

VISVESVARAYA TECHNOLOGICAL UNIVERSITY
BELGAVI-590014



**A Project Phase - 1 Report
On**

“Colour Sorting Machine”

*Submitted in partial fulfillment of the requirements for the 7th semester of **Bachelor of Engineering in Electrical and Electronics Engineering** of Visvesvaraya Technological University, Belagavi*

Submitted by:

RAGHAV AMBASHTA	1RN18EE030
SHALU PRIYAMVADA	1RN18EE035
ANUBHAV AYUSHYAMAN	1RN18EE006
ATULYA MODI	1RN18EE008

Under the Guidance of:
Dr. Sumathi S
Professor & HOD
Dept. of EEE



Department of Electrical and Electronics Engineering
RNS Institute of Technology
Channasandra, Dr. Vishnuvardhan Road, RR Nagar Post, Bengaluru – 560 098
2021-2022

RNS Institute of Technology
Channasandra, Uttarahalli - Kengeri Main Road,
Bengaluru - 560 098

Department of Electrical and Electronics Engineering



CERTIFICATE

Certified that the mini project work entitled “**Colour Sorting Machine**” has been successfully carried out by **Raghav Ambashta (1RN18EE030)**, **Shalu Priyamvada (1RN18EE035)**, **Anubhav Ayushyaman (1RN18EE006)** and **Atulya Modi (1RN18EE008)** bonafide students of **RNS Institute of Technology** in partial fulfillment of the requirements for the **7th semester** of **Bachelor of Engineering in Electrical and Electronics Engineering** of **Visvesvaraya Technological University**, Belagavi, during the academic year 2020-2021. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the report deposited in the departmental library. The project report has been approved as it satisfies the requirements of **7th semester BE, EEE**.

Signature of Guide

Dr. Sumathi S

Signature of HoD
(Dept. of EEE)

Dr. Sumathi S

Signature of Principal

Dr. M K Venkatesha

RNS Institute of Technology
Channasandra, Uttarahalli - Kengeri Main Road,
Bengaluru - 560 098

Department of Electrical and Electronics Engineering



DECLARATION

We hereby declare that the entire work embodied in this project phase – 1 report titles “Colour Sorting Machine” submitted to Visvesvaraya Technological University, Belagavi, is carried out by us at the Department of Electrical and Electronics Engineering, RNS Institute of Technology, Bengaluru under the guidance of Dr. Sumathi S, Professor and HoD, Electrical and Electronics Engineering, RNS Institute OF Technology. This project has not been submitted in part or full for the award of any diploma or degree of this or any other University.

Name	USN	Signature
RAGHAV AMBASHTA	1RN18EE030	
SHALU PRIYAMVADA	1RN18EE035	
ANUBHAV AYUSHYAMAN	1RN18EE006	
ATULYA MODI	1RN18EE008	

ACKNOWLEDGEMENT

Any achievement, be it scholastic or otherwise does not depend solely on the individual efforts but on the guidance, encouragement and cooperation of intellectuals, elders and friends. A number of personalities, in their own capacities have helped us in carrying out this project work.

We would like to take this opportunity to thank them all. We would firstly like to thank **Visvesvaraya Technological University** for giving us the opportunity to work on a technical project.

We are grateful to **Dr. M K Venkatesha**, Principal, RNSIT, Bangalore, for his support towards completing our project.

We deeply express our sincere gratitude to our guides **Dr. Sumathi S, Professor & HOD, Department of EEE, RNSIT, Bangalore** for the able guidance, regular source of encouragement and assistance throughout this project.

We would like to thank all the teaching and non-teaching staff of Department of Electrical and Electronics Engineering RNSIT, Bangalore for their constant support and encouragement.

Date: 07/02/2022

Place: Bengaluru

Raghav Ambashta

1RN18EE030

Shalu Priyamvada

1RN18EE035

Anubhav Ayushyaman

1RN18EE006

Atulya Modi

1RN18EE008

ABSTRACT

In the modern scenario of manufacturing, quality holds an important component for achievement. It is essential to increase manufacturing pace, lower the labor charge and perform the task with utmost precision and efficiency. Sorting of object is an essential process in manufacturing. Merchandise should be taken care of in manufacturing and manual sorting is time consuming and labor extensive.

Colour sorting machines are most commonly used in the sorting of agricultural grain and rice, as well as in the processing of food products, such as coffee, nuts and oil crops. The optical sorter separates any stones, mouse droppings and discolored, toxic or otherwise unacceptable items. Manual work can create consistency problems. Machines can perform assignments superior to human beings.

This project aims at making an automatic sorting tool which helps the sorting mechanism based on color. For sensing TCS3200 coloration sensor has been used. For actuation, micro servo motors have been used. With the aid of reading the frequency of the output of the sensor, color primarily based absolutely sorting is completed.

TABLE OF CONTENTS

ACKNOWLEDGEMENT	i
ABSTRACT	ii
TABLE OF CONTENTS	iii
LIST OF FIGURES	iv
Chapter 1 INTRODUCTION	1
Chapter 2 WORKING PRINCIPLE	2
2.1 WORKING PRINCIPLE	2
2.2 BLOCK DIAGRAM	3
2.3 CIRCUIT DIAGRAM	4
Chapter 3 DESCRIPTION OF COMPONENTS	5
3.1 ARDUINO UNO	5
3.2 TCS3200 COLOUR SENSOR	11
3.3 SG90 MICRO SERVO	18
Chapter 4 PROGRAMMING LOGIC AND FLOWCHART	22
4.1 PSEUDOCODE	24
4.2 PROGRAMMING LOGIC	24
Chapter 5 APPLICATION	25
Chapter 6 CONCLUSION	26
APPENDIX	27
REFERENCES	31

LIST OF FIGURES

1.1	Block Diagram	3
1.2	Circuit Diagram	4
3.1	Arduino UNO Board	7
3.2	Arduino UNO Pin Diagram	8
3.3	TCS3200 Colour Sensor	11
3.4	Photodiode Array	12
3.5	Working of Photodiode Array	12
3.6	TCS3200 Design	13
3.7	TCS3200 Filter	14
3.8	TCS3200 Pin Diagram	16
3.9	Wiring of TCS3200 to Arduino UNO	17
3.10	SG90 Micro Servo	18
3.11	Controlling SG90	20
3.12	Wire Connections of SG90	21
4.1	Flowchart	22

CHAPTER 1

INTRODUCION

Problem Statement:

“Arduino Based Color Sorting Machine using TCS3200 colour sensor”

Sorting of objects based on color is an essential mechanical process in the manufacturing industry. Manually doing this task consumes a lot of time, money and energy. Product consistency and quality is also very poor. To obtain high quality product with low cost and time investment, a machine which can automate this task is the need of the day. This project aims at solving this problem by making an Arduino based color sorting machine using the TCS3200 color sensor. The implementation makes use of micro servo motors for actuation, color sensor for determining color and microcontroller for logic-based control of all the components. For the construction of mechanical structure, sturdy whiteboards will be used. The application of this machine can be found in lots of manufacturing industries, especially the food processing industry. Rice, coffee, grains and vegetables are the best suited products for sorting based on color. By determining the color of grains, the freshness and quality of product can be obtained. The defective items can then be removed

CHAPTER 2

WORKING PRINCIPLE

2.1 WORKING PRINCIPLE

This project works on the principle that every color is composed of 3 fundamental colors- red, green and blue. By determining the values of these 3 components, colors can be differentiated from one another. For this task the TCS3200 color sensor has been used. It is basically a light to frequency converter that converts the incident light on its photodiode array into frequencies of RGB. Using a pre-programmed logic in the microcontroller, these values are then compared with pre-defined values to determine the color. Using micro servo-motors, the objects are then placed in respective containers.

The process of sorting begins at the hopper where the objects are placed. A mechanical arm controls the dropping of objects and then guides them along a guiding rail to the color sensor platform. Here, the color determines the RGB frequencies of object and sends this information to the microcontroller. The microcontroller then determines the color of the object and then actuated the bottom servo motor to place it in the respective container.

The whole system can be simplified into 4 units:

2.1.1 Input Unit

The essential employment of this unit is to transport the object to sensor unit, while the object goes to the sensor unit the manual rail desires to prevent. The hopper and top servomotor form this unit.

2.1.2 Processing Unit

This unit consists of the TCS3200 color sensor which determines the color of input object. It passes this information to the control unit i.e the Arduino Uno board. It is fed by the input unit and passes the object to output unit.

2.1.3 Output Unit

This unit consists of bottom servomotor and the sorted boxes of items. Based on color of the object in processing unit the bottom servomotor moves in required direction facing towards the respective box containing sorted objects. The object then falls into it and the process is complete.

2.1.4 Control Unit

This unit consists of Arduino Uno board which is the heart of the system. It controls all the above units.

2.2 BLOCK DIAGRAM

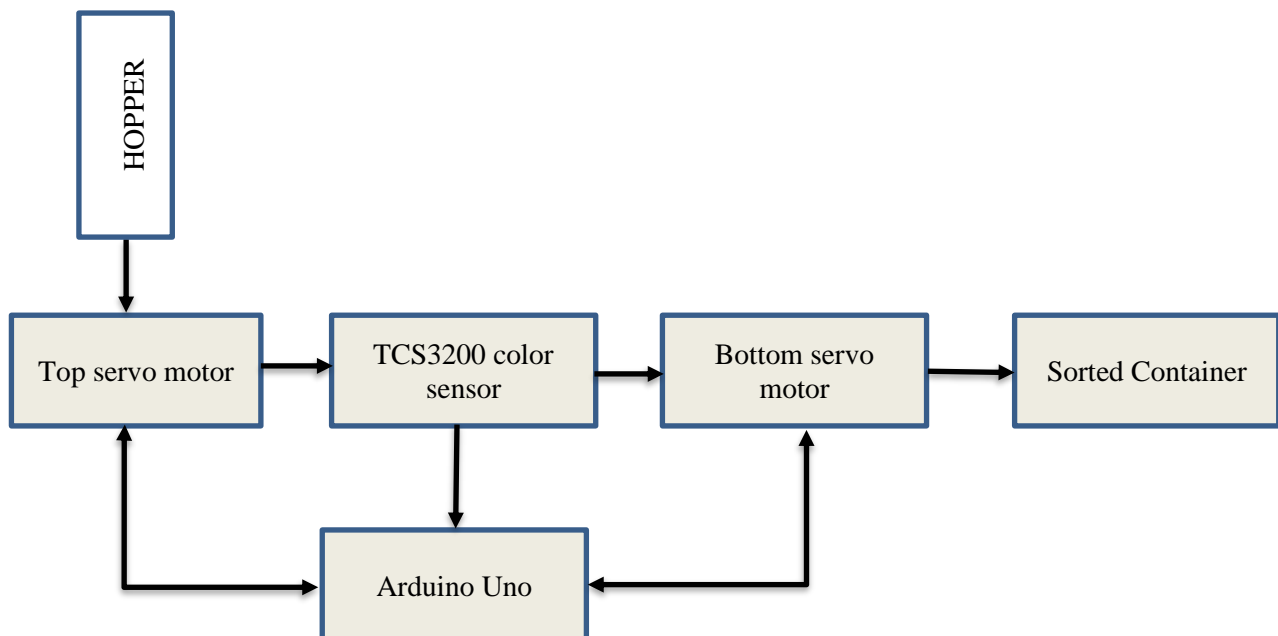


Fig 2.1 Basic Block Diagram

2.2.1 Hopper

It is a long cylindrical plastic tube which contains the objects to be sorted.

2.2.2 Top servo motor

It is a SG90 micro servomotor which actuates the which controls dropping of object from hopper and takes it to the color sensing platform.

2.2.3 TCS3200 color sensor

It determines the color of the object and passes this information to the microcontroller.

2.2.4 Bottom servo motor

Another SG90 micro servo motor that guides the object to its respective sorted container.

2.2.5 Sorted container

They contain objects of every particular color

2.2.6 Arduino Uno

It is a microcontroller board which contains the program of machine and controls all the components of the machine.

2.3 CIRCUIT DIAGRAM

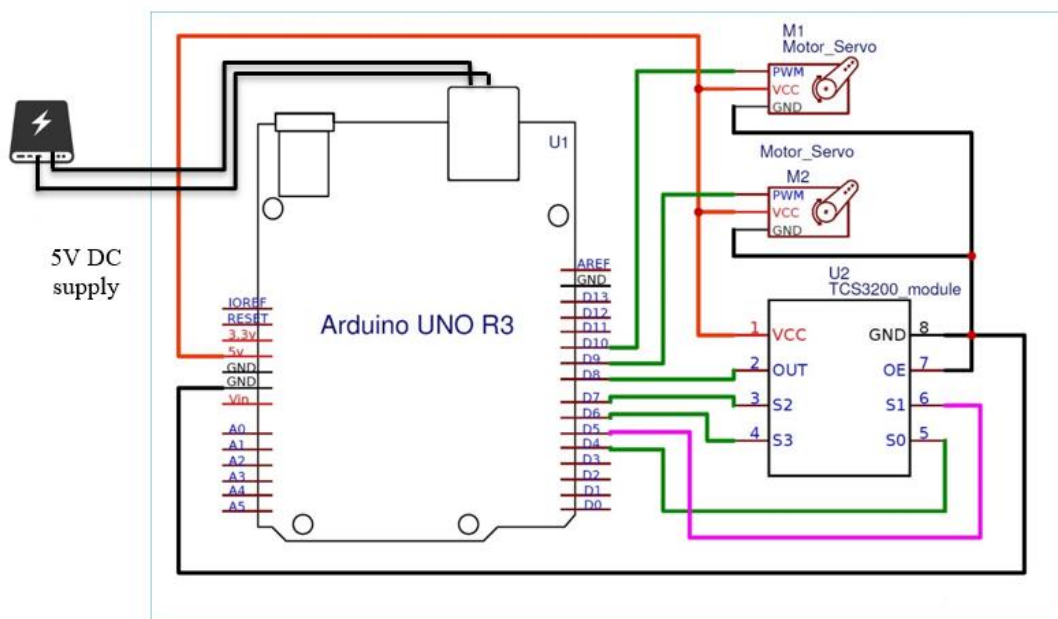


Fig 2.2 Circuit Diagram

CHAPTER 3

COMPONENT LIST

3.1 ARDUINO UNO

Arduino is an open-source electronics platform based on easy-to-use hardware and software. Arduino boards are able to read inputs - light on a sensor, a finger on a button, or a Twitter message - and turn it into an output - activating a motor, turning on an LED, publishing something online. You can tell your board what to do by sending a set of instructions to the microcontroller on the board. To do so you use the Arduino programming language (based on Wiring), and the Arduino Software (IDE), based on Processing.

Over the years Arduino has been the brain of thousands of projects, from everyday objects to complex scientific instruments. A worldwide community of makers - students, hobbyists, artists, programmers, and professionals - has gathered around this open-source platform, their contributions have added up to an incredible amount of accessible knowledge that can be of great help to novices and experts alike.

Arduino was born at the Ivrea Interaction Design Institute as an easy tool for fast prototyping, aimed at students without a background in electronics and programming. As soon as it reached a wider community, the Arduino board started changing to adapt to new needs and challenges, differentiating its offer from simple 8-bit boards to products for IoT applications, wearable, 3D printing, and embedded environments. All Arduino boards are completely open-source, empowering users to build them independently and eventually adapt them to their particular needs. The software, too, is open-source, and it is growing through the contributions of users worldwide

its simple and accessible user experience, Arduino has been used in thousands of different projects and applications. The Arduino software is easy-to-use for beginners, yet flexible enough for advanced users. It runs on Mac, Windows, and Linux. Teachers and students use it to build low cost scientific instruments, to prove chemistry and physics principles, or to get started with programming and robotics. Designers and architects build interactive prototypes, musicians and artists use it for installations and to experiment with new musical instruments. Makers, of course, use it to build many of the projects exhibited at the Maker Faire, for example. Arduino is a key tool to learn new things. Anyone - children, hobbyists, artists, programmers - can start tinkering just following the step-by-step instructions of a kit, or sharing ideas online with other members of the Arduino community.

There are many other microcontrollers and microcontroller platforms available for physical computing. Parallax Basic Stamp, Netmedia's BX-24, Phidgets, MIT's Handyboard, and many others offer similar functionality. All of these tools take the messy details of microcontroller programming and wrap it up in an easy-to-use package. Arduino also simplifies the process of working with microcontrollers, but it offers some advantage for teachers, students, and interested amateurs over other systems:

1. **Inexpensive** - Arduino boards are relatively inexpensive compared to other microcontroller platforms. The least expensive version of the Arduino module can be assembled by hand, and even the pre-assembled Arduino modules cost less than Rs. 1000
2. **Cross-platform** - The Arduino Software (IDE) runs on Windows, Macintosh OSX, and Linux operating systems. Most microcontroller systems are limited to Windows.
3. **Simple, clear programming environment** - The Arduino Software (IDE) is easy-to-use for beginners, yet flexible enough for advanced users to take advantage of as well. For teachers, it's conveniently based on the Processing programming environment, so students learning to program in that environment will be familiar with how the Arduino IDE works.
4. **Open source and extensible software** - The Arduino software is published as open-source tools, available for extension by experienced programmers. The language can be expanded through C++ libraries, and people wanting to understand the technical details can make the leap from Arduino to the AVR C programming language on which it's based. Similarly, you can add AVR-C code directly into your Arduino programs if you want to.
5. **Open source and extensible hardware** - The plans of the Arduino boards are published under a Creative Commons license, so experienced circuit designers can make their own version of the module, extending it and improving it. Even relatively inexperienced users can build the breadboard version of the module in order to understand how it works and save money.

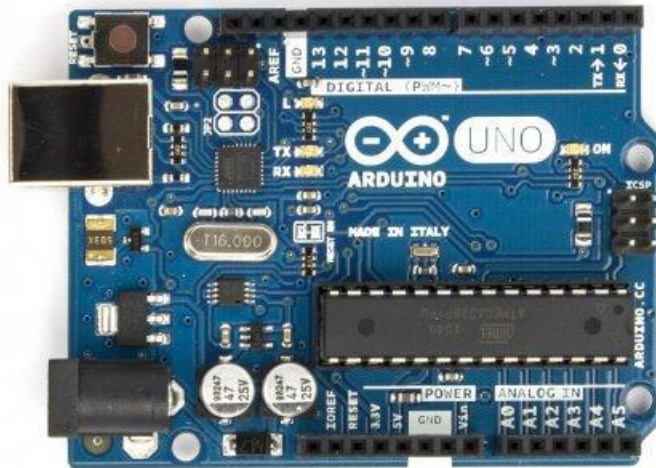


Fig 3.1 Arduino UNO

3.1.1 SPECIFICATIONS

Microcontroller:	ATmega328
Operating Voltage:	5V
Input Voltage (recommended):	7-12V
Input Voltage (limits):	6-20V
Digital I/O Pins:	14 (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
SRAM:	2 KB
EEPROM:	1 KB
Clock Speed:	16 MHz

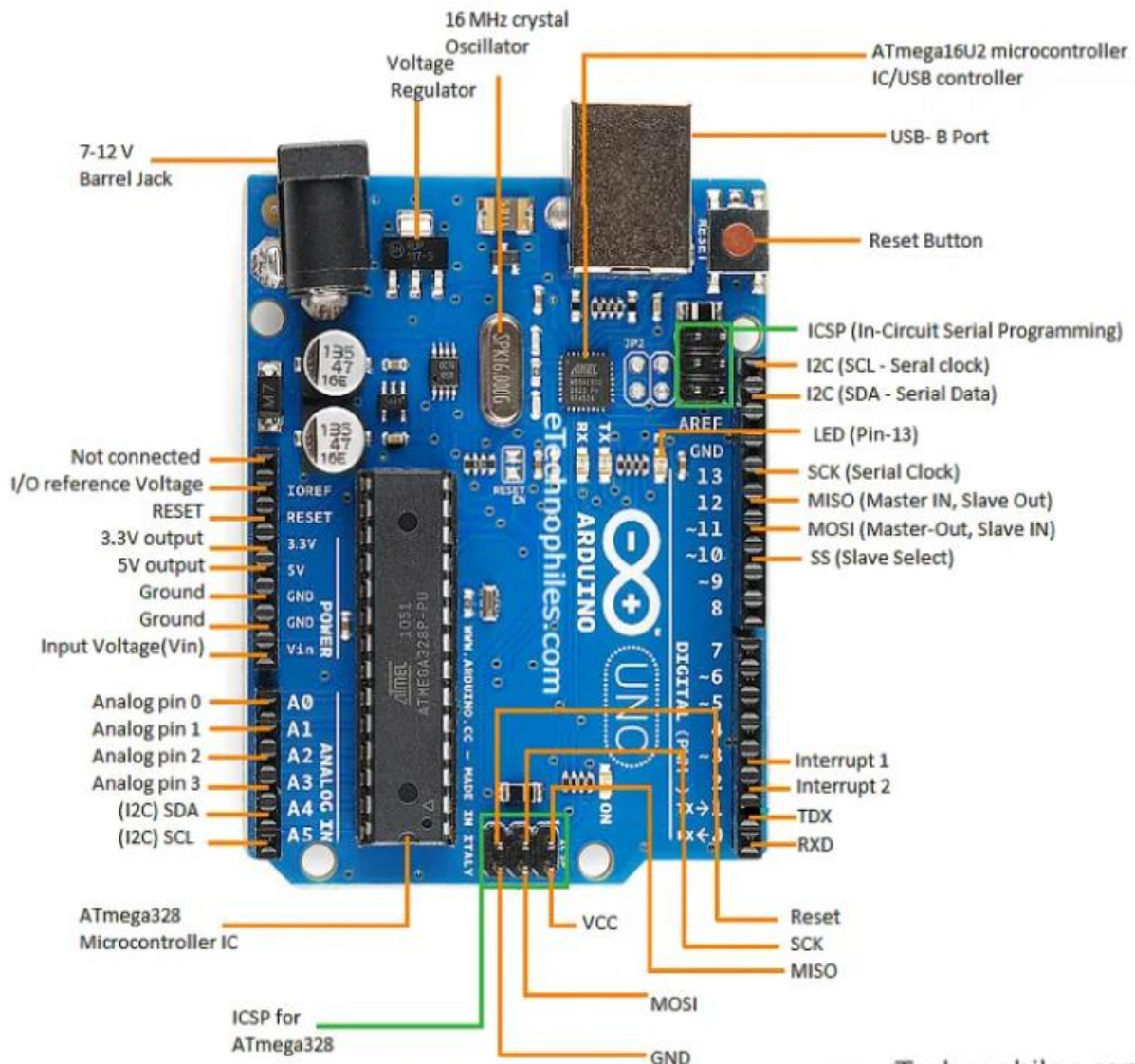


Fig 3.2 Arduino UNO board pin diagram

3.1.2 General Pin functions:

1. **LED:** There is a built-in LED driven by digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.
2. **VIN:** The input voltage to the Arduino/Genuino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.

3. **5V:** This pin outputs a regulated 5V from the regulator on the board. The board can be supplied with power either from the DC power jack (7 - 20V), the USB connector (5V), or the VIN pin of the board (7-20V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage the board.
4. **3V3:** A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA
5. **GND:** Ground pins.
6. **IOREF:** This pin on the Arduino/Genuino board provides the voltage reference with which the microcontroller operates. A properly configured shield can read the IOREF pin voltage and select the appropriate power source or enable voltage translators on the outputs to work with the 5V or 3.3V.
7. **Reset:** Typically used to add a reset button to shields which block the one on the board.

3.1.2 Special Pin Functions

Each of the 14 digital pins and 6 Analog pins on the Uno can be used as an input or output, using pinMode(), digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive 20 mA as recommended operating condition and has an internal pull-up resistor (disconnected by default) of 20-50k ohm. A maximum of 40mA is the value that must not be exceeded on any I/O pin to avoid permanent damage to the microcontroller. The Uno has 6 analog inputs, labeled A0 through A5, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though it is possible to change the upper end of their range using the AREF pin and the analogReference() function.

3.1.3 In addition, some pins have specialized functions:

1. **Serial:** pins 0 (RX) and 1 (TX). Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.
2. **External Interrupts:** pins 2 and 3. These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value.

3. **PWM(Pulse Width Modulation)** 3, 5, 6, 9, 10, and 11 Can provide 8-bit PWM output with the `analogWrite()` function.
4. **SPI(Serial Peripheral Interface)**: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). These pins support SPI communication using the SPI library.
5. **TWI(Two Wire Interface)**: A4 or SDA pin and A5 or SCL pin. Support TWI communication using the Wire library.
6. **AREF(Analog Reference)**: Reference voltage for the analog inputs.

3.1.4 Communication:

The Arduino Uno has a number of facilities for communicating with a computer, another Arduino board, or other microcontrollers. The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). 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, a .inf file is required.

The Arduino Software (IDE) includes a serial monitor which allows simple textual data to be sent to and from the 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 SoftwareSerial library allows serial communication on any of the Uno's digital pins.

3.1.5 Automatic (Software) Reset:

Rather than requiring a physical press of the reset button before an upload, the Arduino Uno board is designed in a way that allows it to be reset by software running on a connected to a

computer. One of the hardware flow control lines (DTR) of the ATmega8U2/16U2 is connected to the reset line of the ATmega328 via a 100 nano farad capacitor.

When this line is asserted (taken low), the reset line drops long enough to reset the chip. This setup has other implications. When the Uno is connected to either a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following half-second or so, the bootloader is running on the Uno. While it is programmed to ignore malformed data (i.e. anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after a connection is opened.

3.2 TCS 3200 COLOUR SENSOR



Fig 3.3 TCS3200 Colour Sensor

3.2.1 How Color Sensors Work

white light is made up of three primary colors (Red, green and blue), which have different wavelengths. These colors combine with each other to form different shades of color. When white light falls on any surface, some wavelengths of light are absorbed and some are reflected, depending on the properties of the surface material. The color we see is a result of which wavelengths are reflected back into our eyes.

Now coming back to the sensor, a typical color sensor includes a high-intensity white LED that projects a modulated light onto the object. To detect the color of reflected light, almost all the color sensors consist of a grid of color-sensitive filter, also known as '**Bayer Filter**' and an array of photodiodes underneath, as shown in the picture below.

A single pixel is made up of 4 filters, one red, one blue, one green and one clear filter (no filter). This pattern is also known as the '**Bayer Pattern**'. Each filter passes light of just a single color to the photodiode beneath, while the clear filter passes light as it is, as shown below. This extra light passed through the clear filter is a major advantage in low light conditions.

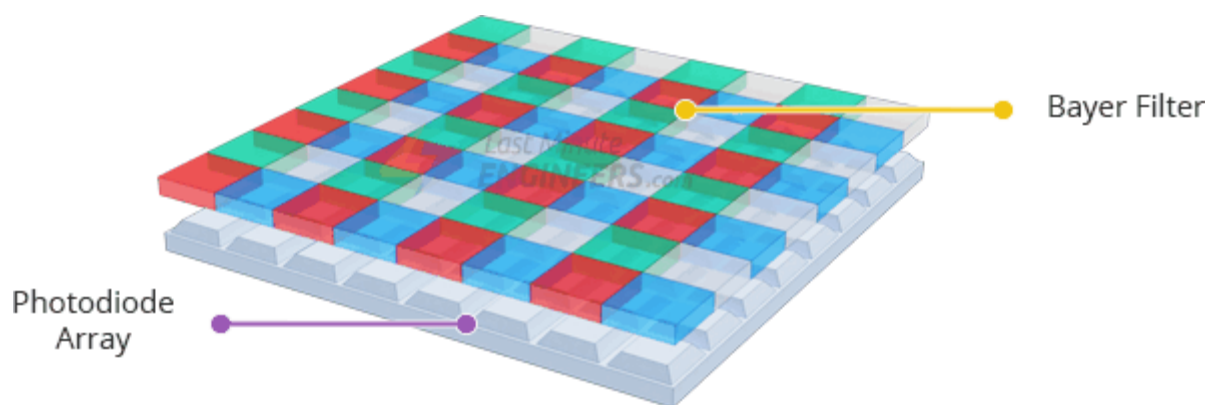


Fig 3.4 Photodiode Array

The processing chip then addresses each photodiode (one color at a time), and measures the intensity of the light. As there is an array of photodiodes, the results are first averaged and then sent out for processing. By measuring the relative level of red, green and blue light, the color of the object is determined.

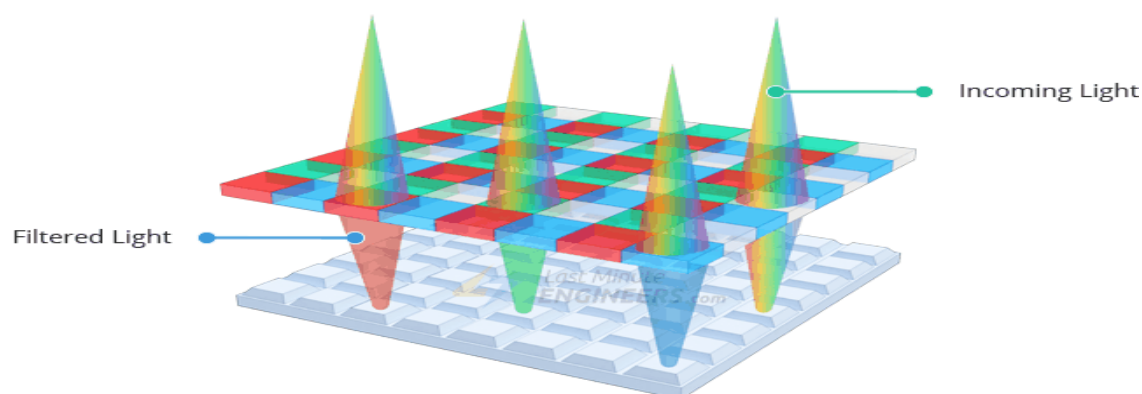


Fig 3.5 Working of Photodiode Array

3.2.2 TCS3200 Design

At the heart of the module is an inexpensive RGB sensor chip from Texas Advanced Optoelectronic Solutions – TCS230. The TCS230 Color Sensor is a complete color detector that can detect and measure an almost infinite range of visible colors. The sensor itself can be seen at the center of the module, surrounded by the four white LEDs. The LEDs light up when the module is powered up and are used to illuminate the object being sensed. Thanks to these LEDs, the sensor can also work in complete darkness to determine the color or brightness of the object. The TCS230 operates on a supply voltage of 2.7 to 5.5 volts and provides TTL logic-level outputs.

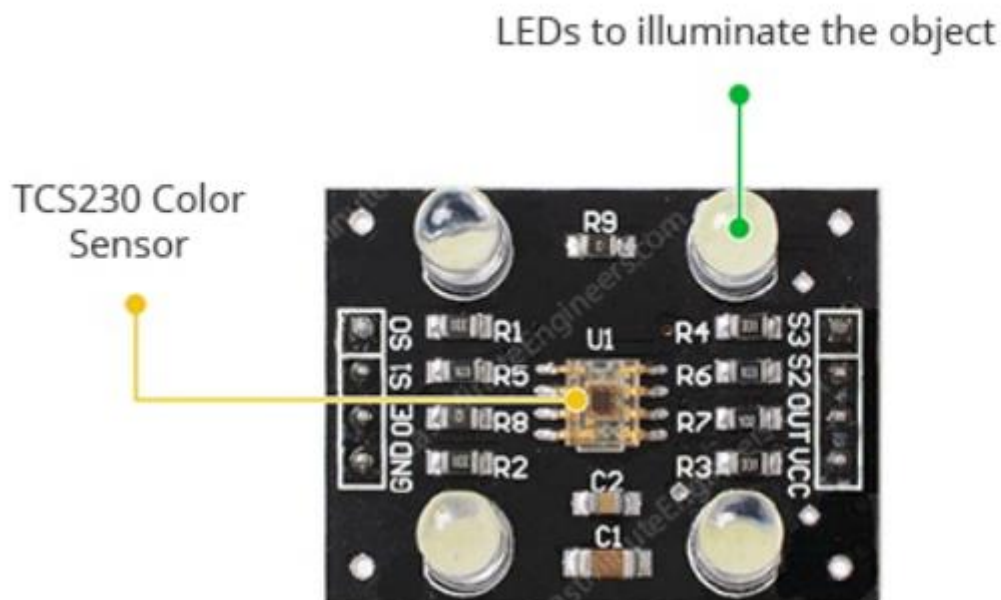


Fig 3.6 TCS3200 Design

3.2.3 Operation

The output of TCS3200 is a square wave with 50% duty cycle. TCS3200 can be interfaced with any microcontroller directly using digital input and digital output pins. Using two control input pins, the full-scale output frequency can be scaled. From the four different types of filter covered photodiodes, each diode can be activated using S2 and S3 selection inputs.

When the diodes with red filter are chosen, only red incident light is measured and the remaining green and blue light are blocked. Then by measuring the frequency, the intensity of the red light can be detected. Using S0, S1 select inputs, frequency scaling factor can be set.

To measure the frequency, the 6th pin is used. This pin is connected to the microcontroller. To determine the color of the object, first set all the input pins as input and output pins as output. Here there is no use of analog pins. Next set the desired frequency scaling by setting the pins S0, S1 either high or low.

Now to detect the color of the object, activate each type of filter using S2 and S3 selection lines. After activation of filter measure the frequency generated on the 6th pin using a microcontroller. When both S2 and S3 pins are low, Red color filters are activated and the intensity of red components of the object is detected. Similarly, driving S2 low and S3 high detects Blue component and driving both S2, S3 to High detects Green components of the object. All these values are collected and compared to get the actual color of the object. Ambient light can cause variations in the measurements so sensor and object should be protected from the ambient light during measurement.

0.01 μ F to 0.1 μ F capacitors should be used for decoupling the power supply lines. For input noise immunity a low impedance electrical connection is required at the device output and device ground. A buffer is required if lines greater than 12 inches are used at the output.

The TCS230 detects colour with the help of an 8 x 8 array of photodiodes, of which sixteen photodiodes have red filters, 16 photodiodes have green filters, 16 photodiodes have blue filters, and remaining 16 photodiodes are clear with no filters. If you look closely at the sensor, you can actually see these filters.

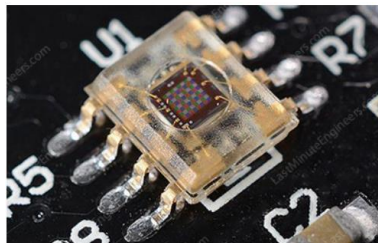


Fig 3.7 TCS3200 Filter

Each 16 photodiodes are connected in parallel. So, we can choose which of the them to read.

If you want to detect only red colour, you can select 16 red filtered photodiodes by setting two pins LOW according to the table. Similarly, you can choose different types of photodiodes by different combinations of S2 and S3.

S2	S3	Photodiode type
LOW	LOW	Red
LOW	HIGH	Blue
HIGH	LOW	Clear (No filter)
HIGH	HIGH	Green

An internal current-to-frequency converter converts readings from photodiodes into a square wave whose frequency is proportional to the intensity of the chosen colour. The range of the typical output frequency is 2HZ~500KHZ.

S0	S1	Output frequency scaling
LOW	LOW	Power down
LOW	HIGH	2%
HIGH	LOW	20%
HIGH	HIGH	100%

The sensor has two more control pins, S0 and S1, which are used for scaling the output frequency. The frequency can be scaled to three different preset values of 2%, 20% or 100%. This frequency-scaling function allows the sensor to be used with a variety of microcontrollers and other devices.

You can get different scaling factor by different combinations of S0 and S1. For the Arduino most applications use the 20% scaling.

3.2.4 Pin Diagram

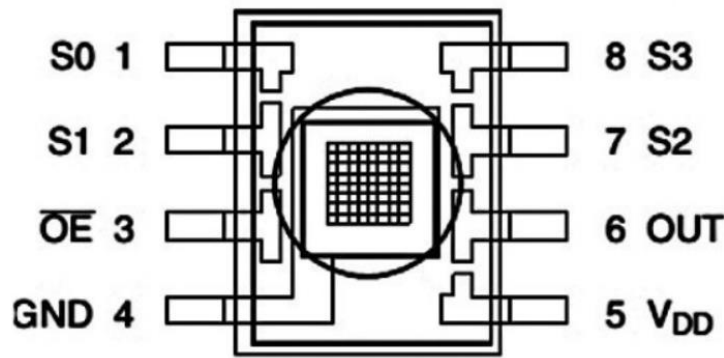


Fig 3.8 TCS3200 Pin Diagram

- Pin-1 and Pin-2 are S0, S1 selection lines respectively. These pins are used for frequency scaling.
- Pin-3, OE, is the output enable pin. This pin is active low.
- Pin-4, GND, is the ground pin. This pin is the power supply ground.
- Pin-5, VDD, is the supply voltage pin.
- Pin-6, OUT, is the output frequency pin. This pin is connected to the microcontroller to read values.
- Pin-7 and Pin-8 are the S2, S3 selection lines respectively. These pins are used as Photodiode type selection inputs.

3.2.5 Specifications

1. The main blocks of this module are TCS3200 RGB sensor chip and 4 white LEDs.
2. The four LED's are given to provide sufficient lighting conditions for the sensor during colour detection.
3. TCS3200 chip consists of 8×8 array of photodiodes which can detect red, blue, green colours.
4. This module works on the input supply voltage of 2.7V to 5.5V.
5. TCS3200 chip converts light intensity into the frequency with high resolution.
6. This module doesn't require an ADC to get digital values and can be connected to digital pins of microcontrollers directly.
7. TCS3200 has a programmable colour and full-scale output frequency.
8. Power down feature is also given to this module.
9. The operating temperature range of this module is from -40°C to 85°C.

3.2.5 Wiring TCS230 Color Sensor to Arduino UNO

Wiring up the TCS 230 to an Arduino is very simple. Every pin is used except the Output Enable pin, and the module is powered safely from the 5-volt output of the Arduino. None of the pins used on the Arduino are critical because the module does not require any pin-specific features, so if you want to use different pins you can do so safely. Just be sure to change the pin numbers in the code to reflect any changes to the wiring.

We actually use two sketches to work with the TCS230 colour sensor.

1. The first sketch (calibration sketch) will help us to obtain the raw data from the sensor.
2. The second sketch (main Arduino sketch) will use the raw data previously received to display RGB values for the colour being sensed.

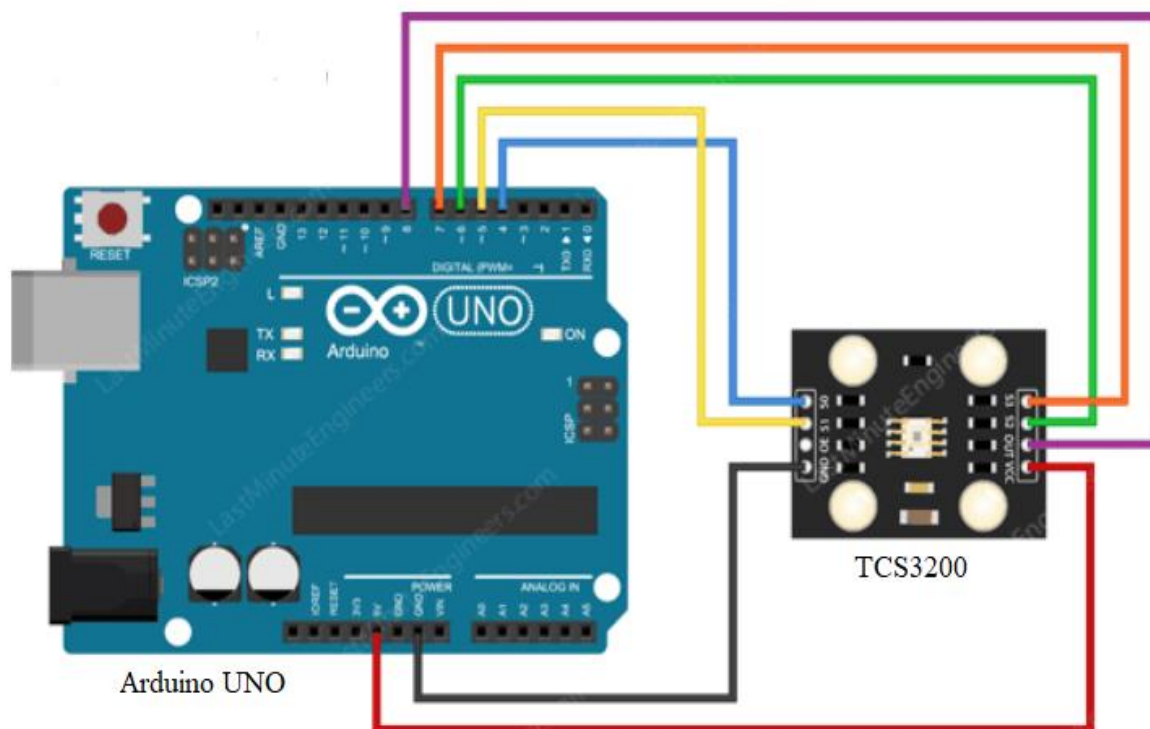


Fig 3.9 Wiring TCS230 Color Sensor to Arduino UNO

3.2.5 Applications of TCS3200

1. TCS3200 is used to detect the colour of the surfaces.
2. This module finds application in Industries, health care, and manufacturing plants.
3. TCS3200 is applied in medical diagnosis systems.
4. For RGB LED consistency control TCS3200 is used.
5. TCS3200 is applied for industrial process control.
6. In laser edge banding machines to detect colour TCS3200 is used.
7. To detect chronic kidney diseases TCS3200 is used for urine analysis.
8. TCS3200 is used in fruit sorting systems.
9. The intensity of blue, red, green radiations can be measured using this sensor module.
10. For classifying different types of metals this module is used.
11. The food industry uses this sensor for sorting of fruits and vegetables.
12. TCS3200 is also used in dental diagnosis and for ammonia detection.
13. For designing of Light-to-Frequency receiver in Multi colour visible light communication, TCS3200 is used.

3.3 SG90 SERVO MOTOR



Fig 3.10 SG90 Micro Servo

A **servo motor** is a type of motor that can rotate with great precision. If you want to rotate an object at some specific angles or distance, then you use a servo motor. It is just made up of a simple motor which runs through a **servo mechanism**.

3.3.1 Servo Motor Working Principle

A servo consists of a Motor (DC or AC), a potentiometer, gear assembly, and a controlling circuit. First of all, we use gear assembly to reduce RPM and to increase torque of the motor. Say at initial position of servo motor shaft, the position of the potentiometer knob is such that there is no electrical signal generated at the output port of the potentiometer. Now an electrical signal is given to another input terminal of the error detector amplifier. Now the difference between these two signals, one comes from the potentiometer and another comes from other sources, will be processed in a feedback mechanism and output will be provided in terms of error signal. This error signal acts as the input for motor and motor starts rotating. Now motor shaft is connected with the potentiometer and as the motor rotates so the potentiometer and it will generate a signal. So as the potentiometer's angular position changes, its output feedback signal changes. After sometime the position of potentiometer reaches at a position that the output of potentiometer is same as external signal provided. At this condition, there will be no output signal from the amplifier to the motor input as there is no difference between external applied signal and the signal generated at potentiometer, and in this situation motor stops rotating.

3.3.2 Servo Motor Working Mechanism

It consists of three parts:

1. Controlled device
2. Output sensor
3. Feedback system

It is a closed-loop system where it uses a positive feedback system to control motion and the final position of the shaft. Here the device is controlled by a feedback signal generated by comparing output signal and reference input signal.

Here reference input signal is compared to the reference output signal and the third signal is produced by the feedback system. And this third signal acts as an input signal to the control the device. This signal is present as long as the feedback signal is generated or there is a difference between the reference input signal and reference output signal.

3.3.3 Controlling Servo Motor

All motors have three wires coming out of them. Out of which two will be used for Supply (positive and negative) and one will be used for the signal that is to be sent from the MCU.

Servo motor is controlled by PWM (Pulse with Modulation) which is provided by the control wires. There is a minimum pulse, a maximum pulse and a repetition rate. Servo motor can turn 90 degree from either direction from its neutral position. The servo motor expects to see a pulse every 20 milliseconds (ms) and the length of the pulse will determine how far the motor turns. For example, a 1.5ms pulse will make the motor turn to the 90° position, such as if pulse is shorter than 1.5ms shaft moves to 0° and if it is longer than 1.5ms than it will turn the servo to 180°.

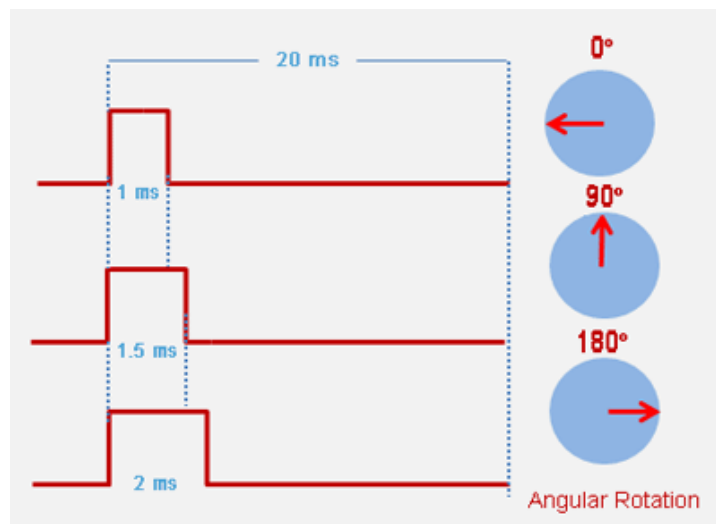


Fig 3.11 Controlling SG90

Servo motor works on **PWM (Pulse width modulation)** principle, means its angle of rotation is controlled by the duration of applied pulse to its Control PIN. Basically servo motor is made up of **DC motor which is controlled by a variable resistor (potentiometer) and some gears**. High speed force of DC motor is converted into torque by Gears. We know that $WORK = FORCE \times DISTANCE$, in DC motor Force is less and distance (speed) is high and in Servo, force is High and distance is less. The potentiometer is connected to the output shaft of the Servo, to calculate the angle and stop the DC motor on the required angle.

Servo motor can be rotated from 0 to 180 degrees, but it can go up to 210 degrees, depending on the manufacturing. This degree of rotation can be controlled by applying the **Electrical Pulse** of proper width, to its Control pin. Servo checks the pulse in every 20 milliseconds. The pulse of 1 ms (1 millisecond) width can rotate the servo to 0 degrees, 1.5 ms can rotate to 90 degrees (neutral position) and 2 ms pulse can rotate it to 180 degree.

3.3.4 Specifications

- Operating Voltage: 3V to 7.2V
- Stall torque at 4.8V: 1.2 kg-cm
- Stall torque at 6.6V: 1.6 kg-c

3.3.5 Motor Wire Connections



Fig 3.11 Wire Connections

- Orange - PWM
- Red - Supply
- Brown - Ground

CHAPTER 4

PROGRAMMING LOGIC AND FLOWCHART

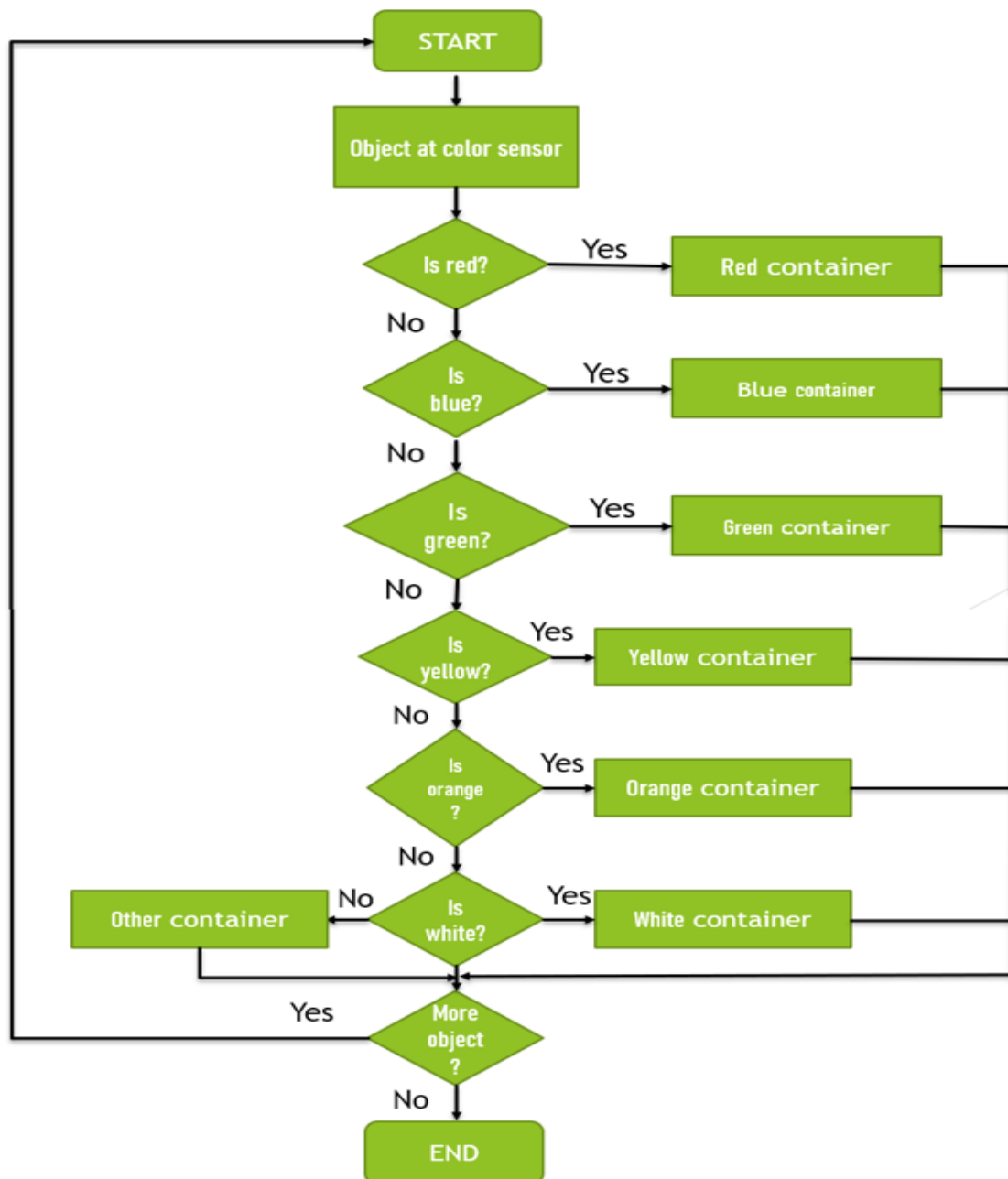


Fig 4.1 Flowchart

Fig 4.1 shows the flow chart of program. The program begins at the color values sent by color sensor to the microcontroller. Using conditional statements these values are compared with pre-defined values of 6 pre-defined colors- red, green, blue, yellow, orange and white. If any of the conditions are satisfied, then the object is put into the respective container. If none of the conditions are satisfied, then the object is put in other color container.

The first condition will be checked. If the color of the object is red, then it will move to red container. If not then the second condition will be checked. If the color is blue, then the object will be moved to blue container. If not, then 3rd condition will be checked. If color is green then it will move to green container and this way, object will be checked for yellow, orange and white color. And if all the conditions fails, then since the object is neither of the pre-defined colors, the object will move to the other color container. After the objects gets dropped on its respective color container, a condition will be checked whether there are more objects in the hopper or not. If there will be more objects in the hopper, then the same process will be repeated for another object. This way all the objects will be sorted on basis of color. If there is no object in the hopper then the program will terminate and the machine will shut down. This way, all the objects present in the hopper will be sorted on the basis of color.

4.1 Pseudocode

```
program starts  
initialize pins  
while hopper is not empty  
    if color == red then  
        bottom servo.rotate(red_angle)  
    else if color == blue then  
        bottom servo.rotate(blue_angle)  
    else if color == green then  
        bottom servo.rotate(green_angle)  
    else if color == yellow then  
        bottom servo.rotate(yellow_angle)  
    else if color == orange then  
        bottom servo.rotate(orange_angle)  
    else if color == white then  
        bottom servo.rotate(white_angle)  
    else  
        bottom servo.rotate(other_angle)  
end while  
program end
```

4.2 Programming Logic

The program starts. All the pins will be initialized. A loop will be used to sort the objects one by one in the respective containers on the basis of colour. If the statement in the loop is true, the loop executes and, inside the loop, using if-else-if block, we sort the objects into its respective colour containers. This way, all the objects will be sorted. Every time the loop executes, the hopper will be checked for the objects. If there are no more objects in the hopper, the loop condition fails and the program terminates.

CHAPTER 5

APPLICATIONS

Application of this project can be seen in multiple sector such as in agricultural sector, food processing sector and in industrial sector.

In agricultural sector, we can consider a case where a farmer produces rice and grains etc. Now, the produced grains will be containing unwanted early materials such as mud, stones etc. These unwanted materials will be sorted based on the colour of the grain. The colour will be fed through the microcontroller and depending on the colour fed, all the unwanted materials will be removed.

In industrial sector, suppose there is a factory manufacturing balls, all of different colours. These colour balls, if sorted based on colour manually using physical labour, will take lot of time and energy and thus the cost of manufacturing will also increase drastically. By using the colour sorting machine, all the different colour balls will be sorted accordingly and the manufacturing process and packaging process will be enhanced drastically.

This project can be used for the industrial purpose in order to reduce the manual labor and improve automation. As we know, sorting of objects on the basis of color is a very frequent task in the industry of almost all sector such as food processing, rice, grains, pulses packaging, etc. This project not only help in reducing manual labor but also reduces production cost by reducing the number or labors. Above all, this machine also improves time-efficiency by automating the process making sorting way faster than when done manually.

CHAPTER 6

CONCLUSION

Sorting of objects based on color is an essential mechanical process in the manufacturing industry. Manually doing this task consumes a lot of time, money and energy. Product consistency and quality is also very poor. To obtain high quality product with low cost and time investment, a machine which can automate this task is the need of the day. So, the objective of this project is to make and demonstrate a color sorting machine that can be used in real world applications. Thus, this project can be used for the industrial purpose in order to reduce the manual labor and improve automation. As we know, sorting of objects on the basis of color is a very frequent task in the industry of almost all sector such as food processing, rice, grains, pulses packaging, etc. This project not only help in reducing manual labor but also reduces production cost by reducing the number of labors. Above all, this machine also improves time-efficiency by automating the process making sorting way faster than when done manually.

APPENDIX

1. ATmega328P Instruction Set Summary

Mnemonics	Operands	Description	Operation	Flags	Clocks
ARITHMETIC AND LOGIC INSTRUCTIONS					
ADD	Rd, Rr	Add two Registers	$Rd \leftarrow Rd + Rr$	Z,C,N,V,H	1
ADC	Rd, Rr	Add with Carry two Registers	$Rd \leftarrow Rd + Rr + C$	Z,C,N,V,H	1
ADIW	RdI,K	Add Immediate to Word	$RdH:RdL \leftarrow RdH:RdL + K$	Z,C,N,V,S	2
SUB	Rd, Rr	Subtract two Registers	$Rd \leftarrow Rd - Rr$	Z,C,N,V,H	1
SUBI	Rd, K	Subtract Constant from Register	$Rd \leftarrow Rd - K$	Z,C,N,V,H	1
SBC	Rd, Rr	Subtract with Carry two Registers	$Rd \leftarrow Rd - Rr - C$	Z,C,N,V,H	1
SBCI	Rd, K	Subtract with Carry Constant from Reg.	$Rd \leftarrow Rd - K - C$	Z,C,N,V,H	1
SBIW	RdI,K	Subtract Immediate from Word	$RdH:RdL \leftarrow RdH:RdL - K$	Z,C,N,V,S	2
AND	Rd, Rr	Logical AND Registers	$Rd \leftarrow Rd \bullet Rr$	Z,N,V	1
ANDI	Rd, K	Logical AND Register and Constant	$Rd \leftarrow Rd \bullet K$	Z,N,V	1
OR	Rd, Rr	Logical OR Registers	$Rd \leftarrow Rd \vee Rr$	Z,N,V	1
ORI	Rd, K	Logical OR Register and Constant	$Rd \leftarrow Rd \vee K$	Z,N,V	1
EOR	Rd, Rr	Exclusive OR Registers	$Rd \leftarrow Rd \oplus Rr$	Z,N,V	1
COM	Rd	One's Complement	$Rd \leftarrow 0xFF - Rd$	Z,C,N,V	1
NEG	Rd	Two's Complement	$Rd \leftarrow 0x00 - Rd$	Z,C,N,V,H	1
SBR	Rd,K	Set Bit(s) in Register	$Rd \leftarrow Rd \vee K$	Z,N,V	1
CBR	Rd,K	Clear Bit(s) in Register	$Rd \leftarrow Rd \bullet (0xFF - K)$	Z,N,V	1
INC	Rd	Increment	$Rd \leftarrow Rd + 1$	Z,N,V	1
DEC	Rd	Decrement	$Rd \leftarrow Rd - 1$	Z,N,V	1
TST	Rd	Test for Zero or Minus	$Rd \leftarrow Rd \bullet Rd$	Z,N,V	1
CLR	Rd	Clear Register	$Rd \leftarrow Rd \oplus Rd$	Z,N,V	1
SER	Rd	Set Register	$Rd \leftarrow 0xFF$	None	1
MUL	Rd, Rr	Multiply Unsigned	$R1:R0 \leftarrow Rd \times Rr$	Z,C	2
MULS	Rd, Rr	Multiply Signed	$R1:R0 \leftarrow Rd \times Rr$	Z,C	2
MULSU	Rd, Rr	Multiply Signed with Unsigned	$R1:R0 \leftarrow Rd \times Rr$	Z,C	2
FMUL	Rd, Rr	Fractional Multiply Unsigned	$R1:R0 \leftarrow (Rd \times Rr) \ll 1$	Z,C	2
FMULS	Rd, Rr	Fractional Multiply Signed	$R1:R0 \leftarrow (Rd \times Rr) \ll 1$	Z,C	2
FMULSU	Rd, Rr	Fractional Multiply Signed with Unsigned	$R1:R0 \leftarrow (Rd \times Rr) \ll 1$	Z,C	2
BRANCH INSTRUCTIONS					
RJMP	k	Relative Jump	$PC \leftarrow PC + k + 1$	None	2
IJMP		Indirect Jump to (Z)	$PC \leftarrow Z$	None	2
JMP ⁽¹⁾	k	Direct Jump	$PC \leftarrow k$	None	3
RCALL	k	Relative Subroutine Call	$PC \leftarrow PC + k + 1$	None	3
ICALL		Indirect Call to (Z)	$PC \leftarrow Z$	None	3
CALL ⁽¹⁾	k	Direct Subroutine Call	$PC \leftarrow k$	None	4
RET		Subroutine Return	$PC \leftarrow STACK$	None	4
RETI		Interrupt Return	$PC \leftarrow STACK$	I	4
CPSE	Rd,Rr	Compare, Skip if Equal	if $(Rd = Rr) PC \leftarrow PC + 2$ or 3	None	1/2/3
CP	Rd,Rr	Compare	$Rd - Rr$	Z, N,V,C,H	1
CPC	Rd,Rr	Compare with Carry	$Rd - Rr - C$	Z, N,V,C,H	1
CPI	Rd,K	Compare Register with Immediate	$Rd - K$	Z, N,V,C,H	1
SBRC	Rr, b	Skip if Bit in Register Cleared	if $(Rr(b)=0) PC \leftarrow PC + 2$ or 3	None	1/2/3
SBRs	Rr, b	Skip if Bit in Register is Set	if $(Rr(b)=1) PC \leftarrow PC + 2$ or 3	None	1/2/3
SBIC	P, b	Skip if Bit in I/O Register Cleared	if $(P(b)=0) PC \leftarrow PC + 2$ or 3	None	1/2/3
SBIS	P, b	Skip if Bit in I/O Register is Set	if $(P(b)=1) PC \leftarrow PC + 2$ or 3	None	1/2/3
BRBS	s, k	Branch if Status Flag Set	if $(SREG(s)=1)$ then $PC \leftarrow PC+k+1$	None	1/2
BRBC	s, k	Branch if Status Flag Cleared	if $(SREG(s)=0)$ then $PC \leftarrow PC+k+1$	None	1/2
BREQ	k	Branch if Equal	if $(Z=1)$ then $PC \leftarrow PC+k+1$	None	1/2
BRNE	k	Branch if Not Equal	if $(Z=0)$ then $PC \leftarrow PC+k+1$	None	1/2
BRCS	k	Branch if Carry Set	if $(C=1)$ then $PC \leftarrow PC+k+1$	None	1/2
BRCC	k	Branch if Carry Cleared	if $(C=0)$ then $PC \leftarrow PC+k+1$	None	1/2
BRSH	k	Branch if Same or Higher	if $(C=0)$ then $PC \leftarrow PC+k+1$	None	1/2
BRLO	k	Branch if Lower	if $(C=1)$ then $PC \leftarrow PC+k+1$	None	1/2
BRMI	k	Branch if Minus	if $(N=1)$ then $PC \leftarrow PC+k+1$	None	1/2
BRPL	k	Branch if Plus	if $(N=0)$ then $PC \leftarrow PC+k+1$	None	1/2
BRGE	k	Branch if Greater or Equal, Signed	if $(N \oplus V=0)$ then $PC \leftarrow PC+k+1$	None	1/2
BRLT	k	Branch if Less Than Zero, Signed	if $(N \oplus V=1)$ then $PC \leftarrow PC+k+1$	None	1/2
BRHS	k	Branch if Half Carry Flag Set	if $(H=1)$ then $PC \leftarrow PC+k+1$	None	1/2
BRHC	k	Branch if Half Carry Flag Cleared	if $(H=0)$ then $PC \leftarrow PC+k+1$	None	1/2
BRTS	k	Branch if T Flag Set	if $(T=1)$ then $PC \leftarrow PC+k+1$	None	1/2
BRTC	k	Branch if T Flag Cleared	if $(T=0)$ then $PC \leftarrow PC+k+1$	None	1/2
BRVS	k	Branch if Overflow Flag is Set	if $(V=1)$ then $PC \leftarrow PC+k+1$	None	1/2
BRVC	k	Branch if Overflow Flag is Cleared	if $(V=0)$ then $PC \leftarrow PC+k+1$	None	1/2

Mnemonics	Operands	Description	Operation	Flags	#Clocks
BRIE	k	Branch if Interrupt Enabled	if (I = 1) then PC ← PC + k + 1	None	1/2
BRID	k	Branch if Interrupt Disabled	if (I = 0) then PC ← PC + k + 1	None	1/2
BIT AND BIT-TEST INSTRUCTIONS					
SBI	P,b	Set Bit in I/O Register	I/O(P,b) ← 1	None	2
CBI	P,b	Clear Bit in I/O Register	I/O(P,b) ← 0	None	2
LSL	Rd	Logical Shift Left	Rd(n+1) ← Rd(n), Rd(0) ← 0	Z,C,N,V	1
LSR	Rd	Logical Shift Right	Rd(n) ← Rd(n+1), Rd(7) ← 0	Z,C,N,V	1
ROL	Rd	Rotate Left Through Carry	Rd(0) ← C, Rd(n+1) ← Rd(n), C ← Rd(7)	Z,C,N,V	1
ROR	Rd	Rotate Right Through Carry	Rd(7) ← C, Rd(n) ← Rd(n+1), C ← Rd(0)	Z,C,N,V	1
ASR	Rd	Arithmetic Shift Right	Rd(n) ← Rd(n+1), n=0...6	Z,C,N,V	1
SWAP	Rd	Swap Nibbles	Rd(3...0) ← Rd(7...4), Rd(7...4) ← Rd(3...0)	None	1
BSET	s	Flag Set	SREG(s) ← 1	SREG(s)	1
BCLR	s	Flag Clear	SREG(s) ← 0	SREG(s)	1
BST	Rr, b	Bit Store from Register to T	T ← Rr(b)	T	1
BLD	Rd, b	Bit load from T to Register	Rd(b) ← T	None	1
SEC		Set Carry	C ← 1	C	1
CLC		Clear Carry	C ← 0	C	1
SEN		Set Negative Flag	N ← 1	N	1
CLN		Clear Negative Flag	N ← 0	N	1
SEZ		Set Zero Flag	Z ← 1	Z	1
CLZ		Clear Zero Flag	Z ← 0	Z	1
SEI		Global Interrupt Enable	I ← 1	I	1
CLI		Global Interrupt Disable	I ← 0	I	1
SES		Set Signed Test Flag	S ← 1	S	1
CLS		Clear Signed Test Flag	S ← 0	S	1
SEV		Set Twos Complement Overflow.	V ← 1	V	1
CLV		Clear Twos Complement Overflow	V ← 0	V	1
SET		Set T in SREG	T ← 1	T	1
CLT		Clear T in SREG	T ← 0	T	1
SEH		Set Half Carry Flag in SREG	H ← 1	H	1
CLH		Clear Half Carry Flag in SREG	H ← 0	H	1
DATA TRANSFER INSTRUCTIONS					
MOV	Rd, Rr	Move Between Registers	Rd ← Rr	None	1
MOVW	Rd, Rr	Copy Register Word	Rd+1:Rd ← Rr+1:Rr	None	1
LDI	Rd, K	Load Immediate	Rd ← K	None	1
LD	Rd, X	Load Indirect	Rd ← (X)	None	2
LD	Rd, X+	Load Indirect and Post-Inc.	Rd ← (X), X ← X + 1	None	2
LD	Rd, -X	Load Indirect and Pre-Dec.	X ← X - 1, Rd ← (X)	None	2
LD	Rd, Y	Load Indirect	Rd ← (Y)	None	2
LD	Rd, Y+	Load Indirect and Post-Inc.	Rd ← (Y), Y ← Y + 1	None	2
LD	Rd, -Y	Load Indirect and Pre-Dec.	Y ← Y - 1, Rd ← (Y)	None	2
LDD	Rd, Y+q	Load Indirect with Displacement	Rd ← (Y + q)	None	2
LD	Rd, Z	Load Indirect	Rd ← (Z)	None	2
LD	Rd, Z+	Load Indirect and Post-Inc.	Rd ← (Z), Z ← Z+1	None	2
LD	Rd, -Z	Load Indirect and Pre-Dec.	Z ← Z - 1, Rd ← (Z)	None	2
LDD	Rd, Z+q	Load Indirect with Displacement	Rd ← (Z + q)	None	2
LDS	Rd, k	Load Direct from SRAM	Rd ← (k)	None	2
ST	X, Rr	Store Indirect	(X) ← Rr	None	2
ST	X+, Rr	Store Indirect and Post-Inc.	(X) ← Rr, X ← X + 1	None	2
ST	-X, Rr	Store Indirect and Pre-Dec.	X ← X - 1, (X) ← Rr	None	2
ST	Y, Rr	Store Indirect	(Y) ← Rr	None	2
ST	Y+, Rr	Store Indirect and Post-Inc.	(Y) ← Rr, Y ← Y + 1	None	2
ST	-Y, Rr	Store Indirect and Pre-Dec.	Y ← Y - 1, (Y) ← Rr	None	2
STD	Y+q, Rr	Store Indirect with Displacement	(Y + q) ← Rr	None	2
ST	Z, Rr	Store Indirect	(Z) ← Rr	None	2
ST	Z+, Rr	Store Indirect and Post-Inc.	(Z) ← Rr, Z ← Z + 1	None	2
ST	-Z, Rr	Store Indirect and Pre-Dec.	Z ← Z - 1, (Z) ← Rr	None	2
STD	Z+q, Rr	Store Indirect with Displacement	(Z + q) ← Rr	None	2
STS	k, Rr	Store Direct to SRAM	(k) ← Rr	None	2
LPM		Load Program Memory	R0 ← (Z)	None	3
LPM	Rd, Z	Load Program Memory	Rd ← (Z)	None	3
LPM	Rd, Z+	Load Program Memory and Post-Inc	Rd ← (Z), Z ← Z+1	None	3
SPM		Store Program Memory	(Z) ← R1:R0	None	-
IN	Rd, P	In Port	Rd ← P	None	1
OUT	P, Rr	Out Port	P ← Rr	None	1
PUSH	Rr	Push Register on Stack	STACK ← Rr	None	2

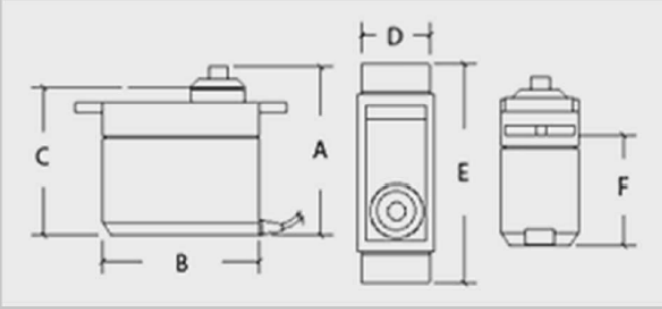
2. Electrical Characteristics of TCS3200

PARAMETER	TEST CONDITIONS	MIN	TYP	MAX	UNIT
V _{OH} High-level output voltage	I _{OH} = - 2 mA	4	4.5		V
V _{OL} Low-level output voltage	I _{OL} = 2 mA	0.25	0.40		V
I _{IH} High-level input current				5	μA
I _{IL} Low-level input current				5	μA
I _{DD} Supply current	Power-on mode		1.4		mA
	Power-down mode		0.1		μA
Full-scale frequency (See Note 4)	S0 = H, S1 = H	500	600		kHz
	S0 = H, S1 = L	100	120		kHz
	S0 = L, S1 = H	10	12		kHz
Temperature coefficient of responsivity	λ ≤ 700 nm, -25°C ≤ T _A ≤ 70°C		± 200		ppm/°C
k _{SVS} Supply voltage sensitivity	V _{DD} = 5 V ±10%		±0.5		%/ V

3. Operating Characteristics at V_{DD} = 5 V, T_A = 25 Degree C, S0 = H, S1 = H values for TCS3200

PARAMETER	TEST CONDITIONS	CLEAR PHOTODIODE S2 = H, S3 = L			BLUE PHOTODIODE S2 = L, S3 = H			GREEN PHOTODIODE S2 = H, S3 = H			RED PHOTODIODE S2 = L, S3 = L			UNIT
		MIN	TYP	MAX	MIN	TYP	MAX	MIN	TYP	MAX	MIN	TYP	MAX	
f _O Output frequency (Note 9)	E _e = 47.2 μW/cm ² , λ _p = 470 nm	12.5	15.6	18.7	61%	84%	22%	43%	0%	6%		kHz		
		(4.7)	(5.85)	(7)										
	E _e = 40.4 μW/cm ² , λ _p = 524 nm	12.5	15.6	18.7	8%	28%	57%	80%	9%	27%				
		(4.7)	(5.85)	(7)										
E _e = 34.6 μW/cm ² , λ _p = 640 nm	13.1	16.4	19.7	5%	21%	0%	12%	84%	105%					
	(4.9)	(6.15)	(7.4)											
R _e Irradiance responsivity (Note 10)	λ _p = 470 nm	331			61%	84%	22%	43%	0%	6%	Hz/ (μW / cm ²)			
		(124)												
	λ _p = 524 nm	386			8%	28%	57%	80%	9%	27%				
		(145)												
	λ _p = 640 nm	474			5%	21%	0%	12%	84%	105%				
		(178)												
Saturation irradiance (Note 11)	λ _p = 470 nm	1813			---	---	---	---	---	μW / cm ²				
		(4839)												
	λ _p = 524 nm	1554			---	---	---	---	---					
		(4138)												
	λ _p = 640 nm	1266			---	---	---	---	---					
		(3371)												
f _D Dark frequency	E _e = 0	2 10			2 10		2 10		2 10		Hz			

4. Dimensions of SG90 Servo Motor

Weight (g)	9	
Torque (kg)	1.5	
Speed(Sec/60deg)	0.09	
A(mm)	30	
B(mm)	23	
C(mm)	27	
D(mm)	12	
E(mm)	33	
F(mm)	16	

5. Arduino IDE

The Arduino Integrated Development Environment (IDE) is a cross platform application (for Windows, Linux, MacOS) that is written in functions from C and C++. It is used to write and upload programs to Arduino compatible boards, but also, with the help of third-party cores, other vendor development boards.

The source code for the IDE is released under the GNU, General Public License, version 2. The Arduino IDE supports the languages C and C++ using special rules of code structuring. The Arduino IDE supplies a software library from the wiring project, which provides many common input and output procedures. User-written code only requires two basic functions, for starting the sketch and the main program loop, that are compiled and linked with a program stub *main()* into an executable cyclic executive program with the GNU toolchain, also included with the IDE distribution. The Arduino IDE employs the program *avrdude* to convert the executable code into a text file in hexadecimal encoding that is loaded into the Arduino board by a loader program in the board's firmware. By default, *avrdude* is used as the uploading tool to flash the user code onto official Arduino boards

REFERENCES

1. “Automatic Colour Sorting Machine Using Arduino Mega Microcontroller”, Aye Myat Myat Myo¹, Zar Chi Soe², International Journal of Latest Technology in Engineering, Management and Applied Science Volume VIII, Issue VIII, August 2019 - An implementation of Colour Sorting Machine using Arduino Mega. The information gained are block diagram, circuit diagram and working of the project.
2. “Design and Development of an Automatic Colour Sorting Machine on Belt Conveyor”, Aung Thike, Zin Zin Moe San, Dr. Zaw Min Of, International Journal of Science and Engineering Applications Volume 8 – Issue 07, 176-179, 2019, ISSN:-2319-7560 - implementation using conveyer belt system. Information gained include working of colour sensor, servo motors and applications of this project.
3. “Programming Arduino Getting Started with Sketches, Book by Simon Monk - A book on Arduino programming in C using Arduino IDE. An essential resource while writing the code for project
4. Data Sheets of different sources and ICs