

we included the timerone library. This library is used to repetitively measure a period of time in microseconds and at the end of each period, an interrupt function will be called.

We have used this library because we want to read from the sensors and control LED’s at the same time. We will have to use the delay in between the traffic signal so we can’t read from the sensors continuously. Therefore we have used this library which will allow us to call a function in which we will read from the sensors continuously and in the loop function, we will control the traffic signals.

#include<TimerOne.h>

In the setup function, we have used the Timer1.initialize(microseconds) function. This must be called before you use any of the other methods of timerone library. “Microseconds” is actually the period of time the timer takes. It is optionally to specify the timer’s period here. The default period is 1 second. Keep in mind that it breaks analogWrite() on digital pins 9 and 10.

Timer1.initialize(100000);

Timer1.attachInterrupt(softInterr) calls a function each time the timer period finishes. We have set the timer period at 100000 so our function will be called after 100 milli seconds.

Timer1.attachInterrupt(softInterr);

In the loop function it is looking if there is any vehicles under the 5 cm distance or not. If there will be vehicle, then the function to that signal will be called.

void loop()  
{  
 // If there are vehicles at signal 1  
 if(S1<t)  
 {  
 signal1Function();  
 }  
 // If there are vehicles at signal 2  
 if(S2<t)  
 {  
 signal2Function();  
 }  
 // If there are vehicles at signal 3  
 if(S3<t)  
 {  
 signal3Function();  
 }  
 // If there are vehicles at signal 4  
 if(S4<t)  
 {  
 signal4Function();  
 }  
}

‘Softinterr()’ is the interrupt function and it will called after every 100 milliseconds. In this function, we have read from the ultrasonic sensors and have calculated the distance.

void softInterr()  
{  
 // Reading from first ultrasonic sensor  
 digitalWrite(triggerpin1, LOW);   
 delayMicroseconds(2);  
 digitalWrite(triggerpin1, HIGH);   
 delayMicroseconds(10);  
 digitalWrite(triggerpin1, LOW);  
 time = pulseIn(echopin1, HIGH);   
 S1= time\*0.034/2;