) of the functional block diagram. Once a barcode has been found, the reader will look within the ARDUINO microcontroller’s code to find the predetermined location in which the box will be placed[18]. Utkarsh Kamble, Shubham Paturkar, Pooja Swami, Shivam Malwade, M. S.Dholkawala, Anand Bhise provides to design the conveyor system used which includes belt speed, belt width, motor selection, belt specification, shaft diameter, pulley, gear box selection, with the help of standard model calculation. During the

project design stage for the transportation of raw materials or finished products, the choice of the method must favour the most cost-effective
solution for the volume of material moved; the plant and its maintenance; its flexibility for adaptation and its ability to carry a variety of loads and even be overloaded at times[19]. Jaroslav Homišin, present effective value of vibration (RMS) and CREST factor for analysis of vibrations in conveyors. Because of the continuous use of conveyors, it is important to consider vibrations while designing the conveyors to avoid failure due to vibrations[20]. ER. Rajput, in this book the operation and control of robots is discussed. ARDUINO cookbook, in this book details and methods of interfacing hardware components such as DC Motor, Servo Motor and RF Transmitter and Receiver is been discussed[21].

In Amir Deshmukh, Mahesh Nagane, Vaibhav Awatade reviews as a color is an interaction between a very small range of electromagnetic waves and the eyes and brain of a person. What people call red, green, or blue are just ways of categorizing what their brain experiences. Light is a type of energy, which makes up a small portion of the electromagnetic spectrum. The region of visible light consists of light with a wavelength between approximately 380 nm to 780 nm[22].

Chapter 3

Design & Construction

3.1 Introduction

Automatic product sorting system using bar-code scanner machines are made to separate many types objects. Here use various kind of equipment's and setup its well and finally made it to prorate.

3.2 Block Diagram

In our project we have set up a Bar-code **Scanner System Robotic Arm Product Pick and Place**. Here, in this system, we are use AC Source for operate. Here we also use a SMPS, Bar-code Scanner, Servo Motor, Gear motor etc.

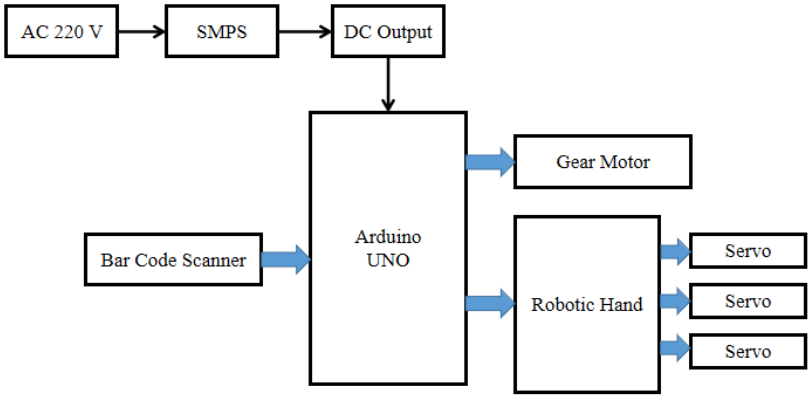

Figure 3.1: Block Diagram of our project

3.3 Circuit Diagram

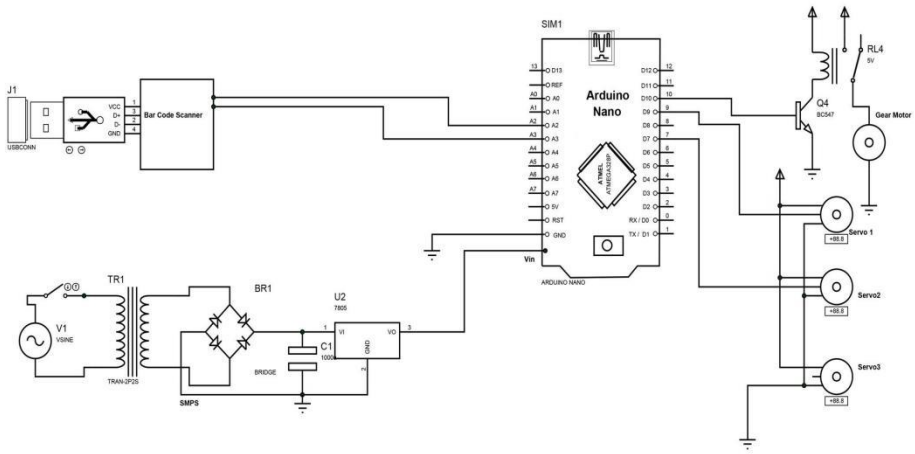

Figure 3.2: Circuit Diagram of Our Project

3.4 Working Principle

Our project is designed to separate objects using bar-code scanner. We are using Arduino Nano for controlling our project, which is acting as the main controller here. Also used here sensors, relays, servo motor, dc gear motors. Everything comes connected with Arduino. Here the current from AC is entering the circuit at 5v volts via SMPS. The bar-code scanner detects whether it is to put right box. When the conveyor belt rotates, the belt will stop when the object comes in front of the scanner. Then bar-code check the object and initiative then a robotic arm handle will put it in a specific box .

3.5 Complete Project Prototype Image :

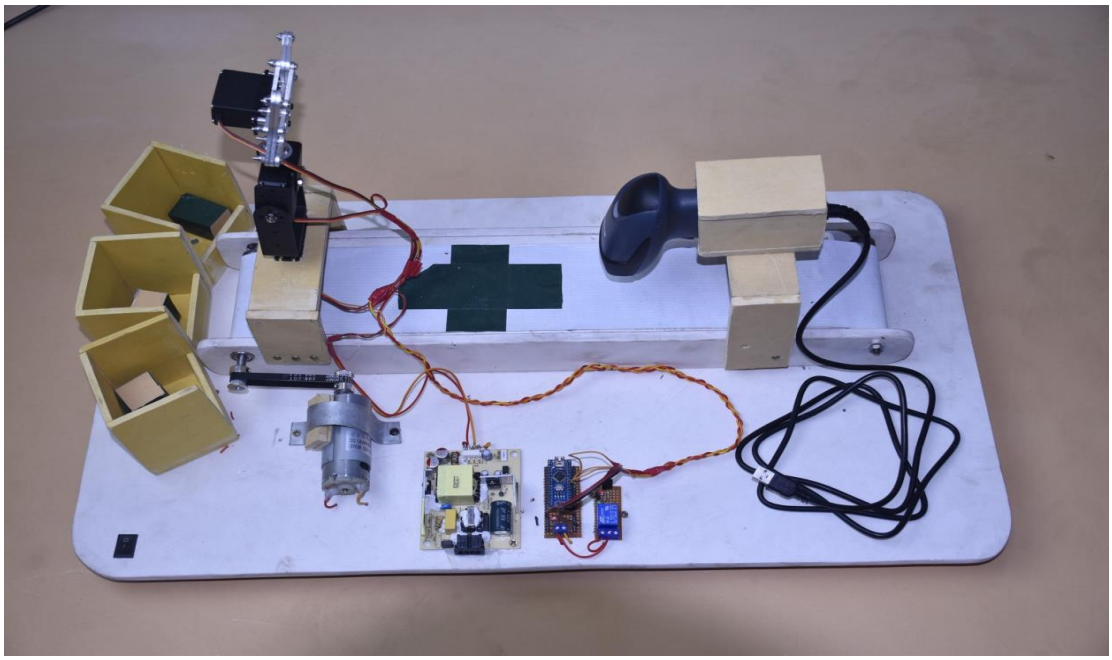

Figure 3.3: Complete Project

3.6 Required Components:

In this system we want to need to make some instruments, These are given below -

1. Arduino Nano.
2. Servo Motor
3. Gear Motor.
4. SMPS
5. Relay.
6. Transistor

7. LED.
8. Diode
9. Resistor

Software Required

1. Arduino IDE Software
2. Proteus Software

3.7 Arduino Nano

Arduino is open-source electronics prototyping platform based on flexible, easy-to-use hardware and software. It's intended for artists, designers, hobbyists, and anyone interested in creating interactive objects or environments. Arduino can sense the environment by receiving input from a variety of sensors and can affect its surroundings by controlling Lights, motors, and other actuators.

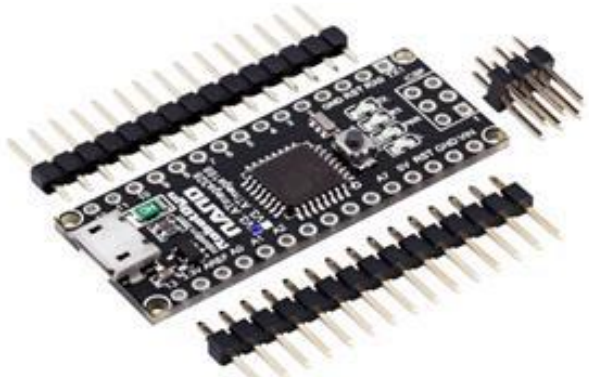

Figure 3.3: Arduino Nano

The microcontroller on the board is programmed using the Arduino programming language (based on Wiring) and the Arduino development environment (based on Processing). Arduino projects can be stand-alone or they can communicate with software on running on a computer (e.g. Flash, Processing, Maxims'). Arduino Nano is a surface mount breadboard embedded version with integrated USB. It is a small, complete, and breadboard friendly component. It has everything that Decimal/ Duemilanove has (electrically) with more analog input pins and onboard +5V AREF jumper. Physically, it is missing power jack. The Nano can automatically sense and switch to the higher potential source of power.

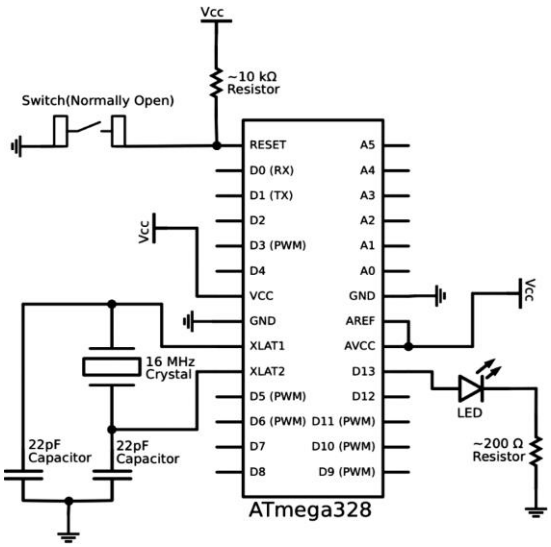

Figure 3.4: Arduino Nano Schematic Diagram

Nano's got the breadboard-ability of the Boarding and the Minibus with smaller footprint than either, so users have more breadboard space. It's got a pin layout that works well with the Mini or the Basic Stamp (TX, RX, ATN, and GND on one top, power and ground on the other). This new version 3.0 comes with ATMEGA328 which offer more programming and data memory space. It has two layers. That make it easier to hack and more affordable. One of the best features of Arduino Nano is, it's easy to use, compact and also small.

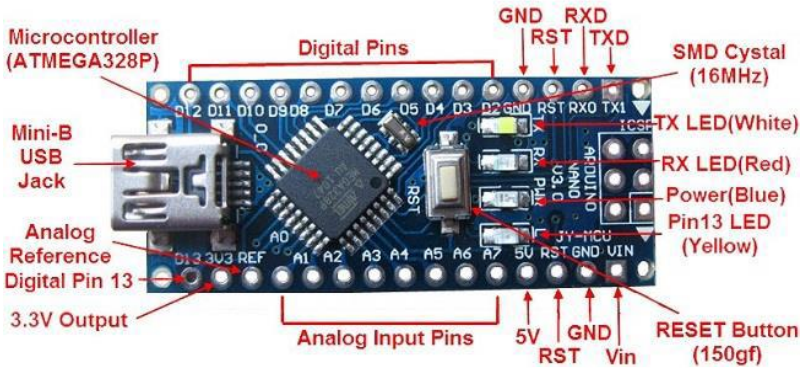

Figure 3.5: How Arduino Nano looks like

Specifications:

- Microcontroller: Atmel ATmega328
- Operating Voltage (logic level): 5 V
- Input Voltage (recommended): 7-12 V

- Input Voltage (limits): 6-20 V
- Digital I/O Pins: 14 (of which 6 provide PWM output)
- Analog Input Pins: 8
- DC Current per I/O Pin: 40 mA
- Flash Memory: 32 KB (of which 2KB used by boot loader)
- SRAM : 2 KB
- EEPROM: 1 KB
- Clock Speed: 16 MHz
- Dimensions: 0.70” x 1.70”

Features:

- Automatic reset during program download
- Power OK blue LED
- Green (TX), red (RX) and orange (L) LED
- Auto sensing/switching power input
- Small mini-B USB for programming and serial monitor
- ICSP header for direct program download
- Manual reset switch

Micro-controller IC ATmega328p

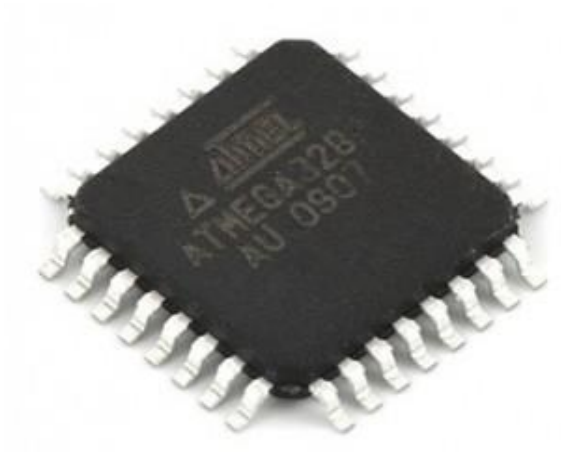

Figure 3.6: Micro-controller IC AT mega 328p

The high-performance Microchip Pico Power 8-bit AVR RISC-based micro-controller combines 32KB ISP flash memory with read-while-write capabilities, 1024B EEPROM, 2KB SRAM, 23 general purpose I/O lines, 32 general purpose working registers, three flexible timer/counters with compare modes, internal and external interrupts, serial programmable USART, a byte-oriented 2-wire serial interface, SPI serial port, a 6-channel 10-bit A/D converter (8-channels in TQFP and QFN/MLF packages), programmable watchdog timer with internal oscillator, and five software selectable power saving modes. The device operates between 1.8-5.5 volts. By executing powerful instructions in a single clock cycle, the device achieves throughputs approaching 1 MIPS per MHz, balancing power consumption and processing speed.

3.8 SMPS (Switch Mode Power Supply)

A switched-mode power supply (SMPS) is an electronic circuit that converts power using switching devices that are turned on and off at high frequencies, and storage components such as inductors or capacitors to supply power when the switching device is in its non-conduction state. Switching power supplies have high efficiency and are widely used in a variety of electronic equipment, including computers and other sensitive equipment requiring stable and efficient power supply. A switched-mode power supply is also known as a switch-mode power supply or switching-mode power supply.

Figure 3.7: SMPS (Switch Mode Power Supply)

A switched-mode power supply (switching-mode power supply, switch-mode power supply, switched power supply, SMPS, or switcher) is an electronic power supply that incorporates a switching regulator to convert electrical power efficiently. Like other power supplies, an SMPS transfers power from a DC or AC source (often mains power) to DC loads, such as a personal computer, while converting voltage and current characteristics. Unlike a linear power supply, the pass transistor of a switching-mode supply continually switches between low-dissipation, full-on and full-off states, and spends very little time in the high dissipation transitions, which minimizes wasted energy. A hypothetical ideal switched-mode power supply dissipates no power. Voltage regulation is achieved by varying the ratio of on-to-off time (also known as duty cycles). In contrast, a linear power supply regulates the output voltage by continually dissipating power in the pass transistor. This higher power conversion efficiency is an important advantage of a switched-mode power supply. Switched-mode power supplies may also be substantially smaller and lighter than a linear supply due to the smaller transformer size and weight.

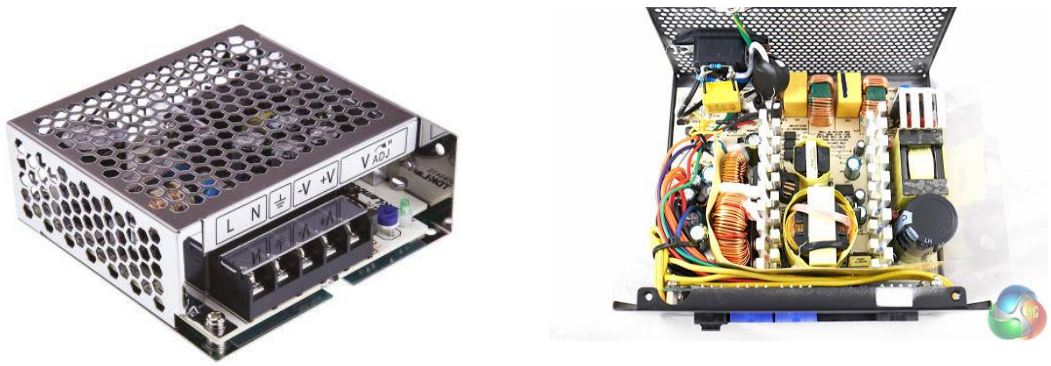

Figure 3.8 SMPS (Switch Mode Power Supply)

Switching regulators are used as replacements for linear regulators when higher efficiency, smaller size or lighter weight are required. They are, however, more complicated; their switching currents can cause electrical noise problems if not carefully suppressed, and simple designs may have a poor power factor.

- Input Voltage: AC 100 – 264V 50 / 60Hz
- Output Voltage: 5V DC, 0-20A

- Output voltage: Adjustment Range: ±20%
- Protections: Overload / Over Voltage / Short Circuit
- Auto-Recovery After Protection
- Universal AC input / Full range
- 100% Full Load Burn-in Test
- Cooling by Free Air Convection
- High Quality and High Performance
- LED power supply with a metal body for hidden installation for LED lighting
- Design with Built-in EMI Filter, improve signal precision.
- Certifications: CE & RoHs
- No Minimum Load.
- Compact Size Light Weight.
- High Efficiency, Reliability & low energy consumption
- Category – Switch Mode Power Adaptor (SMPS)

3.9 DC Gear Motor

Description:

A DC motor is any motor within a class of electrical machines whereby direct current electrical power is converted into mechanical power. A 12v DC motor is small and inexpensive, yet powerful enough to be used for many applications.

Specification:

- Voltage: 12V DC
- Gear ratio: 1/31
- No-load speed: 200RPM
- Rated Speed: 140RPM
- Rated torque: 10kg.cm
- Rated current: 2.5Amp
- Length of Motor(including spindle): 106mm/4.17″

Figure 3.9: DC Gear Motor

3.10 Motor Pulley

A **pulley** is a wheel on an axle or shaft that is designed to support movement and change of direction of a taut cable or belt, or transfer of power between the shaft and cable or belt. In the case of a pulley supported by a frame or shell that does not transfer power to a shaft, but is used to guide the cable or exert a force, the supporting shell is called a block, and the pulley may be called a sheave.

A pulley may have a groove or grooves between flanges around its circumference to locate the cable or belt. The drive element of a pulley system can be a rope, cable, belt, or chain. The earliest evidence of pulleys dates back to Ancient Egypt in the Twelfth Dynasty (1991-1802 BCE) and Mesopotamia in the early 2nd millennium BCE. In Roman Egypt, Hero of Alexandria (c. 10-70 CE) identified the pulley as one of six simple machines used to lift weights. Pulleys are assembled to form a block and tackle in order to provide mechanical advantage to apply large forces. Pulleys are also assembled as part of belt and chain drives in order to transmit power from one rotating shaft to another. Plutarch's *Parallel Lives* recounts a scene where Archimedes proved the effectiveness of compound pulleys and the block-and-tackle system by using one to pull a fully laden ship towards him as if it was gliding through water.

Figure 3.10: Pulley

3.11 Resistor

A **resistor** is a passive two-terminal electrical component that implements electrical resistance as a circuit element. In electronic circuits, resistors are used to reduce current flow, adjust signal levels, to divide voltages, bias active elements, and terminate transmission lines, among other uses. High-power resistors that can dissipate many watts of electrical power as heat, may be used as part of motor controls, in power distribution systems, or as test loads for generators. Fixed resistors have resistances that only change slightly with temperature, time or operating voltage. Variable resistors can be used to adjust circuit elements (such as a volume control or a lamp dimmer), or as sensing devices for heat, light, humidity, force, or chemical activity.

Two typical schematic diagram symbols are as follows:

(a) resistor, (b) rheostat (variable resistor), and (c) potentiometer

Figure 3.11: IEC resistor symbol

The notation to state a resistor's value in a circuit diagram varies.

One common scheme is the RKM code following IEC 60062. It avoids using a decimal separator and replaces the decimal separator with a letter loosely associated with SI prefixes corresponding with the part's resistance. For example, *8K2* as part marking code, in a circuit diagram or in a bill of materials (BOM) indicates a resistor value of 8.2 kΩ. Additional zeros imply a tighter tolerance, for example *15M0* for three significant digits. When the value can be expressed without the need for a prefix (that is, multiplicator 1), an "R" is used instead of the decimal separator. For example, *1R2* indicates 1.2 Ω, and *18R* indicates 18 Ω.

Specifications:

- Resistance: 220 Ohms
- Power (Watts): 0.25W, 1/4W
- Temperature Coefficient: 350ppm/Celcius
- Tolerance: +/- 5%
- Case: Axial

A resistor is a passive two-terminal electrical component that implements electrical resistance as a circuit element. Resistors act to reduce current flow, and, at the sometime, act to lower voltage levels within circuits. Resistors may have fixed resistances or variable resistances, such as those founding thermostats, visitors, trimmers, photo resistors, hamsters and potentiometer. The current through a resistor is in direct proportion to the voltage across the resistor's terminals. This relationship is represented by Ohm's law

Figure 3.12: Resistor

Theory of operation

The behavior of an ideal resistor is dictated by the relationship specified by Ohm ‘slaw:

$$V = I.R$$

Ohm's law states that the voltage (V) across a resistor is proportional to the current (I), where the constant of proportionality is the resistance (R).

Equivalently, Ohm's law can be stated:

$$I = V/R$$

This formulation states that the current (I) is proportional to the voltage (V) and inversely proportional to the resistance (R). This is directly used in practical computations. For example, if a 300-ohm resistor is attached across the terminals of a12 volt battery, then a current of12 / 300 = 0.04 amperes flows through that resistor.

3.12 Servo Motor

A servomotor is a closed-loop servomechanism that uses position feedback to control its motion and final position. The input to its control is a signal (either analogue or digital) representing the position commanded for the output shaft. The motor is paired with some type of position encoder to provide position and speed feedback. In the simplest case, only the position is measured.

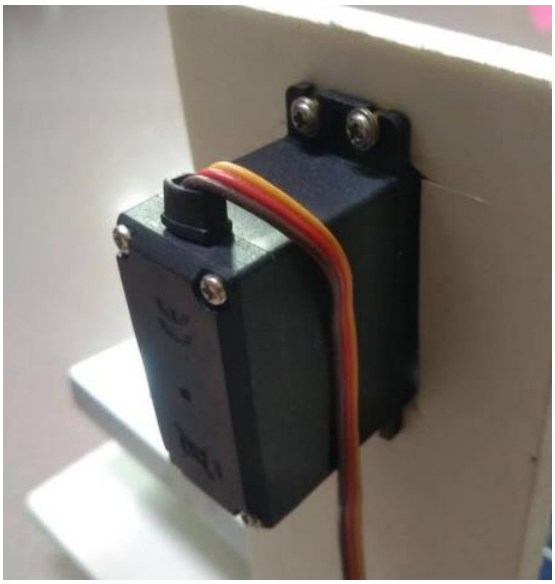

Figure 3.13: Servo Motor

The measured position of the output is compared to the command position, the external input to the controller. If the output position differs from that required, an error signal is generated which then causes the motor to rotate in either direction, as needed to bring the output shaft to the appropriate position. As the positions approach, the error signal reduces to zero and the motor stops.

The very simplest servomotors use position-only sensing via a potentiometer and bang-bang control of their motor; the motor always rotates at full speed (or is stopped). This type of servomotor is not widely used in industrial motion control, but it forms the basis of the simple and cheap servos used for radio-controlled models. More sophisticated servomotors use optical rotary encoders to measure the speed of the output shaft and a variable-speed drive to control the motor speed. Both of these enhancements, usually in combination with a PID control algorithm, allow the servomotor to be brought to its commanded position more quickly and more precisely, with less overshooting.

3.13 Relay

A relay is an electrically operated switch. Many relays use an electromagnet to mechanically operate a switch, but other operating principles are also used, such as solid-state relays. Relays are used where it is necessary to control a circuit by a separate low-power signal, or where several circuits must be controlled by one signal. The first relays were used in long distance telegraph circuits as amplifiers: they repeated the signal coming in from one circuit and re-transmitted it on another circuit. Relays were used extensively in telephone exchanges and early computers to perform logical operations.

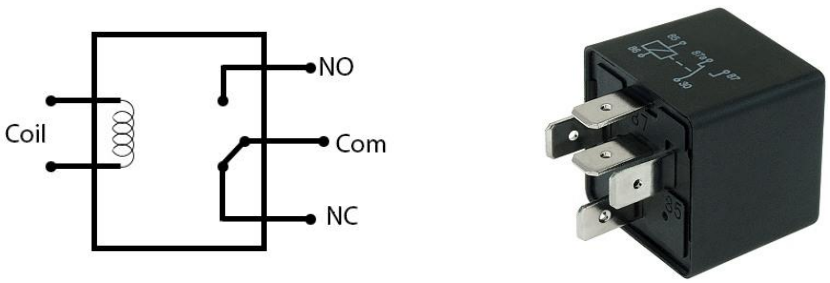

Figure 3.14: Relay

A type of relay that can handle the high power required to directly control an electric motor or other loads is called a contactor. Solid-state relays control power circuits with no moving parts, instead using a semiconductor device to perform switching. Relays with calibrated operating characteristics and sometimes multiple operating coils are used to protect electrical circuits from overload or faults; in modern electric power systems these functions are performed by digital instruments still called "protective relays".

Magnetic latching relays require one pulse of coil power to move their contacts in one direction, and another, redirected pulse to move them back. Repeated pulses from the same input have no effect. Magnetic latching relays are useful in applications where interrupted power should not be able to transition the contacts. Magnetic latching relays can have either single or dual coils. On a single coil device, the relay will operate in one direction when power is applied with one polarity, and will reset when the polarity is reversed. On a dual coil device, when polarized voltage is applied to the reset coil the contacts will transition. AC controlled magnetic latch relays have single coils that employ steering diodes to differentiate between operate and reset commands.

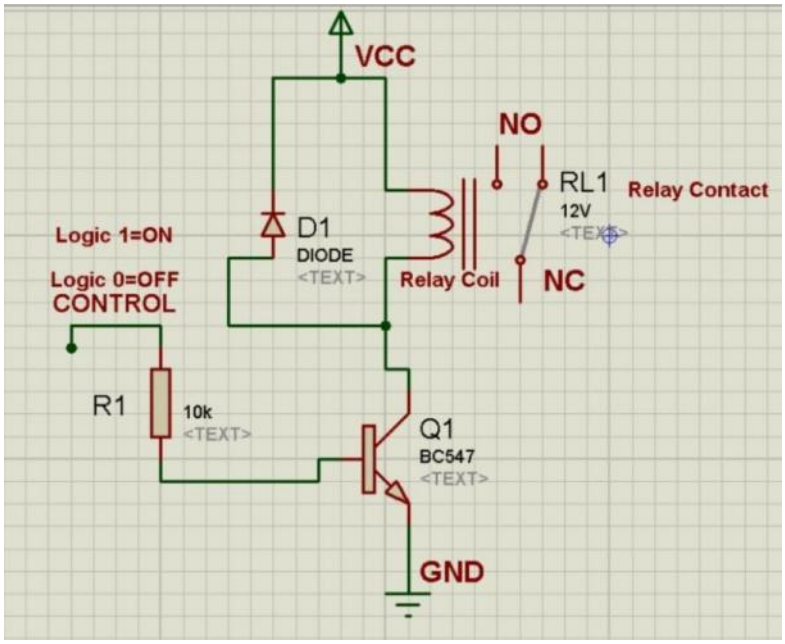

Figure 3.15: Transistor Switching Circuit.

The circuit above is called a low-side switch, because the switch – our transistor – is on the low (ground) side of the circuit. Alternatively, we can use a PNP transistor to create a high-side switch: Similar to the NPN circuit, the base is our input, and the

emitter is tied to a constant voltage. A relay is an electrically operated switch of mains voltage. It means that it can be turned on or off, letting the current go through or not. Controlling a relay with the Arduino is as simple as controlling an output such as an LED. The relay module is the one in the figure below.

This module has two channels (those blue cubes). There are other varieties with one, four and eight channels.

Mains voltage connections:

In relation to mains voltage, relays have 3 possible connections:

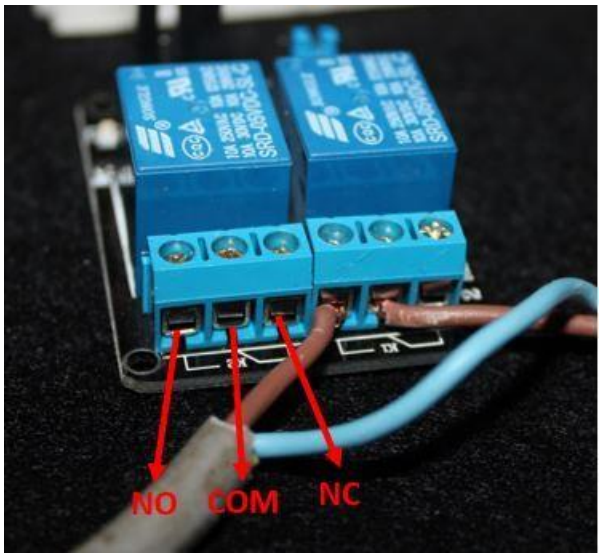

Figure 3.16: Pin diagram of Relay Module

- **COM**: common pin

- **NO (Normally Open):** there is no contact between the common pin and the normally open pin. So, when you trigger the relay, it connects to the COM pin and supply is provided to a load

- **NC (Normally Closed):** there is contact between the common pin and the normally closed pin. There is always connection between the COM and NC pins, even when the relay is turned off. When you trigger the relay, the circuit is opened and there is no supply provided to a load.

Pin wiring:

The connections between the relay module and the Arduino are really simple:

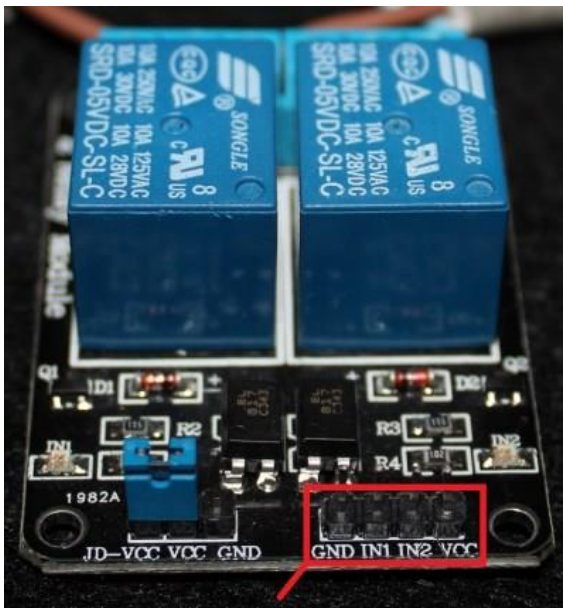

Figure 3.17: Main Voltage Connection

- **GND**: goes to ground
- **IN1**: controls the first relay (it will be connected to an Arduino digital pin)
- **IN2**: controls the second relay (it should be connected to an Arduino digital pin if you are using this second relay. Otherwise, you don't need to connect it)
- **VCC**: goes to 5V

3.14 Arduino Software

The digital micro-controller unit named as Arduino Nano can be programmed with the Arduino software IDE. There is no any requirement for installing other software rather than Arduino. Firstly, Select "Arduino Nano from the Tools, Board menu (according to the micro-controller on our board). The IC used named as ATmega328 on the Arduino Nano comes pre burned with a boot loader that allows us to upload new code to it without the use of an external hardware programmer.

Communication is using the original STK500 protocol (reference, C header files). We can also bypass the boot loader and programs the microcontroller through the ICSP (In Circuit Serial Programming) header. The ATmega16U2 (or 8U2 in the rev1 and rev2 boards) firmware source code is available. The ATmega16U2/8U2 is loaded with a DFU boot loader, which can be activated by: On Rev1 boards: connecting the solder jumper on the back of the board (near the map of Italy) and then resetting the 8U2. On Rev2 or later boards: there is a resistor that pulling the 8U2/16U2 HWB line to ground, making it easier to put into DFU mode.

The Arduino Nano is one of the latest digital microcontroller units and has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega328 provides UART TTL at (5V) with serial communication, which is available on digital pins 0 -(RX) for receive the data and pin no.1 (TX) for transmit the data. An ATmega16U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The '16U2 firmware uses the standard USB COM drivers, and no external driver is needed. However, on Windows, an .in file is required. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the Arduino board.

The RX and TX LEDs on the board will flash when data is being transmitted via the USB-to-serial chip and USB connection to the computer (but not for serial Communication on pins 0 and 1). A Software Serial library allows for serial communication on any of the Nano's digital pins. The ATmega328 also supports I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus. Arduino programs are written in C or C++ and the program code written for Arduino is called sketch. The Arduino IDE uses the GNU tool chain and AVR Lab to compile programs, and for uploading the programs it uses argued. As the Arduino platform uses Atmel microcontrollers, Atmel's development environment, AVR Studio or the newer Atmel Studio, may also be used to develop software for the Arduino.

The Arduino Integrated Development Environment - or Arduino Software (IDE) - contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions and a series of menus. It connects to the Arduino and Genuino hardware to upload programs and communicate with them.

Writing Sketches

Programs written using Arduino Software (IDE) are called sketches. These sketches are written in the text editor and are saved with the file extension .ino. The editor has features for cutting/pasting and for searching/replacing text. The message area gives feedback while saving and exporting and also displays errors. The console displays text output by the Arduino Software (IDE), including complete error messages and other information. The bottom right hand corner of the window displays the configured board and serial port. The toolbar buttons allow you to verify and upload programs, create, open, and save sketches, and open the serial monitor.

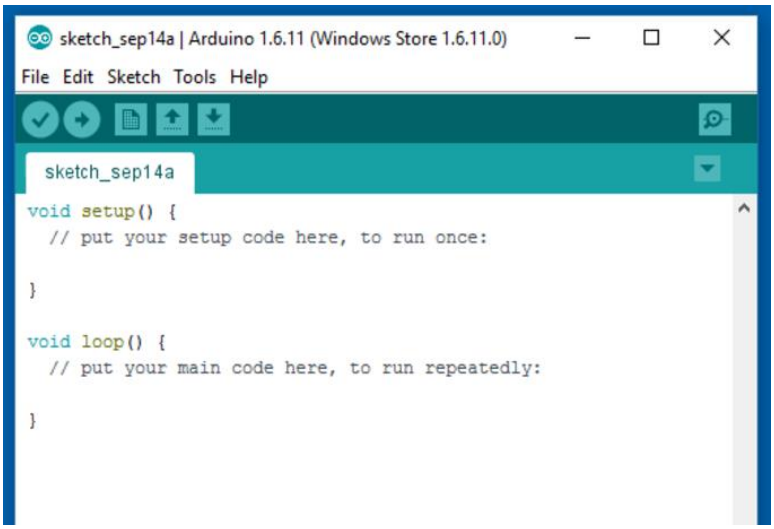

Figure 3.18: Arduino Software Interface IDE

Upload Using Programmer

This will overwrite the bootloader on the board; you will need to use Tools > Burn Bootloader to restore it and be able to Upload to USB serial port again. However, it allows you to use the full capacity of the Flash memory for your sketch. Please note that this command will NOT burn the fuses. To do so a Tools -> Burn Bootloader command must be executed.

Uploading

Before uploading your sketch, you need to select the correct items from the Tools > Board and Tools > Port menus. The boards are described below. On the Mac, the serial port is probably something like /dev/tty.usbmodem241 (for an Uno or Mega2560 or Leonardo) or /dev/tty.usbserial-1B1 (for a Duemilanove or earlier USB board), or /dev/tty.USA19QW1b1P1.1 (for a serial board connected with a Keyspan USB-to-Serial adapter). On Windows, it's probably COM1 or COM2 (for a serial board) or COM4, COM5, COM7, or higher (for a USB board) - to find out, you look for USB serial device in the ports section of the Windows Device Manager. On Linux, it should be /dev/ttyACMx , /dev/ttyUSBx or similar. Once you've selected the correct serial port and board, press the upload button in the toolbar or select the Upload item from the Sketch menu. Current Arduino boards will reset automatically and begin the upload. With older boards (pre-Diecimila) that lack auto-reset, you'll need to press the reset button on the board just before starting the upload. On most boards, you'll see the RX and TX LEDs blink as the sketch is uploaded. The Arduino Software (IDE) will display a message when the upload is complete, or show an error.

Libraries

Libraries provide extra functionality for use in sketches, e.g. working with hardware or manipulating data. To use a library in a sketch, select it from the Sketch > Import Library menu. This will insert one or more #include statements at the top of the sketch and compile the library with your sketch. Because libraries are uploaded to the board with your sketch, they increase the amount of space it takes up. If a sketch no longer needs a library, simply delete its #include statements from the top of your code.

Third-Party Hardware

Support for third-party hardware can be added to the hardware directory of your sketchbook directory. Platforms installed there may include board definitions (which appear in the board menu), core libraries, bootloaders, and programmer definitions. To install, create the hardware directory, then unzip the third-party platform into its own sub-directory. (Don't use "arduino" as the sub-directory name or you'll override the built-in Arduino platform.) To uninstall, simply delete its directory.

Serial Monitor

This displays serial sent from the Arduino or Genuino board over USB or serial connector. To send data to the board, enter text and click on the "send" button or press enter. Choose the baud rate from the drop-down menu that matches the rate passed to Serial.begin in your sketch.

3.15 Proteus Software

The Proteus Design Suite is a proprietary software tool suite used primarily for electronic design automation. The software is used mainly by electronics design engineers and technicians to create schematics and electronics prints for manufacturing printed circuit boards. .The first version of what is now the Proteus Design Suite was called PC-B and was written by the company chairman, John Jameson, for DOS in 1988. Schematic Capture support followed in 1990 with a port to the Windows environment shortly thereafter. Mixed mode SPICE Simulation was first integrated into Proteus in 1996 and microcontroller simulation then arrived in Proteus in 1998.

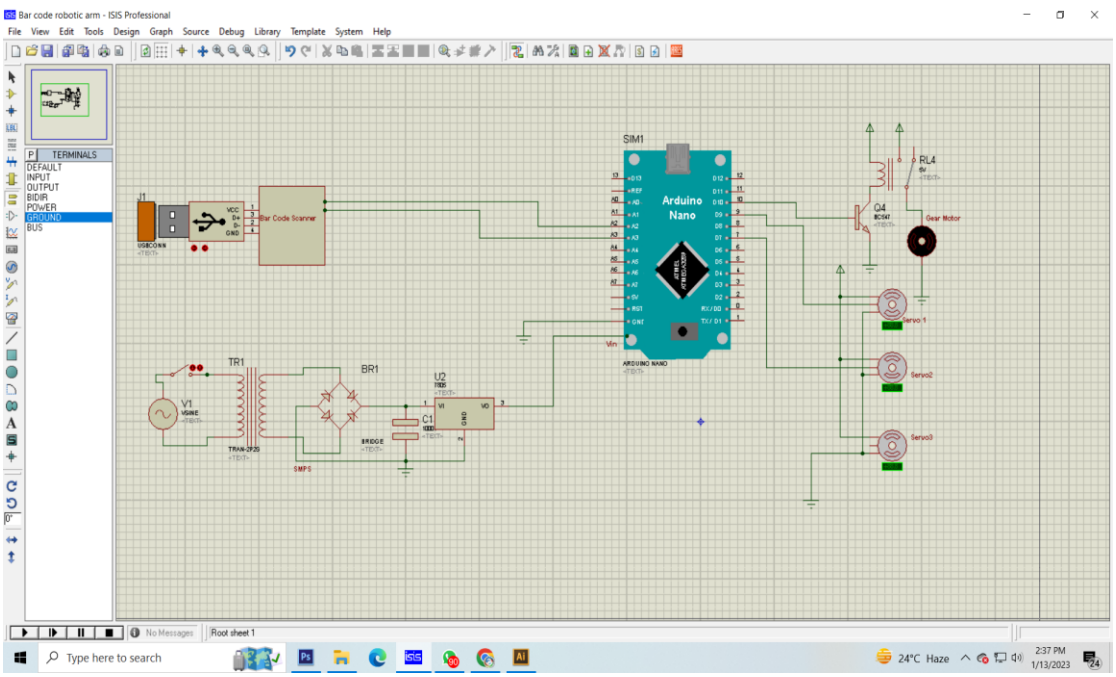

Figure 3.19: Our Project Design in Proteus Software

Chapter 4

Experimental Result

4.1 Results

After making our project we observe it very careful. It works as we desire. Our project give output perfectly and all equipment are work perfectly. We check how much it works and we get perfect output from this project.

- Firstly When the system is on then conveyor belt will be start.
- Then Bar-code scanner scan its bar-code then separate the objects in 3 different box.
- Then micro-controller send a signal to the robotic arm to separate the product as our desire.

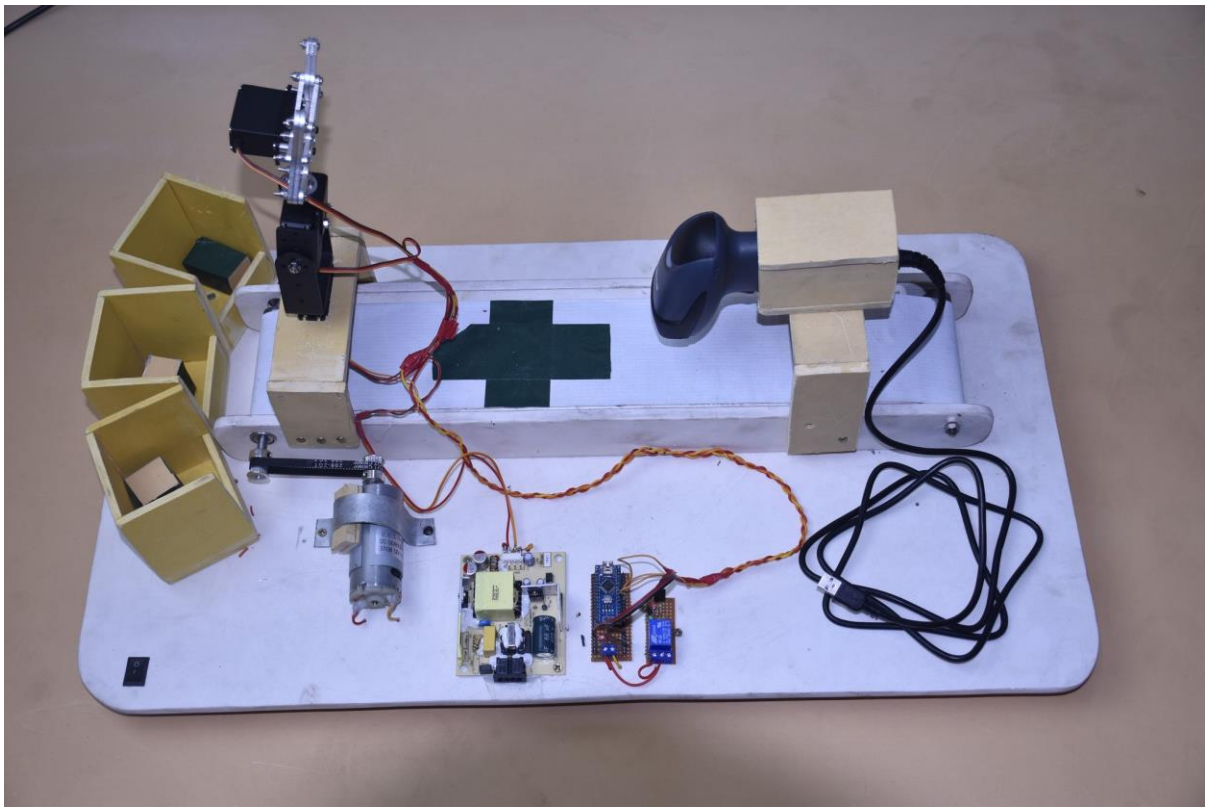

Figure 4.1: Working Project Image

4.2 Discussion

This kind of project already many of student study before, but they control it PLC system. We control conveyor belt through a mobile phone and conveyor belt roll with the help of a gear motor. When we made this project then we face some of problem. We did not run it smoothly. After hard work and team work, we did it.

4.3 Application

The project has a major application in the

- It can be used for Industrial work.
- It can be used in pharmacy.
- It can be use for metal non metal product separating company.

4.4 Advantages

There are many advantages of our system because of its accuracy. Some of the advantages are pointed out below:

- Able to Bar-code Scanning Process
- Good accuracy to find object.
- Time saving machine for industrial work.
- Shorting product automatically.
- Very effectively work for product detecting.
- Reduce energy waste.
- No Oil consumption.
- Less skill technicians is sufficient to operate.
- Installation is simplified very much.
- Less time and more profit.
- Simple construction
- Ease of operation.

4.5 Limitations

It is a demo project so we found some limitation. In future we will work for reduce this kind of limitation. These limitations are –

- It is a demo project so its accuracy is 85%.

- Our project may delay in work for small range instruments.
- Product will be place in manually.

Chapter 5

Conclusion

5.1 Conclusion

The main objective of this project was to develop an object detecting system based on bar-code scanning.. This was successfully implemented. We consider this project as a journey where we acquired knowledge and also gained some insights into the subject which we have shared in this report. Micro controller was used to control the various operations. More features can be added to this system as follows: depending on the size, shape and weight of the objects, sorting operations can be implemented. Sorting operation can be improvised using a piston arrangement.

5.2 Future Scope of Work

The model can be improved by making some changes in the program and components. Some suggestions are given below-

- We can add a monitoring-based control to automate control.
- We will add product counting system.
- We will add IoT monitoring system.
- We will increase its working accuracy level.

REFERENCE

1. Calgar Elbuken, Mir Behrad Khamesee, and Mustafa Yavuz, „Design and Implementation of a micromanipulation System using a Magnetically Levitated MEMS Robot,“ IEEE/asme transactions on mechatronics, Vol. 14, pp. 4, August 2009

2. Ritsuya Oshima, Toshio Takayama, Toru Omata, Kazuyuki Kojima, Kozo Takase, and Naofumi Tanaka, „Assemblable Three-Fingered Nine-Degrees-of-Freedom Hand for Laparoscopic Surgery“, IEEE/asme transactions on mechatronics, vol. 15, no. 6, december 2010

3. Kurt E. Clothier and Ying Shang, „A Geometric Approach for Robotic arm kinematics with Hardware Design, Electrical Design, and Implementation“ Hindawi Publishing Corporation, Journal of Robotics, Volume 2010, Article ID 984823, 10 pages, doi:10.1155/2010/984823

4. Yong Zhang, Brandon K. Chen, Xinyu Liu, and Yu Sun, „Autonomous Robotic Pick- and-Place of Microobjects“, IEEE transactions on robotics, vol. 26, no. 1, february 2010.

5. Michael Dumiak,”Book Scanning Robots for Degitize Delicate Texts” IEEE Spectrum,vol 1,January 2008 in www.spectrum.ieee .org

6. N. Zemiti, G. Morel, T. Ortmaier, and N. Bonnet, “Mechatronic design of A new robot for force control in minimally invasive surgery,” IEEE/ASME Trans. Mechatronics, vol. 12, no.2, pp. 143–153, Apr. 2007.

7. S. Fatikow, T. Wich, H. Hulsen, T. Sievers, and M. Jahnisch, “Microrobot System for automatic nanohandling inside a scanning electron microscope,”IEEE/ASME Trans.Mechatronics, vol. 12, no. 3, pp. 244–252, Jun. 2007.

8. W. Ding, H. Zhang, and C. Cetinkaya, “Rolling resistance moment-based Adhesion characterization of microspheres,” *J*. Adhes., vol. 84, no. 12,Pp. 996–1006, 2008.

9 W. H. Wang, X. Y. Liu, and Y. Sun, “Contact detection in microrobotic Manipulation,” Int. *J. Robot. Res.*, vol. 26, pp. 821–828, 2007.

10 R. H. Taylor and D. Stoianovic, “Medical robotics in computer-integrated Surgery,” IEEE Trans. Robot. Autom., vol. 19, no. 5, pp. 765–781, Oct.2003, 2010.

11 I. W. Park, B. J. Lee, S. H. Cho, Y. D. Hong, and J. H. Kim,“Laser-based kinematic calibration of robot manipulator using differential kinematics,” IEEE/ASME Transactions on Mechatronics, pp. 1–9, 2011.

12 W. K. Veitschegger and C. H.Wu, “Robot calibration and compensation,”IEEE Journal of Robotics and Automation, vol. 4,No. 6, pp. 643–656, 1988**.**

13. Y. Sun and J. M. Hollerbach, “Active robot calibration algorithm,”In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA ‟08), pp. 1276–1281, May 2008.

APPDENDIX

```
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27 ,16,2);
#include <Servo.h>

Servo myservo1;
Servo myservo2;
Servo myservo3;

int ir1 = 2;
int ir11;


int motor = 4;

String incomingByte; // for incoming serial data

void setup() {
  Serial.begin(9600); // opens serial port, sets data rate to 9600 bps

pinMode(2,INPUT);
pinMode(3,INPUT);
pinMode(4,OUTPUT);
  myservo1.attach(5);  // attaches the servo on pin 9 to the servo object
     myservo2.attach(6);  // attaches the servo on pin 9 to the servo object
        myservo3.attach(7);  // attaches the servo on pin 9 to the servo object
 myservo1.write(20);              // tell servo to go to position in variable 'pos'
     myservo2.write(80);              // tell servo to go to position in variable 'pos'
        myservo3.write(60);              // tell servo to go to position in variable 'pos'
delay(1000);
}
void(* resetFunc) (void) = 0;  // declare reset fuction at address 0

void loop() {
 ir11 = digitalRead(ir1);
  digitalWrite(motor, HIGH);

if(ir11 == LOW){
digitalWrite(motor,LOW);
  delay(2000);
  digitalWrite(motor, HIGH);
  delay(500);
  digitalWrite(motor,LOW);
  delay(3000);
}
if(ir11 == HIGH){
```

```
}

  if (Serial.available() > 0) {

    incomingByte = Serial.readStringUntil('\n');
    Serial.println(incomingByte);

if(incomingByte =="60"){
digitalWrite(motor,LOW);
  delay(500);
 myservo1.write(10);              // tell servo to go to position in variable 'pos'
     myservo2.write(80);              // tell servo to go to position in variable 'pos'
        myservo3.write(60);              // tell servo to go to position in variable 'pos'
delay(1000);
    digitalWrite(motor,LOW);
  delay(500);
 myservo1.write(10);              // tell servo to go to position in variable 'pos'
     myservo2.write(80);              // tell servo to go to position in variable 'pos'
        myservo3.write(60);              // tell servo to go to position in variable 'pos'
delay(1000);
     myservo2.write(0);              // tell servo to go to position in variable 'pos'
delay(1000);
     myservo1.write(70);              // tell servo to go to position in variable 'pos'
delay(1000);
     myservo2.write(180);              // tell servo to go to position in variable 'pos'
delay(1500);
     myservo3.write(110);              // tell servo to go to position in variable 'pos'
delay(1000);
  myservo1.write(10);              // tell servo to go to position in variable 'pos'
delay(1000);
 myservo1.write(10);              // tell servo to go to position in variable 'pos'
     myservo2.write(80);              // tell servo to go to position in variable 'pos'
        myservo3.write(60);
delay(1000);
  digitalWrite(motor, HIGH);
delay(500);
resetFunc();

}
 if(incomingByte =="90"){

  digitalWrite(motor,LOW);
  delay(500);
 myservo1.write(10);              // tell servo to go to position in variable 'pos'
     myservo2.write(80);              // tell servo to go to position in variable 'pos'
        myservo3.write(60);              // tell servo to go to position in variable 'pos'
delay(1000);
     myservo2.write(0);              // tell servo to go to position in variable 'pos'
delay(1000);
```

```
    myservo1.write(70);              // tell servo to go to position in variable 'pos'
delay(1000);
    myservo2.write(180);              // tell servo to go to position in variable 'pos'
delay(1500);
    myservo1.write(10);              // tell servo to go to position in variable 'pos'
delay(1000);
 myservo1.write(10);             // tell servo to go to position in variable 'pos'
    myservo2.write(80);              // tell servo to go to position in variable 'pos'
       myservo3.write(60);
delay(1000);
  digitalWrite(motor, HIGH);
delay(500);
resetFunc();

 }
 if(incomingByte =="120"){
digitalWrite(motor,LOW);
  delay(500);
 myservo1.write(10);             // tell servo to go to position in variable 'pos'
    myservo2.write(80);              // tell servo to go to position in variable 'pos'
       myservo3.write(60);              // tell servo to go to position in variable 'pos'
delay(1000);
   digitalWrite(motor,LOW);
  delay(500);
 myservo1.write(10);             // tell servo to go to position in variable 'pos'
    myservo2.write(80);              // tell servo to go to position in variable 'pos'
       myservo3.write(60);              // tell servo to go to position in variable 'pos'
delay(1000);
    myservo2.write(0);              // tell servo to go to position in variable 'pos'
delay(1000);
    myservo1.write(70);              // tell servo to go to position in variable 'pos'
delay(1000);
    myservo2.write(180);              // tell servo to go to position in variable 'pos'
delay(1500);
    myservo3.write(0);              // tell servo to go to position in variable 'pos'
delay(1000);
  myservo1.write(10);              // tell servo to go to position in variable 'pos'
delay(1000);
 myservo1.write(10);             // tell servo to go to position in variable 'pos'
    myservo2.write(80);              // tell servo to go to position in variable 'pos'
       myservo3.write(60);
delay(1000);
  digitalWrite(motor, HIGH);
delay(500);
resetFunc();

 }

}
}
```