FEBRUARY 1, 2017

DESIGN PROPOSAL

PLASTIC BOTTLE SORTING MACHINE

AER201
PROF. M. REZA EMAMI
TA: WASI SYED

GROUP 48
MICHAEL KWOK, 1002699519
LING LONG, 1002516618
SHICHEN LU, 1002527839

DIVISION OF ENGINEERING SCIENCE
UNIVERSITY OF TORONTO

Table of Contents

1. Executive Summary

The purpose of this document is to outline a proposal for a design of an automated bottle sorting machine. The machine must be able to distinguish between 2 different types of bottles and determine whether or not the bottles have a cap on. Then, based on the previous properties, the machine must place each bottle in a specific sorting bin.

The proposed solution is a machine which will automatically queue and orient the bottles using a centrifuge and position the bottles to be accessed by an RGB color sensor. The color sensor will then distinguish the type of the bottle and the presence of a bottle cap. The bottle is then dispensed through a two-stage, paddle-based sorting mechanism into a sorting bin based on the properties of the bottle. The overall process is estimated to take about 5 seconds to sort each bottle.

The allowed budget of the project is $230 CAD, which will be provided by the group members.

The construction and design of the machine will be split into 3 main components between the three team members working on the proposal. Shichen Lu will develop the microcontroller component, which consists of all the software needed to run and automate the machine. Michael Kwok will develop the electromechanical component, which consists of constructing and optimizing the physical structure and mechanisms of the machine. Ling Long will develop the circuitry component, which consists of constructing the circuits and power sources needed to run all the sensors, motors, and other electrical components in the machine.

2. Problem Formation

2.1 Statement of Need

A recycling company is receiving bottles in a continuous, unsorted stream. These bottles must be sorted in accordance to their plastic composition (i.e. Made of transparent or opaque plastic, presence of a plastic bottle cap) so that the correct recycling process can be used on each bottle. Sorting bottles by hand is costly, slow, and prone to human error. Therefore, an efficient and automated sorting machine is desired. To meet the needs of the processing facility, the machine must be portable, able to process unsorted bottles loaded in 10 at a time in under 3 minutes, and able to distinguish between transparent/opaque bottles, and if there is a bottle cap present on the bottle. In addition, the final sorted bottles must be placed in separate, removable, and easily accessible containers.

2.2 Objectives

High Level Objective: Design a portable and automated bottle sorting machine that can sort plastic bottles into four separate categories, depending on the type of bottle and the presence of a bottle cap, in a timely manner.

Detailed Objectives: See table 1

Table 1: Detailed Objectives

Objective	Metric	Constraint	Criteria
Solution is portable	Dimensions (m) Weight (kg)	Dimensions: machine fits within a 0.55m x 0.55m x 0.55m box (excl. power cable) Weight: machine weighs less than 10kg (incl. power cable)	Dimensions: smaller is better Weight: lighter is better
Solution is affordable	Cost ($)	The total cost of the parts required to build the machine must not exceed $230 CAD (excl. labour costs)	Less is better
Solution can sort bottles in a timely manner	Time (min.)	The time from when the machine begins active operation to when the machine returns to a standby state must not exceed 3 minutes per 10 bottle load (excl. load/unload time)	Less is better
Solution allows for timely bottle loading and unloading	Time (min.) Degree of machine manipulation required	Total load time per 10 bottle load must not exceed 1 minute. Total unloading time per 10 bottle load must not exceed 0.5 min. During loading and unloading, no parts of the machine should need to be disassembled.	Less is better
Solution is autonomous	Degree of human interaction required for machine operation	In processing a 10-bottle load, the only external assistance the machine can receive is the loading of the bottles into the machine and the input of a “start” signal.	n/a

<table>
<tr><td>Solution does not significantly damage bottes</td><td>Degree of additional scratches/deformation on bottles after machine operation</td><td>Machine must not significantly damage the bottles (ie. Deformation or scratching) during its operation</td><td>Less is better</td></tr>
<tr><td>Solution correctly sorts bottles</td><td>Number of bottles sorted to the correct containers</td><td>n/a</td><td>More is better</td></tr>
<tr><td>Solution must have a UI that meets requirements</td><td colspan="3">The user interface must be:
1. Able to indicate the completion of the sorting process,
2. Able to display the total number of bottles processed,
3. Able to display the number of bottles in each sorted category,
4. Able to display the total time of operation, and
5. Robust and self-explanatory to the user
6. Contain an emergency stop button that immediately ceases all operations of the robot</td></tr>
</table>

3. SURVEY

For the background survey, we have separated the bottle sorting process of the robot into three major steps, as follows:

1. Loading: Processing of bottle input
2. Scanning: Determination of bottle type & presence of bottle cap
3. Sorting: Distributing bottles into correct sorting bins

3.1 REGARDING A MARKET SURVEY

There is a plethora of commercial companies (Anker Anderson, Krones AG, etc.) that sell large-scale, high-speed bottle sorting and bottle unscrambling machines. However, these designs offer functionality and speed that is not required by this proposal. As such, they are much too bulky and expensive for the purposes of this proposal, and these designs will not be considered.

3.2 LOADING: PROCESSING OF BOTTLE INPUT

The purpose of this step is to take the input of 10 bottles that are simultaneously dumped and process them into a single stream in which one bottle at a time is fed through the machine. This allows the bottles to be individually scanned and sorted later in the process.

3.2.1 BASIC HOPPER MECHANISMS

The premise of a hopper mechanism [1] involves a large bin for holding the bottles, with a small opening at the bottom that a single bottle is able to pass through at a time (see figure 1). Bottles are dumped into the bin, and by repeated agitation of the bottles, they will eventually tumble and fall through the opening at the bottom of the bin. The overall mechanism is quite simple. However, due to the limitations of this proposal, it may be difficult to sufficiently agitate the bin such that bottles do not get stuck and fail to fall through the opening. In addition, the rate the bottles will be processed is not consistent, due to the semi-random agitation process used to process the bottles.

Figure 1: Basic Hopper Mechanism

3.2.2 CENTRIFUGAL HOPPER MECHANISMS

The premise of a centrifugal mechanism involves a large spinning plate on which bottles are dumped (see figure 2). Due to the centrifugal force acting on the bottles, they are pushed towards the outer ring of the plate in a lateral orientation. There, a collecting mechanism can collect the bottles such that they are placed in a single stream. Compared to the hopper mechanism, this mechanism is bigger and more complex, but offers a much more consistent processing rate for bottles, mainly because the centrifugal force acts on all bottles simultaneously.

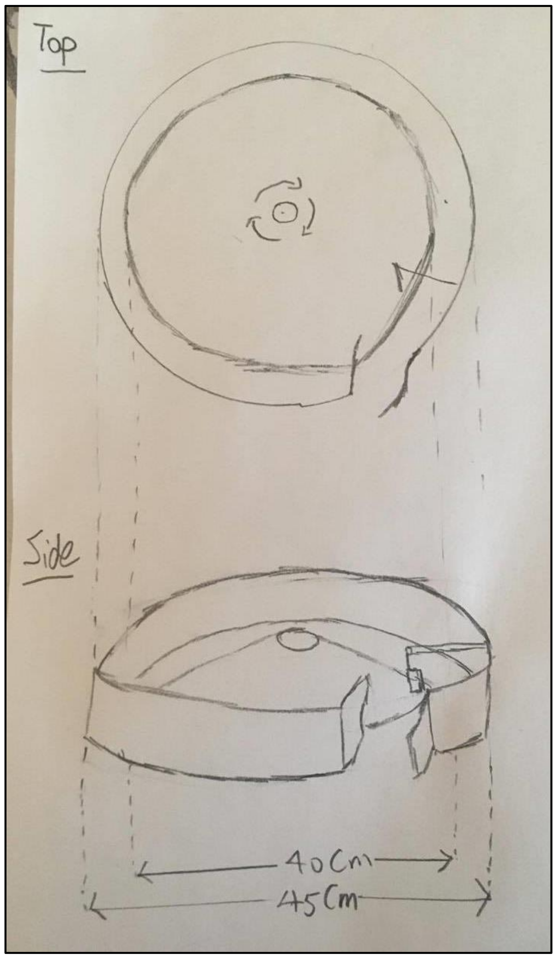


Figure 2: Centrifugal Mechanism

3.2.3 PIT-LIFTING HOPPER

The premise of a pit-lifting hopper is to have an elevator-esque mechanism to continuously lift bottles one by one from a bin filled with bottles (see figure 3). The "bottle elevator" then drops each bottle off at a platform such that they are sorted into a single stream of bottles. This mechanism often has issues with the bottles dropping out of the bottle elevator, and due to the nature of picking up one bottle at a time from the pit, this mechanism is relatively slow. However, unlike the previous two mechanisms, this one can process the bottles into a stream such that the bottles are oriented width-wise and roll as they travel.

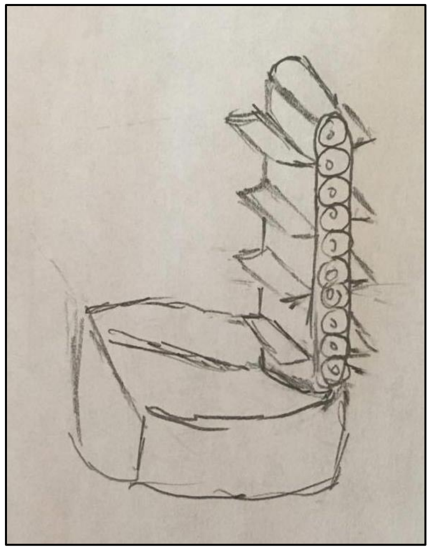


Figure 3: Pit-Lifting Hopper Mechanism

3.3 Scanning: Determination of Bottle Type & Presence of Bottle Cap

3.3.1 IR-Based Reflectivity Sensors

IR-based reflectivity sensors project infrared radiation forward, and use a photo-transistor to determine the intensity of the infrared radiation that has hit an object and bounced back towards the sensor. Because infrared radiation somewhat passes through translucent material, the sensor can be aimed at the body of the bottle to determine if the bottle has a transparent or opaque body. This allows us to determine the difference between the Eska and Yop bottles. However, the range on IR-based reflectivity sensors is generally short, and the output signal is subject to frequent fluctuations and noise. Therefore, some noise reduction method is needed to process the output signal.

A chose a TCRT5000L [2] IR Reflectivity Sensor to test the effectiveness of this type of sensor. We setup a test circuit supplying 9v to the reflectivity sensor and measured the change in its voltage out put when pointed at certain objects 1.5cm away from the tip of the sensor (see figure 4). The results are shown in the following table (table 2):

Table 2: TCRT5000L Sensor Voltage Output Changes for Various Detected Objects

Object to be Detected	Sensor Voltage Output Change
Side of Yop Bottle	-7.9V
Side of Eska Bottle	-0.7V
Side of Eska Bottle with Label	-3.5V
Top of Yop Bottle with Cap	-7.8V
Top of Yop Bottle without Cap	-2.1V
Top of Eska Bottle with Cap	-4.0V
Top of Eska Bottle without Cap	-0.5V

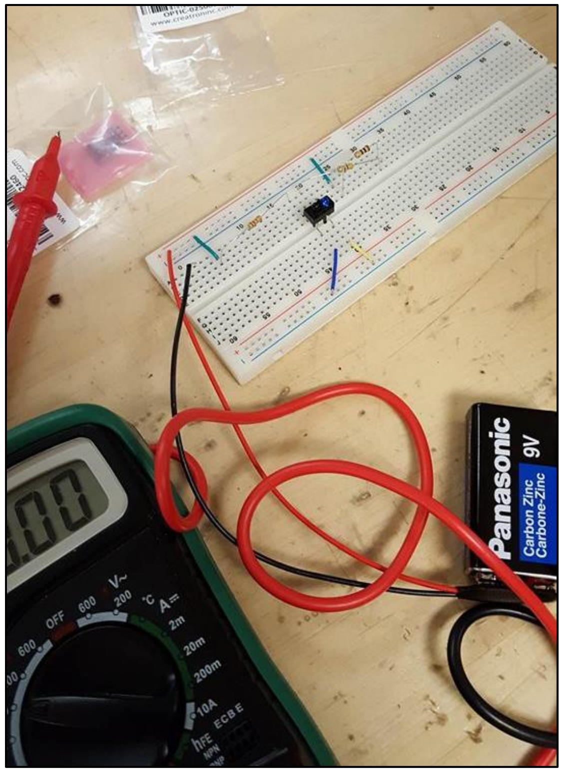


Figure 4: Testing Setup for Reflectivity Sensor

3.3.2 Color Sensors

Color sensors project a (usually) white light forward, and use a combination of RGB filters and a photodiode to convert the received light into an RGB signal. Because the Eska bottles are transparent, while the Yop bottles are either blue, red or yellow, the color sensor can distinguish between the different bottle types. And because the bottle cap of the Yop is red, while the bottle cap of the Eska is blue, the sensor can distinguish between bottles with and without caps. The color sensor requires minimal background light to function at its best, and in the presence of overly intense background light, the color sensor fails to function at all. The noise in the output signal depends on the intensity of the background light.

A TCS34725 [3] Color Sensor was chosen to test the effectiveness of these types of sensors. We setup a test circuit using an Arduino nano to read and operate the color sensor (see figure 5). The sensor was aimed the tip of the bottle from a sideways angle and a distance of 1.5cm. Note that this color sensor in particular outputted red, green, blue, and clear values. Due to signal noise, the outputted values vary in about a 100 units range around the value given in the table (table 3).

Table 3: TCS34725 Output RGB and Clear Values for Various Objects

Object	Red Value	Green Value	Blue Value	Clear Value
Yop with Cap	3000	1200	1100	5000
Yop without Cap	16000	15000	12000	50000
Eska with Cap	900	980	1080	2500
Eska without Cap	3800	4100	3600	11000

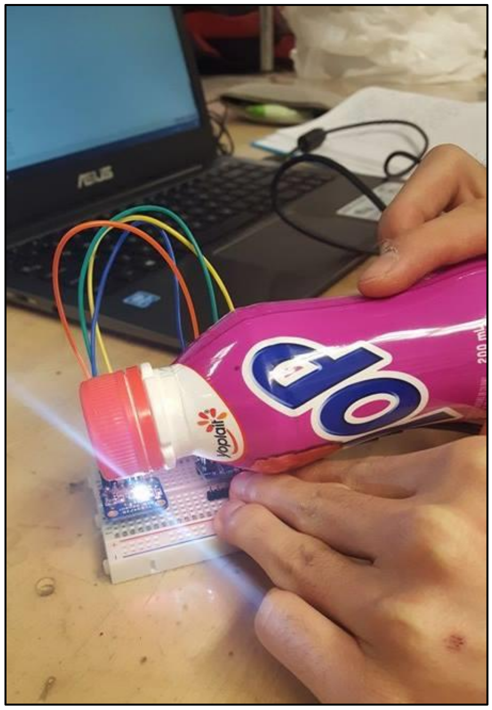


Figure 5: Testing Setup for Color Sensor

3.3.3 Weight Sensors

Weight sensors output a signal depending on the weight of the object being weighed. There is a weight difference between the Eska and the Yop bottles that is significant enough (~2g) to be detected by certain accurate weight sensors. However, weight sensors often require the object to be stationary over the sensor for a short amount of time before being able to produce a steady signal. In addition, the presence of the bottle cap also changes the weight of the bottle, such that the Yop bottle with cap may be mistaken for an Eska bottle without cap.

Because the presence of the bottle cap would make certain configurations of the Yop/Eska bottles indistinguishable from others by weight, we neglected to test the effectiveness of this sensor.

3.3.4 Acoustic Sensors

Acoustic sensors output a signal depending on the frequency of that it detects. The idea behind the utilization of this sensor is to mechanically blow a stream of air across the top of each bottle. This would produce a different sound signature depending on if the bottles have a cap or not. The problem with this method is in obtaining a consistent and distinct sound signature for each bottle. Due to the conceptual impracticality of this design, this sensor was not tested.

3.4 Sorting: Distributing bottles into correct sorting bins

3.4.1 Paddle-Based Sorter

This sorting mechanism has two sets of paddles. The first paddle will send a bottle coming into the sorter into either the right or the left column of the sorter, after which another paddle in each of the columns will again separate incoming bottles into one of two slots (see figures 6 and 7). In total, this sorter will be able to put a bottle into one of four slots. This sorter has a high sorting speed that depends on how fast the paddles can move from one position to the other, however this sorter takes a large amount of physical space, and requires careful timing, as the bottles move through the sorter without stopping. In addition, the circuitry implementation for this sorter is fairly complex due to the amount of paddles and the bi-directional motion they require.

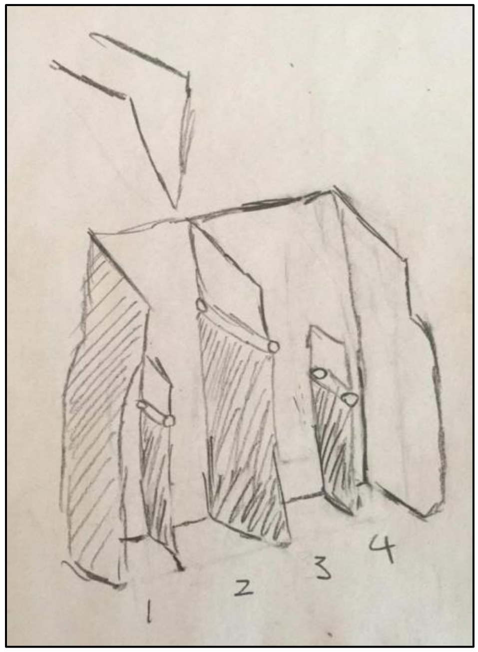


Figure 6: Paddle-Based Sorter

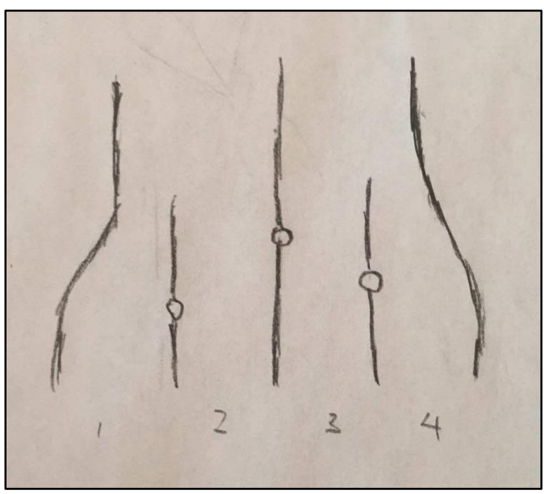


Figure 7: Paddle Setup for Paddle-Based Sorter

3.4.2 Rotating Sorter

This sorting mechanism contains a bottle stopper and a rotating tube [4]. The bottle enters the sorter and is stopped by the stopper. The rotating mechanism can then rotate the tube to point towards a certain sorting box (see figure 8). Afterwards the stopper releases the bottle and the bottle falls through the tube into the sorting box that the tube points to. This sorter can sort items into several different boxes, depending on the arrangement of the sorting boxes around the sorter. The stopping mechanism in this sorter also allows for more leniency in the timing and configuration of the sorter in its operation. However, this sorter has a relatively slow speed that is determined by how fast the rotating mechanism can rotate the tube to point to the correct sorting box.

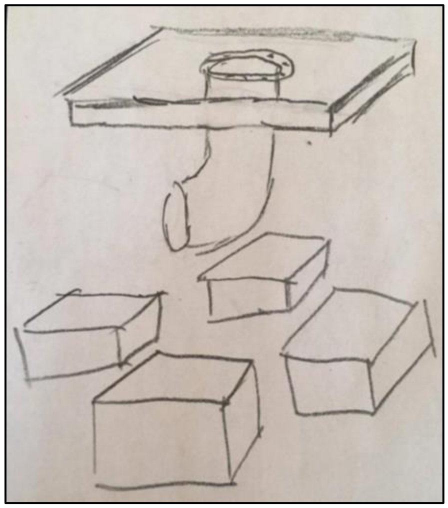


Figure 8: Rotating Sorter

3.4.3 Ramp-Based Sorter

This sorting mechanism employs a ramp that directs where a bottle can go. By adjusting the rotation of the ramp using a motor, a bottle flowing into the sorting mechanism can be directed into multiple different sorting bins (see figure 9). Its simplicity allows for a very easy to implement solution. However, compared to the rotating sorter, the range of motion smaller. Additionally, this solution limits the speed of the bottle processing by the speed of the motor turning, and this is especially slow when the motor needs to change the direction of the ramp from one end to the opposite end.

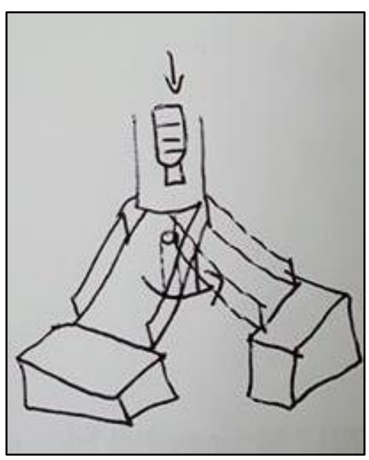


Figure 9: Ramp-Based Sorter

4.CONCEPTUALISATION

4.1 FUNCTIONAL DECOMPOSITION

In order to conceptualize a final design, it was necessary to decompose the goal of the machine into separate components which perform a specific task necessary to achieve the ultimate goal. These separate components should not significantly impact the workings of the other components of the machine, and should be easily integrated with the other components to make one combined machine. In regards to the task of sorting bottles, the following components were decided upon, slightly different from the surveyed components:

1. A mechanism/area to load the 10 bottles in before the start of operation
2. A mechanism to properly feed/orient the bottles through the machine to be processed
3. A mechanism to detect whether the bottle is Eska or Yop
4. A mechanism to detect whether the bottle has a cap or not
5. A mechanism to feed the bottle into the correct bin after being processed by sensors

These components, integrated with some sort of mechanical design, were decided to be necessary/sufficient for a complete design. Consideration of each component was necessary to progress in the design proposal.

4.2 MECHANISM TO LOAD BOTTLES INTO THE MACHINE

A mechanism to load the bottles is necessary and important as a design consideration because the structure of the loading mechanism determines the structure of the overall machine. To be an effective loading mechanism, it was determined that not only does the mechanism should be compact, but also be aiding the feeding process. By prototyping a centrifuge mechanism (see figure 10), we were able to see that this loading mechanism was both compact and output bottles with the correct orientation at a steady pace, suitable for further processing.

Figure 10: Prototype Centrifugal Bottle Hopper

4.3 Mechanism to Feed and Orient the Bottles

It was decided that rather than sorting the bottles all at once, it is necessary to sort the bottles one at a time to accurately process each bottle and guide them to their respective bins. As such, it is determined that the most effective way is through streamlining the entire process, meaning that there will be a continuous queue of bottles waiting to be processed.

4.4 Mechanism to Detect Whether the Bottle is Yop or Eska

This mechanism is necessary in the project because it is one of the primary objectives of the proposal. In considering the mechanism to distinguish the bottles, it is necessary to consider the distinguishing features of the Eska and Yop bottles.

The main features distinguishing the Yop and Eska Bottles are listed below (see table 4):

Table 4: Distinguishing Features of Eska and Yop Bottles

Feature	**Yop**	**Eska**
Weight, with Cap	15g	14g
Weight, without Cap	14g	13g
Shape	Cylindrical	Cylindrical
Diameter	5.0cm	5.8cm
Height	14.6cm	15.6cm
Color(bottle)	White	Transparent
Transparency	Opaque	Transparent
Material	High-Density Polyethylene	Polyethylene Terephthalate
Color(label)	Red	Blue

As a conceptual idea, each feature was considering as a possibility in our design. The following metrics were used to decide whether a feature should be used in the design to distinguish between the bottles; these are further detailed in table 5:

1. Cost of equipment needed to distinguish the feature
2. Effectiveness and range of equipment regarding the specifications of our bottles and project
3. Ease of implementation of feature into mechanical design (constraints on size and shape fitting into actual machine design)
4. Simplicity of design (complexity of implementation)
5. Reliability of design (ease of replacement and risk of failure)

Table 5: Conceptualization Table for Bottle Type Detection

Distinguishing Feature/Solution	Cost	Effectiveness	Ease of Implementation	Simplicity	Reliability
Weight/Pressure Sensor	Cheap, ~$10 for load sensor	Relatively ineffective, sensitivity of sensor vs difference in weight low	Not easy, need to make a mechanical method for bottle to be weighed	Relatively simple circuit implementation	Reliable

Shape/Camera & Software Setup	Varying	Relatively ineffective, as the shapes are similar	Not easy, especially on software side	Difficult	indeterminate
Transparency/ Reflectivity sensor	Cheap, <$5 for an IR transmitter /receiver	Relatively effective, as	Relatively easy	Simple	Difficult to determine
Color/Color Sensor	Medium cost for our purposes (~$10)	Relatively effective considering the significant color difference	Relatively easy	Very simple	Reliable as determined by testing

4.5 Cap Detecting Mechanism

In order to detect whether a bottle has a cap or not, the distinguishing features between a bottle with a cap and no cap must be considered:

1. Size (height of bottle and width of cap area)
2. Color in area of cap
3. Weight
4. Shape (having no cap creates an opening in a bottle)

Again, the following metrics were used to decide whether a feature should be used in the design to distinguish between the bottles (see table 6):

1. Cost of equipment needed to distinguish the feature
2. Effectiveness and range of equipment regarding the specifications of our bottles and project
3. Ease of implementation of feature into mechanical design (constraints on size and shape fitting into actual machine design)
4. Simplicity of design (complexity of implementation)
5. Reliability of design (ease of replacement and risk of failure)

Table 6: Conceptualization Table for Cap Detection Mechanism

Solution Mechanism	Cost	Effectiveness	Ease of Implementation	Simplicity	Reliability
Pressure Sensor	Cheap, $10 for load sensors	Relatively ineffective, sensitivity of sensor vs difference in weight low	Not easy, need to make a mechanical method for bottle to be weighed	Relatively simple circuit implementation	Reliable

Shape detection	Varying depending on implementation	Relatively ineffective, as the shapes are similar	Not easy, especially on software side	Not simple	Difficult to determine
Reflectivity sensor	Cheap, <$5 for an IR transmitter/ receiver	Relatively effective, as the sensors have a decent range and the caps	Need to orient the bottle to be positioned facing towards sensors	Relatively okay	Difficult to determine
Color Sensor	Low cost (~$10)	Very effective	Relatively easy	Simple	Reliable
Acoustic Sensor	Medium Cost (~$20)	Unknown	Difficult	Not simple	Difficult to determine

4.6 AHP to Choose the Cap and Bottle Type Detection Method

By using the "Analytical Hierarchy Process [9]", the Cap/Bottle distinguishing mechanisms were compared and weighted against each other, giving the following result (see table 7):

Table 7: Weighted AHP Score for Proposed Solutions

Proposed Solution	**Weighted AHP score**
Pressure Sensor	0.157
Color Sensor	0.262
Acoustic Sensor	0.113
Reflectivity Sensor	0.219
Image Processing(Camera)	0.180
Shape Sensing	0.123

Refer to Appendix B for calculations and Matrices of AHP scores.

Here we choose the color sensor, as it has the highest weighted AHP score, and is able to determine both the bottle type and the presence of a bottle cap.

4.7 Mechanisms to Sort the Bottles After Being Processed with Sensors

In order to choose a mechanism to sort the bottle, several criteria were chosen in order to evaluate the viability of our possible solutions.

Criteria:

1. Speed
2. Effectiveness
3. Ease of Implementation
4. Cost
5. Reliability

From surveying, the possible designs for the sorting mechanisms were narrowed down to the following designs:

1. Paddle-Based Sorter
2. Rotating Tube Sorter
3. Ramp Sorting Mechanism

For a clearer description of these mechanisms, refer to section 3.4.

4.8 Utility Based Decision For Bottle Sorting Mechanisms

To more clearly analyse our criteria for the sorting mechanism, a Utility Based Requirements Model is presented with even more criterion in the following table (see table 8 and figure 11):

Table 8: Utility Based Decision Making Table

Design Aspect	Metric	Constraints	Utility
Speed	Time it takes for the bottles to be sorted once they reach the mechanism(s)	Must Sort At Least As Fast as 10 bottles in 3 Minutes	**Faster is Better** The faster the bottles are sorted, the more successful the design is.
Effectiveness	Percentage of Bottles Sorted Correctly(%)	Must Sort at least 4 bottles correctly out of 10	**More Correct is Better** More bottles sorted is a direct indicator of the success of the design.
Ease of Implementation	Time needed for Design team to Build/Implement(days)	Must be built within the Timescale of the project(2 months)	**Faster is Better** The faster a design is to implement, the more time can be allocated to other factors in the project
Cost	Cost of implementation($)	Whole Project must not exceed $230 CAD	**Cheaper is Better** The cheaper the exit mechanism, the more money may be spent on improving the rest of the design.
Reliability	Percentage of Failure(%)	Must fail at a rate less than 100%	**Less is Better** Clearly, not failing is an ideal trait for a component to have.

And Utility functions are used to weigh the various solutions against there criterion.

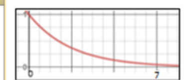
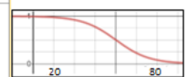
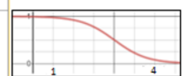
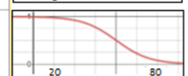
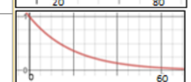
Utility Function Model Comparing Sorting Mechanisms									
			Paddle-Based Sorter		Rotating Sorter		Moving Ramp Sorter		
	Objectives	WEIGHT	Estimated Value	Utility Value	Estimated Value	Utility Value	Estimated Value	Utility Value	Utility Function
1.	Speed of Sorting Mechanism	5	2	0.65	5	0.32	3	0.31	
2.	Effectiveness of Sorting Mechanism	3	80	0.21	60	0.45	50	0.57	
3.	Ease of Implementation	1	2	0.76	3	0.51	3	0.51	
4.	Cost of implementation	2	20	0.87	30	0.80	40	0.69	
5.	Reliability	3	80	0.21	40	0.42	50	0.60	
.	UTILITY WEIGHTED SUM		1.58	1.58	1.32	1.32	1.11	1.11	
	NORMALIZED TOTAL		0.39	0.39	0.33	0.33	0.2775	0.2775	

Figure 11: Utility Function Model

Based on the Utility Function Model, we can observe that the Paddle-based Sorter is Effective with respect to these criteria compared to our other designs.

5. Specification

5.1 Final Design Overview and Schematics

The final design (see figure 12) was designed as an amalgamation of the best design choices for each separate function of the machine. The bottles are loaded into the machine by dumping them into a cylindrical hopper, implemented as a centrifuge. The cylinder has a small opening where the bottles move through once the centrifuge is turned on. The bottles pass through one at a time. After leaving the centrifuge, the bottles are guided via a guiding pole mechanism and propelled through gravity. As the bottles move along the guiderails, a combination of a tripwire and a color sensor detects the type of bottle and the presence of a cap. Once the bottle reaches the end of the guiderail, it enters a paddle-based sorting system where it is directed to a certain sorting bin based on the properties of the bottle. Each component and the process is discussed in much further detail in this section.

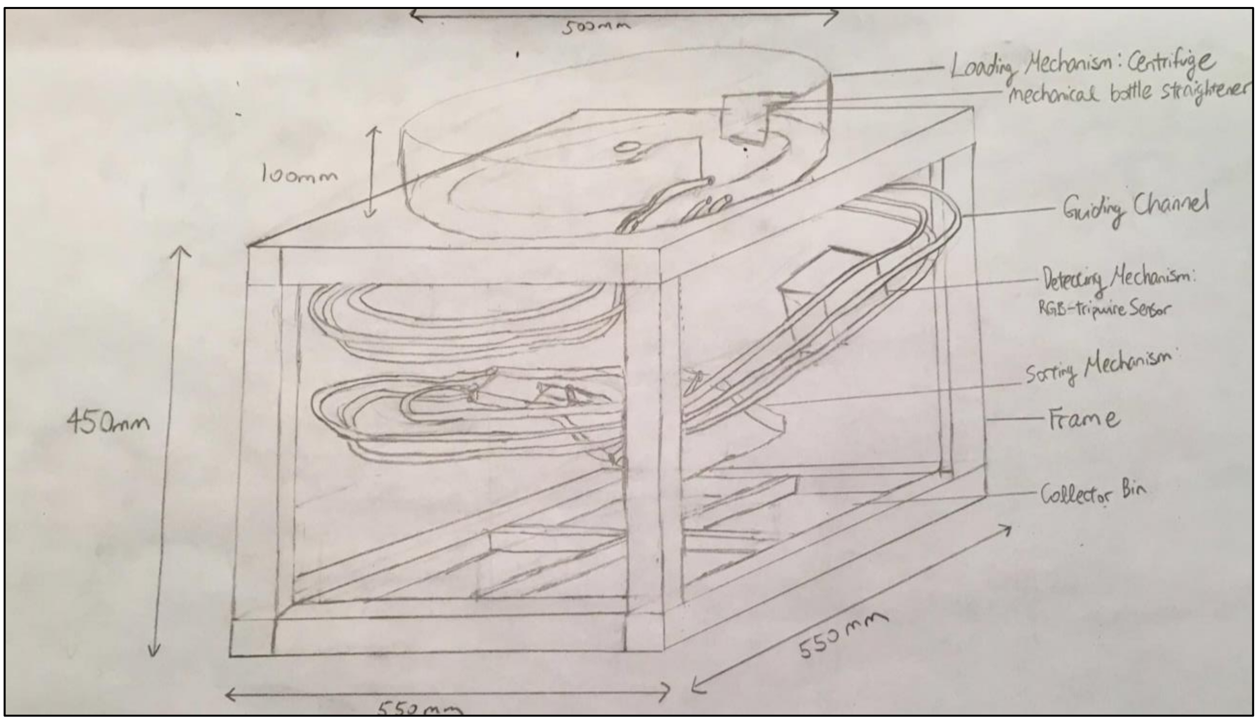


Figure 12: Final Design Sketch

5.2 Loading Mechanism

During conceptualization, many different loading mechanisms were considered. However, it was determined that the best method of unloading and orienting the bottles is through a centrifuge (see figure 13). This mechanism requires minimum effort from the operator, who only needs to dump the batch of bottles into the centrifuge, and also provides a continuous stream of bottles out of the centrifuge. The motor chosen for the centrifuge is the M1N10FB11G [6], a brushed DC motor. This motor was chosen because it has low relative torque, but a very fast rotation speed, which is important for our centrifuge design that has a relatively low moment of inertia due to its lightweight material selection and the light weight of the bottles it will hold. Additionally, it is cheap and easy to implement using a microcontroller.

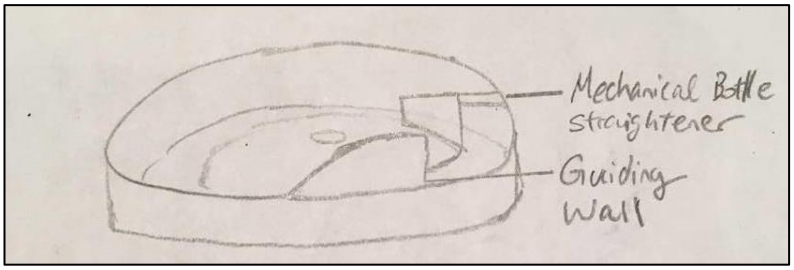


Figure 13: Loading Mechanism Sketch

5.2.1 Centrifuge

As shown in figure 14, the centrifuge mechanism during loading is powered by a DC motor, which is attached to a shaft that directly connects to the inner aluminum dome of the centrifuge. When operating, the shaft and the entire inner aluminum dome will spin along with the motor. This circular motion creates a centrifugal force that projects outwards. In this case, the bottles are loaded on top of the dome, and when the operation begins, the spinning motion will push all the bottles to the side and thus lining all the bottles along the perimeter.

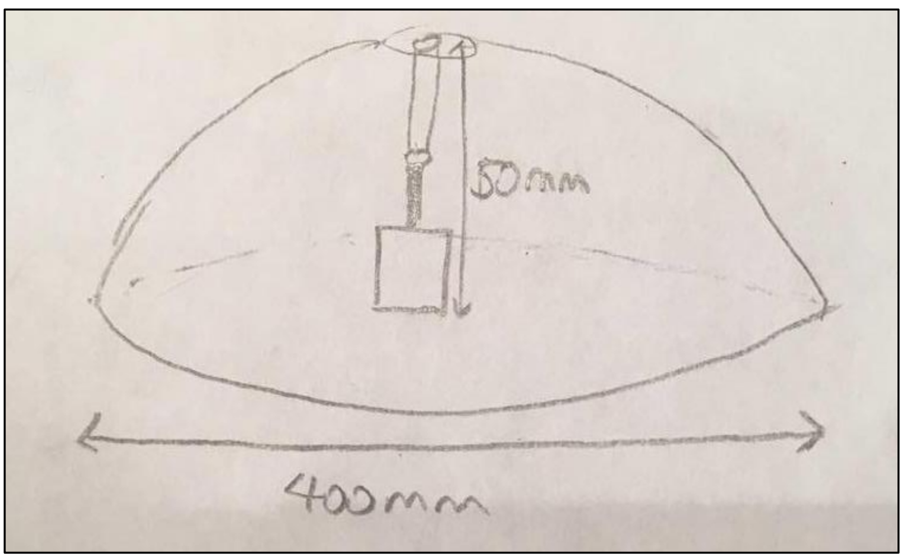


Figure 14: Loading Mechanism Sketch (Bottom)

5.2.2 Guiding Wall

The guiding channel implemented in the centrifuge (see figure 13) provides a smooth transition of the bottle from the centrifuge into the guiderails. This is crucial, as any bumps could potentially jam the machine. Thus, by introducing such feature into the loading mechanism, it greatly reduces the chances that bottles are jammed.

5.2.3 Mechanical bottle straightener

To ensure that the bottles are coming out of the centrifuge in a streamlined fashion and in the horizontal orientation (length-wise), a metal plate is installed, as shown in figure 15. This mechanism will push back any bottles that are perpendicular to the side and only allows those that are parallel through. Furthermore, the metal plate has a slanted bottom side to accommodate the rotating centrifuge.

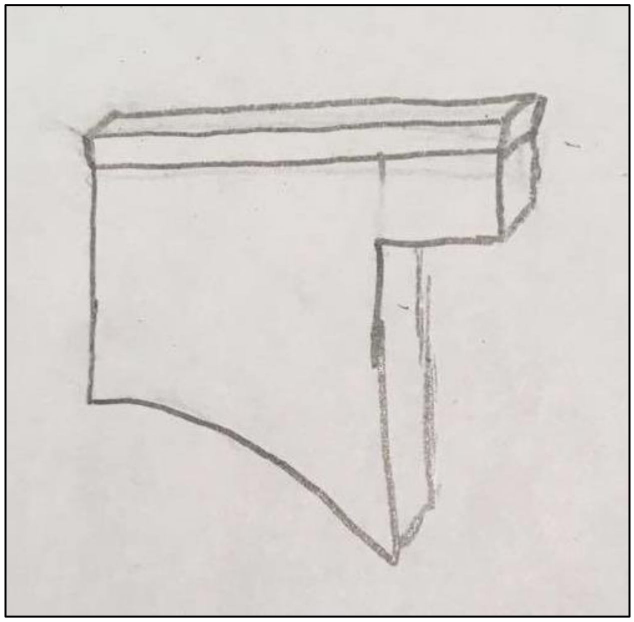


Figure 15: Mechanical Bottle Straightening Mechanism Sketch

5.3 Guiding Rails

The guiderails (see figure 16) form the curved path that bottles will travel on from the exit of the centrifuge to the start of the sorting mechanism. Compared to our other designs, such as the v-shaped ramp, this method of guiding the bottles is relatively simple to design and construct, especially considering the curvature along the frame. In addition, the open space allows for easy attachment of the sensor mechanism along the guardrails.

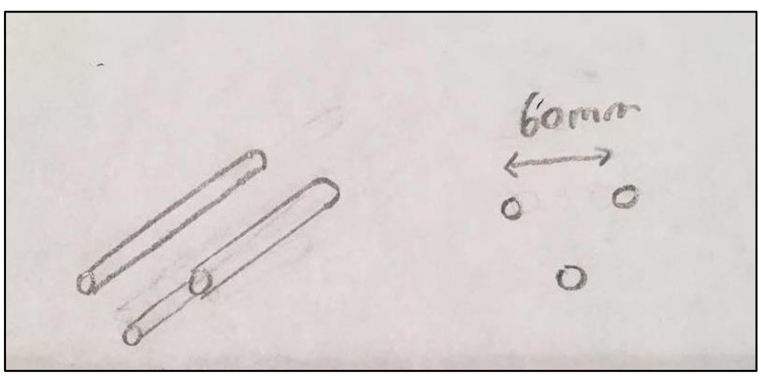


Figure 16: Guiding Rail Specifications Sketch

5.4 Mechanism to distinguish bottle type

During conceptualisation, functionality of sorting was split into two different categories: one mechanism to sort a bottle based on whether it has a cap or not, and another mechanism to distinguish between Yop and Eska. However, the design team recognised that the color sensor alone could preform both of these functions.

Therefore, the final mechanism chosen was the usage of an RGB color sensor, model TCS34725. It takes advantage of the fact that the bottles to be sorted can be divided into the four different categories solely by the color at the cap area of the bottle:

- Yop with Cap: Red
- Yop with no Cap: White
- Eska with Cap: Blue
- Eska without Cap: Transparent

In order to implement the sensor, an attachment bracket will be used to attach the sensors to the guard rails such that the sensor is aimed at the middle of the bottle widthwise. The color sensor will be implemented alongside a tripwire sensor. By reading the tripwire and color sensors, we are able to determine both the type of bottle and the presence of a bottle cap. The specific algorithm used to determine the bottle properties is discussed in further detail in section 5.7 of this proposal.

The TCS34725 sensor was chosen because of its cheap cost ($10) and effectiveness. During testing, it was determined to have sufficient range and sensitivity to be positioned in the guardrails on our machine, and the sensor was able to effectively distinguish between the cap-area colors of the bottles, including if the area was transparent. The tripwire sensor will be custom made with an infrared light sensor, and a laser diode.

5.5 Sorting Mechanism

The sorting mechanism was chosen to be a ramp that separates the bottles in 2 different stages. The bottle enters the sorting mechanism via gravity, where it flows through a channel controlled by a gate. Based on the position of the gate, the bottle is guided in one of two different directions depending on whether the bottle is Yop or Eska. This channel and gate mechanism occurs again in the second stage, where the bottles are separated by whether they have a cap or not.

The gates are mini- “flaps” that are rotated via servo motors. There is a tripwire at the beginning of the sorting mechanism to detect the window of time for which the gates can change, to account for the next bottle coming to the sorting mechanism. An example run of the exit mechanism is outlined in figure 17.

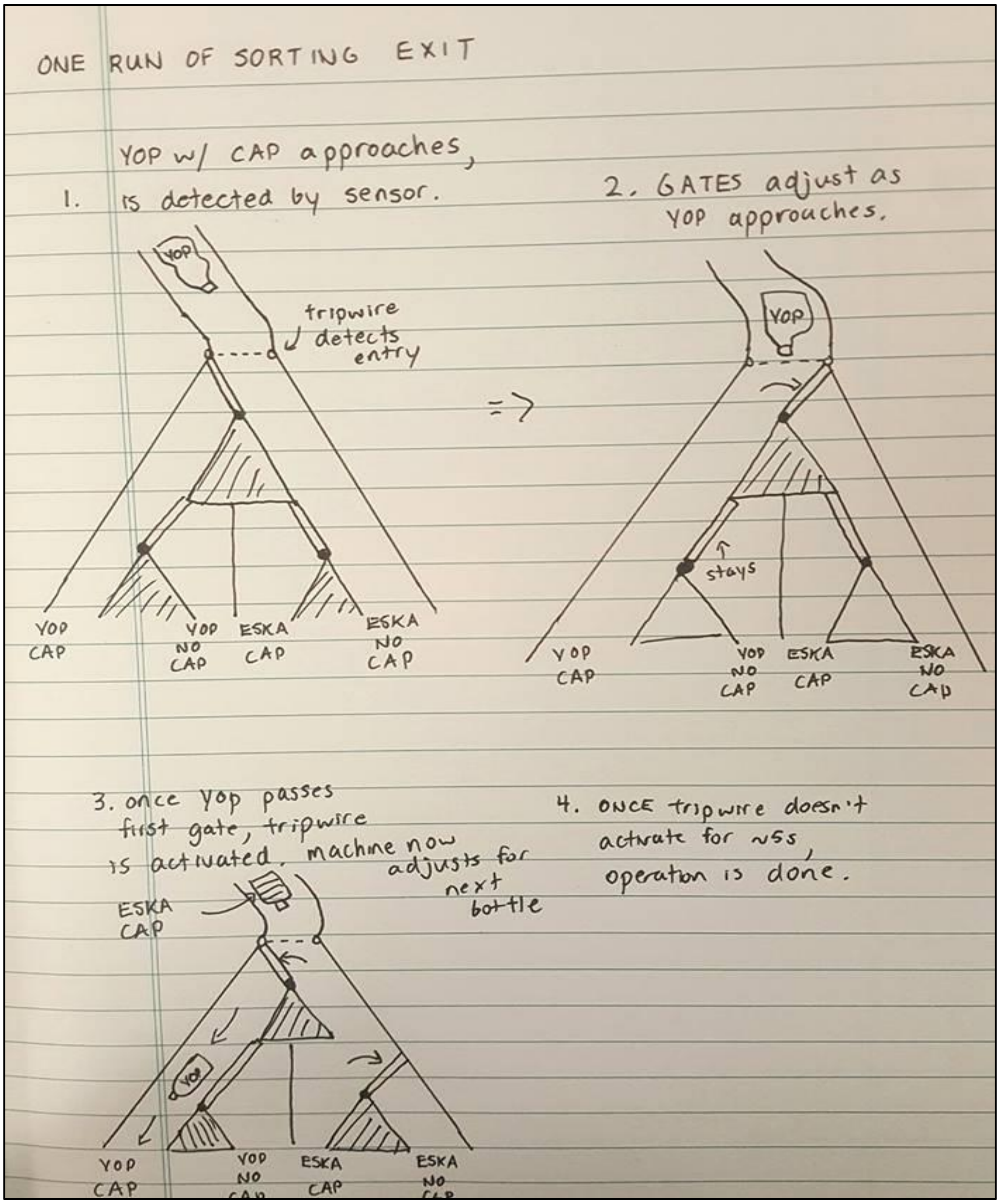


Figure 17: Sorting Mechanism Functional Depiction

The gates will be implemented using FS-90 mini Servo Motors. These motors were chosen because they are relatively cheap (~$5 each) and they are easy to use and implement with a microcontroller. Additionally, the specifications show that they provide the sufficient range of motion and speed to adjust the gate quickly enough for the continuous stream of bottles.

5.6 Materials Selection

5.6.1 Frame

After considering numerous options, mainly polystyrene, wood, steel and aluminum, it is determined that the material that best fits the weight and cost constraints is wood. Because the frame must support the overall structure of the robot, the main material criteria to evaluate here it's the strength of the material. According to the material tables from engineering tool box in Appendix E, although aluminum has the highest tensile strength out of all the materials and a relatively low density when compared to other metals, wood is strong enough for our purposes, making it a lighter and cheaper alternative. Furthermore, constructing using wood is much simpler than construction using metal materials.

5.6.2 Centrifuge

The centrifuge which is constructed on top of the main frame and is to be composed of both wood and aluminum. The base of the centrifuge will be wood, while the curved moving ramp in the middle and the sides will be constructed out of aluminum. The reasoning behind the difference in the materials is to prevent aluminum from constantly scratching against itself, which would cause galling and significantly accelerate the wear of the material. Furthermore, since due to the weight criteria, aluminum and wood were chosen as the best options to create a structure that is lightweight and stable. This choice will also lessen the power required by the motor to spin the centrifuge, as the overall weight of the components decreases.

5.6.3 Guiding Rail

The guiding rail is to be made from aluminum rods, as it they are very malleable and easy to work with. In addition, they have a relatively light weight to meet the constraints laid out in the objectives. The choice of aluminum over wood is mainly due to the lack of malleability of wood. Furthermore, when considering even more malleable metals such as brass and bronze, we note that their price is significantly higher than that of the aluminum. Thus, aluminum remains the optimal choice.

5.6.4 Sorting Flaps

Because the main functionality of the sorting flaps is to redirect bottles to the correct path, the strength of the material can be overlooked. However, because a servo motor is driving this flap, the moment of inertia cannot be too great such that the torque of the motor is unable to move the flap at a sufficiently fast speed. As such, it is wise to choose a material that has a lower density. With this in mind, plywood sheets are then the best material to choose, as they have a density of 0.6g/cm^3, which is significantly less than that of the aluminum, which is around 2.70 g/cm^3[7]. Furthermore, the flaps do not require any additional processing aside from cutting out the desired shape, and thus does not have to be malleable.

5.6.5 Collection Bin

Since the collection bin is a part of the entire design but does not have any other function aside from storing the sorted bottles, it is determined that the material used will be HDPE, as it is light, compact, and cheap. The combination of its light weight and its low cost makes it the most viable option, preventing the bins from contributing to the overall cost or the weight by a significant amount. This allows us to allocate more budget to more important parts.

5.6.6 Main Body of Sorting Mechanism

Because the frame of the sorting mechanism requires a lot of shaping, aluminum sheets are the most appropriate materials to use. Unlike wood, aluminum sheets are malleable and can easily be folded into different geometries [7].

5.7 PIC Microcontroller

5.7.1 *Microcontroller Selection*

The microcontroller selected was the PIC18F4620 [8]. The main reason for choosing this microcontroller is because it was provided. Additionally, this is a midrange PIC microcontroller that was able to provide ample processing speed, memory capacity, and number of pins to meet the specifications of our project. Also, this series of microcontroller is able to interface with the C programming language, which is our preferred development language.

5.7.2 *Algorithm for Determination of Bottle Categorization*

The setup of our solution positions a color sensor alongside a tripwire in a position perpendicular to the motion of the bottle. Thus, the color sensor is able to sense the color of the cap-area of the bottle from the side. However, the orientation of the bottle as it passes the color sensor is unknown (ie. We do not know if the top of the bottle or the bottom of the bottle passes by the sensor first). To solve this, we will take and process a reading from the color sensor each time the tripwire switches from on to off or off to on (these points correspond to when the bottle starts passing by the sensor and when the bottle has fully passed the sensor). This way, both the color of the top and the bottom of the bottle is recorded.

We will separate the major colors that the color sensor will be expected to detect into the following categories:

- Transparent
- Red
- Blue
- White
- Other

Now, based on the two color readings recorded per bottle (bottle top color and bottle bottom color) we can determine the bottle type:

- “Red” and “other” readings: Yop with cap
- “White” and “other” readings: Yop without cap
- “Blue” and “transparent” readings: Eska with cap
- “Transparent” and “transparent” readings: Eska without cap

5.7.3 *Extendibility: Label Detection*

As specified by the extendibility section of the RFP, it is desirable, but not necessary, for the designed machine to detect the presence of a label on the bottle. We accomplish this by taking a third color sensor reading 0.1 seconds after the tripwire is first tripped. This reading will read the color in the middle of the bottle, allowing us to determine the presence of a label on the bottle as follows, assuming that the type of the bottle has already been determined as in section 5.7.2

- Yop and “other” reading: Yop with label
- Yop and “white” reading: Yop without label
- Eska and “other” reading: Eska with label
- Eska and “transparent” reading: Eska without label

5.7.3 Main Program Execution

After being initialized, the microcontroller will run in the program loop shown in figure 18. While the microcontroller is in the standby state, the user will also be able to access the UI displays of number of bottles processed, amount of bottles in each category, current date and time, and total processing time. The user accesses this information using the keypad keys, and the information is displayed on the LCD. The subroutines mentioned in the main program flowchart are detailed in the pseudocode.

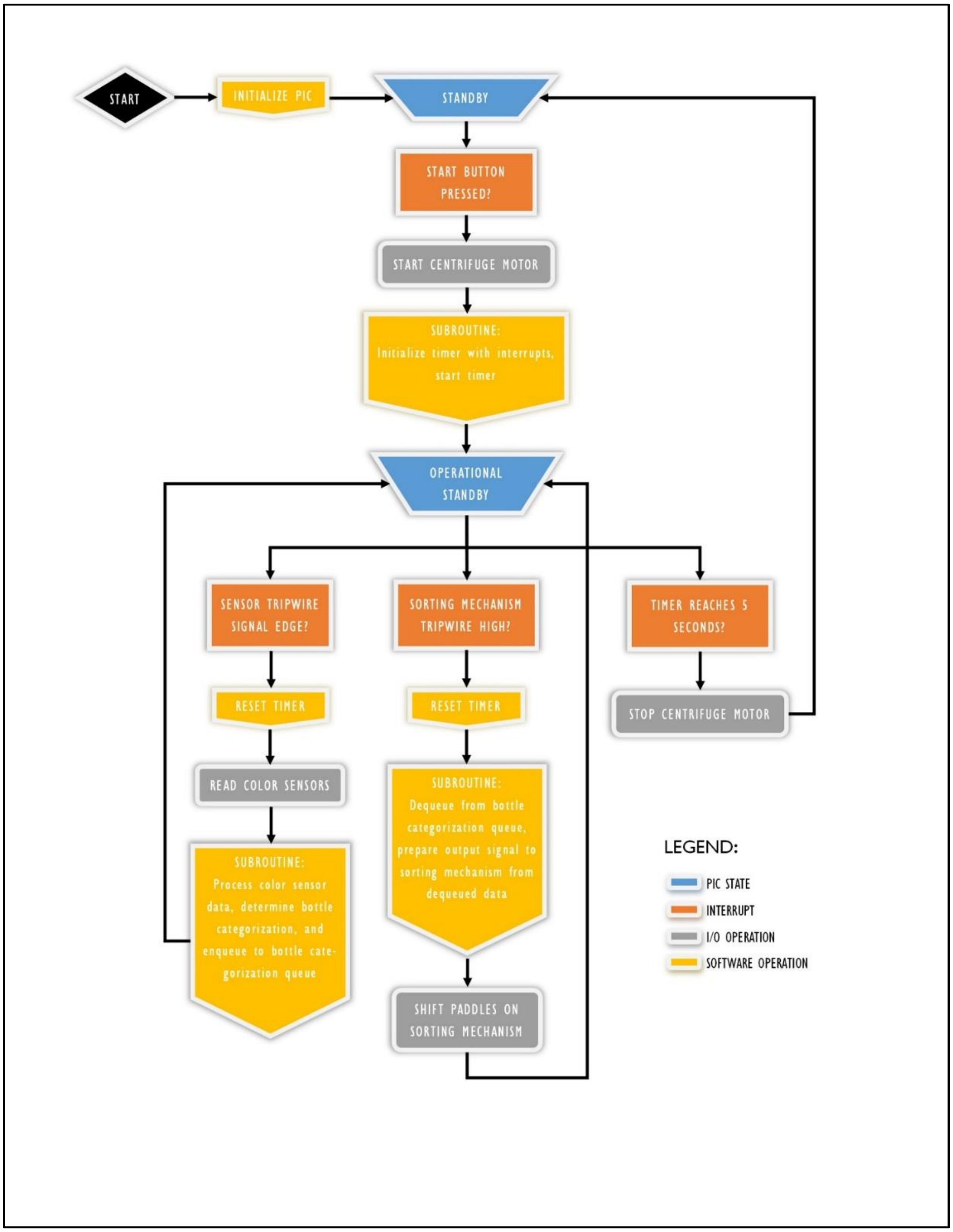


Figure 18: Main Program Flowchart

5.7.4 PSEUDOCODE

The following figures (figures 19 to 24) show the pseudocode implementation of the main file for the microcontroller setup.

```
FUNCTION main
    Initialize ports
    Initialize timer1 with interrupt when time is 5 seconds
    Initialize timer2
    Initialize PWM
    Initialize LCD
    Enable keypad interrupts

    Initialize queue bottle_type_queue          //queue to store determined bottle type
    Initialize integer variable bottlecount_disp    //variable to determine what bottle counts to display
    Initialize boolean color_sensor_flag        //0 if next color_sensor_tripwire signal edge signifies the start of a bottle
                                                //1 if next color_sensor_tripwire signal edge signifies the end of a bottle
    Initialize integer array color_sensor_data[3]   //stores three instances of color sensor data to determine categorization of a bottle
                                                //first index: color at bottle head
                                                //second index: color at bottle tail
                                                //third index: color at bottle middle
                                                //0: transparent
                                                //1: red
                                                //2: blue
                                                //3: white
                                                //4: other
    Initialize integer array bottlecount[5]     //bottlecount[0] = total bottles
                                                //bottlecount[1] = YOP with cap, no label
                                                //bottlecount[2] = YOP without cap, no label
                                                //bottlecount[3] = YOP with cap, yes label
                                                //bottlecount[4] = YOP without cap, yes label
                                                //bottlecount[5] = Eska with cap, no label
                                                //bottlecount[6] = Eska without cap, no label
                                                //bottlecount[7] = Eska with cap, yes label
                                                //bottlecount[8] = Eska without cap, yes label

    WHILE true
        Go to home in LCD
        PRINT "standby, "
        Go to new line in LCD
        PRINT "press A to begin"
    END WHILE
END FUNCTION

FUNCTION interrupt
    CASE interruptID OF
        keypad_A: CALL bottle_processing
        keypad_B: CALL bottle_count
        keypad_C: CALL bottle_time
        keypad_D: CALL date_time
        keypad_*: CALL emergency_stop
        tripwire_color_sensor: CALL sensor_tripped
        tripwire_sorting: CALL sorter_tripped
        timer1: CALL operation_stop
    END CASE
END FUNCTION
```

Figure 19: Pseudocode Page 1

```
FUNCTION bottle_processing
    Enable tripwire_color_sensor to interrupt on positive and negative edges
    Enable tripwire_sorting to interrupt on positive edges
    Start timer1
    Start timer2
    Start centrifuge motor

    WHILE true
        PRINT "operation in progress"
    END WHILE
END FUNCTION

FUNCTION bottle_count
    CASE bottlecount_disp OF
        0:  Disable keypad interrupts
            Go to home in LCD
            PRINT "Bottle Counts"
            Go to new line in LCD
            PRINT "Total: " bottlecount[0]
            Enable keypad interrupts
            SET bottlecount_disp TO 1
        1:  Disable keypad interrupts
            Go to home in LCD
            PRINT "YOP+CAP: " bottlecount[1]
            Go to new line in LCD
            PRINT "NAKED YOP: " bottlecount[2]
            Enable keypad interrupts
            SET bottlecount_disp TO 2
        2:  Disable keypad interrupts
            Go to home in LCD
            PRINT "YOP+CAP+LBL: " bottlecount[3]
            Go to new line in LCD
            PRINT "YOP+LBL: " bottlecount[4]
            Enable keypad interrupts
            SET bottlecount_disp TO 3
        3:  Disable keypad interrupts
            Go to home in LCD
            PRINT "ESKA+CAP: " bottlecount[5]
            Go to new line in LCD
            PRINT "NAKED ESKA: " bottlecount[6]
            Enable keypad interrupts
            SET bottlecount_disp TO 4
        4:  Disable keypad interrupts
            Go to home in LCD
            PRINT "ESKA+CAP+LBL: " bottlecount[7]
            Go to new line in LCD
            PRINT "ESKA+LBL: " bottlecount[8]
            Enable keypad interrupts
            SET bottlecount_disp TO 0
    END CASE
END FUNCTION
```

Figure 20: Pseudocode Page 2

```
FUNCTION bottle_time
    Disable keypad interrupts
    Go to home in LCD
    PRINT "OPERATION TIME"
    Go to new line in LCD
    PRINT timer2 in seconds
    Enable keypad interrupts
END FUNCTION

FUNCTION date_time
    Initialize structure time           //time.sec stores seconds as integer
                                        //time.min stores minute as integer
                                        //time.hour stores hour (24-hour format) as integer
                                        //time.day stores day as integer
                                        //time.month stores month as integer
                                        //time.year stores year as integer
    WHILE true
        READ current time from RTC
        SET time TO current time from RTC
        Go to home in LCD
        PRINT "DATE: " time.year "/" time.month "/" time.day
        Go to new line in LCD
        PRINT "TIME: " time.hour ":" time.min ":" time.sec
    END WHILE
END FUNCTION

FUNCTION emergency_stop
    Stop centrifuge motor
    Stop timer1
    Stop timer2
    Disable all interrupts
    PRINT "emergency stop"
END FUNCTION
```

Figure 21: Pseudocode Page 3

```
FUNCTION sensor_tripped
    CASE color_sensor_flag OF
        0:  READ color_sensor_signal                                    //color_sensor_signal is the output of the color sensor
            CALL interpret_color_sensor WITH color_sensor_signal        //it consists of four values: Red, Green, Blue, Clear
            SET color_sensor_data[0] TO output of interpret_color_sensor    //they will be referenced as color_sensor_signal.R, color_sensor_signal.G,
            Wait 0.1 seconds                                            //color_sensor_signal.B, and color_sensor_signal.C, respectively
            READ color_sensor_signal
            CALL interpret_color_sensor WITH color_sensor_signal
            SET color_sensor_data[3] TO output of interpret_color_sensor
            SET color_sensor_flag TO 1
        1:  READ color_sensor_signal
            CALL interpret_color_sensor WITH color_sensor_signal
            SET color_sensor_data[1] TO output of interpret_color_sensor
            Enqueue color_sensor_data to bottle_type_queue
            SET color_sensor_flag TO 0
END FUNCTION

FUNCTION sorter_tripped
    Reset timer1
    Initialize integer array curr_bottle_data[3]           //temporarily stores color sensor data of next bottle to be sorted
    Initialize int bottle_type                             //temporarily stores the bottle categorization of next bottle to be sorted
    Dequeue from bottle_type_queue into curr_bottle_data   //bottle_type:  1: YOP with cap, no label
    CALL bottle_type_handler WITH curr_bottle_data         //              2: YOP without cap, no label
    SET bottle_type TO return of bottle_type_handler       //              3: YOP with cap, yes label
    CALL sorter_motor_handler WITH bottle_type             //              4: YOP without cap, yes label
                                                           //              5: Eska with cap, no label
                                                           //              6: Eska without cap, no label
                                                           //              7: Eska with cap, yes label
                                                           //              8: Eska without cap, yes label
END FUNCTION

FUNCTION operation_stop
    Stop centrifuge motor
    Stop timer1
    Stop timer2
    Disable tripwire interrupts
    WHILE true
        PRINT "operation completed"
    END WHILE
END FUNCTION

FUNCTION interpret_color_sensor
    IF color_sensor_signal.R is around 3.8k AND color_sensor_signal.G is around 4.1k AND color_sensor_signal.B is around 3.6k AND color_sensor_signal.C is around 11k
        RETURN 0
    ELSE IF color_sensor_signal.R is around 3.0k AND color_sensor_signal.G is around 1.2k AND color_sensor_signal.B is around 1.1k AND color_sensor_signal.C is around 5k
        RETURN 1
    ELSE IF color_sensor_signal.R is around 900 AND color_sensor_signal.G is around 980 AND color_sensor_signal.B is around 1.1k AND color_sensor_signal.C is around 25k
        RETURN 2
    ELSE IF color_sensor_signal.R is around 16k AND color_sensor_signal.G is around 15k AND color_sensor_signal.B is around 12k AND color_sensor_signal.C is around 50k
        RETURN 3
    ELSE
        RETURN 4
    END IF
END FUNCTION
```

Figure 22: Pseudocode Page 4


```
FUNCTION operation_stop
    Stop centrifuge motor
    Stop timer1
    Stop timer2
    Disable tripwire interrupts
    WHILE true
        PRINT "operation completed"
    END WHILE
END FUNCTION

FUNCTION interpret_color_sensor
    IF color_sensor_signal.R is around 3.8k AND color_sensor_signal.G is around 4.1k AND color_sensor_signal.B is around 3.6k AND color_sensor_signal.C is around 11k
        RETURN 0
    ELSE IF color_sensor_signal.R is around 3.0k AND color_sensor_signal.G is around 1.2k AND color_sensor_signal.B is around 1.1k AND color_sensor_signal.C is around 5k
        RETURN 1
    ELSE IF color_sensor_signal.R is around 900 AND color_sensor_signal.G is around 980 AND color_sensor_signal.B is around 1.1k AND color_sensor_signal.C is around 25k
        RETURN 2
    ELSE IF color_sensor_signal.R is around 16k AND color_sensor_signal.G is around 15k AND color_sensor_signal.B is around 12k AND color_sensor_signal.C is around 50k
        RETURN 3
    ELSE
        RETURN 4
    END IF
END FUNCTION
```


Figure 23: Pseudocode Page 5

```
FUNCTION bottle_type_handler
    IF curr_bottle_data[0] = 1 or 4 AND curr_bottle_data[1] = 1 or 4 AND curr_bottle_data[2] = 3
        Increment bottle_count[0] by 1
        Increment bottle_count[1] by 1
        RETURN 1
    ELSE IF curr_bottle_data[0] = 3 or 4 AND curr_bottle_data[1] = 3 or 4 AND curr_bottle_data[2] = 3
        Increment bottle_count[0] by 1
        Increment bottle_count[2] by 1
        RETURN 2
    ELSE IF curr_bottle_data[0] = 1 or 4 AND curr_bottle_data[1] = 1 or 4 AND curr_bottle_data[2] = 4
        Increment bottle_count[0] by 1
        Increment bottle_count[3] by 1
        RETURN 3
    ELSE IF curr_bottle_data[0] = 3 or 4 AND curr_bottle_data[1] = 3 or 4 AND curr_bottle_data[2] = 4
        Increment bottle_count[0] by 1
        Increment bottle_count[4] by 1
        RETURN 4
    ELSE IF curr_bottle_data[0] = 0 or 2 AND curr_bottle_data[1] = 0 or 2 AND curr_bottle_data[2] = 0
        Increment bottle_count[0] by 1
        Increment bottle_count[5] by 1
        RETURN 5
    ELSE IF curr_bottle_data[0] = 0 AND curr_bottle_data[1] = 0 AND curr_bottle_data[2] = 0
        Increment bottle_count[0] by 1
        Increment bottle_count[6] by 1
        RETURN 6
    ELSE IF curr_bottle_data[0] = 0 or 2 AND curr_bottle_data[1] = 0 or 2 AND curr_bottle_data[2] = 4
        Increment bottle_count[0] by 1
        Increment bottle_count[7] by 1
        RETURN 7
    ELSE IF curr_bottle_data[0] = 0 AND curr_bottle_data[1] = 0 AND curr_bottle_data[2] = 4
        Increment bottle_count[0] by 1
        Increment bottle_count[8] by 1
        RETURN 8
END FUNCTION

FUNCTION sorter_motor_handler
    CASE bottle_type OF
        1: Set sorting paddles to direct bottle into YOP with cap bucket
        2: Set sorting paddles to direct bottle into YOP without cap bucket
        3: Set sorting paddles to direct bottle into Eska with cap bucket
        4: Set sorting paddles to direct bottle into Eska without cap bucket
    END CASE
END FUNCTION
```

Figure 24: Pseudocode Page 6

5.7.4 Pin Assignments

The following table (table 9) provides the pin assignments for the microcontroller.

<u>Table 9: Pin Assignments for PIC Microcontroller</u>

Pin Name	*I/0*	*Analog/Digital*	*Description of Assignment*
RA0:1	I	A	Receives Signal from Color sensor
RA2	O	D	Generates Signal to Power Loading Motor
RA3:4	I	A	Receives Signal from 2 Tripwires
RD0:5	I	D	Interface with Keypad
RB0:6	O	D	Generates Signal to Control Servo Motor Gates
RC0:7 RE0:2	-	-	Interface with LCD display
RA5	I	D	Receives Emergency Stop Pushbutton Signal

5.8 Calculations

5.8.1 Weight

Top (Loading Mechanism: Centrifuge + resting surface)

$$W_{Resting\ Surface} = (0.55m)^2 \times 0.00635m \times 500\frac{kg}{m^3} = 0.96kg$$

$$W_{Cylinder} = \pi \times 0.1m \times 0.5m \times 0.00205m \times 2700\frac{kg}{m^3} = 0.87kg$$

$$W_{Centrifuge} = \pi \times 0.05m \times (0.25m^2 - (0.25m - 0.00205)^2) \times 2700\frac{kg}{m^3} = 0.43kg$$

Frame

$$W_{frame} = 0.0254m \times 0.0508m \times 0.2032m \times 2700\frac{kg}{m^3} \times 2.5 = 0.328\ kg$$

Bins

$$W_{Bins} = 0.08\ kg \times 4 = 0.32kg$$

Railings

$$W_{railings} = \pi \times (\frac{0.003175m}{2})^2 \times 3.65m \times 2700\frac{kg}{m^3} \times 4 = 0.312kg$$

Sorting Mechanism

$$W_{sorting\ max} = 0.4m \times 0.002052m \times 0.3566m \times 2700\frac{kg}{m^3} = 0.79kg$$

Housing for circuitry and Controller

$$W_{housing} = (0.15m)^2 \times 0.002052m \times 3 \times 2700\frac{kg}{m^3} = 0.373kg$$

$$\boldsymbol{Total\ Weight\ of\ Robot}$$
$$= 0.96kg + 0.87kg + 0.43kg + 0.328kg + 0.32kg + 0.312kg + 0.79kg + 0.373kg$$
$$= 4.38kg$$

5.8.2 Angle of Decline of guiding rail

$$F_{net} = F_g - F_f = ma$$

Since we want a constant velocity downwards, along the guiding rail, therefore the acceleration must be 0.

$$F_g = F_f$$

$$mgsin\theta = \mu mgcos\theta$$

Since the coefficient of friction between the rail and the bottle is 0.15 and that there are 3 point of contacts between the bottle and the rail, therefore the equation becomes

$$sin\theta = 0.45cos\theta$$

$$\theta = 24°$$

5.8.3 Characteristics of DC motor for Centrifuge

$$Mass\ of\ Centrifuge + all\ the\ possible\ Load = 0.43kg + 0.015kg \times 10\ bottles = 0.58kg$$

$$F_{centripetal} = F_f$$

$$m\frac{v^2}{r} = \mu mg$$

$$v = \sqrt{r\mu g} = \sqrt{0{,}20m \times 0.4 \times 9.8\frac{m}{s^2}} = 0.88m/s$$

$$\omega = \frac{v}{r} = 4.4\,rad/s$$

For the centrifuge moment of inertia,

$$I = \int\int \rho^2 dA = \int\int \rho^2 r d\theta dz = 4.3 \times 10^{-3} kgm^2$$

For Bottles,

$$I = mr^2 = 0.15kg \times (0.2m)^2 = 6 \times 10^{-3} kgm^2$$

$$\sum I = 1.3 \times 10^{-2} kg\,m^2$$

Angular Momentum,

$$L = I \times \omega = 1.3 \times 10^{-2} kg\,m^2 \times 4.4\,rad/s = 0.0572 kgm^2/s$$

Assuming achieving this angular momentum in 3 seconds,

$$\tau = \frac{dL}{dt} = \frac{0.0572kgm^2/s}{3s} = 0.019kgm^2/s^2$$

Thus, the DC motor we choose must have a torque greater than $0.019kgm^2/s^2$.

5.8.4 Characteristics of Servo Motor

Moment of the motor,

$$I = \frac{m}{12}(4h^2 + w^2) = \frac{0.1kg}{12}(4 * 0.1^2 + 0.05^2) = 3.54 \times 10^{-4} kgm^2$$

Because we want the motor to have the speed to switch quickly, thus, assume that $\omega\ is\ 6$,

$$L = I \times \omega = 3.54 \times 10^{-4} kg\,m^2 \times 6\,rad/s = 2.125 \times 10^{-3} kgm^2/s$$

The motor have to be able to activate straight away so assume time = 0.3s

$$\tau = \frac{dL}{dt} = \frac{2.125 \times \frac{10^{-3} kgm^2}{s}}{0.3s} = 7.08{\times}10^{-3} kgm^2/s^2$$

Thus, the DC motor we choose must have a torque greater than $0.019 kgm^2/s^2$.

5.9 Basic Circuit Schematics

5.9.1 Motor for Spinning Loading Centrifuge

The motor is a brushless DC motor intended to spin when the machine is operational. This is a simple circuit to build as it simply requires power to be supplied to the motor when the machine is turned on.

5.9.2 Laser Tripwire

The circuit for the laser tripwire consists of a photo resistor and a laser diode, which will allow the machine to detect when a trip wire is broken. An example schematic is shown below (see figure 25) of a very simple, possible circuit. Many are possible.

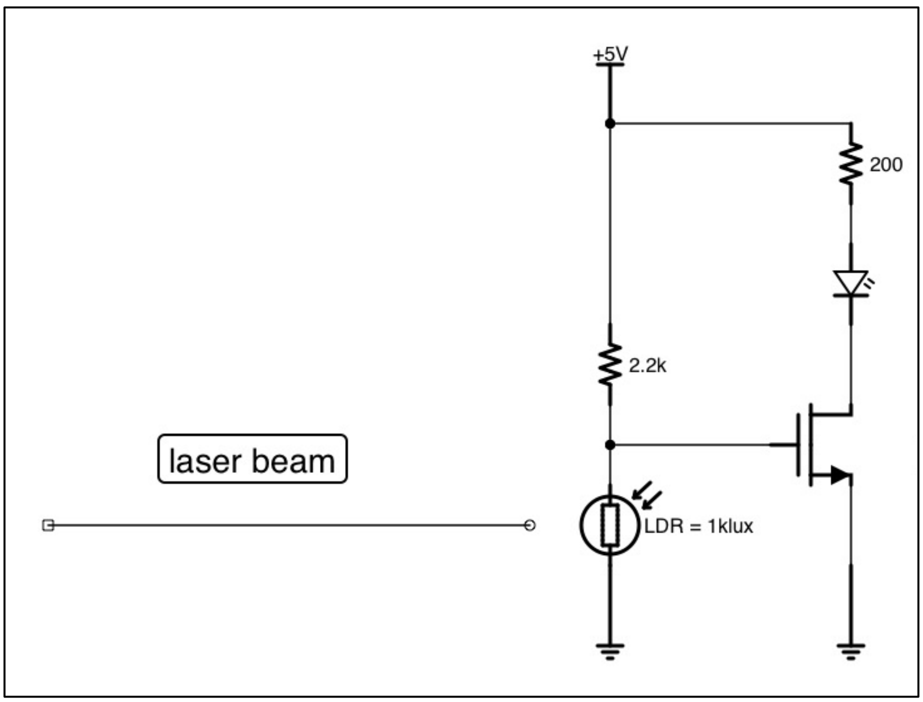


Figure 25: Laser Tripwire Circuit Diagram

5.9.3 Color Sensor

The TCS34725 is a colour sensor that requires 4 pin connections to the microcontroller, and is run at 5V with low (<1mA) current. The schematic for the color sensor is given below (figure 26).

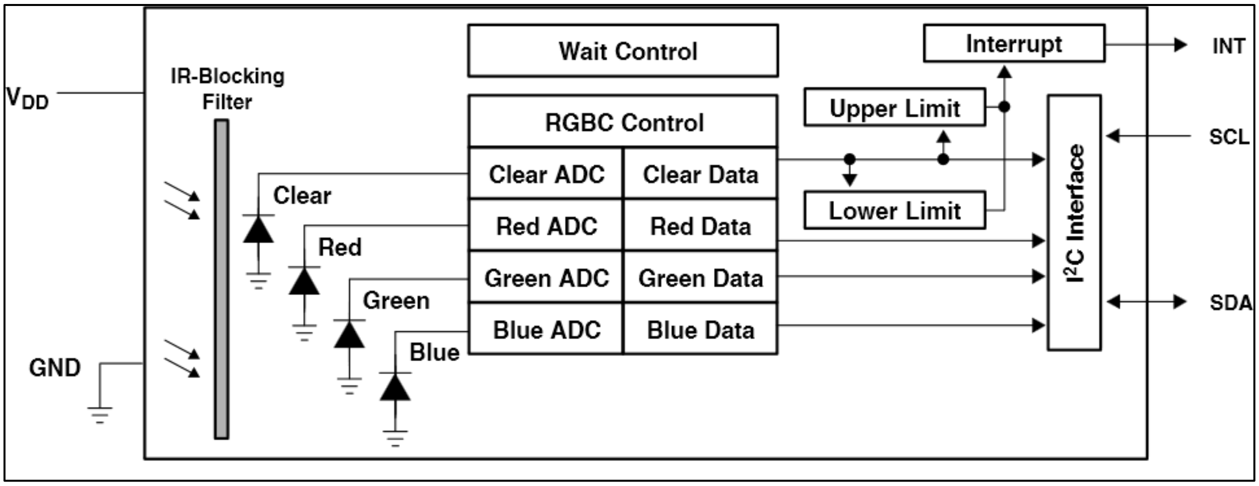


Figure 26: Color Sensor Circuit Schematic

5.9.4 Servo Motor Control

The project requires the control of 3 separate servo motors, which each run at 5V and ~200mA peak Current. A sample schematic is presented with a PIC board to a servo motor, using a 7805 Voltage Regulator (figure 27). The servo motors chosen are FS90

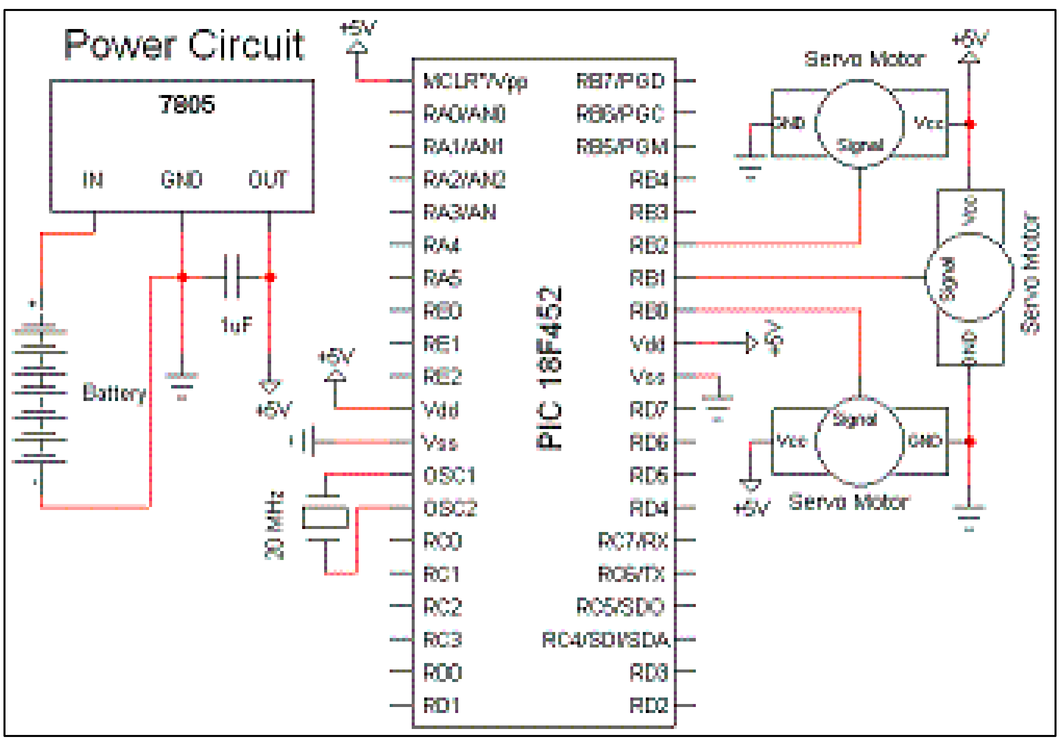


Figure 27: Servo Motor Control Circuit Schematics

6 Project Management

6.1 Overview

Aside from specifying the design components of the project, this section of the proposal is paramount to describe the actual work process to turn the design into its proposed fruition. Based on the scheduling and allocated time for this project, it is decided that the project is split into 3 main time periods.

1. Project Introduction and Conceptualization

This period constitutes the main design aspect of the project- all the necessary skills are taught and learned in this period of time for the project to move forward in its design. The project proposal is written with group members collaborating to choose a final design. Concepts and ideas are prototyped using simple mechanisms to verify the validity of a final concept. This period ends with the completion of the project Proposal.

2. Subsystem Construction and Fabrication

After the Project proposal is completed, each group member must work individually on their own subsystem, relatively apart from the other group members. The goal of this time is to construct a subsystem that is fully functioning so that during the later weeks of the project, the subsystems may be combined with little to no issues with a combined machine. This part of the project requires group members to be highly independent and efficient.

3. Project Integration and Finalization

After each group member has completed their respective subsystem, it is necessary to collaborate to combine the subsystems into one functioning machine. It is expected that combining the subsystems will lead to numerous bugs, which constitutes this period of time as a "debugging" period of time. This period of time requires the team members to work together to debug the machine and finalise aspects of the project. The project concludes with the presentation and the completion of a project report.

6.2 Statement of Work

Due to the nature of the project, it is necessary to split the work into work that is done individually as subsystem members of a group, and collaborative work to integrate the subsystems, etc. This does not restrict group members from helping each other on each portion of the project, as long as each subsystem is completed on time. Regardless of whether work is individual or not, is it necessary to establish a timeline for which work is done, as well as the tasks that each subsystem member must complete.

6.2.1 GANTT Chart for Project Planning

A GANTT [13] Chart is presented below (figure 28) to outline the events of this project, and the tasks and progress that are to be expected over the semester.

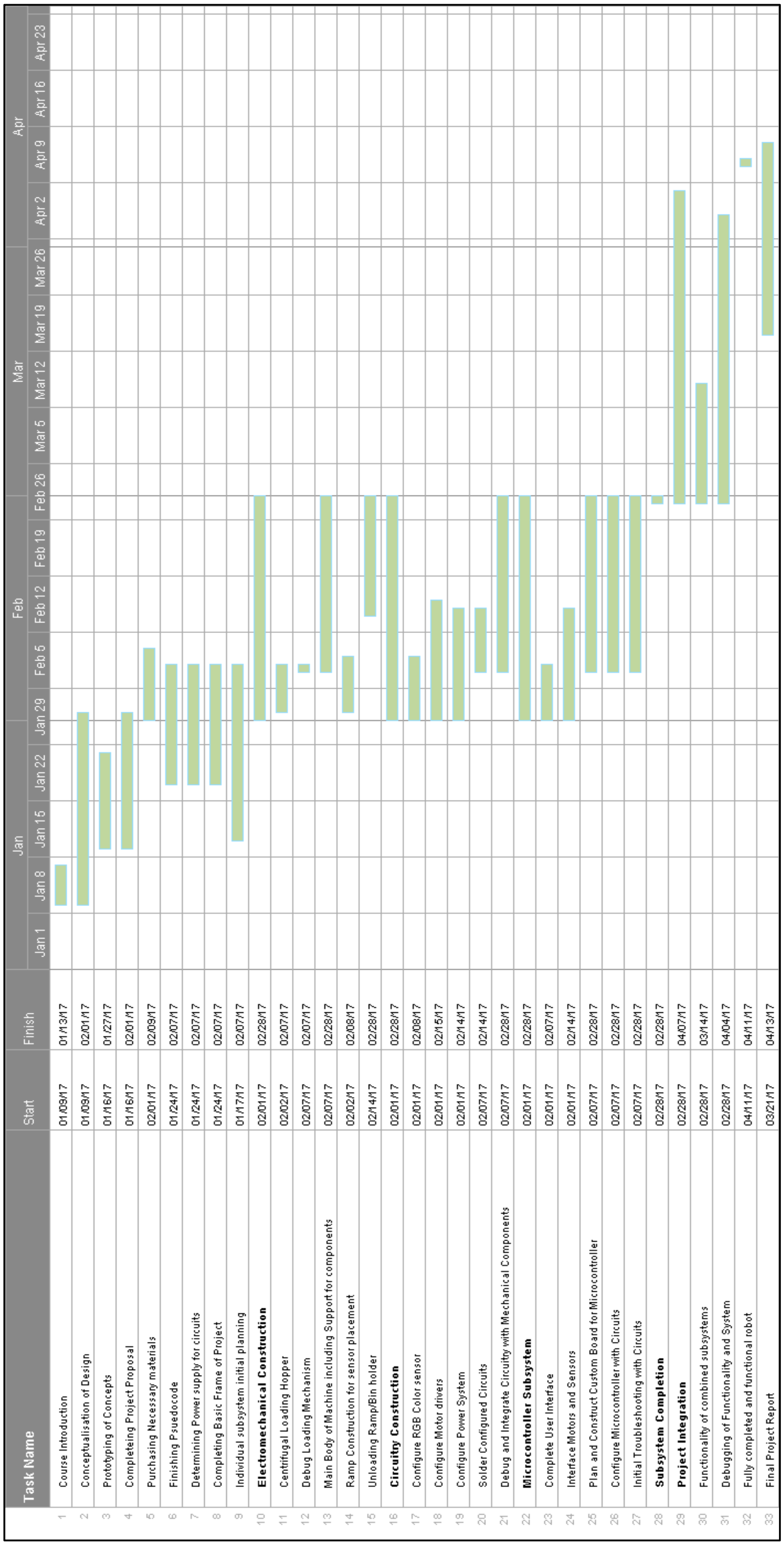


Figure 28: GANTT Chart

6.2.2 PERT chart for project planning.

The following section employs PERT [14] to evaluate the length and viability of the project. The chart below (figure 29) shows a diagram of how the events lead to each other, and the 2 tables below (tables 10 and 11) show calculations and analysis of the project planning.

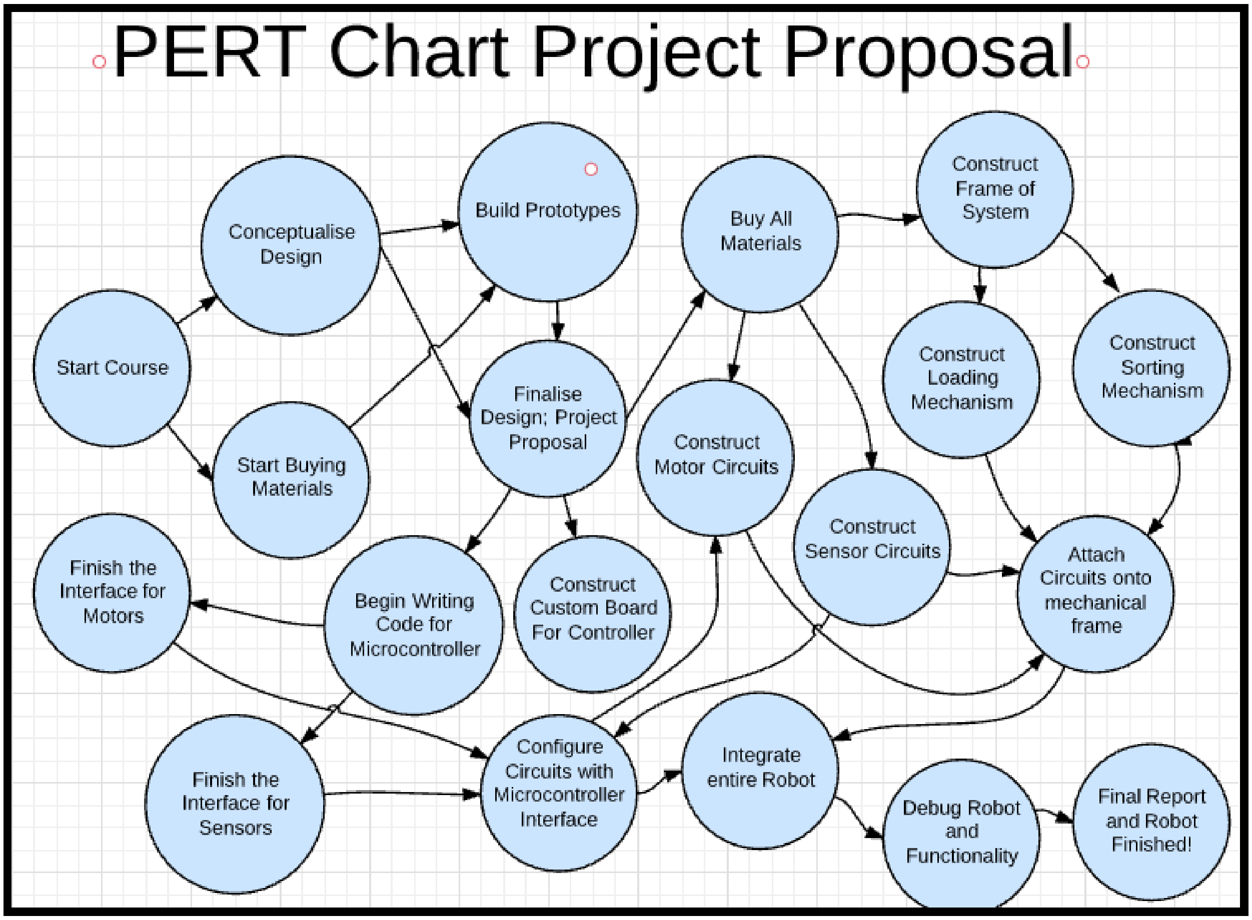


Figure 29: PERT Chart

Table 10: PERT Chart Activity Duration and Predecessors

Activity	Activity #	Duration (weeks)	Immediate Predecessor Activities
Start Course	1	1	-
Conceptualise Design	2	3	1
Build Prototypes	3	2	3,4
Start Buying Materials	4	1	1
Finalise Design: Project Proposal	5	1	2,3
Construct Custom Board for Controller	6	4	5
Buy All Materials	7	1	5
Begin Writing Code for Microcontroller	8	1	5

Construct Motor Circuits	9	2	7
Construct Sensor Circuits	10	2	7
Construct Frame of system	11	2	7
Construct Loading System	12	2	11
Construct Sorting Mechanism	13	2	11
Finish the interface for Motors	14	2	8
Finish the interface for Sensors	15	2	8
Attach Circuits onto mechanical frame	16	1	9,10,11,12
Configure Circuits with Microcontroller Interface	17	1	9,10,14,15
Integrate and build functionality of entire robot	18	4	16,17
Debug Robot and functionality	19	2	18
Final Report and Robot completion	20	2	20

Table 11: PERT Chart Activity Analysis

Activity Number	Duration (weeks)	Immediate Predecessor Activities	Expected Duration(weeks)	Earliest Start(ES)	Latest Start(LS)	Total Float(TF)	Variance
1	1	-	1	0	0	0	0
2	3	1	3	1	1	0	0
3	2	2,4	2	2	4	2	0
4	1	1	1	1	1	0	0
5	1	2,3	1.33	3.33	533	2	0.44
6	1	5	1.33	4.66	6.66	2	0.18
7	1	5	1.33	4.66	6.66	2	0.18
8	1	5	1.33	4.66	6.66	2	0.18
9	2	7	2.67	7.33	9.33	2	1.20
10	2	7	2.17	6.83	8.83	2	1.01
11	2	7	2.17	6.83	8.83	2	1.01
12	2	11	2.83	9.13	12.13	2	0.77
13	2	11	2.83	9.13	12.13	2	0.77
14	2	8	2.67	6.67	8.67	2	0.56
15	2	8	2.67	6.67	8.67	2	0.56
16	1	9,10,11,12	1.17	8.00	12.00	4	0.21
17	1	9,10,14,15	1.17	8.00	12.00	4	0.21
18	4	16,17	4.17	10.17	14.17	4	1.14
19	2	18	2.67	12.83	17.83	5	0.69
20	1	19	1.17	13.80	18.80	5	2.54

The Expected Time to complete the project will be the sum of the expected time to complete each task on the Critical Path.

$T_e = \ 15.3\ Weeks, \sigma^2 = 2.24\ Weeks.$

The formulas and algorithms used to calculate these values are given in Appendix C.

6.3 Task Assignment

This section [8] gives a description of all the specific tasks which each member is assigned to perform, regardless of scheduling and teamwork.

6.3.1 Microcontroller

The microcontroller member of the group is responsible for all the software and code needed to provide functionality to the circuits. This member is responsible for constructing a board for the microcontroller, writing code to activate the motors, sensors, and sorting mechanism, as well as creating an interface on the microcontroller.

6.3.2 Circuits

The circuits member of the group is responsible for creating the circuits necessary for the microcontroller to control the electrical components of the project. This includes obtaining all necessary sensors and motors, and soldering the circuits so that they may be easily implemented with the mechanical components, and may be connected to the microcontroller for use. The circuits member must also consider the power required in the project, and acquire a power supply and wires to power all the electrical components in the circuit, including the microcontroller.

6.3.3 Electromechanical

The electromechanical member of the project group is responsible for creating the structure and overall design of the project. This includes obtaining all necessary building materials and supplies, and working together with the group to create a design where the circuits and microcontroller may be integrated. The electromechanical member must accommodate the designs of the circuit/microcontroller member In order to be successful.

7. BUDGET

This section of the proposal summarises the cost of all the components chosen to be a part of the into the 3 subsystems for ease of viewing (see tables 12 to 14).

7.1 PURCHASING BUDGET

Table 12: Budgeting for Circuitry Components

Name of Component	Number of Component Needed	Total Price (tax not included)
TCS 34725 Colour Sensor	1	$7.95 https://www.adafruit.com/product/1334
Capacitors, Resistors, sodder, Miscellaneous circuit components	Many	<$15(estimation, prices of components relatively negligible)
High Speed Brushed DC motor (M1N10FB11G)	1	$5.12 http://www.digikey.ca/product-detail/en/nmb-technologies-corporation/M1N10FB11G/P14354-ND/2417078
Power Supply and Cables(Cables cost much less than PSU itself)	1	$20 https://techsourcecanada.ca/power-supplies/450w-power-supply/
Servo Motor(FS-90)	3	$5.95*3=17.85 http://www.trossenrobotics.com/9g-plastic
Trip-wire Circuit (price is from laser diode + Photo Resistor)	2	2($0.89+$1.00)=$3.95 https://www.dollartree.com/2-in-1-Laser-Pointer-Key-Chains/p338654/index.pro http://www.digikey.com/product-detail/en/luna-optoelectronics/PDV-P8103/PDV-P8103-ND/480610
Protoboards for soldering	3	3*$4.88= $14.64 https://www.creatroninc.com/product/half-size-protoyping-board-breadboard-type/
L7805 Voltage Regulator	1	$0.95 https://www.sparkfun.com/products/107
Subtotal for Circuits	$84.51	

Table 13: Budgeting for Electromechanical Components

Name of Component	Number of Component Needed	Total Price (tax not included)

2" X 2" X 8' Wood	3	$4.68 * 3 = $14.04 https://www.lowes.ca/dimensional-lumber/top-choice-2-x-2-x-8-ft-brown-pressure-treated-lumber_g2463012.html
14" X 5' Aluminum Sheet	2	$13.99 * 2 = $27.98 https://www.homehardware.ca/en/rec/index.htm/Building-Supplies/Building-Materials/Roofing-Products/Underlayment/Valley-Flashing/14-x-5-Roll-12-Gauge-Aluminum-Flashing/_/N-2pqfZ67l/Ne-67n/No-24/Ntk-All_EN/R-I2610847?Ntt=Jahine&Num=0
1/8'' Diameter X 12 Feet Aluminum Rod (Product # 6061)	4	$5.40 * 4 = $21.6 https://www.metalsdepot.com/products/alum2.phtml?page=round
¼' X 2' X 2' Wood Sheet	1	$7.99 https://www.homehardware.ca/en/rec/index.htm/Building-Supplies/Forest-Products/Panels/Fir-Dfp/G1s/2-x-2-x-1-4-Good-One-Side-Fir-Plywood/_/N-2pqfZ67l/Ne-67n/Ntk-All_EN/R-I2811434?Ntt=Plywood+1%2F4
Collection Bins	2	$3 * 2 = $6 http://www.dollarama.com/
Screws, Washers, Nuts	Many	< $20
Gears and Shafts	Many	<$10
Subtotal for Electromechanical		$ 97.61

Table 14: Budgeting for Microcontroller Components

Name of Component	Number of Component Needed	Total Price (tax not included)
Driver Board	1	$25 Design Kit
PIC	1	$5 Design Kit
LCD + Keypad	1	$6 Design Kit
RTC	1	$6 Design Kit
Subtotal for Micro Controller		$42.00

Grand Total: $229.12

7.2 Power Budget

This section of the proposal details the power needs of all of the electronic components of the circuit, and outlines the power supply needed to provide this power, originating from a standard 120V AC wall outlet. The results are summarized in the following table (table 15):

Table 15: Power Budget

Circuit Component Name	Supply Current (mA)	Supply Voltage (V)	Power Total (W)
Color sensor TCS34725	0.235	5V	0.001175
PIC18F4620 Microcontroller	1.25	5V	0.00625
3x Micro Servo Motor(FS90)	3*600	4.8V	3* 2.88=8.64
High Speed Brushed DC motor	800	12V	9.6
IR-tripwire circuit	0.9	5V	0.0045

Total Power: ~ 19W

Estimated Efficiency: ~ 50%

Power Supply Needed: 38W, 12V.

This means that the Computer PSU that was chosen as described in the budget will be much more than sufficient to power the project.

8. Conclusion

The project proposal is intended to describe to the reader the goal of the project, and the intentions of the team members in the choices that were made regarding the proposed design as well as the scheduling of the project. Through analysis of each component of the project, the design has been optimised to a point where it exceeds the expectations of the project, and falls well within the constraints given. The intent is for the group to follow the details outlined in this document to successfully construct a recycling sorting robot. Overall, the design is reliable, cheap, simple to implement, and most of all effective.

9. Appendix

Appendix A: References

[1]Kunio S. et al(1968) 'Mechanism of Gravity Flow of particles from a hopper', *Industrial & Engineering Chemistry Process Design and Development*378-383

[2] "TCRT500L infrared reflector" *Robot Shop. N.p, n.d* Web 01 Feb 2017 Available: http://www.robotshop.com/media/files/PDF/vishay-TCRT5000L-infrared-reflector-specs.pdf\

[3 "RGB Color Sensor with IR Filter and White LED - TCS34725." *Adafruit Industries Blog RSS*. N.p., n.d. Web. 01 Feb. 2017. Available: https://www.adafruit.com/product/1334

[4] FuzzyWobble. "Recycling Robot - Test Run." *YouTube*. YouTube, 13 Aug. 2016. Web. 01 Feb. 2017. Available: https://www.youtube.com/watch?v=ajvYIVDUu-M

[5] "Aer201- Engineering Design" *Emami, Reza.* Web, N.p. 01 Feb, 2017. Available: *http://aer201.aerospace.utoronto.ca/course/syllabus.htm*

[6] "M1N10FB11G." *NMB Technologies Corporation | Motors, Solenoids, Driver Boards/Modules | DigiKey*. N.p., n.d. Web. 01 Feb. 2017.Available: http://www.digikey.ca/product-detail/en/nmb-technologies-corporation/M1N10FB11G/P14354-ND/2417078

[7]Emami, Reza. "Multidisciplinary Engineering Design, From theory to practice." McGraw-Hill, 2017 Edition, Canada, Entire Book. Date of Access: Feb 1, 2017.

[8] "PIC18F2525/2620/4525/4620 Data Sheet". Microchip. N.p, n.d. Web. Available:

http://ww1.microchip.com/downloads/en/DeviceDoc/39626e.pdf

[9] Saaty, T.L. (2008) 'Decision making with the analytic hierarchy process', Int. J. Services Sciences, Vol. 1, No. 1, pp.83–98.

[10] "Laser Trip Wire Circuit : Repository." *Circuits Laser Trip Wire Circuit L32185 - Next.gr*. N.p., n.d. Web. 01 Feb. 2017. Available: http://www.next.gr/circuits/laser-trip-wire-circuit-l32185.html

[11] "Servo Motor Control: Schematic." *PyroElectro Electronics News Projects Tutorials RSS*. N.p., n.d. Web. 01 Feb. 2017. http://www.pyroelectro.com/tutorials/servo_motor/schematic.html

[12] "RGB Color Sensor with IR Filter and White LED - TCS34725." *Adafruit Industries Blog RSS*. N.p., n.d. Web. 01 Feb. 2017. Available: https://cdn-shop.adafruit.com/datasheets/TCS34725.pdf

[13] "What Is a Gantt Chart?" *What Is a Gantt Chart? Gantt Chart Information, History and Software*. N.p., n.d. Web. 01 Feb. 2017. Available: http://www.gantt.com/index1.htm?utm_expid=11664174-46.EjGD5xWgTOia25IJzkiv2w.1&utm_referrer=https%3A%2F%2Fwww.google.ca%2F

[14] "Critical Path Analysis and PERT Charts: Planning and Scheduling More Complex Projects." *Critical Path Analysis and PERT - Project Management from Mind Tools*. N.p., n.d. Web. 01 Feb. 2017. Available: https://www.mindtools.com/critpath.html

[15]"Engineering Materials", *Engineeringtoolbox.com*, 2017. [Online]. Available: http://www.engineeringtoolbox.com/engineering-materials-properties-d_1225.html. [Accessed: 31- Jan- 2017].

[16]"Friction and Friction Coefficients", *Engineeringtoolbox.com*, 2017. [Online]. Available: http://www.engineeringtoolbox.com/friction-coefficients-d_778.html. [Accessed: 31- Jan- 2017].

APPENDIX B: AHP FOR DETERMINING BOTTLE/CAP DISTINGUISHING METHOD

Pair wise Comparisons of Importance of Criteria Matrix

	Cost	Effectiveness	Ease of Implementation	Simplicity	Reliability	Aesthetic	NORMALISED
Cost	1/1	1/2	1/2	1/1	1/2	5/1	0.157
Effectiveness	2/1	1/1	1/1	2/1	1/1	7/1	0.211
Ease of Implementation	2/1	1/1	1/1	1/1	1/1	7/1	0.211
Simplicity	1/1	1/2	1/1	1/1	1/1	5/1	0.157
Reliability	2/1	1/1	1/1	1/1	1/1	7/1	0.211
Aesthetic	1/5	1/7	1/7	1/5	1/7	1/1	0.053

The Relative Importance Scales Goes As Follows:
1 Equal 2 Barely Better 3 Moderate 4 Quite Better 5 Stronger
6 Much stronger 7 Very strong 8 Heavily Favored 9 Extremely Favored

The above table shows the Relative importance table of each criterion normalised.

Comparison of Proposed Bottle/Cap Distinguishing Methods using Normalization

Cost

	Normalised Value	Relative Value
Shape Detection	0.080	2
Color Sensor	0.234	7
Image Processing Software	0.053	1
Pressure Sensor	0.234	7
Reflectivity sensor	0.234	7
Acoustic Sensor	0.166	5

Simplicity

	Normalised Value	Relative Value
Shape Detection	0.153	2
Color Sensor	0.255	9
Image Processing Software	0.053	1
Pressure Sensor	0.208	7
Reflectivity sensor	0.208	7
Acoustic Sensor	0.173	4

Cost

	Normalised Value	Relative Value
Shape Detection	0.122	4
Color Sensor	0.166	5
Image Processing Software	0.074	2
Pressure Sensor	0.251	7
Reflectivity sensor	0.251	7
Acoustic Sensor	0.166	5

Effectiveness

	Relative Value	Normalised Value
Image Processing Software	5	0.164
Pressure Sensor	3	0.120
Shape Detection	1	0.068
Color Sensor	9	0.302
Reflectivity sensor	6	0.194
Acoustic Sensor	4	0.152

Ease of Implementation

	Relative Value	Normalised Value
Shape Detection	2	0.103
Color Sensor	8	0.255
Image Processing Software	2	0.103
Pressure Sensor	7	0.218
Reflectivity sensor	7	0.218
Acoustic Sensor	2	0.103

Effectiveness

	Relative Value	Normalised Value
Image Processing Software	5	0.181
Pressure Sensor	4	0.121
Shape Detection	5	0.181
Color Sensor	9	0.281
Reflectivity sensor	7	0.233
Acoustic Sensor	2	0.081

Summary Matrix of AHP for Bottle/ Cap Distinguishing

PROPOSED SOLUTIONS FOR DETECTING BOTTLE/CAP	NORMALISED MATRIX OF RELATIVE VALUES OF EACH SOLUTION						RELATIVE WEIGHT OF EACH CRITERION	OVERALL SUMMARY SCORE OF EACH SOLUTION
	Cost	Simplicity	Reliability	Ease of Implementation	Effectiveness	Aesthetic		
SHAPE DETECTION	0.080	0.153	0.181	0.103	0.068	0.122	0.157	0.117
COLOR SENSOR	0.234	0.255	0.281	0.255	0.302	0.166	0.157	0.262
IMAGE PROCESSING SOFTWARE	0.053	0.053	0.181	0.103	0.154	0.074	0.211	0.113
PRESSURE SENSOR	0.234	0.208	0.121	0.218	0.120	0.251	0.211	0.180
REFLECTIVITY SENSOR	0.234	0.208	0.233	0.218	0.194	0.251	0.211	0.219
ACOUSTIC SENSOR	0.166	0.173	0.061	0.103	0.152	0.166	0.053	0.123

Appendix C: PERT Algorithm and Method

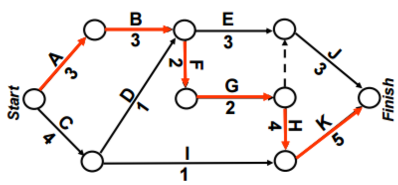
AER201 - Engineering Design

Project Planning

U. Toronto Institute for Aerospace Studies

Critical Path Method (CPM)

- **Earliest Start (ES):** *The earliest time the activity can start (through the longest path).*
- **Project Duration:** *Check the activities that terminate at Finish event. Add their own duration to their ES. The largest value is the Project Duration. (Project Duration below = 19)*
- **Latest Start (LS):** *The latest time the activity can start (through the longest path) and still have the project be completed within the project duration time.*
- **Total Float (TF):** *The difference between the latest possible starting time and the earliest possible starting time of the activity. It indicates how much the activity can be delayed while still allowing the entire project to be completed on time.*
- **Critical Path (CP):** *The sequence of activities for which* **TF = 0.**



Activity	Duration	Preceded By	ES	LS	TF
A	3		0	0	0
B	3	A	3	3	0
C	4		0	1	1
D	1	C	4	5	1
E	3	B , D	6	13	7
F	2	A , B , D	6	6	0
G	2	C , F	8	8	0
H	4	G	10	10	0
I	1	C	4	13	9
J	3	E , G	10	16	6
K	5	F , H , I	14	14	0

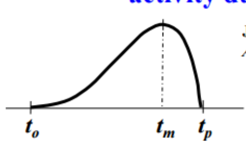
AER201 - Engineering Design

Project Planning

U. Toronto Institute for Aerospace Studies

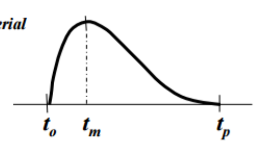
Program Evaluation & Review Technique (PERT)

- **Explicitly incorporates uncertainties regarding activity durations into CPM.**
- **Assign 3 durations (instead of 1 in CPM) for the completion of each activity:**
  - **Optimistic Estimate (t_0):** *the shortest time in which the activity can be completed assuming everything goes right.*
  - **Most-likely Estimate (t_m):** *most likely time that will be required to complete the activity.*
  - **Pessimistic Estimate (t_p):** *the longest time it will take this activity to be completed assuming many things go wrong.*
- **A statistically significant number of engineers have been interviewed for exemplary projects, and it is shown that the estimate of each activity duration follows the Beta Probability Distribution.**



Left-skewed

J.R. Meredith, and S.J. Mantel, "*Project Management: a Managerial Approach*," 2nd Edition, John Wiley and Sons, New York, 1989.



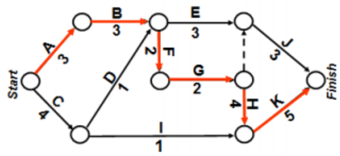
Right-skewed

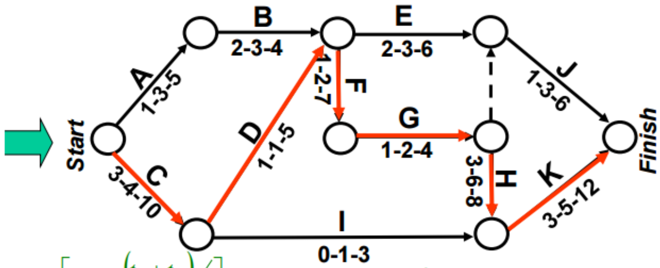
AER201 - Engineering Design

Project Planning

U. Toronto Institute for Aerospace Studies

Program Evaluation & Review Technique (PERT)





(Project Duration = 19 weeks)

$$t_e = \frac{\left[2t_m + \left(t_o + t_p\right)\!/_2\right]}{3} \qquad \sigma^2 = \left(\frac{t_p - t_o}{6}\right)^2 \qquad TF = LS - ES$$

Expected Time to Complete the Project:

$$T_e = \sum_{Critical\,Path} t_e = 21.34 \quad \text{(weeks)}$$

Variance of the Project Completion Duration:

$$\sigma_e^2 = \sum_{Critical\,Path} \sigma^2 = 5.55$$

Activity	Expected Time	ES	LS	TF	Variance
A	3.00	0.00	0.50	0.50	0.44
B	3.00	3.00	3.50	0.50	0.11
C	4.83	0.00	0.00	0.00	1.36
D	1.67	4.83	4.83	0.00	0.44
E	3.33	6.50	14.84	8.34	0.44
F	2.67	6.50	6.50	0.00	1.00
G	2.17	9.17	9.17	0.00	0.25
H	4.17	11.34	11.34	0.00	0.25
I	1.17	4.83	14.34	9.51	0.25
J	3.17	11.34	18.17	6.83	0.69
K	5.83	15.51	15.51	0.00	2.25

AER201 - Engineering Design

Project Planning

U. Toronto Institute for Aerospace Studies

Program Evaluation & Review Technique (PERT)

Q: what is the chance of completing the project in a specific duration?

Expected Time to Complete the Project:

$$T_e = \sum_{Critical\,Path} t_e = 21.34 \quad \text{(weeks)}$$

Given a sufficient number of activities, the project duration follows a normal probability distribution. (Central Limit Theorem)

Variance of the Project Completion Duration:

$$\sigma_e^2 = \sum_{Critical\,Path} \sigma^2 = 5.55$$

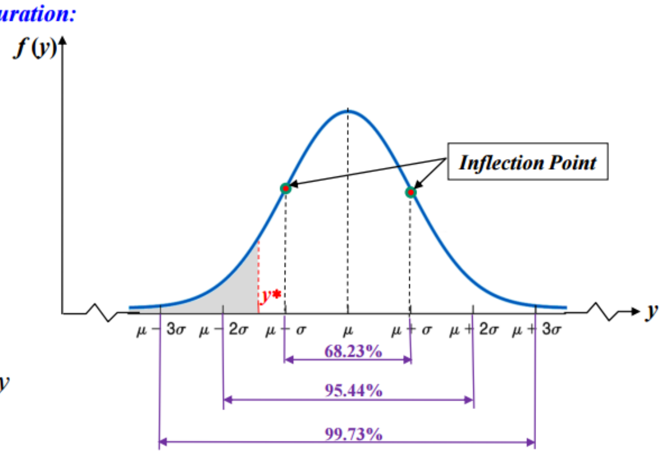
Normal Probability Density Function

$$f(y) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\left[(y-\mu)^2/_{2\sigma^2}\right]}$$

$$\int_{-\infty}^{+\infty} f(y)\,dy = 1$$

$$\Pr(y < y^*) = \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^{y^*} e^{-\left[(y-\mu)^2/_{2\sigma^2}\right]} dy$$

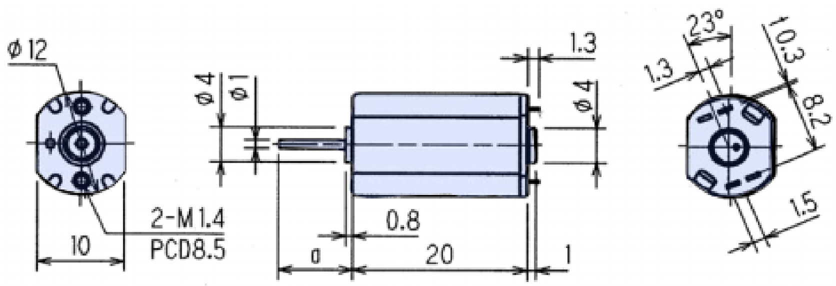
$$\Pr(y > y^*) = 1 - \Pr(y < y^*)$$



NMB-MAT
Minebea Motor Manufacturing Corporation
eMINEBEA.COM

M1N10

Outline UNIT:mm



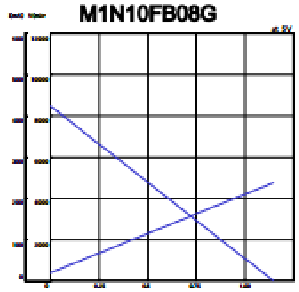
Weight: 17.7 g

Specifications

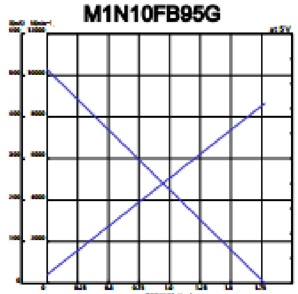
Model	Operating Voltage (V)	Rated Voltage (V)	No Load Speed (min⁻¹)	No Load Current (mA)	Rated Load (gf·cm)	Rated Load (mN·m)	Rated Load Speed (min⁻¹)	Rated Load Current (mA)	Starting Torque (gf·cm)	Starting Torque (mN·m)	Starting Current (mA)	Shaft Length (mm)
M1N10FB08G	1 to 6	5.0	8483	20	3.1	0.3	6258	77	11.7	1.1	238	a■8.5
M1N10FB95G	1 to 6	5.0	10277	20	3.1	0.3	8542	90	18.1	1.8	433	a■8.5
M1N10FB09F	1 to 6	5.0	10822	23	3.1	0.3	8714	96	15.7	1.5	399	a■6.5
M1N10FB10G	1 to 6	5.0	13316	36	3.1	0.3	10988	131	17.5	1.7	575	a■8.5
M1N10FB11G	1 to 6	5.0	15906	36	3.1	0.3	13850	145	23.7	2.3	874	a■8.5

Characteristics

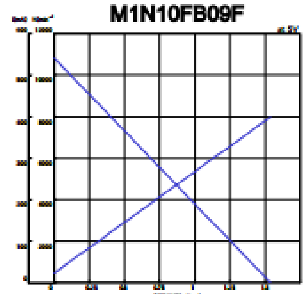
M1N10FB08G



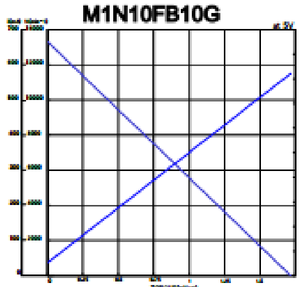
M1N10FB95G



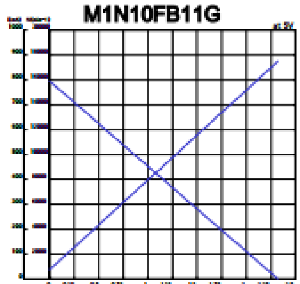
M1N10FB09F



M1N10FB10G



M1N10FB11G



Model No.: FS90

Product Description

Features

-Micro analog plastic gears servo
-Operating Voltage:4.8-6Volts
-Interface: (like JR)
-Wire length: 20cm

Power	4.8V	6V
Speed	0.12sec/60degree	0.07sec/60degree
Torque	1.3kg.cm/18.9oz.in	
Weight	9g(0.32oz)	
Size	23.2*12.5*22.0mm	
Application	for Air plane	

Connection description

Orange = Signal input
Red = +5V
Brown = 0V

Typical Signals

MIN_WIDTH 544 // shortest pulse sent to a servo 0°
MAX_WIDTH 2400 // longest pulse sent to a servo 180°
NEUTRAL_PULSE_WIDTH 1500 // MID pulse width when servo is at 90°
REFRESH_INTERVAL 20000 // min time to refresh in microseconds

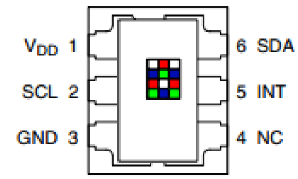
TCS3472
COLOR LIGHT-TO-DIGITAL CONVERTER with IR FILTER

TAOS135 – AUGUST 2012

Features

- Red, Green, Blue (RGB), and Clear Light Sensing with IR Blocking Filter
  - Programmable Analog Gain and Integration Time
  - 3,800,000:1 Dynamic Range
  - Very High Sensitivity — Ideally Suited for Operation Behind Dark Glass
- Maskable Interrupt
  - Programmable Upper and Lower Thresholds with Persistence Filter
- Power Management
  - Low Power — 2.5-μA Sleep State
  - 65-μA Wait State with Programmable Wait State Time from 2.4 ms to > 7 Seconds
- I²C Fast Mode Compatible Interface
  - Data Rates up to 400 kbit/s
  - Input Voltage Levels Compatible with V_{DD} or 1.8 V Bus
- Register Set and Pin Compatible with the TCS3x71 Series
- Small 2 mm × 2.4 mm Dual Flat No-Lead (FN) Package

PACKAGE FN
DUAL FLAT NO-LEAD
(TOP VIEW)



Package Drawing Not to Scale

Applications

- RGB LED Backlight Control
- Light Color Temperature Measurement
- Ambient Light Sensing for Display Backlight Control
- Fluid and Gas Analysis
- Product Color Verification and Sorting

End Products and Market Segments

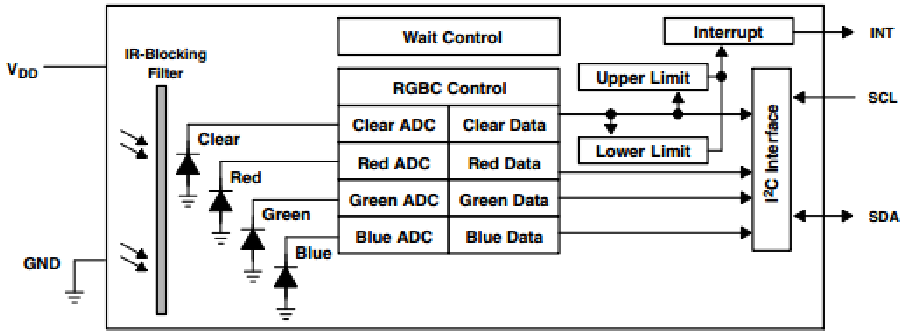
- TVs, Mobile Handsets, Tablets, Computers, and Monitors
- Consumer and Commercial Printing
- Medical and Health Fitness
- Solid State Lighting (SSL) and Digital Signage
- Industrial Automation

Description

The TCS3472 device provides a digital return of red, green, blue (RGB), and clear light sensing values. An IR blocking filter, integrated on-chip and localized to the color sensing photodiodes, minimizes the IR spectral component of the incoming light and allows color measurements to be made accurately. The high sensitivity, wide dynamic range, and IR blocking filter make the TCS3472 an ideal color sensor solution for use under varying lighting conditions and through attenuating materials.

The TCS3472 color sensor has a wide range of applications including RGB LED backlight control, solid-state lighting, health/fitness products, industrial process controls and medical diagnostic equipment. In addition, the IR blocking filter enables the TCS3472 to perform ambient light sensing (ALS). Ambient light sensing is widely used in display-based products such as cell phones, notebooks, and TVs to sense the lighting environment and enable automatic display brightness for optimal viewing and power savings. The TCS3472, itself, can enter a lower-power wait state between light sensing measurements to further reduce the average power consumption.

Functional Block Diagram



Detailed Description

The TCS3472 light-to-digital converter contains a 3 × 4 photodiode array, four analog-to-digital converters (ADC) that integrate the photodiode current, data registers, a state machine, and an I^2C interface. The 3 × 4 photodiode array is composed of red-filtered, green-filtered, blue-filtered, and clear (unfiltered) photodiodes. In addition, the photodiodes are coated with an IR-blocking filter. The four integrating ADCs simultaneously convert the amplified photodiode currents to a 16-bit digital value. Upon completion of a conversion cycle, the results are transferred to the data registers, which are double-buffered to ensure the integrity of the data. All of the internal timing, as well as the low-power wait state, is controlled by the state machine.

Communication of the TCS3472 data is accomplished over a fast, up to 400 kHz, two-wire I^2C serial bus. The industry standard I^2C bus facilitates easy, direct connection to microcontrollers and embedded processors.

In addition to the I^2C bus, the TCS3472 provides a separate interrupt signal output. When interrupts are enabled, and user-defined thresholds are exceeded, the active-low interrupt is asserted and remains asserted until it is cleared by the controller. This interrupt feature simplifies and improves the efficiency of the system software by eliminating the need to poll the TCS3472. The user can define the upper and lower interrupt thresholds and apply an interrupt persistence filter. The interrupt persistence filter allows the user to define the number of consecutive out-of-threshold events necessary before generating an interrupt. The interrupt output is open-drain, so it can be wire-ORed with other devices.

Power Management

Power consumption can be managed with the Wait state, because the Wait state typically consumes only 65 µA of I_{DD} current. An example of the power management feature is given below. With the assumptions provided in the example, average I_{DD} is estimated to be 152 µA.

Table 1. Power Management

SYSTEM STATE MACHINE STATE	PROGRAMMABLE PARAMETER	PROGRAMMED VALUE	DURATION	TYPICAL CURRENT
Wait	WTIME	0xEE	43.2 ms	0.065 mA
	WLONG	0		
RGBC Init			2.40 ms	0.235 mA
RGBC ADC	ATIME	0xEE	43.2 ms	0.235 mA

Average I_{DD} Current = ((43.2 × 0.065) + (43.2 × 0.235) + (2.40 × 0.235)) / 89 ≈ 152 µA

Keeping with the same programmed values as the example, Table 2 shows how the average I_{DD} current is affected by the Wait state time, which is determined by WEN, WTIME, and WLONG. Note that the worst-case current occurs when the Wait state is not enabled.

Table 2. Average I_{DD} Current

WEN	WTIME	WLONG	WAIT STATE	AVERAGE I_{DD} CURRENT
0	n/a	n/a	0 ms	291 µA
1	0xFF	0	2.40 ms	280 µA
1	0xEE	0	43.2 ms	152 µA
1	0x00	0	614 ms	82 µA
1	0x00	1	7.37 s	67 µA

Terminal Functions

TERMINAL NAME	TERMINAL NO.	TYPE	DESCRIPTION
GND	3		Power supply ground. All voltages are referenced to GND.
INT	5	O	Interrupt — open drain (active low).
NC	4	O	No connect — do not connect.
SCL	2	I	I^2C serial clock input terminal — clock signal for I^2C serial data.
SDA	6	I/O	I^2C serial data I/O terminal — serial data I/O for I^2C .
V_{DD}	1		Supply voltage.

Available Options

DEVICE	ADDRESS	PACKAGE – LEADS	INTERFACE DESCRIPTION	ORDERING NUMBER
TCS34721†	0x39	FN–6	I^2C Vbus = V_{DD} Interface	TCS34721FN
TCS34723†	0x39	FN–6	I^2C Vbus = 1.8 V Interface	TCS34723FN
TCS34725	0x29	FN–6	I^2C Vbus = V_{DD} Interface	TCS34725FN
TCS34727	0x29	FN–6	I^2C Vbus = 1.8 V Interface	TCS34727FN

† Contact TAOS for availability.

Absolute Maximum Ratings over operating free-air temperature range (unless otherwise noted)†

Supply voltage, V_{DD} (Note 1) . . . . 3.8 V
Input terminal voltage . . . . −0.5 V to 3.8 V
Output terminal voltage . . . . −0.5 V to 3.8 V
Output terminal current . . . . −1 mA to 20 mA
Storage temperature range, T_{stg} . . . . −40°C to 85°C
ESD tolerance, human body model . . . . 2000 V

† Stresses beyond those listed under "absolute maximum ratings" may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated under "recommended operating conditions" is not implied. Exposure to absolute-maximum-rated conditions for extended periods may affect device reliability.

NOTE 1: All voltages are with respect to GND.

Recommended Operating Conditions

	MIN	NOM	MAX	UNIT
Supply voltage, V_{DD} (TCS34721 & TCS34725) (I^2C V_{bus} = V_{DD})	2.7	3	3.6	V
Supply voltage, V_{DD} (TCS34723 & TCS34727) (I^2C V_{bus} = 1.8 V)	2.7	3	3.3	V
Operating free-air temperature, T_A	−30		70	°C

Operating Characteristics, V_{DD} = 3 V, T_A = 25°C (unless otherwise noted)

PARAMETER		TEST CONDITIONS	MIN	TYP	MAX	UNIT
I_{DD}	Supply current	Active		235	330	μA
		Wait state		65		
		Sleep state — no I^2C activity		2.5	10	
V_{OL}	INT, SDA output low voltage	3 mA sink current	0		0.4	V
		6 mA sink current	0		0.6	
I_{LEAK}	Leakage current, SDA, SCL, INT pins		−5		5	μA
I_{LEAK}	Leakage current, LDR pin		−5		5	μA
V_{IH}	SCL, SDA input high voltage	TCS34721 & TCS34725	0.7 V_{DD}			V
		TCS34723 & TCS34727	1.25			
V_{IL}	SCL, SDA input low voltage	TCS34721 & TCS34725			0.3 V_{DD}	V
		TCS34723 & TCS34727			0.54	

Optical Characteristics, V_{DD} = 3 V, T_A = 25°C, AGAIN = 16×, ATIME = 0xF6 (unless otherwise noted) (Note 1)

PARAMETER		TEST CONDITIONS	Red Channel			Green Channel			Blue Channel			Clear Channel			UNIT
			MIN	TYP	MAX	MIN	TYP	MAX	MIN	TYP	MAX	MIN	TYP	MAX	
R_e	Irradiance responsivity	λ_D = 465 nm Note 2	0%		15%	10%		42%	65%		88%	11.0	13.8	16.6	counts/ μW/ cm^2
		λ_D = 525 nm Note 3	4%		25%	60%		85%	10%		45%	13.2	16.6	20.0	
		λ_D = 615 nm Note 4	80%		110%	0%		14%	5%		24%	15.6	19.5	23.4	

NOTES:
1. The percentage shown represents the ratio of the respective red, green, or blue channel value to the clear channel value.
2. The 465 nm input irradiance is supplied by an InGaN light-emitting diode with the following characteristics: dominant wavelength λ_D = 465 nm, spectral halfwidth $\Delta\lambda½$ = 22 nm.
3. The 525 nm input irradiance is supplied by an InGaN light-emitting diode with the following characteristics: dominant wavelength λ_D = 525 nm, spectral halfwidth $\Delta\lambda½$ = 35 nm.
4. The 615 nm input irradiance is supplied by a AlInGaP light-emitting diode with the following characteristics: dominant wavelength λ_D = 615 nm, spectral halfwidth $\Delta\lambda½$ = 15 nm.

RGBC Characteristics, VDD = 3 V, TA = 25°C, AGAIN = 16×, AEN = 1 (unless otherwise noted)

PARAMETER	TEST CONDITIONS	MIN	TYP	MAX	UNIT
Dark ADC count value	E_e = 0, AGAIN = 60×, ATIME = 0xD6 (100 ms)	0	1	5	counts
ADC integration time step size	ATIME = 0xFF	2.27	2.4	2.56	ms
ADC number of integration steps (Note 5)		1		256	steps
ADC counts per step (Note 5)		0		1024	counts
ADC count value (Note 5)	ATIME = 0xC0 (153.6 ms)	0		65535	counts
Gain scaling, relative to 1× gain setting	4×	3.8	4	4.2	×
	16×	15	16	16.8	
	60×	58	60	63	

NOTE 5: Parameter ensured by design and is not tested.

Wait Characteristics, V_{DD} = 3 V, T_A = 25°C, WEN = 1 (unless otherwise noted)

PARAMETER	TEST CONDITIONS	CHANNEL	MIN	TYP	MAX	UNIT
Wait step size	WTIME = 0xFF		2.27	2.4	2.56	ms
Wait number of integration steps (Note 1)			1		256	steps

NOTE 1: Parameter ensured by design and is not tested.

AC Electrical Characteristics, V_{DD} = 3 V, T_A = 25°C (unless otherwise noted)

PARAMETER†		TEST CONDITIONS	MIN	TYP	MAX	UNIT
$f_{(SCL)}$	Clock frequency (I²C only)		0		400	kHz
$t_{(BUF)}$	Bus free time between start and stop condition		1.3			μs
$t_{(HDSTA)}$	Hold time after (repeated) start condition. After this period, the first clock is generated.		0.6			μs
$t_{(SUSTA)}$	Repeated start condition setup time		0.6			μs
$t_{(SUSTO)}$	Stop condition setup time		0.6			μs
$t_{(HDDAT)}$	Data hold time		0			μs
$t_{(SUDAT)}$	Data setup time		100			ns
$t_{(LOW)}$	SCL clock low period		1.3			μs
$t_{(HIGH)}$	SCL clock high period		0.6			μs
t_F	Clock/data fall time				300	ns
t_R	Clock/data rise time				300	ns
C_i	Input pin capacitance				10	pF

† Specified by design and characterization; not production tested.

PARAMETER MEASUREMENT INFORMATION

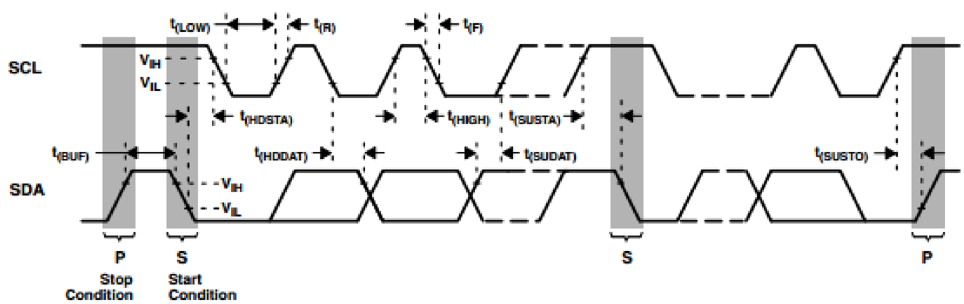


Figure 1. Timing Diagrams

Power Management

Power consumption can be managed with the Wait state, because the Wait state typically consumes only 65 µA of I_{DD} current. An example of the power management feature is given below. With the assumptions provided in the example, average I_{DD} is estimated to be 152 µA.

Table 1. Power Management

SYSTEM STATE MACHINE STATE	PROGRAMMABLE PARAMETER	PROGRAMMED VALUE	DURATION	TYPICAL CURRENT
Wait	WTIME	0xEE	43.2 ms	0.065 mA
	WLONG	0		
RGBC Init			2.40 ms	0.235 mA
RGBC ADC	ATIME	0xEE	43.2 ms	0.235 mA

Average I_{DD} Current = ((43.2 × 0.065) + (43.2 × 0.235) + (2.40 × 0.235)) / 89 ≈ 152 µA

Keeping with the same programmed values as the example, Table 2 shows how the average I_{DD} current is affected by the Wait state time, which is determined by WEN, WTIME, and WLONG. Note that the worst-case current occurs when the Wait state is not enabled.

Table 2. Average I_{DD} Current

WEN	WTIME	WLONG	WAIT STATE	AVERAGE I_{DD} CURRENT
0	n/a	n/a	0 ms	291 µA
1	0xFF	0	2.40 ms	280 µA
1	0xEE	0	43.2 ms	152 µA
1	0x00	0	614 ms	82 µA
1	0x00	1	7.37 s	67 µA

Wait Characteristics, V_{DD} = 3 V, T_A = 25°C, WEN = 1 (unless otherwise noted)

PARAMETER	TEST CONDITIONS	CHANNEL	MIN	TYP	MAX	UNIT
Wait step size	WTIME = 0xFF		2.27	2.4	2.56	ms
Wait number of integration steps (Note 1)			1		256	steps

NOTE 1: Parameter ensured by design and is not tested.

AC Electrical Characteristics, V_{DD} = 3 V, T_A = 25°C (unless otherwise noted)

PARAMETER†		TEST CONDITIONS	MIN	TYP	MAX	UNIT
$f_{(SCL)}$	Clock frequency (I^2C only)		0		400	kHz
$t_{(BUF)}$	Bus free time between start and stop condition		1.3			μs
$t_{(HDSTA)}$	Hold time after (repeated) start condition. After this period, the first clock is generated.		0.6			μs
$t_{(SUSTA)}$	Repeated start condition setup time		0.6			μs
$t_{(SUSTO)}$	Stop condition setup time		0.6			μs
$t_{(HDDAT)}$	Data hold time		0			μs
$t_{(SUDAT)}$	Data setup time		100			ns
$t_{(LOW)}$	SCL clock low period		1.3			μs
$t_{(HIGH)}$	SCL clock high period		0.6			μs
t_F	Clock/data fall time				300	ns
t_R	Clock/data rise time				300	ns
C_i	Input pin capacitance				10	pF

† Specified by design and characterization; not production tested.

PARAMETER MEASUREMENT INFORMATION

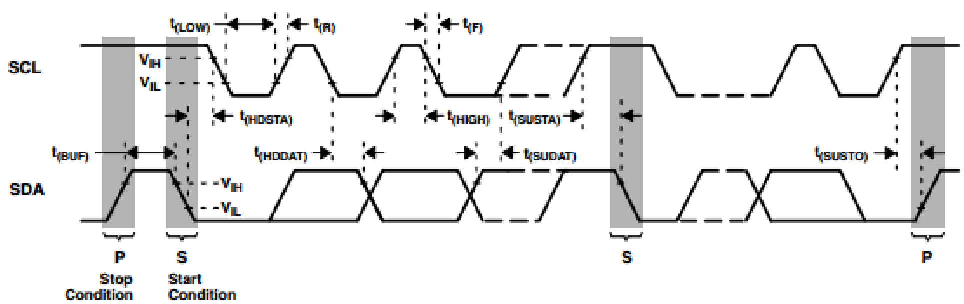
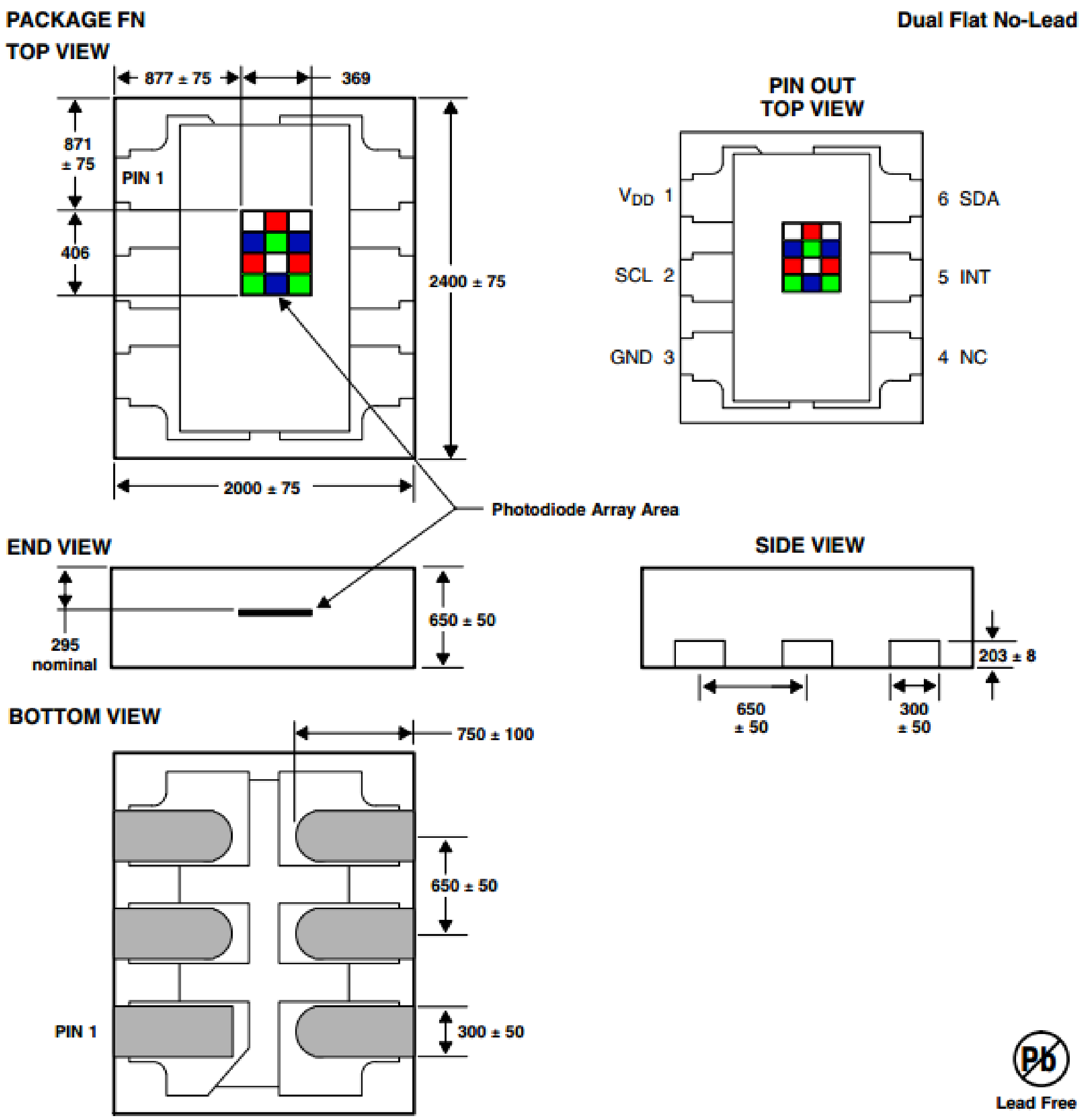


Figure 1. Timing Diagrams

PACKAGE INFORMATION



NOTES:
- A. All linear dimensions are in micrometers. Dimension tolerance is ± 20 μm unless otherwise noted.
- B. The die is centered within the package within a tolerance of ± 3 mils.
- C. Package top surface is molded with an electrically nonconductive clear plastic compound having an index of refraction of 1.55.
- D. Contact finish is copper alloy A194 with pre-plated NiPdAu lead finish.
- E. This package contains no lead (Pb).
- F. This drawing is subject to change without notice.

Figure 12. Package FN — Dual Flat No-Lead Packaging Configuration

PRINCIPLES OF OPERATION

System States

An internal state machine provides system control of the RGBC and power management features of the device. At power up, an internal power-on-reset initializes the device and puts it in a low-power Sleep state.

When a start condition is detected on the I^{2}C bus, the device transitions to the Idle state where it checks the Enable Register (0x00) PON bit. If PON is disabled, the device will return to the Sleep state to save power. Otherwise, the device will remain in the Idle state until the RGBC function is enabled (AEN). Once enabled, the device will execute the Wait and RGBC states in sequence as indicated in Figure 5. Upon completion and return to Idle, the device will automatically begin a new Wait-RGBC cycle as long as PON and AEN remain enabled.

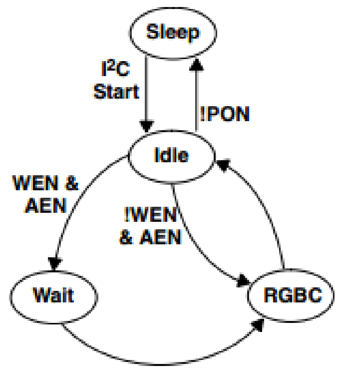


Figure 6. Simplified State Diagram

TYPICAL CHARACTERISTICS

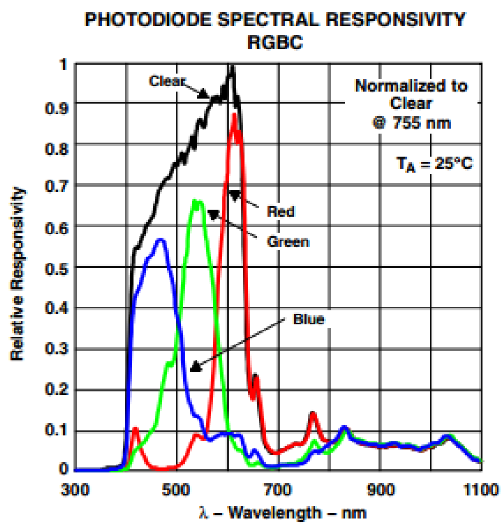


Figure 2

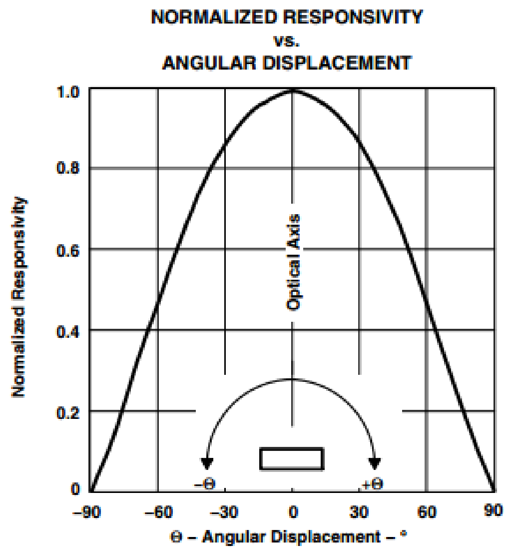


Figure 3

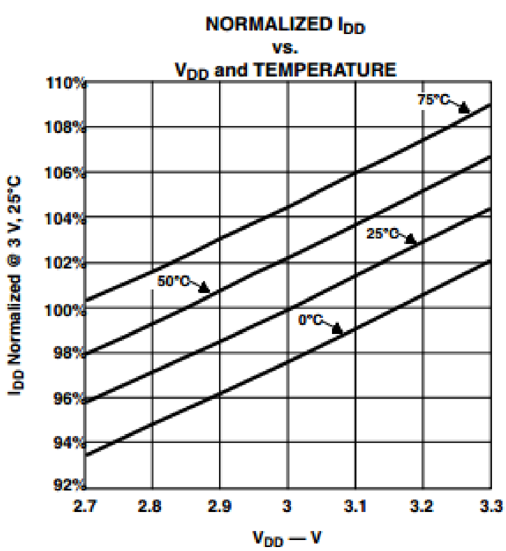


Figure 4

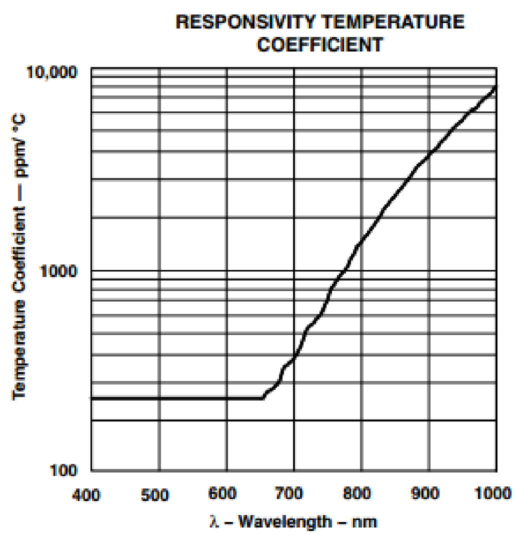


Figure 5

SOLDERING INFORMATION

The FN package has been tested and has demonstrated an ability to be reflow soldered to a PCB substrate. The process, equipment, and materials used in these test are detailed below.

The solder reflow profile describes the expected maximum heat exposure of components during the solder reflow process of product on a PCB. Temperature is measured on top of component. The components should be limited to a maximum of three passes through this solder reflow profile.

Table 15. Solder Reflow Profile

PARAMETER	REFERENCE	DEVICE
Average temperature gradient in preheating		2.5°C/sec
Soak time	t_{soak}	2 to 3 minutes
Time above 217°C (T1)	t_1	Max 60 sec
Time above 230°C (T2)	t_2	Max 50 sec
Time above T_{peak} −10°C (T3)	t_3	Max 10 sec
Peak temperature in reflow	T_{peak}	260°C
Temperature gradient in cooling		Max −5°C/sec

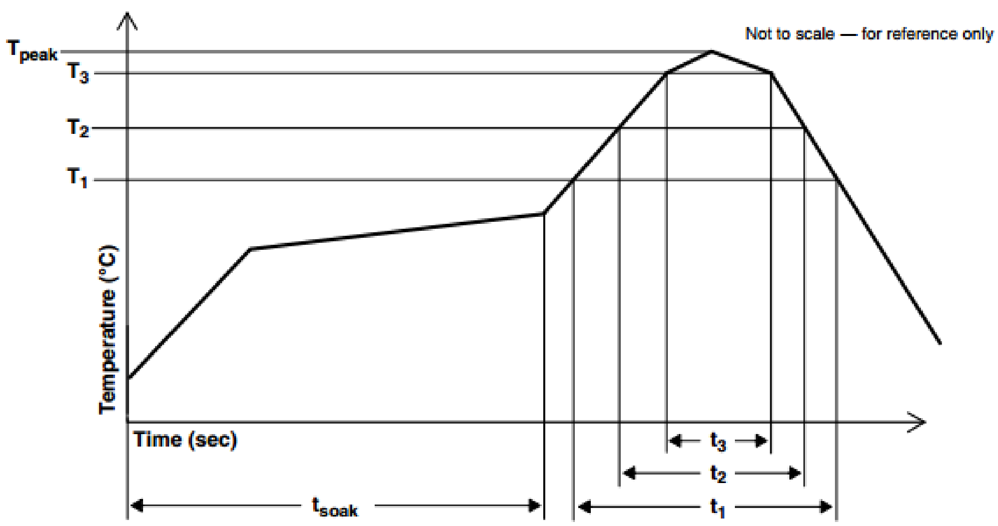


Figure 14. Solder Reflow Profile Graph

STORAGE INFORMATION

Moisture Sensitivity

Optical characteristics of the device can be adversely affected during the soldering process by the release and vaporization of moisture that has been previously absorbed into the package. To ensure the package contains the smallest amount of absorbed moisture possible, each device is dry-baked prior to being packed for shipping. Devices are packed in a sealed aluminized envelope called a moisture barrier bag with silica gel to protect them from ambient moisture during shipping, handling, and storage before use.

The Moisture Barrier Bags should be stored under the following conditions:

Temperature Range	< 40°C
Relative Humidity	< 90%
Total Time	No longer than 12 months from the date code on the aluminized envelope if unopened.

Rebaking of the reel will be required if the devices have been stored unopened for more than 12 months and the Humidity Indicator Card shows the parts to be out of the allowable moisture region.

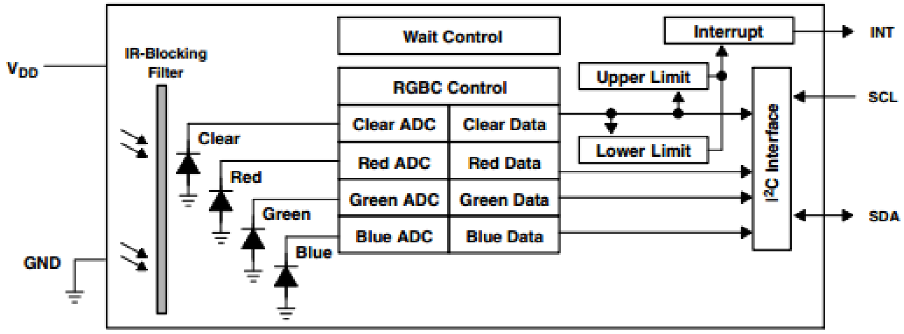
Opened reels should be used within 168 hours if exposed to the following conditions:

Temperature Range	< 30°C
Relative Humidity	< 60%

If rebaking is required, it should be done at 50°C for 12 hours.

The FN package has been assigned a moisture sensitivity level of MSL 3.

Functional Block Diagram



Detailed Description

The TCS3472 light-to-digital converter contains a 3 × 4 photodiode array, four analog-to-digital converters (ADC) that integrate the photodiode current, data registers, a state machine, and an I^{2}C interface. The 3 × 4 photodiode array is composed of red-filtered, green-filtered, blue-filtered, and clear (unfiltered) photodiodes. In addition, the photodiodes are coated with an IR-blocking filter. The four integrating ADCs simultaneously convert the amplified photodiode currents to a 16-bit digital value. Upon completion of a conversion cycle, the results are transferred to the data registers, which are double-buffered to ensure the integrity of the data. All of the internal timing, as well as the low-power wait state, is controlled by the state machine.

Communication of the TCS3472 data is accomplished over a fast, up to 400 kHz, two-wire I^{2}C serial bus. The industry standard I^{2}C bus facilitates easy, direct connection to microcontrollers and embedded processors.

In addition to the I^{2}C bus, the TCS3472 provides a separate interrupt signal output. When interrupts are enabled, and user-defined thresholds are exceeded, the active-low interrupt is asserted and remains asserted until it is cleared by the controller. This interrupt feature simplifies and improves the efficiency of the system software by eliminating the need to poll the TCS3472. The user can define the upper and lower interrupt thresholds and apply an interrupt persistence filter. The interrupt persistence filter allows the user to define the number of consecutive out-of-threshold events necessary before generating an interrupt. The interrupt output is open-drain, so it can be wire-ORed with other devices.

Appendix E: Material Properties [15]

Composites

Material	Density - ρ - (10^3 kg/m^3)	Tensile Modulus - E - (GPa)	Tensile Strength - σ - (GPa)	Specific Modulus - E/ρ -	Specific Strength - σ/ρ -	Maximum Service Temperature (^oC)
Short-fiber						
Glass-filled epoxy (35%)	1.9	25	0.3	13.2	0.16	80 - 200
Glass-filled polyester (35%)	2.0	15.7	0.13	7.85	0.065	80 - 125
Glass-filled nylon (35%)	1.6	14.5	0.2	8.95	0.12	75 - 110
Unidirectional						
S-glass epoxy (45%)	1.8	39.5	0.87	21.8	0.48	80 - 215
Carbon epoxy (61%)	1.6	142	1.73	89.3	1.08	80 - 215
Kevlar epoxy (53%)	1.35	63.6	1.1	47.1	0.81	80 - 215

Metals

Material	Density - ρ - (10^3 kg/m^3)	Tensile Modulus - E - (GPa)	Tensile Strength - σ - (GPa)	Specific Modulus - E/ρ -	Specific Strength - σ/ρ -	Maximum Service Temperature (^oC)
Cast Iron, grade 20	7.15	100	0.14	14.3	0.02	230 - 300
Steel, AISI 1045	7.7 - 8.03	205	0.585	26.3	0.073	500 - 650
Aluminum 2045-T4	2.7	73	0.45	27	0.17	150 - 250
Aluminum 6061-T6	2.7	69	0.27	25.5	0.10	150 - 250

Ceramics

Material	Density - ρ - (10^3 kg/m^3)	Tensile Modulus - E - (GPa)	Tensile Strength - σ - (GPa)	Specific Modulus - E/ρ -	Specific Strength - σ/ρ -	Maximum Service Temperature (^oC)
Alumina	3.8	350	0.17	92.1	0.045	1425 - 1540
MgO	3.6	205	0.06	56.9	0.017	900 - 1000

Plastics

Material	Density - ρ - (10^3 kg/m^3)	Tensile Modulus - E - (GPa)	Tensile Strength - σ - (GPa)	Specific Modulus - E/ρ -	Specific Strength - σ/ρ -	Maximum Service Temperature (^oC)
Nylon 6/6	1.15	2 - 3.6	0.082	2.52	0.071	75 - 100
Polyethylene (HDPE)	0.9 - 1.4	0.18 - 1.6	0.015			
Polypropylene	0.9 - 1.24	1.4	0.033	1.55	0.037	50 - 80
Epoxy	1.25	3.5	0.069	2.8	0.055	80 - 215
Phenolic	1.35	3.0	0.006	2.22	0.004	70 - 120

- *1 GPa = 1.45 10^6 psi*
- *1 kg/m^3 = 0.0624 lb/ft^3*

Wood Species	Stress *(psi)*: Bending, Horizontal Shear		Compression: Perpendicular to Grain		Compression: Parallel to Grain	
	Wet	Dry	Wet	Dry	Wet	Dry
Birch, Yellow	1417	1668	477	715	960	1200
Fir, Douglas	1417	1668	417	625	1360	1700
Larch, Western	1417	1668	417	625	1360	1700
Maple, Red	1271	1495	410	615	880	1100
Oak, Black	1369	1610	590	885	920	1150
Pine, Eastern White	1222	1438	223	335	960	1200
Redwood	1320	1553	433	650	1200	1500

APPENDIX F: COEFFICIENT OF FRICTION [16]

Friction Coefficients for some Common Materials and Materials Combinations

Materials and Material Combinations		Static Frictional Coefficient - μ_s -	
		Clean and Dry Surfaces	Lubricated and Greasy Surfaces
Aluminum	Aluminum	1.05 - 1.35	0.3
Aluminum-bronze	Steel	0.45	
Aluminum	Mild Steel	0.61	
Brake material[2)]	Cast iron	0.4	
Brake material[2)]	Cast iron (wet)	0.2	
Brass	Steel	0.35	0.19
Brass	Cast Iron	0.3[1)]	
Brick	Wood	0.6	
Bronze	Steel		0.16
Bronze	Cast Iron	0.22[1)]	
Bronze - sintered	Steel		0.13
Cadmium	Cadmium	0.5	0.05
Cadmium	Chromium	0.41	0.34
Cadmium	Mild Steel	0.46[1)]	
Cast Iron	Cast Iron	1.1, 0.15[1)]	0.07[1)]
Cast Iron	Oak	0.49[1)]	0.075[1]
Cast iron	Mild Steel	0.4, 0.23[1)]	0.21, 0.133[1)]
Car tire	Asphalt	0.72	
Car tire	Grass	0.35	
Carbon (hard)	Carbon	0.16	0.12 - 0.14
Carbon	Steel	0.14	0.11 - 0.14
Chromium	Chromium	0.41	0.34
Copper-Lead alloy	Steel	0.22	
Copper	Copper	1	0.08
Copper	Cast Iron	1.05, 0.29[1)]	
Copper	Mild Steel	0.53, 0.36[1)]	0.18[1)]
Diamond	Diamond	0.1	0.05 - 0.1
Diamond	Metal	0.1 - 0.15	0.1
Glass	Glass	0.9 - 1.0, 0.4[1)]	0.1 - 0.6, 0.09-0.12[1)]
Glass	Metal	0.5 - 0.7	0.2 - 0.3
Glass	Nickel	0.78	0.56
Graphite	Steel	0.1	0.1
Graphite	Graphite (in vacuum)	0.5 - 0.8	
Graphite	Graphite	0.1	0.1
Hemp rope	Timber	0.5	
Horseshoe	Rubber	0.68	
Horseshoe	Concrete	0.58	
Ice	Ice	0.02 - 0.09	
Ice	Wood	0.05	
Ice	Steel	0.03	
Iron	Iron	1.0	0.15 - 0.20
Lead	Cast Iron	0.43[1)]	
Leather	Oak	0.61, 052[1]	
Leather	Metal	0.4	0.2
Leather	Wood	0.3 - 0.4	
Leather	Clean Metal	0.6	
Leather fiber	Cast iron	0.31	
Leather fiber	Aluminum	0.30	
Magnesium	Magnesium	0.6	0.08
Masonry	Brick	0.6 - 0.7	
		0.7 - 1.1	