EMBEDDED SYSTEMS MINI-PROJECT

Semester V

BTECH IN INFORMATION TECHNOLOGY:

IT Section B1 – Group-1

SUBMITTED BY:

NAME	REGISTRATION NUMBER	ROLL NUMBER
Shivakalyan N	200911022	05
Ganesh s Nayak	200911008	02
Giri teja	200911001	01

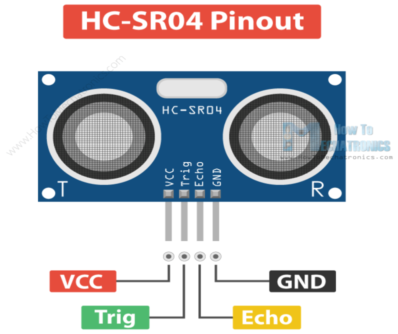
MANIPAL INSTITUTE OF TECHNOLOGY
MANIPAL
A Constituent Institution of Manipal University

TABLE OF CONTENTS

Problem statement: Interfacing the Doppler sensor - HC SR04 to LPC1768 microcontroller and displaying the distance of an object on a 7-segment display.

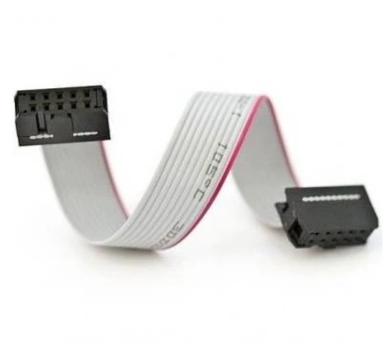
Hardware Requirements:

HC SR04 ultrasonic sensor : The HC-SR04 ultrasonic sensor uses sonar to determine the distance to an object. This sensor reads from 2cm to 400cm (0.8inch to 157inch) with an accuracy of 0.3cm (0.1inches), which is good for most hobbyist projects. In addition, this particular module comes with ultrasonic transmitter and receiver modules.



2. LPC 1768 microcontroller kit : It is a powerful 32-bit ARM Cortex-M3 processor running upto 100 MHz with 512 KB flash and 32 KB RAM, which makes it far more capable than popular 8-bit prototyping alternatives.

3. FRC Cables : It is a powerful 32-bit ARM Cortex-M3 processor running upto 100 MHz with 512 KB flash and 32 KB RAM, which makes it far more capable than popular 8-bit prototyping alternatives.



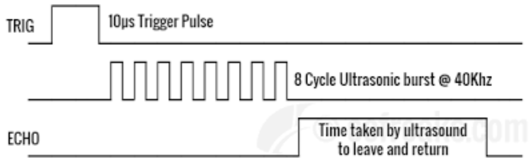
4. **Jumper Cables :** These cables are used make sophisticated connections between the kit and the sensor. Four female to female jumper cables are usedin the setup.

Software Requirements:

1. Keil microvision4 simulator
2. Flash Magic

Methodology:

The HC-SR04 Ultrasonic Distance/Ranging Sensor uses ultrasound to measure distance from an object ahead of the sensor. HCSR-04 module uses 40Khz ultrasound to measure distance between itself and any object ahead of it with a sensing range of 2 centimeters to 4 meters A short pulse of 10us is applied to the Trigger pin to start distance measurement. After receiving a trigger pulse, the HC-SR04 Module sends a burst of 8 ultrasonic pulses at 40Khz. It will then output high on Echo for the amount of time taken for the sound waves to reach back. This time obtained is then used to calculate the distance.


TRIG
10µs Trigger Pulse
8 Cycle Ultrasonic burst @ 40Khz
ECHO
Time taken by ultrasound to leave and return

We know speed of sound in air,

Vs = 343 m/s = 0.0343 cm/us

We also know the time it took for sound waves to emit and echo back, lets call this time taken T. Now, by using basic distance formula we can find the distance as:

Distance Travelled = Speed x Time taken DT = 343 m/s x T seconds

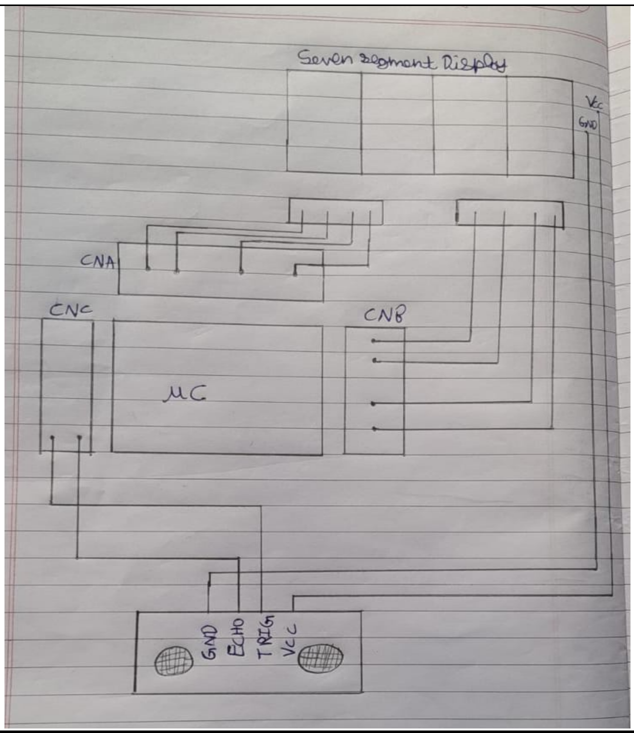
Now, since we will be measuring ECHO ON-Time in microseconds and also to get distance in centimetres we can change the units as follows:

DT in cm = 0.0343 cm/us x T us

After this we divide the computed value by 2 since the waves have travelled double distance

D = DT / 2 = 0.0343 x T / 2 cm

Detailed Diagram along with pin details



CODE:

```
// Trigger: p0.15(output)
// echo : p0.16(input)

#include <lpc17xx.h>
#define PRESCALE (25-1)
#include<stdlib.h>
#include<stdio.h>


char sevenseg[] = {0x3f,0x6,0x5b,0x4f,0x66,0x6d,0x7d,0x7,0x7f,0x6f};


int round_val(float num){
        float val = num - abs(num);
    if(val <= 0.5)
            return (int)num;
        else
            return (int)num + 1;
}
void initTimer(void);
void startTimer(void);
unsigned int stopTimer(void);
void delayUS(unsigned int microseconds);
void delayMS(unsigned int milliseconds);


void initTimer(void)
{
    LPC_TIM0->CTCR = 0x0;
    LPC_TIM0->PR = PRESCALE; //Increment TC at every 24999+1 clock cycles
    LPC_TIM0->TCR = 0x02; //Reset Timer
}


void startTimer(void)
{
    LPC_TIM0->TCR = 0x02; //Reset Timer
    LPC_TIM0->TCR = 0x01; //Enable timer
}


unsigned int stopTimer(void)
{
    LPC_TIM0->TCR = 0x00; //Disable timer
```

```
   return LPC_TIM0->TC;  //timer counter register
}


void delayUS(unsigned int microseconds) //Using Timer0
{
   LPC_TIM0->TCR = 0x02; //Reset Timer
LPC_TIM0->TCR = 0x01; //Enable timer
   while(LPC_TIM0->TC < microseconds); //wait until timer counter reaches the desired
delay
 LPC_TIM0->TCR = 0x00; //Disable timer
}


void delayMS(unsigned int milliseconds)
{
   delayUS(milliseconds * 1000);
}


void delay_trigger(){  //at start 10 microsec high is set
   LPC_GPIO0->FIOSET = (0x1<<15);
   delayUS(10);
   LPC_GPIO0->FIOCLR = (0x1<<15);
}


int echo_ans(){
   float pulse_time = 0,distance=0;
 while((LPC_GPIO0->FIOPIN & (0x1<<16)) == 0x0);     //Wait till echo is low
   startTimer();
            //Initialize the echo timer
   while((LPC_GPIO0->FIOPIN & (0x1<<16)) == 0x1<<16); //Wait till echo is high
   pulse_time = stopTimer();
   //Get count of echo timer
distance = (0.0343*pulse_time)/2;
return round_val(distance);
}


void display(int number,int displayTimeSeconds){
   unsigned int j,i,timeUS = 1000000*displayTimeSeconds,temp;
   startTimer();
   while(LPC_TIM0->TC <timeUS){
      temp = number;
      for(j=0;j<4;j++){
            LPC_GPIO1->FIOPIN = j<<23;
```

```
         LPC_GPIO0->FIOPIN = sevenseg[(temp%10)]<<4;    //extracting each digit from
temp and displaying
      for(i=0;i<1000;i++);
         temp /= 10;
       }
  }


  stopTimer();
  LPC_GPIO1->FIOPIN = 00<<23;
  LPC_GPIO0->FIOPIN = 0xF9<<4; //Display a dot
}


int main(void){
  int distance;
  SystemInit();
  SystemCoreClockUpdate();




 LPC_PINCON->PINSEL0 = 0x0; //SET TO GPIO
  LPC_PINCON->PINSEL1 = 0x0;    //SET TO GPIO (For the echo pin)
  LPC_GPIO0->FIODIR = 0x0FFF0;  //p0.4 to p0.15
  //Decoder
  LPC_PINCON->PINSEL3 = 0x0;        //Decoder    GPIO config
  LPC_GPIO1->FIODIR = 0xF<<23; //Decoder output config




 //Timer setup
  initTimer();
  while(1){
delay_trigger();  //10micro sec high
distance = echo_ans();
    display(distance,1);
  }
}
```

SNAPSHOTS OF OUTPUT: