



## PROJECT 5: ULTRASONIC RANGE FINDER

EE 663: Real Time and Embedded Systems

### Abstract

A stand-alone QNX C/C++ program triggers an ultrasonic range finder with a digital pulse lasting less than 1ms and determines the distance between the sensor and the obstacle based on the echo pulse generated by the ultrasonic transceiver. The distance is measured based on the time between the transmission of the pulse to the detection of the first wave front arriving back at the sensor.

**Author : Vyoma Sharma ; Siddharth Ramkrishnan**

Email ID : sg5232@rit.edu ; sxr4316@rit.edu

## Table of Contents

1. Problem Statement.....	2
1.1. Design Constraints .....	2
1.2. Technical Side Note .....	2
2. Overview .....	3
3. Areas of Focus .....	3
4. Analysis / Design.....	4
5. Test Plan .....	5
6. Project Results .....	5
7. Lessons Learned .....	6
8. Conclusion.....	7

## 1. Problem Statement

Design and implement an embedded, stand-alone QNX Neutrino program to measure the distance between the rear bumper of your car and any objects behind the vehicle while parking.

### 1.1. Design Constraints

- ✚ The distance is measured at a rate of 10 times per second.
- ✚ The results of the measurement is displayed on the console so that the value does not scroll.
- ✚ The measured results are rounded to the nearest inch and displayed as integer values only.
- ✚ Out-of-range measurements are represented as a flashing asterisk.
- ✚ After measurements are ended, display the maximum and minimum distances measured.
- ✚ In your report, include test cases and explicit results indicating the practical range of your ultrasound sensor

### 1.2. Technical Side Note

- ✚ Positive pulse, that triggers the sensor must remain high for at least 10  $\mu$ s.
- ✚ The duration of the output from the ultrasound sensor will be from 100  $\mu$ s to 18 ms.
- ✚ The speed of sound in air at room temperature is assumed to be 769.55 mph (17.6 in / s)

## 2. Overview

The objective of the system is to trigger the ultrasonic range finder, with a 'positive' ping, lasting at least 10  $\mu\text{s}$ , and then monitor the echo data line for signal transitions. The time between the rising and falling edge of the echo signal line is directly proportional to the distance between the sensor and the obstacle.

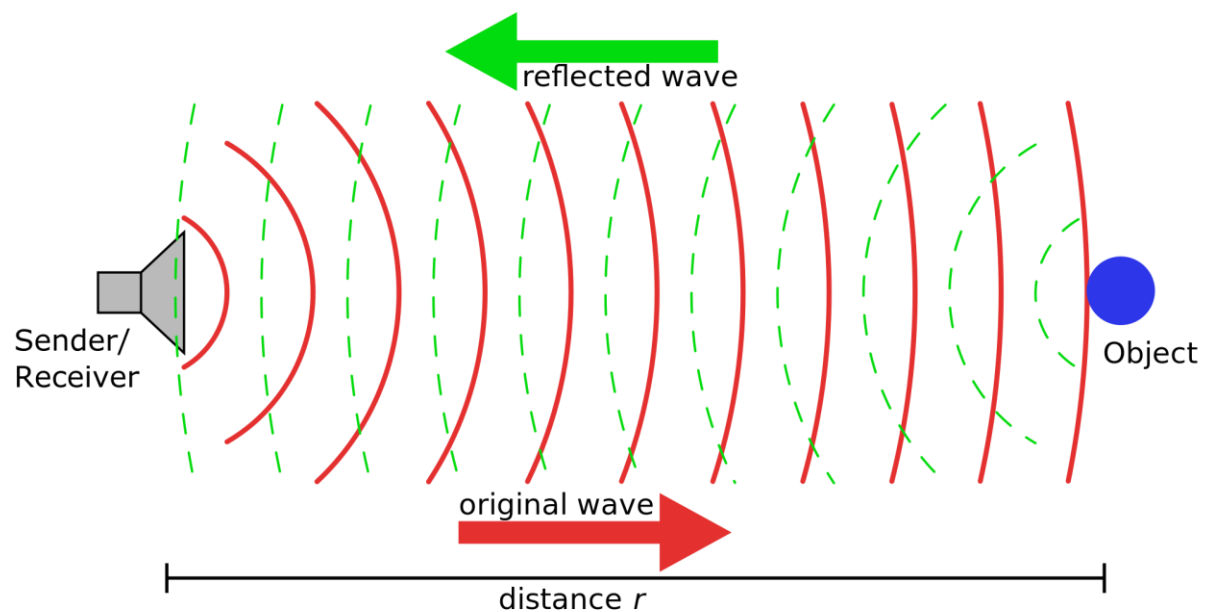


Figure 1. Principle of working of Ultrasonic Range Finder (Georg Wiora (Dr. Schorsch) via Wikimedia))

$$\text{Distance} = (\text{Speed of Sound in medium}) \times (\text{Propagation time}) / 2$$

## 3. Areas of Focus

The simple nature of the project meant, there was not concrete split of tasks among the team members. The algorithm and the functional design was jointly implemented by the members of the team. The comprehensive hardware information required for the project was already provided along with the class material.

## 4. Analysis / Design

The requirements are approached in a two prong approach. There are two independent lines of control operating in the system, one line of control is the timer which autonomously generates the sensor trigger 'ping' pulses, lasting 1 ms at the rate of once every 100 ms. The other line of control is the *main()* thread, which continually polls the echo line and monitors for signal transitions. The system hardware clock ticks are captured to determine the time interval between the various events occurring in the system.

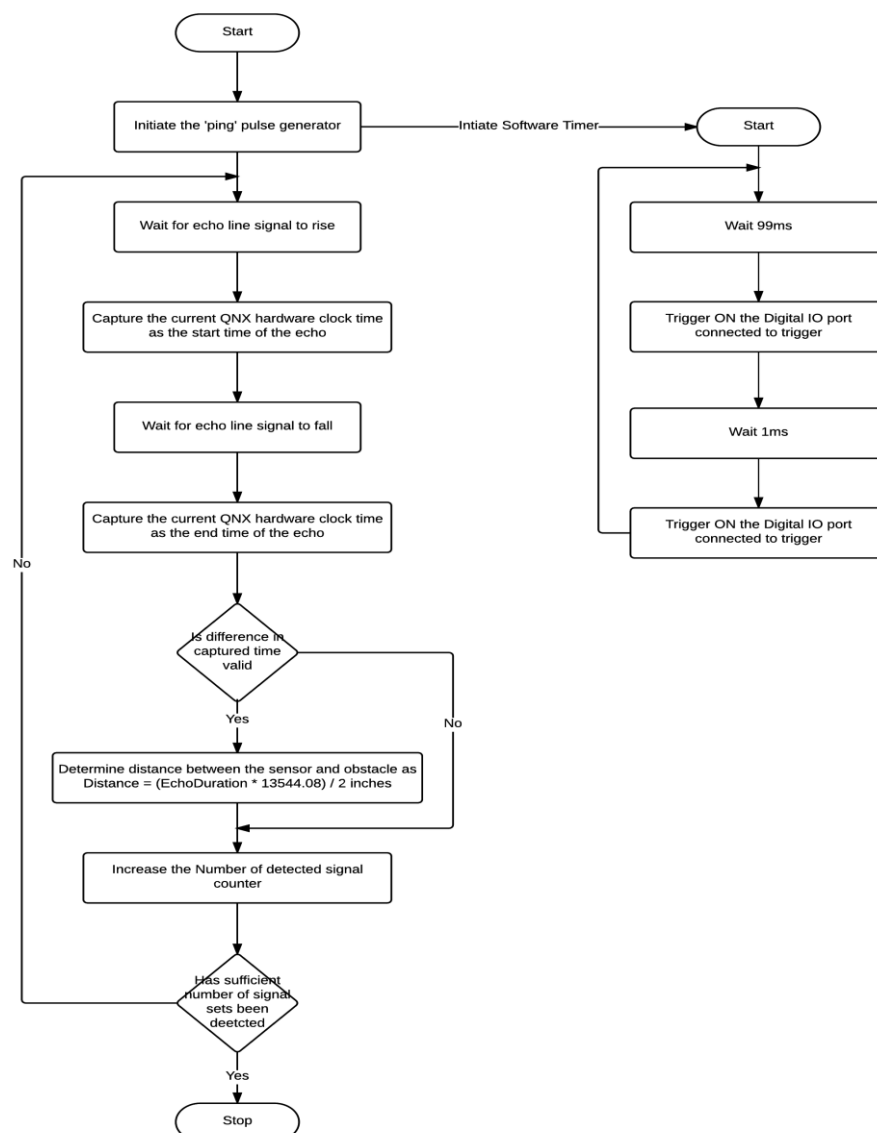


Figure 2. Implementation algorithm for operation of Ultrasonic Sensor

## 5. Test Plan

The standard test plan consist of the obstacle placed at different distances from the sensor. The standard required test case distances were

- 3 inches
- 2 feet (24 inches)
- Obstacle beyond valid range
- Obstacle too close to the sensor

Additionally a dynamically varying distance test is performed where the obstacle moves towards and away from the sensor to detect the updating speed of the sensor detection system.

## 6. Project Results

### *Ultrasonic Sensor Operation*

<i>Echo Duration : 0.005668 s</i>	<i>Measured Distance : 38 inches</i>
<i>Echo Duration : 0.001063 s</i>	<i>Measured Distance : 7 inches</i>
<i>Echo Duration : 0.002464 s</i>	<i>Measured Distance : 17 inches</i>
<i>Echo Duration : 0.001132 s</i>	<i>Measured Distance : 8 inches</i>
<i>Echo Duration : 0.001942 s</i>	<i>Measured Distance : 13 inches</i>
<i>Echo Duration : 0.017140 s</i>	<i>Measured Distance : 116 inches</i>
<i>Echo Duration : 0.016995 s</i>	<i>Measured Distance : 115 inches</i>
<i>Echo Duration : 0.016816 s</i>	<i>Measured Distance : 114 inches</i>
<i>Echo Duration : 0.016688 s</i>	<i>Measured Distance : 113 inches</i>
<i>Echo Duration : 0.016580 s</i>	<i>Measured Distance : 112 inches</i>
<i>Echo Duration : 0.000796 s</i>	<i>Measured Distance : 5 inches</i>
<i>Echo Duration : 0.001933 s</i>	<i>Measured Distance : 13 inches</i>
<i>Echo Duration : 0.000599 s</i>	<i>Measured Distance : 4 inches</i>
<i>Echo Duration : 0.016490 s</i>	<i>Measured Distance : 112 inches</i>
<i>Echo Duration : 0.016463 s</i>	<i>Measured Distance : 111 inches</i>
<i>Echo Duration : 0.016218 s</i>	<i>Measured Distance : 110 inches</i>
<i>Echo Duration : 0.000447 s</i>	<i>Measured Distance : 3 inches</i>
<i>Echo Duration : 0.000187 s</i>	<i>Measured Distance : 1 inches</i>
<i>Echo Duration : 0.000210 s</i>	<i>Measured Distance : 1 inches</i>
<i>Echo Duration : 0.048625 s</i>	<i>Measured Distance : *****</i>
<i>Echo Duration : 0.048044 s</i>	<i>Measured Distance : *****</i>
<i>Echo Duration : 0.047918 s</i>	<i>Measured Distance : *****</i>
<i>Echo Duration : 0.017536 s</i>	<i>Measured Distance : 119 inches</i>
<i>Echo Duration : 0.018208 s</i>	<i>Measured Distance : *****</i>
<i>Echo Duration : 0.018979 s</i>	<i>Measured Distance : *****</i>
<i>Echo Duration : 0.002948 s</i>	<i>Measured Distance : 20 inches</i>
<i>Echo Duration : 0.001816 s</i>	<i>Measured Distance : 12 inches</i>
<i>Echo Duration : 0.002614 s</i>	<i>Measured Distance : 18 inches</i>
<i>Echo Duration : 0.002046 s</i>	<i>Measured Distance : 14 inches</i>
<i>Echo Duration : 0.000168 s</i>	<i>Measured Distance : 1 inches</i>

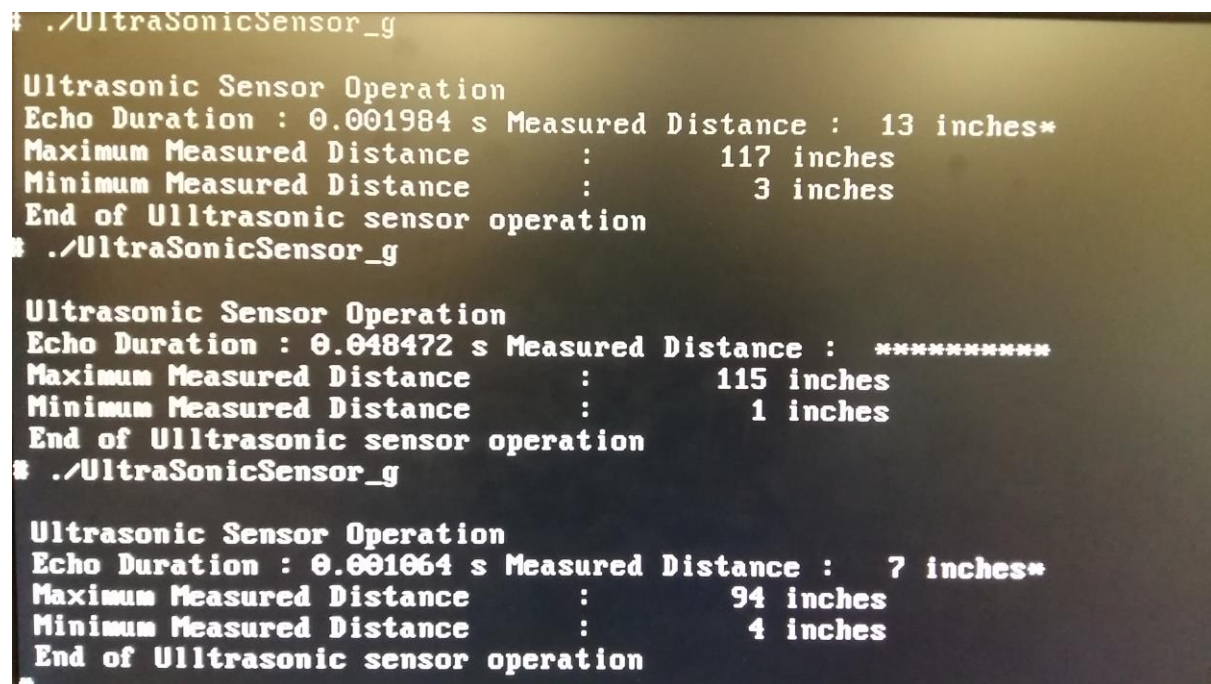
*Maximum            Measured Distance        :            119 inches*

*Minimum           Measured Distance        :            1 inches*

*End of Ultrasonic sensor operation*

It is observed that the sensor control system detects a valid distance only when the measured echo duration is within the range of 100  $\mu$ s and 18 ms, in accordance with the sensor and project specifications.

Each run of the software, detects 500 echoes from the sensor, which are analysed and interpreted.



```
# ./UltraSonicSensor_g

Ultrasonic Sensor Operation
Echo Duration : 0.001984 s Measured Distance : 13 inches*
Maximum Measured Distance : 117 inches
Minimum Measured Distance : 3 inches
End of Ultrasonic sensor operation
# ./UltraSonicSensor_g

Ultrasonic Sensor Operation
Echo Duration : 0.048472 s Measured Distance : *****
Maximum Measured Distance : 115 inches
Minimum Measured Distance : 1 inches
End of Ultrasonic sensor operation
# ./UltraSonicSensor_g

Ultrasonic Sensor Operation
Echo Duration : 0.001064 s Measured Distance : 7 inches*
Maximum Measured Distance : 94 inches
Minimum Measured Distance : 4 inches
End of Ultrasonic sensor operation
#
```

Figure 3. Sample Output Data generated from the system

As required, it is observed that there is no scrolling of data for each application execution, the minimum and maximum distance observed are displayed once the required number of data samples have been accumulated.

## 7. Lessons Learned

The major takeaways from the project were the use of simplistic data polling options whenever available. Initially in order to minimize the number of clock cycles used by the application, it was intended that the interrupt events which toggle the DIO port, would initiate individual threads which would measure the current pulse, determine the distance and terminate. This would be repeated for all the operations of the sensor. The extensive instruction of control operation caused variations in the square pulse generation.

## **8. Conclusion**

The project was a simple introduction to using timers, and digital IO signals to communicate with sensor and other hardware peripherals. The use of multiple threads and attempts to complicate the implementation, although delayed the project, was extremely relevant as a learning experience. The final solution of splitting the pulse generation and the signal polling allows for greater flexibility in determining the rate of measurement, without any significant changes in the system operation.