Traffic lights aware & Line following robot

practical steps in the design & implementation of the robot

Team no. 16

contents

- overview
- testing environment
- design goals
- core components
- apparent problems
- design approaches &failed plans

overview

The line following robot we are about to discuss is a 3-wheeled vehicle implementing an automation system using only logic gates “meaning no use of micro controller chips or remote controllers “.

In addition to the line following robot itself , we are also implementing a miniature testing environment “as required”
consisting of :

- **A map representing roads in a city with turns and crossroads**
- **3 Traffic light devices “implemented using counters,LEDs , etc..”**
- **2 car counters”to count passing cars”**

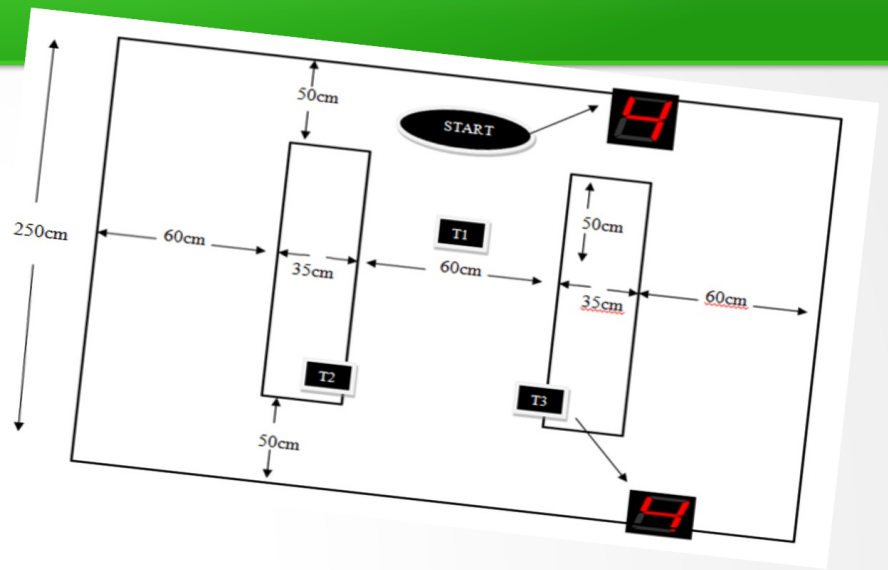
We'll start with the testing environment..

The testing environment

the map

-A 2.5*2.5 map is made according

to the specifications given.

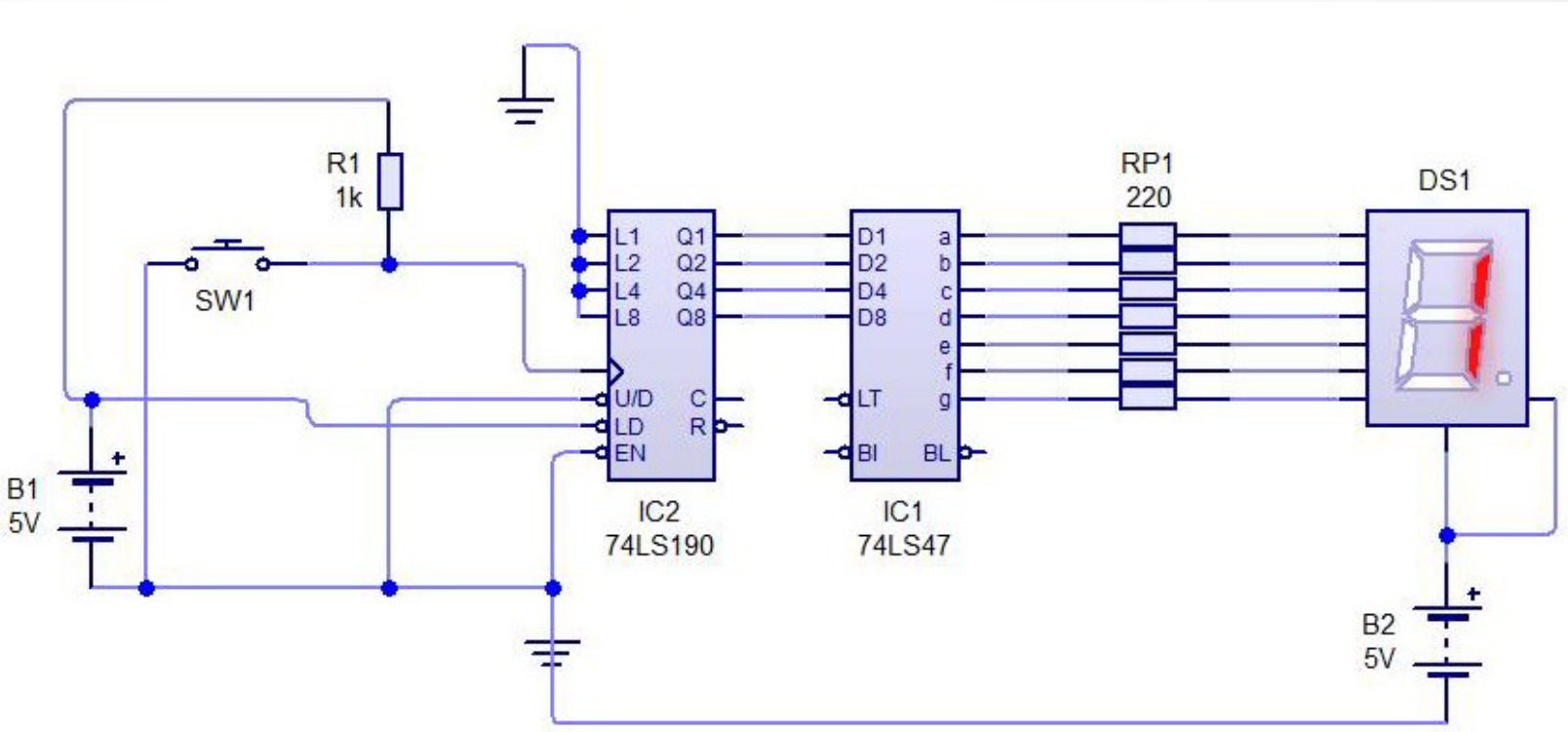


the 3 traffic lights

- Traffic lights are working according to specific sequence:
  - 5 seconds Yellow
  - 10 seconds Green
  - 5 seconds Red
- They are implemented all using 4017 “decade counter ic” along with the NE555 timer “provides the clock signal”.

2 counters

- There should be two counters to indicate how many times the robot has passed from a specific point.
- This will be achieved using a sensor “consisting of an IR led and a photo diode” Coupled with a counter circuit built around the 74190 BCD counter and the 7447 BCD to 7-segment decoder.



Design goals

To make the robot follow the line "line follower part".

the idea is simple"a robot that follows a black line on a white background"

the solution is also somewhat simple:

a couple of IR emitter/receiver pairs and a motor driver module should do the job……But it's not that easy.

the robot should also be able to detect and act according to the traffic light sequence of the three traffic lights.

the robot should be able to detect an obstacle and stop immediately .

We use a sensor and an interface into the logic that actually controls the robot, more details later.

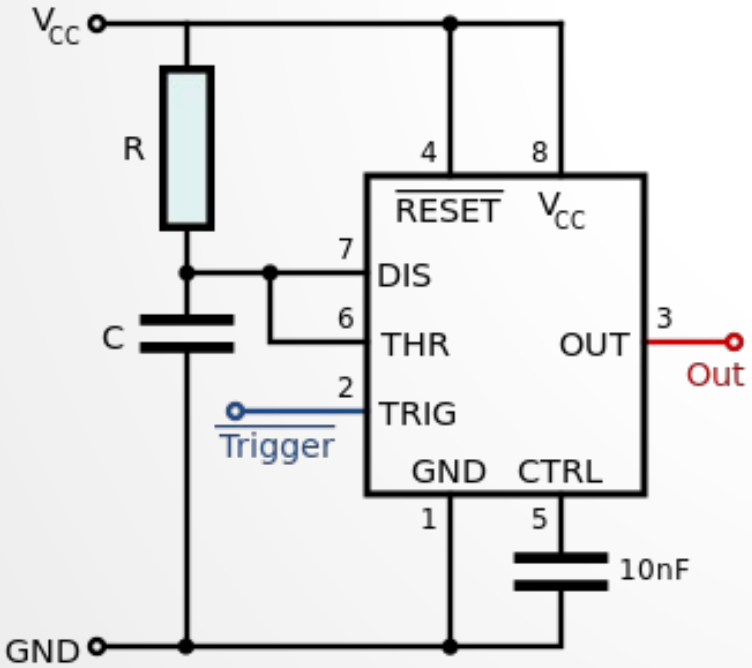
Core components

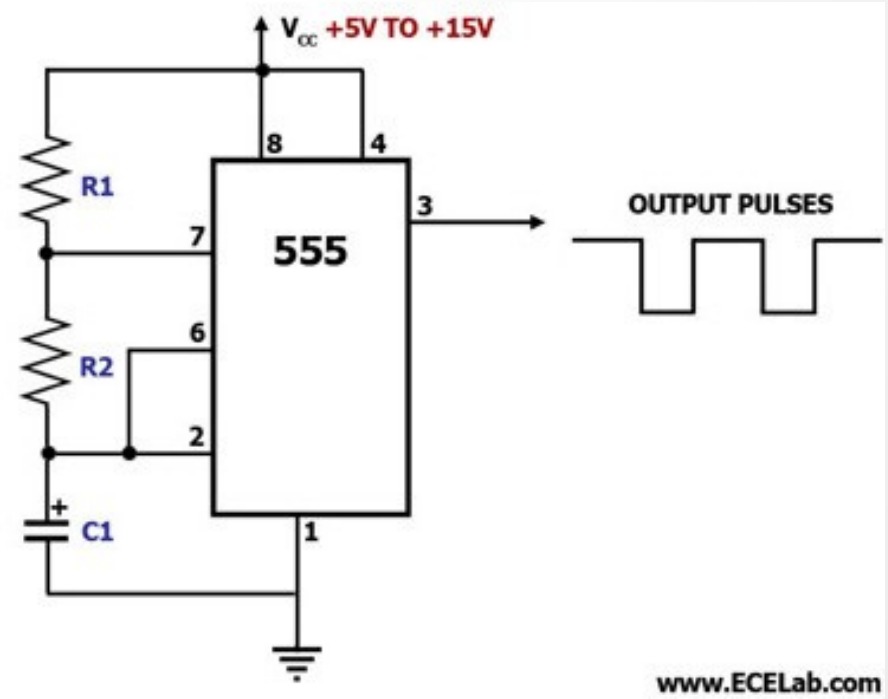
1-The notorious 555 timer.

This timer is used in two operating modes for 2 different applications:
First: as a clock wave generator “astable multi-vibrator” to power up the BCD counter in the traffic lights' circuits.

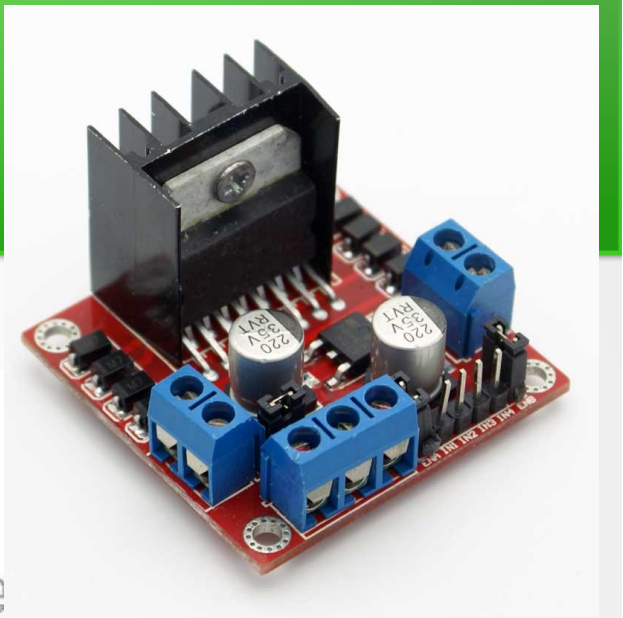
Second: we use it as a”mono stable multi vibrator”or one-shot,

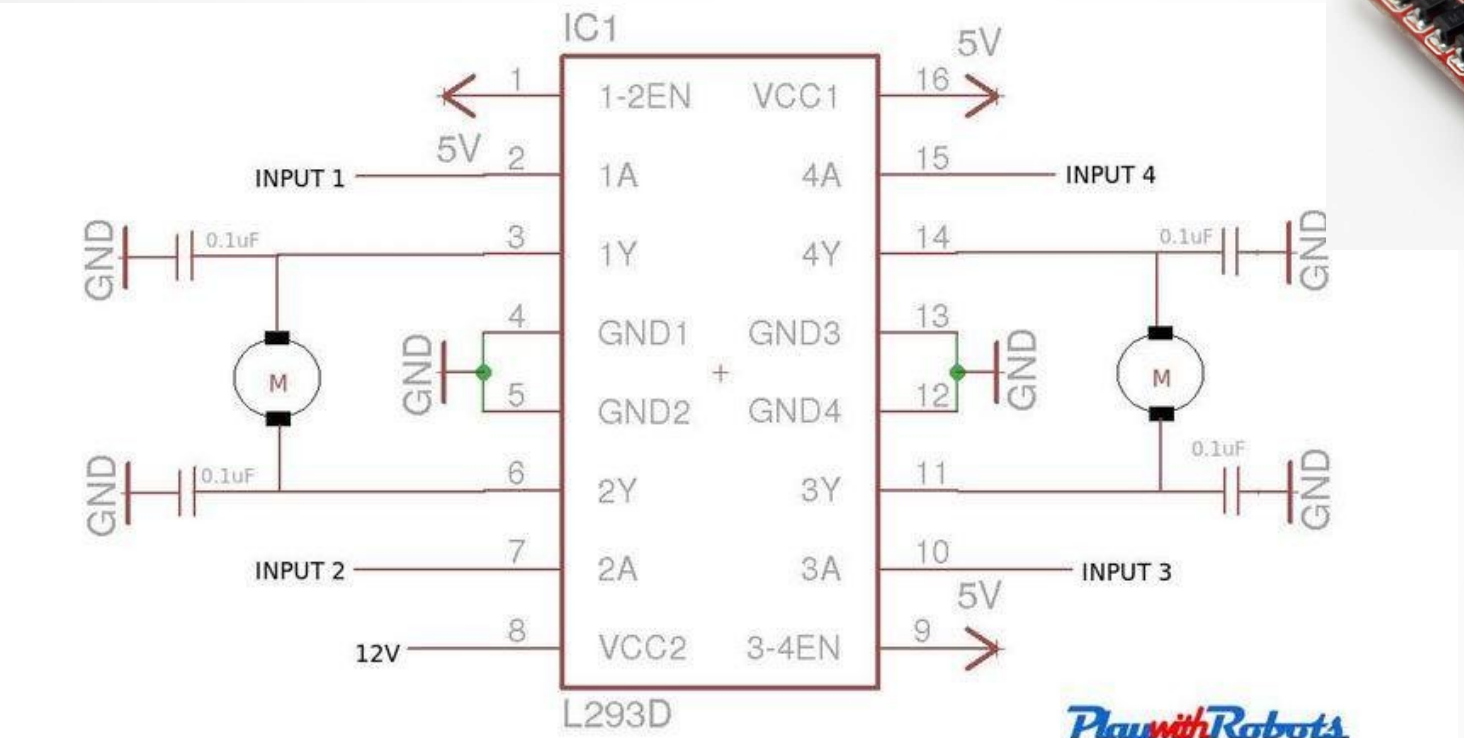
and it plays an essential role in solving the “decision delay “problem.





2-the driver module





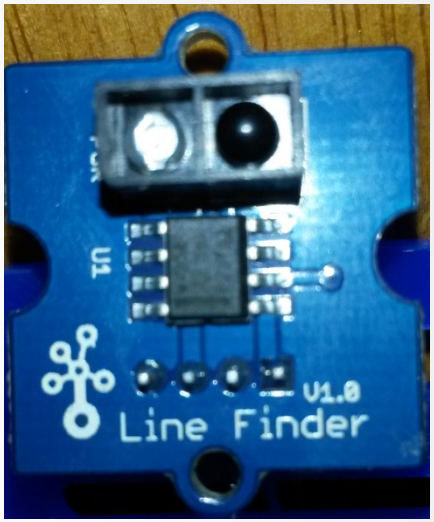
Circuit diagram

1A	2A	1Y	2Y	Motor 1
Logic 0	Logic 0	0	0	Stop
Logic 1	Logic 0	12V	0	Clockwise
Logic 0	Logic 1	0	12V	Anti-clockwise
Logic 1	Logic 1	12V	12V	Brake

contains L293NE, heat sink, input pins and output pins ,has a protection against high voltage and over heating.

3- 2 DC motors.

The two bi-directional DC motors are totally controlled by the driver module.



4-IR sensor modules

- It has 4 pins; VCC, GND and two other output pins for digital and analog output
- It's used in this project to differentiate between the white and black lines
- The analog pin generates output voltage depending on amount of black and white it detects, it is used in pulse width modulation circuit which will be discussed later.
- There is also a collision sensor (It's also an Infrared sensor module) which brakes the motors if there is an obstacle in front of the robot

5- 74190 BCD counter & 7447 -BCD to seven segment- decoder

both of these components are used in conjunction to provide an implementation of the IR-sensor based car counting system.

6-Brain " combinational logic "

The brain of motor is used to take decisions (forward-right-left-stop) depending on the output of sensors and traffic lights. All the situations the robot may meet during its journey are included in the truth table and are converted into a logic circuit using a Karnaugh-map.

apparent problems

From studying the given environment setup,it became evident that a couple of challenging situations will have to be dealt with during the design phase:

detecting the traffic lights' output

the robot must be able to know the state of one or two traffic lights at a time

We used an IR emitter coupled with the traffic lights On the robot side ,we used a receiver

how to delay turning decision at crossroad

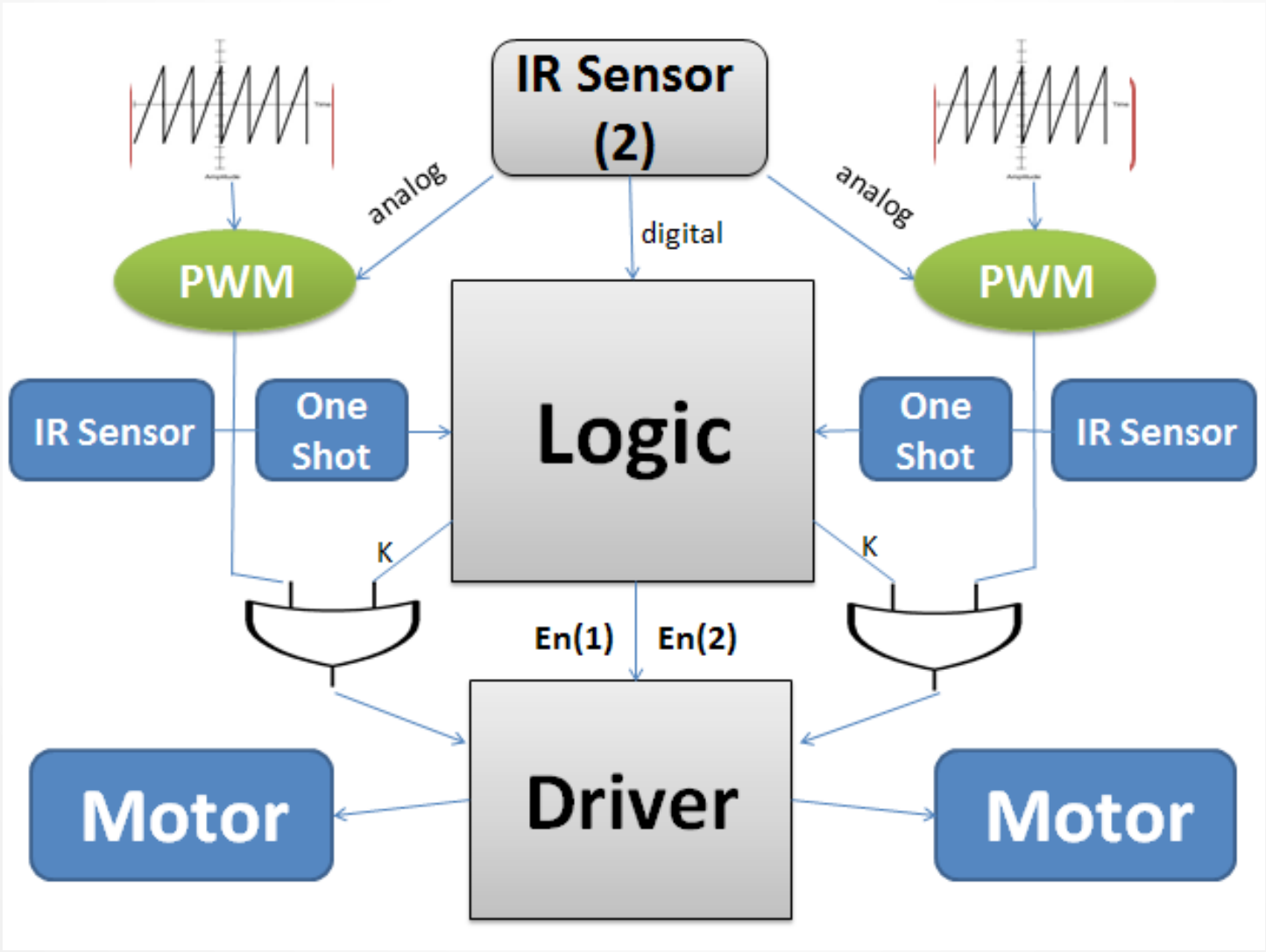
- the decision to make a right/left turn or to stop will not be taken at the moment

the state of the lights are detected

- It must be delayed by somehow making the robot remember those states
- we used NE555 as a one shot with a specific “up-time” fitting our needs , thus making the robot “as if” it remembers the state of the lights when the time have come to make a decision

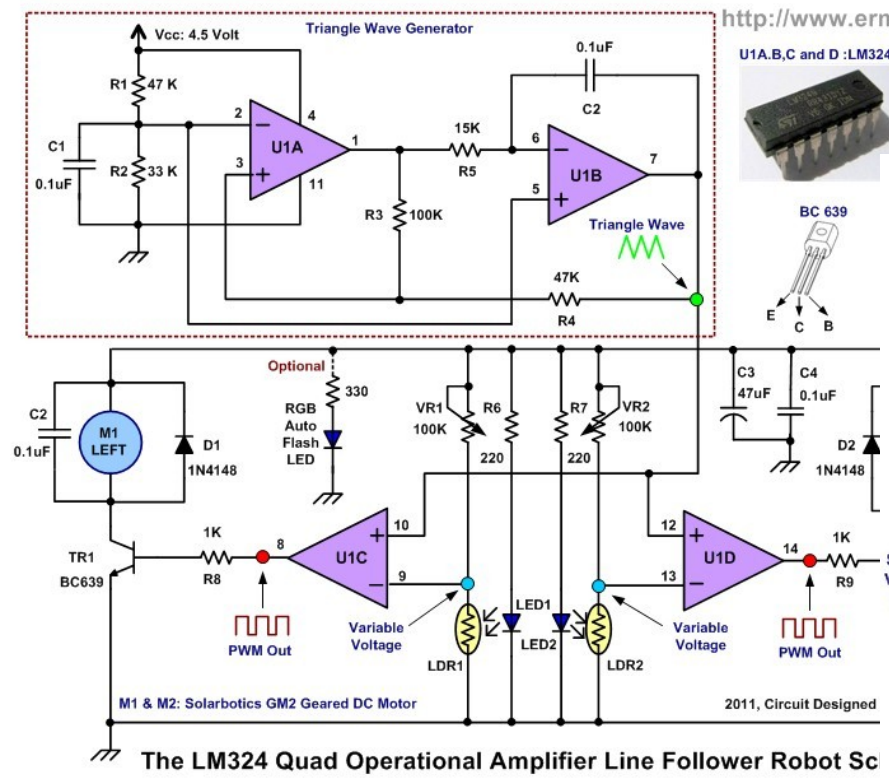
Design approaches -The PWM approach

with pulse width modulation “a technique used by micro controllers “ ,the motion of the robot would be smooth with little complications on the logic side.

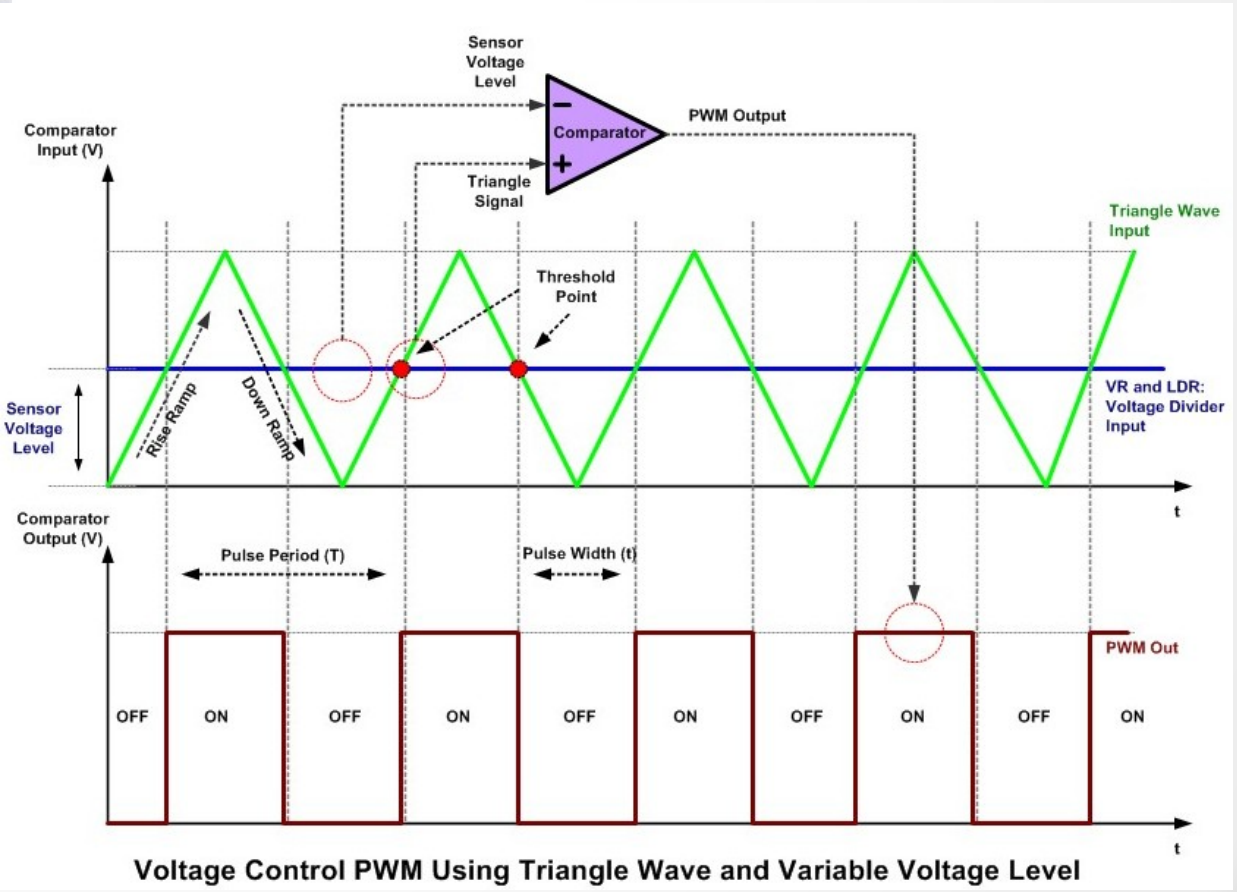


Pulse width modulation circuit

- The PWM circuit contains an op-amp comparator which compares two inputs (the triangle wave and the analog output of the line sensors) and produces a modulated digital output square signal to drive the motors



The LM324 Quad Operational Amplifier Line Follower Robot Sc

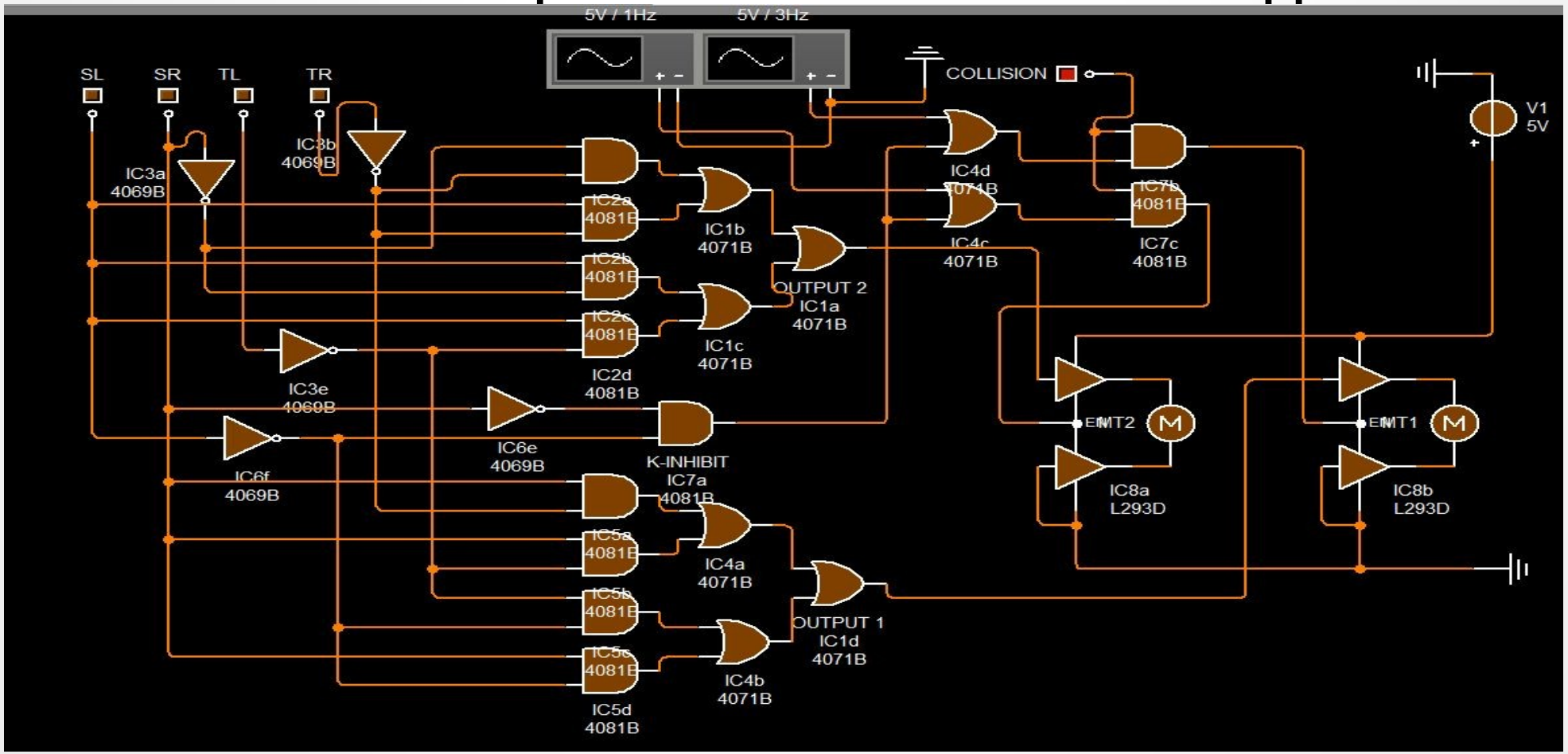


Voltage Control PWM Using Triangle Wave and Variable Voltage Level

Pulse width modulation circuit cont'd

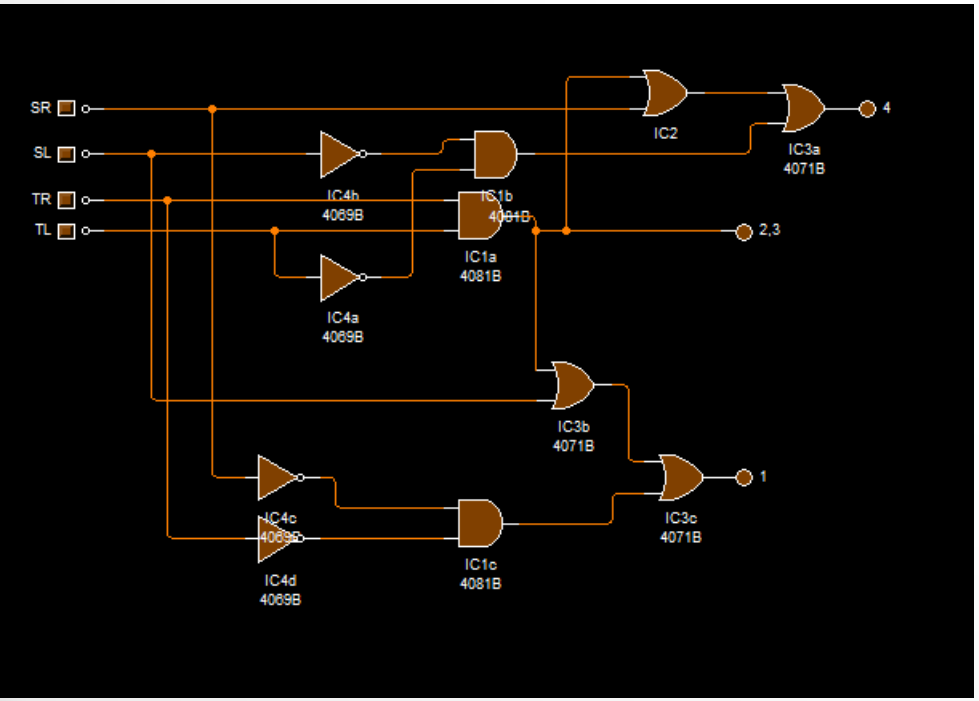
- after a couple of trials and tests on breadboards

we decided to cancel this approach due to conflicts with logic in a situation like 'the starting black node',as the pwm would completely paralyze the robot,also we were already late on schedule to deal with other subtle technical problems that come with this approach.



Another brain for the robot

- The second brain didn`t depend on the PWM and we used a vero board to make it easier to design as the PCB board would have needed a lot of jumpers
- After designing and using this logic circuit we faced the problem that the robot turns weren`t sharp enough to keep it tracking the line so we had to change that brain to the final one which we've discussed.



Sl	Sr	Tl	Tr	Input1	Input2	Input3	Input4
0	0	0	0	1	0	0	1
0	0	0	1	0	0	0	1
0	0	1	0	1	0	0	0
0	0	1	1	1	1	1	1
0	1	0	0	0	0	0	1
0	1	0	1	0	0	0	1
0	1	1	0	0	0	0	0
0	1	1	1	1	1	1	1
1	0	0	0	1	0	0	0
1	0	0	1	1	0	0	0
1	0	1	0	1	0	0	0
1	0	1	1	1	1	1	1
1	1	0	0	1	0	0	1
1	1	0	1	1	0	0	1
1	1	1	0	1	0	0	1
1	1	1	1	1	1	1	1