

TI Analog design contest 2009

# Pothole Detection

Project Report

2009-  
2010

By:

Anoosh G

Raghunandan Srinivasan

Sundar Aditya

Under the  
Guidance of:

Dr. Nagendra  
Krishnapura

IIT MADRAS

## Table Of Contents

<a href="#">1 Abstract.....</a>	<a href="#">4</a>
<a href="#">2 Motivation.....</a>	<a href="#">4</a>
<a href="#">3 Introduction.....</a>	<a href="#">4</a>
<a href="#">4 Design Requirements.....</a>	<a href="#">5</a>
<a href="#">5 System Block Diagram.....</a>	<a href="#">6</a>
<a href="#">5.1 Hardware Blocks.....</a>	<a href="#">6</a>
<a href="#">5.2 Software Blocks.....</a>	<a href="#">6</a>
<a href="#">6 Hardware and Software Used.....</a>	<a href="#">7</a>
<a href="#">7 Sensor Description.....</a>	<a href="#">7</a>
<a href="#">7.1 Introduction.....</a>	<a href="#">7</a>
<a href="#">7.2 Circuit Diagram.....</a>	<a href="#">8</a>
<a href="#">7.3 Sensor Characteristics.....</a>	<a href="#">8</a>
<a href="#">7.3 Sensor Response.....</a>	<a href="#">9</a>
<a href="#">8 Algorithm and Flowchart.....</a>	<a href="#">10</a>
<a href="#">8.1 MSP Algorithm.....</a>	<a href="#">10</a>
<a href="#">8.2 Flowcharts.....</a>	<a href="#">12</a>
<a href="#">8.2.1 Flowchart of Algo.....</a>	<a href="#">12</a>
<a href="#">8.2.2 Flowchart of System.....</a>	<a href="#">13</a>
<a href="#">9 Other Use Case Scenarios.....</a>	<a href="#">14</a>
<a href="#">10 Prototype picture.....</a>	<a href="#">15</a>
<a href="#">11 Cost of Prototype.....</a>	<a href="#">15</a>
<a href="#">12 Scope For Improvement.....</a>	<a href="#">16</a>
<a href="#">13 MSP Code.....</a>	<a href="#">16</a>

<a href="#">13.1 Source : Main.c.....</a>	<a href="#">16</a>
<a href="#">13.2 Source : init.asm.....</a>	<a href="#">19</a>
<a href="#">13.3 Source : interrupt_asm.s43.....</a>	<a href="#">19</a>

# 1 Abstract

In this report the authors describe the design of a hand-held obstacle detector (OD) which detects discontinuities in terrain and alerts users of potential hazards like open manholes, ditch, protrusions etc. which are common on Indian roads. The device is meant to be an aid for the visually challenged. It is a low cost, low power, hand held device that can be carried like a TV remote.

## 2 Motivation

It is common knowledge that Indian roads are fraught with many dangers such as open manholes, abrupt bumps, large stones etc. without adequate warning signs, making travel, both on foot as well as on an automobile very risky, especially at night. There is a need for a reliable and robust early warning system which would accurately detect abrupt discontinuities in terrain like potholes, manholes, bumps etc. which can be used by pedestrians. This report describes the design of a prototype early warning system to detect discontinuities in the terrain, a project undertaken for the TI Analog Design Contest for Indian Universities 2009.

## 3 Introduction

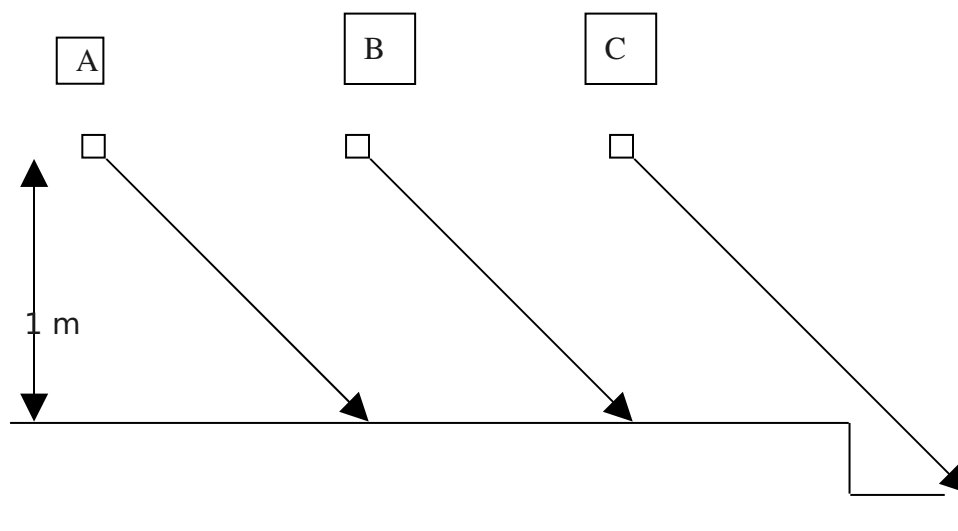
The OD works on the principle of SONAR. An ultrasonic transducer emits pulses at regular intervals, the interval being a function of the speed of travel. A receiver block listens to the echo. The echo round-trip time is a function of height (see figure below). The receiver detects abrupt changes in terrain by comparing the round-trip times. The system-level description is given in the block diagram below. The authors have designed a system specifically for pedestrian use. In the subsequent sections the design of this system is described. Other possible applications are discussed in section 6. The principle remains the same in all use-case scenarios.

## 4 Design Requirements

The nature of the terrain some distance ahead needs to be determined. Lets call this the “look ahead” distance. It is intuitive that this distance is a function of speed of travel. We need information before it is too late to act on. For pedestrian usage, this distance was determined to be 1.5 m, considering a nominal walking speed of roughly 1 m/s. The frequency of timing is also important. It should be high enough not to miss out any significant stretch of terrain. It too depends on the speed of travel. The Figure illustrates the functioning of the OD.

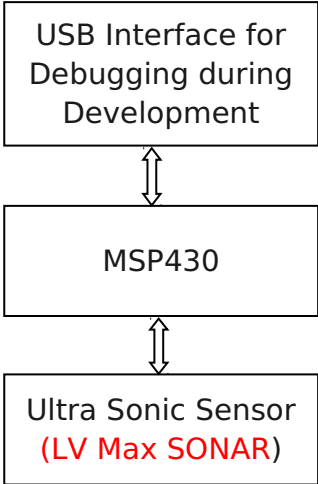
An important aspect of the system is the delay involved. The timing unit should trigger the ultrasonic sensor to emit a short pulse. The receiver has to listen to the echo and determine if there is any discontinuity before the sensor transmits another pulse. The OD is designed to be a handheld device, operated at waist level. The height at which the OD is operating is 1m worst case. Therefore the round trip time is utmost 10 ms, assuming that the acoustic wave is traveling in air (speed of sound in air = 340 m/s approx). Therefore the sensor needs to be triggered at a time period larger than 10ms.

Any outlier in round-trip time is detected as a discontinuity. Hence, an accurate outlier detection algorithm is required. Accuracy here is with respect to minimizing the number of false positives. A common source of error (false positive) is detecting a gradient as a discontinuity. This needs to be avoided. In case a discontinuity is detected, a clear warning needs to be given to the user.



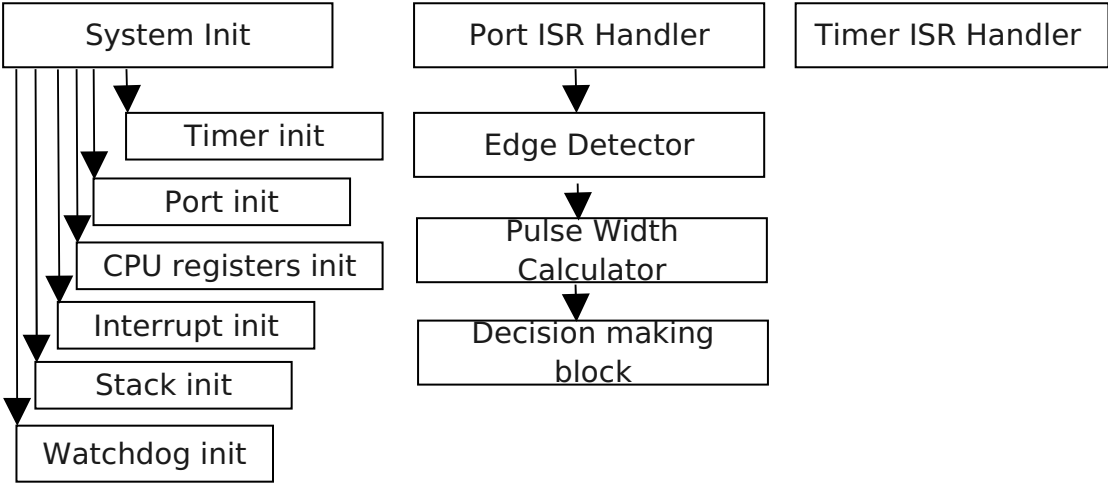
# 5 System Block Diagram

## 5.1 Hardware Blocks



## 5.2 Software Blocks

The software block here refers to the code on the MSP that decodes the Ultra Sonic Sensor Output and takes action based on the change in sensor output.



# 6 Hardware and Software Used

1. MSP430 and eZ430-RF2500 Evaluation Board. (Texas Instruments)
  - i. Interruptible i/o pins
  - ii. Timer A
  - iii. Capture Compare Register
2. IAR Systems IDE. (Texas Instruments)
3. Ultra Sonic Sensor. (LV-Max Sonar EZ1)
4. Voltage Regulators (Texas Instruments)
  - i. TPS71550
  - ii. TPS71537

# 7 Sensor Description

## 7.1 Introduction

The sensor used is LV-Max Sonar EZ1. Sonar range finder. The sensor uses Sonar waves to determine the distance at which obstacle lies.

The sensor needs a 5v supply which is provided using a voltage regulator. The o/p is in the form of a PWM output. The sensor has a calibrated sensitivity of  $147\mu\text{s}/\text{inch}$  (microseconds/inch). The PWM pin of the sensor is utilized using a pull up resistor ( $10K\Omega$ ) between Vdd(5V) and the pin. The Ton(on time) of the pulse o/p varies accordingly as the distance of the obstacle lying in front.

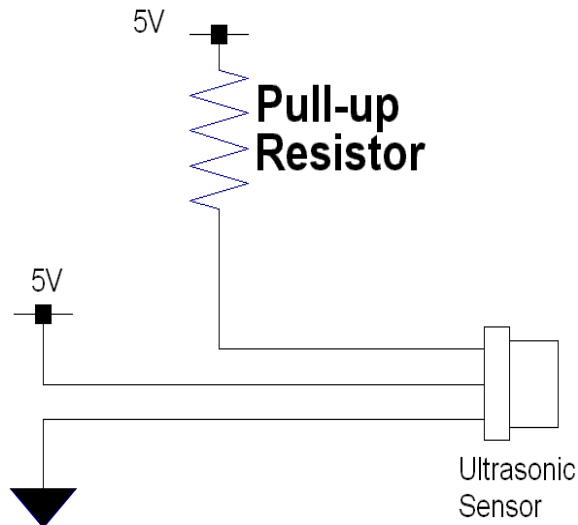
The pulse o/p of the PWM pin was checked on an oscilloscope during the initial testing of the sensor functionalities.

The outputs are shown below.



an

## 7.2 Circuit Diagram



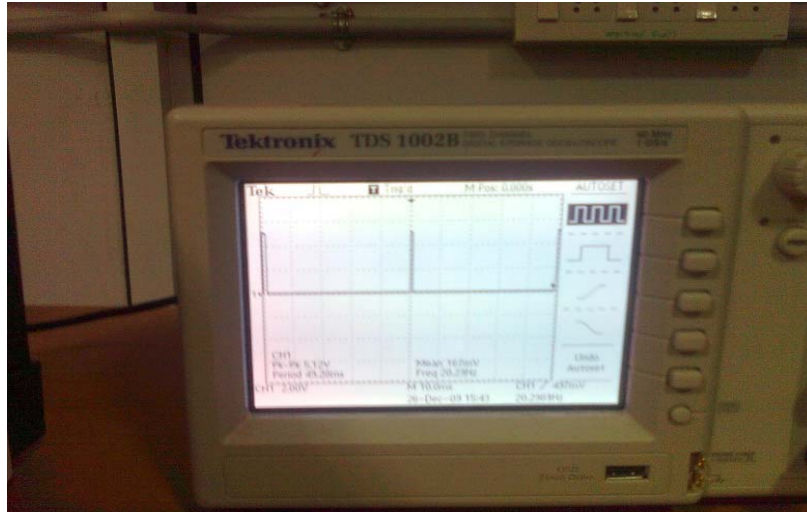
## 7.3 Sensor Characteristics

The sensor has an operational voltage range of 2.5-5V and a pulse width sensitivity of approximately  $147\mu\text{s}/\text{inch}$ . This means that for a change of 1 inch in the distance covered by the ultrasonic wave, the change in the pulse width is  $147\mu\text{s}$ . However, this is an approximate number and tests show that the sensor's response is not linear. For smaller distances, the sensitivity is higher and for larger distances sensitivity is lower. A detailed study of the sensor's response is required if such a device is actually put to large scale production, but is beyond the scope of this project.

The sensor sends out ultrasonic waves (42kHz) and has in built electronics for optimization. People detection requires high sensitivity, yet a narrow beam angle requires low sensitivity. The LV-MaxSonar<sup>®</sup>-EZ1<sup>®</sup> balances the detection of people with a narrow beam width.

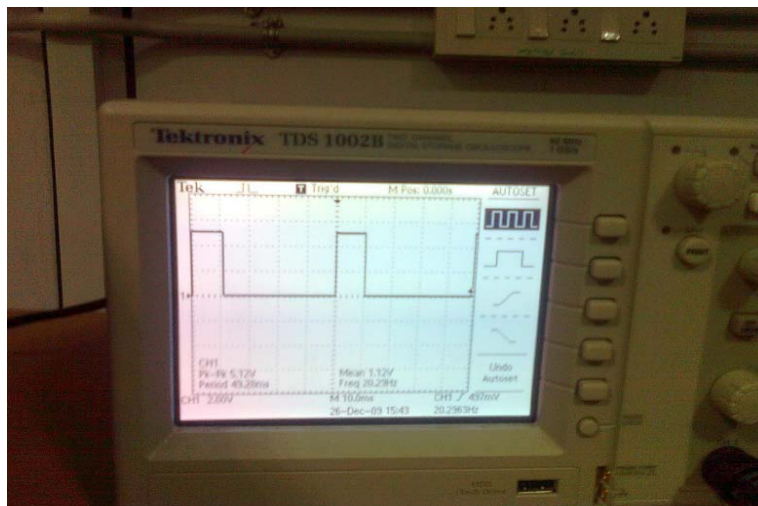
## 7.3 Sensor Response

Near distances (approximately 1 inch):



The pulse time period is 50millisecond approx. with a peak to peak voltage of 5.12V. On-time is less than 2 milliseconds.

Far distances (approximately 1 metre):



The pulse characteristics are the same. But the on-time (10 milliseconds) of the pulse shows considerable change

# 8 Algorithm and Flowchart

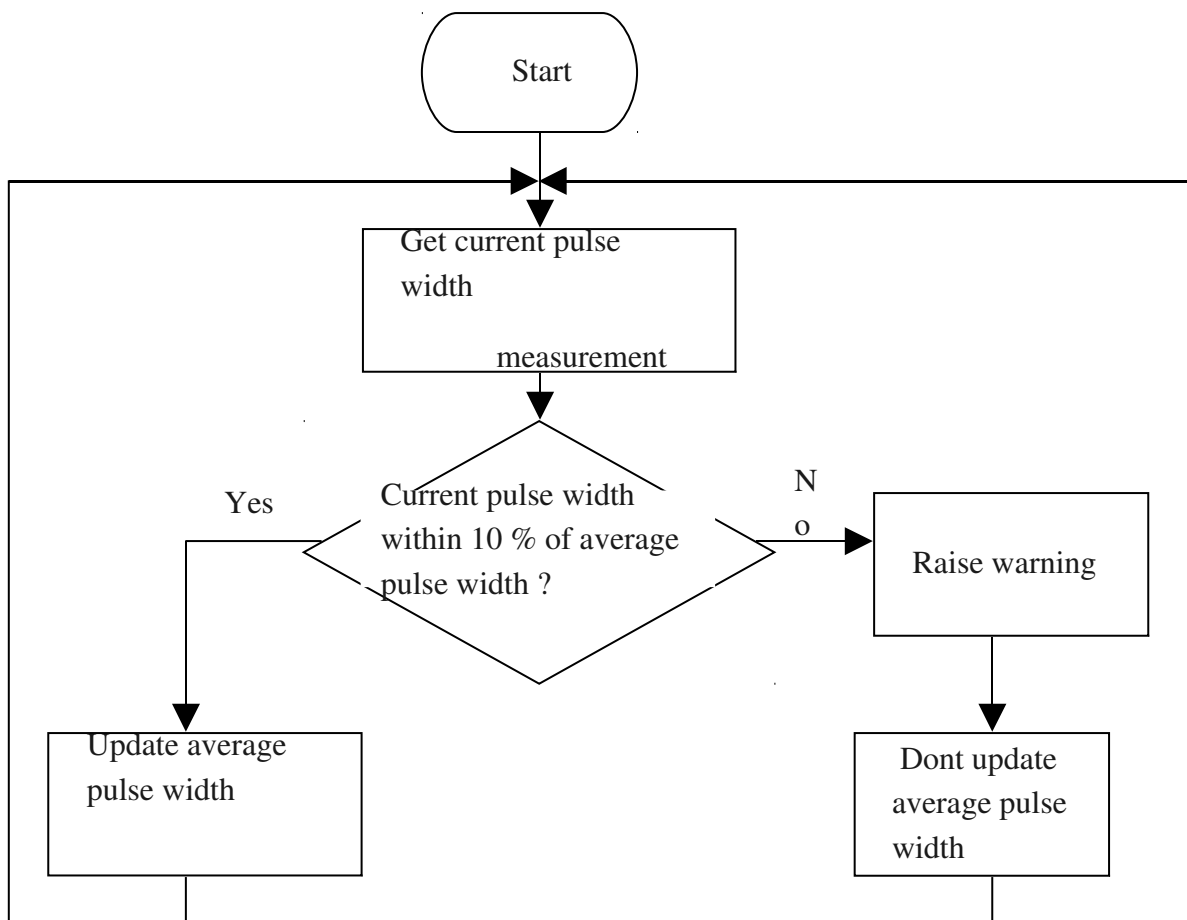
## 8.1 MSP Algorithm

1. Init()
  - a. Enable global interrupts.
  - b. Clear CPU Registers.
2. Port2\_initialize()
  - a. Set one of the pins of Port2 to input direction (This pin gets the input from the sensor).
  - b. Enable interrupts for the corresponding pin.
  - c. Set the interrupt edge select to rising edge.
3. Timer\_A initialize()
  - a. Set one of the capture compare registers to capture mode and connect the input of the capture compare block to the input from the sensor (Port2 pin).
  - b. Select SMCLK as the clock input to the timer module.
  - c. Set clock divider mode to divide by 1 mode.
  - d. Set the timer to continuous mode.
4. Pulse\_width() – Interrupt Service Routine(ISR) for an interrupt on the input pin
  - a. If the pin is rising edge sensitive then,
    - i. Reset the timer.
    - ii. Set the Port 2 interrupt select to falling edge sensitive.
  - b. If the pin if falling edge sensitive then,
    - i. Call Pulse\_width\_calculate().
    - ii. Set the Port 2 interrupt select to rising edge sensitive
  - c. Reset the interrupt flags.

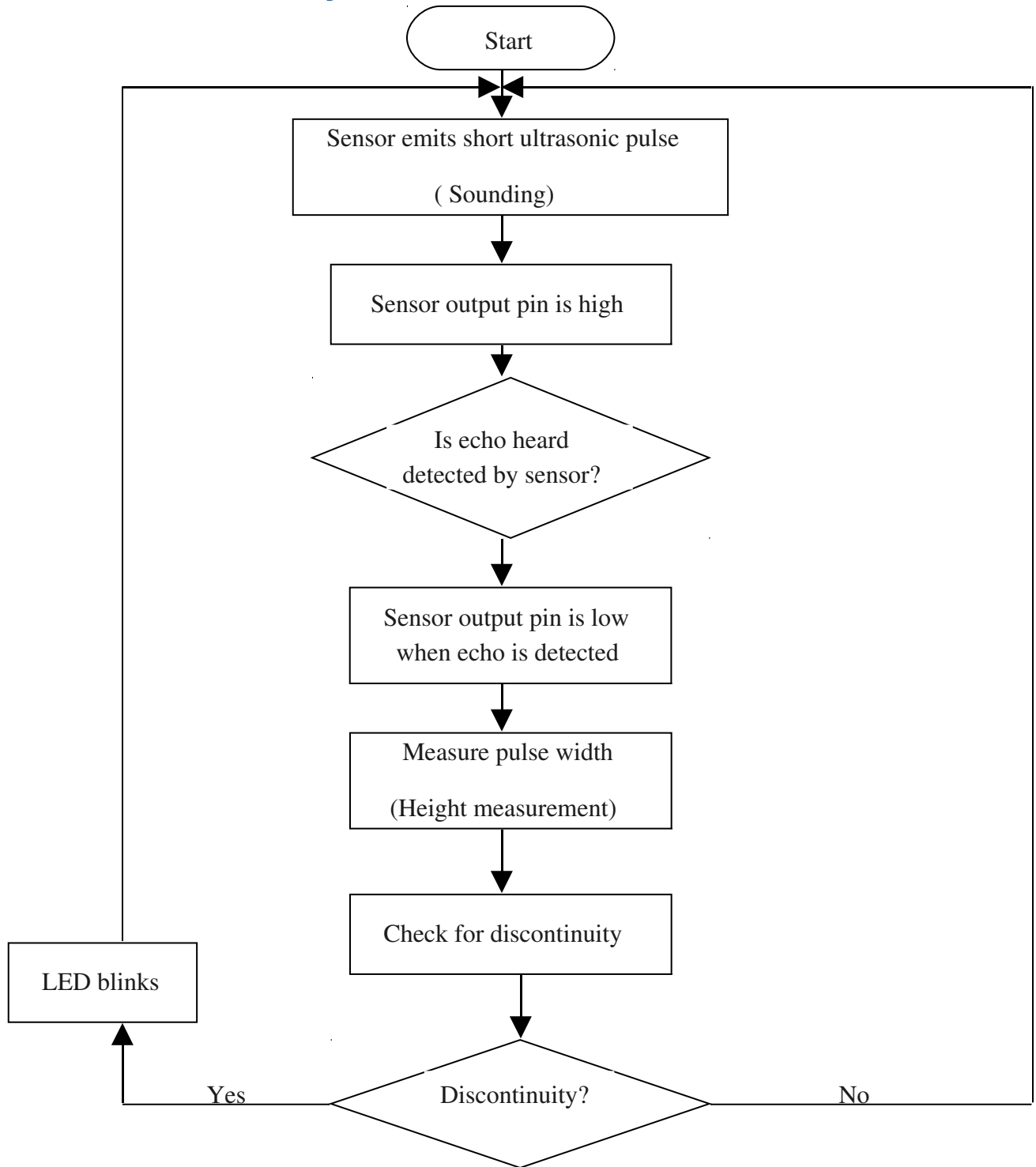
5. Timer\_overflow() - Interrupt Service Routine(ISR) when the timer overflows.
  - a. Increment a global variable 'abc'.
  - b. If the input is low, set the variable abc to 0.
  - c. Copy the contents of the variable to the CPU register R4.
  - d. Reset the interrupt flag.
6. Pulse\_width\_calculate()
  - a. Move the value of the timer that is captured in the capture/compare register to the CPU register R5.
  - b. Call function detect\_pothole().
7. detect\_pothole()
  - a. Compare the number of clock ticks of the current pulse with the cumulative average.
    - i. If the difference is lesser, then use the current sample to update the cumulative average of all sample widths.
    - ii. If the difference is greater, set a warning flag and wait for next interrupt from pulse width information from input. If the flag is ON for 10 consecutive pulses, then sound the buzzer to alarm the person of a discontinuity.

## 8.2 Flowcharts

### 8.2.1 Flowchart of Algo



## 8.2.2 Flowchart of System



# 9 Other Use Case Scenarios

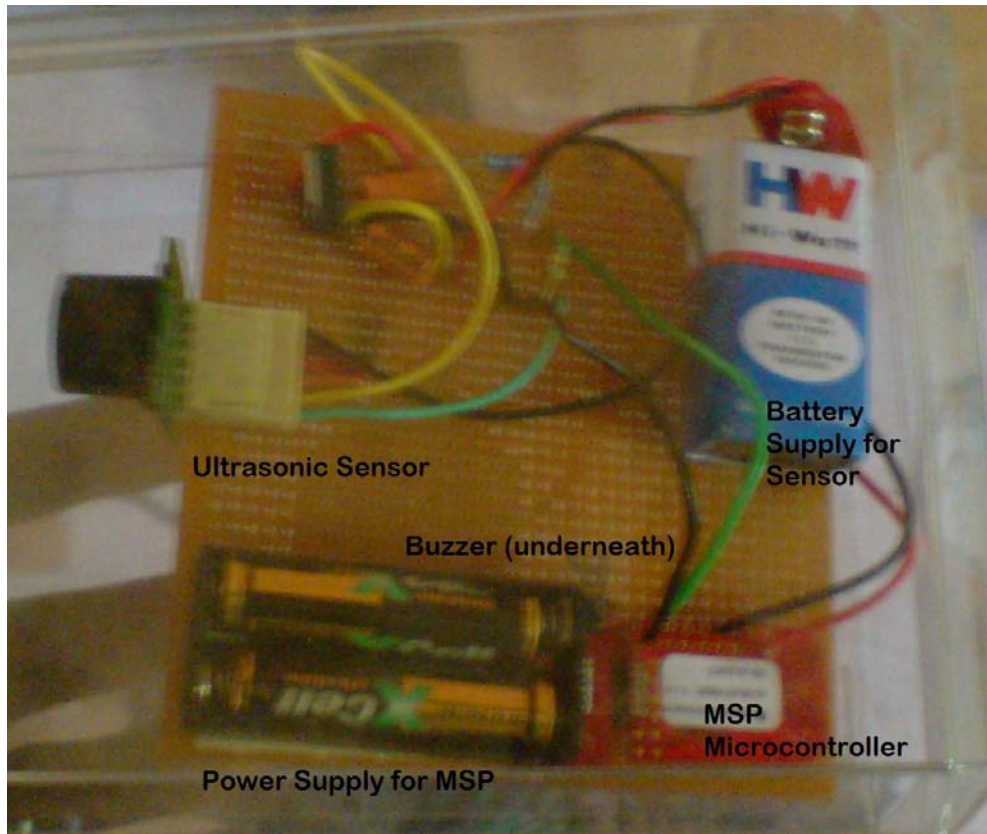
## Automobiles

A lot of interest has been shown by militaries all over the world on unmanned vehicles. Unmanned vehicles may become commonplace in the coming decades for operations on hostile terrain. The obstacle detector designed by the authors can play a very useful role in these vehicles. They can act as the eyes for the vehicle. The OD can also find application in regular automobiles for pothole detection and avoiding collisions with other vehicles, especially in highways where drowsiness is a common problem faced by people and which leads to many accidents.



The OD mounted here detects all possible obstacles like potholes, trenches, bumps, other vehicles, people etc.

## 10 Prototype picture



## 11 Cost of Prototype

Component	Cost in INR
MSP	<1000
Sensor	750
Voltage Regulators	40
Buzzer	10
Manufacturing Cost	<1800

## 12 Scope For Improvement

Phased array ultrasonics can be used instead of the single ultrasonic sensor used in this project. It is an advanced form of ultrasonics testing. A phased array basically refers to a bank of ultrasonic sensors. This kind of a device can be used to give more information to the user than a simple ultrasonic sensor.

The Phased Array(PA) probe consists of many small elements/individual transceivers, each of which can be pulsed separately. The multiple waves add up to one single wave front travelling at a set angle. In other words, the beam angle can be set just by programming the pulse timings. In other words, the beam can be steered electronically. A common application of phased array is imaging. For eg: using PA ultrasonic a defect in a solid can be detected along with its coordinates using the received sound waves. This could be incorporated in our project for more robust functioning of the obstacle detector. But the cost, complexity and power consumption are much more than a stand-alone transceiver.

The code attached below is designed to adjust to the user's height preferences in the first 10 seconds of switching on the OD. Any changes from this reference height is beeped as an alarm to the user. This becomes a problem if the user wants to suddenly sit or stand up. To take care of this problem, a reset switch can be added in the hardware and the software can be programmed to forget all the previous data and start afresh.

## 13 MSP Code

### 13.1 Source : Main.c

```
const int startup = 50, numberoferrorsamples = 5, steadystatelimit = 500;
const int startuplimit = 2000, power = 2;
int overflow=0,initial=0,ov1=0, ov2=0, t1=0, t2=0, dist=0, maxdist=0, num=0;
long int cumulative_average=0,samples=0;
```

```

void Port1_initialize()          //Setting initial states for warning LEDs
{
    P1DIR=0xFF;
    P1OUT=0x02;
    P3DIR=0x01;
    P3OUT=0x00;
}

void Port2_initialize()
{
    P2DIR = 0x00;          //Setting all port pins to input mode
    P2IE = 0x02;          //Enabling interrupts on pin 4 (P2.1)
    P2IES = 0x02;         //Choosing high-low transition for pin 4
    P2IFG = 0x00;         //Resetting interrupt flags
}

void wait()                    //Programmed delay before buzzer o/p is reset
{
    int i;
    for(i=0;i<30000;i++);
    for(i=0;i<30000;i++);
    for(i=0;i<30000;i++);
    for(i=0;i<30000;i++);
    for(i=0;i<30000;i++);
    for(i=0;i<30000;i++);
    for(i=0;i<30000;i++);
    for(i=0;i<30000;i++);
    for(i=0;i<30000;i++);
    for(i=0;i<30000;i++);
}

void detect_pothole()
{
    if(initial>startup)        //Ensuring that the startup time is over
    {
        asm("MOV.W R4,ov1");
        asm("MOV.W R5,t1");
        asm("MOV.W R6,ov2");
        asm("MOV.W R7,t2");
        dist = (ov2*65536+t2) - (ov1*65536+t1);
        if(samples==0) dist=0;
        if(dist<0) dist=-dist; //Measuring difference between width of
                                //current pulse and the cumulative average

        if(dist>(steadystatelimit)//+startuplimit/((samples+1)^power))
        {
            num++; //Maintain flag if the width has changed from the average
            if(num>numberoferrorsamples)
            {
                //If the deviation is sustained raise alarm
                P1OUT = 0x01;
                P3OUT = 0x01;
                asm("MOV.W dist,R9");
                wait();
                P1OUT = 0x02;
                P3OUT = 0x00;
            }
        }
    }
}

```

```

    }
    else //If there is no deviation use current sample to update
    { //the cumulative average
        num=0;
        samples=samples+1;
        if(samples>1)
            cumulative_average=(cumulative_average*(samples-1)+(ov1*65536+t1))/
(samples);
        else
        {
            samples=1;
            cumulative_average=ov1*65536+t1;
        }
        asm("MOV.W cumulative_average, R7");
    }
    asm("MOV.W dist,R8");
}
else
    initial++;
}

```

```

void switchled()
{
    if(P2IES==0x00) //If interrupted and set to rising edge sensitive
    {
        TACTL |=0x04; //Restart timer
        TACCTL2 = CM_2+CCIS_3+SCS+CAP;
        P2IES=0x02;
    }
    else if(P2IES==0x02) //If interrupted and set to falling edge sensitive
    {
        TACCTL2 = CM_2+CCIS_2+SCS+CAP;
        pulse_width_calculate(); //Calculate width of the pulse
        P2IES=0x00;
    }
    P2IFG =0x00; //Resetting flag registers
}

```

```

void timer_overflow() //Update overflow register if timer overflows
{
    if(P2IES==0x00)
        overflow=0;
    else
        overflow++;
    asm("MOV.W overflow,R4");
    TACCTL0 &= ~(0x1);
}

```

```

void TimerA_initialize()
{

```

```

TACTL = TASSEL_2 + ID_0 + MC_2; //+ TAIE; //Choosing SMCLK, undivided,
//continuous mode (counts to
//FFFF) and enabling interrupts

TACCTL2 = CM_2+CCIS_2+SCS+CAP; //Capturing on falling edge, input GND,
//synchronous capture, capture mode (2nd
//capture/compare register)
}

void main()
{
    WDTCTL = WDTPW + WDTHOLD; //Stops Watchdog
    asm ("MOV.W #0x600,SP"); //Initializing Stack Pointer
    init();
    Port1_initialize();
    Port2_initialize();
    TimerA_initialize();
    while(1);
}

```

## 13.2 Source : init.asm

```

PUBLIC          init
PUBLIC          keypad_init
PUBLIC          ir_init
PUBLIC          watchdog_init
PUBLIC          SetupTA
RSEG CODE
;ORG           0x01000
init           NOP
BIS.W         #GIE,SR
MOV.B         #KP+REM+RTC      , &UIMask_MSB
MOV          #00,R4
MOV          #00,R5
MOV          #00,R6
MOV          #00,R7
MOV          #00,R8
MOV          #00,R9
MOV          #00,R10
RET
END

```

## 13.3 Source : interrupt\_asm.s43

```

EXTERN          main
EXTERN          switchled

```

```
EXTERN          timer_overflow
EXTERN          detect_pothole

PUBLIC pulse_width_calculate

pulse_width_calculate  MOV TACCR2,R5
                       CALL #detect_pothole
                       RET

ISR_4             CALL #timer_overflow
                 RETI

ISR_10           ALL #switchled
                 RETI

                ORG 0FFF0h
                DW ISR_4
                ORG 0FFE6h
                DW ISR_10
                END
```