

# *TABLE OF CONTENTS*

## Table of Content

Acknowledgements.....	3
Chapter 1: History of Car Accidents .....	4
Chapter 2: Introduction .....	5
Chapter 3: Block Diagrams.....	7
Chapter 4: Components.....	9
Chapter 5: Hardware.....	29
Chapter 6: Software.....	39
Chapter 7: References.....	61
Chapter 8: Conclusion.....	62





## **HISTORY OF CAR ACCIDENTS**

### **❖ History of car accidents:**

Ever since cars were invented, accidents used to occur, putting the lives of many citizens in danger.

The first reported car accident was in 1869, when the Irish scientist Mary Ward was riding in a steam-powered automobile built by her cousins. As they bumped into a bend in the road, Ward was thrown from her seat and fell in the vehicle's path. One of the wheels rolled over her and broke her neck, killing her instantly.



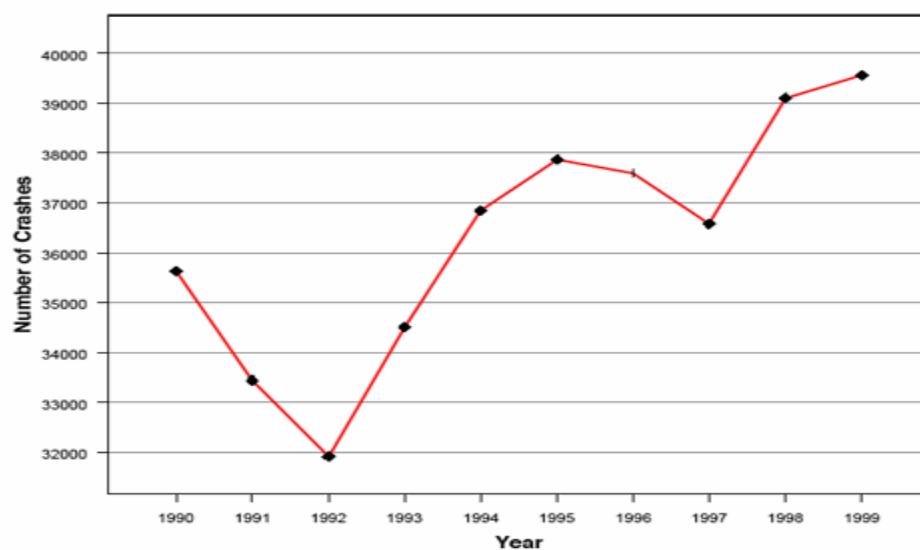
# Introduction:

Till our current days, over speeding on high ways is still one of the major deadly problems that we suffer from all around the world.

As shown below:

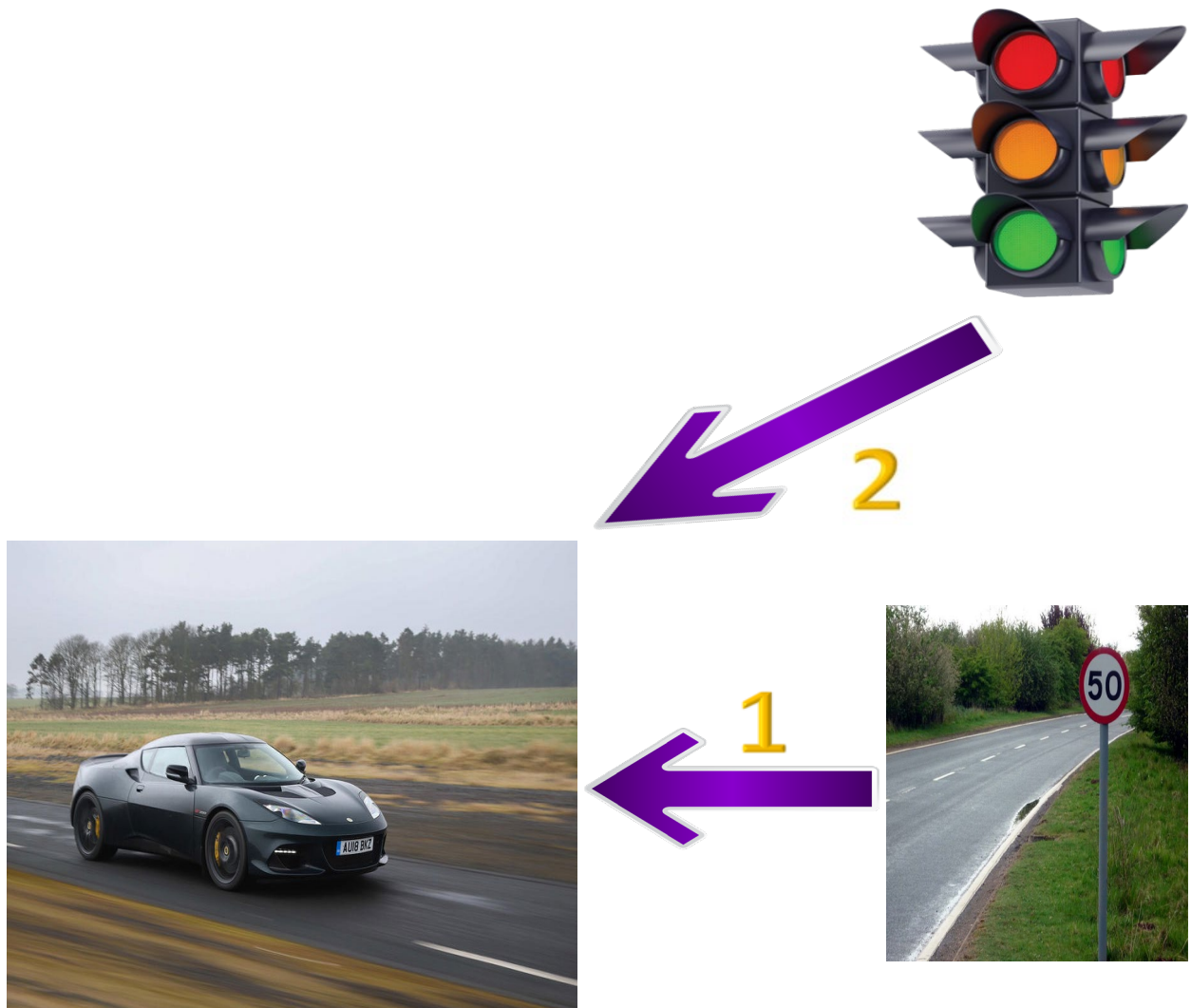


A statistical study between the years 1990 and 1999 showed that the number of people killed by car accidents per/year is increasing drastically and is still increasing till our current days; the study is shown in the graph below:



So, in order to solve this problem, we have an idea that could decrease the amount of car accidents per/year; this idea is shown down below:

# Solution:



Our idea introduce that we design a traffic speed limit system which implement a smart device in every civil moving vehicle and transform speed signs/traffic signs in to smart devices that can send speed limit and force cars to stop on traffic sign by this way we could decrease accidents by limiting the speed on highways, streets, accidents on cross roads etc... and also this idea can have many other advantages :

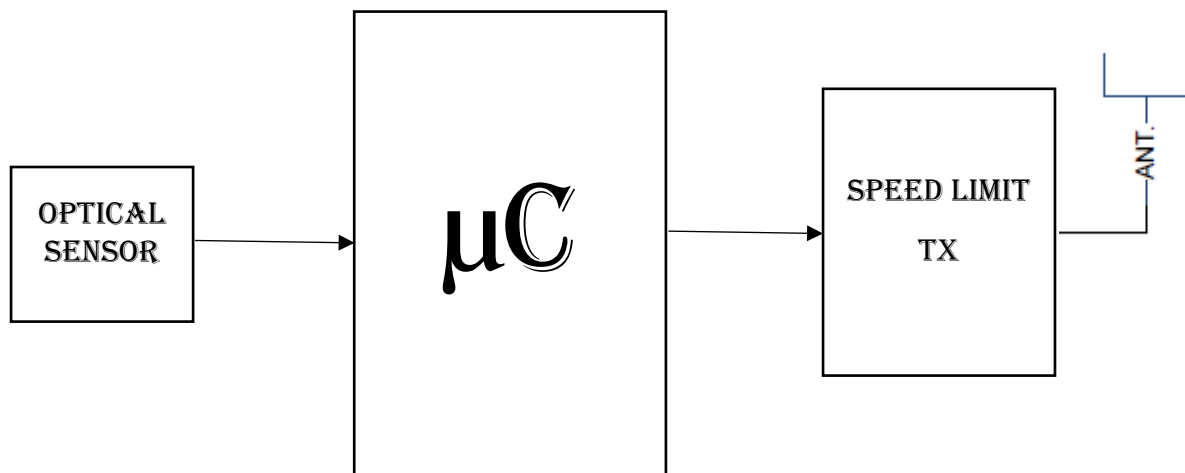
1<sup>st</sup> Decrease accidents (major point).

2<sup>nd</sup> can also decrease traffic.

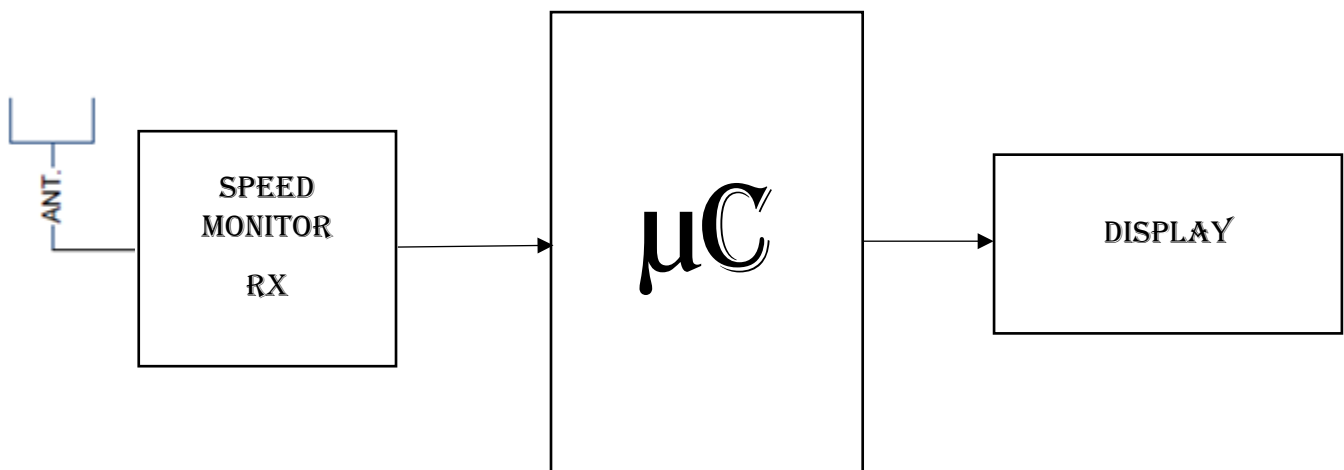
3<sup>rd</sup> can help police to capture criminals by stopping his/her car by using speed limit.

## SPEED CONTROL ON HIGH WAY BLOCK DIAGRAM

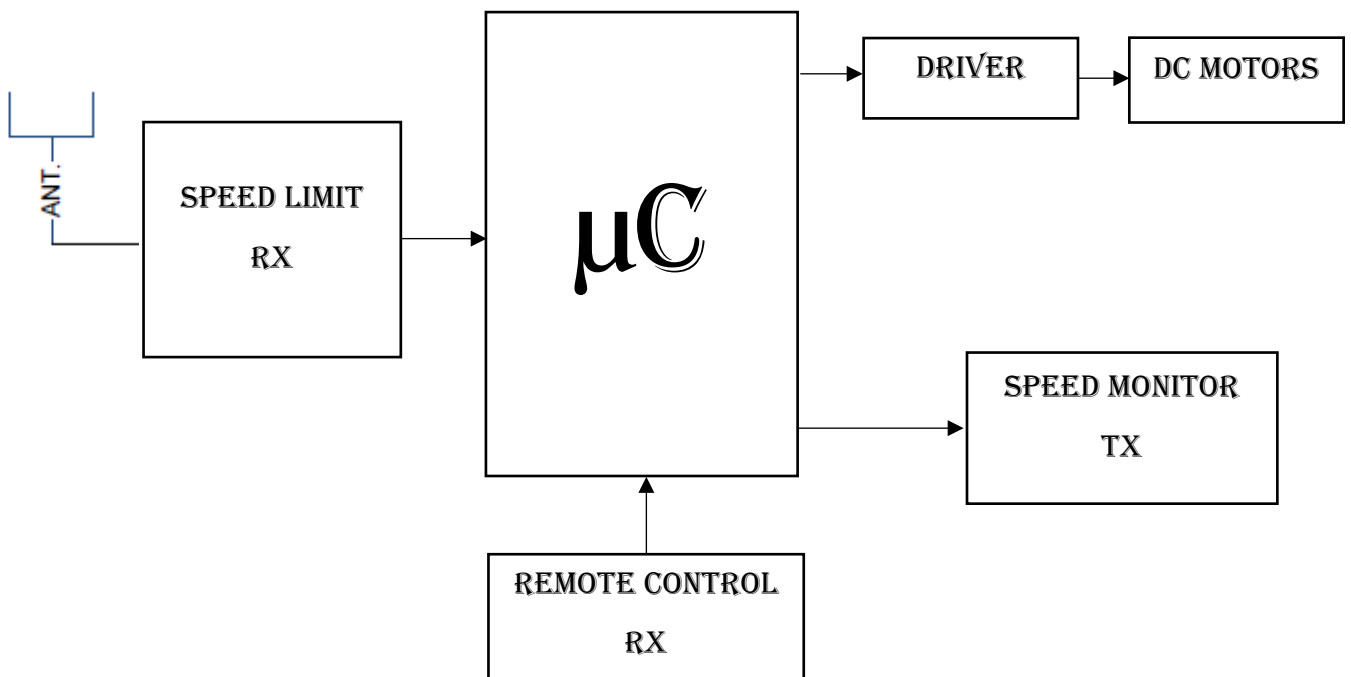
### SPEED LIMIT SIGN



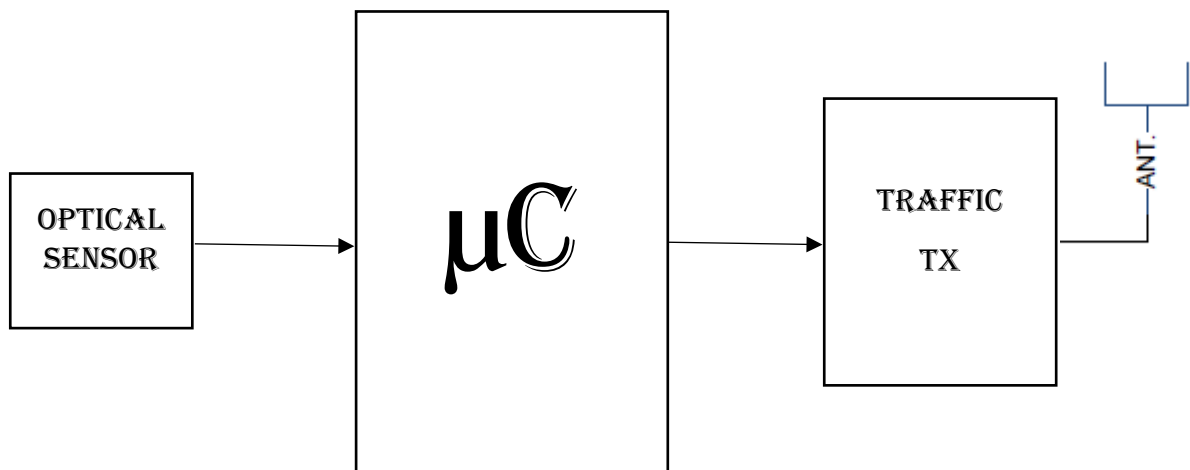
### SPEED MONITOR



## VEHICLE



## TRAFFIC SIGN





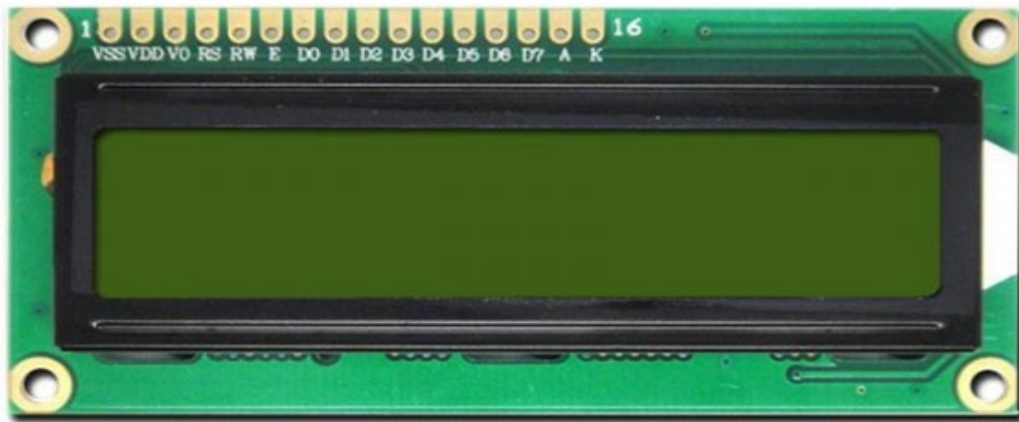
## **COMPONENTS**

**Table of components:**

<b>Number</b>	<b>components</b>
1	LCD Display 16X2
2	Motor Driver
3	PIC16F876A
4	PIC16F88
5	HC-11 Wireless Serial Port Module
6	DC Motors
7	Photo transistor
8	Laser
9	Crystal Oscillator
10	DC/DC Down Converter 15W 12V to 5V

## Components Explanation:

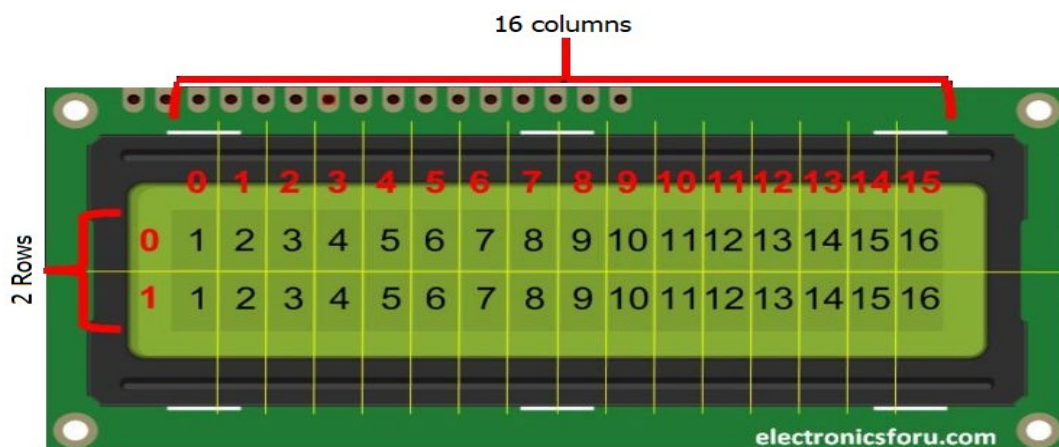
### ❖ LCD Display:



#### ➤ Introduction:

LCD (Liquid Crystal Display) screen is an electronic display used to display values, variables, etc....

Our LCD is 16x2 means it can display 16 characters per line and there are 2 such lines. In this LCD each character is displayed in 5x7 pixel matrix.



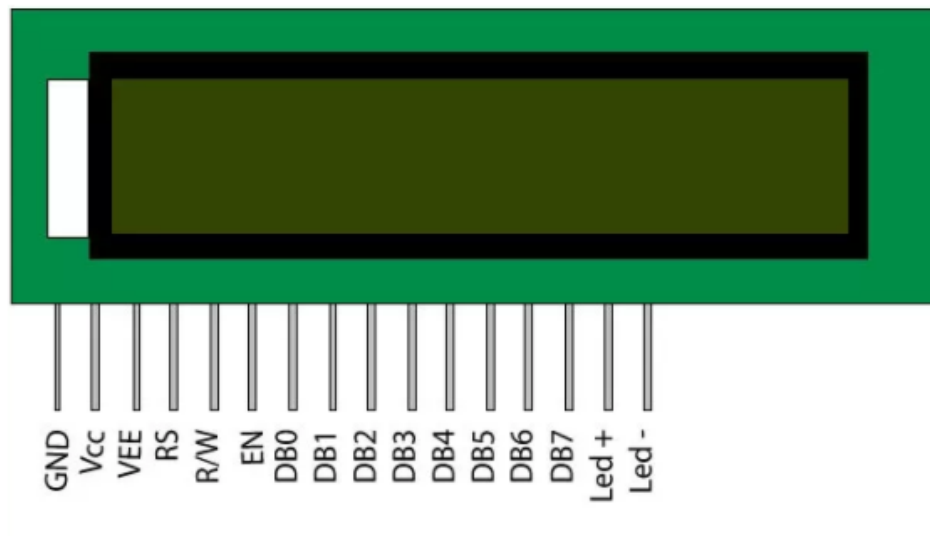
#### ➤ Principle of operation:

We have in LCD two registers (Command and Data registers).

- The command register stores the command instructions given to the LCD. A command is an instruction given to LCD to define a task like clearing its screen, controlling display, etc....
- The data register stores the data to be displayed on the LCD, as letters, numbers etc....

➤ **Pin Description:**

- Pin diagram:

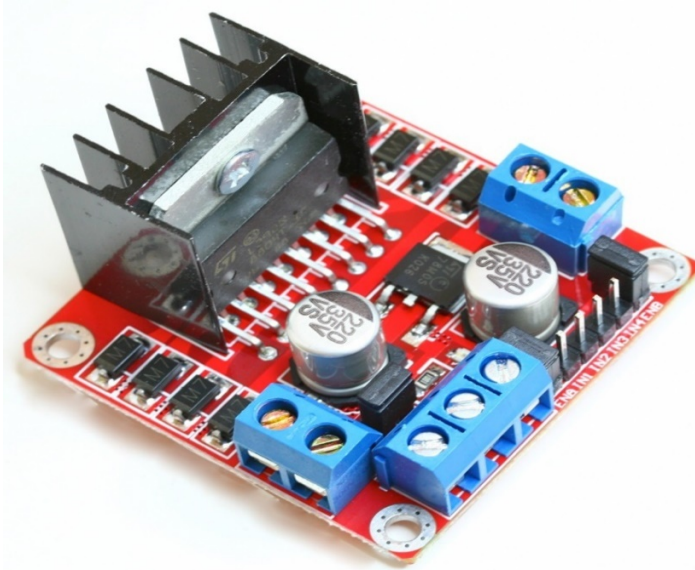


Pin No	Function	Name
1	Ground (0V)	Ground
2	Supply voltage; 5V (4.7V – 5.3V)	Vcc
3	Contrast adjustment; through a variable resistor	V <sub>EE</sub>
4	Selects command register when low; and data register when high	Register Select
5	Low to write to the register; High to read from the register	Read/write
6	Sends data to data pins when a high to low pulse is given	Enable
7	8-bit data pins	DB0
8		DB1
9		DB2
10		DB3
11		DB4
12		DB5
13		DB6
14		DB7
15	Backlight V <sub>CC</sub> (5V)	Led+
16	Backlight Ground (0V)	Led-

➤ **Utilization in project:**

In our project this component will be used to display the speed we want to limit for the car and the real speed of the car in (m/s).

## ❖ L298N Dual Motor Driver(controller):



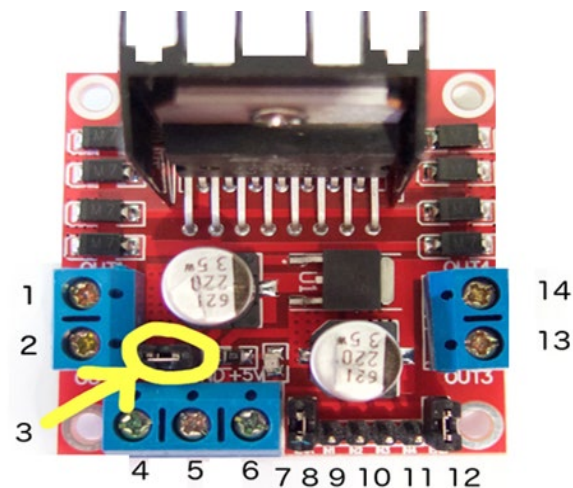
### ➤ Introduction:

This driver allows you to control the speed and the direction of two Dc motors. The L298N H-Bridge can be used with motors that have a voltage between (5 to 35v) Dc.

### ➤ Principle of operation:

- To control one or two Dc motors. First connect each motor to the A and B connections on the L298N and ensure that the polarity is the same on both inputs of the motors, then connect your power supply, the positive to pin (4) and the negative(ground) to pin 5 of the board, the power supply maximum voltage is 35v DC, but when supply voltage is greater than 12v we should remove the 12v jumper. Finally, we need six digital pins and two PWM (Pulse Width Modulation), after we connect them to the microcontroller, we will be ready to control it by programing.
- The motor direction is controlled by sending a HIGH or LOW signal to the drive for each motor (or channel). For example, for motor one, a HIGH to (IN1) and a LOW to (IN2) will cause it to turn in one direction, and a LOW and HIGH will cause it to turn in the other direction. However, the motors will not turn until a HIGH is set to the enable pin (7 for motor one, 12 for motor two). And they can be turned off with a LOW to the same pin(s). However, if you need to control the speed of the motors, the PWM signal from the digital pin connected to the enable pin can take care of it.

➤ **Pin Description:**

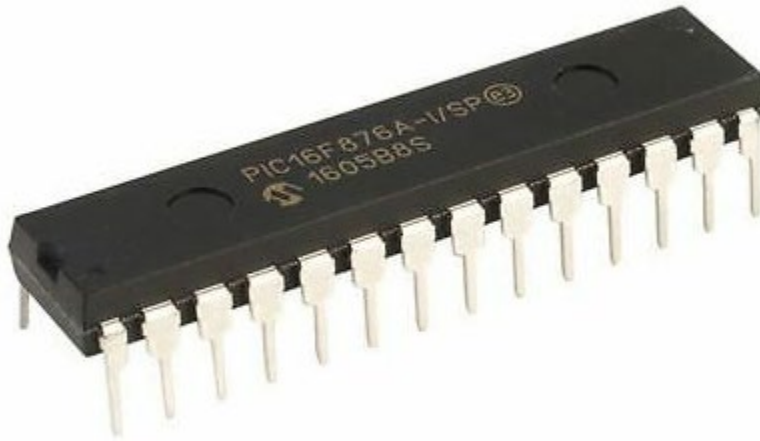


PIN	Function
1	DC motor 1 "+" or stepper motor A+
2	DC motor 1 "-" or stepper motor A-
3	12V jumper - remove this if using a supply voltage greater than 12V DC. This enables power to the onboard 5V regulator
4	Connect your motor supply voltage here, maximum of 35V DC. Remove 12V jumper if >12V DC
5	GND
6	5V output if 12V jumper in place, ideal for powering your MC (etc)
7	DC motor 1 enable jumper. Leave this in place when using a stepper motor. Connect to PWM output for DC motor speed control.
8	IN1
9	IN2
10	IN3
11	IN4
12	DC motor 2 enable jumper. Leave this in place when using a stepper motor. Connect to PWM output for DC motor speed control
13	DC motor 2 "+" or stepper motor B+
14	DC motor 2 "-" or stepper motor B-

➤ **Utilization in project:**

In our project this component will be used to control the speed of the DC motors, and controlling its direction (forward and backward).

## ❖ PIC16F876A:



### ➤ Introduction:

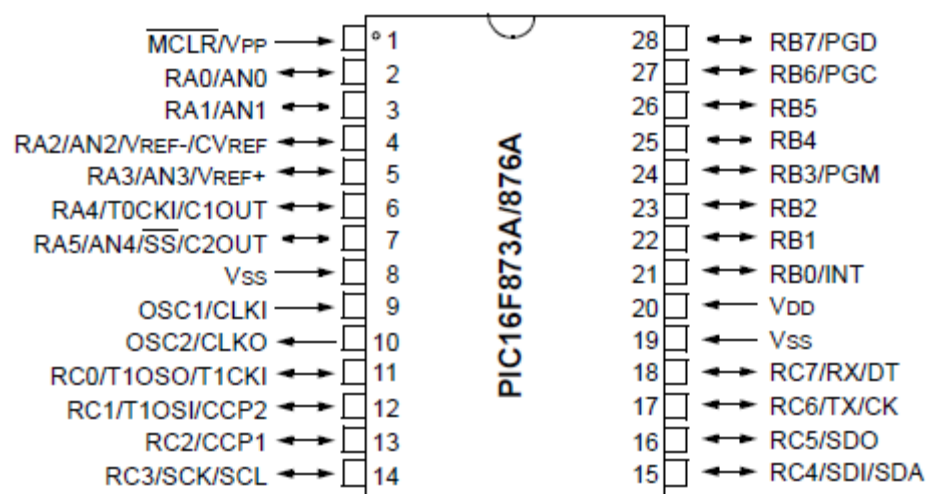
PIC16F876A is a 28-pin microcontroller, microcontroller is a processor consists from memory and ram used to control your projects, so it saves you building projects as (Home automation system etc....).

### ➤ Pin Description:

PIC16F876A has 3 Ports in total which are:

PORTA	It has 6 Pins in total starting from Pin # 2 to Pin # 7. Port A Pins are labelled from RA0 to RA5 where RA0 is the label of first Pin of Port A.
PORTB	It has 8 Pins in total starting from Pin # 21 to Pin # 28. Port B Pins are labelled from RB0 to RB7 where RB0 is the label of first Pin of Port B.
PORTC	It has 8 Pins in total. Its pins are not aligned together. First four Pins of Port C are located at Pin # 11 – Pin # 14, while the last four are located at Pin # 15 – Pin # 18

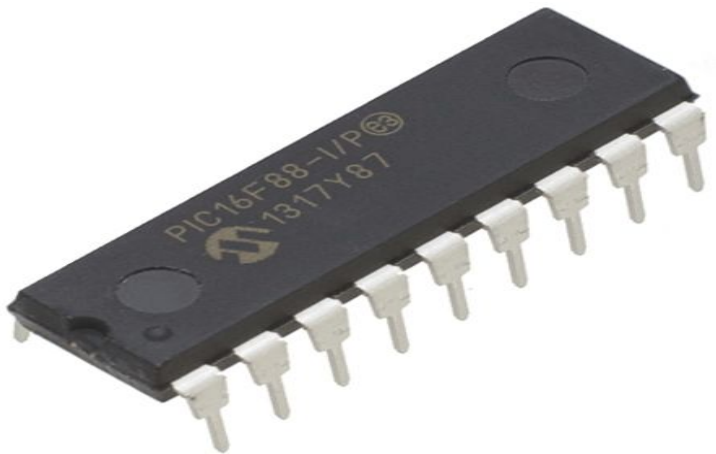
All these Ports are labelled in below figure:



➤ **Utilization in project:**

In our project this component will be the main part that is responsible for measuring the speed of the car, receive the speed limit data from the transmitter responsible for speed limiting, driving the dc motors and transmitting the speed value of car.

## ❖ PIC16F88:



### ➤ Introduction:

PIC16F88 is an 18-pin microcontroller, microcontroller is a processor consists from memory and ram used to control your projects, so it saves you building projects as (Home automation system etc....).

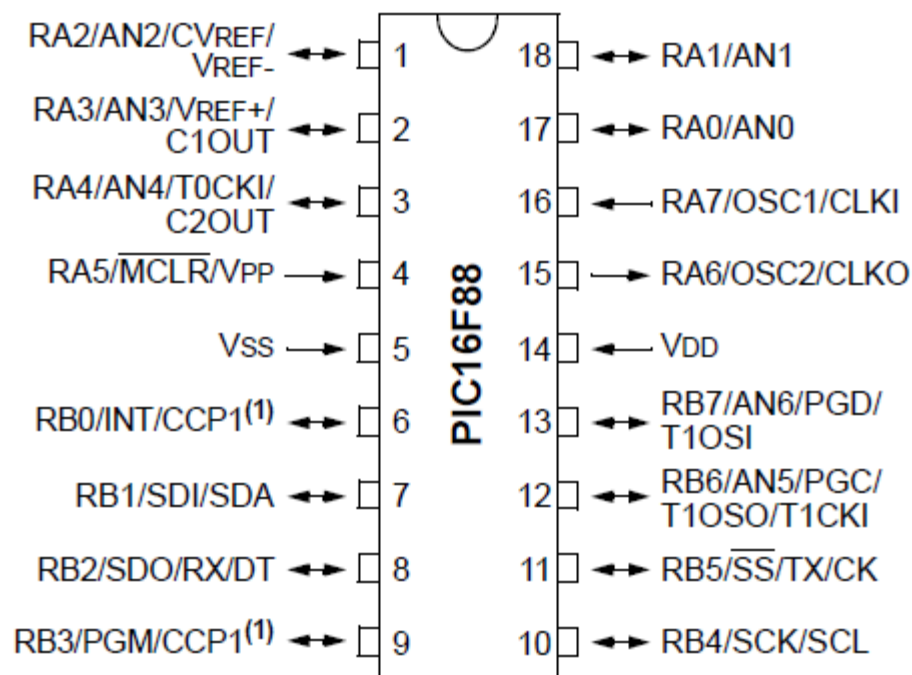
### ➤ Pin Description:

PIC16F88 has 2 Ports in total which are:

PORTA	It has 8 Pins in total. Its pins are not aligned together, RA0-RA1 are on pins number 17-18, RA2 to RA5 are on pin 1 to pin 4 RA6-RA7 are on pins number 15-16.
PORTB	It has 8 Pins in total. Its pins are not aligned together. First four Pins of Port B are located at Pin # 6 – Pin # 9, while the last four are located at Pin # 10 – Pin # 13.



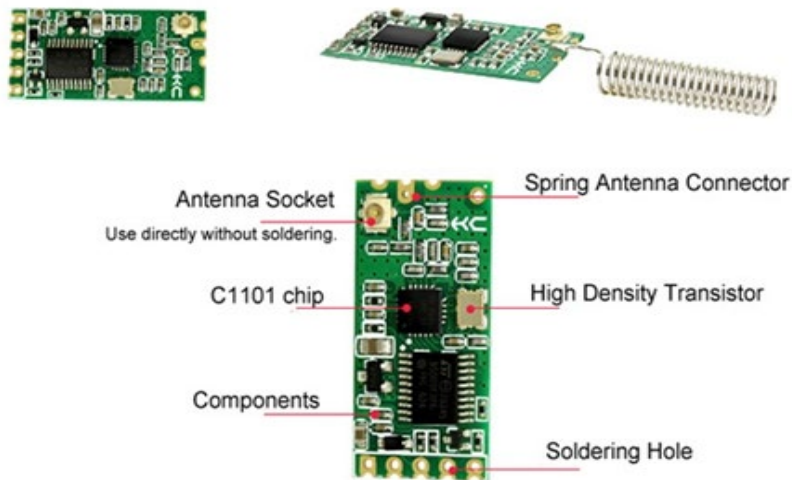
All these Ports are labelled in below figure:



➤ **Utilization in project:**

In our project this component will be used in the speed limit sign in which it transmits the speed limit value to vehicle, also will be used to receive the speed limit value and the speed value of the vehicle then display the data.

## ❖ HC-11 Wireless Serial Port Module:



### ➤ Introduction:

HC-11 433MHz with Wireless RF FSK Transceiver or (Tx and Rx) Module with Spring Antenna of multi-channel embedded wireless transmission module. Band Width is 434.4—439.0MHz. It can set multiple channels with a Step 400KHz for each channel, with 20 channels in total. The maximum transmit power is (10dBm) , with 40m communication distance in wide open areas.

### ➤ Pin Description:

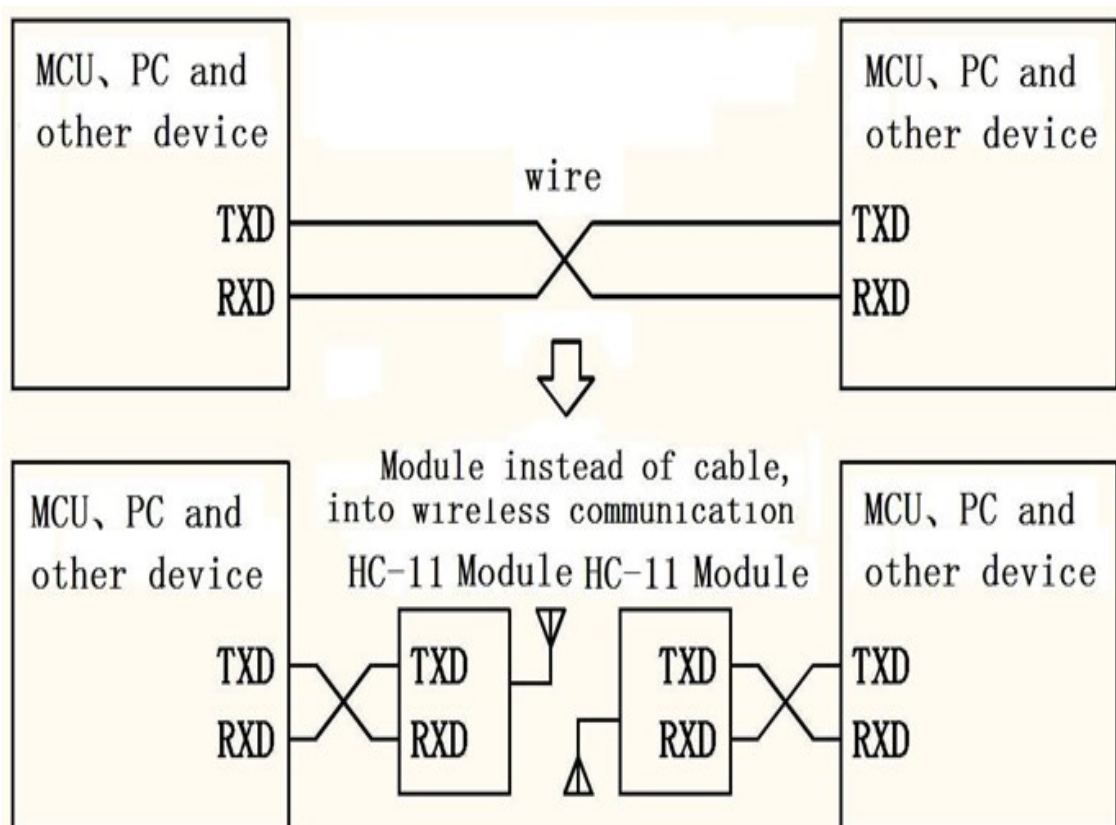
Pin diagram:



pin	definition	I/O	explain
1	VCC		Power pin, the requirements of 3.2V to 5.5V DC power supply, the supply current is not less than 100mA.
2	GND		Ground
3	RXD	input, with pull up resistor to internal power supply	UART input, 3.3V TTL level, internal 1K resistor in series
4	TXD	Output, with pull up resistor to external VCC	UART output, 3.3V TTL level, internal 1K resistor in series
5	SET	input, internal 10K pull up resistor	Parameter setting pin, the low level is effective, internal 1K resistor in series
6	ANT	RF input/output	433MHz antenna pin
7	GND		Ground
8	GND		Ground
9	NC		NC
ANT1	ANT	RF input/output	IPEX20279-001E-03 antenna seat
ANT2	ANT	RF input/output	433MHz spring antenna welding hole

➤ **Principle of operation:**

**1. Brief introduction of the principle of operation:**



As shown in the figure above hc-11 module used to replace physical cables with wireless modules

Were each module connected with a MCU which take TX pin of MCU to RX pin of hc-11 and RX pin of

MCU to TX pin of hc-11,each module on it's own works in halp duplex but full duplex with each other.

## 2. HC-11 Serial communication data registers (RX/TX) :

### SCDR Register

Address:	\$102F							
	Bit 7	6	5	4	3	2	1	Bit 0
Read:	R7	R6	R5	R4	R3	R2	R1	R0
Write:	T7	T6	T5	T4	T3	T2	T1	T0

SDCR Register is the main register in the HC-11 module it consists of two 8bit registers Read register and write register.

Read Register (RX) : used for data reception.

Write Register (TX) : used for data transmission.

## 3. Modules Modulation (FSK) Principle Of Operation:

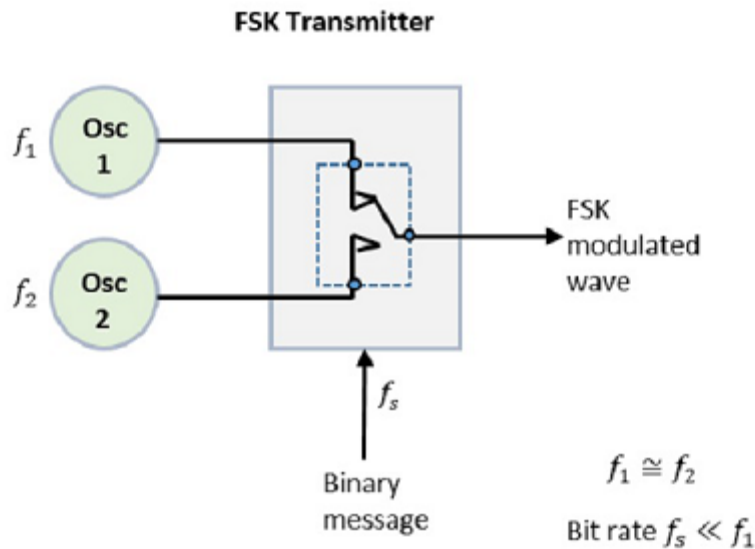
Types of FSK modulation:

- 1- Binary level FSK (1,0) where  $M=2^1$  signal levels
- 2- M-ary FSK (11,00,10,01) where  $M=2^2=4$  signal levels

The main difference between the two types that only M-ary fsk have higher bitrate in sending data because it have more signal levels so the data rate increase but here we will explain about the binary fsk since hc-11 operates according to it's principle.

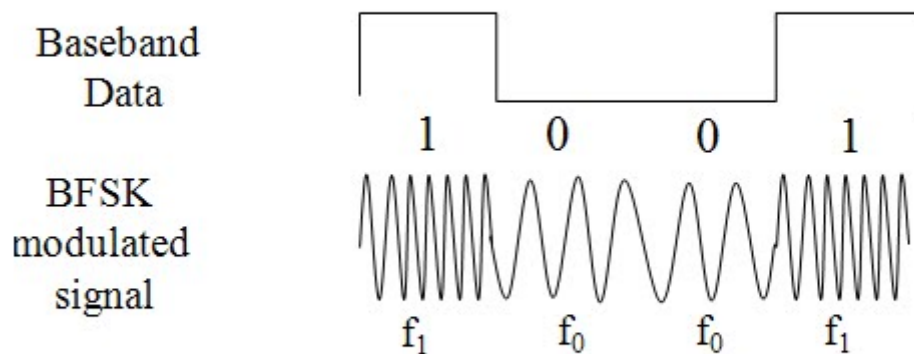
- Binary level FSK :

Block diagram of fsk modulator:



Wave forms:

## Frequency Shift Keying (FSK)



$$\text{where } f_0 = A \cos(\omega_c - \Delta\omega)t \text{ and } f_1 = A \cos(\omega_c + \Delta\omega)t$$

The above signals represent the inputs and output of the modulator block diagram so baseband data is the binary code  $\text{rect}(t)$ , carries signals  $(f_1(t), f_0(t))$ , output signal is the  $\text{FSK}(t)$ .

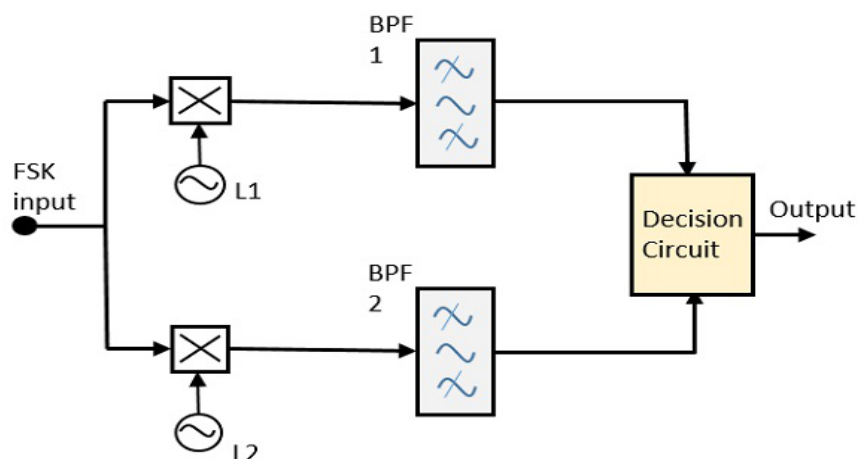
When the binary code =1 then the output  $FSK(t)=A\cos(2\pi f_c+f_1)$  and when the binary code =0 then  $FSK(t)=A\cos(2\pi f_c+f_0)$ , such that  $f_1 \gg f_2$ .

Block diagram of fsk demodulator:

- 1- Asynchronous Detector (non-coherent)
- 2- Synchronous Detector (coherent)

A receiver is called coherent if it's in phase with the transmitter signal when a receiver is out of phase with the transmitter is called non-coherent, here we're going to discuss coherent detector.

Block diagram of Coherent detector:



Rule of each element :

- L1 and mixer used as frequency up converter to raise the carrier frequency above noise.
- BPF used to limit noise in bandwidth in the received signal.
- Decision device it could be a comparator used to detect if the received signal is (1 or 0).

#### ➤ Utilization in project:

This component is used in our project as serial communication device to send/receive serial data between all devices in the project.

## ❖ Dc Motor:

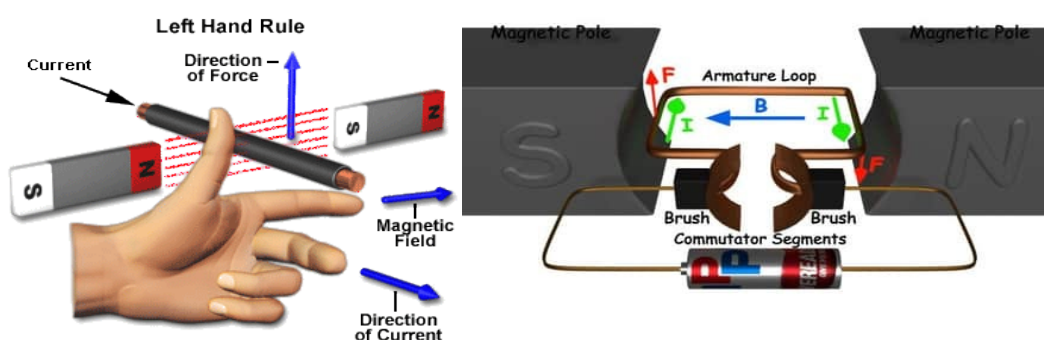


### ➤ Introduction:

The DC motor is an electrical machine which converts electrical energy into mechanical energy.

### ➤ Principle of operation:

DC motor is a machine that converts dc power into mechanical energy. Its operation is based on the principle that when a current carrying conductor is placed in a magnetic field, the conductor experiences a mechanical force. The direction of the force is given by Fleming's left-hand rule.



### ➤ Utilization in project:

In our project this component will be used to enable us moving the car.

## ❖ Phototransistor:

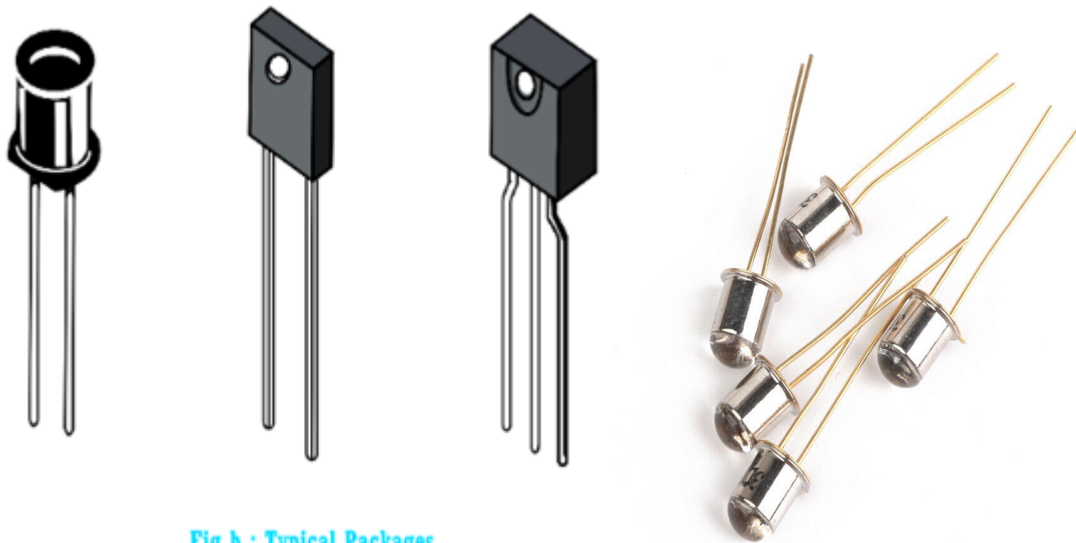
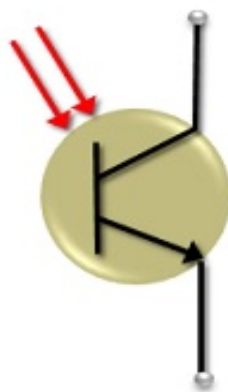


Fig b : Typical Packages

### ➤ Introduction:






Phototransistors convert the incident light into electrical signal. Instead of providing the base current for triggering the transistor, the light rays are used to control the base region. The base terminal is made up of the material which shows sensitivity towards the light. The two arrows point towards phototransistor indicates that it is triggered by the light incident on it. The circuit symbol of the phototransistor is described in the diagram below:



Symbolic  
representation of  
Phototransistor



➤ **Principle of operation:**

- The output of the phototransistor is taken from the emitter terminal and the light rays are allowed to enter the base region. The value of the current of the electric signal generated by the phototransistor depends on the light intensity of the light falling on the base of transistor.
- The phototransistor can be operated in three regions that are the cut-off region, active region, and the saturation region.
  - 1) The cut-off region and saturation region: can be used to operate the transistor as the switch.
  - 2) The active region: is used for generating current. It is depending on several factor such as:
    1. DC current gain of the transistor
    2. Time constant
    3. Luminous Sensitivity
    4. Area of the collector-base junction
    5. Wavelength of the incident light
- Applications of Phototransistors
  -  Counting Systems.
  -  Encoder sensing and object detection.
  -  Printers and Optical control remotes.
  -  Light detector.
  -  Level Indication and Relays.

➤ **Utilization in project:**

In our project this component will be used as a photo detector which is supplied with laser light source. The aim is to check if the car pass or not.

## ❖ Laser Sensor (KY-008):

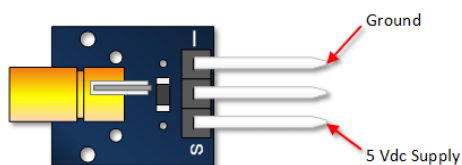


### ➤ Introduction:

The Laser sensor (KY-008) gives a small light beam or emits a dot shaped, red laser beam.

### ➤ Principle of operation:

The module has three pins from left to right (pin 1) is signal output, (pin 2) is +5 volts (DC), and (pin 3) is GND. Using this system. This sensor theoretically measures up to 1000 cm or 10000 mm. Laser is basically concentrated on light source. LDR detector (Light Dependent Resistor).



### ➤ Characteristics:

The KY-008 Laser transmitter module consists of a 650nm red laser diode head and a resistor.

Operating Voltage	5V
Output Power	5mW
Wavelength	650nm
Operating Current	less than 40mA
Working Temperature	-10°C ~ 40°C [14°F to 104°F]
Dimensions	18.5mm x 15mm [0.728in x 0.591in]

☞ **Warning:** This is a low power laser device, however as with all laser devices care should be taken when in use. You should never look directly in to its beam or point the laser at another person. Doing so may cause permanent eye damage. This item is not suitable for children.

➤ **Utilization in project:**

In our project this component will be used as continuous light source propagated toward the phototransistor. The aim of the laser is to let the photo transistor detects if the car passed.

## ❖ crystal oscillator:



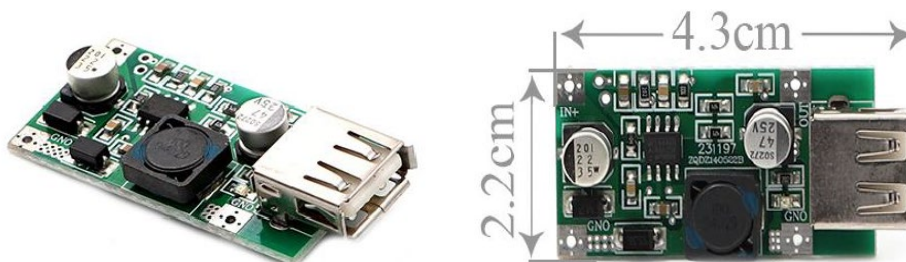
### ➤ Introduction:

The crystal oscillator is an electronic device that generates electric signal by the vibrating of the crystal (piezo electric).

### ➤ Utilization in project:

In our project this component will be used to determine the oscillation speed that the microcontroller operates on effective ( $f_{oscillator} = f_{oscillator}/4$ ).

## ❖ DC/DC Down Converter 15W 12V to 5V:



### ➤ Introduction:

The DC/DC Down Converter is step down converter that converts an input DC volt (5-35v) to (5v) at output.

### ➤ Characteristics:

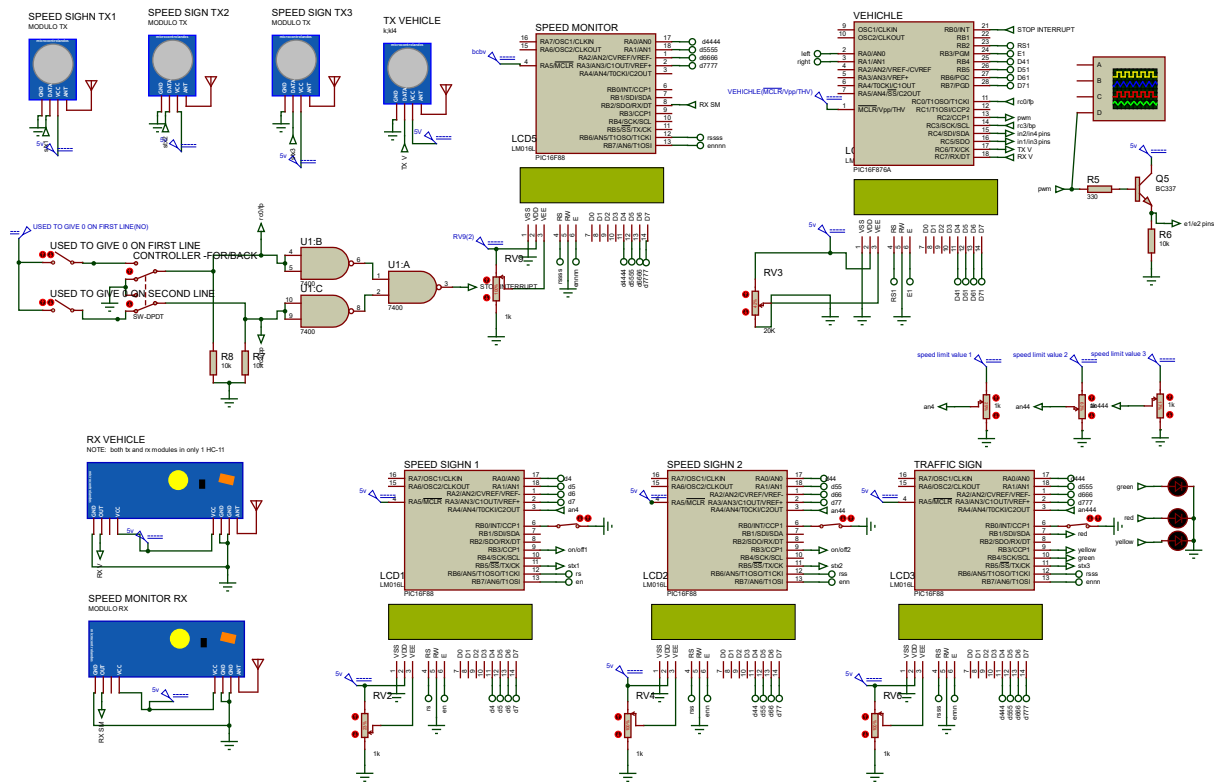
- ❖ Input Voltage: 5-35V.
- ❖ Output Voltage: 3-25V, the default voltage is 5V.
- ❖ Current: 3A
- ❖ Interface: USB
- ❖ Size: 4.3x2.2cm

### ➤ Utilization in project:

In our project this component will be used to supply the two sign transmitters and the traffic transmitter and remote-control display.

# HARDWARE

## Project schematic:

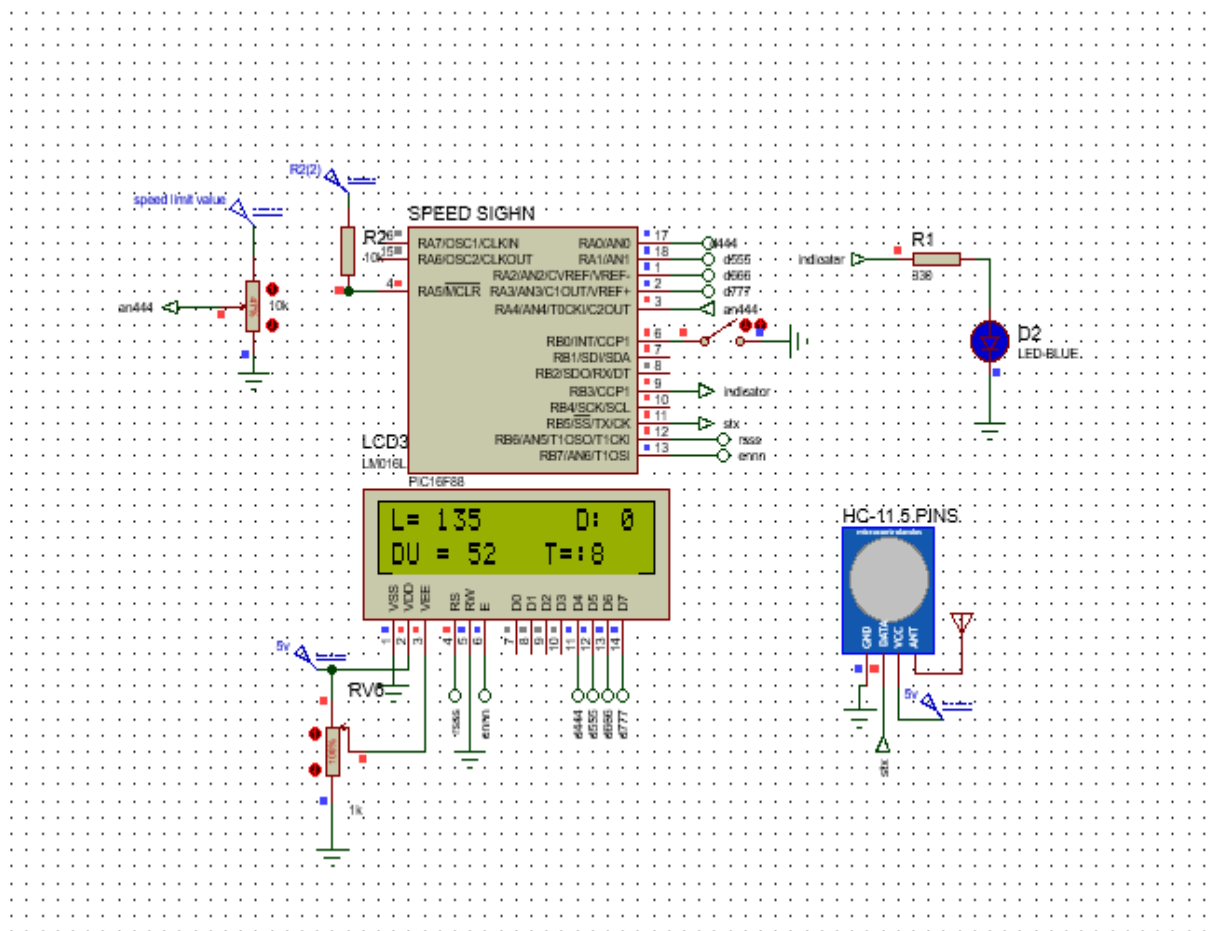


## Components:

Components	QNT	Value
PIC16F88	4	-
PIC16F876A	1	-
IC817 (Transistor)	1	-
IC74LS00(NAND)	1	-
Display	4	-
Laser Diode	3	-
Photo Transistor	3	-
Oscillator	3	20MHZ
LED	5	-
Screw socket	5	-
USB Socket	5	-
Capacitors	13	100nf
Capacitors	8	22pf
Capacitors	5	470μF

Variable Resistor	3	10k $\Omega$
Resistors	3	150 $\Omega$
Resistors	8	10k $\Omega$
Resistors	9	330 $\Omega$
Resistors	11	220 $\Omega$
Resistors	5	0 $\Omega$

#### ❖ Limit sign:

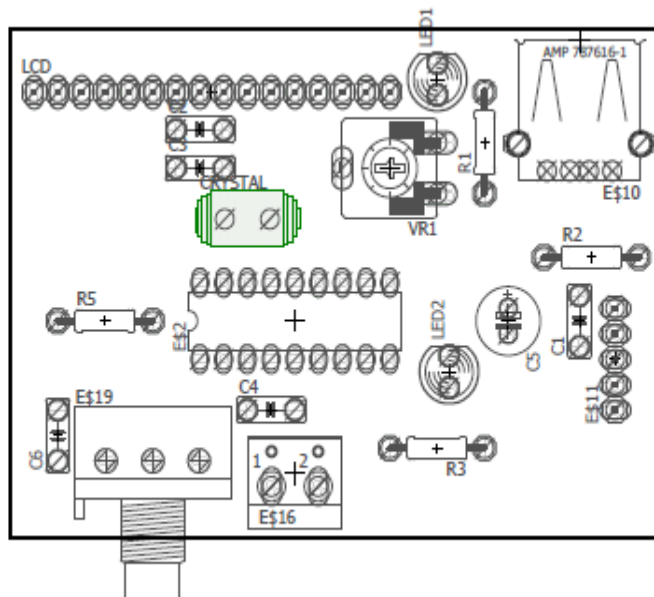


#### ➤ Principle of Operation:

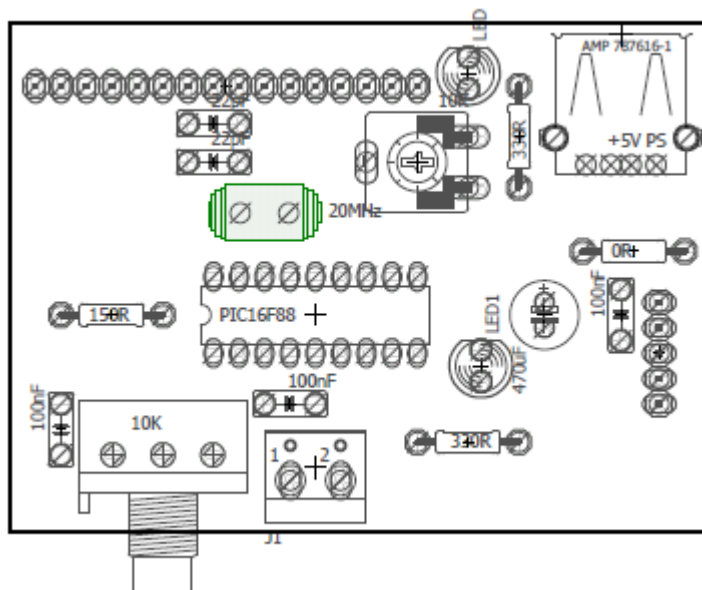
This section is used in our project to read the value of variable resistor then convert it to digital value, also display the value on 16x2 LCD display and finally transmit the limit frames to the vehicle.

➤ **Boards:**

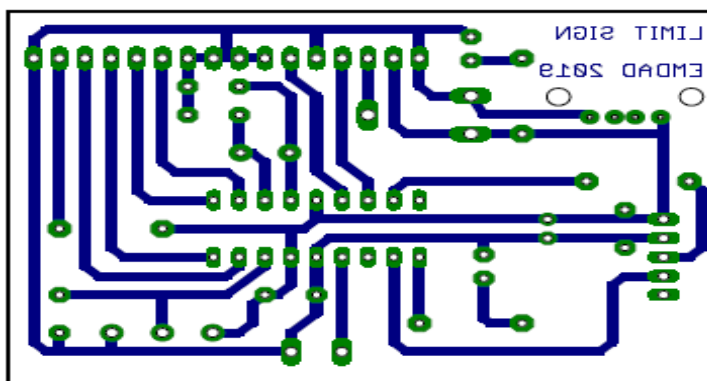
- Names:



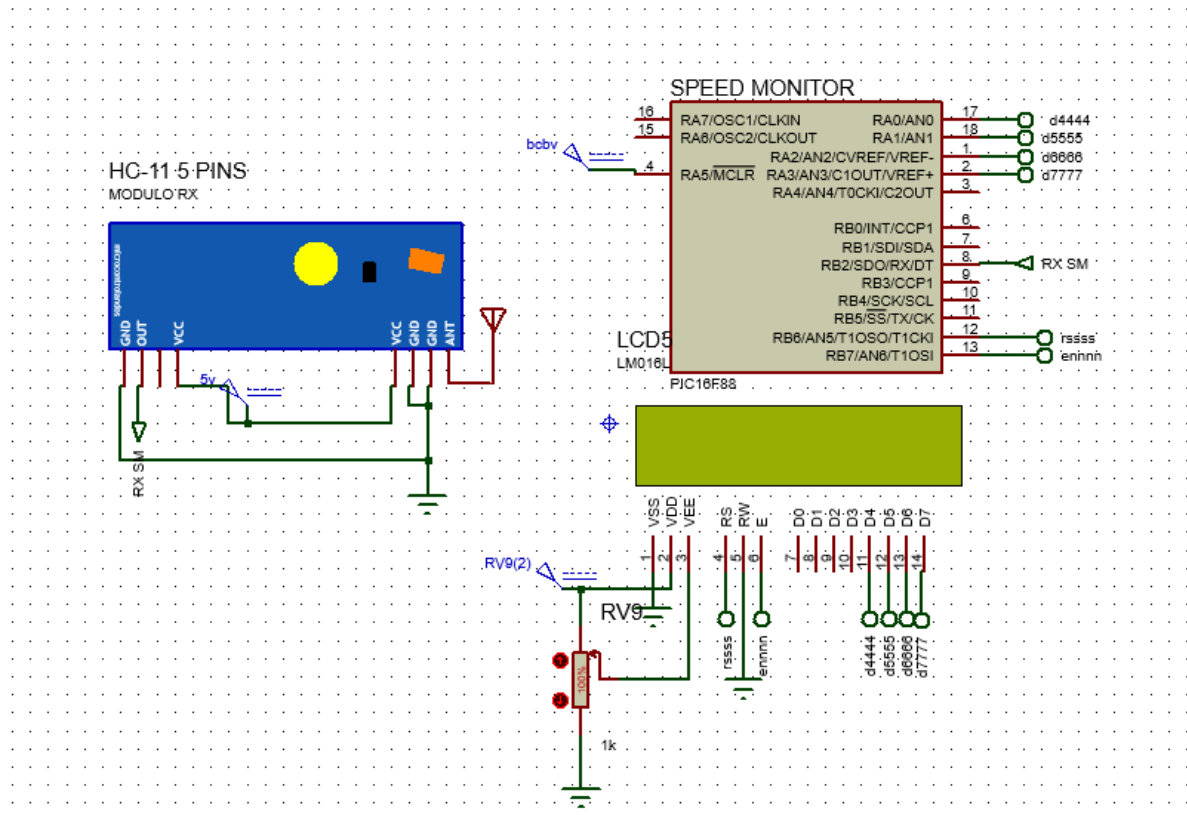
- Values:



- Wires:



## ❖ Speed Monitor:

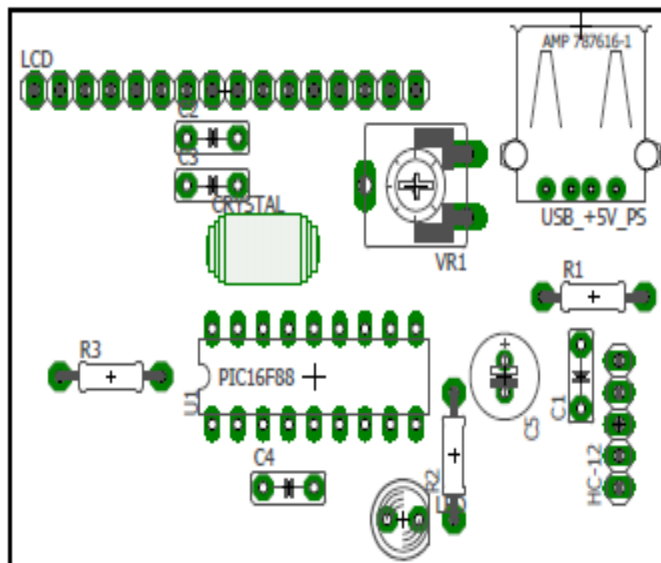


### ➤ Principle of Operation:

This section is used in our project to monitor the data on vehicle, by receiving frames from the vehicle and reconstruct data.

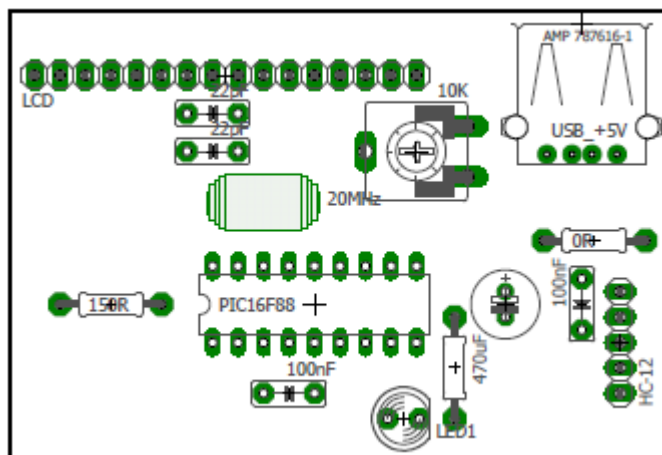
### ➤ Boards:

- Names:

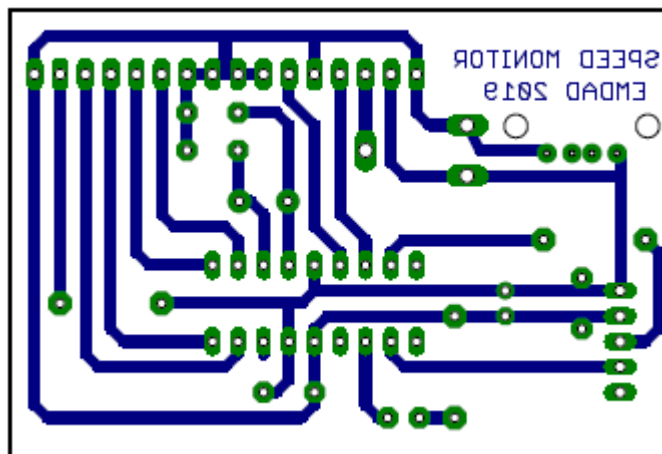




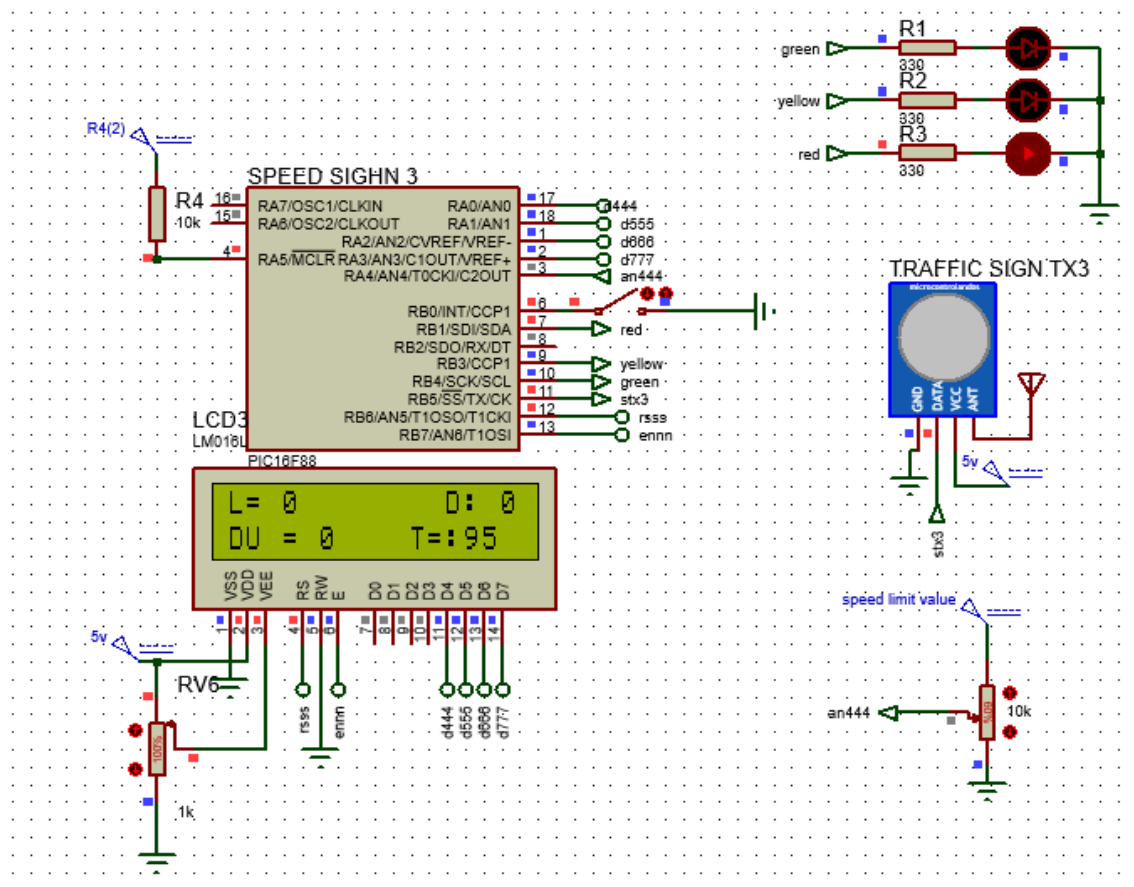
- Values:



- Wires:



## ❖ Traffic Sign:

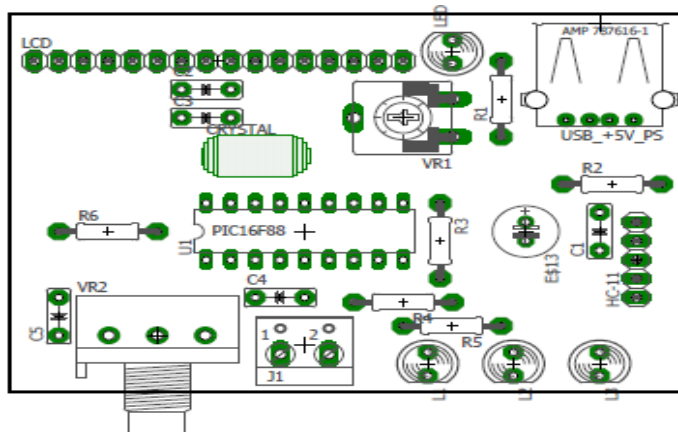


### ➤ Principle of Operation:

This section is used in our project as a traffic sign when car passes on the sign the MC wake up and then send stop frames to the car and turn red led on for 5s then the yellow led turn on for 2s and finally the MC read the value of variable resistor then convert it to digital value ,then transmits move frames to the car and turn on the green led for 5s then the MC go back to sleep.

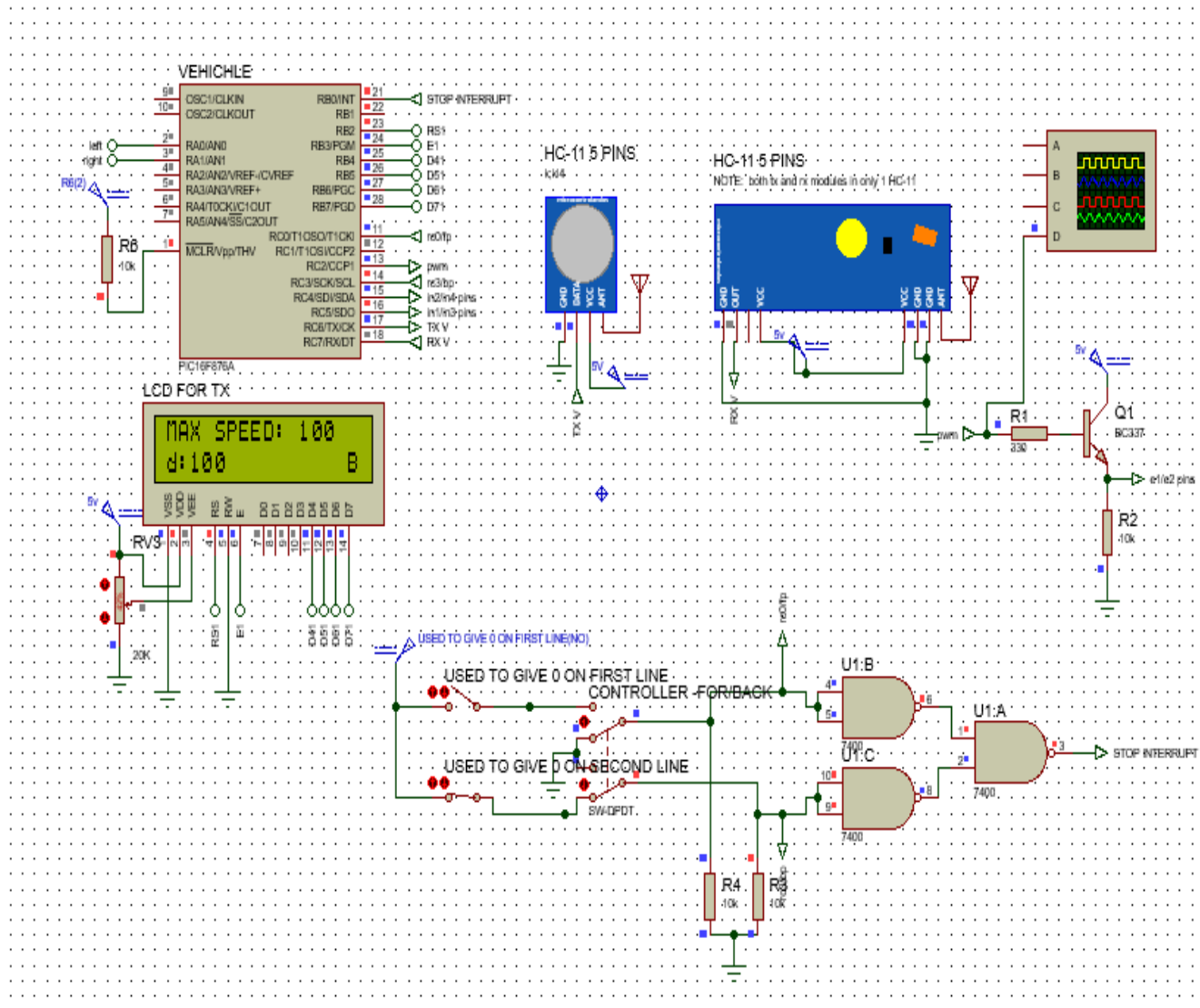
### ➤ Boards:

- Names:





## ❖ Main Car:

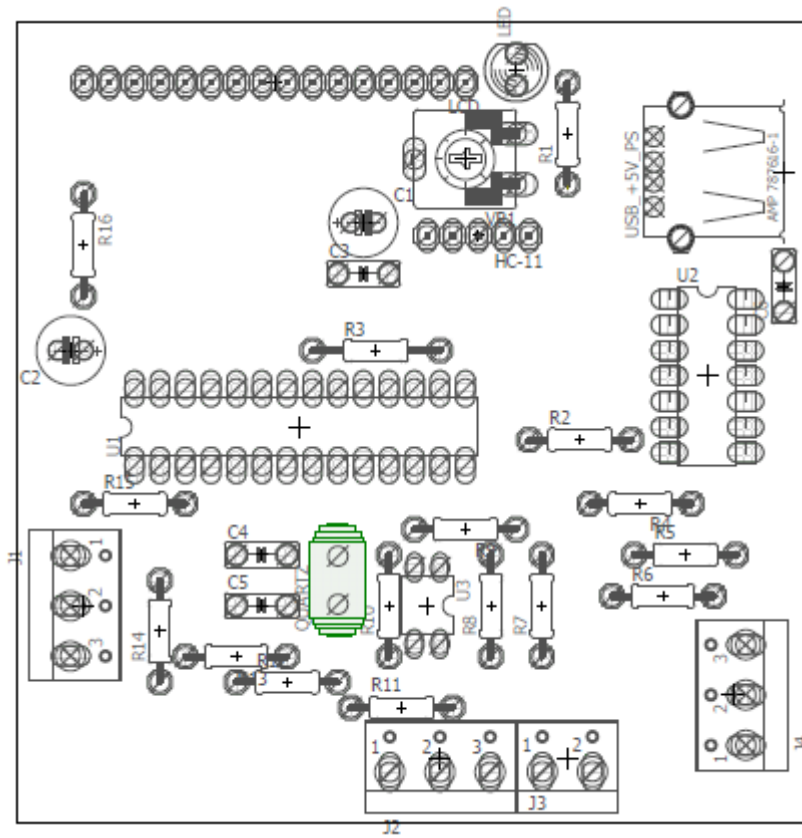


### ➤ Principle of Operation:

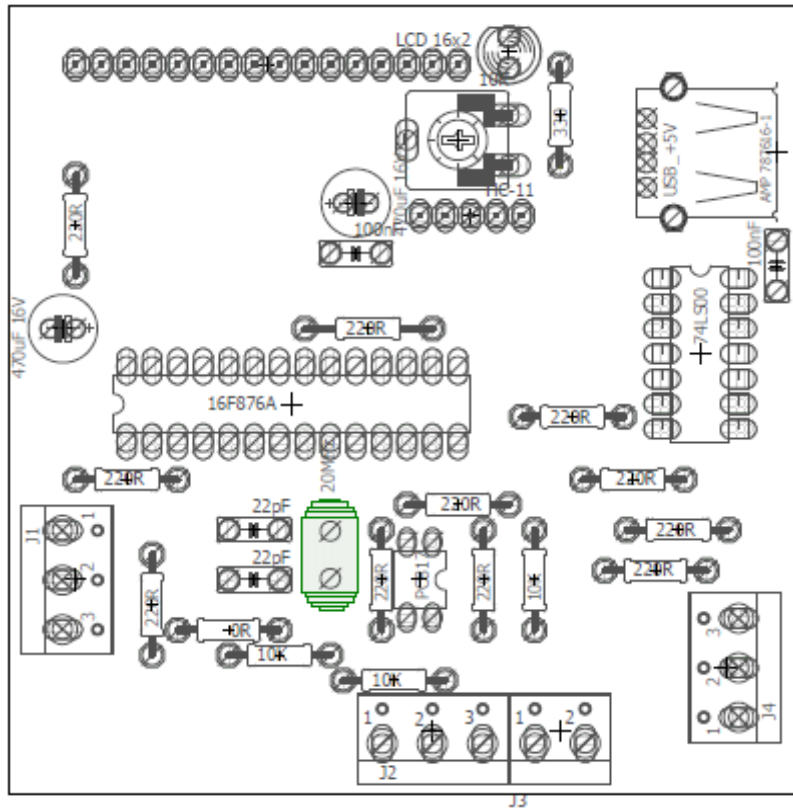
This section is used as the main part of the project which controls the car moving, receives the data from limit sign and then transmits the data to speed monitor.

➤ **Boards:**

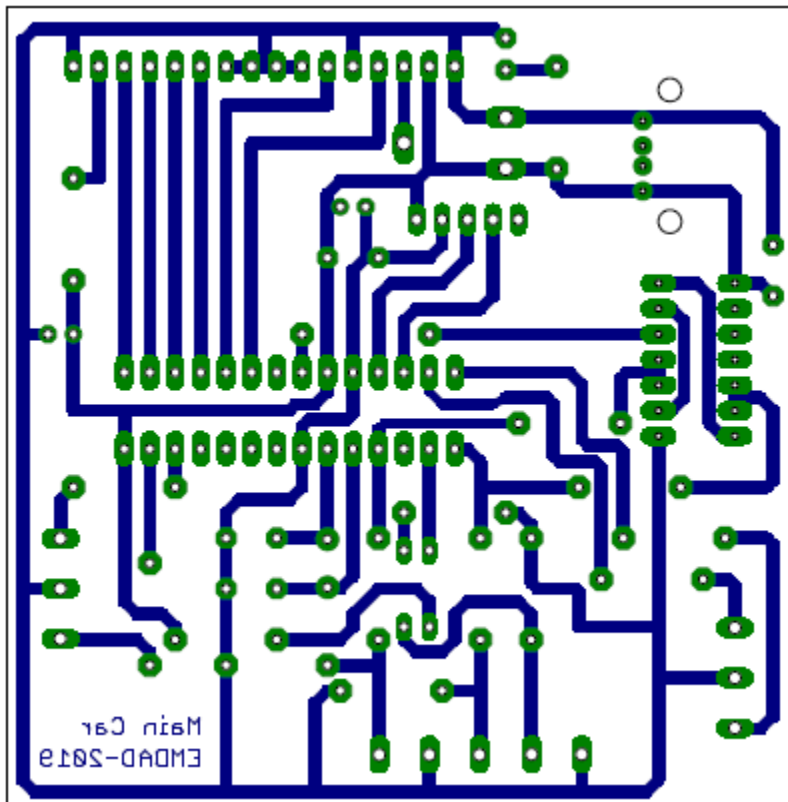
- Names:



- Values:



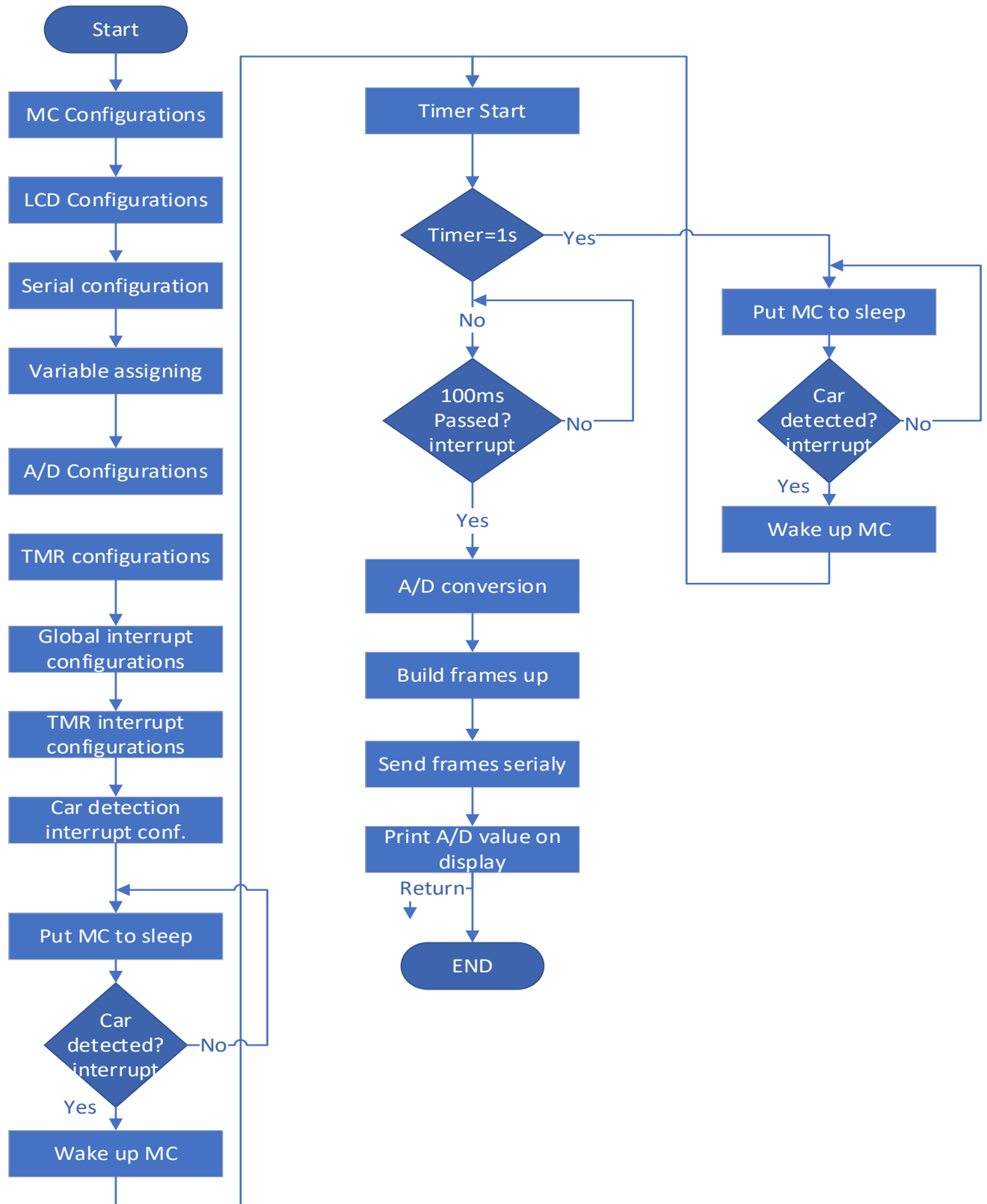
- Wires:



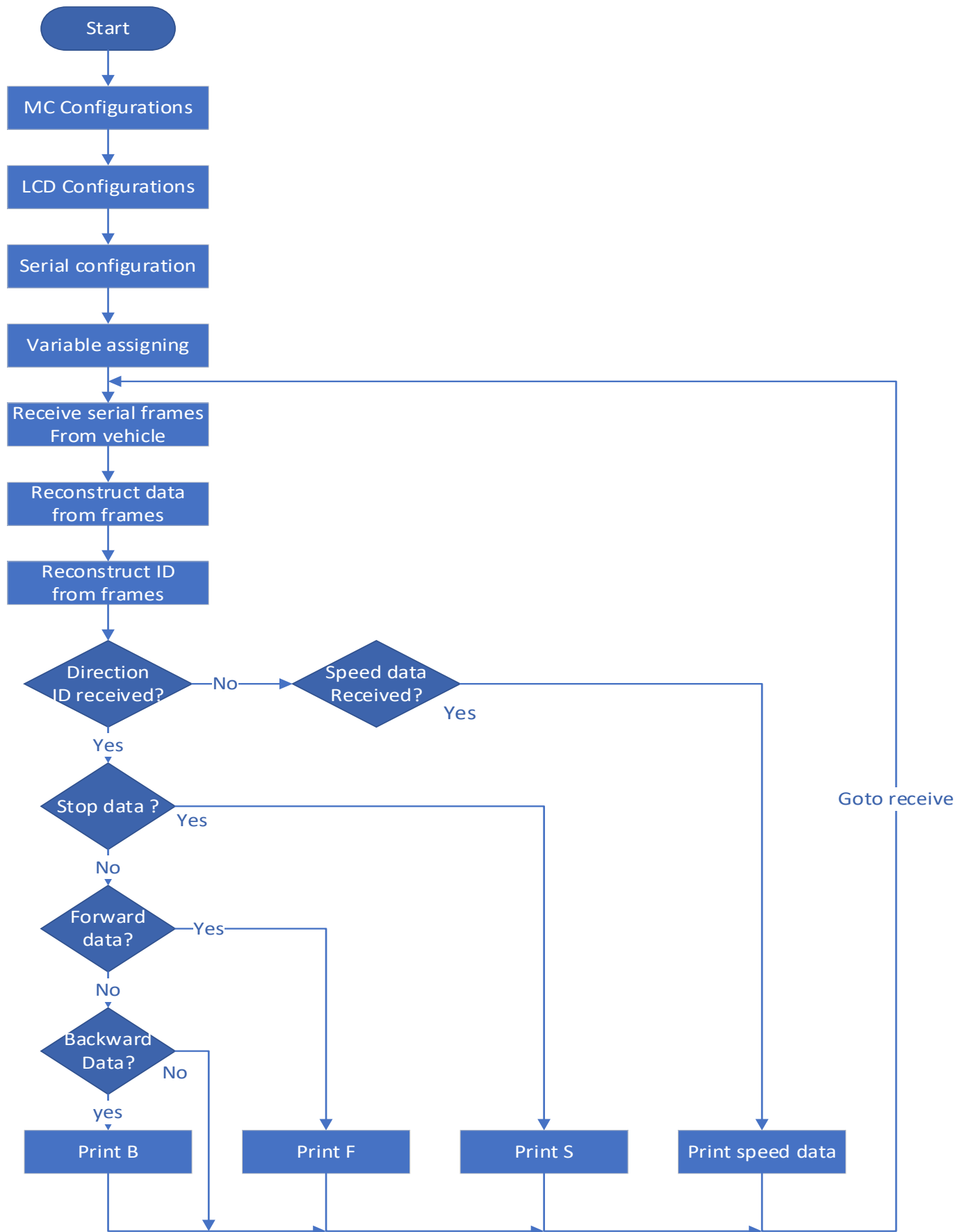
## SOFTWARE

### ❖ Flow Charts

#### ➤ Limit sign:

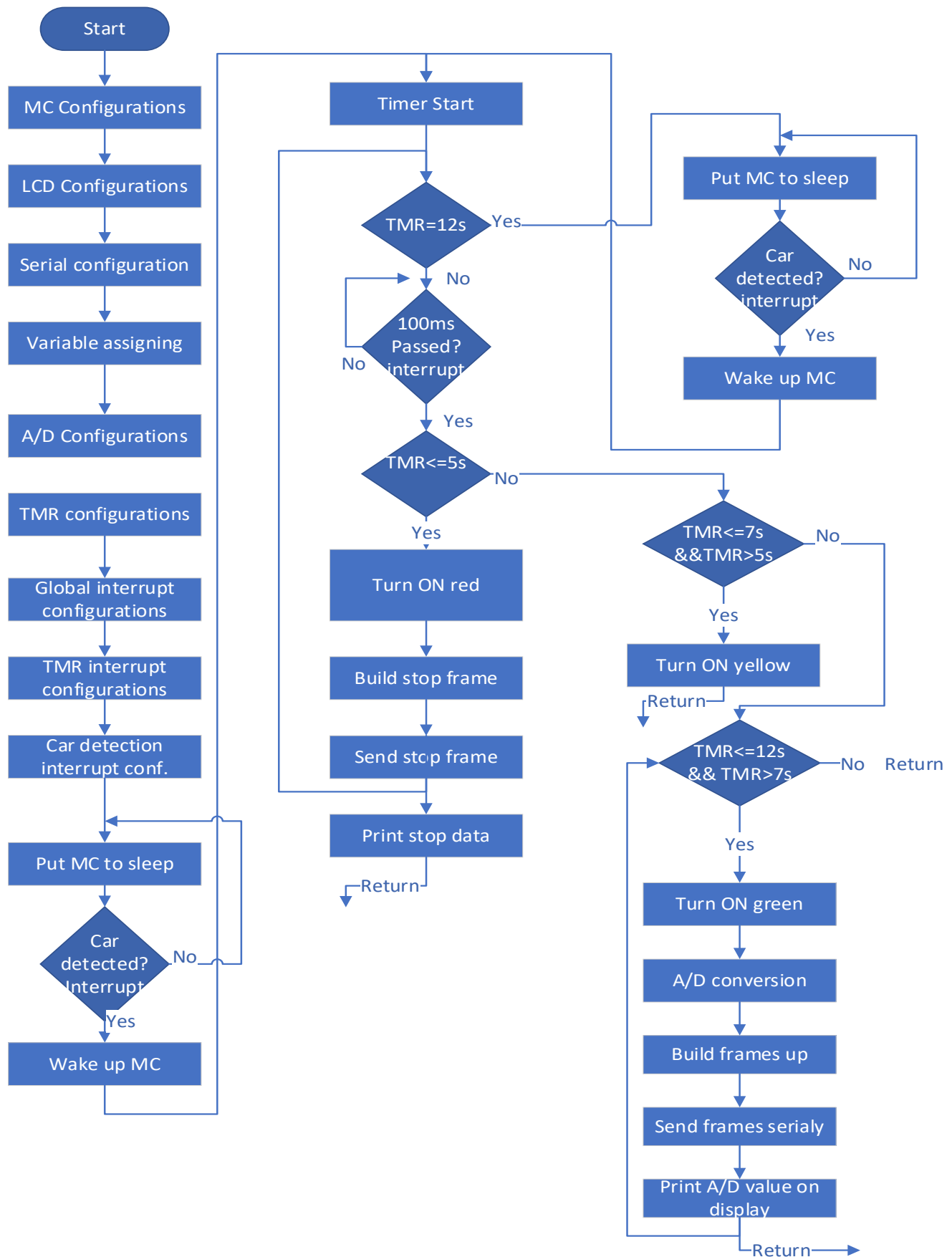


### ❖ Speed Monitor:

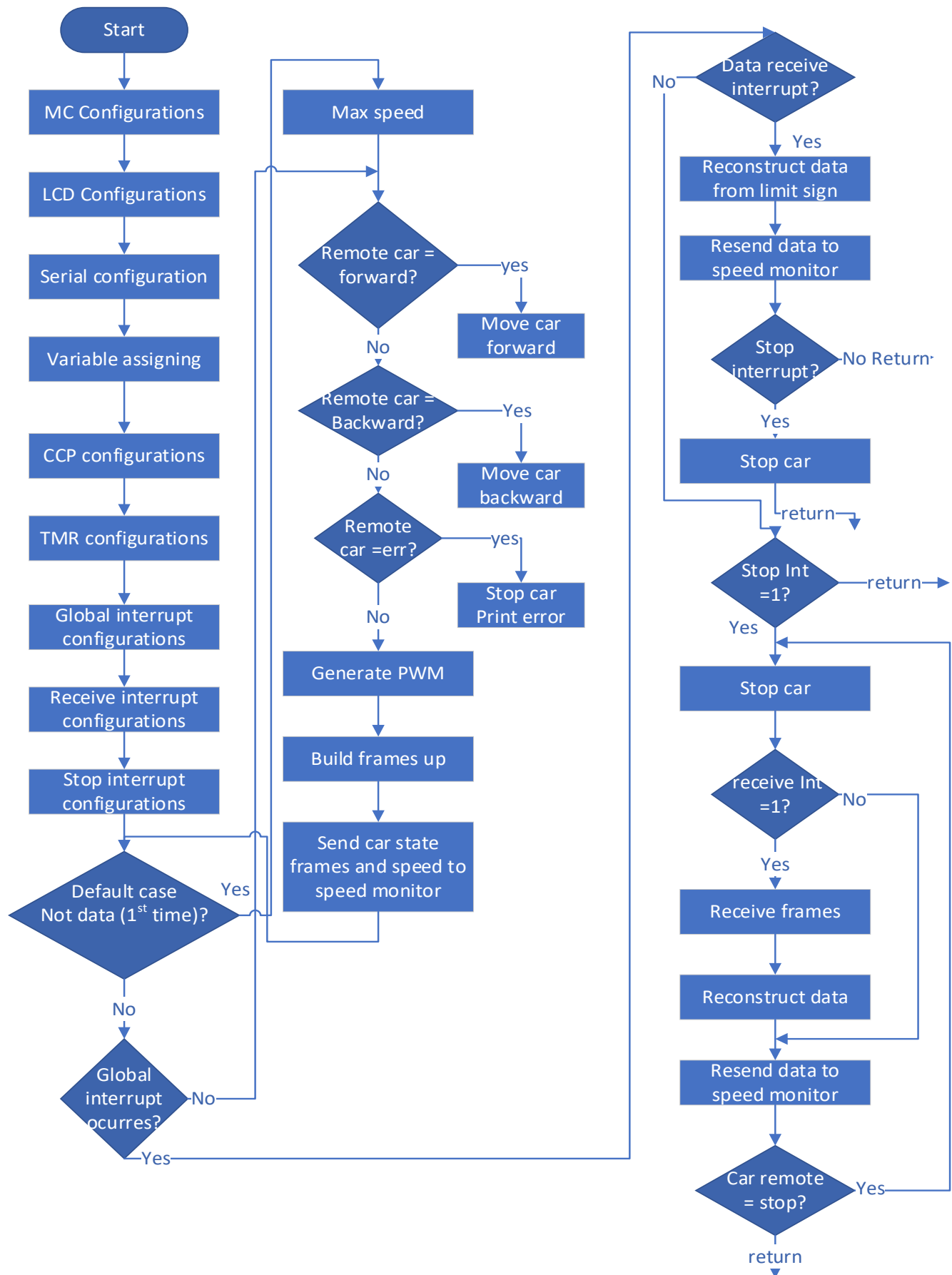




## ❖ Traffic Sign:



## ❖ Main Car:



**❖ Codes:**

➤ **Limit sign:**

```
Device=16F88
XTAL 20      ' Set Xtal Frequency

ON_HARDWARE_INTERRUPT GoTo INT ''''in case of interrupt originate to int

''''''''''analog to digital converter
initializations''''''''''''''''''''
ADIN_RES 8
Declare ADIN_TAD frc          ' RC OSC chosen
Declare ADIN_STIME 50         ' Allow 50us sample time
TRISA = %00010000            ' Configure AN4 (PORTA.4) as an input
'ADCON1 = %00000000
ANSEL = %00010000'set an4 as analog


''''''''''''''''''''declare
display''''''''''''''''''''''''''''''
Declare LCD_DTPIN PORTA.0
Declare LCD_RSPIN PORTB.6
Declare LCD_ENPIN PORTB.7


''''''''''''''''''''SERIAL USART DECLERATIONS''''''''''''''''''''
Declare HSERIAL_BAUD = 9600
RCSTA.7=1'ENABLES RC6 AS TX AND RC7 AS RX
TXSTA.5=1'ENABLES TRANSMISSION


''''''''''''''''''''VARIABLES''''''''''''''''''''''''''''''
Dim x      As Byte'used to contain value of analog to digital conversion
Dim d      As Byte'used for duty cycle
Dim C      As Byte'used as counter to that contain 120 for timer 1
Dim dataaa As Byte'used to contain the value of nibble to be send
Dim frame  As Byte'used to contain the frame that we need to send


''''''''''''''''''''INTERRUPT CONFIGURATION''''''''''''''''''''
'a) RB0 INTERRUPT CONFIGURATION
Symbol INTF = INTCON.1        ' PORTB RBO Change Interrupt Flag
Symbol INTE = INTCON.4        ' PORTB INTERRUPT ON RB0
Symbol RBPV = OPTION_REG.7    ' PortB pull-ups
Symbol INTEDG = OPTION_REG.6
Symbol GIE = INTCON.7         ' Global interrupt enable/disable
Symbol PIE = INTCON.6         ' PERIPHERAL INTERRUPTS ENABLE/DISABLE
GIE = 1                        ' ENABLE ALL INTERRUPTS
RBPV = 0                       ' ENABLE PORTB INTERNAL PULLUPS RESISTOR
INTE = 1                       ' ENABLE INTERRUPT ON PORTB
INTEDG=1                      ' EXECUTE INTERRUPT ON FALLING EDGE

'b) TMR1 CONFIGURATION
Symbol TMR1ON=T1CON.0         'timer one enable bit
Symbol T1PS0 =T1CON.4         'timer one prescalers bit
Symbol T1PS1 =T1CON.5
Symbol TM1IF= PIR1.0          'timer one interrupt flag bit
Symbol TM1IE =PIE1.0          'timer one interrupt inable bit
```

```

PIE=1                                'ENABLE PERIPHIRAL INTERRUPTS
TMR1ON=1                             'ENABLE TMR1
TM1IE=1                             'ENABLE INTERRUPT ON TMR1
T1PS0=1                             'SET THE PS0 AND PS1 TO 1 1 MEANS 8 MAX
PRESCALER FOR TMR1                    '
T1PS1=1
TM1IF=0                             'CLEAR TMR1IF
C=10
'PORTS INTIALISATION

Input  PORTA.0
Output PORTB.3
PORTB.3=0
Sleep
PORTB.3=1
Print Cls

LOOP:
If C=0 Then      'Means when timer finished
C=10
Print Cls
Print At 1,1 , "L= ",Dec x , "      "
d=((x*100)/255)
Print At 1,13,"D:",BIN frame.7
Print At 2,1 , "DU= ",Dec d," "
Print At 2,11 , "T=:",Dec C," "
PORTB.3=0
Sleep
PORTB.3=1
EndIf
GoTo LOOP

INT:
If TM1IF = 1 Then GoTo T1_INT
If INTF =1      Then GoTo RB0_INT

RB0_INT:
Print Cls
TMR1H=$0B
TMR1L=$DC
INTF=0
Context Restore

T1_INT:
x = ADIn 4      'green data
d=((x*100)/255)
dataa = d
GoSub AND15 ;data and %00001111
frame =64'%01000000
frame =frame+ dataa
HRSOut frame
dataa = d
GoSub AND240 ;data and %11110000
dataa=dataa / 16 ;swap data from HIGHER nibble to LOWER
frame =80'%01010000
frame = frame + dataa
HRSOut frame
Print At 1,1 , "L= ",Dec x , "      "
Print At 1,13,"D: ",BIN frame.7
Print At 2,1 , "DU = ",Dec d," "
Print At 2,11 , "T=:",Dec C," "

```

```
Dec C
TMR1H=$0B
TMR1L=$DC
TM1IF=0
Context Restore
```

```
AND15:
ClearBit dataa,7
ClearBit dataa,6
ClearBit dataa,5
ClearBit dataa,4
Return
```

```
AND240:
ClearBit dataa,0
ClearBit dataa,1
ClearBit dataa,2
ClearBit dataa,3
Return
```

```
End
```

## ➤ Speed Monitor:

```
Device =16F88
Declare XTAL 20
ALL_DIGITAL TRUE

'LCD DECLERATIONS

Declare LCD_DTPIN PORTA.0
Declare LCD_RSPIN PORTB.6
Declare LCD_ENPIN PORTB.7

'SERIAL DECLERATIONS
HSERIAL_CLEAR = On ' clear the buffer before receiving
HSERIAL_BAUD = 9600
Symbol OERR = RCSTA.1 ' Overrun Error
Symbol FERR = RCSTA.2 ' Framing Error
Symbol CREN = RCSTA.4 ' Continuous Receive Enable
Symbol SREN = RCSTA.5 ' Single Receive Enable
Symbol SPEN = RCSTA.7 ' Serial Port Enable
SPEN=1
CREN=1

'VARIABLES
Dim x As Byte'received data
Dim address As Byte
Dim dataa As Byte
Dim state As Byte
Dim oldstate As Byte
Dim S As Byte
Dim limit As Byte
Dim oldlimit As Byte
Dim nibble0 As Byte
Dim nibble1 As Byte
Dim n0 As Bit
Dim n1 As Bit

'CLEARING PORTS AND REGISTERS

Print Cls

'main program

Loop:

x = HRSin 'Receive a byte serially into x

'If OERR =1 Or FERR =1 Then GoTo Loop

address = x ;ADDRESS and 240
ClearBit address,0
ClearBit address,1
ClearBit address,2
ClearBit address,3
```

```

dataa = x           ;data and 15
ClearBit dataa,7
ClearBit dataa,6
ClearBit dataa,5
ClearBit dataa,4

Select address

Case 128    '10000000

n0=1

nibble0=dataa

Case 144    '10010000

n1=1

nibble1=dataa *16

Case 160    '10100000 remote address

state=x

If state = 161 Then '10100001 Stop

Print At 2,16 , "S"

S=0

Print At 2,1, "DU:",Dec S , "%  "

ElseIf state=162 Then    '%10100010 forward

Print At 2,16, "F"

Print At 2,1, "DU:",Dec limit, "%  "

ElseIf state=163 Then    '%10100011 backward

Print At 2,16, "B"

Print At 2,1, "DU:",Dec limit, "%  "

EndIf

Case Else

GoTo Loop 'case not desired frame

End Select

If n0=1 And n1=1 Then

n0=0

n1=0

limit =nibble0 + nibble1

oldlimit=limit

```

```
Print At 1,1," L:" ,Dec limit,"%  "
EndIf
Print At 1,14,"D:",BIN x.7
GoTo Loop
End
```



➤ **Traffic Sign:**

Device=16F88

```
XTAL 20      ' Set Xtal Frequency
```

```
ON_HARDWARE_INTERRUPT GoTo INT ''''in case of interrupt originate to int
```

```
''''''''''''''''''''analog to digital converter initializations''''''''''''''''''''
```

ADIN RES 8

Declare **ADIN TAD** frc ' RC OSC chosen

**Declare** **ADIN** **STIME** 50' Allow 50us sample time

```
TRISA = %00010000      ' Configure AN4 (PORTA.4) as an input
```

```
ANSEL = %00010000 'set an4 as analog
```

```
''''''''''''''''''''''''''''''''declare display''''''''''''''''''''''''''''''''
```

```
Declare LCD DTPIN PORTA.0
```

Declare **LCD** **RSPIN** **PORTB.6**

Declare **LCD ENPIN** **PORTB.7**

```
'''SERIAL USART DECLERATIONS'''
```

Declare **HSERIAL BAUD** = 9600

```
RCSTA.7=1 'ENABLES RC6 AS TX AND RC7 AS RX
```

```
TXSTA.5=1 'ENABLES TRANSMISSION
```

```
'''VARIABLES'''
```

Dim x As Byte'used to contain the value of analog to digital conversion

```
Dim d As Byte 'used for duty cycle
```

```
Dim C As Byte 'used as counter to that contain 120 for timer 1
```

```
Dim dataa As Byte'used to contain the value of nibble to be send
```

```
Dim frame As Byte'used to contain the frame that we need to send
```

[illegible]

'a) RB0 INTERRUPT CONFIGURATION

```
Symbol INTF = INTCON.1      ' PORTB RBO Change Interrupt Flag
```

```
Symbol INTE = INTCON.4      ' PORTB INTERRUPT ON RB0
```

**Symbol** RBPUP = OPTION REG.7    ' PortB pull-ups

**Symbol** INTEDG = **OPTION** REG.6

**Symbol** GIE = INTCON.7      ' Global interrupt enable/disable

```
Symbol PIE = INTCON.6      ' PERIPHERAL INTERRUPTS ENABLE/DISABLE
```

```

GIE = 1                                ' ENABLE ALL INTERRUPTS

RBPV =0                                ' ENABLE PORTB INTERNAL PULLUPS RESISTOR

INTE =1                                ' ENABLE INTERRUPT ON PORTB

INTEDG=1                                ' EXECUTE INTERRUPT ON RAISING EDGE


'b) TMR1 CONFIGURATION

Symbol TMR1ON=T1CON.0                  'timer one enable bit

Symbol T1PS0 =T1CON.4                  'timer one prescalers bit

Symbol T1PS1 =T1CON.5

Symbol TM1IF = PIR1.0                  'timer one interrupt flag bit

Symbol TM1IE =PIE1.0                  'timer one interrupt inable bit

PIE=1      'ENABLE PERIPHIRAL INTERRUPTS

TMR1ON=1    'ENABLE TMR1

TM1IE=1     'ENABLE INTERRUPT ON TMR1

T1PS0=1     'SET THE PS0 AND PS1 TO 1 1 MEANS 8 MAX PRESCALER FOR TMR1

T1PS1=1

TM1IF=0     'CLEAR TMR1IF

C=120 'contains c=cr+cy+cg=50+20+50=120 and timer 1 over flow on 0.1 s for fosc
=20 MH'

////////////////////////////////////////'PORTS INTIALISATION'////////////////////////////////////////

Input PORTA.0

Output PORTB.3 'yellow light

Output PORTB.1 'red light

Output PORTB.4 'green light

Sleep 'wait for interrupt on RB0 this mode can reduce power consumption

Print Cls

```

```

'////////////////////////////////main program////////////////////////////////'

LOOP:

If C=0 Then      'Means when timer finished

PORTB.4=0

PORTB.1=0

PORTB.3=0

C=120

Print Cls

Print At 1,1 , "L= ",Dec x , "    "

d= ((x*100)/255)

Print At 1,13,"D:",BIN frame.7

Print At 2,1 , "DU= ",Dec d, "  "

Print At 2,11 , "T=:",Dec C, "  "

Sleep

EndIf

GoTo LOOP

'////////////////////////////////interrupt section////////////////////////////////'

INT:

If TM1IF = 1 Then GoTo T1_INT

If INTF =1      Then GoTo RB0_INT

RB0_INT:

Print Cls

TMR1H=$0B'timer 100 ms with Fosc =20MHz

TMR1L=$DC

INTF=0

Context Restore

T1_INT:

If C>70 Then 'from 120 to 70 for red cr=50 so Tred= 50*0.1 = 5s

PORTB.3=0

PORTB.4=0

```

```

PORTB.1=1 'turn on red light

x=0      'red data
d=((x*100)/255)
dataa = d

GoSub AND15 ;data and %00001111

frame =96'%01100000

frame =frame+ dataa

HRSOut frame

dataa = d

GoSub AND240 ;data and %11110000

dataa=dataa / 16 ;swap data from HIGHER nibble to LOWER

frame =112'%01110000

frame = frame + dataa

HRSOut frame

ElseIf C<=70 And C>50 Then 'from 70 to 50 for yellow cy=20 so Ty= 20*0.1
= 2s

PORTB.4=0

PORTB.1=0

PORTB.3=1'turn on yellow light

ElseIf C>0 Then 'from 50 to 0 for green cg=50 so Tg= 50*0.1 = 5s

PORTB.1=0

PORTB.3=0

PORTB.4=1 'turn on green light

x = ADIn 4      'green data
d=((x*100)/255)
dataa = d

GoSub AND15 ;data and %00001111

frame =64'%01000000

frame =frame+ dataa

HRSOut frame

dataa = d

```

```

GoSub AND240;data and %11110000

dataa=dataa / 16 ;swap data from HIGHER nibble to LOWER

frame =80'%01010000

frame = frame + dataa

HRSOut frame

EndIf

Print At 1,1 ,"L= ",Dec x ,"    ""

d=((x*100)/255)

Print At 1,13,"D: ",BIN frame.7

Print At 2,1 ,"DU = ",Dec d," ""

Print At 2,11 ,"T=: ",Dec C," ""

Dec C

TMR1H=$0B'timer 100 ms with Fosc =20MHz and for 1.5 second timer 100ms *
15

TMR1L=$DC

TM1IF=0

Context Restore

AND15:
ClearBit dataa,7
ClearBit dataa,6
ClearBit dataa,5
ClearBit dataa,4
Return

AND240:
ClearBit dataa,0
ClearBit dataa,1
ClearBit dataa,2
ClearBit dataa,3
Return

End

```

## ➤ Main Car:

```
Device =16F877A
Declare XTAL 20
ALL_DIGITAL TRUE
ON_HARDWARE_INTERRUPT GoTo I_NT 'in case of interrupt originate to I_NT

'DISPLAY DECLERATIONS
Output PORTD
Declare LCD_DTPIN PORTD.4
Declare LCD_ENPIN PORTD.0
Declare LCD_RSPIN PORTD.1

'VARIABLES
'a) for Rx
Dim x As Byte
Dim address As Byte
Dim dataa As Byte
Dim speedlimit As Byte
Dim oldspeedl As Byte
Dim nibble1 As Byte
Dim nibble2 As Byte
Dim n4 As Bit
Dim n5 As Bit
Dim n6 As Bit
Dim n7 As Bit
Dim fr As Bit
Dim fr2 As Bit
Dim RCF As Bit
Dim CLRD As Bit

'b) for transmitter

Dim state As Byte
Dim x1 As Byte
Dim sdataa As Byte
Dim sframe As Byte

'c) for motor control

Dim d As Byte
Dim duty As Byte
Dim conv As Byte
Dim ferr As Bit

'SERIAL USART DECLERATIONS
HSERIAL_BAUD = 9600
HSERIAL_CLEAR = On 'clear the buffer before receiving
Symbol CREN = RCSTA.4 'Continuous Receive Enable
Symbol SPEN = RCSTA.7 'Serial Port Enable
Symbol TXEN = TXSTA.5 'Transmit Enable
Symbol FERRR =RCSTA.2 'used to detect if buad is different
Symbol OERR =RCSTA.1 'used to detect if there is an interfernce
SPEN=1 'enable serial on rc6 and rc7
CREN=1 'enable receiver
TXEN=1 'enable transmitter
```

```

'CCP DECELARATION

Symbol c1 = PORTC.0 '/f
Symbol c2 = PORTC.3 '/b
Output PORTC.4 'used on first enable pin of the driver
Output PORTC.5 'used on the second enable pin on the driver
CCP1_PIN = PORTC.2 'used to generate pwm signal on the driver

'INTERRUPT DECLARATION

Symbol PEIE = INTCON.6 'Peripheral Interrupt Enable
Symbol GIE = INTCON.7 'Global Interrupt Enable
Symbol PBPUP=OPTION_REG.7 'portb pull up
PBPUP=0 'enable internal pullup resistor on portB
GIE = 1 'enable the global interrupt
PEIE = 1 'enable the periphiral interrupts

'a)portb interrupt

Symbol INTF = INTCON.1 ' RB0 External Interrupt Flag
Symbol INTE = INTCON.4 ' RB0 External Interrupt Enable
INTE=1 'enalbe rb0 interrupt
OPTION_REG.6=0 'on rissing edge 1

'b)receive interrupt

Symbol RCIE= PIE1.5 'interrupt enable of receiver
Symbol RCIF= PIR1.5 'interrupt flag of receiver
RCIE=1 'enable interrupt on receiver

'CLEARING PORTS AND REGISTERS

n4 =0
n5 =0
n6 =0
n7 =0
RCF=0
fr =0
Print Cls

'main program

If RCF = 0 Then

'TO GIVE MAX SPEED IN CASE OF DEFUALT CASE

speedlimit=100
oldspeedl=100
Print At 1,1,"MAX SPEED: 100  "
CLRDR=1
INTF=1
EndIf

remote: 'VEHICLH REMOTE CAR SECTION TO DETECT MOVING STATE
Input PORTC.0
Input PORTC.3
PORTC.0=0
PORTC.3=0
If c1=1 And c2=0 Then 'forward case
Print At 2,16,"F"

```

```

PORTC.4=1 'forward pin
PORTC.5=0 'backward pin
GoTo motor
ElseIf c1=0 And c2=1 Then 'backward case
Print At 2,16,"B"
PORTC.4=0 'forward pin
PORTC.5=1 'backward pin
GoTo motor
ElseIf c1=1 And c2 =1 Then 'error case
Print At 1,1 ,"err Remote"
PORTC.4=0 'forward pin
PORTC.5=0 'backward pin
ferr=1
EndIf
GoTo remote

motor: 'MOTOR CONTROLS USING PWM

If ferr=1 Then
Print Cls
ferr=0
Print At 1,1,"LV: " ,Dec speedlimit," "
Print At 1,14,"D:",BIN x.7
EndIf
conv=((255*speedlimit)/(100))
HPWM 1,conv,2000
Print At 2,1,"d:",Dec speedlimit," "
x1= speedlimit ;data1(speed1)
sdataa = x1
;data and %00001111
ClearBit sdataa,7
ClearBit sdataa,6
ClearBit sdataa,5
ClearBit sdataa,4
sframe =128'%10000000
sframe =sframe+ sdataa
HRSOut sframe
sdataa = x1
;data and %11110000
ClearBit sdataa,0
ClearBit sdataa,1
ClearBit sdataa,2
ClearBit sdataa,3
sdataa=sdataa / 16 ;swap data from HIGHER nibble to LOWER
sframe =144'%10010000
sframe = sframe + sdataa
HRSOut sframe
If c1=0 And c2=0 Then
state =161'%10100001 'Stop
ElseIf c1=1 And c2=0 Then
state =162'%10100010 'forward
ElseIf c1=0 And c2=1 Then
state =163'%10100011 'forward
EndIf
HRSOut state
GoTo remote

I_NT: INTERRUPT SECTION

If RCIF = 1 Then GoTo R_X
If INTF=1 Then GoTo st_op

```



## Context Restore

```
R_X: 'INTERRUPT WHILE RECEIVING
x = HRSin 'Receive a byte serially into x
fr=0
If c1=0 And c2=0 Then
INTF=0
Print At 2,16,"S"
PORTC.4=0 'forward pin
PORTC.5=0 'backward pin
PORTC.4=0 'forward pin
PORTC.5=0 'backward pin
EndIf
If ferr=1 Or CLRD=1 Then
Print Cls
ferr=0
CLRD=0
Print At 2,1,"d:",Dec duty," "
EndIf
RCF=1

'If FERRR=1 Or OERR=1 Then ERRR=1 : GoTo SKIPP 'used to check if any
error in received data optional

address = x
ClearBit address,0
ClearBit address,1
ClearBit address,2
ClearBit address,3
dataa = x ;data and 15
ClearBit dataa,7
ClearBit dataa,6
ClearBit dataa,5
ClearBit dataa,4
Select address
Case 64 '01000000
n4=1
fr=0
nibble1=dataa
Case 80 '01010000
n5=1
fr=0
nibble2=dataa *16
Case 96 '01010000
n6=1
nibble1=dataa
fr2=0
Case 112 '01110000
n7=1
fr2=0
nibble2=dataa *16
Case Else
If address.7=1 Then ' address = 128 Or address =144 Or address =160 Or
address = 176 Then
speedlimit = oldspeedl
Context Restore
EndIf
End Select
If n4=1 And n5=1 Then
fr=1
speedlimit =nibble1 + nibble2
```

```

n4=0
n5=0
oldspeedl=speedlimit
Print At 1,1,"LV: " ,Dec speedlimit,"    "
EndIf
If n6 =1 And n7=1 Then
fr2=1
speedlimit =nibble1 + nibble2
n6=0
n7=0
oldspeedl=speedlimit
Print At 1,1,"LV: " ,Dec speedlimit,"    "
EndIf
Print At 1,14,"D:",BIN x.7
If fr=1 Or fr2=1 Then
If c1=1 And c2=0 Then
Print At 2,16,"F"
PORTC.4=1 'forward pin
PORTC.5=0 'backward pin
GoTo motor1
ElseIf c1=0 And c2=1 Then
Print At 2,16,"B"
PORTC.4=0 'forward pin
PORTC.5=1 'backward pin
GoTo motor1
ElseIf c1=0 And c2=0 Then
Print At 2,16,"S"
PORTC.4=0 'forward pin
PORTC.5=0 'backward pin
EndIf
motor1:
conv=((255*speedlimit)/(100))
HPWM 1,conv,2000
Print At 2,1,"d:",Dec speedlimit,"    "
EndIf

'SKIPP:
'If ERRR=1 Then speedlimit = oldspeedl:ERRR=0 'optional

x1= speedlimit ;data1(speedl)
sdataa = x1

;data and %00001111

ClearBit sdataa,7
ClearBit sdataa,6
ClearBit sdataa,5
ClearBit sdataa,4
sframe =128 '%10000000
sframe =sframe+ sdataa
HRSOut sframe
sdataa = x1

';data and %11110000

ClearBit sdataa,0
ClearBit sdataa,1
ClearBit sdataa,2
ClearBit sdataa,3
sdataa=sdataa / 16 ;swap data from HIGHER nibble to LOWER
sframe =144 '%10010000

```

```

sframe = sframe + sdataa
HRSOut sframe
If c1=0 And c2=0 Then
state =161'%10100001      'Stop
ElseIf c1=1 And c2=0 Then
state =162'%10100010      'forward
ElseIf c1=0 And c2=1 Then
state =163'%10100011      'forward
EndIf
HRSOut state
RCIF=0
Context Restore
st_op: 'INTERRUPT USING RB0'
c1=0
c2=0
Print Cls'clear display each time interrupt occurs
Wait: 'wait till the c1 and c2 are different from 0-
If RCIF =1 Then x=HRSin
If CLRDR=1 Then
Print At 1,1,"MAX SPEED: 100 "
Else
Print At 1,1,"LV: " ,Dec speedlimit,"    "
Print At 1,14,"D:",BIN x.7
EndIf
If c1=0 And c2=0 Then 'goto stop in case controller gives 0v on pin 3
and pin 4
duty =0
'Print At 2,1 ,"d:",Dec duty,"    "
Print At 2,16,"S"
PORTC.4=0'forward pin
PORTC.5=0'backward pin
PORTC.4=0
PORTC.5=0
Print At 2,1,"d:",Dec duty,"    "
HPWM 1,0,2000
HPWM 1,0,2000
If RCIF =1 Then
'If FERRR=1 Or OERR=1 Then ERRR=1 : GoTo SKIP2
RCF=1
address = x ;ADDRESS and 240
ClearBit address,0
ClearBit address,1
ClearBit address,2
ClearBit address,3
dataa = x ;data and 15
ClearBit dataa,7
ClearBit dataa,6
ClearBit dataa,5
ClearBit dataa,4
Select address
Case 64 '01000000
n4=1
fr=0
nibble1=dataa
Case 80 '01010000
n5=1
fr=0
nibble2=dataa *16
Case 96 '01010000
n6=1
nibble1=dataa

```

```

Case 112      '01110000
n7=1
nibble2=dataaa *16
Case Else
If address.7=1 Then
speedlimit = oldspeedl
Context Restore
EndIf
End Select
If n4=1 And n5=1 Then
fr=1
speedlimit =nibble1 + nibble2
n4=0
n5=0
oldspeedl=speedlimit
Print At 1,1,"LV: " ,Dec speedlimit,"    "
EndIf
If n6 =1 And n7=1 Then
speedlimit =nibble1 + nibble2
n6=0
n7=0
oldspeedl=speedlimit
Print At 1,1,"LV: " ,Dec speedlimit,"    "
EndIf
Print At 1,14,"D:",BIN x.7
'SKIPP2:
RCIF = 0
EndIf
x1= speedlimit ;data1(speedl)
sdataa = x1
;data and %00001111
ClearBit sdataa,7
ClearBit sdataa,6
ClearBit sdataa,5
ClearBit sdataa,4
sframe =128'%10000000
sframe =sframe+ sdataa
HRSOut sframe
sdataa = x1
;data and %11110000
ClearBit sdataa,0
ClearBit sdataa,1
ClearBit sdataa,2
ClearBit sdataa,3
sdataa=sdataa / 16 ;swap data from HIGHER nibble to LOWER
sframe =144'%10010000
sframe = sframe + sdataa
HRSOut sframe
state =161'%10100001      'Stop
HRSOut state
GoTo Wait
EndIf
INTF=0
Context Restore
End

```

## REFERENCES

### ❖ Internet:

- **PIC MICROCONTROLLERS:**

- **P16F877A:**

DATASHEET LINK:

<https://www.theengineeringprojects.com/2017/06/pic16f877a.html>

- **P16F876A:**

DATASHEET LINK:

<http://www.datasheetcafe.com/pic16f876a-datasheet-microcontroller-microchip/>

- **P16F88:**

DATASHEET LINK:

<https://reviseomatic.org/rOmV4/rOmV4/page/136/PIC16F88>

- **Driver:**

DATASHEET LINK:

<https://forum.arduino.cc/index.php?action=dlattach;topic=366727.0;attach=147606>

- **Display:**

DATASHEET LINK:

<https://www.engineersgarage.com/electronic-components/16x2-lcd-module-datasheet>

- **HC-11(Transmitter and Receiver):**

DATASHEET LINK:

<https://www.elecrow.com/download/HC-11.pdf>

<https://www.electfreaks.com/store/blog/post/cc1101-433mhz-wireless-rf-transceiver-module-v1-9-user-guide.html>

- **Crystal Oscillator:**

<http://www.kynixsemiconductor.com/News/28.html>

[https://en.wikipedia.org/wiki/Crystal\\_oscillator](https://en.wikipedia.org/wiki/Crystal_oscillator)