THE HOT WHEELS CAR RACE

REPORT 3: Final Report

GROUP MEMBERS :

Kalki Bhavsar	AU1841029
Mansi Dobariya	AU1841131
Nirva Sangani	AU1841074

Guided by: Prof. Anurag Lakhani

Course:

Embedded System Design,
School of Engineering and Applied Science,
Winter 2020.

SUMMARY :

Motivation :

Kids, teens or adults; the people of all ages find the fast and furious HotWheels Cars fascinating to play with. The center motive of the kids while playing is **“whose car is the fastest among the group”**. To solve the problem, we decided to make a setup and an embedded system that basically helps two kids to race their cars and decide the winner. **Our main focus is making the system with fun parts like LCD display, buzzer, lighting on the declaration of winner, etc so as to make the kids enjoy the race in their real time.**

Description :

Our setup will be made up of the 2 HotWheels racing tracks with the starting and ending lines. Two cars will be able to run on these 2 non-intersecting tracks at a time. Initially, the 2 kids will keep their car before the starting line. A barrier is kept after the starting line. T

To start the race, the kids will have to start the system by pushing the push button and counting will be started from 3.. 2.. 1... 0(GO) in the LCD/7-segment display. If in case, before the counting ends, if any of the 2 cars is detected before the starting line, an alert message will be displayed on the LCD screen and a buzzer will ring to indicate the same. The barrier won’t open in this case.

If the cars are all set in their positions, then and only the barrier will open after count = 0. The competitors will then give a push to the car and the cars will race on the track. One the way, there will be 2 sensors attached at a fixed distance near the track which will detect the presence of the car.

We have a fixed distance calculated while setting up the sensors. The code will measure the time interval when the car passes between the two IR sensors. With the help of the distance formula, the code dumped in the arduino will calculate the speed of each of the cars and it will be displayed on the LCD screen. In the end, the kid with the greater speed will be displayed as the winner on the LCD screen. We will be using LEDs for the celebration of the winner for fun purposes.

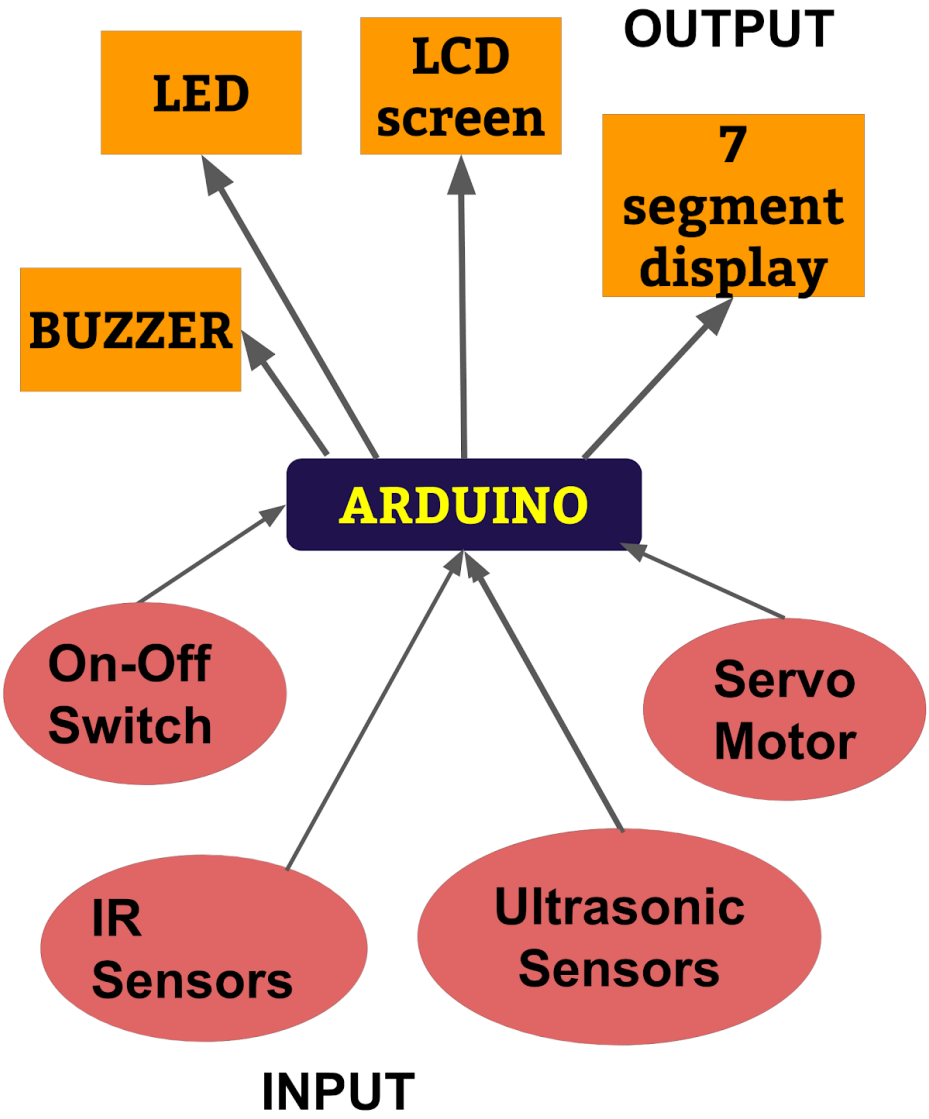
Final Outcomes:

A fun HotWheels Car racing setup made for kids to make them know whose car is the fastest with an interesting enjoyable process from starting till the end. The kids will be able to know which car is the fastest and by what amount (measured in terms of velocity).

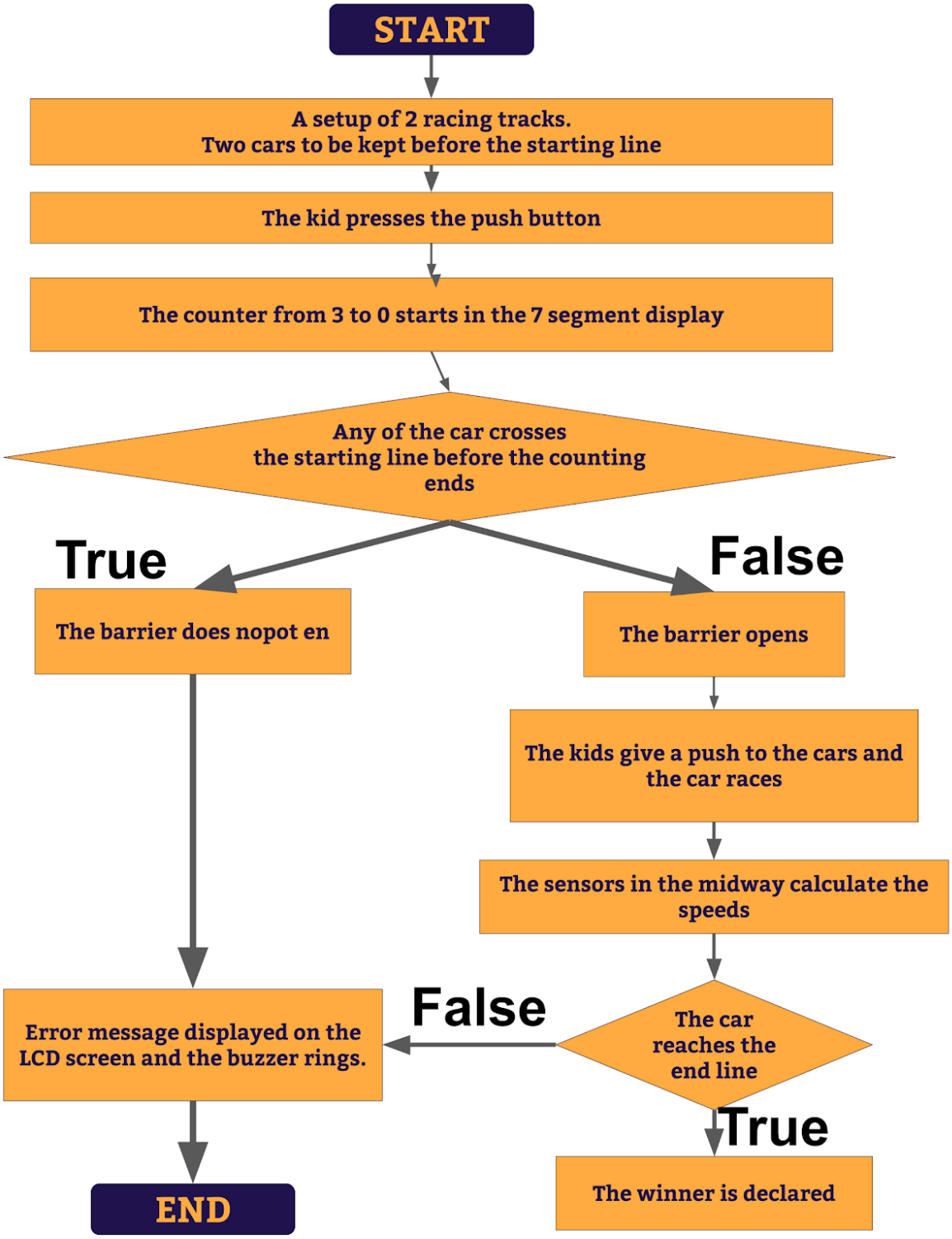
Components:

Sr. no	Component Name	Quantity	Selection Criteria	Type
1	Arduino Mega	1	The main microcontroller	Controller
2	5V: Battery	1	To provide power/ input voltage to the microcontroller	Input
3	Ultrasonic sensor	2	For the detection of car presence before the starting line	Input
4	IR sensors	4	For sensing the car presence	Input
5	Rocker Switches	1	To On/Off the system	Input
6	Speaker	1	For speaking the instructions and interaction	Output
7	LCD 20x4 with 12 C module	1	For displaying messages, alerts, instructions, etc.	Output
8	Step motor/ Servo Motor	4	For rotation of the barrier connected with the lever of the step motor	Output
9	LED	As required	As indicators, celebration lights	Output
10	Buzzer	1	As a alert indicator of crossing the starting line before the starting of the race	Output
11	7 segment display	1	For displaying the initial counting before the starting of the race	Output
12	Breadboard	3	For circuit base	Connectors
13	All types of Jumper wires	2-sets each	For connections	Connectors

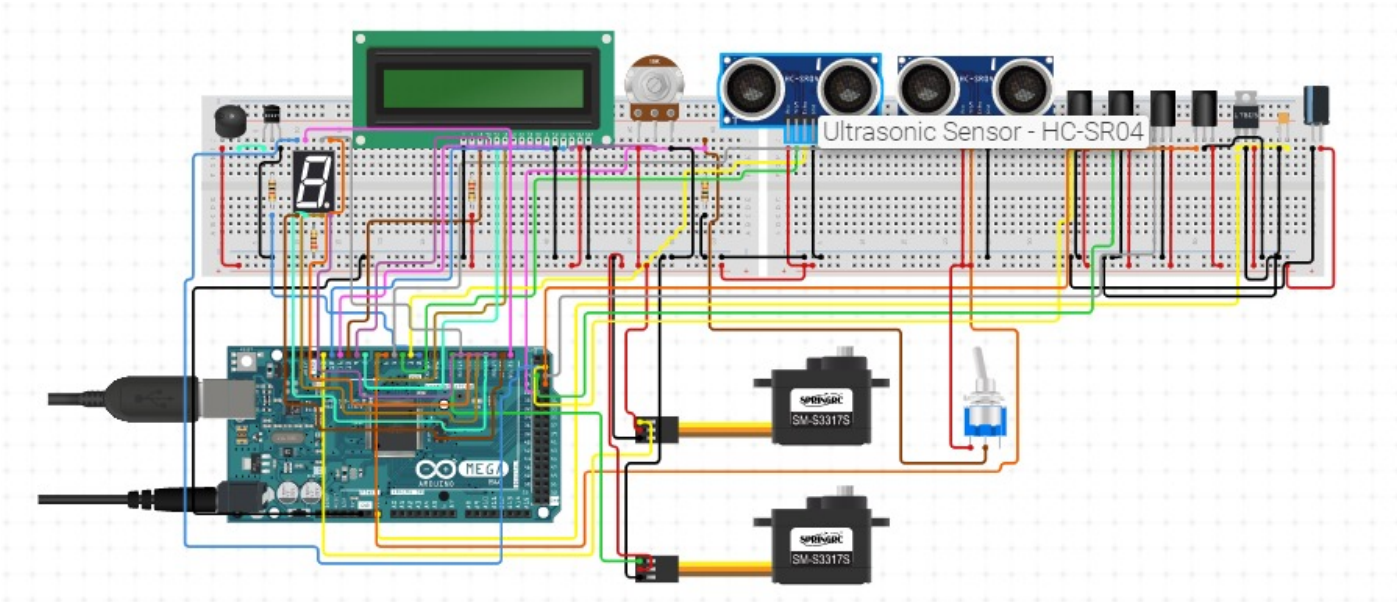
BLOCK DIAGRAM:



FLOWCHART :

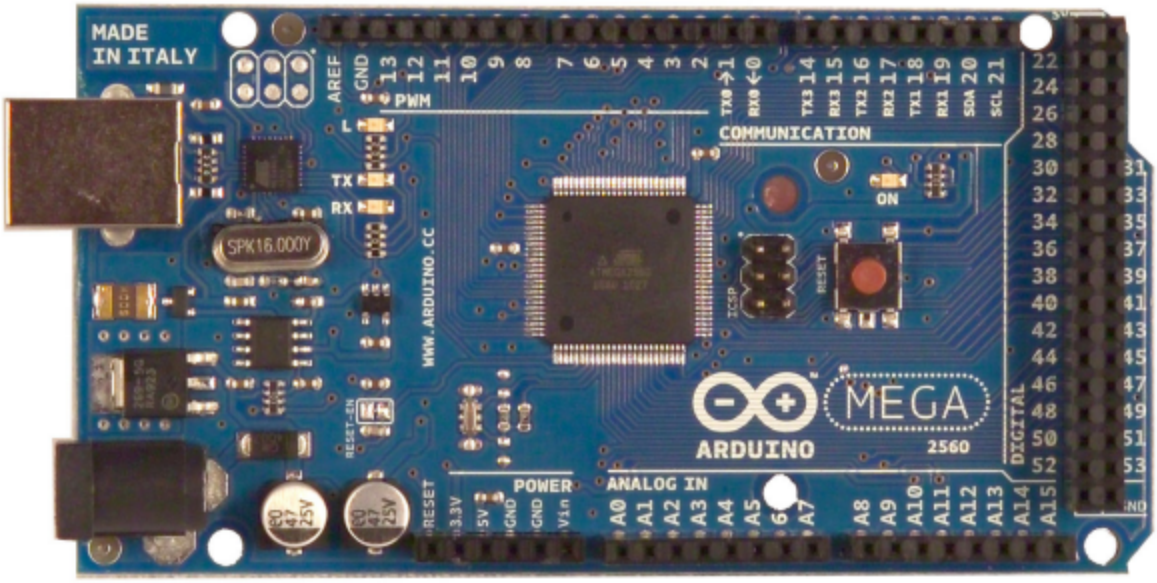

START
A setup of 2 racing tracks.
Two cars to be kept before the starting line
The kid presses the push button
The counter from 3 to 0 starts in the 7 segment display
Any of the car crosses the starting line before the counting ends
True
False
The barrier does nopot en
The barrier opens
The kids give a push to the cars and the car races
The sensors in the midway calculate the speeds
The car reaches the end line
False
True
Error message displayed on the LCD screen and the buzzer rings.
The winner is declared
END

Circuit Diagram:



Data-Sheets of All Components :

Arduino Mega



Memory

The ATmega2560 has 256 KB of flash memory for storing code (of which 8 KB is used for the bootloader), 8 KB of SRAM and 4 KB of EEPROM (which can be read and written with the EEPROM library).

Input and Output

Each of the 54 digital pins on the Mega can be used as an input or output, using pinMode() , digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 kOhms. In addition, some pins have specialized functions:

- **Serial: 0 (RX) and 1 (TX); Serial 1: 19 (RX) and 18 (TX); Serial 2: 17 (RX) and 16 (TX); Serial 3: 15 (RX) and 14 (TX).** Used to receive (RX) and transmit (TX) TTL serial data. Pins 0 and 1 are also connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.
- **External Interrupts: 2 (interrupt 0), 3 (interrupt 1), 18 (interrupt 5), 19 (interrupt 4), 20 (interrupt 3), and 21 (interrupt 2).** These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attachInterrupt() function for details.
- **PWM: 0 to 13.** Provide 8-bit PWM output with the analogWrite() function.
- **SPI: 50 (MISO), 51 (MOSI), 52 (SCK), 53 (SS).** These pins support SPI communication using the SPI library. The SPI pins are also broken out on the ICSP header, which is physically compatible with the Uno, Duemilanove and Diecimila.
- **LED: 13.** There is a built-in LED connected to digital pin 13. When the pin is HIGH

LCD 12x2 Character

12 x 2 Character LCD

FEATURES

- Type: character
- Display format: 12 x 2 characters
- Built-in controller: ST7066 or equivalent
- Duty cycle: 1/16
- 5 x 8 dots includes cursor
- +5 V power supply
- LED can be driven by pin 1, pin 2, or A
- Material categorization: for definitions of compliance please see www.vishay.com/doc?99912

RoHS COMPLIANT

MECHANICAL DATA		
ITEM	STANDARD VALUE	UNIT
Module dimension	55.7 x 32.0 x 14.5 (max.)	mm
Viewing area	46.0 x 14.5	
Dot size	0.45 x 0.60	
Dot pitch	0.55 x 0.70	
Mounting hole	31.2 x 29.0	
Character size	2.65 x 5.50	

ABSOLUTE MAXIMUM RATINGS					
ITEM	SYMBOL	STANDARD VALUE			UNIT
		MIN.	TYP.	MAX.	
Supply voltage for logic	V_{DD} to V_{SS}	-0.3	-	7.0	V
Supply voltage for LCD	V_{DD} to V_0	-0.3	-	13.0	
Input voltage	V_I	V_{SS}	-	V_{DD}	
Operating temperature	T_{OP}	-20	-	+70	°C
Storage temperature	T_{ST}	-30	-	+80	°C

Note

- V_{SS} = 0 V, V_{DD} = 5.0 V

ELECTRICAL CHARACTERISTICS						
ITEM	SYMBOL	CONDITION	STANDARD VALUE			UNIT
			MIN.	TYP.	MAX.	
Supply voltage for logic	V_{DD} to V_{SS}	-	4.5	5.0	5.5	V
Supply voltage for LCD (1)	V_{DD} to V_0	-20 °C	-	-	5.7	V
		25 °C	-	4.2	-	
		70 °C	3.5	-	-	
Input high voltage	V_{IH}	-	0.7 V_{DD}	-	V_{DD}	V
Input low voltage	V_{IL}	-	V_{SS}	-	0.6	mA
Output high voltage	V_{OH}	-	3.9	-	-	V
Output low voltage	V_{OL}	-	-	-	0.4	mA
Supply current	I_{DD}	V_{DD} = +5 V	-	1.2	-	mA

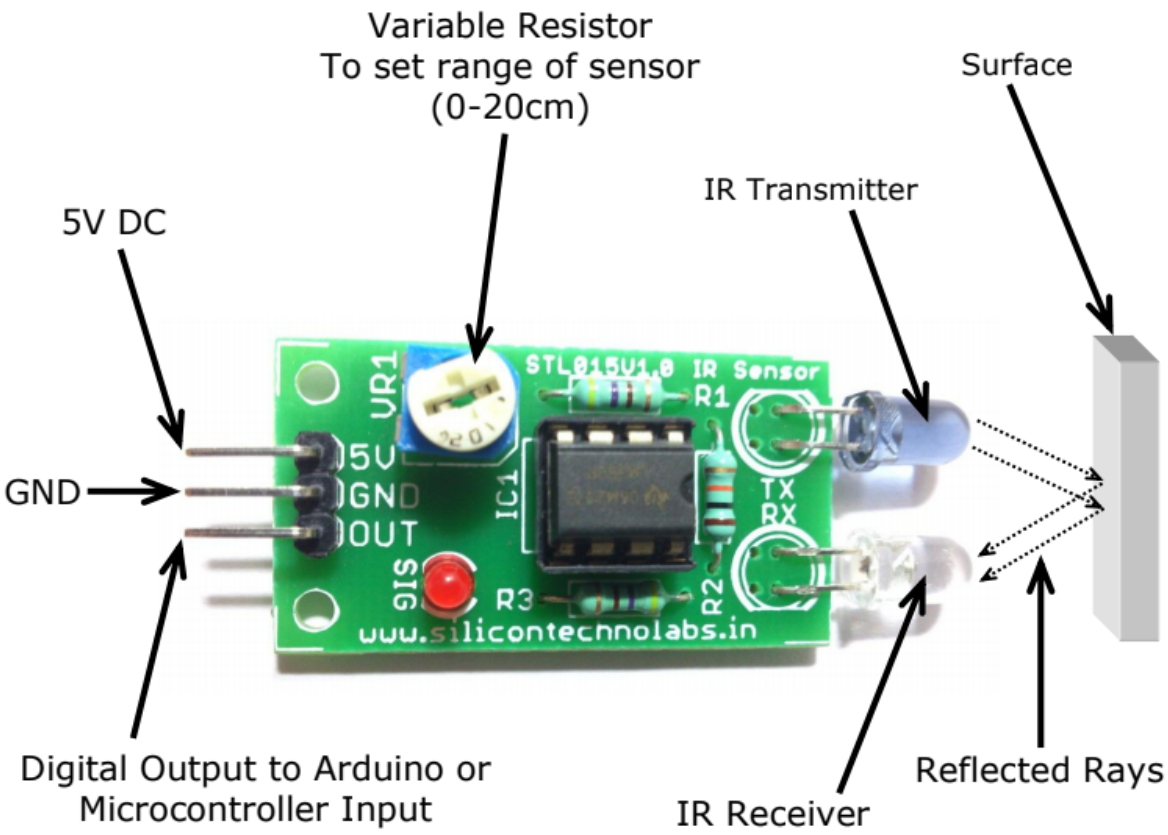
Note

- Please design the VOP adjustment circuit on customer's main board

OPTIONS									
PROCESS COLOR						BACKLIGHT			
TN	STN GRAY	STN YELLOW	STN BLUE	FSTN B&W	STN COLOR	NONE	LED	EL	CCFL
yes	yes	yes	yes	yes	-	yes	yes	-	-

For detailed information, please see the "Product Numbering System" document.

IR Sensor


Variable Resistor
To set range of sensor
(0-20cm)
Surface
IR Transmitter
5V DC
GND
VR1
5V
GND
OUT
IC1
STL015V1.0 IR Sensor
R1
TX
RX
R2
R3
SIG
www.silicontechnolabs.in
Digital Output to Arduino or
Microcontroller Input
IR Receiver
Reflected Rays

1.Descriptions

The Multipurpose Infrared Sensor is an add-on for your line follower robot and obstacle avoiding robot that gives your robot the ability to detect lines or nearby objects. The sensor works by detecting reflected light coming from its own infrared LED. By measuring the amount of reflected infrared light, it can detect light or dark (lines) or even objects directly in front of it. An onboard RED LED is used to indicate the presence of an object or detect line. Sensing range is adjustable with inbuilt variable resistor.

The sensor has a 3-pin header which connects to the microcontroller board or Arduino board via female to female or female to male jumper wires. A mounting hole for easily connect one or more sensor to the front or back of your robot chassis.

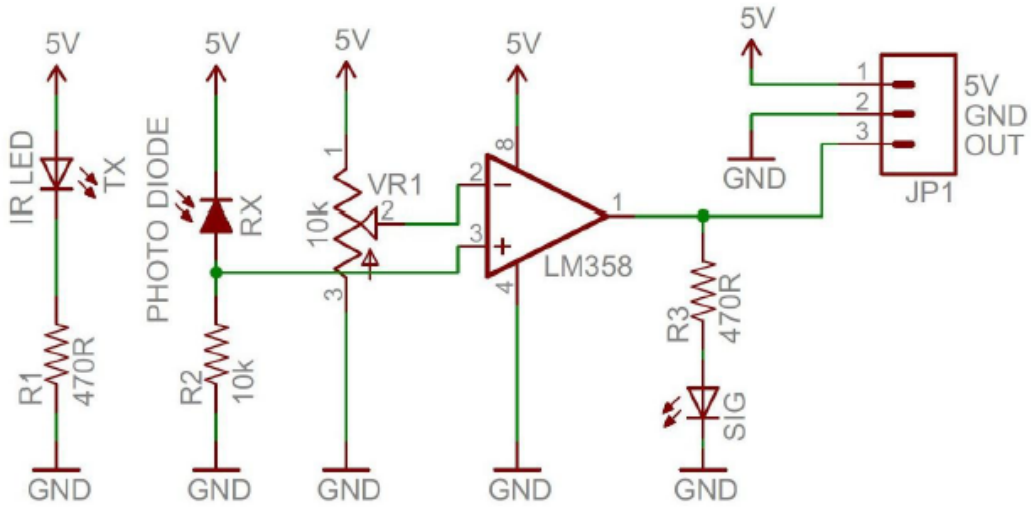
2.Features

- 5VDC operating voltage.
- I/O pins are 5V and 3.3V compliant.
- Range: Up to 20cm.
- Adjustable Sensing range.
- Built-in Ambient Light Sensor.
- 20mA supply current.
- Mounting hole.

3.Specifications

- Size: 50 x 20 x 10 mm (L x B x H)
- Hole size: ϕ2.5mm

4.Schematics



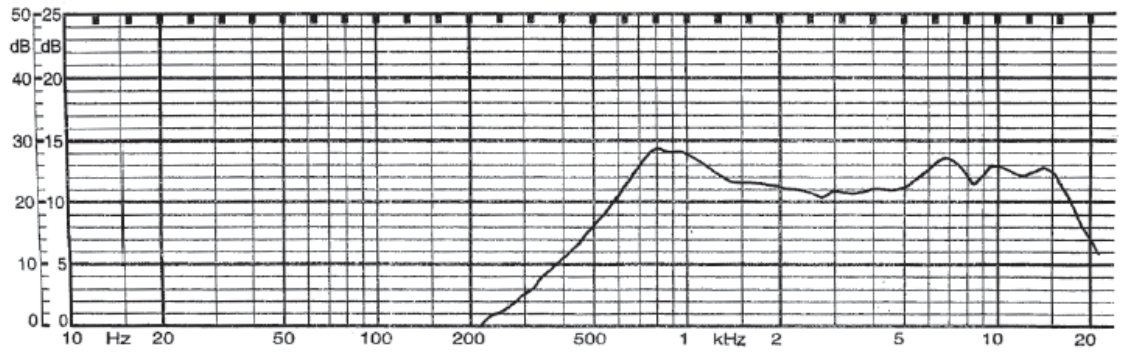
Speaker

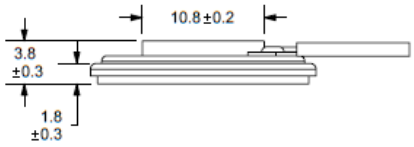
SPECIFICATIONS

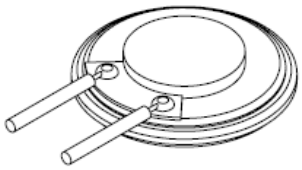
parameter	conditions/description	min	nom	max	units
nominal size	20 mm				
impedance	at 1 kHz, 1 V	6.8	8	9.2	Ω
resonant frequency	at 1 V	600	750	900	Hz
sound pressure level	1 W, 50 cm ave., at 0.8, 1, 1.2, 1.5 kHz	83	86	89	dB
response	10 dB max.	Fo		20,000	Hz
input power			0.5	1	W
operation	must be normal at program source		0.5		W
buzz, rattle, etc.	must be normal at sine wave		2		V dc
magnet	size: 8 x 1 mm				
load test	24 hours of white noise at		0.5		W
heat test	20 ~ 50% RH for 24 hours	58	60	62	°C
humidity test	90 ~ 95% RH for 24 hours	38	40	42	°C
RoHS	yes				

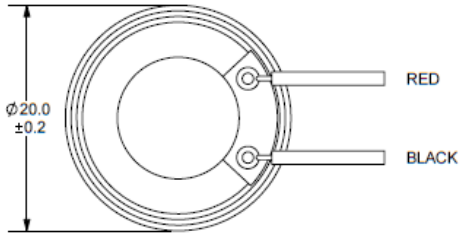
FREQUENCY RESPONSE CURVE

parameter	conditions/description
potentiometer range	50 dB
rectifier	RMS
lower limit frequency	20 Hz
wr. speed	100 mm/sec
zero level	60 dB









SPECIFICATIONS:
WIRE: UL1007, #28AWG
LENGTH: 152mm

TOLERANCE:
±0.5mm UNLESS OTHERWISE SPECIFIED

RoHS

CUI INC
20050 SW 112th Ave.
Tualatin, OR 97062
Phone: 503-612-2300
800-275-4899
Fax: 503-612-2383
Website: www.cui.com

TITLE: LOW PROFILE SPEAKER		REV: A
PART NO. CLS0201MA-L152	UNITS: MM [INCHES]	
DRAWN BY: ZRJ	APPROVED BY:	SCALE: 2:1

Ultrasonic Sensor:

Ultrasonic Ranging Module HC - SR04

Product features:

Ultrasonic ranging module HC - SR04 provides 2cm - 400cm non-contact measurement function, the ranging accuracy can reach to 3mm. The modules includes ultrasonic transmitters, receiver and control circuit. The basic principle of work:
(1) Using IO trigger for at least 10us high level signal,
(2) The Module automatically sends eight 40 kHz and detect whether there is a pulse signal back.
(3) IF the signal back, through high level , time of high output IO duration is the time from sending ultrasonic to returning.
Test distance = (high level time×velocity of sound (340M/S) / 2,

Wire connecting direct as following:

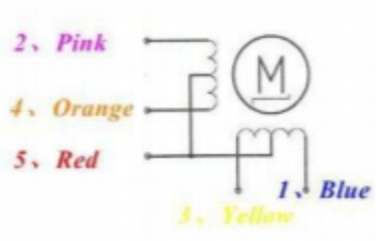
- 5V Supply
- Trigger Pulse Input
- Echo Pulse Output
- 0V Ground

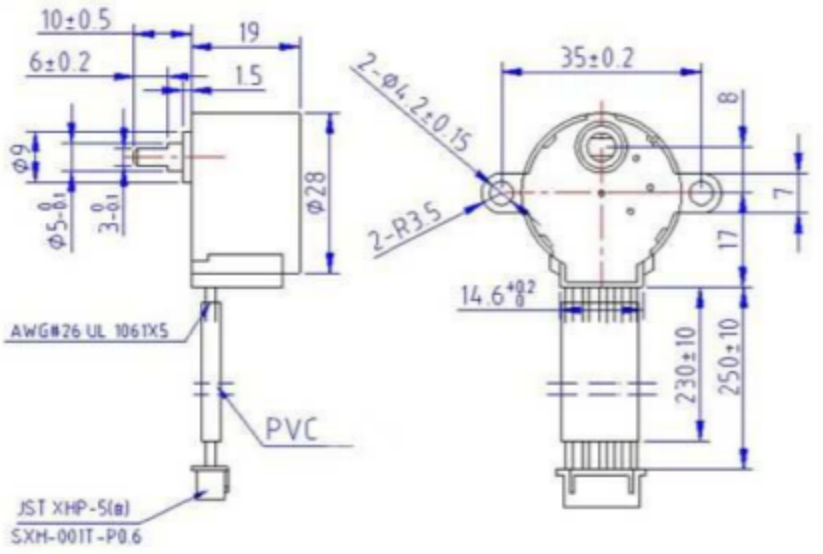
Electric Parameter

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

Step Motor

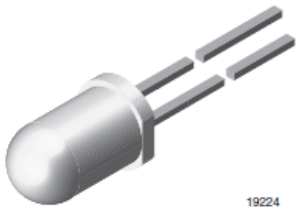
Rated voltage :	5VDC
Number of Phase	4
Speed Variation Ratio	1/64
Stride Angle	5.625°/64
Frequency	100Hz
DC resistance	50Ω±7%(25℃)
Idle In-traction Frequency	> 600Hz
Idle Out-traction Frequency	> 1000Hz
In-traction Torque	>34.3mN.m(120Hz)
Self-positioning Torque	>34.3mN.m
Friction torque	600-1200 gf.cm
Pull in torque	300 gf.cm
Insulated resistance	>10MΩ(500V)
Insulated electricity power	600VAC/1mA/1s
Insulation grade	A
Rise in Temperature	<40K(120Hz)
Noise	<35dB(120Hz,No load,10cm)
Model	28BYJ-48 – 5V





LED

Universal LED in Ø 5 mm Tinted Diffused Package



19224

FEATURES

- For DC and pulse operation
- Luminous intensity categorized
- Standard T-1¾ package
- TLUR640. without stand-offs
- Material categorization:
  For definitions of compliance please see www.vishay.com/doc?99912

RoHS COMPLIANT
HALOGEN FREE
GREEN (5-2008)

PRODUCT GROUP AND PACKAGE DATA

- Product group: LED
- Package: 5 mm
- Product series: standard
- Angle of half intensity: ± 30°

APPLICATIONS

- General indicating and lighting purposes

PARTS TABLE

PART	COLOR	LUMINOUS INTENSITY (mcd)			at I_F (mA)	WAVELENGTH (nm)			at I_F (mA)	FORWARD VOLTAGE (V)			at I_F (mA)	TECHNOLOGY
		MIN.	TYP.	MAX.		MIN.	TYP.	MAX.		MIN.	TYP.	MAX.		
TLUR6400	Red	4	15	-	10	-	630	-	10	-	2	3	20	GaAsP on GaAs
TLUR6401	Red	4	15	32	10	-	630	-	10	-	2	3	20	GaAsP on GaAs

ABSOLUTE MAXIMUM RATINGS (T_{amb} = 25 °C, unless otherwise specified)
TLUR6401

PARAMETER	TEST CONDITION	SYMBOL	VALUE	UNIT
Reverse voltage		V_R	6	V
DC forward current		I^F	20	mA
Surge forward current	$t_p \leq 10$ µs	I_{FSM}	1	A
Power dissipation	$T_{amb} \leq 65$ °C	P_V	60	mW
Junction temperature		T_j	100	°C
Operating temperature range		T_{amb}	- 40 to + 100	°C
Storage temperature range		T_{stg}	- 55 to + 100	°C
Soldering temperature	t ≤ 5 s, 2 mm from body	T_{sd}	260	°C
Thermal resistance junction/ambient		R_{thJA}	500	K/W

OPTICAL AND ELECTRICAL CHARACTERISTICS (T_{amb} = 25 °C, unless otherwise specified)
TLUR640., RED

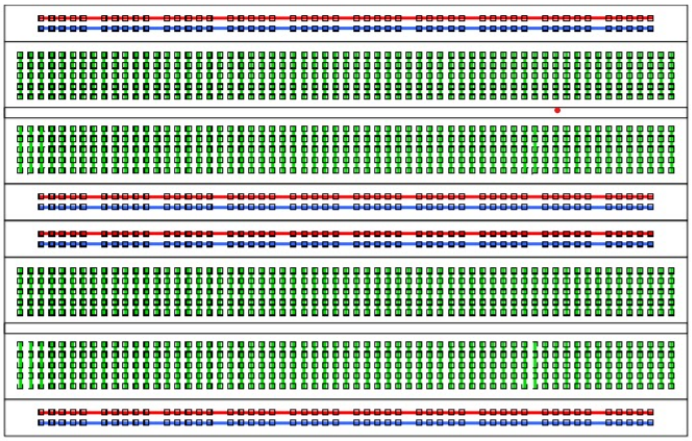
PARAMETER	TEST CONDITION	PART	MIN.	TYP.	MAX.	UNIT	MIN.
Luminous intensity (1)	I_F = 10 mA	TLUR6400	I_V	4	15	-	mcd
		TLUR6401	I_V	4	15	32	mcd
Dominant wavelength	I_F = 10 mA		λ_d	-	630	-	nm
Peak wavelength	I_F = 10 mA		λ_p	-	640	-	nm
Angle of half intensity	I_F = 10 mA		φ	-	± 30	-	deg
Forward voltage	I_F = 20 mA		V_F	-	2	3	V
Reverse voltage	I_R = 10 µA		V_R	6	15	-	V
Junction capacitance	V_R = 0 V, f = 1 MHz		C_j	-	50	-	pF

Note
(1) In one packing unit $I_{Vmin.}/I_{Vmax.} \leq 0.5$

Breadboard

Internal Connections

The BB1660 and BB1660T breadboards have four rows of 63 vertical columns. Each column has 5 connected holes each (the green lines). This is the circuit area. The are also 8 "rails" (or distribution strips) for power and ground running horizontally (the red and blue lines).
A distribution strip can be used to carry a signal if it is not needed for power or ground.



Solderless BreadBoard Specifications

BB300 Body Material: White ABS Plastic with Black Printed Legend
BB400/BB830/BB1660 Body Material: White ABS Plastic with Color Printed Legend
BB400T/BB830T/BB1660T Body Material: Transparent ABS Plastic with Color Printed Legend

Hole Pitch/Style: 0.1" (2.54 mm), Square Wire Holes

ABS Heat Distortion Temperature: 84° C. (183° F.)

Spring Clip Contact: Phosphor Bronze with Plated Nickel Finish
Contact Life: 50,000 insertions
Rating: 36 Volts, 2 Amps

Insertion Wire Size: 21 to 26 AWG wire, or 0.025" Square post headers
0.016 to 0.028 inches diameter (0.4 to 0.7mm diameter)

Backing: Peelable adhesive tape for attaching to a surface.
Metal back plates provided with 1660 tie point breadboards.
Metal Back Plate Thickness: 0.031 inches (0.8mm)

All BPS BreadBoards are Lead-Free and ***RoHS Compliant***.

Buzzer

Features

- Black in colour
- With internal drive circuit
- Sealed structure
- Wave solderable and washable
- Housing material: Noryl

RoHS Compliant

Applications

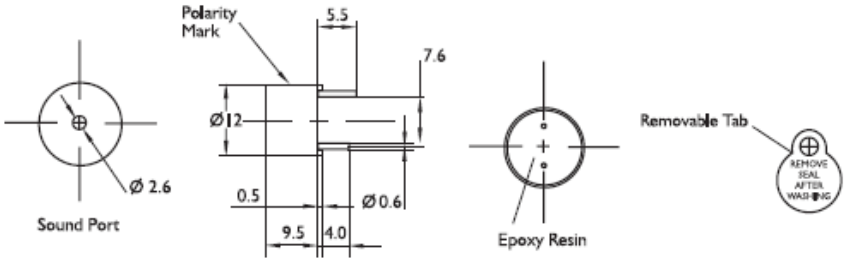
- Computer and peripherals
- Communications equipment
- Portable equipment
- Automobile electronics
- POS system
- Electronic cash register

Specifications:

Rated Voltage	: 6V DC
Operating Voltage	: 4 to 8V DC
Rated Current*	: ≤30mA
Sound Output at 10cm*	: ≥85dB
Resonant Frequency	: 2300 ±300Hz
Tone	: Continuous
Operating Temperature	: -25°C to +80°C
Storage Temperature	: -30°C to +85°C
Weight	: 2g

*Value applying at rated voltage (DC)

Diagram



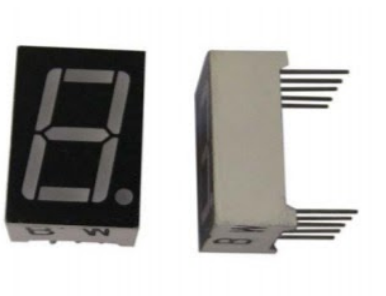
Dimensions : Millimetres
Tolerance : ±0.5mm

Part Number Table

Description	Part Number
Buzzer, Electromech, 6V DC	ABI-009-RC

7 segment display

PRODUCT DESCRIPTION
(1) 0.56 Inch (14.20mm) Digit Height
(2) Low current operation
(3) Excellent color and font characteristics
(4) Colors: White, blue, red, yellow and green
(5) Gray or black color background
(6) Common Anode
(7) RoHs Compliant Part

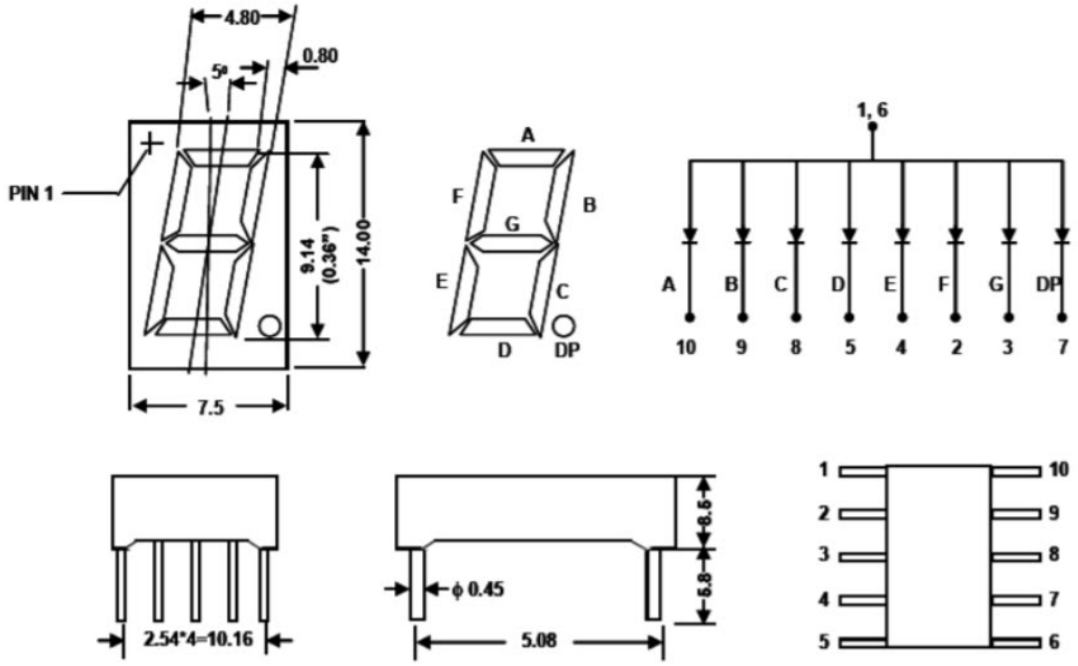


Absolute Maximum Rating (Ta = 25℃)

PARAMETER	RED	AMBER	GREEN	BLUE	WHITE	UNITS
DC Forward Current Per Segment	30	30	25	30	20	mA
Peak Current Per Segment (1)	70	50	50	25	25	mA
Avg. Forward Current (Pulse Operation) Per Segment	30	30	25	25	25	mA
Derating Linear From 25℃ Per Segment	0.3					mA/℃
Reverse Voltage (2)	3					V
Operating Temperature	-25 to +85					℃
Storage Temperature	-30 to +85					℃

(1) Pulse conditions of 1/10 duty and 0.1msec width, for long operating life, max. of 20mA recommended

(2) Reverse biasing of the dot matrix is not recommend, will cause damage to the leds



Rocker switch

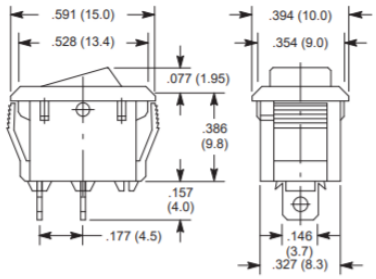
Micro Snap–In Nylon

Features

- Fits Standard Panel Cutout
- RoHS Compliant

NTE Type No.	Circuitry	Action			Actuator	Legend	Diag No.
54–870	SPST	ON	NONE	OFF	White Nylon	Yes	S108
54–872	SPST	ON	NONE	OFF	Black Nylon	Yes	S108
54–874	SPST	ON	NONE	OFF	Red Nylon	No	S108

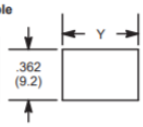
S108



Mounting Hole

Table 1.

Panel Thickness	Y
.029 (.750) – .049 (1.25)	.535 (13.6)
.049 (1.25) – .079 (2.00)	.539 (13.7)
.079 (2.00) – .098 (2.50)	.543 (13.8)

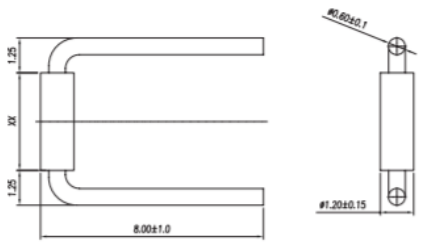


Specifications

Current Rating: 6A 125VAC, 3A 250VAC, 3A 250VAC T100/55
Insulation Resistance: 100 Megohms (min.) at 500VDC
Dielectric Strength: 1000V RMS (min.) for 1 minute
Temperature Rating: +32° to +212°F (0° to +100°C)
Electrical Life: 10,000 cycles
Mechanical Life: 50,000 cycles
Terminal Type: Solder Lug
Mounting Hole: See Table 1.

Jumper wires

Diagram



Dimensions : Millimetres

Linear Dimension (Tolerance Standard GB/T 14486-2008)																
Variable Range	to 6	Over 6 to 10	Over 10 to 18	Over 18 to 30	Over 30 to 50	Over 50 to 80	Over 80 to 120	Over 120 to 160	Over 160 to 200	Over 200 to 250	Over 250 to	all	to 5	Over 5 to 30	Over 30 to 120	Over 120 to
Tolerance	±0.1	±0.14	±0.19	±0.25	±0.32	±0.45	±0.57	±0.72	±0.88	±1.05	±1.25	0.5	0.025	0.06	0.12	0.4

Notes:

1. XX for lengths of jumper wires
2. Wires Lengths, Colours and Quantity:
   2mm: bare wire, 10 Pcs
   5mm: Red, 10 Pcs
   7mm: Orange, 10 Pcs
   10mm: Yellow, 10 Pcs
   12mm: Green, 10 Pcs
   15mm: Blue, 10 Pcs
   17mm: Purple, 10 Pcs
   20mm: Grey, 10 Pcs
   22mm: White, 10 Pcs
   25mm: Brown,10 Pcs
   50mm: Red, 10 Pcs
   75mm: Orange, 10 Pcs
   100mm: Yellow, 10 Pcs
   125mm: Green, 10 Pcs
3. Wire Range: 22AWG/Single Conductor

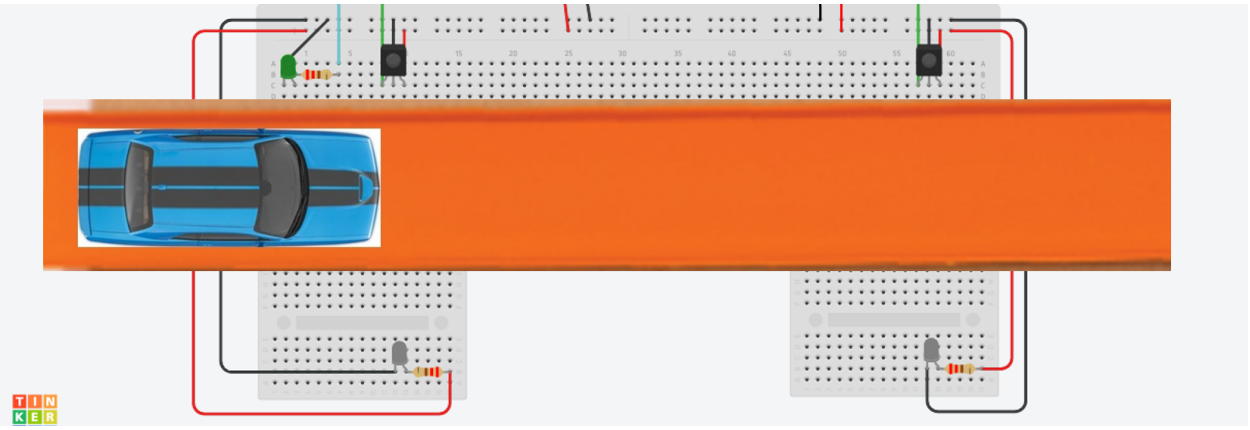
Part Number Table

Description	Part Number
140 Pcs Hard Jumper Wire	MC001810

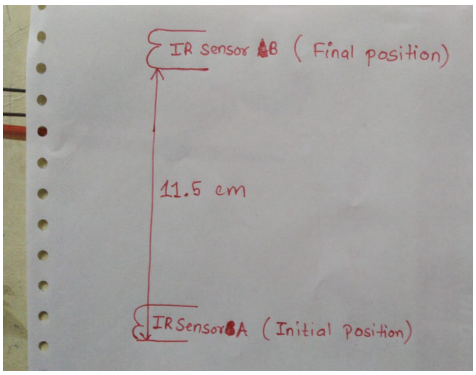
Calculations:

1. Calculating the speed of the car

IR Sensor A IR Sensor B



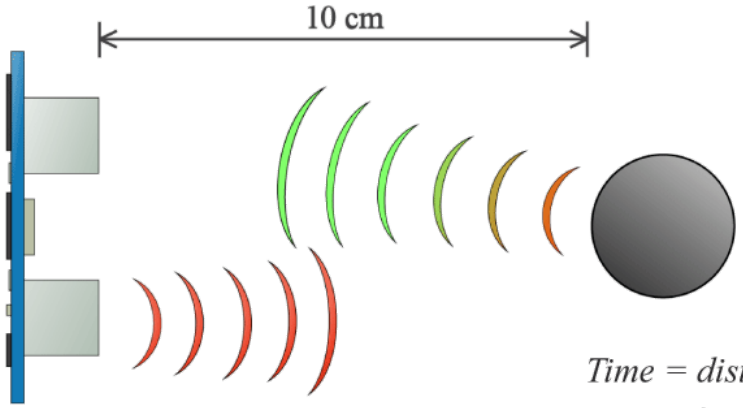
- The distance between two IR sensors is calculated as



- We have the distance between two sensors. We call the instant when car passes in front of IR Sensor A as initial position/ initial_time and the the instant when car passes in front of IR Sensor B as final position/ final_time
- time_difference = final_time - initial_time
- Speed = distance / time
- Speed = 11.5 cm / time_difference
- Speed = (11.5 km/ 100000)/ ((time_difference*1000/3600) hr)
- Speed = 414/time_difference km/hr

2. Measuring the distance of object from the ultrasonic sensor

Ref: https://howtomechatronics.com/tutorials/arduino/ultrasonic-sensor-hc-sr04/



speed of sound:

$v = 340\ m/s$

$v = 0{,}034\ cm/\mu s$

Time = distance / speed:

$t = s / v = 10 / 0{,}034 = 294\ \mu s$

Distance:

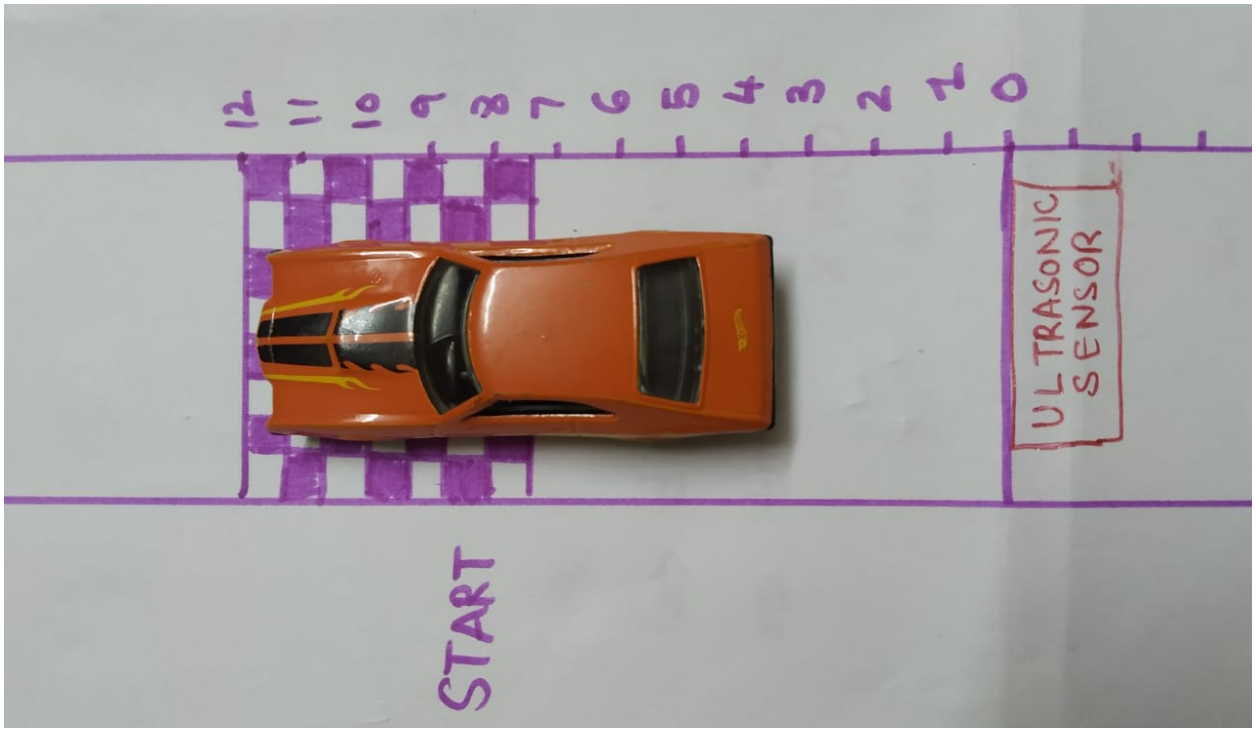
$s = t \cdot 0{,}034 / 2$

(3) Calculating the distance of the car from the sensor

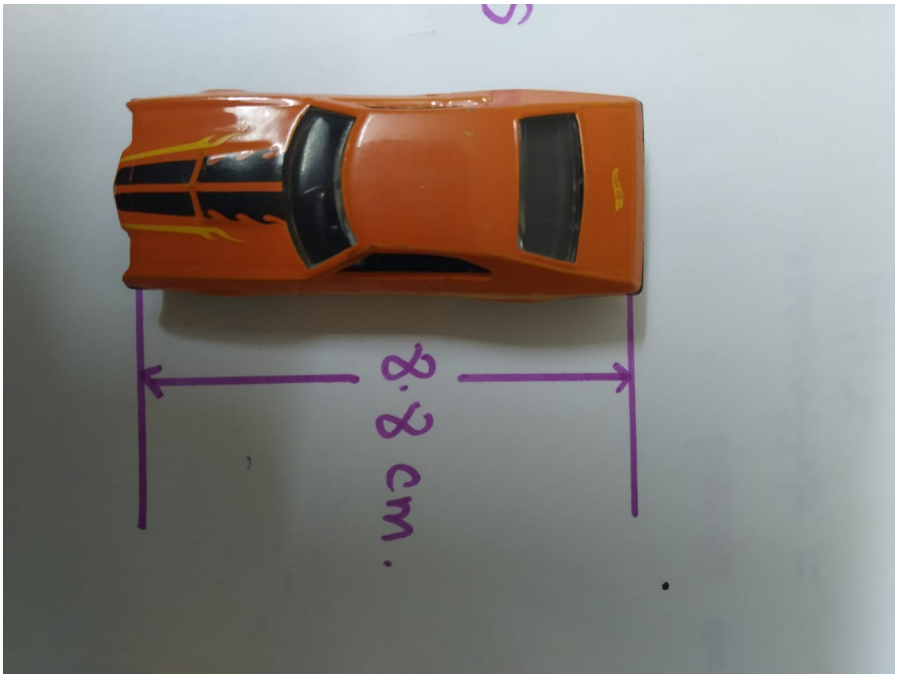
The initial position of the car is shown in the figure.

- **The maximum distance between the rear part of the car and the ultrasonic sensor**
  = distance of the starting line from the ultrasonic sensor
  - the length of the car

- **The maximum distance of the front part car from the ultrasonic sensor = 12 cm**
- **The maximum distance of the rear part car from the ultrasonic sensor = 12 - 8.8 cm = 3.2 cm**



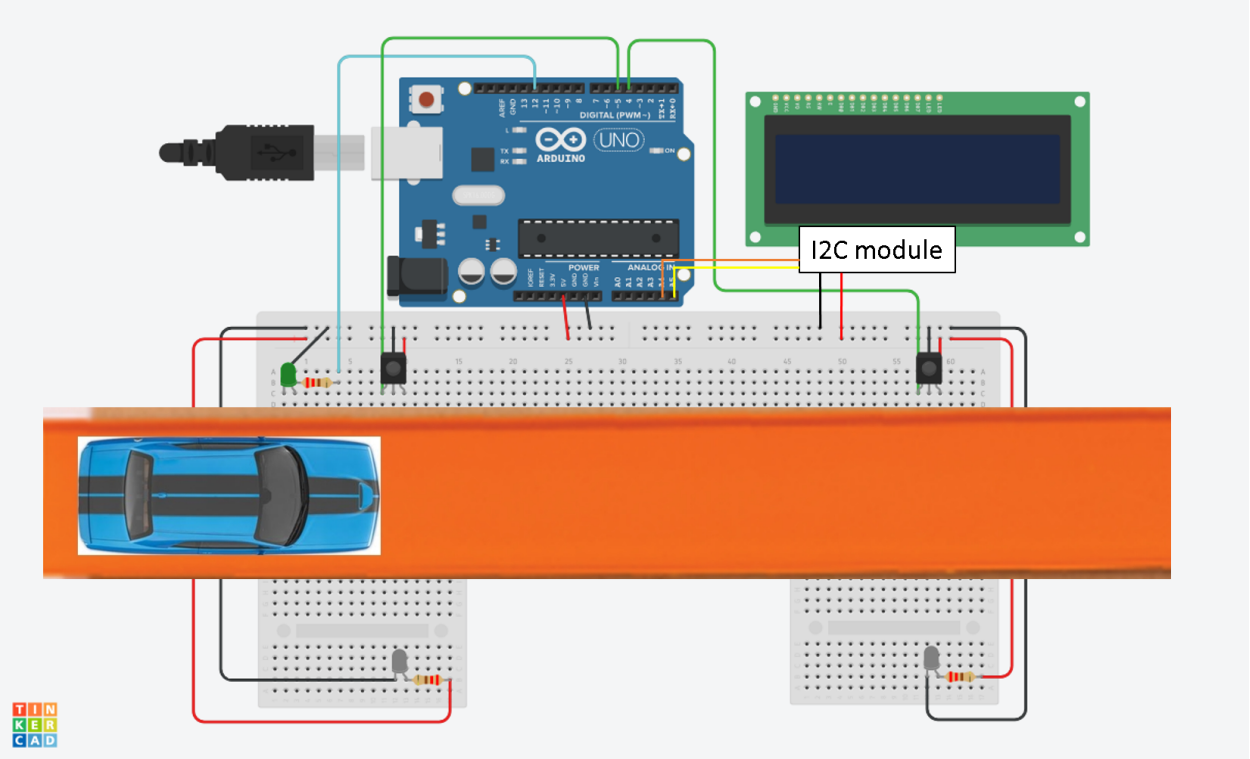
Calculating the length of the car is shown in the figure below:



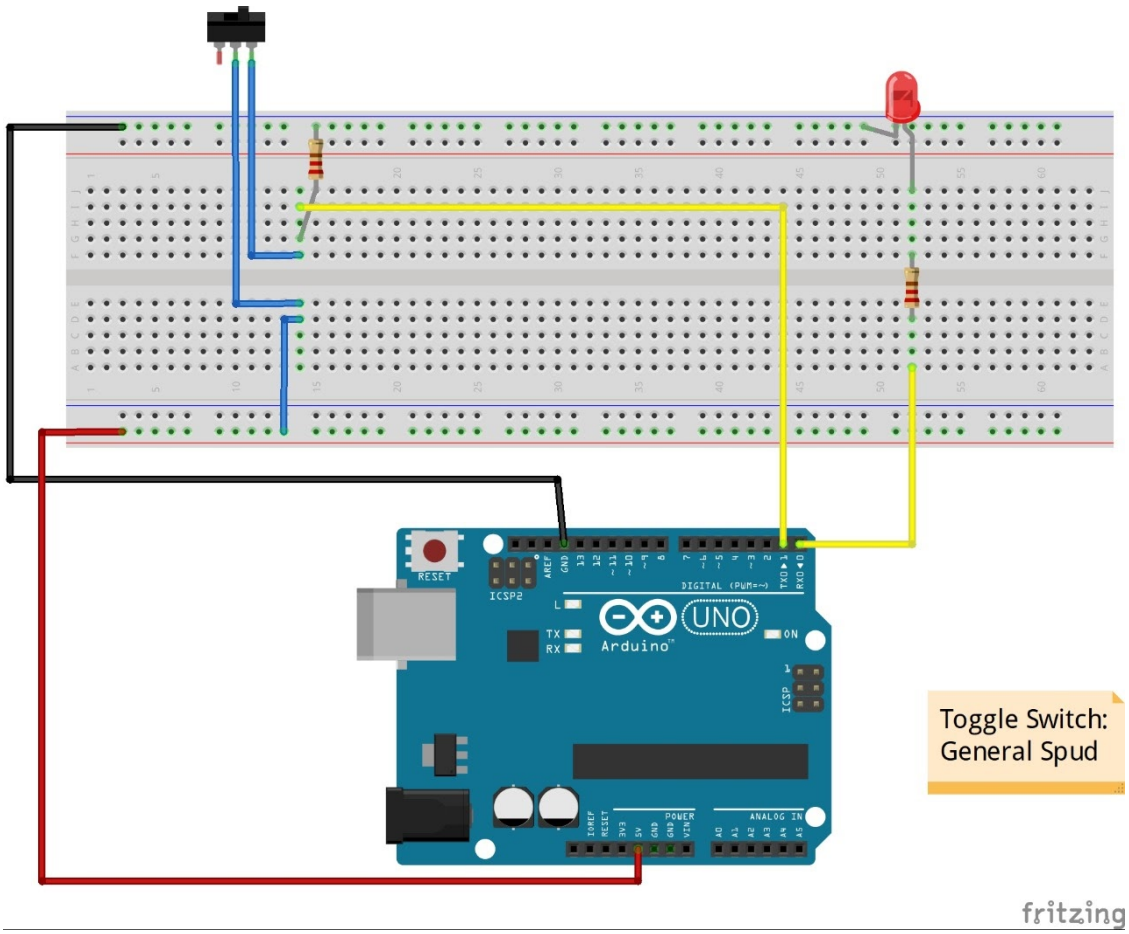
Individual Circuits:

(1) IR Sensor (Complexity 1)
and
(2) LCD (Complexity 2)

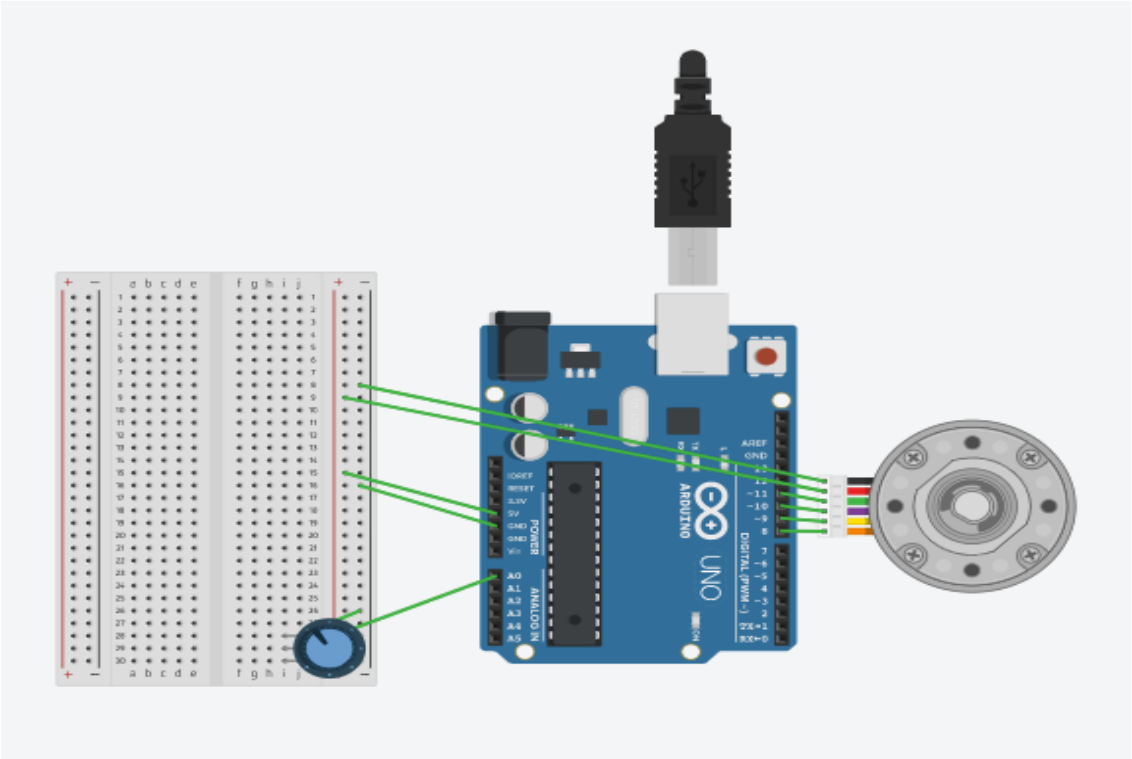
Ref:
https://create.arduino.cc/projecthub/NerdFatherRJ/speeduino-speed-tracker-c8fcae?ref=tag&ref_id=tracking&offset=7



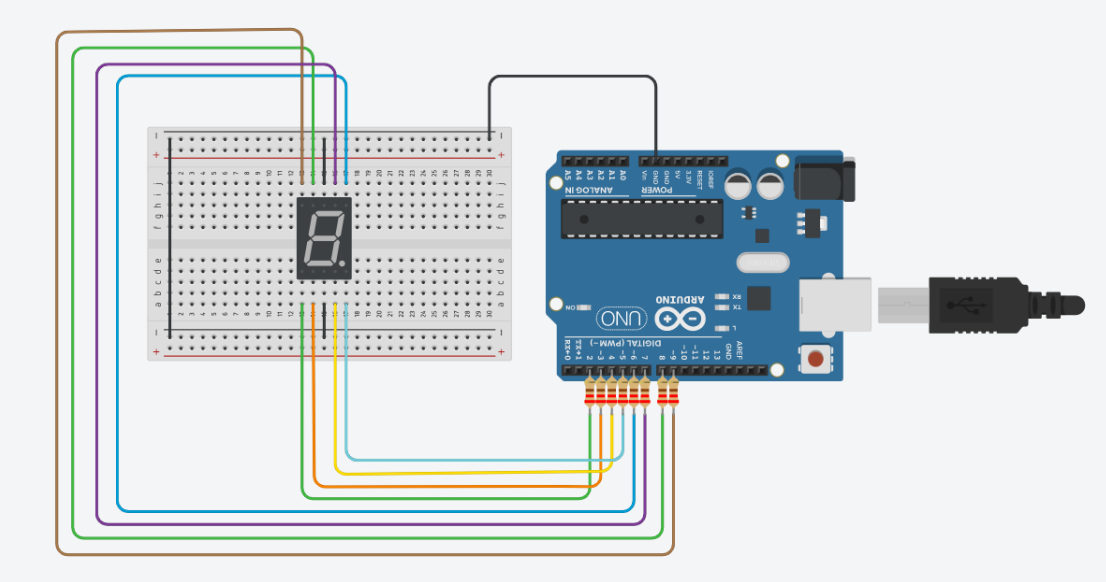
(3) Switches


RESET
ICSP2
UNO
Arduino
TX
RX
ON
ICSP
DIGITAL (PWM~)
POWER
ANALOG IN
Toggle Switch:
General Spud
fritzing

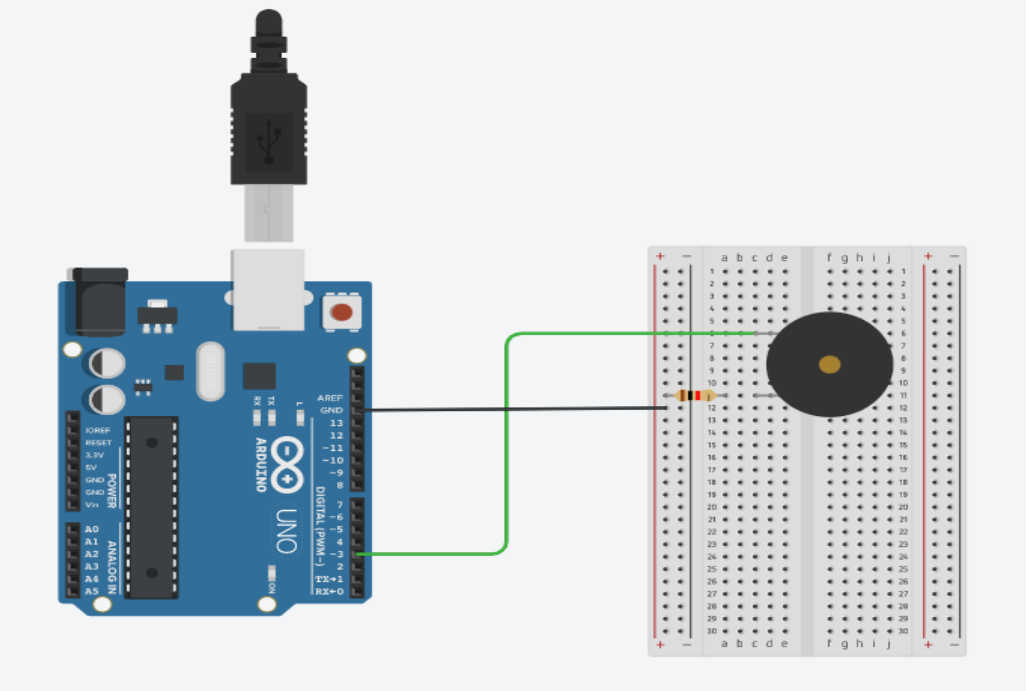
(4) Stepper Motor (Complexity 3)



(5) 7 segment display

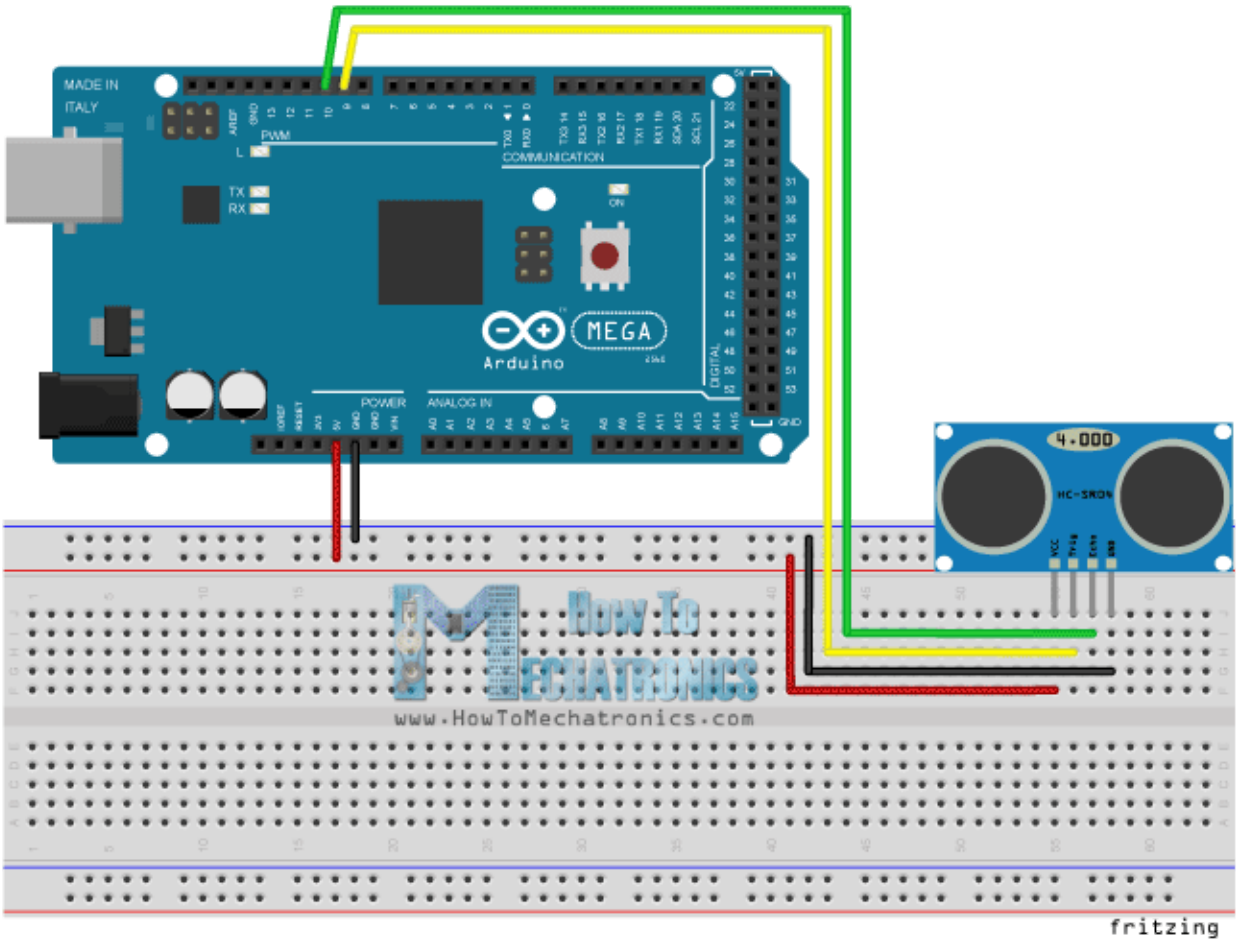


(6) Speaker (Complexity 6)

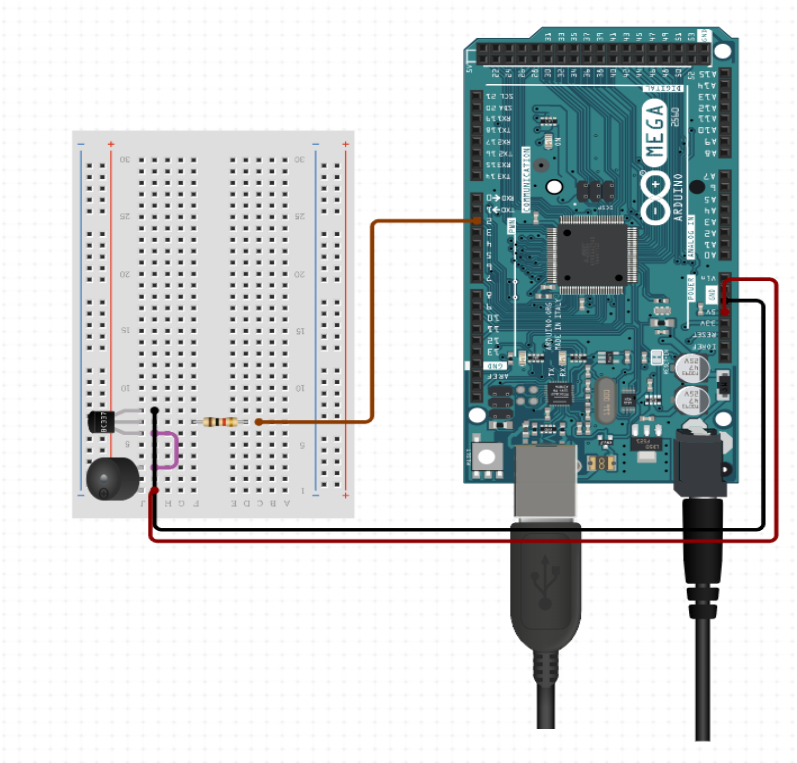


(7) Ultrasonic sensor (Complexity 4)

Reference :
https://howtomechatronics.com/tutorials/arduino/ultrasonic-sensor-hc-sr04/



(8) Buzzer (Complexity 5)



Individual Codes:

(1) IR Sensor (Complexity 1)

```
const int ledPin = 12;
byte irPinA = 4;
byte irPinB = 5;
byte irValA;
byte irValB;
float difference;    // store final_time - initial_time
float speed; // calculated speed
unsigned long initial_time;              // IR sensor at irPinA
unsigned long final_time;                // IR sensor at irPinB
float speed_constant = 414;

// ((distance between IR sensors in mm) x 3600)/1000) to convert mm/millis to
km/h)
// In this project we have distance =  115 mm between IR sensors

void setup()
{
        pinMode(irPinA, INPUT);
        pinMode(irPinB, INPUT);
        pinMode(ledPin, OUTPUT);
        digitalWrite(ledPin, HIGH); //Green LED HIGH: ok to next ride.
        Serial.begin(9600); // Starts the serial communication
}//end setup
```

```
void loop()
{
          irValA = digitalRead(irPinA);
          irValB = digitalRead(irPinB);

          if (irValA == LOW){
                    initial_time = millis();
                    digitalWrite(ledPin, LOW);
                    delay(30);
         }
         if (irValB == LOW){
                    final_time = millis();
                    difference = final_time - initial_time;

                  //After we get the final time, calculate the velocity
                    speed  = speed_constant / difference;
                  //get the Speed converted from mm/millis to km/h.

                  //Display calculations in the Serial Monitor to check the values
                    Serial.print("T1: ");
                    Serial.println(initial_time);
                    Serial.print("T2: ");
                    Serial.println(final_time);
                    Serial.print("Time interval in milli seconds:  ");
                    Serial.println(difference);
                    Serial.print("Speed: ");
                    Serial.println(speed);
                    Serial.println(" km\\hr");
                    digitalWrite(ledPin, HIGH);
               }

}//end Loop
```

(2) LCD (Complexity 2)

```
//include the required directories

#include <LiquidCrystal_I2C.h>
#include <Wire.h>

LiquidCrystal_I2C lcd(0x27,16,2);        // I2C address

byte customChar0[8] = {  //some art for the LCD screen
B01000,
B01100,
B01110,
B01111,
B01110,
B01100,
B01000,
B00000
};

void configure_LCD(){ //function to config the LCD display

lcd.createChar(0, customChar0);
lcd.setCursor(0,0);
lcd.write(0); lcd.write(0); lcd.write(0); lcd.write(0); lcd.write(0);
lcd.setCursor(5,0);
lcd.print("SPEED TEST");
lcd.setCursor(15,0);
lcd.write(0); lcd.write(0); lcd.write(0); lcd.write(0); lcd.write(0);

lcd.setCursor(0,1);
lcd.print("P1:");
lcd.setCursor(0,2);
lcd.print("P2:");
lcd.setCursor(0,3);
lcd.print("Speed:");

}
```

```
void setup()
{
        lcd.init();
        lcd.backlight();
        configure_LCD();
        serial.begin(9600); // Starts the serial communication
}//end setup


void loop()
{
          lcd.setCursor(6, 1);
          lcd.print(initial_time);
          lcd.setCursor(6, 2);
          lcd.print(final_time);
          lcd.setCursor(6,3);
          lcd.print(speed);
          lcd.print("km/h");
          delay(10000);
        //After 10 seconds, clear the screen with its original configuration
          lcd.clear();
          configure_LCD(); //Reset LCD to its original display screen
    }

}//end Loop
```

(3) Switches

```
int switch_pin = 4;
int led_pin = 5;

byte leds = 0;

void setup() {
        pinMode(switch_pin, INPUT);
        pinMode(led_pin, OUTPUT);
}

void loop() {
        if(digitalRead(switch_pin) == HIGH){
          digitalWrite(led_pin, LOW);
        }
        if(digitalRead(switch_pin) == LOW){
          digitalWrite(led_pin, HIGH);
        }

}
```

(4) Stepper Motor (Complexity 3)

```
#include <Servo.h>
Servo myservo;  // create servo object to control a servo
// twelve servo objects can be created on most boards
int pos = 0;    // variable to store the servo position
void setup() {
 myservo.attach(9);  // attaches the servo on pin 9 to the servo object
}
void loop() {
 for (pos = 0; pos <= 180; pos += 1) { // goes from 0 degrees to 180 degrees
  // in steps of 1 degree
  myservo.write(pos);           // tell servo to go to position in variable 'pos'
  delay(15);                    // waits 15ms for the servo to reach the position
 }
 delay(10000); // Wait for 10 seconds to let car go away and finish the race
 for (pos = 180; pos >= 0; pos -= 1) { // goes from 180 degrees to 0 degrees
  myservo.write(pos);           // tell servo to go to position in variable 'pos'
  delay(15);                    // waits 15ms for the servo to reach the position
 }
}
```

(5) 7 segment display

```
byte pins[8] = {2,3,4,6,7,8,9};              // The digital pins that we are using
int duration = 500;                  // The delay between the number changes
byte num3 = 0x5E;                    // digit 3
byte num2 = 0x5B;          //  digit 2
byte num1 = 0x0C;          // digit 1
byte num0 = 0x3F;                    //digit  0
void printN(byte _digit) { // The function that prints the numbers
        for (int i = 0; i < 8; i++) {
                digitalWrite(pins[i], bitRead(_digit, i ));
                        //Serial.println(pins[i] + ' w ' + bitRead(_digit, i ));
        }
}
void setup() {                       // Our setup function, in which we initialise the
above-mentioned pins
        Serial.begin(9600);
        printN(num3); delay(duration);                  // display 3
        printN(num2); delay(duration);        // display 2
        printN(num1); delay(duration);                  // display 1
       printN(num0); delay(duration);
       // Basically, what this line does: print 0 on the display then wait 1
      second

        for(int i = 0; i <= 9; i++) {
                pinMode(pins[i], OUTPUT);
                //Serial.println(pins[i]);
        }
}
}
```

(6) Speaker (Complexity 6)

```
int note [] = {261, 294, 120, 240, 320, 396, 450, 255, 156, 503, 354, 687, 422,
659, 798, 200, 740, 784, 430, 880};
// Initialize notes with some random frequency

void setup(){ // Initial setup of the PIN3  as output
  pinMode(3, OUTPUT);
}
void loop(){
        for (int i=0;i<=20;i++)
        {
                tone(3,note[i],500);
                delay(200);
        }
        for (int i=20;i>=0;i--)
        {
                tone(3,note[i],500);
                delay(200);
        }
}
```

(7) Ultrasonic Sensor (Complexity 4)
And
(8) Buzzer (Complexity 5)

```
// defines pins numbers
const int trigPin = 9;
const int echoPin = 10;
const int piezoPin = 8;

// defines variables
float duration;
float distance;
float min_diff;

void setup() {
	pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
	pinMode(echoPin, INPUT); // Sets the echoPin as an Input
	Serial.begin(9600); // Starts the serial communication
}
void loop() {
	// Clears the trigPin
	digitalWrite(trigPin, LOW);
	delayMicroseconds(2);
	// Sets the trigPin on HIGH state for 10 micro seconds
	digitalWrite(trigPin, HIGH);
	delayMicroseconds(10);
	digitalWrite(trigPin, LOW);
	// Reads the echoPin, returns the sound wave travel time in microseconds
	duration = pulseIn(echoPin, HIGH);
	// Calculating the distance
	distance= duration*0.034/2;
	// Prints the distance on the Serial Monitor
	Serial.print("Distance: ");
	Serial.print(distance);
	min_diff = 12-8.8;
	if(distance > min_diff){
		tone(piezoPin, 1000, 500);
	}
}
```

DECIDED TIME-LINE OF OUR PROJECT:

DATE	SUBMISSION	TAGERT WORK TO BE COMPLETED
13/02/2020	First report submission	System decision and description
05/03/2020	Second report submission	Collecting and understanding all the components required
26/03/2020	Third report submission	Each separate module working properly
07/04/2020	-	Assembling of all modules together and debugging of the errors
09/04/2020	Final report and project submission	Adding perfections, testing and completing the project.
15/04/2020	Public demonstration of the project	The maintenance of the project and getting the real-time user feedback.
09/04-16/04	Project demo and viva	The complete project demonstration and answering the questions.

ACTUAL TIME-LINE OF OUR PROJECT:

DATE	SUBMISSION	WORK COMPLETED
13/02/2020	First report submission	System decision and description
05/03/2020	Second report submission	Collecting and understanding all the components required
18/04/2020	Third report submission	Each separate module working properly