

**Mobile Robotics Documentation**

Dan Carpenter and Elias Peluso

Millersville University

AENG 467: Mobile Robotics

Dr. John Wright

December 10, 2021

### **Abstract**

Mobile robotics is an area within applied engineering that is distinguishable from other applications of robotics. It involves applying engineering, design, and coding practices to creating a machine that can solve a physical problem. With a prodigious amount of interconnected parts and environmental variables, the work typically involves fine-tuning and experimenting with the code, components, and design. This essay describes in detail the given task for a mobile robotics competition, including the results and conclusions. The parts listing, flow chart, final code, and component tech sheets are included as appendices.

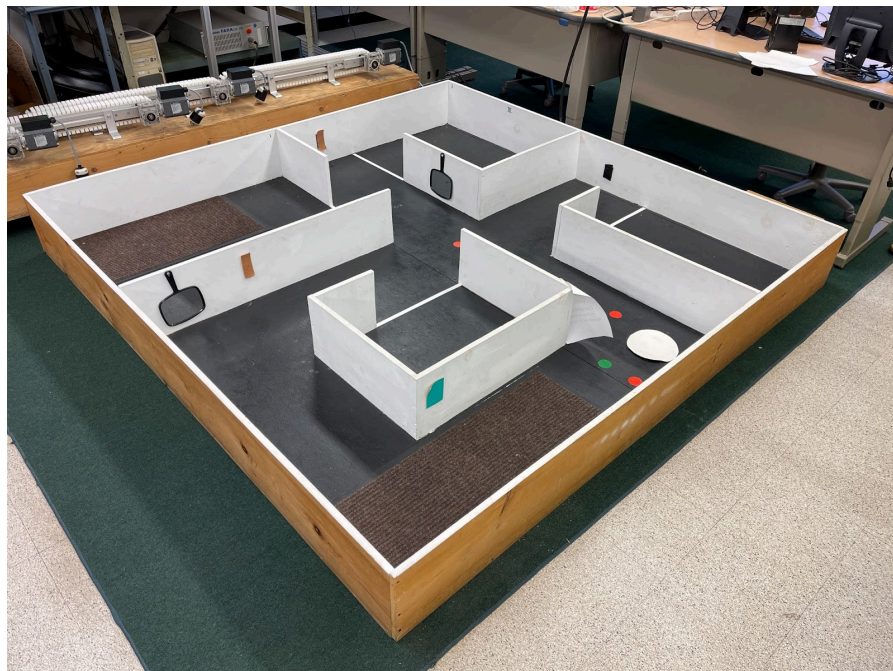
**Table of Contents**

Project Description .....	Page 4
Problem Statement.....	Page 4
Impact.....	Page 5
Objectives .....	Page 6
Procedure .....	Page 9
Results .....	Page 12
Conclusion .....	Page 15
Appendices .....	Page 17
Appendix A: Parts Listing .....	Page 17
Appendix B: Process Flow Chart/Algorithm .....	Page 20
Appendix C: Complete Programming Code with Comments .....	Page 26
Appendix D: Fire Contest Rules Official Document.....	Page 36
Appendix E: Component Tech Sheets .....	Page 42

## Project Description

### Problem Statement

This research and development project involves creating a robot that can function in a simulation of a real-world environment to complete a specific task. As outlined in the “Fire Contest Rules” document (see Appendix D), the objective of this project was to create a firefighting mobile robot that navigates through a maze (a simulation of a house) to extinguish a candle. This general goal is relatively simple, but the challenge has underlying complexity that adds additional problems to the statement.





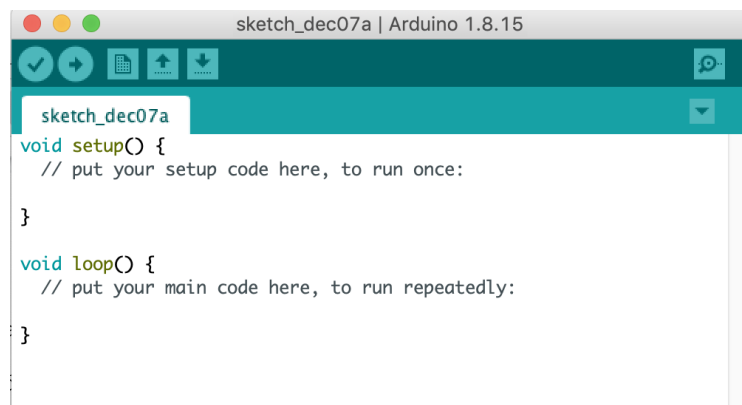


Some other factors that increase complexity are the components (see Appendix A), the trial conditions (see Appendix D) and environmental conditions. First of all, it is noteworthy that this challenge requires participants to program at least ten unique electronic components (see Appendix A). The fact that all of these components must work together in the same code adds a problem to the challenge: stable and reliable code. Second, the conditions of each trial run are randomized each round, adding another problem. Because conditions such as candle location, candle mat size, and occupant location (see Appendix D) are uncertain, the robot must be programmed to function under this uncertainty. Third, environmental conditions such as lighting, discoloration in the maze, background objects, and obstacles in the maze may disrupt the intended function of sensors. Therefore, these problems must be accounted for and tested prior to the competition.

### **Impact**

The purpose of this project is to provide students with exposure to a simulation of a real “on-the-field” robotics challenge. As a “senior-level” and “capstone” robotics course, AENG 467 requires the application of a great amount of knowledge from various areas of applied engineering. These areas include design, coding, testing, engineering, refining, critical thinking,

and teamwork skills. As explained in the problem statement, the explicit goal of this project is to create a robot that functions in a simulation of a real-world house fire environment. This reflects the global purpose of this course: to prepare students for real-world robotics experiences by creating a hands-on, independent work environment.



```
sketch_dec07a | Arduino 1.8.15
sketch_dec07a
void setup() {
  // put your setup code here, to run once:
}

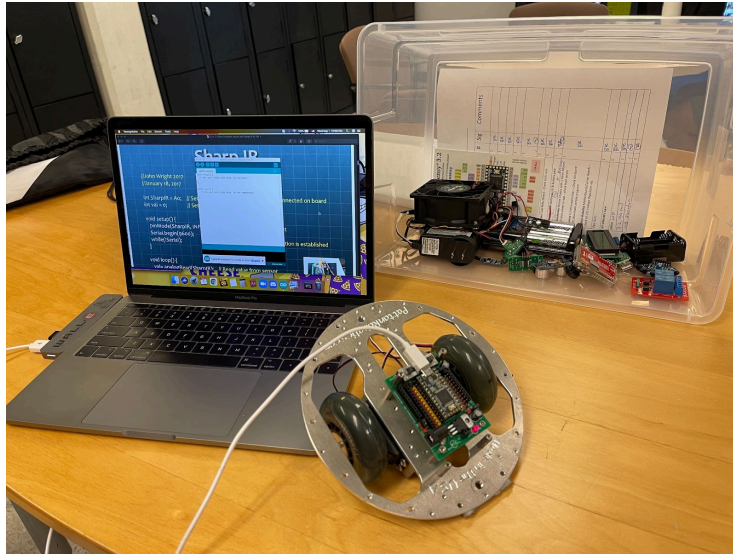
void loop() {
  // put your main code here, to run repeatedly:
}
```

## Objectives

As mentioned previously, the objective of this project was to create a firefighting mobile robot that navigates through a maze (a simulation of a house) to extinguish a candle. The official contest rules (see Appendix D) explain the entire competition very clearly:

“The main challenge of this contest is to build an autonomous robot using a Teensy Microcontroller that can find its way through an arena that represents a model house, find a lit candle that represents a fire in the house, and extinguish the fire in the shortest time. This task simulates the real-world operation of an autonomous robot performing a fire protection function in a real house. The goal of the contest is to advance robot technology and knowledge while using robotics as an educational tool. The contest shall also teach students about the limitations of technology, and how to deal with complex control problems and situations” (Wright, 2021, page 2).

Within this general end goal of creating a firefighting robot, there are a few “checkpoints” the robot must pass to ensure it will succeed. The first was for the robot to autonomously drive down one of the maze corridors and turn right. This required the programming and engineering of both servos and at least one type of vision sensor (see Appendix A). The second checkpoint required the robot to autonomously drive down a corridor and into one of the “rooms,” stopping on the white line at the entrance. This added the need to implement components such as the line tracking sensor, and possibly additional vision sensors (see Appendix A). Three additional functions required of the robot were to detect whether or not a flame is present (utilizing the flame detection sensor), determine whether a green color, a red color, or no color is present (utilizing the color camera), and to display messages on the LCD screen (see Appendix A). The third checkpoint compiled all previous tasks and added the final step: if the robot detects a flame after stopping on a white line, it must drive towards the flame and extinguish it. This required the use of any approved flame extinguishing medium, but the provided electric fan was used by all participants (see Appendix A). Finally, the final competition entailed the use of all previously established rules, but the conditions of the maze were completely randomized.



With all this in mind, it can be inferred that at least ten components will need to be utilized in the competition. The objectives of this project can be broken down into two specific areas: engineering and coding. The “engineering” portion of this project is very much dictated by the limitations of the contest rules (see Appendix D) and the provided components (see Appendix A). The first objective of the process was to build a reliable and robust robot design. In order for the robot to be effective, the components must be mounted in a way that makes sense given the previously established rules of the competition. For example, the line sensor’s receptacle must be pointed such that it will read the white lines on the ground, and the flame sensor must be mounted at a level at which it will reliably detect a flame. So, to summarize, the physical objectives entailed the creation of an effective robot design.

The second area of objectives is coding. Using the Arduino coding platform, participants were required to write C++ code for the robot throughout (see Appendix C). As mentioned previously, code must be created to provide instructions to all components and make them work together consistently and in a way that makes sense given the competition criteria. For example, the line sensor must be labeled as an input, the servos must be labeled as outputs, and

participants must create code that instructs the servos to stop once a line is detected. All of these decisions are clearly laid out in the code flowchart (see Appendix B). Additionally, the color detection camera component was required to be programmed independently using MicroPython (see Appendix A). To summarize, the main objective in terms of coding was to increase reliability as much as possible.

### **Procedure**

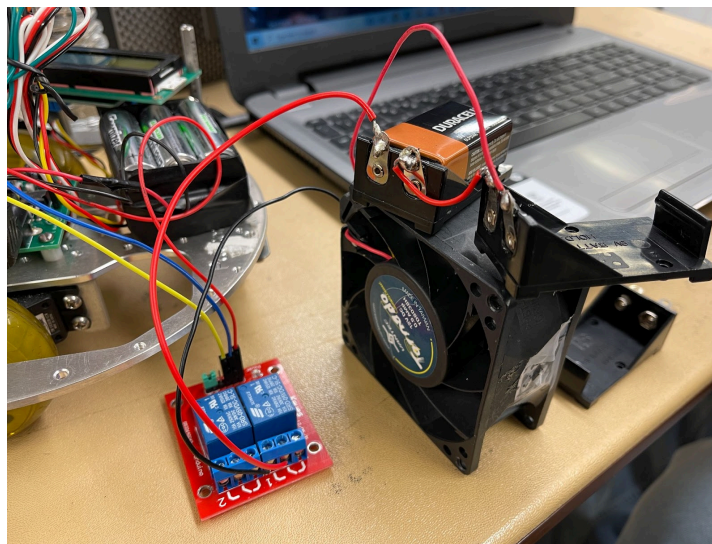
The process of actually carrying out this objective took the form of a free and independent project. AENG 467 class sessions were almost entirely unstructured, requiring teams to each plan their own course of action. A typical class session primarily involved testing various codes on the robot throughout, occasionally making physical adjustments when needed. At the beginning of the semester, students usually tested robots to verify that the code was correct and that the components were responding as intended. Toward the end of the semester, this trend shifted toward testing the robot in the actual maze and refining code that was already working, but was not yet perfect.

Now, the exact intended sequence of events for Dan and Elias's robot will be explained (see Appendix B). The robot was programmed to navigate through the maze and enter a sequence of rooms in a specific order. This plan was accomplished by starting the robot facing straight towards the far end of the maze so it would drive down the main hallway. By facing this way, it was possible for the robot to reliably enter three of the rooms by only making right turns (with the guidance of the sonar on the right side of the robot).

The robot was activated by pressing the "start" button. Upon pressing this button, the robot turned in place on the start pad, scanning the color camera to see if either red or green was detected. After this, the robot centered itself it drove straight down the main hallway, using the

right sonar to remain in line with the right wall. All the while in this main code, the robot was scanning for a line, scanning for red or green, and scanning for IR input (if it approaches a wall head on it will back up) (see Appendix B).

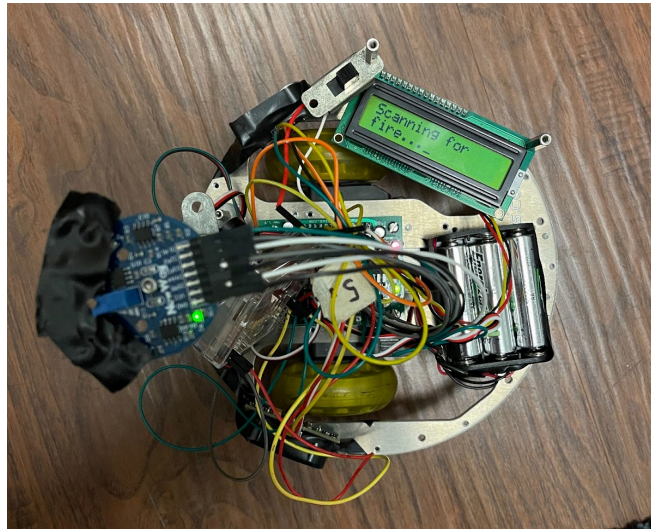
After the robot had traveled to the four-way intersection, it would turn right into the next hallway. After that it would find room #2 by turning right again. If the flame was not in that room the robot would back out and take a right turn and head back down the hallway from where it came from, turning right towards the far end of the main hallway. Then the robot would turn right again and find the second room. If the candle was not in this room, the robot would back out and turn right again. Then it would go straight down the hallway and immediately turn right to find the third room.



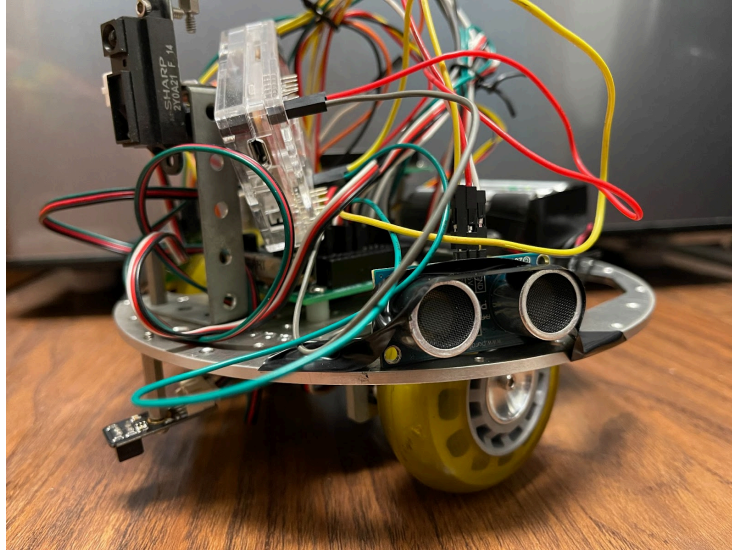
Now, the problem was how to program the robot to enter room #1 (the room in the middle of the maze), because it was not possible to enter that room using only right turns. If the robot approaches from the far side of the maze, a left turn is required to enter room #1. Since the left sonar needed to be activated at that point, a “room counter” was implemented into the code. The best way to do this was by using the line sensor because it was already supposed to stop and see the line before entering the room. Each time that the line sensor saw a line it added to the



counter and stored it into an integer variable. By doing this the robot would know how many rooms it found in the maze and where it was in the ordered sequence of rooms. Once it found the third room by continuously turning right, the room variable would then be 3. After that, we programmed the code to go into an if statement when the variable was greater than 2. So, our robot would then leave the third room by turning left to go straight down the main hallway (see Appendix B). Then, it would turn right and immediately turn left to get into the last room. So in conclusion, the robot was programmed to approach the rooms in a predetermined sequence, first by only making right turns, then by activating the left sonar for the final room.



If the robot did, in fact, detect a flame in one of the rooms, it would drive forward to enter the room. It would follow a similar pattern of using a sonar to drive alongside the wall, only this time it would use the left sonar to stay alongside the left wall when it was in one of the four rooms. The fan and the solo flame sensor were mounted on the back left side of the robot, so once a flame was detected next to the fan, the fan would turn on to extinguish it.



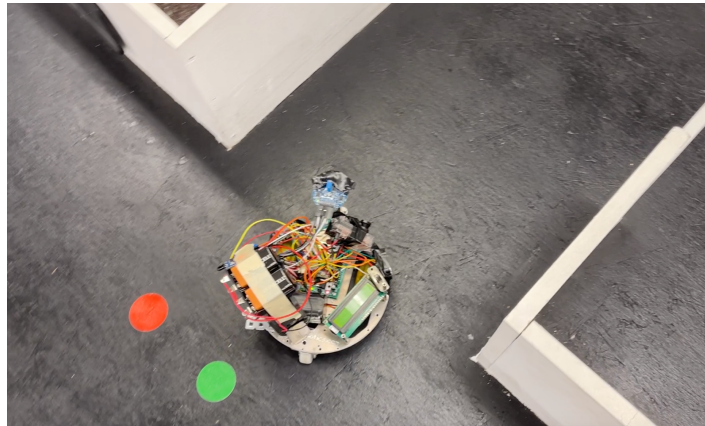
## Results

The final competition involved three attempts for every participating team. As mentioned before, the maze conditions were randomized each run, so a robot needed to be reliable and versatile in order to be successful.

Our first run was successful because our robot was able to navigate through the maze and extinguish the candle in the second room. In the beginning, the robot did not detect the Flash (the “red” occupant with the color camera), but it drove around the Flash and avoided crashing into him. This was acceptable because the robot did not misidentify the Flash and did not touch him (touching an “occupant” is grounds for disqualification; see Appendix D). The robot drove down the hallway and successfully detected the Green Lantern, who “evacuated.” The robot then turned right to go to the first room. The candle was not in the first room, so the robot left and successfully drove into the second room. While entering the second room, the robot turned too sharply and crashed into the entrance of the second room. After being stuck against the wall for about 20 seconds, our robot moved a bit, but then unfortunately got caught on the right sonar. It was at this moment that the contest judge created a change to the rules that applied to the



remainder of the rounds: the robot will not be disqualified for touching a wall for more than three seconds. Our robot eventually detected the candle in the room and wiggled free from the entrance. It entered its flame tracking subroutine code inside the room and was able to extinguish the candle. This run was a huge success.



For the second run the candle was placed in the fourth room. This room was the most difficult room to get to because it required our line sensors to properly count the number of rooms in sequence so the code could be changed appropriately for this fourth room. Due to time constraints, we unfortunately did not have a chance to test our robot's ability to enter that room. We were instead making adjustments to our code to improve reliability in entering into the first room. We were not certain the robot had the ability to enter the other three rooms, but at the end of the day on Monday (the day before the competition), our robot was able to get into the first three rooms and that was a huge success. So, we added code after the final session of Monday to attempt at getting into the fourth room, but we did not have a chance at testing it in the maze.

Our second trial for the competition was our chance to test whether or not the robot could enter room #1. At the start, the camera successfully saw both superheroes in the hallway and the robot headed towards the second room. While traveling to the second room the line sensor was triggered by one of the dots in the middle of the hallway, which unfortunately disrupted our room

counter. The robot left the second room, but thought it was leaving the third room, so it turned left and was not able to recover from that misstep. Had the robot not mistakenly detected the dots in the hallway, the code could have worked because the robot did, in fact, change its routine after the third line scan like we had hoped in our program.

The candle was placed in room #2 for our third run (the first room in our sequence). Much of the class was expecting our robot to easily succeed in this run and put out the candle because room #2 was, overall, the easiest room for anyone's robot to reliably enter. We were, however, very nervous about this run because the color camera was suddenly having white balance issues that were not present since the last run. It was confusing and concerning because the code did not change since the last run. Since we unfortunately were unable to test our robot in the maze between runs, we could not properly diagnose this issue, so we had to simply attempt the maze as is. As soon as the robot was activated in the maze, however, it did not detect the Flash, just as we had feared. It crashed into the Flash and was disqualified. It was a shame, because we could have fixed the robot's thresholds in five minutes if given the opportunity to test in the maze again.

Had our robot succeeded 2/3 runs, we would have won the competition. Instead we came in second place for the class because we were one of only two teams to have successfully completed a run, and the other team was approximately 40 seconds faster.



## Conclusion

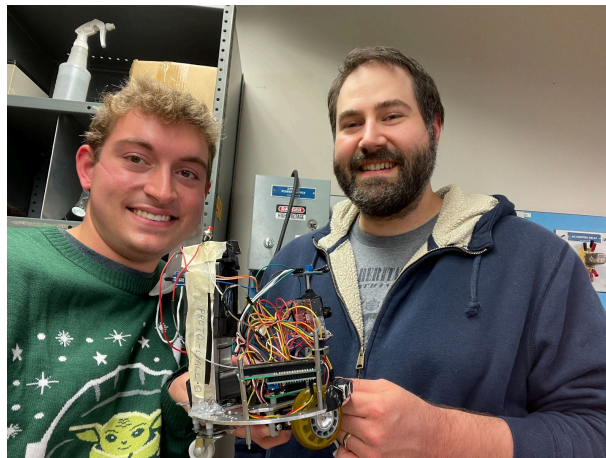
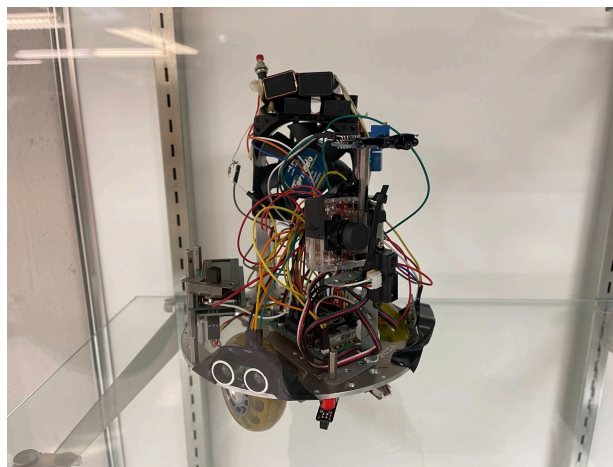
The purpose of this project was to gain “real-world” experience in robotics by researching and developing a robot to function in a simulation of a real-world environment. This semester project gave us hands-on exposure to programming the Arduino Teeny Microcontroller with Arduino C++. We also learned how to program the OpenMV camera to detect colors using MicroPython (see Appendix A). To accomplish this we had to integrate the two programming platforms together, so they would work together.

Even though everyone had made terrific progress and put a great amount of effort into their robots for this project, it is safe to say that everyone was disappointed with the results of the competition. As mentioned previously, there were only two successful runs out of the entire class. For our team in particular, we were satisfied with the result of the first run, and we were content with the robot failing the second run because the assigned room was untested. However, it was saddening that the spontaneous color detection issues cost us the candle in the third run. If the robot had detected the Flash up close in the third run, it would not have been disqualified.

We did not realize the camera was having trouble detecting the Flash up close until it was too late. While working with the camera this semester we knew that the distances were a problem. This was an issue we were unable to solve and once it cost us the competition, it was disappointing. If we would have taken more time with the camera in the maze we would have been able to improve the accuracy of the color detection. We didn't realize how quickly we ran out of time at the end of the semester and there were so many things that still needed to be tested to remain competitive. We never got a chance to test getting into the fourth room because we still had our hands full working on improving reliability in the other three rooms. Before the maze

closed last Monday we were able to get the robot to get into the first room very reliably and we knew it was able to jog around the maze and get into the other rooms.

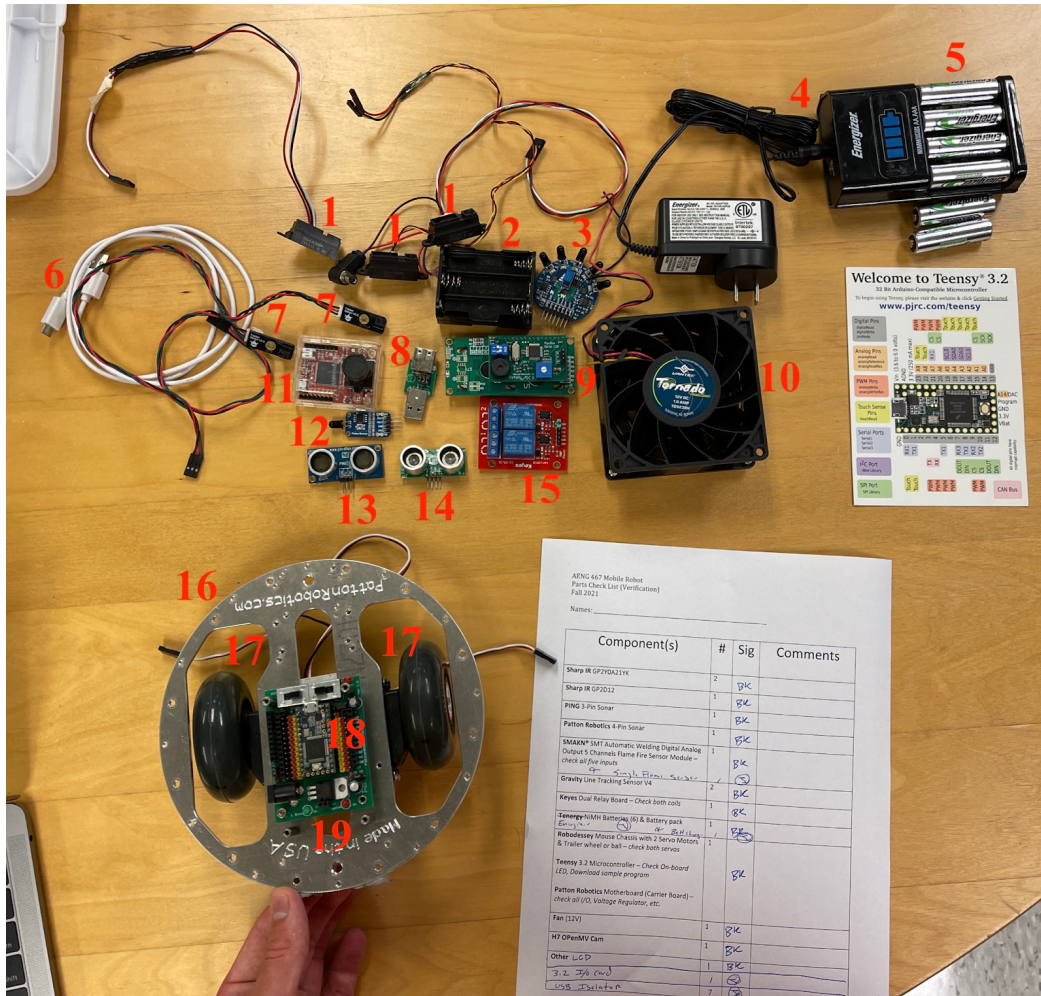
We enjoyed working on the robot and we learned a lot of terrific things in this class. This class is very practical and comprehensive because these skills are directly applicable to on-the-field work. If we were to take the class again, we would have spent more time on the color detection camera because it cost us the competition. If we had succeeded on our third run, that would have been spectacular because it would have been an accurate representation of how much we have grown as roboticists over the semester. All in all, we still did a great job with the robot and we will take the knowledge that we learned in this class and apply it to projects that we work on in the future.





















## Appendices

### Appendix A: Parts Listing

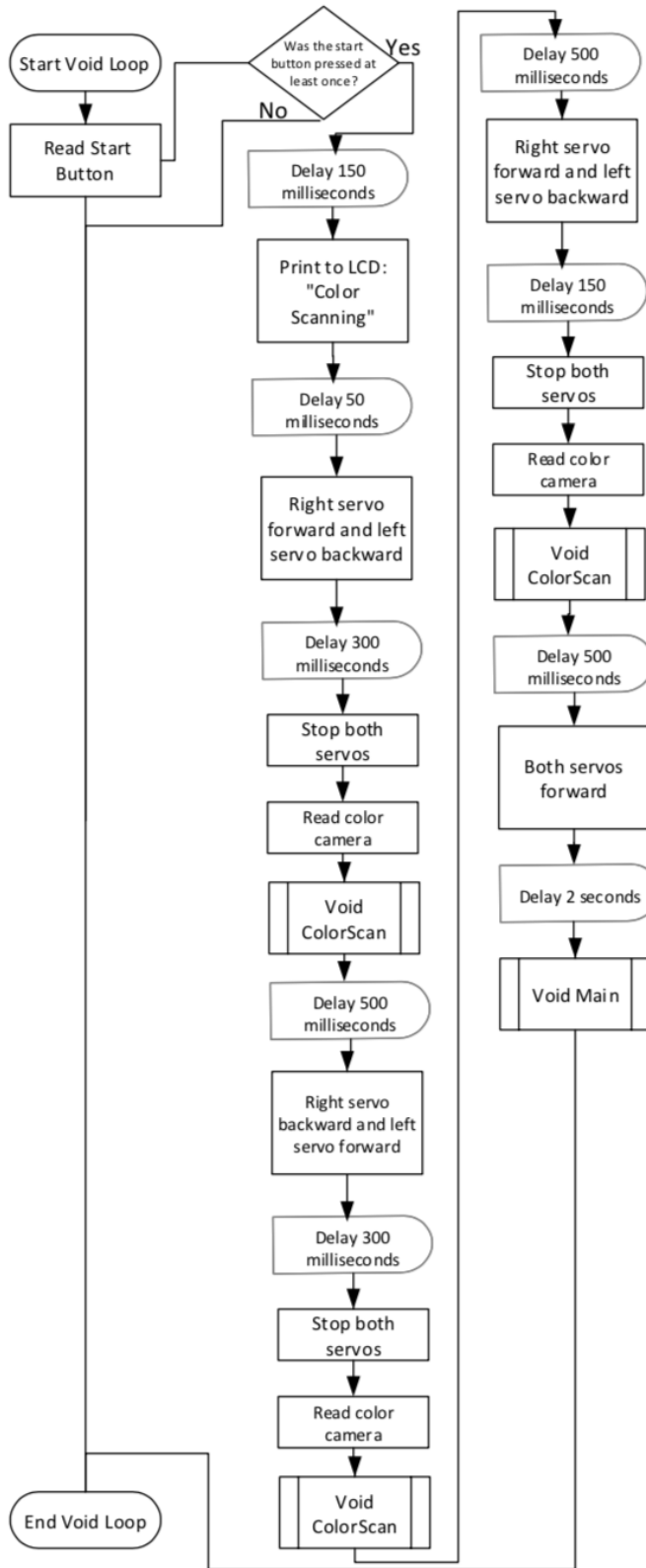


The above photo contains all components included in the project kit prior to the start of the project. The red numbers adjacent to each component correspond with the numbered component names in the following chart. If the stock image of the exact product is available from the corresponding website link, it is inserted into the chart. Following the pictured components are a list of non-pictured and miscellaneous components. Their numbers do not correspond to the above photo. If the exact component used is not available online but a similar product is available, this product will be linked instead.

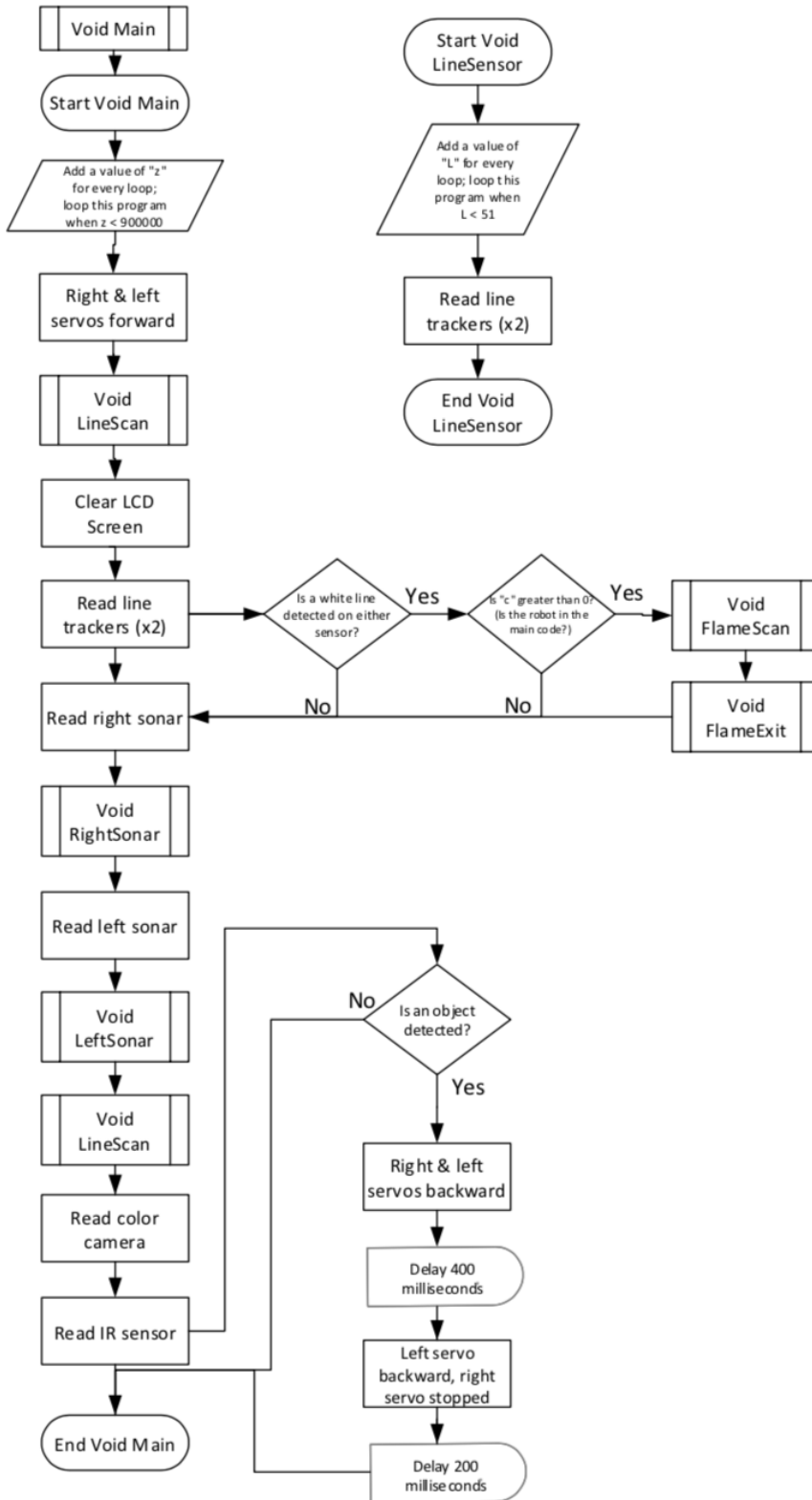
#	Part Name	Part Photo	Function	Quantity	Part Number	Official Name	Price for 1	Link to Purchase	Notes	Tech Sheets?
1	IR Sensor		Detect when an object is approximately 5 inches away from the receptacles.	3	GP2Y0A21YK	Infrared Proximity Sensor - Sharp IR	\$17.95	<a href="https://www.amazon.com/Infrared-Proximity-Sensor-Sharp-GP2Y0A21YK/dp/B001HYRGJUR">https://www.amazon.com/Infrared-Proximity-Sensor-Sharp-GP2Y0A21YK/dp/B001HYRGJUR</a>	Three duplicates of this component were included in the kit (three components are pictured), but only one was used in the actual project.	See Appendix E, item # 1.
2	Battery Pack	N/A	Hold the batteries for 6x AA alkaline batteries and connect to the board via a Vin DC power plug.	1	N/A	Similar product: *6AA Battery Holder With 2.1 mm x 5.5mm Connector 9V Output 2 Pack by Corpco	Price for similar: \$6.99	Link to similar: <a href="https://www.amazon.com/Battery-Holder-Connector-Output-Corpco/dp/B011RX4DOU">https://www.amazon.com/Battery-Holder-Connector-Output-Corpco/dp/B011RX4DOU</a>		
3	5-Channels Flame Sensor		Detect whether or not a flame is present. Out of all five pins, determine which pin(s) was/were activated.	1	OP0716	*SMAKN* SMT Automatic Welding Digital Analog Output 5 Channels Flame Fire Sensor Module	\$1.30	<a href="https://www.alibaba.com/product-detail/S-Channel-Flame-Sensor-Module-Digital_60818193967.html">https://www.alibaba.com/product-detail/S-Channel-Flame-Sensor-Module-Digital_60818193967.html</a>		
4	Battery Charger and AC-DC Adapter		Charge the six AA batteries required to power the robot (only four at a time).	1	CH1HRWB-4	Energizer NiMH Battery Charger AA_AAA	\$41.99	<a href="https://www.amazon.com/Energizer-Charger-Rechargeable-Batteries-Charges/dp/B072S9Z5JQ/?ref=asc_df_B072S9Z5JQ?tag=&amp;linkCode=df0&amp;hvadid=309832782859&amp;hvpos=&amp;hvmw=&amp;hvrand=17642960076213515443&amp;hvpone=&amp;hvtwo=&amp;hvmq=&amp;hvdev=c=&amp;hvdymd=&amp;hvllocint=&amp;hvllocphy=1025088&amp;hvtargid=pla-529109013532&amp;mf=&amp;agripid=70155173188&amp;th=1">https://www.amazon.com/Energizer-Charger-Rechargeable-Batteries-Charges/dp/B072S9Z5JQ/?ref=asc_df_B072S9Z5JQ?tag=&amp;linkCode=df0&amp;hvadid=309832782859&amp;hvpos=&amp;hvmw=&amp;hvrand=17642960076213515443&amp;hvpone=&amp;hvtwo=&amp;hvmq=&amp;hvdev=c=&amp;hvdymd=&amp;hvllocint=&amp;hvllocphy=1025088&amp;hvtargid=pla-529109013532&amp;mf=&amp;agripid=70155173188&amp;th=1</a>		
5	AA Rechargeable Batteries (x6)		Provide power to the robot when mobile.	1	B01LVYMU30	Energizer AANiMH Rechargeable Batteries	\$20.99	<a href="https://www.amazon.com/Energizer-Rechargeable-NiMH-Battery-Batteries/dp/B01LVYMU30/?tag=hyprod-20&amp;linkCode=df0&amp;hvadid=309789297054&amp;hvpos=&amp;hvmw=&amp;hvrand=1315686558482734246&amp;hvpone=&amp;hvtwo=&amp;hvmq=&amp;hvdev=c=&amp;hvdymd=&amp;hvllocint=&amp;hvllocphy=1025088&amp;hvtargid=pla-572817680052&amp;th=1">https://www.amazon.com/Energizer-Rechargeable-NiMH-Battery-Batteries/dp/B01LVYMU30/?tag=hyprod-20&amp;linkCode=df0&amp;hvadid=309789297054&amp;hvpos=&amp;hvmw=&amp;hvrand=1315686558482734246&amp;hvpone=&amp;hvtwo=&amp;hvmq=&amp;hvdev=c=&amp;hvdymd=&amp;hvllocint=&amp;hvllocphy=1025088&amp;hvtargid=pla-572817680052&amp;th=1</a>	All 6 batteries required are sold together in this one pack, so the quantity for this part is listed as 1.	
6	USB Cable		This cable is used for connecting the Teensy board or the OpenMV Camera to a PC for coding.	1	N/A	Micro USB to USB 2.0 cable	Price for similar: \$5.26	Link to similar: <a href="https://www.amazon.com/AmazonBasics-Male-Micro-Cable-Black/dp/B0711PXXZ2/ref=asc_df_B0711PXXZ2?tag=hyprod-20&amp;linkCode=df0&amp;hvadid=198093101467&amp;hvpos=&amp;hvmw=&amp;hvrand=10932286270169966215&amp;hvpone=&amp;hvtwo=&amp;hvmq=&amp;hvdev=c=&amp;hvdymd=&amp;hvllocint=&amp;hvllocphy=1025088&amp;hvtargid=pla-359221356272&amp;psc=1">https://www.amazon.com/AmazonBasics-Male-Micro-Cable-Black/dp/B0711PXXZ2/ref=asc_df_B0711PXXZ2?tag=hyprod-20&amp;linkCode=df0&amp;hvadid=198093101467&amp;hvpos=&amp;hvmw=&amp;hvrand=10932286270169966215&amp;hvpone=&amp;hvtwo=&amp;hvmq=&amp;hvdev=c=&amp;hvdymd=&amp;hvllocint=&amp;hvllocphy=1025088&amp;hvtargid=pla-359221356272&amp;psc=1</a>		
7	Line Tracking Sensor		Detect a white line on the floor (in contrast to the black floor of the competition maze)	2	N/A	Gravity Line Tracking Sensor V4	\$4.90	<a href="https://www.dfrobot.com/product-85.html">https://www.dfrobot.com/product-85.html</a>	In this competition, the line sensor is not being used to "track" lines; it is instead being used to detect when the robot passes by a line.	See Appendix E, item # 2.
8	USB Isolator		When connected to the USB cable (6), this component's onboard pins provide users the option to bypass signals such as USB power.	1	ADUM1460	*USB to USB Board Isolation Protection Module*	\$14.27	<a href="https://www.google.com/shopping/product/1?q=usb+to+usb+isolator+arduino&amp;prds=apl:11070602780806692536_etc.11070602780806692536_0.psd:11070602780806692536_sas:X&amp;ved=0ahUKEwj6I_qbedl0AhXYgnIEHXaUcDwQ9wGC&amp;ao">https://www.google.com/shopping/product/1?q=usb+to+usb+isolator+arduino&amp;prds=apl:11070602780806692536_etc.11070602780806692536_0.psd:11070602780806692536_sas:X&amp;ved=0ahUKEwj6I_qbedl0AhXYgnIEHXaUcDwQ9wGC&amp;ao</a>		
9	LCD Module		Display text and other characters on the LCD screen. It can also be programmed to make "beeping" sounds.	1	NHD-0216K1Z-NSW-FBW-L	*PowerTip PC1602Q B 16 x 2 LCD Display*	\$9.70	<a href="https://www.haw&amp;usa.com/manufacturers/newhaven/displaya/2x16-character-lcd/mhd-0216k1z-bw-fbw-l?gclid=Cj0KCQIAqbyNBhC2ARIsALDwAs4FsSBxGkQ3Z7rZkEhX-4iaM9iNlM_7-OluAQARt93pMbcvdykaAmCBEAL_w_wB">https://www.haw&amp;usa.com/manufacturers/newhaven/displaya/2x16-character-lcd/mhd-0216k1z-bw-fbw-l?gclid=Cj0KCQIAqbyNBhC2ARIsALDwAs4FsSBxGkQ3Z7rZkEhX-4iaM9iNlM_7-OluAQARt93pMbcvdykaAmCBEAL_w_wB</a>		See Appendix E, item # 3.
10	Electric Fan		Spin the blades to blow air. For this competition, the fan is used in conjunction with the dual relay board (15) to extinguish a candle.	1	TD8038H	*Vantec Tomado Double Ball Bearing High Performance Case Fan*	\$16.99	<a href="https://www.eio.com/products/vantec-tb8038h-tornado-double-ball-bearing-high-performance-case-fan?variant=15314358075458&amp;gclid=Cj0KCQIAqbyNBhC2ARIsALDwAs4FsSBxGkQ3Z7rZkEhX-4iaM9iNlM_7-OluAQARt93pMbcvdykaAmCBEAL_w_wB">https://www.eio.com/products/vantec-tb8038h-tornado-double-ball-bearing-high-performance-case-fan?variant=15314358075458&amp;gclid=Cj0KCQIAqbyNBhC2ARIsALDwAs4FsSBxGkQ3Z7rZkEhX-4iaM9iNlM_7-OluAQARt93pMbcvdykaAmCBEAL_w_wB</a>		
11	Color Detection Camera		Can be programmed through MicroPython to track objects, determine colors, and provide live camera feed. In this competition, the camera must be programmed to determine when a red object is present, when a green object is present, and when neither are present.	1	SKU:DFR0517	H7 OpenMV Cam	\$65.00	<a href="https://www.dfrobot.com/product-1648.html?gclid=Cj0KCQIAqbyNBhC2ARIsALDwAs4FsSBxGkQ3Z7rZkEhX-4iaM9iNlM_7-OluAQARt93pMbcvdykaAmCBEAL_w_wB">https://www.dfrobot.com/product-1648.html?gclid=Cj0KCQIAqbyNBhC2ARIsALDwAs4FsSBxGkQ3Z7rZkEhX-4iaM9iNlM_7-OluAQARt93pMbcvdykaAmCBEAL_w_wB</a>		See Appendix E, item # 4.
12	1-Channel Flame Sensor		Detect whether or not a flame is present.	1	N/A	*IR Infrared 4 Wire Flame Detection Sensor Module IR Flame Sensor Module Detector Smartsense*	\$0.70	<a href="https://www.google.com/shopping/product/1?q=SMT+Automatic+Welding+Digital+Analog+Output+1+Channel+Flame+Fire+Sensor+Module&amp;prds=apl:1002760721866293786_etc.1002760721866293786_0.psd:1002760721866293786_sas:X&amp;ved=0ahUKEwio9vTxNL0AHUS3IEH4BDTcQ9pwGC&amp;ao#q=sm">https://www.google.com/shopping/product/1?q=SMT+Automatic+Welding+Digital+Analog+Output+1+Channel+Flame+Fire+Sensor+Module&amp;prds=apl:1002760721866293786_etc.1002760721866293786_0.psd:1002760721866293786_sas:X&amp;ved=0ahUKEwio9vTxNL0AHUS3IEH4BDTcQ9pwGC&amp;ao#q=sm</a>		See Appendix E, item # 5.
13	3-Pin Sonar		Send out a soundwave ping to determine the distance, in inches, from the receptacles to the nearest object.	1	28015	PING Ultrasonic Distance Sensor	\$34.95	<a href="https://www.parallax.com/product/ping-ultrasonic-distance-sensor/">https://www.parallax.com/product/ping-ultrasonic-distance-sensor/</a>		See Appendix E, item # 6.
14	4-Pin Sonar		Send out a soundwave ping to determine the distance, in inches, from the receptacles to the nearest object.	1	HC_SR04	Patton Robotics 4-Pin Sonar	\$3.75	<a href="https://pattonrobotics.com/products/ultrasonic-sensor-and-cables">https://pattonrobotics.com/products/ultrasonic-sensor-and-cables</a>		

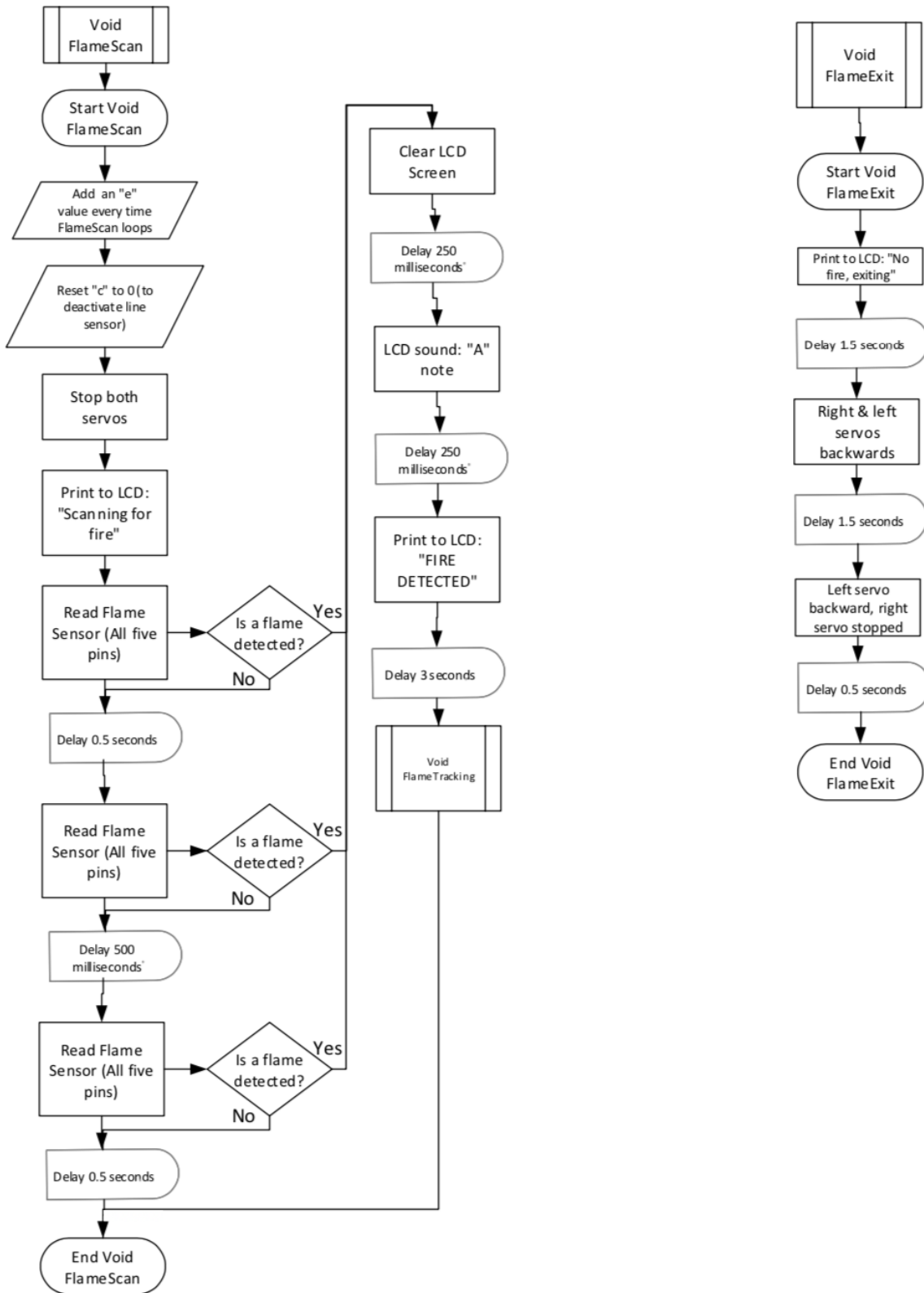
#	Part Name	Part Photo	Function	Quantity	Part Number	Official Name	Price for 1	Link to Purchase	Notes	Tech Sheets?
15	Dual Relay Board		Open or close an attached circuit based on inputs received from the microcontroller. In this competition, it is used to control the electric fan (10).	1	VUPN667	Dual Relay Module for Arduino (D66)	\$7.99	<a href="https://vee.to.net/products/dual-relay-module-for-arduino-d66">https://vee.to.net/products/dual-relay-module-for-arduino-d66</a>		See Appendix E, item # 7.
16	Chassis	N/A	Functions as the base of the robot, housing all components.	1	N/A	Robodessey Mouse Chassis	\$3.95	Link to similar: <a href="https://www.robotshop.com/en/sumo-otto-robot-chassis.html?gclid=Cj0KCQIAcQyNBhC2ARIsALDwAsAFWYFRIP2DjkP2X150m50IPjG1bSEh76jWl-7F8M7BGkth2PkMaAgWjEALw_wcB">https://www.robotshop.com/en/sumo-otto-robot-chassis.html?gclid=Cj0KCQIAcQyNBhC2ARIsALDwAsAFWYFRIP2DjkP2X150m50IPjG1bSEh76jWl-7F8M7BGkth2PkMaAgWjEALw_wcB</a>		
17	Servo Motor	N/A	With positional and directional awareness, the servo motors can be controlled to spin at a specified speed and direction. In this competition, they are used to drive the wheels of the robot to provide physical mobility.	2	Similar product: FSS103R	Similar product: "Continuous Rotation Servo" by Fee Tech	\$11.95	Link to similar: <a href="https://www.adafruit.com/product/1542?gclid=Cj0KCQIAcQyNBhC2ARIsALDwAsCbFvBGGY-uTnlmexxaE1ZqdWbxyYURXBFL6iWXB66Xe5KGf4m7YwEaA5pEALw_wcB">https://www.adafruit.com/product/1542?gclid=Cj0KCQIAcQyNBhC2ARIsALDwAsCbFvBGGY-uTnlmexxaE1ZqdWbxyYURXBFL6iWXB66Xe5KGf4m7YwEaA5pEALw_wcB</a>	Each servo receptacle is attached to included standard generic skateboard wheels in order to drive the robot.	See Appendix E, item # 8.
18	Microcontroller		Allows storage for C++ code, programmed through the Teensyduino PC program.	1		"Teensy 3.2 USB Microcontroller Development Board"	\$19.80	<a href="https://www.robotshop.com/en/teensy-32-usb-microcontroller-development-board.html?gclid=Cj0KCQIAcQyNBhC2ARIsALDwAsBSaTJdqbaNjA6Pb2yayv6R3HiEakA2bxv-bA4bjU5ovbERtuwAueZfEALw_wcB">https://www.robotshop.com/en/teensy-32-usb-microcontroller-development-board.html?gclid=Cj0KCQIAcQyNBhC2ARIsALDwAsBSaTJdqbaNjA6Pb2yayv6R3HiEakA2bxv-bA4bjU5ovbERtuwAueZfEALw_wcB</a>		See Appendix E, item # 9.
19	Motherboard		It allows all components to communicate with the microcontroller (18), provides power switches for both the analog and digital sides, and houses a micro USB port to allow the microcontroller to connect with a PC.	1	PRT3	"Patton Robotics Carrier Board"	\$18.00	<a href="https://pattonrobotics.com/products/patton-robotics-teensy-motherboard-kit">https://pattonrobotics.com/products/patton-robotics-teensy-motherboard-kit</a>		
Not Pictured										
20	Trailer Wheel	N/A	Attached at the rear on the bottom of the chassis (16), it keeps the chassis stable and level by providing a third wheel.	1	N/A	N/A	N/A	N/A		
21	Aluminum Bars	N/A	Manufactured in the metals lab, these bars were adhered to the bottom of the fan (10) and to the top of the chassis (16) such that the fan was elevated to the height of the candle used in the competition.	4 (3" by 0.25" by 0.25" aluminum bars)	N/A	N/A	N/A	N/A	This item is custom manufactured.	
22	Duracell 9V Battery		Power the electric fan (10).	1	N/A	Duracell Coppertop 9V Batteries - 2pk Alkaline Battery	\$7.99	<a href="https://www.target.com/p/duracell-coppertop-9v-batteries-2pk-alkaline-battery/-/A-10292372?ref=tbl_adv_XS000000&amp;FID=google_e_pla_d&amp;fndarc=tmv&amp;DFA=7170000083356537&amp;CPNG=PLA_DVM%2B065800000XZnDQC-Duracell_AD_GoogleSearch_Coppertop_2021_Flight&amp;adgroup=PLA_Duracell=Coppertop&amp;ID=700000001393753pgs&amp;network=g&amp;device=c&amp;location=1025089&amp;gclid=Cj0KCQIAcQyNBhC2ARIsALDwAsA2RqP2mSpC1QAMiO8vnpW1a0M-VUVK29zcvSroAKopJhPNYkAwxAjMEALw_wcB&amp;gclid=awda">https://www.target.com/p/duracell-coppertop-9v-batteries-2pk-alkaline-battery/-/A-10292372?ref=tbl_adv_XS000000&amp;FID=google_e_pla_d&amp;fndarc=tmv&amp;DFA=7170000083356537&amp;CPNG=PLA_DVM%2B065800000XZnDQC-Duracell_AD_GoogleSearch_Coppertop_2021_Flight&amp;adgroup=PLA_Duracell=Coppertop&amp;ID=700000001393753pgs&amp;network=g&amp;device=c&amp;location=1025089&amp;gclid=Cj0KCQIAcQyNBhC2ARIsALDwAsA2RqP2mSpC1QAMiO8vnpW1a0M-VUVK29zcvSroAKopJhPNYkAwxAjMEALw_wcB&amp;gclid=awda</a>	2 batteries are needed, but because two are sold together in the same pack, the quantity is listed as 1.	
23	Various copper jumper cables and wires	N/A	Connect all components to the motherboard (19).	N/A	N/A	Similar product: "Dupont Jumper wire 10CM 20CM 30CM Male to Male + Female to Male + Female to Female Jumper Wire Dupont Cable for arduino DIY KIT"	Price of similar: \$4.04	<a href="http://www.officematics.com/dupont-jumper-wire-10cm-20cm-30cm-male-to-male-female-to-male-female-to-female-jumper-wire-dupont-cable-for-arduino-diy-kit-p-141201.html">http://www.officematics.com/dupont-jumper-wire-10cm-20cm-30cm-male-to-male-female-to-male-female-to-female-jumper-wire-dupont-cable-for-arduino-diy-kit-p-141201.html</a>		
24	Various screws, nuts, bolts, screwdriver, allen wrenches, and other metal connectors	N/A	Connect all components to the chassis (16).	N/A	N/A	N/A	N/A	N/A		
25	Adhesives	N/A	Masking tape, electrical Tape, and zip ties were used as adhesives. These adhesives were used to attach the electric fan (10) to the aluminum bars and bundle the wires together.	N/A	N/A	N/A	N/A	N/A		
26	Hot Glue Gun (with glue)	N/A	Adhere the aluminum bars to the chassis (16).	1	N/A	Similar product: "Gluernous Mini Hot Glue Gun with 30 Glue Sticks for Crafts School DIY Arts Home Quick Repairs, 20W, Blue"	Price of similar: \$12.79	<a href="https://www.amazon.com/Gluernous-Sticks-Crafts-School-Repairs/dp/B08FTHWC94">https://www.amazon.com/Gluernous-Sticks-Crafts-School-Repairs/dp/B08FTHWC94</a>		
27	Push button, normally open	N/A	Function as the "start" button to activate the robot's main code routine.	1	N/A	Similar product: "Twidec/10Pcs 1A 250V AC 2 Pins SPST Black + Red Normal Open Mini Momentary Push Button Switch with Pre-soldered Wires PBS-110-XBKRF"	Price of similar: \$10.99	<a href="https://www.amazon.com/Twidec-Normal-Momentary-Pre-soldered-PBS-110-XBKRF/dp/B07RW2YZ4W/ref=asc_df_B07RW2YZ4W/?tag=&amp;linkCode=df0&amp;hvadid=366288746572&amp;hvpos=&amp;hmnst=g&amp;hvrnd=68338398944147103668&amp;hvpon=&amp;hvpw=&amp;hvqm=&amp;hvdv=&amp;hvcmd=&amp;hvcnt=&amp;hvcsh=1025089&amp;hvtar:pid=819692632745&amp;m=&amp;adgrpid=81879716528&amp;th=1">https://www.amazon.com/Twidec-Normal-Momentary-Pre-soldered-PBS-110-XBKRF/dp/B07RW2YZ4W/ref=asc_df_B07RW2YZ4W/?tag=&amp;linkCode=df0&amp;hvadid=366288746572&amp;hvpos=&amp;hmnst=g&amp;hvrnd=68338398944147103668&amp;hvpon=&amp;hvpw=&amp;hvqm=&amp;hvdv=&amp;hvcmd=&amp;hvcnt=&amp;hvcsh=1025089&amp;hvtar:pid=819692632745&amp;m=&amp;adgrpid=81879716528&amp;th=1</a>	Only 1 of these components is necessary for this project. However, the only online listing sells this component in a 10-pack with other similar components.	

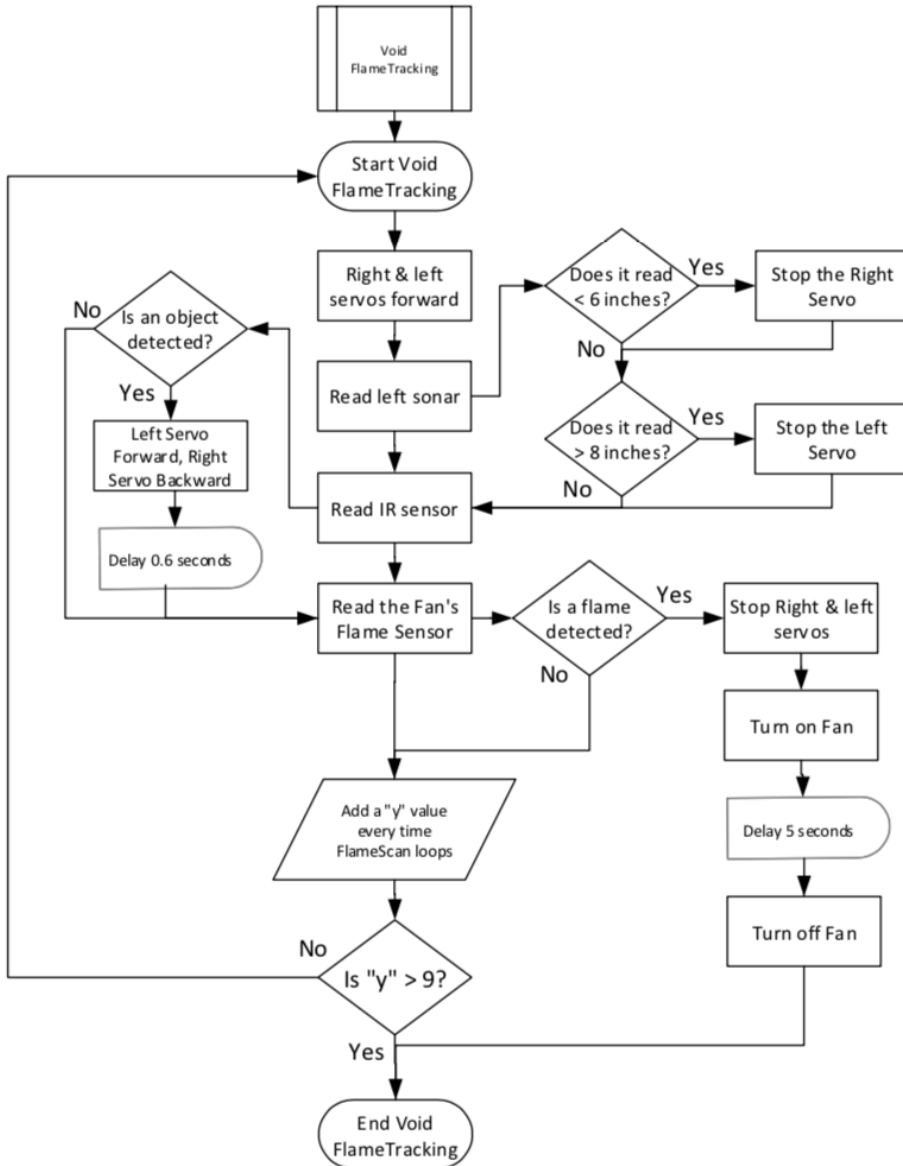
**Appendix B: Process Flow Chart/Algorithm**

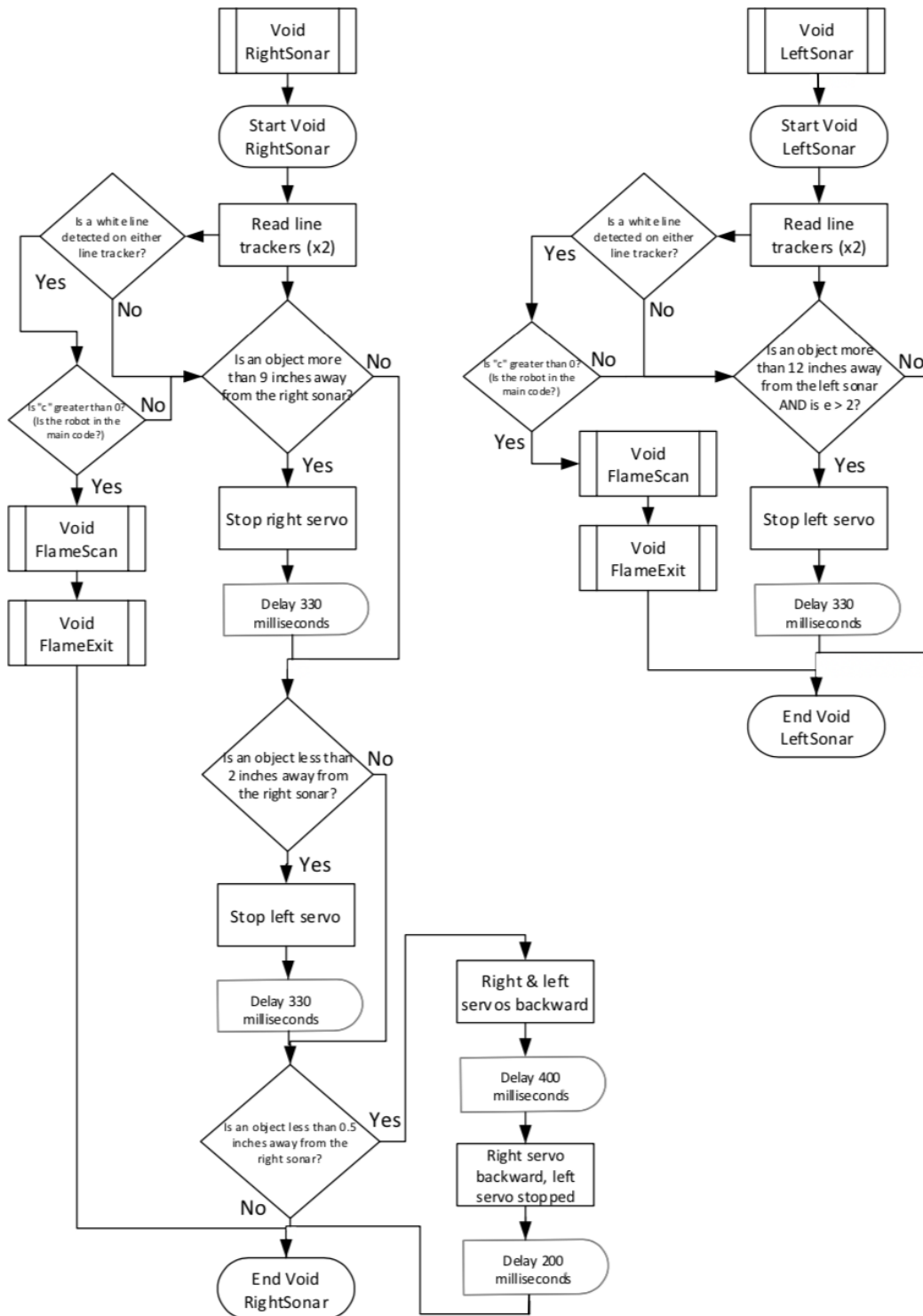


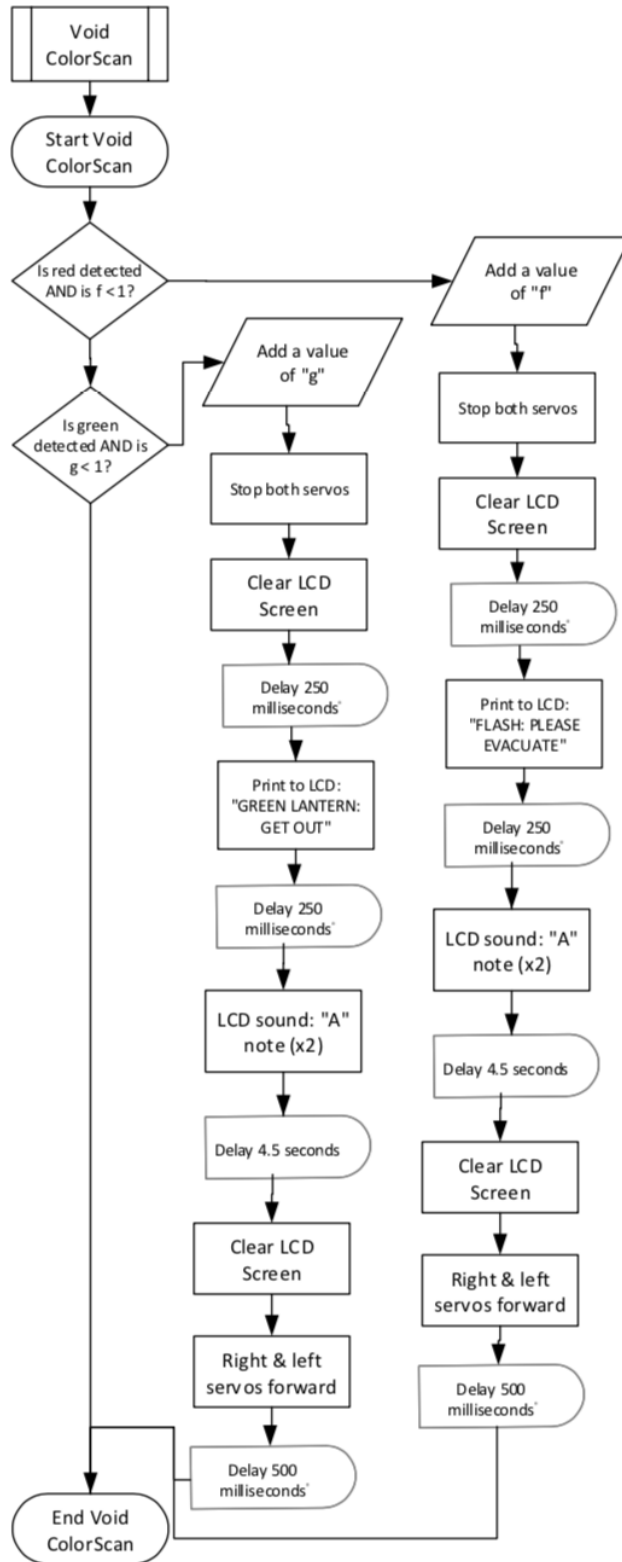














## Appendix C: Complete Programming Code with Comments

```

#include <HCSR04.h> // library for right (4-pin) sonar
#include <Ping.h> // library for left (3-pin w/ LED) sonar

HCSR04 SonarR(A4,A5); // Right sonar is in pin A4 and A5
byte ServoL = 3; // left wheel is pin 3
byte ServoR = 5; // right wheel is pin 5
Ping ping = Ping(A3); // Left sonar is pin A3

int SharpIR = A9; //SharpIR input is A9
int valIR = 0; //set initial reading to 0

#include<SoftwareSerial.h> // Library for LCD
const int rxPin=7; // LCD is pin 7

const int txPin=A1; //LCD is in pin A1. Messages are below:
const char message1[]="Scanning for fire...";
const char message2[]="FIRE DETECTED!!!";
const char message3[]="No fire. Exiting.";
const char message4[]="FLASH: PLEASE EVACUATE!";
const char message5[]="GREEN LANTERN: GET OUT!";
const char message6[]="EXTINGUISHING FLAME";
const char message7[]="Color Scanning";
const char message8[]="Exploring now";
const char message9[]="Hooray!!!";
SoftwareSerial mySerial= SoftwareSerial(rxPin, txPin); // LCD Setup

int LineTrackS1 = A6; // Line Tracker 1 is pin A6
int valLine1 = LOW; // Set initial value to "low"
int LineTrackS2 = A7; // Line Tracker 2 is pin A7
int valLine2 = LOW; // Set initial value to "low"

// 5-Pin Flame Sensor Setup
int FlameS1 = 11; // East pin is pin 11
int valFlame1 = 0; // Set initial value to 0
int FlameS2 = 10; // Northeast pin is pin 10
int valFlame2 = 0; // Set initial value to 0
int FlameS3 = 9; // North pin is pin 9
int valFlame3 = 0; // Set initial value to 0
int FlameS4 = 8; // Northwest pin is pin 8
int valFlame4 = 0; // Set initial value to 0
int FlameS5 = 7; // West pin is pin 7
int valFlame5 = 0; // Set initial value to 0

int FlameS0 = 2; // 1-Pin Flame Sensor Setup
int valFlame0 = 0; // Set initial value to 0

int CamR = 0; // Color Camera (RED value) is connected to pin 0
int valColorR = 0; // Set initial value to 0

int CamG = 1; // Color Camera (GREEN value) is connected to pin 1
int valColorG = 0; // Set initial value to 0

```

```

int Relay = 6; // Relay (fan) input is in pin 6.
.
int StartButton = 12; // Start button is in pin 12.
int valStartButton = 0; // Set initial value to 0

int f = 0; // This integer is used for Color Camera (RED). Once the robot sees the Flash once, it will stop scanning for red.
int g = 0; // This integer is used for Color Camera (GREEN). Once the robot sees Green Lantern once, it will stop scanning for green.
int e = 0; // This integer is used for the line sensor. After the robot sees the first three rooms, the left sonar will turn on so it can enter the fourth room.
int c = 0; // This integer is used in the flamescan and flameexit code. The line sensor will only scan when the robot is in the drive code.

void setup() {
  pinMode(ServoR, OUTPUT); // set right wheel servos as output
  pinMode(ServoL, OUTPUT); // set left wheel servos as output
  pinMode(SharpIR, INPUT); // Sharp IR input
  pinMode(13, OUTPUT); // Flame sensor output
  pinMode(LineTrackS1, INPUT); // Line Tracker 1 Input
  pinMode(LineTrackS2, INPUT); // Line Tracker 2 Input
  pinMode(FlameS1, INPUT); // East flame sensor input
  pinMode(FlameS2, INPUT); // Northeast flame sensor input
  pinMode(FlameS3, INPUT); // North flame sensor input
  pinMode(FlameS4, INPUT); // Northwest flame sensor input
  pinMode(FlameS5, INPUT); // West flame sensor input
  pinMode(FlameS0, INPUT); // 1-Pin flame sensor input
  pinMode(CamR, INPUT); // Color Camera (RED) Input
  pinMode(CamG, INPUT); // Color Camera (GREEN) Input
  pinMode(StartButton, INPUT); // Start Button Input
  pinMode(Relay, OUTPUT); // Relay (fan) setup
  Serial.begin(9600); // Set serial rate
  pinMode(txPin, OUTPUT); // LCD setup
  digitalWrite(txPin, LOW);
  mySerial.begin(9600); // Set serial rate for LCD
  mySerial.write(12); // Clear Screen
  delay(1000); // delay 1 second
}

void loop() {
  valStartButton = digitalRead(StartButton); // Read start button
  Serial.println(valStartButton); // print value to serial
  Serial.println("Waiting for button press"); // print message to serial
  if (valStartButton > 0) { // If the start button is pressed, the robot will turn left to scan for red, turn right to scan for green, turn left a bit to straighten out and scan both colors again, then drive forward.
    Serial.println("Here we go!"); // Print message to serial
    delay(150);

    mySerial.print(message7); // Print message to LCD: "Color Scanning"
    delay(50);
    analogWrite(ServoR, 150); // Turn left (right servo forward and left servo backward)
    analogWrite(ServoL, 150);
    delay(300);
    analogWrite(ServoR, 0); // Stop both servos
    analogWrite(ServoL, 0);
    valColorR=digitalRead(CamR); // Red Color Camera Read
    Serial.print("RED = ");
    Serial.println(valColorR); // Color Camera print to serial monitor
    valColorG=digitalRead(CamG); // Green Color Camera Read
    Serial.print("GREEN = ");
    Serial.println(valColorG); // Color Camera print to serial monitor
    ColorScan(); // Color Scan Subroutine
    delay(500);
  }
}

```



```

analogWrite(ServoR, 255); // turn right (right servo backward and left servo forward)
analogWrite(ServoL, 255);
delay(300);
analogWrite(ServoR, 0);
analogWrite(ServoL, 0);
valColorR=digitalRead(CamR);
Serial.print("RED = ");
Serial.println(valColorR);
ColorScan();
delay(500);
valColorG=digitalRead(CamG);
Serial.print("GREEN = ");
Serial.println(valColorG);
ColorScan();
delay(500);

mySerial.print(message8); // Print to LCD: "Exploring Now"
analogWrite(ServoR, 150); // turn left a bit
analogWrite(ServoL, 150);
delay(150);
analogWrite(ServoR, 0);
analogWrite(ServoL, 0);
valColorR=digitalRead(CamR);
Serial.print("RED = ");
Serial.println(valColorR);
valColorG=digitalRead(CamG);
Serial.print("GREEN = ");
Serial.println(valColorG);
ColorScan();
delay(500);

analogWrite(ServoR, 150); // Drive forward (Both wheels forward)
analogWrite(ServoL, 255);
delay(2000); // This delay allows the bot to avoid accidentally reading the red and green dots by the starting line.
Main(); // Main Program subroutine
}
}

void Main() {
for (int z = 0; z < 900000; z++){ // The main program loops virtually indefinitely
c++; // Every time the robot is in the main code, the "c" value increases past 0, allowing the robot to scan the line sensor
(becomes relevant later).

analogWrite(ServoR, 150);
analogWrite(ServoL, 255);
LineSensor(); // Line Sensor rapid scan subroutine
mySerial.write(12);
valLine1=digitalRead(LineTrackS1); // Line Sensors read
valLine2=digitalRead(LineTrackS2);
SonarR.ReadEchoInches(); // Right sonar reading and print to serial
Serial.print("Right Inches = ");
Serial.println(SonarR.Inches);
RightSonar(); // Right Sonar response subroutine

ping.fire(); // left sonar reading and print to serial
Serial.print(ping.inches());
Serial.print("Left Inches = ");
Serial.println();
LeftSonar(); // Left sonar response subroutine
LineSensor();

```

```

valColorR=digitalRead(CamR); // Color camera readings and responses
Serial.print("RED = ");
Serial.println(valColorR);
valColorG=digitalRead(CamG);
Serial.print("GREEN = ");
Serial.println(valColorG);
ColorScan();

valIR = digitalRead(SharpIR); //Sharp IR reading and print to serial
Serial.print("IR = ");
Serial.println(valIR);
LineSensor();
if (valIR > 0){ // If the sharp IR detects an object, the robot will back up and turn left.
  Serial.print("backing up and turning left");
  analogWrite(ServoR, 255); // Left and right servos backwards
  analogWrite(ServoL, 150);
  delay(400);
  analogWrite(ServoR, 150); // Right servo backwards, left servo stopped
  delay(200);
}
if (valLine1 > 0 || valLine2 > 0){ // If the robot sees a white line, it will stop and begin the flamescan and then the flameexit
subroutines.
  if (c > 0){ // It will only scan for a line once it is in the main program.
    FlameScan(); // Flamescan subroutine
    FlameExit(); // Flameexit subroutine
  }
}
}
}
}

void FlameScan() { // Once the line sensor detects a line, the robot will stop on the line and scan for a flame three times.
  analogWrite(ServoR, 0);
  analogWrite(ServoL, 0);
  Serial.println("Room found"); // Print message to serial
  int c = 0; // Reset "c" to zero so the robot stops scanning for a line.
  e++; // This counter adds one value for every room the robot enters. This counter enables new code after the robot exits the
third room.
  delay(250);
  mySerial.write(12);
  delay(250);
  mySerial.print(message1); // Print to LCD: scanning for fire
  Serial.println("Scanning for fire");
  delay(2000);
  Serial.println("_____"); // Print a line between values for visual organization on the serial
  valFlame1=digitalRead(FlameS1); // Flame sensors (All five, East through West) read value and print to serial
  Serial.print("East: ");
  Serial.println(valFlame1);
  valFlame2=digitalRead(FlameS2);
  Serial.print("NorthEast: ");
  Serial.println(valFlame2);
  valFlame3=digitalRead(FlameS3);
  Serial.print("North: ");
  Serial.println(valFlame3);
  valFlame4=digitalRead(FlameS4);
  Serial.print("NorthWest: ");
  Serial.println(valFlame4);
  valFlame5=digitalRead(FlameS5);
  Serial.print("West: ");
  Serial.println(valFlame5);
  delay(500);
  Serial.println("_____");
  valFlame1=digitalRead(FlameS1);

```

```

Serial.print("East: ");
Serial.println(valFlame1);
valFlame2=digitalRead(FlameS2);
Serial.print("NorthEast: ");
Serial.println(valFlame2);
valFlame3=digitalRead(FlameS3);
Serial.print("North: ");
Serial.println(valFlame3);
valFlame4=digitalRead(FlameS4);
Serial.print("NorthWest: ");
Serial.println(valFlame4);
valFlame5=digitalRead(FlameS5);
Serial.print("West: ");
Serial.println(valFlame5);
delay(500);
Serial.println("_____");
valFlame1=digitalRead(FlameS1);
Serial.print("East: ");
Serial.println(valFlame1);
valFlame2=digitalRead(FlameS2);
Serial.print("NorthEast: ");
Serial.println(valFlame2);
valFlame3=digitalRead(FlameS3);
Serial.print("North: ");
Serial.println(valFlame3);
valFlame4=digitalRead(FlameS4);
Serial.print("NorthWest: ");
Serial.println(valFlame4);
valFlame5=digitalRead(FlameS5);
Serial.print("West: ");
Serial.println(valFlame5);
delay(500);

if (valFlame1 > 0){ // If a flame is detected on the East pin, the LCD will print a message, then the robot will enter the
FlameTracking subroutine.
  mySerial.write(12);
  delay(250);
  mySerial.write(220); // LCD "A" note sound
  delay(250);
  mySerial.print(message2); // Print "fire detected" to LCD
  Serial.println("FIRE DETECTED");
  digitalWrite(13, HIGH); // light up pin 13
  delay(3000);
  FlameTracking(); // FlameTracking subroutine
}
if (valFlame2 > 0){ // If a flame is detected on the NorthEast pin, the LCD will print a message, then the robot will enter the
FlameTracking subroutine.
  mySerial.write(12);
  delay(250);
  mySerial.write(220);
  delay(250);
  mySerial.print(message2);
  Serial.println("FIRE DETECTED");
  digitalWrite(13, HIGH);
  delay(3000);
  FlameTracking();
}
if (valFlame3 > 0){ // If a flame is detected on the North pin, the LCD will print a message, then the robot will enter the
FlameTracking subroutine.
  mySerial.write(12);
  delay(250);
  mySerial.write(220);

```

```

    delay(250);
    mySerial.print(message2);
    Serial.println("FIRE DETECTED");
    digitalWrite(13, HIGH);
    delay(3000);
    FlameTracking();
}
if (valFlame4 > 0){ // If a flame is detected on the NorthWest pin, the LCD will print a message, then the robot will enter the
FlameTracking subroutine.
    mySerial.write(12);
    delay(250);
    mySerial.write(220);
    delay(250);
    mySerial.print(message2);
    Serial.println("FIRE DETECTED");
    digitalWrite(13, HIGH);
    delay(3000);
    FlameTracking();
}
if (valFlame5 > 0){ // If a flame is detected on the West pin, the LCD will print a message, then the robot will enter the
FlameTracking subroutine.
    mySerial.write(12);
    delay(250);
    mySerial.write(220);
    delay(250);
    mySerial.print(message2);
    Serial.println("FIRE DETECTED");
    digitalWrite(13, HIGH);
    delay(3000);
    FlameTracking();
}
}

void RightSonar() { // Responses to right sonar readings subroutine
    valLine1=digitalRead(LineTrackS1); // Line Sensors read
    valLine2=digitalRead(LineTrackS2);

    if (SonarR.Inches > 9) { // If the wall is more than 9 inches from the right sonar, the right wheel will stop so the robot can
    approach the empty corridor.
        analogWrite(ServoR, 0);
        analogWrite(ServoL, 255);
        Serial.println("Turning Right");
        delay(330); // Delay 330 milliseconds
        valLine1=digitalRead(LineTrackS1);
        valLine2=digitalRead(LineTrackS2);
        if (valLine1 > 0 || valLine2 > 0){ // If the robot sees a white line, it will stop and begin the flamescan and then the flameexit
        subroutines.
            if (c > 0){ // It will only scan for a line once it is in the main program.
                FlameScan();
                FlameExit();
            }
        }
    }
}
if (SonarR.Inches < 2) { // If the wall is less than 2 inches from the sonar, the left wheel will stop so the robot steers left away
from the wall.
    analogWrite(ServoR, 150);
    analogWrite(ServoL, 0);
    Serial.println("Approaching Right Wall");
    delay(330); // Delay 330 milliseconds
    valLine1=digitalRead(LineTrackS1);
    valLine2=digitalRead(LineTrackS2);
    if (valLine1 > 0 || valLine2 > 0){

```

```

    if (c > 0){
        FlameScan();
        FlameExit();
    }
}
}
if (SonarR.Inches < 0.25) { // If the wall is less than 0.25 inches from the right sonar, the robot will back up and turn left a bit
to re-align itself in the hallway.
    Serial.print("backing up and turning left");
    analogWrite(ServoR, 255);
    analogWrite(ServoL, 150);
    delay(400);
    analogWrite(ServoR, 150);
    delay(200);
    valLine1=digitalRead(LineTrackS1);
    valLine2=digitalRead(LineTrackS2);
    if (valLine1 > 0 || valLine2 > 0){
        if (c > 0){
            FlameScan();
            FlameExit();
        }
    }
}
else // If none of the above conditions are true, the robot will resume driving forward with both servos.
    analogWrite(ServoR, 150);
    analogWrite(ServoL, 255);
}

void LeftSonar() { // Responses to left sonar readings subroutine
    if (ping.inches() > 12 && e > 2) { // If the wall is more than 12 inches away from the left sonar, the left servo will stop so the
robot approaches the empty corridor to the left.
        // NOTE: This subroutine will not activate until the robot has already scanned a line three times ("e" value is greater than 2)

        analogWrite(ServoL, 0);
        analogWrite(ServoR, 150);
        valLine1=digitalRead(LineTrackS1);
        valLine2=digitalRead(LineTrackS2);
        delay(330);
        valLine1=digitalRead(LineTrackS1);
        valLine2=digitalRead(LineTrackS2);
        if (valLine1 > 0 || valLine2 > 0){
            if (c > 0){
                FlameScan();
                FlameExit();
            }
        }
    }
}

void FlameExit() {
    int c = 0; // The "c" value is set to 0 so the robot cannot scan for a line until it enters the main program.
    if (e < 3){ // If robot has scanned less than three lines, it will back up and turn right.
        mySerial.write(12);
        delay(250);
        mySerial.print(message3); // Print message to LCD: "No fire, exiting room"
        delay(1500);
        analogWrite(ServoR, 255);
        analogWrite(ServoL, 150);
        delay(2000);
        analogWrite(ServoL, 255);
        analogWrite(ServoR, 255);
        delay(500);
    }
}

```

```

    mySerial.write(12);
  }
  if (e > 2) { // After the robot has scanned more than two lines, the robot will instead back up and turn left toward the last room.
    mySerial.write(12);
    delay(250);
    mySerial.print(message3);
    delay(1500);
    analogWrite(ServoR, 255);
    analogWrite(ServoL, 150);
    delay(2000);
    analogWrite(ServoR, 150);
    analogWrite(ServoL, 150);
    delay(400);
    mySerial.write(12);
  }
}

```

void FlameTracking() { // Once the robot has detected a candle in the room, it will enter a room and drive in line with the left wall. Once a flame is detected next to the fan, it will turn on the fan to extinguish it.

```

  for(int x = 0; x < 900000; x++){ // Repeat FlameTracking virtually indefinitely
    Serial.println("Tracking Flame");
    analogWrite(ServoR, 150);
    analogWrite(ServoL, 255);
    valIR = digitalRead(SharpIR);

    ping.fire();
    Serial.print("Left Inches = ");
    Serial.print(ping.inches());
    Serial.println();

    valIR = digitalRead(SharpIR);
    Serial.print("IR = ");
    Serial.println(valIR);

    valFlame0 = digitalRead(FlameS0); // Reading the 1-pin flame sensor (by the fan) and print value to serial
    Serial.print("Flame by Fan = ");
    Serial.println(valFlame0);
    Serial.println("_____"); // Print a line between readings for serial visual clarity

    valIR = digitalRead(SharpIR);

    if (ping.inches() < 9) { // If the wall is less than 9 inches from the left sonar, the right servo will stop so the robot can steer away from the left wall.
      Serial.println("Evading wall");
      analogWrite(ServoR, 0);
      analogWrite(ServoL, 255);
      delay(450);
      if (valIR > 0) {
        Serial.println("Turning right");
        analogWrite(ServoR, 255);
        analogWrite(ServoL, 255);
        delay(500);
      }
    }
  }
  if (ping.inches() > 11) { // If the wall is more than 11 inches from the left sonar, the left servo will stop so the robot approaches the left wall.
    Serial.println("Approaching wall");
    analogWrite(ServoR, 150);
    analogWrite(ServoL, 0);
    delay(50);
    if (valIR > 0) {
      Serial.println("Turning right");
    }
  }
}

```

```

    analogWrite(ServoR, 255);
    analogWrite(ServoL, 255);
    delay(500);
  }
}
if (valIR > 0){ // If the sharp IR detects an object, the robot will turn right.
  analogWrite(ServoR, 255);
  analogWrite(ServoL, 255);
  delay(500);
}
if (valFlame0 < 1){ // If a flame is detected next to the fan, the robot will turn left a bit, then the fan will turn on and the robot
will stop.
  Serial.println("Extinguishing");
  analogWrite(ServoR, 150);
  analogWrite(ServoL, 150);
  delay(50);
  analogWrite(ServoR, 0);
  analogWrite(ServoL, 0);
  digitalWrite(Relay, HIGH); // Turn on fan
  delay(250);
  mySerial.print(message2); // message on "Extinguishing" on LCD
  delay(7000);
  digitalWrite(Relay, LOW); // Turn off fan
  mySerial.write(12);
  delay(250);
  mySerial.print(message9); // Print "hooray" to LCD
  delay(3000);
}
}
}
void LineSensor() { // Line sensor rapid scan subroutine
  for (int i = 0; i < 51; i++){ // With no delays, repeat the line scan 50 times.
    valLine1=digitalRead(LineTrackS1);
    valLine2=digitalRead(LineTrackS2);
  }
}
void ColorScan() { // Subroutine for responses to red and green color camera readings
  if (valColorR > 0 && f < 1){ // If the camera reads "RED" and has not already read "RED" before, the robot will stop and print
a message to serial.
    f++; // add a value to red color count to prevent the camera from responding to red again
    analogWrite(ServoR, 0);
    analogWrite(ServoL, 0);
    mySerial.write(12);
    delay(250);
    mySerial.print(message4); // Print message to LCD: "FLASH EVACUATE"
    delay(250);
    mySerial.write(220); // LCD "A" note sound twice
    mySerial.write(220);
    Serial.println("FLASH: PLEASE EVACUATE");
    delay(4500);
    mySerial.write(12);
    analogWrite(ServoR, 150);
    analogWrite(ServoL, 255);
    delay(500);
  }
  if (valColorG > 0 && g < 1){ // If the camera reads "GREEN" and has not already read "GREEN" before, the robot will stop
and print a message to serial.
    g++; // add a value to green color count to prevent the robot from responding to green again
    analogWrite(ServoR, 0);
    analogWrite(ServoL, 0);
    mySerial.write(12);
    delay(250);
  }
}

```

```
mySerial.print(message5); // Print message to LCD: "GREEN LANTERN: GET OUT"  
delay(250);  
mySerial.write(220);  
mySerial.write(220);  
Serial.println("GREEN LANTERN: PLEASE EVACUATE");  
delay(4500);  
mySerial.write(12);  
analogWrite(ServoR, 150);  
analogWrite(ServoL, 255);  
delay(500);  
}  
}
```



## Appendix D: Fire Contest Rules Official Document

**Firefighting Mobile Robot Contest (R&D Project)\***  
**ITEC 467, Mobile Robotics**  
**Dr. John Wright**  
**Department of Applied Engineering, Safety & Technology**  
**Millersville University**

**\*Contest and Rules Adapted and/or cited from the 2007 Trinity College Home Firefighting Robot Contest**

### 1. CONTEST OBJECTIVE

The main challenge of this contest is to build an autonomous robot using a *Teensy* Microcontroller that can find its way through an arena that represents a model house, find a lit candle that represents a fire in the house, and extinguish the fire in the shortest time. This task simulates the real-world operation of an autonomous robot performing a fire protection function in a real house. The goal of the contest is to advance robot technology and knowledge while using robotics as an educational tool. The contest shall also teach students about the limitations of technology, and how to deal with complex control problems and situations.

### 2. DIMENSIONS AND SPECIFICATIONS

The goal of the contest is to make a robot that can operate successfully in the real world, not just in the laboratory. Such a robot must be able to operate successfully where there is uncertainty and imprecision. Therefore, the dimensions and specifications listed in the rules are not exactly what will be encountered at the contest and they are provided as general aids. Contestants may take measurements on the contest maze located in Osburn Hall.

However, the size limits on robots are absolute and will be enforced by the Contest Judge.

### 3. THE HOUSE FLOOR PLAN STRUCTURE AND FEATURES

The arena/maze represents a home, a more realistic fire-fighting environment.

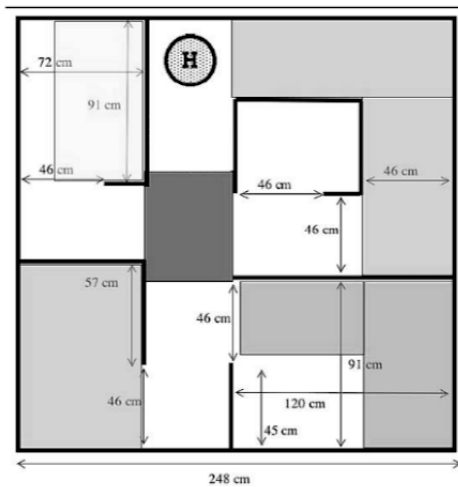
Rugs will be placed in some or all of the rooms and hallways. There will be no shag rugs. Wall hangings including mirrors may be hung from the walls of rooms and hallways. The floor will be painted black and the walls will be white. The condition of these surfaces may vary due to age of the maze and use and therefore are not guaranteed to be "pure" black or white. You must design your robots to deal with this potential imperfection.

A mirror will not be placed in the room where the candle is located but may be placed anywhere else in the arena.

Two ~5cm diameter circles will appear on the floor marking the hallway between all four rooms, and in the two adjoining hallways next to the Home Circle location. One will be Red in color and the other will be green in color. They will be placed ~14cm apart (center to center measurement). Two home occupant (The Green Lantern or The Flash) will exist/stand on one of the like-colored Red/Green circles. During each of the three maze attempts, a location will be randomly selected (one of three) along with occupants (The Green Lantern or The Flash) – they will not stand next to each other, however during any round. You must not touch an occupant with any part of your robot or it will result in a disqualification for that run. Should your robot be able to accurately identify the occupant as The Green Lantern or The Flash and advise them to leave the building using an LCD module, the occupant will be removed from the maze.

All hallways and doorways to room will be approximately 46 cm wide. There will not be a door in the doorways, just a 46 cm opening. There will be a white tape (~width of the walls), stuck to the floor across each doorway to indicate room entrances. The carpeting will not cover up the white tape. However the floor coverings may be light in color.

The robot will start at the Home Circle location. The Home Circle will be a solid white circle on the floor (No H labeled on actual circle). The ~20 cm diameter white Home Circle will be roughly centered in the hallway and may not be taped down/secured to the maze floor. The robot must start within the Home Circle (position determined by the Contest Judge), but once started, it can go in any direction desired.



#### 4. AMBIENT LIGHTING

Part of the challenge of the contest is to make a robot that can operate in real world situations and that includes inconsistent lighting, shadows, glare, etc. Many sensors used by fire-fighting robots are thrown off by stray light sources including IR sources and UV sources present. Participants are urged to devise shades, covers, and other means to reduce the effects of stray sources.

#### 5. ROBOT OPERATION

Once turned on, the robot must be autonomous without any human intervention. That is, they are to be computer controlled and not manually controlled devices.

A robot may bump into or touch the walls of the arena as it travels, but it cannot mark, dislodge or damage the walls in doing so. There will not be a penalty for touching a wall, but there is a penalty for moving along the wall while in contact with it for more than 3 continuous seconds. Should more than 3 seconds of continuous contact occur, the robot will be disqualified for that run.

The robot cannot leave anything behind as it travels through the arena. It cannot make any marks on the floor of the arena that aid in navigation as it travels. Any robot that deliberately, in the judges' opinion, damages the contest arena (including the walls) will be disqualified. This does not include any accidental marks or scratches made in moving around.

The robot must show the Contest Judge by using the provided LCD module/screen that it has found the candle before it attempts to put it out (i.e. "Candle Found" / "Scanning for Candle" or similar messages). For example, the robot cannot just flood the arena structure with CO<sub>2</sub> thereby putting the candle out by accident. Failure to use the LCD module properly will result in a disqualification for that run. It is the responsibility of the teams to ensure accurate use of the LCD to the Contest Judge.

#### 6. PUTTING OUT THE CANDLE

The robot must not use any destructive or dangerous methods to put out the candle. It may use such substances as water, air, CO<sub>2</sub>, etc., but any method or material that is dangerous or will damage the arena is prohibited. Halon is not allowed because it is harmful to the environment.

It will be permissible to put out the candle by blowing air or other oxygen-bearing gas (pure oxygen is not allowed). However, this is not a practical method of extinguishing a fire in the real world. So, robots that do not use air streams to blow out the candle will receive a 30% Extinguishing Time Reduction (ETR).

A robot is not allowed to knock over the candle. Should this occur, a robot would be disqualified for that run. Touching the candle is permitted, yet is not recommended.

The robot must come within 30 cm of the candle before it attempts to extinguish the flame. There will be three provided circle segments of different sizes that will be provided to the 467 teams at the onset of the semester. These segments may be of different sizes during the competition.

**7. ROBOT SIZE**

Robot must be able to fit in a box 31 cm long by 31 cm wide by 27 cm high. The robot cannot separate into multiple parts and must never extend itself beyond the dimensions allowed.

**8. ROBOT WEIGHT**

There are no restrictions on the weight of the robot.

**9. ROBOT CONSTRUCTION MATERIALS**

There are no restrictions on the types of materials used in the construction of the robot except that all robots will use the provided chassis/drive motors and microcontroller. Sensor use is also restricted to those introduced/provided for the challenge as to make for a fair contest. This is a contest primarily focused on the successful design and execution of code. Teams will be provided NiMH rechargeable batteries (chargers are located in the lab), but the robots may utilize other power sources provided that the servo drive power be limited to 7.2VDC or below. One may use a different battery chemistry such as Alkaline non-rechargeable batteries, but the servo drive voltage may not exceed 7.2VDC. Lithium battery chemistry of any type is prohibited. Other power sources (non-drive) on board may exceed 7.2VDC, but teams are responsible for all sensors, microcontrollers, fans, etc. Exceeding the voltage rating of a device may damage the technology.

**10. THE CANDLE**

The candle flame will be from 15 cm to 20 cm above the nominal floor level. The candle thickness normally will be between 1 cm and 3 cm. The exact height and size of the flame will change throughout the contest depending upon the condition of candle and its surroundings. The robot is required to find the candle no matter what the size of the flame is at that particular moment.

The candle will be placed at random in one of the rooms in the arena. The candle has an equal chance of being in any of the 4 rooms in each of the robot's 3 official trials or runs. It is possible for the candle to be in the same room on two of the robot's three runs. If it happens that the candle is placed in the same room for both the 1st and 2nd trials, then the contest official will make sure that it is a different room for the third and last trial. Thus every robot will have the candle in at least 2 rooms and possibly 3, during its 3 trials.

The candle will not be placed in a hallway, but it might be placed just inside a doorway of a room. It will be randomly placed in one of three valid potential corners identified in the maze.

The contestants cannot measure or touch the candle before it is used. Any violation will result in immediate disqualification from the competition of the team and the robot.

The candle will be mounted on a small gray plastic box/container. This base is used to help keep the candle from tipping over easily, but it will be possible to knock the candle over by bumping into it (which you don't want to do)!

**11. SENSORS**

Sensors are restricted to those identified at the onset of the semester by the Course Instructor/Contest Judge. Additional types of sensors are prohibited from the contest.

Contestants are not allowed to place any markers, beacons or reflectors on the walls or floors to aid in the robot's navigation.

During the course of the contest, sunlight may come into the contest room through open outside doors. The sunlight will not shine directly on the arenas, but may be detectable by very sensitive sensors. Part of the challenge of this contest is to design a robot that can find the candle flame and ignore everything else.

**12. THE ORDER OF RUNNING**

The robots will be assigned numbers to determine the order in which they will compete in the contest. Each robot will make a trial run in the arena in the order in which it is assigned. The robots will compete consecutively and when everyone is done with their first attempt the whole process will repeat for the second and third attempts.

While the robots will be impounded during the contest week, contestants will have time between their trials (during the contest) to make any adjustments, modifications or repairs to their robot, but once the robot before them has completed its trial, then they will have 1 minute to get their robot in the arena and started on its trial. There will be a special clock at each arena that the judges will start when they call for the next contestants to get ready. The robot must begin its trial before that clock reaches 1 minute. Any robot that is not ready to run after 1 minute will forfeit its

chance at that trial. It may still compete in any other trials. Once assigned, the order of running will not be changed. If you are not ready, then you've missed your turn. The time between turns is undetermined and is controlled by how long the other competitors take to complete their trials.

The contestants will show the judge how to start the robot, and identify the front of the robot.

Once the robot is ready and the judge knows how to start it, he/she will then place the robot into the maze on the home circle (position/direction determined by the judge). The two home occupant's location and identity will then be selected along with the location of the candle and the size of the circle segment to be used. This process shall be determined using a random number generator program. The judge will then place the candle in proper room/corner/white corner segment location.

Finally, the judge will then press whatever button is necessary to start the robot.

### **13. TIME LIMITS**

In order to achieve the contest objective of building a robot that can find and extinguish a fire in a house, finding the fire within a reasonable period of time is very important. The maximum time limit for a robot to find the candle will be 5 minutes. After 5 minutes the trial will be stopped. Any time the robot does not move at all for 30 seconds, the trial will be stopped and result in a disqualification for that run. Stopping a trial run for any of the above reasons will have no impact on any of the other two trial runs that the robot has.

### **14. STARTING THE ROBOT MANUALLY**

There can be one and only one button that can be pressed to start the robot. This button must be labeled as such, i.e., "START". "RUN", "GO", etc. The button must be easily accessible, however. If a wire or connection of any sort should become loose due to difficulty in reaching the start button, it will not be grounds for another contest run. It is the responsibility of the teams to ensure a robust robot design.

Also, any program necessary must be downloaded to the robot before it is put into the arena. Once that is done then the specific "start button" and only that "start button" can be pressed to start the robot. If for any reason the robot does not start, that trial is over resulting in a disqualification for that run should the robot not move within 30 seconds.

### **15. ROOM FACTOR**

In order to make the contest realistic and to encourage the creation of smart robots, we have deliberately added uncertainty into the contest. The robot does not know in which of the 4 rooms the candle has been placed. Sometimes a robot gets lucky and the candle is in the first room it searches and sometimes the candle is in the 4th room searched. The unfairness of this is that finding the candle in the 4th room you look in is a lot harder and takes longer than finding it in the 1st room you search. To reduce the impact of "luck" and give some credit to the more sophisticated robots that can search multiple rooms successfully, there will be a Room Factor involved in the scoring that will be multiplied by the Time Score to get the Operating Score. The more rooms a robot has to search before it finds the candle, the lower the Room Factor and thus the better the Operating Score.

- If the candle is in the 1st room searched, the Room Factor will be 1.0
- If the candle is in the 2nd room searched, the Room Factor will be 0.85
- If the candle is in the 3rd room searched, the Room Factor will be 0.50
- If the candle is in the 4th room searched, the Room Factor will be 0.35

It does not matter in which order the robot searches the rooms. The only thing that matters is how many rooms the robot has searched before it finds the candle.

After searching a room with a lit candle in it, there is no further reduction of room factor. This is true whether or not the robot extinguishes the candle. No matter how many more rooms the robot searches, there will be no effect on room factor. Should a robot completely enter a room, it may not enter any additional rooms whether the candle is present in that room or not.

**16. SCORING PROCEDURE**

- A. Record the Actual Time (AT) in seconds needed to put out the candle
- B. Record the Room Factor (RF)  
1st room = 1.0, 2nd room = 0.85, 3rd room = 0.50, 4th room = 0.35
- C. Multiply the Time Score by the Room Factor and Extinguishing Time Reduction (ETR is optional) for that trial. (OS = AT x RF x ETR)
- D. A OS < 5 min will qualify as a successful run.  
See 467 Syllabus for grade determination for the Performance portion of the evaluation.
- E. A disqualified run = 60 min for each run that a robot is disqualified.
- F. Total Time (TT) for the Contest Winner is determined by adding the times from the top three trials. An awesome, coveted, custom trophy will be awarded to the contest team with the lowest TT for their robot, and the winning team will automatically earn an A for the 467 final course grade. Contest teams that extinguish all three candles will be excused from the code exam (100% eval).

**17. SAFETY**

The Contest Judge may stop any robot at any time if he/she feels that it is performing, or is about to perform, any action that is dangerous or hazardous to people or equipment. No robot is allowed to use any flammable or combustible processes.

**18. UNKNOWN FACTOR**

Part of any automation project "in the real world" will present unknown challenges to the automation engineers that are developing the controls solution. This may include a change in the scope of work, a change in the deadline, a change in the technologies to be implemented, etc.

So in this contest, an unknown change or event, referred to as the Unknown Factor, will be inserted into the contest before the final evaluation at discretion of the Contest Judge. The "Unknown Factor" will test the teams' ability to react for unforeseen circumstances that typically happen during a final install of an automated solution. Reacting to this type of issue is as much of a test as any other element presented in the rules as stated. It is important for all engineers to keep their composure and react to the change as a challenge in a positive manner.

**19. PROFESSIONALISM & ETHICS**

Any unprofessional AND/OR unethical behavior exhibited by a team member during the course of this R&D project may result in a grade reduction for that student at the discretion of the Course Instructor/Contest Judge.

**20. INTERPRETING THE RULES & CHANGES**

In all matters of interpreting these rules before and during the contest and in any issues not covered by these rules, the decisions of the Contest Judge will be final, and the runs may be modified as needed for clarification or "at will" by the Contest Judge. If a team should have any question or concern about anything related to the rules and their interpretation, they are encouraged to speak with the Contest Judge ASAP for further information.

Wright, J. (2021). Firefighting Mobile Robot Contest (R&D Project). John R. Wright.

Retrieved December 9, 2021, from

<https://sites.millersville.edu/jwright/Fire%20Contest%20Rules%20-%20Final%20Project%20Fall%202021.pdf>.

## Appendix E: Component Tech Sheets

This appendix contains all available tech sheets for the components used in this project (see Appendix A). Not all components had tech sheets that were available online, but those that did are listed in the same order as they appear in Appendix A. The pages containing the tech sheets are not numbered, but the order in which the tech sheet for each component is presented is laid out in the following list. Even if the complete tech sheet for a component is unavailable, in some cases a printed version of the linked shopping website (see Appendix A) will be presented in place of the tech sheet. Note that in some cases, the tech sheet presented here may not be for the exact same product as listed in Appendix A.

1. IR Sensor (<https://www.sparkfun.com/datasheets/Components/GP2Y0A21YK.pdf>)
2. Line Tracking Sensor  
([https://wiki.dfrobot.com/Line\\_Tracking\\_Sensor\\_for\\_Arduino\\_V4\\_SKU\\_SEN0017](https://wiki.dfrobot.com/Line_Tracking_Sensor_for_Arduino_V4_SKU_SEN0017))
3. LCD Module  
([https://www.hawkusa.com/sites/hawk-dev.ent.c-g.io/files/hawk\\_item/NWHVN/Series%20LCD%20Character/NHD-0216K1Z-NSW-FBW-L/spec/NHD-0216K1Z-NSW-FBW-L-spec.pdf](https://www.hawkusa.com/sites/hawk-dev.ent.c-g.io/files/hawk_item/NWHVN/Series%20LCD%20Character/NHD-0216K1Z-NSW-FBW-L/spec/NHD-0216K1Z-NSW-FBW-L-spec.pdf))
4. Color Detection Camera  
([https://www.dfrobot.com/product-1648.html?gclid=Cj0KCQiAqbyNBhC2ARIsALDwAsDnRz\\_TYK7yqTzDfs8O0wm1t6\\_1mUzjabZqv8Rknvbj\\_uXG\\_GKXSKkaAh4aEALw\\_wcB](https://www.dfrobot.com/product-1648.html?gclid=Cj0KCQiAqbyNBhC2ARIsALDwAsDnRz_TYK7yqTzDfs8O0wm1t6_1mUzjabZqv8Rknvbj_uXG_GKXSKkaAh4aEALw_wcB))
5. 1-Channel Flame Sensor  
(<http://rogerbit.com/wp-content/uploads/2018/01/Flame-sensor-arduino.pdf>)
6. 3-Pin Sonar (<https://www.parallax.com/product/ping-ultrasonic-distance-sensor/>)

7. Dual Relay Board (<http://www.handsontec.com/dataspecs/2Ch-relay.pdf>)
8. Servo Motor  
(<https://www.verical.com/datasheet/adafruit-servo-motors-154-5038561.pdf>)
9. Microcontroller (Teensy 3.2) (<https://www.pjrc.com/teensy/techspecs.html>)



# GP2Y0A21YK/ GP2Y0D21YK

## ■ Features

1. Less influence on the color of reflective objects, reflectivity
2. Line-up of distance output/distance judgement type  
Distance output type (analog voltage) : **GP2Y0A21YK**  
Detecting distance : 10 to 80cm  
Distance judgement type : **GP2Y0D21YK**  
Judgement distance : 24cm  
(Adjustable within the range of 10 to 80cm [Optionally available])
3. External control circuit is unnecessary
4. Low cost

## ■ Applications

1. TVs
2. Personal computers
3. Cars
4. Copiers

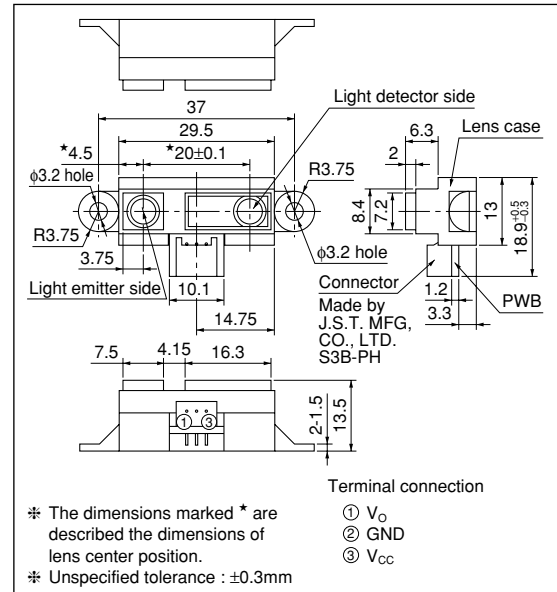
## ■ Absolute Maximum Ratings (T<sub>a</sub>=25°C, V<sub>CC</sub>=5V)

Parameter	Symbol	Rating	Unit
Supply voltage	V <sub>CC</sub>	-0.3 to +7	V
Output terminal voltage	V <sub>O</sub>	-0.3 to V <sub>CC</sub> +0.3	V
Operating temperature	T <sub>opr</sub>	-10 to +60	°C
Storage temperature	T <sub>stg</sub>	-40 to +70	°C

## General Purpose Type Distance Measuring Sensors

## ■ Outline Dimensions

(Unit : mm)



■ Recommended Operating Conditions

Parameter	Symbol	Rating	Unit
Operating supply voltage	V <sub>CC</sub>	4.5 to +5.5	V

■ Electro-optical Characteristics

(T<sub>a</sub>=25°C, V<sub>CC</sub>=5V)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit	
Distance measuring range	ΔL	*1 *3	10	—	80	cm	
Output terminal voltage	GP2Y0A21YK	V <sub>O</sub>	L=80cm *1	0.25	0.4	0.55	V
	GP2Y0D21YK	V <sub>OH</sub>	Output voltage at High *1	V <sub>CC</sub> -0.3	—	—	V
		V <sub>OL</sub>	Output voltage at Low *1	—	—	0.6	V
Difference of output voltage	GP2Y0A21YK	ΔV <sub>O</sub>	Output change at L=80cm to 10cm *1	1.65	1.9	2.15	V
Distance characteristics of output	GP2Y0D21YK	V <sub>O</sub>	*1 *4 *2	21	24	27	cm
Average Dissipation current	I <sub>CC</sub>	L=80cm *1	—	30	40	mA	

Note) L : Distance to reflective object

\*1 Using reflective object : White paper (Made by Kodak Co. Ltd. gray cards R-27 · white face, reflective ratio ; 90%)

\*2 We ship the device after the following adjustment : Output switching distance L=24cm±3cm must be measured by the sensor

\*3 Distance measuring range of the optical sensor system

\*4 Output switching has a hysteresis width. The distance specified by V<sub>O</sub> should be the one with which the output L switches to the output H

Fig.1 Internal Block Diagram

Fig.2 Internal Block Diagram

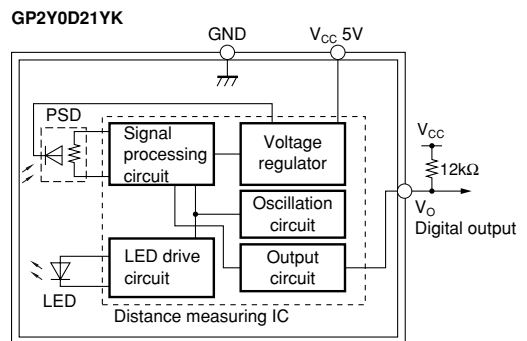
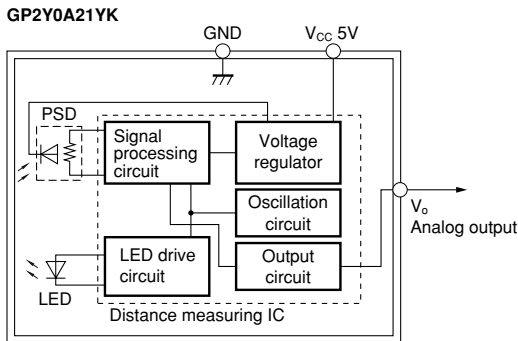


Fig.3 Timing Chart

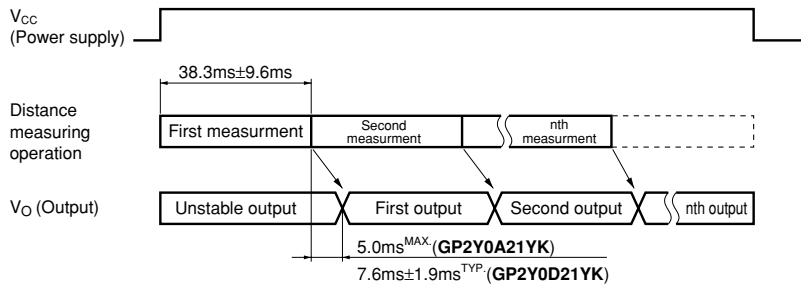


Fig.4 Distance Characteristics

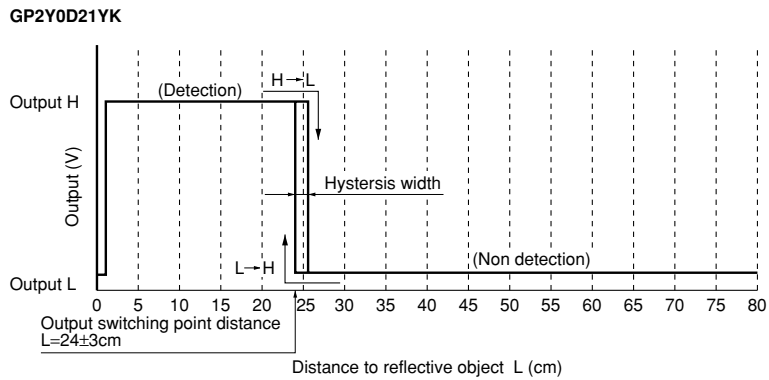
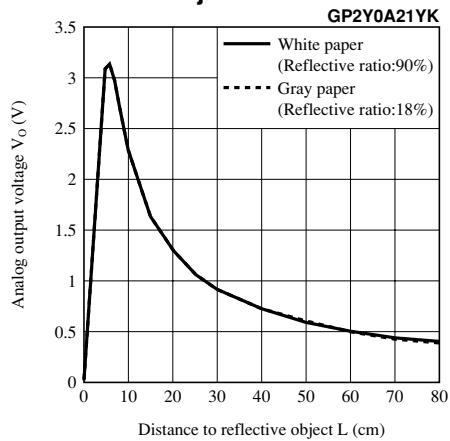


Fig.5 Analog Output Voltage vs. Distance to Reflective Object



## NOTICE

- The circuit application examples in this publication are provided to explain representative applications of SHARP devices and are not intended to guarantee any circuit design or license any intellectual property rights. SHARP takes no responsibility for any problems related to any intellectual property right of a third party resulting from the use of SHARP's devices.
- Contact SHARP in order to obtain the latest device specification sheets before using any SHARP device. SHARP reserves the right to make changes in the specifications, characteristics, data, materials, structure, and other contents described herein at any time without notice in order to improve design or reliability. Manufacturing locations are also subject to change without notice.
- Observe the following points when using any devices in this publication. SHARP takes no responsibility for damage caused by improper use of the devices which does not meet the conditions and absolute maximum ratings to be used specified in the relevant specification sheet nor meet the following conditions:
  - (i) The devices in this publication are designed for use in general electronic equipment designs such as:
    - Personal computers
    - Office automation equipment
    - Telecommunication equipment [terminal]
    - Test and measurement equipment
    - Industrial control
    - Audio visual equipment
    - Consumer electronics
  - (ii) Measures such as fail-safe function and redundant design should be taken to ensure reliability and safety when SHARP devices are used for or in connection with equipment that requires higher reliability such as:
    - Transportation control and safety equipment (i.e., aircraft, trains, automobiles, etc.)
    - Traffic signals
    - Gas leakage sensor breakers
    - Alarm equipment
    - Various safety devices, etc.
  - (iii) SHARP devices shall not be used for or in connection with equipment that requires an extremely high level of reliability and safety such as:
    - Space applications
    - Telecommunication equipment [trunk lines]
    - Nuclear power control equipment
    - Medical and other life support equipment (e.g., scuba).
- Contact a SHARP representative in advance when intending to use SHARP devices for any "specific" applications other than those recommended by SHARP or when it is unclear which category mentioned above controls the intended use.
- If the SHARP devices listed in this publication fall within the scope of strategic products described in the Foreign Exchange and Foreign Trade Control Law of Japan, it is necessary to obtain approval to export such SHARP devices.
- This publication is the proprietary product of SHARP and is copyrighted, with all rights reserved. Under the copyright laws, no part of this publication may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, in whole or in part, without the express written permission of SHARP. Express written permission is also required before any use of this publication may be made by a third party.
- Contact and consult with a SHARP representative if there are any questions about the contents of this publication.

SKU:SEN0017 (<https://www.dfrobot.com/product-85.html>)

---

## Introduction



(<https://www.dfrobot.com/product-85.html>)

The DFRobot **Line Tracking Sensor for Arduino** (<https://www.dfrobot.com/product-85.html>) can detect the white lines against a black backdrop and black lines against a white backdrop. The single line-tracking signals can provide a stable TTL output signal to ensure accuracy and reliability. Optionally, a multi-channel mix is easy to install with the necessary line-tracking robot sensors.

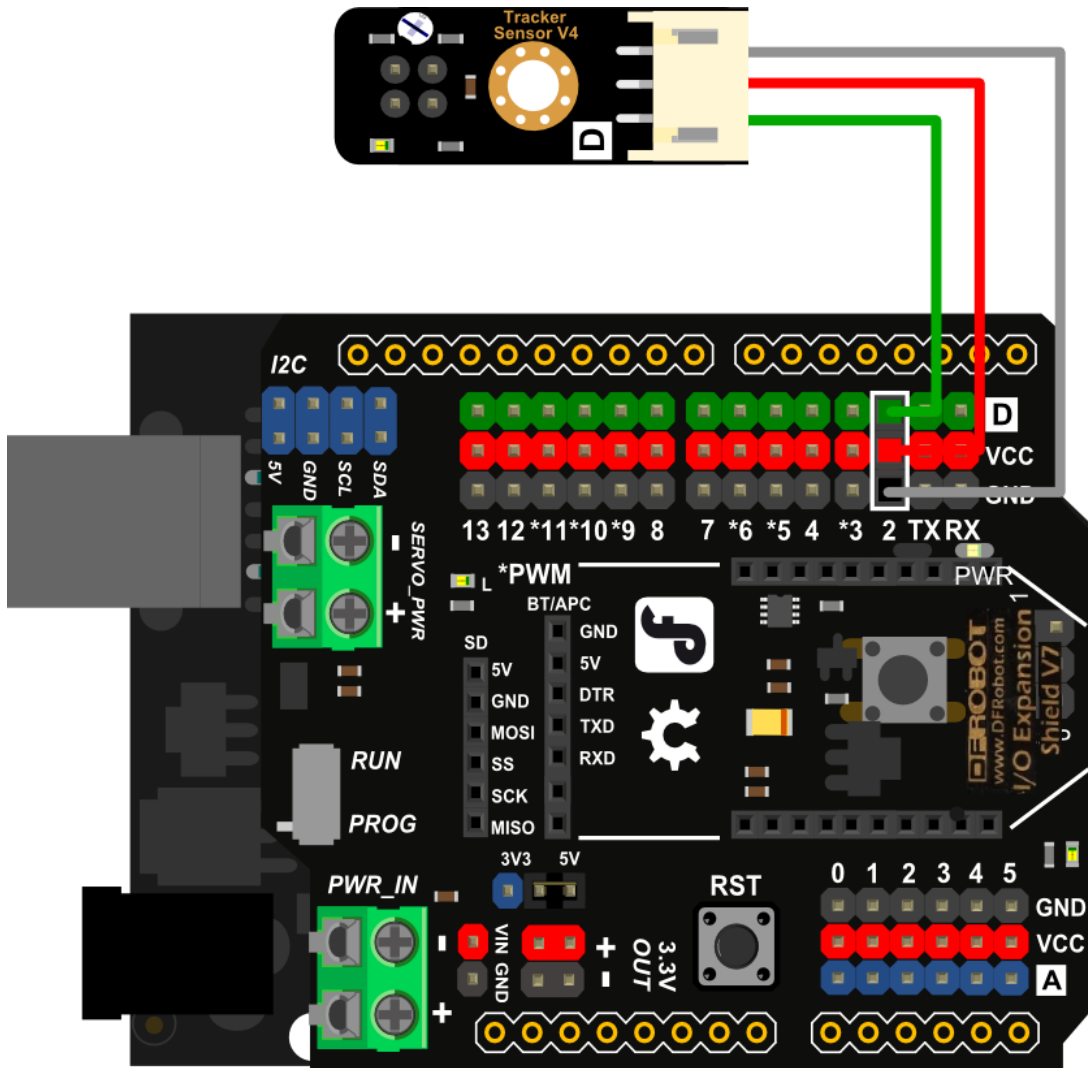
- Version Update The best distance between objects such as the ground and the sensor is 1-2 cm. V5 removes the variable resistor and is therefore easier to use now.

## Specification

- Power supply: 3.3~5V
- Detecting Range: 1~2cm
- Operating current: <10mA
- Operating temperature range: 0°C ~ + 50°C
- Output interface: 3-wire interface (1 - signal, 2 - power, 3 - power supply negative)
- Output: TTL(Black for LOW output, White for HIGH output)
- Module Size: 10mm×28mm (1.1x 0.4 in)
- Module Weight: About 10g

## Tutorial

### Connection Diagram



## Sample Code

```

//Arduino Sample Code
void setup()
{
  Serial.begin(9600);
}
void loop()
{
  Serial.println(digitalRead(2)); // print the data from the sensor
  delay(500);
}

```

## FAQ

Q&A	Some general Arduino Problems/FAQ/Tips
A	For any questions, advice or cool ideas to share, please visit the <a href="https://www.dfrobot.com/forum/">DFRobot Forum</a> (https://www.dfrobot.com/forum/).

## More Documents

 Get Digital Line Tracking(Following) Sensor For Arduino (<https://www.dfrobot.com/product-85.html>) from DFRobot Store or DFRobot Distributor.

(<https://www.dfrobot.com/index.php?route=information/distributorsloop>)

[https://wiki.dfrobot.com/Line\\_Tracking\\_Sensor\\_for\\_Arduino\\_V4\\_SKU\\_SEN0017](https://wiki.dfrobot.com/Line_Tracking_Sensor_for_Arduino_V4_SKU_SEN0017)

# NHD-0216K1Z-NSW-FBW-L

## Character Liquid Crystal Display Module

NHD-	Newhaven Display
0216-	2 lines x 16 characters
K1Z-	Model
N-	Transmissive
SW-	Side White LED Backlight
F-	FSTN(-)
B-	6:00 view
W-	Wide Temperature (-20°C~+70°C)
L-	Low Power 20mA
	<b>RoHS Compliant</b>

**Newhaven Display International, Inc.**

2511 Technology Drive, Suite 101

Elgin IL, 60124

Ph: 847-844-8795

Fax: 847-844-8796

[www.newhavendisplay.com](http://www.newhavendisplay.com)

[nhtech@newhavendisplay.com](mailto:nhtech@newhavendisplay.com)

[nhsales@newhavendisplay.com](mailto:nhsales@newhavendisplay.com)

## Document Revision History

Revision	Date	Description	Changed by
0	10/5/2007	Initial Release	-
1	12/17/2009	User Guide Reformat	BE
2	1/7/2010	Optical revised	BE

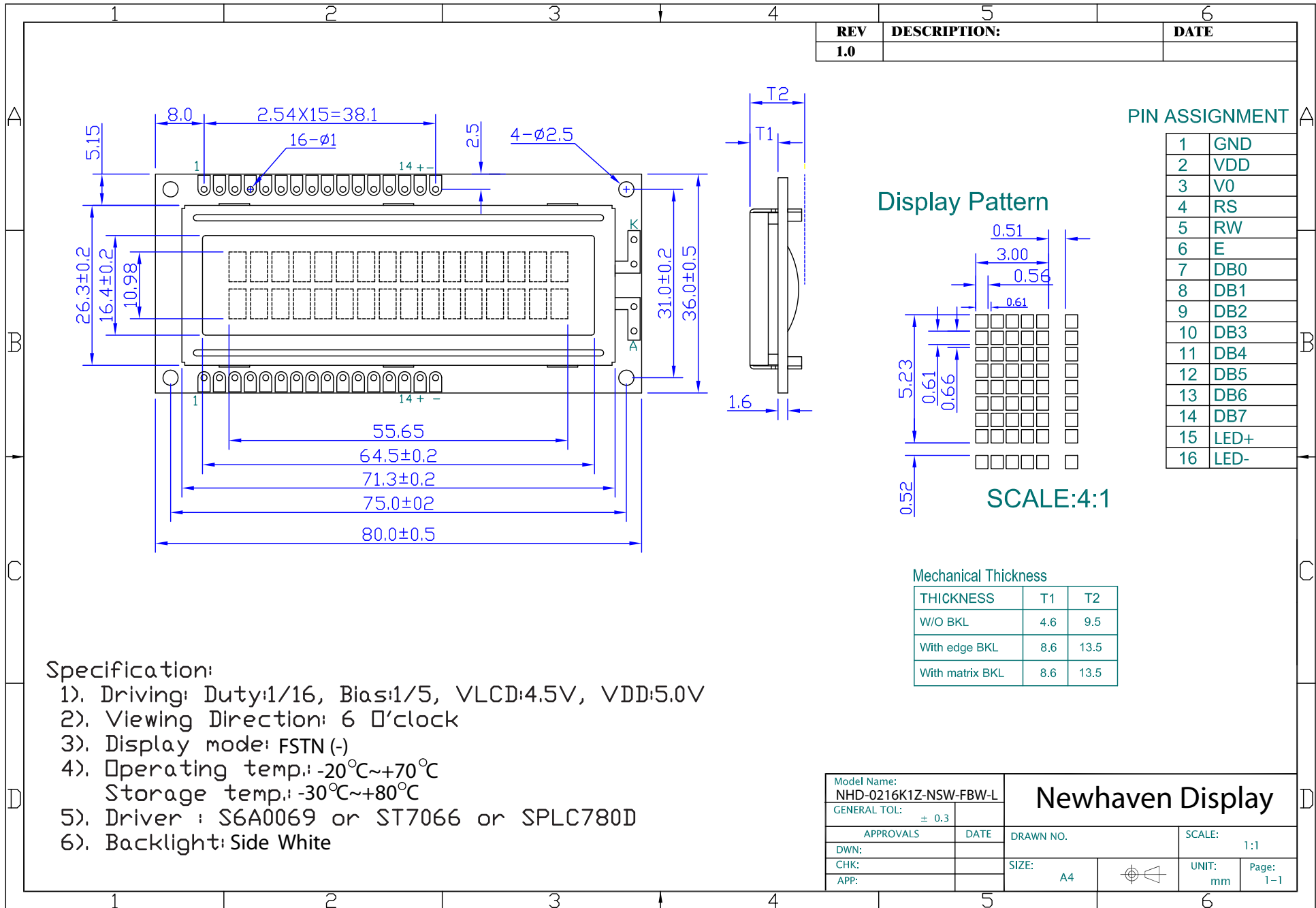
## Functions and Features

- 2 lines x 16 characters
- Built-in controller (SPLC780D or equivalent)
- +5.0V Power Supply
- 1/16 duty, 1/5 bias
- RoHS compliant

## Mechanical Drawing



# Mechanical Drawing

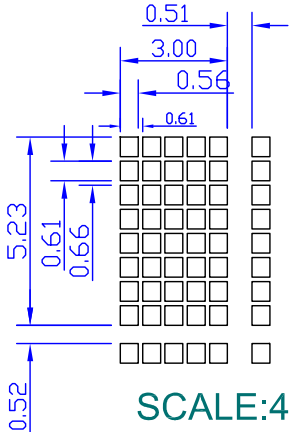


REV	DESCRIPTION:	DATE
1.0		

### PIN ASSIGNMENT

1	GND
2	VDD
3	V0
4	RS
5	RW
6	E
7	DB0
8	DB1
9	DB2
10	DB3
11	DB4
12	DB5
13	DB6
14	DB7
15	LED+
16	LED-

### Display Pattern



SCALE:4:1

### Mechanical Thickness

THICKNESS	T1	T2
W/O BKL	4.6	9.5
With edge BKL	8.6	13.5
With matrix BKL	8.6	13.5

- Specification:
- 1). Driving: Duty:1/16, Bias:1/5, VLCD:4.5V, VDD:5.0V
  - 2). Viewing Direction: 6 o'clock
  - 3). Display mode: FSTN(-)
  - 4). Operating temp: -20°C~+70°C  
Storage temp: -30°C~+80°C
  - 5). Driver : S6A0069 or ST7066 or SPLC780D
  - 6). Backlight: Side White

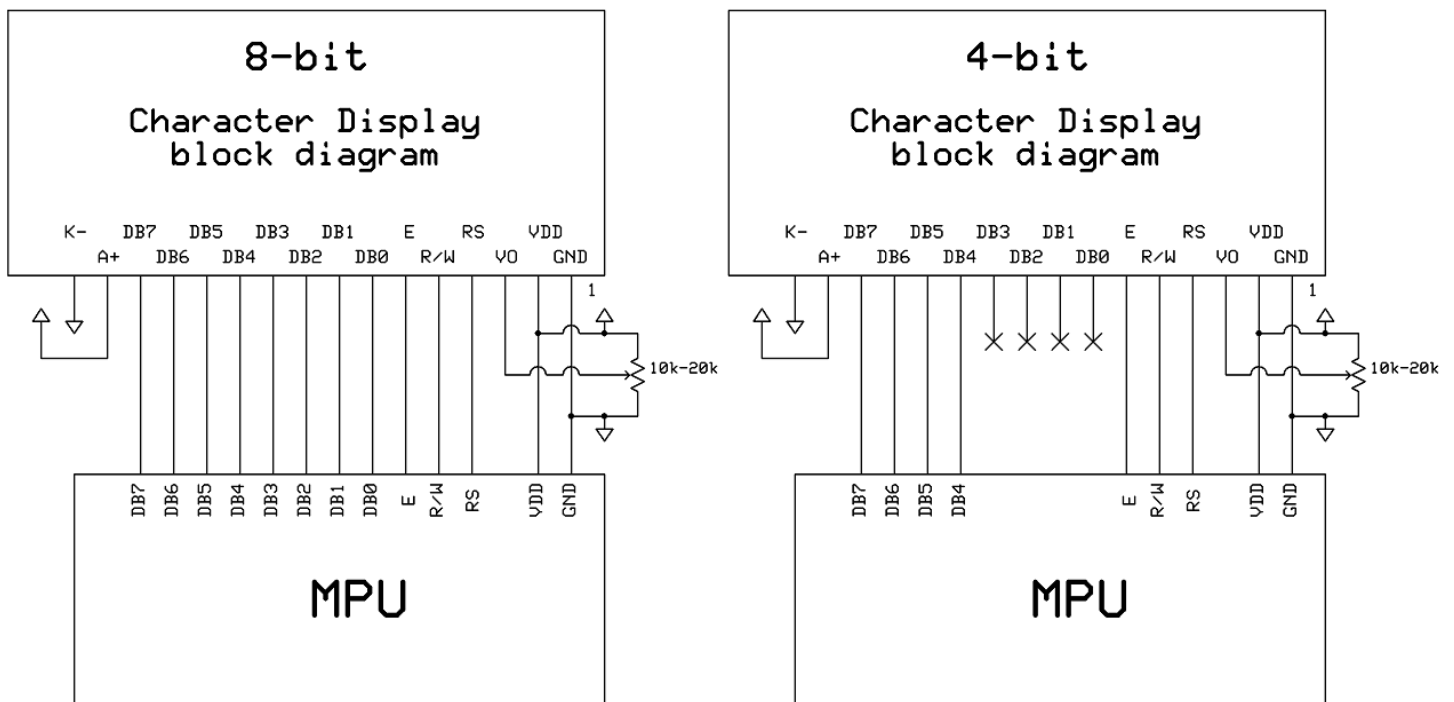
Model Name: NHD-0216K1Z-NSW-FBW-L		<b>Newhaven Display</b>	
GENERAL TOL: ± 0.3			
APPROVALS	DATE	DRAWN NO.	SCALE: 1:1
DWN:		SIZE: A4	UNIT: mm Page: 1-1
CHK:			
APP:			

## Pin Description and Wiring Diagram

Pin No.	Symbol	External Connection	Function Description
1	VSS	Power Supply	Ground
2	VDD	Power Supply	Supply Voltage for logic (+5.0V)
3	VO	Adj Power Supply	Power supply for contrast (approx. 0.5V)
4	RS	MPU	Register select signal. RS=0: Command, RS=1: Data
5	R/W	MPU	Read/Write select signal, R/W=1: Read R/W: =0: Write
6	E	MPU	Operation enable signal. Falling edge triggered.
7-10	DB0 – DB3	MPU	Four low order bi-directional three-state data bus lines. These four are not used during 4-bit operation.
11-14	DB4 – DB7	MPU	Four high order bi-directional three-state data bus lines.
15	LED+	Power Supply	Power supply for LED Backlight (+5.0V via on-board resistor)
16	LED-	Power Supply	Ground for Backlight

Recommended LCD connector: 2.54mm pitch pins

Backlight connector: --- Mates with: ---



## Electrical Characteristics

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
Operating Temperature Range	Top	Absolute Max	-20	-	+70	°C
Storage Temperature Range	Tst	Absolute Max	-30	-	+80	°C
Supply Voltage	VDD		4.7	5.0	5.5	V
Supply Current	IDD	Ta=25°C, VDD=5.0V	-	1.5	2.5	mA
Supply for LCD (contrast)	VDD-V0	Ta=25°C	-	4.5	-	V
"H" Level input	Vih		2.2	-	VDD	V
"L" Level input	Vil		0	-	0.6	V
"H" Level output	Voh		2.4	-	-	V
"L" Level output	Vol		-	-	0.4	V
Backlight Supply Voltage	Vled	-	-	5.0	-	V
Backlight Supply Current	Iled	Vled=5.0V	-	20	-	mA

## Optical Characteristics

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
Viewing Angle – Vertical (top)	AV	Cr ≥ 3	-	20	-	°
Viewing Angle – Vertical (bottom)	AV	Cr ≥ 3	-	50	-	°
Viewing Angle – Horizontal (left)	AH	Cr ≥ 3	-	30	-	°
Viewing Angle – Horizontal (right)	AH	Cr ≥ 3	-	30	-	°
Contrast Ratio	Cr		3	5	-	-
Response Time (rise)	Tr	-	-	150	250	ms
Response Time (fall)	Tf	-	-	150	250	ms

## Controller Information

Built-in SPLC780D. Download specification at [http://www.newhavendisplay.com/app\\_notes/SPLC780D.pdf](http://www.newhavendisplay.com/app_notes/SPLC780D.pdf)

## Table of Commands

Instruction	Instruction Code										Description	Execution time (fosc=270KHz)
	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0		
Clear Display	0	0	0	0	0	0	0	0	0	1	Write "20H" to DDRAM and set DDRAM address to "00H" from AC	1.52ms
Return Home	0	0	0	0	0	0	0	0	1	-	Set DDRAM address to "00H" from AC and return cursor to its original position if shifted. The contents of DDRAM are not changed.	1.52ms
Entry Mode Set	0	0	0	0	0	0	0	1	I/D	S	Assign cursor moving direction and enable the shift of entire display	38μs
Display ON/OFF Control	0	0	0	0	0	0	1	D	C	B	Set display(D), cursor(C), and blinking of cursor(B) on/off control bit.	38μs
Cursor or Display Shift	0	0	0	0	0	1	S/C	R/L	-	-	Set cursor moving and display shift control bit, and the direction, without changing of DDRAM data.	38μs
Function Set	0	0	0	0	1	DL	N	F	-	-	Set interface data length (DL: 8bit/4-bit), numbers of display line (N: 2-line/1-line) and, display font type (F:5x10 dots/5x8 dots)	38μs
Set CGRAM Address	0	0	0	1	AC5	AC4	AC3	AC2	AC1	AC0	Set CGRAM address in address counter.	38μs
Set DDRAM Address	0	0	1	AC6	AC5	AC4	AC3	AC2	AC1	AC0	Set DDRAM address in counter	38μs
Read Busy Flag and Address Counter	0	1	BF	AC6	AC5	AC4	AC3	AC2	AC1	AC0	Whether during internal operation or not can be known by reading BF. The contents of address counter can also be read.	
Write Data to RAM	1	0	D7	D6	D5	D4	D3	D2	D1	D0	Write data into internal RAM (DDRAM/CGRAM).	38μs
Read Data from RAM	1	1	D7	D6	D5	D4	D3	D2	D1	D0	Read data from internal RAM (DDRAM/CGRAM).	38μs

## Display character address code:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
40	41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	4F

## Timing Characteristics

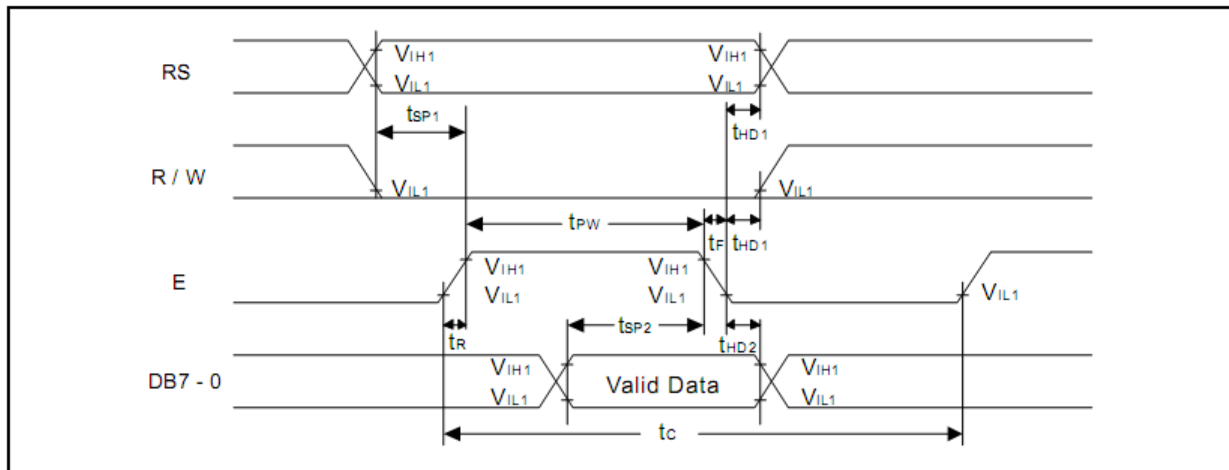
### 6.5.3. Write mode (Writing Data from MPU to SPLC780D)

Characteristics	Symbol	Limit			Unit	Test Condition
		Min.	Typ.	Max.		
E Cycle Time	$t_C$	500	-	-	ns	Pin E
E Pulse Width	$t_{PW}$	230	-	-	ns	Pin E
E Rise/Fall Time	$t_R, t_F$	-	-	20	ns	Pin E
Address Setup Time	$t_{SP1}$	40	-	-	ns	Pins: RS, R/W, E
Address Hold Time	$t_{HD1}$	10	-	-	ns	Pins: RS, R/W, E
Data Setup Time	$t_{SP2}$	80	-	-	ns	Pins: DB0 - DB7
Data Hold Time	$t_{HD2}$	10	-	-	ns	Pins: DB0 - DB7

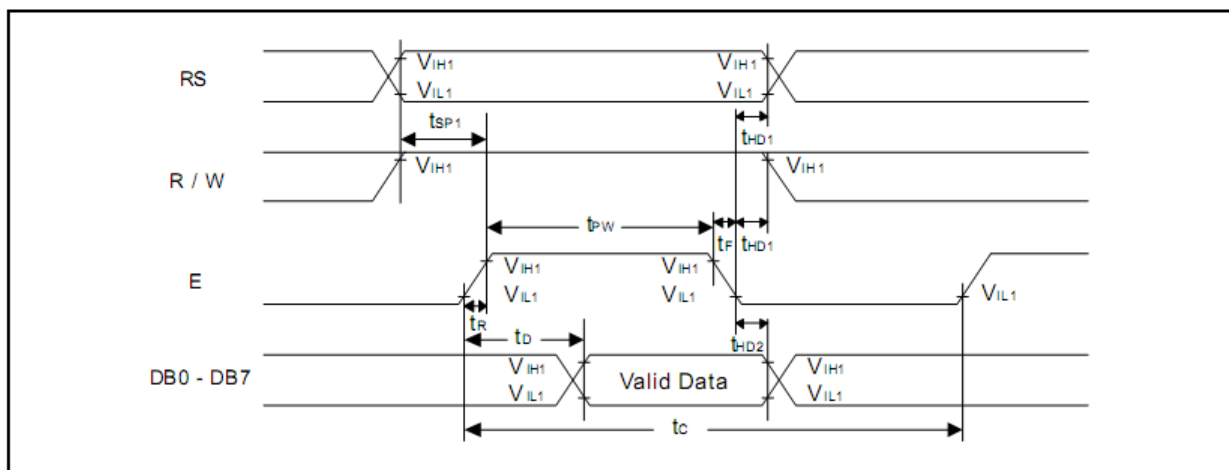
### 6.5.4. Read mode (Reading Data from SPLC780D to MPU)

Characteristics	Symbol	Limit			Unit	Test Condition
		Min.	Typ.	Max.		
E Cycle Time	$t_C$	500	-	-	ns	Pin E
E Pulse Width	$t_W$	230	-	-	ns	Pin E
E Rise/Fall Time	$t_R, t_F$	-	-	20	ns	Pin E
Address Setup Time	$t_{SP1}$	40	-	-	ns	Pins: RS, R/W, E
Address Hold Time	$t_{HD1}$	10	-	-	ns	Pins: RS, R/W, E
Data Output Delay Time	$t_D$	-	-	120	ns	Pins: DB0 - DB7
Data hold time	$t_{HD2}$	5.0	-	-	ns	Pin DB0 - DB7

### 6.5.6. Write mode timing diagram (Writing Data from MPU to SPLC780D)



### 6.5.7. Read mode timing diagram (Reading Data from SPLC780D to MPU)



## Built-in Font Table

Lower 4 Bits \ Upper 4 Bits	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
xxxx0000	CG RAM (1)			0	a	P	`	P				-	タ	ミ	&	P
xxxx0001	(2)		!	1	A	Q	a	q			。	ア	チ	△	ä	q
xxxx0010	(3)		"	2	B	R	b	r			「	イ	ツ	×	ß	θ
xxxx0011	(4)		#	3	C	S	c	s			」	ウ	テ	モ	ε	8
xxxx0100	(5)		\$	4	D	T	d	t			、	エ	ト	ト	μ	Ω
xxxx0101	(6)		%	5	E	U	e	u			・	オ	ナ	1	ø	ü
xxxx0110	(7)		&	6	F	V	f	v			ヲ	カ	ニ	ヨ	ρ	∑
xxxx0111	(8)		'	7	G	W	g	w			ヲ	キ	ヌ	ラ	g	π
xxxx1000	(1)		(	8	H	X	h	x			イ	ク	ネ	リ	√	∞
xxxx1001	(2)		)	9	I	Y	i	y			ウ	ケ	ル	ル	'	γ
xxxx1010	(3)		*	:	J	Z	j	z			エ	コ	ハ	レ	j	≠
xxxx1011	(4)		+	;	K	[	k	<			オ	サ	ヒ	ロ	*	≠
xxxx1100	(5)		,	<	L	¥	l	l			カ	シ	フ	ク	φ	∞
xxxx1101	(6)		-	=	M	]	m	)			ユ	ス	ハ	ン	も	÷
xxxx1110	(7)		.	>	N	^	n	→			ヨ	セ	ホ	°	ñ	
xxxx1111	(8)		/	?	O	_	o	←			ツ	リ	マ	°	ö	■

## Example Initialization Program

8-bit Initialization:

```

/*****/
void command(char i)
{
    P1 = i;           //put data on output Port
    D_I =0;          //D/I=LOW : send instruction
    R_W =0;          //R/W=LOW : Write
    E = 1;
    Delay(1);        //enable pulse width >= 300ns
    E = 0;           //Clock enable: falling edge
}
/*****/
void write(char i)
{
    P1 = i;           //put data on output Port
    D_I =1;          //D/I=LOW : send data
    R_W =0;          //R/W=LOW : Write
    E = 1;
    Delay(1);        //enable pulse width >= 300ns
    E = 0;           //Clock enable: falling edge
}
/*****/
void init()
{
    E = 0;
    Delay(100);      //Wait >15 msec after power is applied
    command(0x30);   //command 0x30 = Wake up
    Delay(30);       //must wait 5ms, busy flag not available
    command(0x30);   //command 0x30 = Wake up #2
    Delay(10);       //must wait 160us, busy flag not available
    command(0x30);   //command 0x30 = Wake up #3
    Delay(10);       //must wait 160us, busy flag not available
    command(0x38);   //Function set: 8-bit/2-line
    command(0x10);   //Set cursor
    command(0x0c);   //Display ON; Cursor ON
    command(0x06);   //Entry mode set
}
/*****/
```



```

4-bit Initialization:
/*****/
void command(char i)
{
    P1 = i;                //put data on output Port
    D_I =0;                //D/I=LOW : send instruction
    R_W =0;                //R/W=LOW : Write
    Nybble();              //Send lower 4 bits
    i = i<<4;              //Shift over by 4 bits
    P1 = i;                //put data on output Port
    Nybble();              //Send upper 4 bits
}
/*****/
void write(char i)
{
    P1 = i;                //put data on output Port
    D_I =1;                //D/I=HIGH : send data
    R_W =0;                //R/W=LOW : Write
    Nybble();              //Clock lower 4 bits
    i = i<<4;              //Shift over by 4 bits
    P1 = i;                //put data on output Port
    Nybble();              //Clock upper 4 bits
}
/*****/
void Nybble()
{
    E = 1;
    Delay(1);              //enable pulse width >= 300ns
    E = 0;                 //Clock enable: falling edge
}
/*****/
void init()
{
    P1 = 0;
    P3 = 0;
    Delay(100);            //Wait >15 msec after power is applied
    P1 = 0x30;             //put 0x30 on the output port
    Delay(30);             //must wait 5ms, busy flag not available
    Nybble();              //command 0x30 = Wake up
    Delay(10);             //must wait 160us, busy flag not available
    Nybble();              //command 0x30 = Wake up #2
    Delay(10);             //must wait 160us, busy flag not available
    Nybble();              //command 0x30 = Wake up #3
    Delay(10);             //can check busy flag now instead of delay
    P1= 0x20;              //put 0x20 on the output port
    Nybble();              //Function set: 4-bit interface
    command(0x28);         //Function set: 4-bit/2-line
    command(0x10);         //Set cursor
    command(0x0F);         //Display ON; Blinking cursor
    command(0x06);         //Entry Mode set
}
/*****/

```

## Quality Information

Test Item	Content of Test	Test Condition	Note
High Temperature storage	Endurance test applying the high storage temperature for a long time.	+80°C , 48hrs	2
Low Temperature storage	Endurance test applying the low storage temperature for a long time.	-30°C , 48hrs	1,2
High Temperature Operation	Endurance test applying the electric stress (voltage & current) and the high thermal stress for a long time.	+70°C 48hrs	2
Low Temperature Operation	Endurance test applying the electric stress (voltage & current) and the low thermal stress for a long time.	-20°C , 48hrs	1,2
High Temperature / Humidity Operation	Endurance test applying the electric stress (voltage & current) and the high thermal with high humidity stress for a long time.	+40°C , 90% RH , 48hrs	1,2
Thermal Shock resistance	Endurance test applying the electric stress (voltage & current) during a cycle of low and high thermal stress.	0°C,30min -> 25°C,5min -> 50°C,30min = 1 cycle 10 cycles	
Vibration test	Endurance test applying vibration to simulate transportation and use.	10-55Hz , 15mm amplitude. 60 sec in each of 3 directions X,Y,Z For 15 minutes	3
Static electricity test	Endurance test applying electric static discharge.	VS=800V, RS=1.5kΩ, CS=100pF One time	

**Note 1:** No condensation to be observed.

**Note 2:** Conducted after 4 hours of storage at 25°C, 0%RH.

**Note 3:** Test performed on product itself, not inside a container.

## Precautions for using LCDs/LCMs

See Precautions at [www.newhavendisplay.com/specs/precautions.pdf](http://www.newhavendisplay.com/specs/precautions.pdf)

## Warranty Information and Terms & Conditions

[http://www.newhavendisplay.com/index.php?main\\_page=terms](http://www.newhavendisplay.com/index.php?main_page=terms)

Register & Complete the Informations to Get Coupons Worth \$33 (<https://www.dfrobot.com/blog-1581.html?tracking=61921bb55f8d3>)



Search

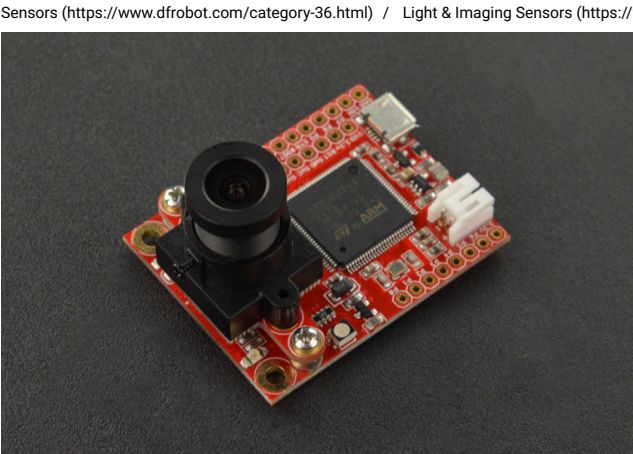


LOGIN/SIGN UP

HOME (/) COMMUNITY (<https://community.dfrobot.com>) FORUM (<https://www.dfrobot.com/forum>) WIKI (<https://wiki.dfrobot.com>) BLOG (<https://www.dfrobot.com/blog>) PRODUCT LINES ∨ EDUCATION (<https://edu.dfrobot.com>)

\$USD

(<https://www.dfrobot.com/i> route=account/login)



# OpenMV Cam H7 – A Machine Vision Camera

SKU:DFR0517 Brand:Other Reward Points: 650 WISH LIST (/INDEX.PHP?ROUTE=ACCOUNT/WISHLI

\$65.00

In Stock

Categories: [Sensors](https://www.dfrobot.com/category-36.html) (<https://www.dfrobot.com/category-36.html>) [All Products](https://www.dfrobot.com/category-48.html) (<https://www.dfrobot.com/category-48.html>) [Light & Imaging Sensors](https://www.dfrobot.com/category-58.html) (<https://www.dfrobot.com/category-58.html>)

Topic: [Artificial Intelligence Hardware](https://www.dfrobot.com/topic-284.html) (<https://www.dfrobot.com/topic-284.html>)

\* Model:  H7 Plus  H7 (<https://www.dfrobot.com/product-2263.html>) (<https://www.dfrobot.com/product-1648.html>)

LCD Shield  WiFi Shield (<https://www.dfrobot.com/product-2264.html>) (<https://www.dfrobot.com/product-2262.html>)

Gravity I/O Expansion Shield (<https://www.dfrobot.com/product-1791.html>)

Quantity:

[BUY IT NOW](#) [ADD TO CART](#) [WISHLIST](#)



Frequently Bought Together

OpenMV Cam H7 – A Machine Vision Camera (<https://www.dfrobot.com/product-1648.html>) + Lummer Wires 7 1" (<https://www.dfrobot.com/product-873.html>) + Flat Noodle Micro (<https://www.dfrobot.com/product-1076.html>) + Gravity I/O (<https://www.dfrobot.com/product-1791.html>) + Horned Sunnam (<https://www.dfrobot.com/product-1865.html>) = [BUY IT NOW](#)

You have chosen:0 Total amount: \$0

## INTRODUCTION

The OpenMV Cam H7 comes with a MT9M114 image sensor is capable of taking 640x480 8-bit Grayscale images or 640x480 8-bit BAYER images at 40 FPS when the resolution is above 320x240 and 80 FPS when it is below. Most simple algorithms will run between 40-80 FPS on QVGA (320x240) resolutions and below. Your image sensor comes with a 2.1mm lens on a standard M12 lens mount. If you want to use more specialized lenses with your image sensor you can easily buy and attach them yourself.

## FEATURES

- The STM32H743VI ARM Cortex M7 processor running at 480 MHz with 1MB SRAM and 2MB of flash. All I/O pins output 3.3V and are 5V tolerant. The processor has the following I/O interfaces:
- A full speed USB (12Mbps) interface to your computer. Your OpenMV Cam will appear as a Virtual COM Port and a USB Flash Drive when plugged in.
- A μSD Card socket capable of 100Mbps reads/writes which allows your OpenMV Cam to take pictures and easily pull machine vision assets off of the μSD card.
- A SPI bus that can run up to 80Mbps allowing you to easily stream image data off the system to either the LCD Shield, the WiFi Shield, or another microcontroller.
- An I2C Bus (up to 1Mb/s), CAN Bus (up to 1Mb/s), and an Asynchronous Serial Bus (TX/RX, up to 7.5Mb/s) for interfacing with other microcontrollers and sensors.
- A 12-bit ADC and a 12-bit DAC.
- Three I/O pins for servo control.
- Interrupts and PWM on all I/O pins (there are 10 I/O pins on the board).
- An RGB LED and two high power 850nm IR LEDs.

## APPLICATIONS

- The OpenMV Cam can be used for the following things currently (more in the future):
- Frame Differencing
  - Color Tracking
  - Marker Tracking
  - Face Detection
  - Eye Tracking

### OpenMV Cam H7 – A Machine Vision Camera

\$65.00

In Stock

Quantity:

[BUY IT NOW](#)

[ADD TO CART](#)

- INTRODUCTION
- FEATURES
- APPLICATIONS
- SPECIFICATION
- DOCUMENTS
- SHIPPING LIST
- REVIEW
- FAQ

[BACK TO TOP](#)

- Optical Flow
- QR Code Detection/Decoding
- Data Matrix Detection/Decoding
- Linear Barcode Decoding
- AprilTag Tracking
- Line Detection
- Circle Detection
- Rectangle Detection
- Template Matching
- Image Capture
- Video Recording

## SPECIFICATION

Processor	ARM® 32-bit Cortex®-M7 CPU
	w/ Double Precision FPU
	480 MHz (1027 DMIPS)
	Core Mark Score: 2400
	(compare w/ Raspberry Pi 2: 2340)
RAM Layout	64KB Stack
	256KB .DATA/.BSS/Heap
	512KB Frame Buffer/Stack
	256KB DMA Buffers
Flash Layout	128KB Bootloader
	128KB Embedded Flash Drive
	1792KB Firmware
	(2MB Total)
Supported Image Formats	Grayscale
	RGB565
	JPEG (and BAYER)
Maximum Supported Resolutions	Grayscale: 640x480 and under
	RGB565: 320x240 and under
	Grayscale JPEG: 640x480 and under
	RGB565 JPEG: 640x480 and under
Lens Info	Focal Length: 2.1mm
	Aperture: F2.0
	Format: 1/6"
	HFOV = 60.7°, VFOV = 47.5°
	Mount: M12*0.5
	IR Cut Filter: 650nm (removable)

## Electrical Info

All pins are 5V tolerant with 3.3V output. All pins can sink or source up to 25mA. P6 is not 5V tolerant in ADC or DAC mode. Up to 120mA may be sinked or sourced in total between all pins. VIN may be between 3.6V and 5V. Do not draw more than 250mA from your OpenMV Cam's 3.3V rail.

Weight	19g
Length	45mm
Width	36mm
Height	30mm

---

## DOCUMENTS

- Quick reference for the openmvcam (<http://docs.openmv.io/openmvcam/quickref.html>)
- OpenMV Cam Schematic (<https://github.com/openmv/openmv-boards/raw/master/openmv4/base/base.pdf>)
- Unofficial 3D CAD Model (<https://grabcad.com/library/openmv-cam-m7-1>)

---

## SHIPPING LIST

- OpenMV Cam H7 x1
- Pin Header x2

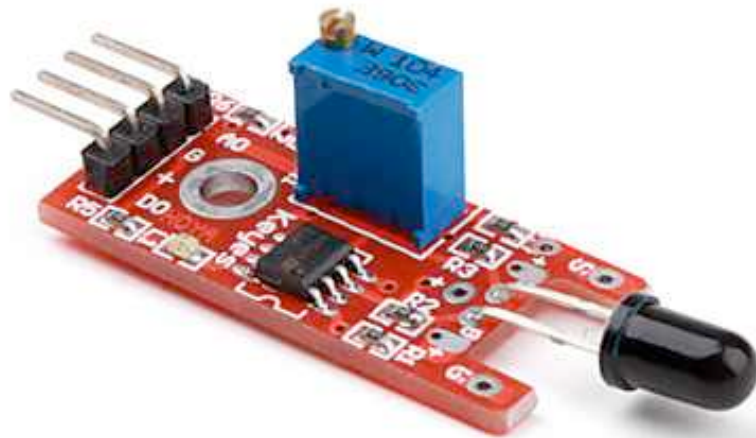
---

## REVIEW

---

## FAQ

## Flame Sensor Module



### Introduction

This module is sensitive to the flame and radiation. It also can detect ordinary light source in the range of of a wavelength 760nm-1100 nm. The detection distance is up to 100 cm.

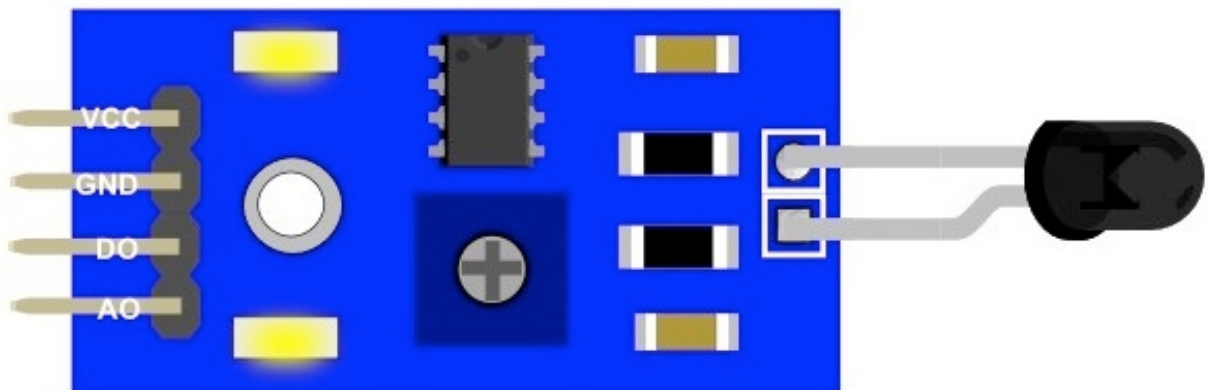
The Flame sensor can output digital or analog signal. It can be used as a flame alarm or in fire fighting robots.

## Description

- Detects a flame or a light source of a wavelength in the range of 760nm-1100 nm
- Detection distance: 20cm (4.8V) ~ 100cm (1V)
- Detection angle about 60 degrees, it is sensitive to the flame spectrum.
- Comparator chip LM393 makes module readings stable.
- Adjustable detection range.
- Operating voltage 3.3V-5V
- Digital and Analog Output
  - DO digital switch outputs (0 and 1)
  - AO analog voltage output
- Power indicator and digital switch output indicator

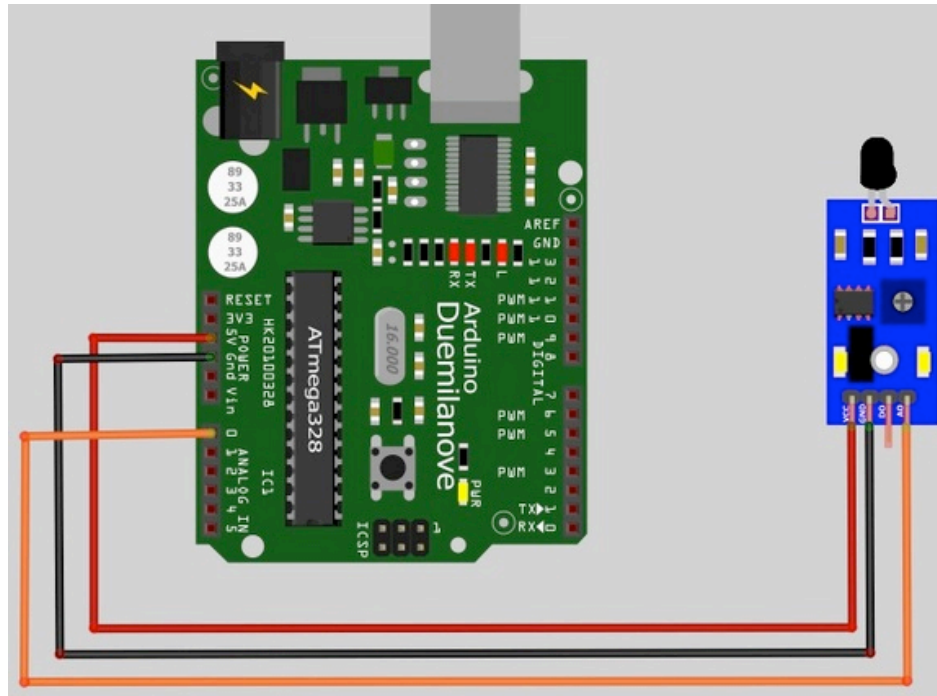
## Interface Description (4-wire)

- 1) VCC -- 3.3V-5V voltage
- 2) GND -- GND
- 3) DO -- board digital output interface (0 and 1)
- 4) AO -- board analog output interface



## Arduino Example

Here is sample code and connection to Arduino board. The analog output can be connected to any analog input pin on Arduino.



### *AnalogReadSerial*

*Reads an analog input on pin 0, prints the result to the serial monitor. Attach the center pin of a potentiometer to pin A0, and the outside pins to +5V and ground.*

*This example code is in the public domain.*

```
*/

// the setup routine runs once when you press reset:
void setup() {
  // initialize serial communication at 9600 bits per second:
  Serial.begin(9600);
}

// the loop routine runs over and over again forever:
void loop() {
  // read the input on analog pin 0:
  int sensorValue = analogRead(A0);
  // print out the value you read:
  Serial.println(sensorValue);
  delay(1);        // delay in between reads for stability
}
```





(https://www.parallax.com)

Propeller +Education +Support +ShopAbout Parallax +Community +Sales +

Search...

Home (https://www.parallax.com) / Sensors (https://www.parallax.com/product-category/sensors/) / Rangefinders (https://www.parallax.com/product-category/sensors/rangefinders/) / PING))) Ultrasonic Distance Sensor

Catalog

Propeller 1

(https://www.parallax.com/product-category/propeller-1/)

(17)

Propeller 2

(https://www.parallax.com/product-category/propeller-2/)

(52)

BASIC Stamp

(https://www.parallax.com/product-category/basic-stamp/)

(26)

Shield For Arduino

(https://www.parallax.com/product-category/shield-for-arduino/)

(9)

Micro:bit

(https://www.parallax.com/product-category/microbit/)

(16)

ActivityBot 360°

(https://www.parallax.com/product-category/activitybot-360/)

(29)

Boe-Bot (https://www.parallax.com/product-category/boe-bot/)

(26)

Cyber:bot

(https://www.parallax.com/product-category/cyberbot/)

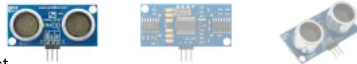
(28)

Shield-Bot

(https://www.parallax.com/product-category/shield-bot/)

(25)

SumoBot WX (https://www.parallax.com/product-



# PING))) Ultrasonic Distance Sensor

SKU 28015

\$34.95

213 in stock

Quantity:

Add to cart

### Quantity Discounts

Quantity	Price
1 - 4	\$34.95
5 - 19	\$33.21
20+	\$31.46

Description	Downloads	Discussion (0)
-------------	-----------	----------------

## Product Description

### PING))) Ultrasonic Distance Sensor Overview:

Our PING)))™ ultrasonic sensor provides an easy method of distance measurement. This sensor is perfect for any number of applications that require you to perform measurements between moving or stationary objects.

Interfacing to a microcontroller is a snap. A single I/O pin is used to trigger an ultrasonic burst (well above human hearing) and then "listen" for the echo return pulse. The sensor measures the time required for the echo return, and returns this value to the microcontroller as a variable-width pulse via the same I/O pin.

The PING))) Ultrasonic Distance Sensor is compatible with all Parallax educational robots (https://www.parallax.com/education/robotics/). It is also supported by many bonus projects (https://learn.parallax.com/tutorials/hardware/accessories/ping) for BASIC Stamp, Arduino, Propeller, and micro:bit on our Learn site.

Category/Sumobot-Wx/)

(11)

All Robotics

(<https://www.parallax.com/Product-Category/All-Robotics/>)

Category/All-Robotics/)

(57)

Sensors

(<https://www.parallax.com/Product-Category/Sensors/>)

Category/Sensors/)

(75)

Kits

(<https://www.parallax.com/Product-Category/Sensors/S->

Category/Sensors/S-

Kits/)

Temperature/Humidity

(<https://www.parallax.com/Product-Category/Sensors/Temperature->

Category/Sensors/Temperature-

Humidity/)

Pressure/Flex/RPM

(<https://www.parallax.com/Product-Category/Sensors/Pressure->

Category/Sensors/Pressure-

Flex-Rpm/)

Proximity/Motion

## Related Products

Category/Sensors/Proximity-

Motion/)

Infrared

(<https://www.parallax.com/Product-Category/Sensors/Infrared/>)

Category/Sensors/Infrared/)

(11)

Human Input Devices

(<https://www.parallax.com/Product-Category/Sensors/Human->

Category/Sensors/Human-

Input-Devices/)

GPS

(<https://www.parallax.com/Product-Category/Sensors/Gps/>)

### LaserPING 2m Rangefinder

Category/Sensors/Gps/)

\$29.99

(2)

(<https://www.parallax.com/product/laserping-2m-rangefinder/>)

(<https://www.parallax.com/Product-Category/Sensors/Gas/>)

Add to cart (?add-to-cart=1761)

Category/Sensors/Gas/)

(6)

Color/Light

(<https://www.parallax.com/Product-Category/Sensors/Color->

Category/Sensors/Color-

Light/)

### Key Features:

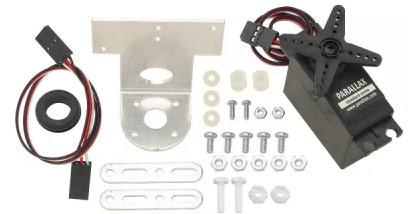
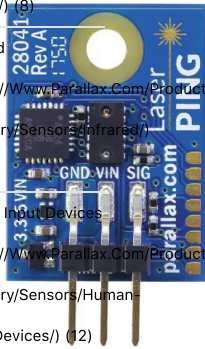
- Provides precise, non-contact distance measurements within a 3 cm to 3 m range
- Ultrasonic measurements work in any lighting condition, making this a good choice to supplement infrared object detectors
- Simple pulse in/pulse out communication requires just one I/O pin
- Burst indicator LED shows measurement in progress
- 3-pin header makes it easy to connect to a development board, directly or with an extension cable, no soldering required

### Application Ideas:

- Security systems
- Interactive animated exhibits
- Parking assistant systems
- Robotic navigation

### PING))) Ultrasonic Distance Sensor Specifications:

- Narrow acceptance angle
- Range: approximately 1 inch to 10 feet (3 cm to 3 m)
- 3-pin male header with 0.1" spacing
- Power requirements: +5 VDC; 35 mA active
- Communication: positive TTL pulse
- Dimensions: 0.81 x 1.8 x 0.6 in (22 x 46 x 16 mm)
- Operating temperature range: +32 to +158 °F (0 to +70 °C)



### PING))) Protector Stand

\$8.99

(<https://www.parallax.com/product/ping-protector-stand/>)

Add to cart (?add-to-cart=1635)

### PING))) Mounting Bracket Kit

\$24.99

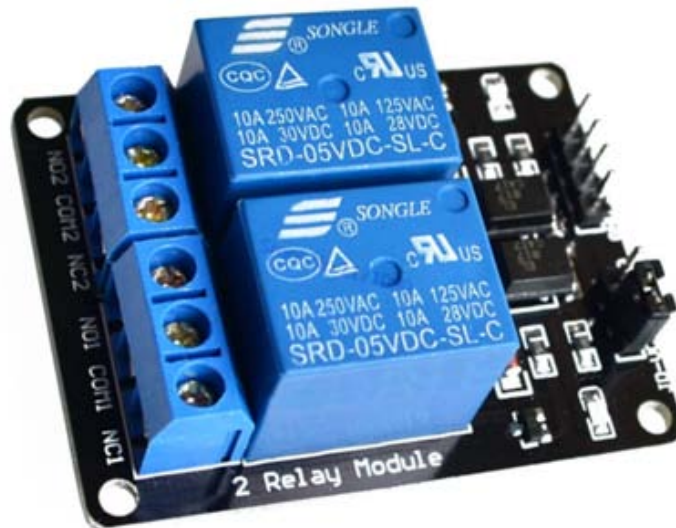
(<https://www.parallax.com/product/ping-mounting-bracket-kit/>)

Add to cart (?add-to-cart=1564)

## User Guide

### 2 Channel 5V Optical Isolated Relay Module

This is a LOW Level 5V 2-channel relay interface board, and each channel needs a 15-20mA driver current. It can be used to control various appliances and equipment with large current. It is equipped with high-current relays that work under AC250V 10A or DC30V 10A. It has a standard interface that can be controlled directly by microcontroller. This module is optically isolated from high voltage side for safety requirement and also prevent ground loop when interface to microcontroller.



#### Brief Data:

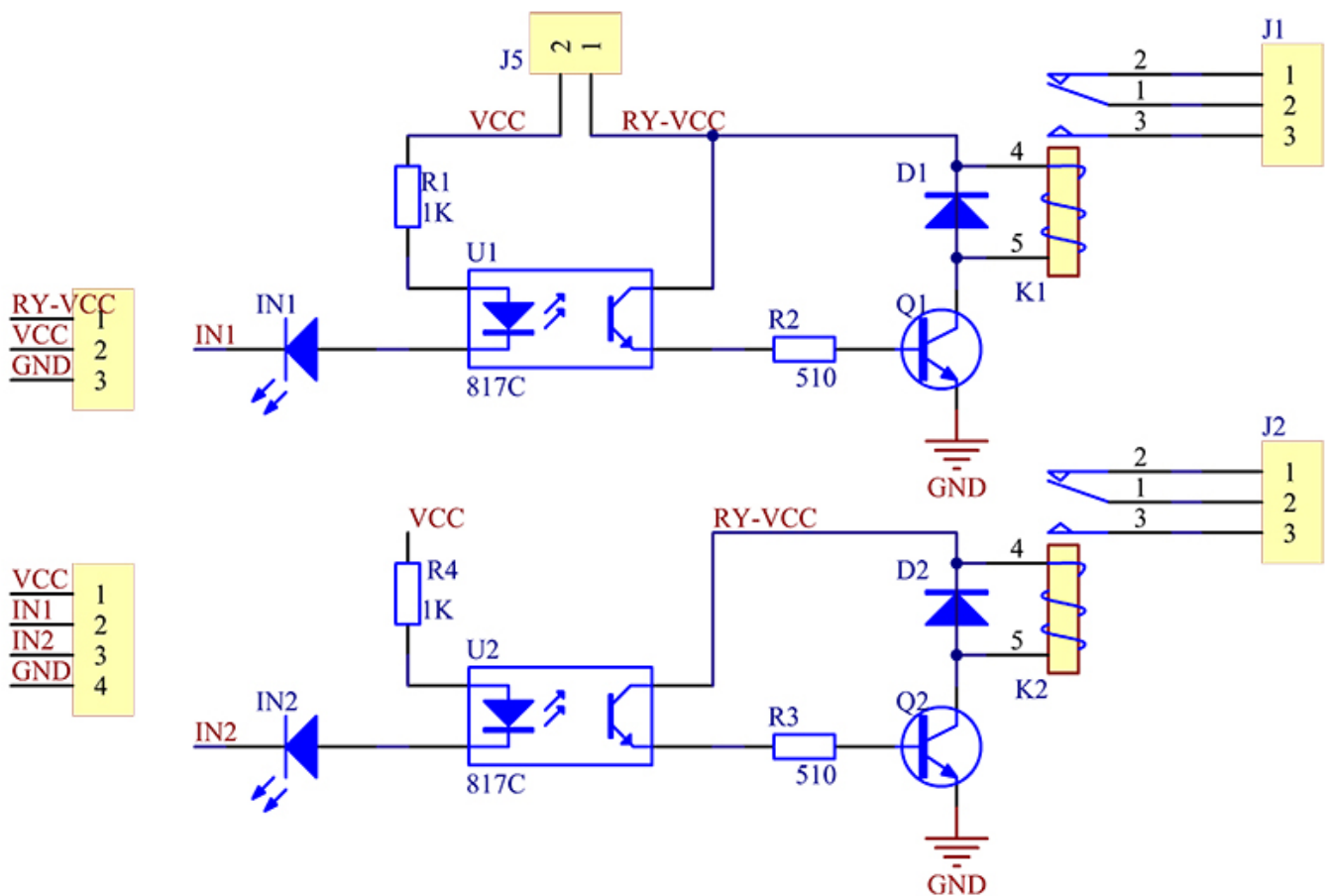
- Relay Maximum output: DC 30V/10A, AC 250V/10A.
- 2 Channel Relay Module with Opto-coupler. LOW Level Trigger expansion board, which is compatible with Arduino control board.
- Standard interface that can be controlled directly by microcontroller ( 8051, AVR, \*PIC, DSP, ARM, ARM, MSP430, TTL logic).
- Relay of high quality low noise relays SPDT. A common terminal, a normally open, one normally closed terminal.
- Opto-Coupler isolation, for high voltage safety and prevent ground loop with microcontroller.

## Schematic:

VCC and RY-VCC are also the power supply of the relay module. When you need to drive a large power load, you can take the jumper cap off and connect an extra power to RY-VCC to supply the relay; connect VCC to 5V of the MCU board to supply input signals.

NOTES: If you want complete optical isolation, connect "Vcc" to Arduino +5 volts but do NOT connect Arduino Ground. Remove the Vcc to JD-Vcc jumper. Connect a separate +5 supply to "JD-Vcc" and board Gnd. This will supply power to the transistor drivers and relay coils.

If relay isolation is enough for your application, connect Arduino +5 and Gnd, and leave Vcc to JD-Vcc jumper in place.

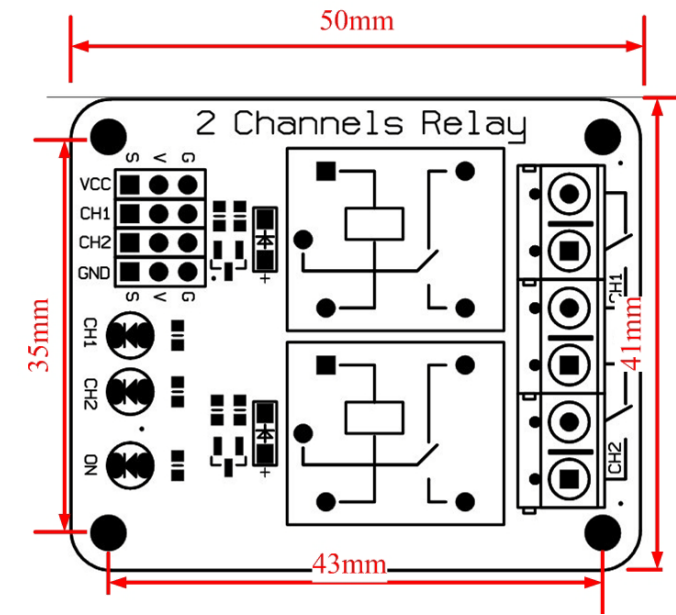


It is sometimes possible to use this relay boards with 3.3V signals, if the JD-VCC (Relay Power) is provided from a +5V supply and the VCC to JD-VCC jumper is removed. That 5V relay supply could be totally isolated from the 3.3V device, or have a common ground if opto-isolation is not needed. If used with isolated 3.3V signals, VCC (To the input of the opto-isolator, next to the IN pins) should be connected to the 3.3V device's +3.3V supply.

NOTE: Some Raspberry-Pi users have found that some relays are reliable and others do not actuate sometimes. It may be necessary to change the value of R1 from 1000 ohms to something like 220 ohms, or supply +5V to the VCC connection.

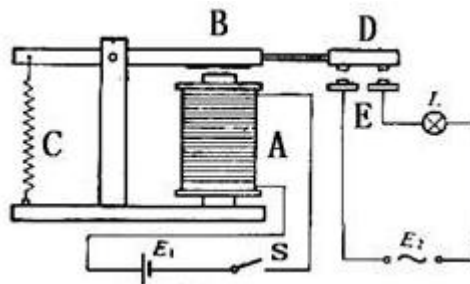
NOTE: The digital inputs from Arduino are Active LOW: The relay actuates and LED lights when the input pin is LOW, and turns off on HIGH.

### Module Layout:



### Operating Principle:

See the picture below: A is an electromagnet, B armature, C spring, D moving contact, and E fixed contacts. There are two fixed contacts, a normally closed one and a normally open one. When the coil is not energized, the normally open contact is the one that is off, while the normally closed one is the other that is on.



Supply voltage to the coil and some currents will pass through the coil thus generating the electromagnetic effect. So the armature overcomes the tension of the spring and is attracted to the core, thus closing the moving contact of the armature and the normally open (NO) contact or you may say releasing the former and the normally closed (NC) contact. After the coil is de-energized, the electromagnetic force disappears and the armature moves back to the original position, releasing the moving contact and normally closed contact. The closing and releasing of the contacts results in power on and off of the circuit.

### Input:

VCC : Connected to positive supply voltage (supply power according to relay voltage)

GND : Connected to supply ground.

IN1: Signal triggering terminal 1 of relay module

IN2: Signal triggering terminal 2 of relay module

### **Output:**

Each module of the relay has one NC (normally close), one NO (normally open) and one COM (Common) terminal. So there are 2 NC, 2 NO and 2 COM of the channel relay in total. NC stands for the normal close port contact and the state without power. NO stands for the normal open port contact and the state with power. COM means the common port. You can choose NC port or NO port according to whether power or not.

### **Testing Setup:**

When a low level is supplied to signal terminal of the 2-channel relay, the LED at the output terminal will light up. Otherwise, it will turn off. If a periodic high and low level is supplied to the signal terminal, you can see the LED will cycle between on and off.

#### For Arduino:

Step 1:

Connect the signal terminal IN1、IN2 of 2-channel relay to digital pin 4 & 5 of the Arduino Uno or ATmega2560 board, and connect an LED at the output terminal.

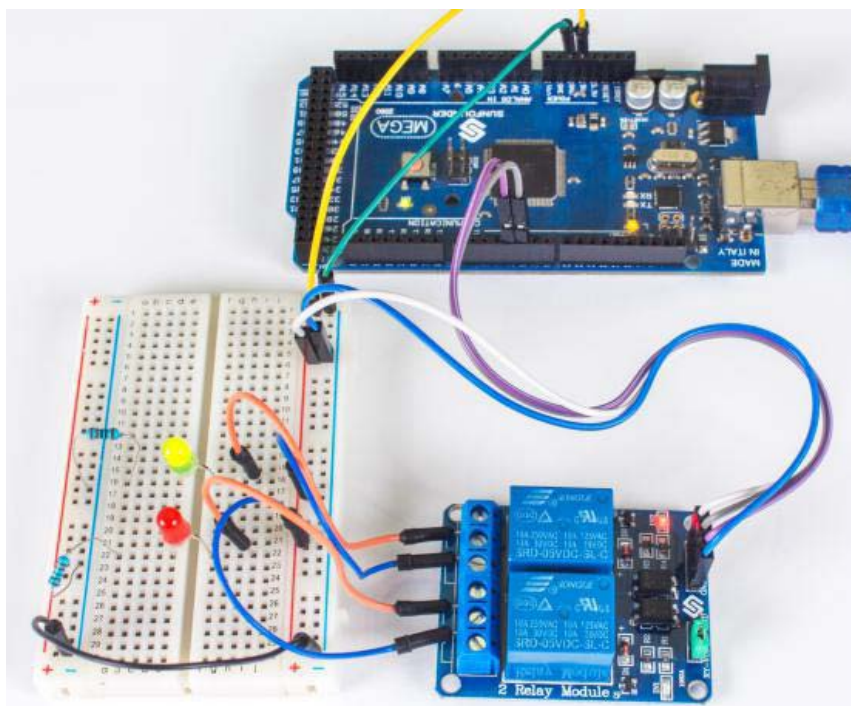
IN1> 4

IN2> 5

Step 2:

Upload the sketch "text\_code" to the Arduino Uno or ATmega2560 board. Then you can see the LED cycle between on and off.

The actual figure is shown below:



For raspberry Pi:



### Step1:

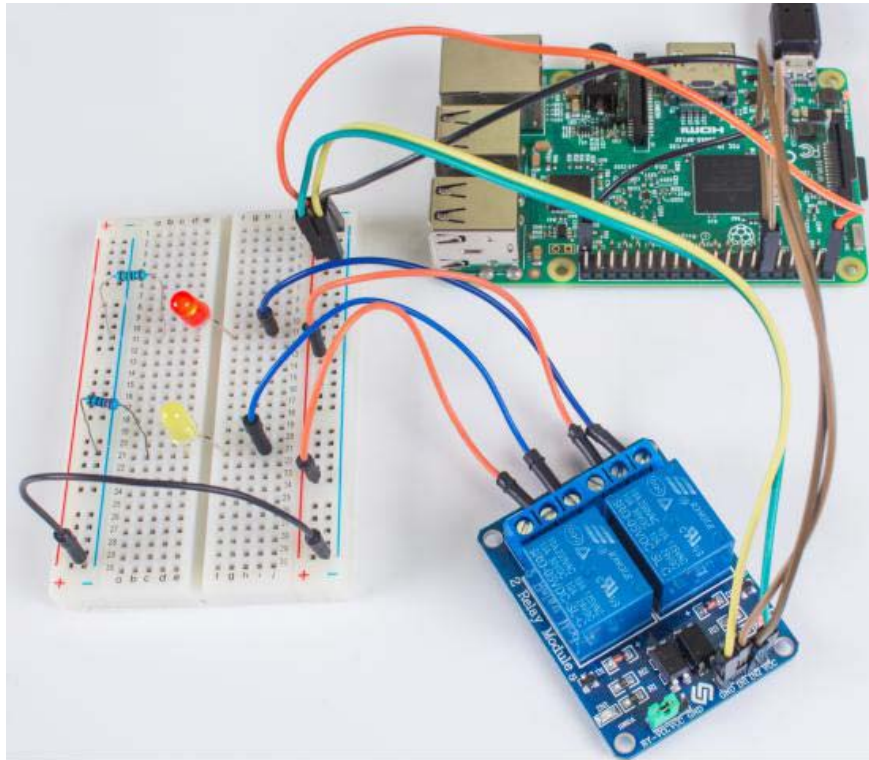
Connect the signal terminal IN2、 IN1 of 2-channel relay to port 17、 18 of the Raspberry Pi, and connect an LED at the output terminal.

IN2 > 17

IN1 > 18

### Step 2:

Run the “test\_code”. Then you can see the LED cycle between on and off.



### Sketch for Arduino:

```
/*
Name: _2_channel_relay
Description: control the 2 channel relay module to ON or OFF
Website: www.handsontec.com
Email: techsupport@handsontec.com
*/

//the relays connect to
int IN1 = 4;
int IN2 = 5;

#define ON 0
#define OFF 1

void setup()
{
  relay_init();//initialize the relay
}

void loop() {
  relay_SetStatus(ON, OFF);//turn on RELAY_1
}
```

```

delay(2000);//delay 2s
relay_SetStatus(OFF, ON);//turn on RELAY_2
delay(2000);//delay 2s
}
void relay_init(void)//initialize the relay
{
    //set all the relays OUTPUT
    pinMode(IN1, OUTPUT);
    pinMode(IN2, OUTPUT);
    relay_SetStatus(OFF, OFF); //turn off all the relay
}
//set the status of relays
void relay_SetStatus( unsigned char status_1, unsigned char status_2)
{
    digitalWrite(IN1, status_1);
    digitalWrite(IN2, status_2);
}

```

## Code for Raspberry Pi:

```

#!/usr/bin/env python
'''
*****
* Filename      : 2_channel_relay.py
* Description   : a sample script for 2-Channel High trigger Relay
* E-mail       : techsupport@handsontec.com
* Website      : www.handsontec.com
* Detail       : New file
*****
'''
import RPi.GPIO as GPIO
from time import sleep

Relay_channel = [17, 18]

def setup():
    GPIO.setmode(GPIO.BOARD)
    GPIO.setup(Relay_channel, GPIO.OUT, initial=GPIO.LOW)
    print "|=====|"
    print "|          2-Channel High trigger Relay Sample          |"
    print "|-----|"
    print "|          Turn 2 channels on off in orders          |"
    print "|          |"
    print "|          17 ==> IN2          |"
    print "|          18 ==> IN1          |"
    print "|          |"
    print "|          |"
    print "|=====|"

def main():
    while True:
        for i in range(0, len(Relay_channel)):
            print '...Relay channel %d on' % i+1
            GPIO.output(Relay_channel[i], GPIO.HIGH)
            sleep(0.5)
            print '...Relay channel %d off' % i+1
            GPIO.output(Relay_channel[i], GPIO.LOW)
            sleep(0.5)

def destroy():
    GPIO.output(Relay_channel, GPIO.LOW)
    GPIO.cleanup()

```



```
if __name__ == '__main__':  
    setup()  
    try:  
        main()  
    except KeyboardInterrupt:  
        destroy()
```



**154**  
ADAFRUIT

[Buy Now](#)



Looking for a discount?

[Check out our current promotions!](#)

Give us a call

**1-855-837-4225**

International: 1-415-281-3866

## Email Us

Sales and New Orders: [sales@verical.com](mailto:sales@verical.com)

Order Support: [support@verical.com](mailto:support@verical.com)

Suppliers: [Visit our seller page](#)

## Company Address

Arrow Electronics, Inc  
9201 East Dry Creek Road  
Centennial, CO 80112



## Continuous Rotation Servo - FeeTech FS5103R

PRODUCT ID: 154

IN STOCK

1

ADD TO CART

 Also include 1 x [Continuous Rotation Servo Wheel \(\)](#)

1-9

10-99

100+

ADD TO WISHLIST

[DESCRIPTION](#)[TECHNICAL DETAILS](#)

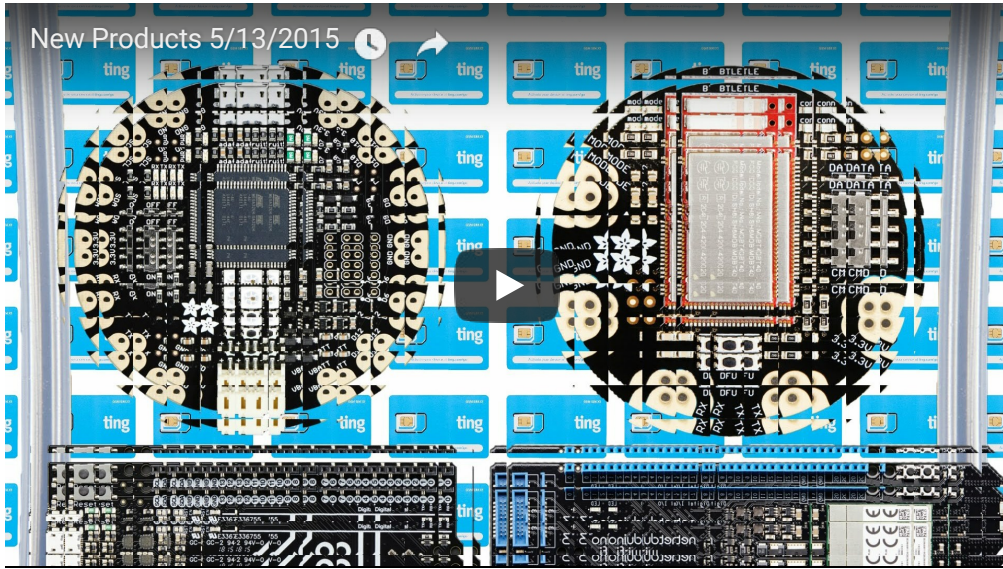
## DESCRIPTION

This servo rotates fully forward or backwards instead of moving to a position. You can use any servo code, hardware or library to control these servos. Good for making simple moving robots. Comes with four different horns, as shown.

To control with an Arduino, we suggest connecting the control wire to pin 9 or 10 and using the [Servo library](#) included with the Arduino IDE ([see here for an example sketch](#)). Position "90" (1.5ms pulse) is stop, "180" (2ms pulse) is full speed forward, "0" (1ms pulse) is full speed backwards. They may require some simple calibration, simply tell the servo to 'stop' and then

gently adjust the potentiometer in the recessed hole with a small screwdriver until the servo stops moving.

**Note:** This product no longer includes the hole to adjust the Zero point.



## TECHNICAL DETAILS

Details:

- Operating Voltage: 4.8V~6V (5V works best)
- Average Speed: ~0.18sec/60°
- Stall Torque (4.8V): 3kg.cm/41.74oz.in
- Stall Torque (6V): 3.2kg.com/44.52oz.in
- Required Pulse: 500us-2500us
- Connector Wire Length: 30cm / 11.8"
- Dimensions: 37mm x 54mm x 20mm / 1.5" x 2.1" x 0.8"
- Weight (no horns): 40g
- Spline Count: 25

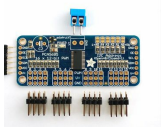
Revision History:

- As of Friday, May 8th, 2015 we're selling a revised FeeTech version with additional horns!

## LEARN



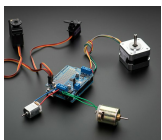
[Adafruit 16 Channel Servo Driver with Raspberry Pi](#)



[Adafruit PCA9685 16-Channel Servo Driver](#)



[LSM303 Accelerometer + Compass Breakout](#)  
Triple Axis Accelerometer and Magnetometer (Compass) breakout board.



[Adafruit Motor Shield V2 for Arduino](#)  
Stackable, high current DC and Stepper motor shield for Arduino





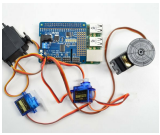
### [Modifying Servos for Continuous Rotation](#)

Make tiny gear-motors for your next robot!



### [Adafruit Motor Selection Guide](#)

Choose the right motor (and controller) for the job!



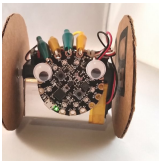
### [Adafruit 16-Channel PWM/Servo HAT & Bonnet for Raspberry Pi](#)

16 channels of servo-bustin' power for your Pi



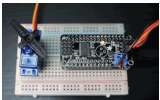
### [Adafruit 8-Channel PWM or Servo FeatherWing](#)

A 8 x servo party for your Feather!



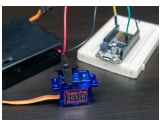
### [Circuit Playground Sound-Controlled Robot](#)

Make your Circuit Playground mobile and change direction via sound



### [CircuitPython Hardware: PCA9685 PWM & Servo Driver](#)

How to use the PCA9685 PWM & servo driver with CircuitPython!



### [Using Servos With CircuitPython and Arduino](#)

How to use servo motors with CircuitPython and Arduino



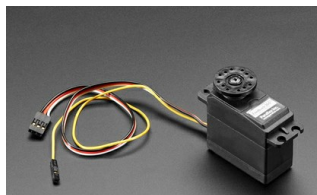
### [Spinning Logo](#)

Build a paper craft hack a day logo powered by Circuit Playground!

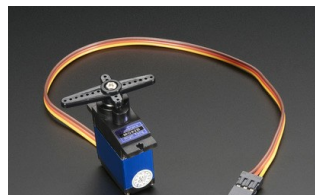
## MAY WE ALSO SUGGEST...



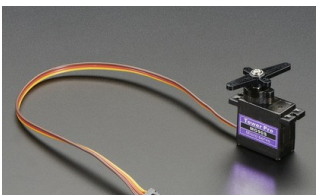
Continuous Rotation Servo



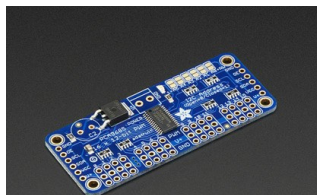
Feedback 360 Degree -



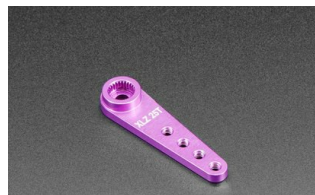
Micro Servo - High



Micro Servo - MG90S High



Adafruit 16-Channel 12-bit



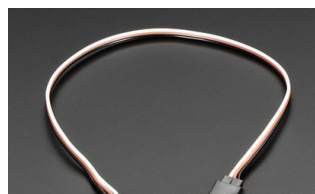
Machined Aluminum Servo



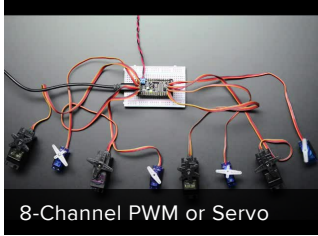
Continuous Rotation Micro



Ball Caster - 3/4" Metal Ball



Servo Extension Cable -



8-Channel PWM or Servo



Analog Feedback Servo



Standard Size - High Torque

## DISTRIBUTORS [EXPAND TO SEE DISTRIBUTORS](#)

[CONTACT](#)

[SUPPORT](#)

[DISTRIBUTORS](#)

[EDUCATORS](#)

[JOBS](#)

[FAQ](#)

[SHIPPING & RETURNS](#)

[TERMS OF SERVICE](#)

[PRIVACY & LEGAL](#)

[ABOUT US](#)

ENGINEERED IN NYC Adafruit®

*"Far and away the best prize that life has to offer is the chance to work hard at work worth doing" - Theodore Roosevelt*



4.9 ★★★★★  
Google  
Customer Reviews

**PJRC Store**

- [Teensy 4.1, \\$26.85](#)
- [Teensy 4.0, \\$19.95](#)
- [Teensy 3.6, \\$29.25](#)
- [Teensy 3.5, \\$24.25](#)
- [Teensy 3.2, \\$19.80](#)
- [Teensy LC, \\$11.65](#)
- [Teensy 2.0, \\$16.00](#)

# Teensy Technical Specifications

**Teensy**

- [Main Page](#)
- [Hardware](#)
  - [Tech Specs](#)
  - [Teensy 3.2 / 3.1](#)
  - [Teensy-LC](#)
- [Getting Started](#)
- [Tutorial](#)
- [How-To Tips](#)
- [Code Library](#)
- [Projects](#)
- [Teensyduino](#)
- [Reference](#)

Feature	Teensy 2.0	Teensy++ 2.0	Teensy LC	Teensy 3.2	Teensy 3.5	Teensy 3.6	Teensy 4.0	Teensy 4.1	Units
Price	<a href="#">\$16.00</a>	<a href="#">\$24.00</a>	<a href="#">\$11.65</a>	<a href="#">\$19.80</a>	<a href="#">\$24.25</a>	<a href="#">\$29.25</a>	<a href="#">\$19.95</a>	<a href="#">\$26.85</a>	US Dollars
Processor Core	ATMEGA32U4 <a href="#">AVR</a>	AT90USB1286 <a href="#">AVR</a>	MKL26Z64VFT4 <a href="#">Cortex-M0+</a>	MK20DX256VLH7 <a href="#">Cortex-M4</a>	MK64FX512VMD12 <a href="#">Cortex-M4F</a>	MK66FX1M0VMD18 <a href="#">Cortex-M4F</a>	IMXRT1062DVL6 <a href="#">Cortex-M7</a>	IMXRT1062DVJ6 <a href="#">Cortex-M7</a>	
FPU	-	-	-	72	120	32	32 & 64	32 & 64	bits
Rated Speed	16	16	48	72	120	180	600	600	MHz
Overclockable	-	-	-	96	-	240	912	912	MHz
Flash Memory	31.5	127	62	256	512	1024	1984	7936	kbytes
Bandwidth	32	32	96	192	192	411	66	66	Mbytes/sec
Cache	-	-	64	256	256	8192	65536	65536	Bytes
RAM	2.5	8	8	64	256	256	1024	1024	kbytes
EEPROM	1024	4096	128 (emu)	2048	4096	4096	1080 (emu)	4284 (emu)	bytes
Direct Memory Access	-	-	4	16	16	32	32	32	Channels
Digital I/O	25	46	27	34	58	58	40	55	Pins
Breadboard I/O	22	36	24	24	40+2	40+2	24	42	Pins
Voltage Output	5V	5V	<a href="#">3.3V / 5V</a>	3.3V	3.3V	3.3V	3.3V	3.3V	Volts
Current Output	20mA	20mA	<a href="#">5mA / 20mA</a>	10mA	10mA	10mA	10mA	10mA	milliAmps
Voltage Input	5V	5V	<a href="#">3.3V Only</a>	5V Tolerant	5V Tolerant	3.3V Only	3.3V Only	3.3V Only	Volts
Interrupts	4	8	18	34	58	58	40	55	Pins
Analog Input	12	8	13	21	27	25	14	18	Pins
Converters	1	1	1	2	2	2	2	2	
Usable Resolution	10	10	12	13	13	13	10	10	Bits
Prog Gain Amp	1	1	-	2	-	-	-	-	
Touch Sensing	-	-	11	12	-	11	-	-	Pins
Comparators	1	1	1	3	3	4	4	4	
Analog Output	-	-	1	1	2	2	-	-	Pins
DAC Resolution	-	-	12	12	12	12	-	-	Bits
Timers	4 Total	4 Total	7 Total	12 Total	17 Total	19 Total	49 Total	49 Total	
PWM, 32 bit	-	-	3	-	-	-	3	3	
PWM, 16 bit	2	2	3	3	4	6	32	32	
PWM, 8-10 bit	2	2	-	-	-	-	-	-	
Total PWM Outputs	7	9	10	12	20	22	27	31	Pins
PDB Type	-	-	-	1	1	1	-	-	
CMT Type	-	-	-	1	1	1	-	-	
Quadrature Enc	-	-	-	-	-	-	4	4	
LPTMR Type	-	-	1	1	1	1	-	-	
PIT/Interval	-	-	2	4	4	4	4	4	
IEEE 1588	-	-	-	-	4	4	4	4	
Systick	-	-	1	1	1	1	1	1	
RTC	-	-	<a href="#">0**</a>	<a href="#">1**</a>	1	1	1	1	
Communication									
USB	1	1	1	1	1	2	2	2	
Serial	1	1	3	3	6	6	7	8	
With FIFOs	-	-	-	2	2	2	7	8	
High Res Baud	-	-	-	3	6	5	-	-	
SPI	1	1	2	1	3	3	2	2	
With FIFOs	-	-	1	1	1	1	2	2	
I2C	1	1	2	2	3	4	3	3	
CAN Bus	-	-	-	1	1	2	3	3	
With CAN-FD	-	-	-	-	-	-	1	1	
Digital Audio In	-	-	1	2	2	2	5*	5*	stereo pins
Digital Audio Out	-	-	1	2	2	2	5*	5*	stereo pins
S/PDIF Input	-	-	-	-	-	-	1	1	
S/PDIF Output	-	-	-	0*	0*	0*	1	1	
MQS Output	-	-	-	-	-	-	1	1	
SD Card	-	-	-	-	1	1	1*	1	
Ethernet	-	-	-	-	1*	1*	-	1	uu