The ISU unit will be used to build a simulation of prototype for an ultrasonic range sensor using the available HC-SR04 ultrasonic range sensors. At every 10 second period, the sensor shall get the pulse width of HC-SR04 output, convert it to the distance in millimeters, and transmit it serially as an EIA232 TTL level 9600 Baud ASCII 4-digit decimal number and an end of line (ASCII code 0Ah) character to a terminal. The unit is made up of the following components, in the table below:

|  |  |  |
| --- | --- | --- |
| Component Name | Value | Quantity |
| Virtual Pattern Generator | 100R | 1 |
| PIC18F252 |  | 1 |
| ISU Unit | PIC18F255 | 1 |
| Virtual Terminal |  | 1 |
| LED | 10mA | 1 |
| Oscilloscope |  | 1 |
| 4 MHz Crystal | - | 1 |

**Wiring of the sensor:**

The four pin male header provides the following terminals:

1 5V Supply

2 TRIG Trigger Pulse Input

3 ECHO Echo Pulse Output

4 0V Ground

**Electric Parameter:**

Working Voltage DC 5V

Working Current 15Ma

Workıng Frequency 40Hz

Max Range 4m

Min Range 2cm

Measuring-Angle 15 degree

Trigger Input Signal 10Us TTL pulse

Dimension 45\*20\*15mm

Terminalls Vcc, Trig, Echo, GND

**Task B:**

The following is a test circuit, that tests the simulation of HC-SR04 using a button swith to apply a pulse to TRIG input, and an oscilloscope to observe the duration of ECHO pulse by setting the trigger to negative edge of TRIG signal.

Change the clock time of the pattern generator in the range of 10us – 5ms at least for 10 evenly distributed values.

Solution:

Convert 5ms to uS :

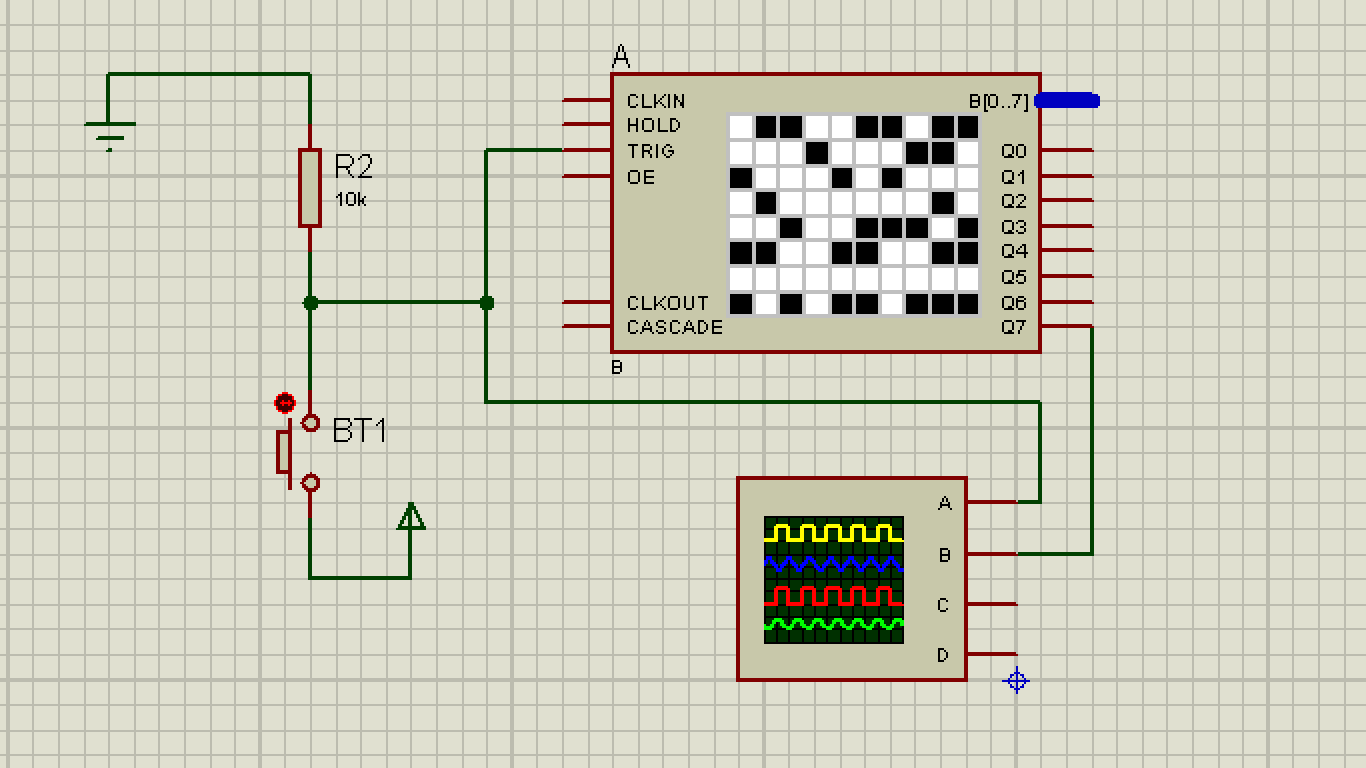
Range is 10uS – 5000uS

Scale: 5000 - 10= 4999 uS

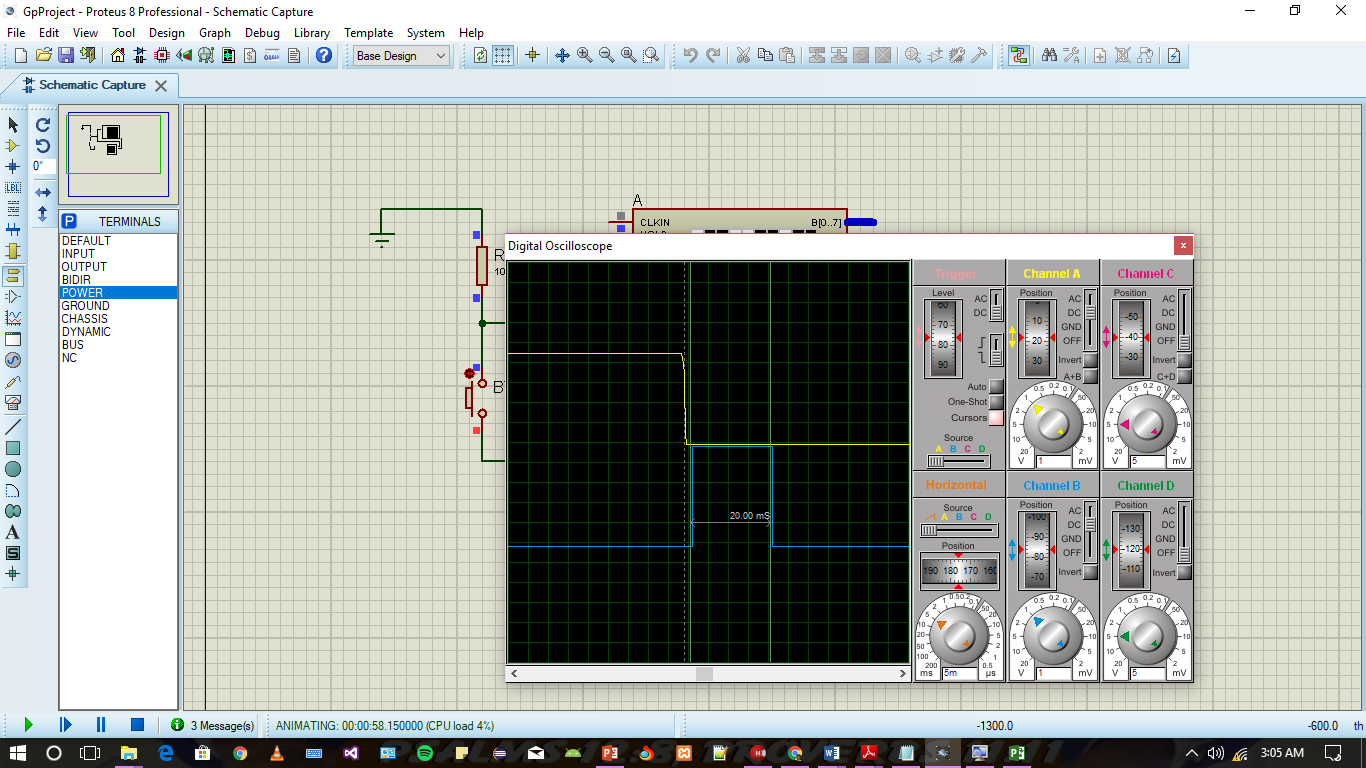
* 4999/10 = 499 separation between each distributed value

|  |  |  |
| --- | --- | --- |
| **Clock Time (of Pattern Generator) uS** | **ECHO Pulse Time (Tp)** | **Distance (Tp X 0.340)** |
| 10 | 100 | 34 |
| 598.1 | 6 | 2.04 |
| 1098 (1096) | 10 | 3.4 |
| 1597 | 16 | 5.44 |
| 2096 | 21 | 7.14 |
| 2595 | 26 | 8.84 |
| 3094 | 31 | 10.54 |
| 3593 | 36 | 12.24 |
| 4092 | 41 | 13.94 |
| 4591 <=5ms | 46 | 15.64 |

Schema Design

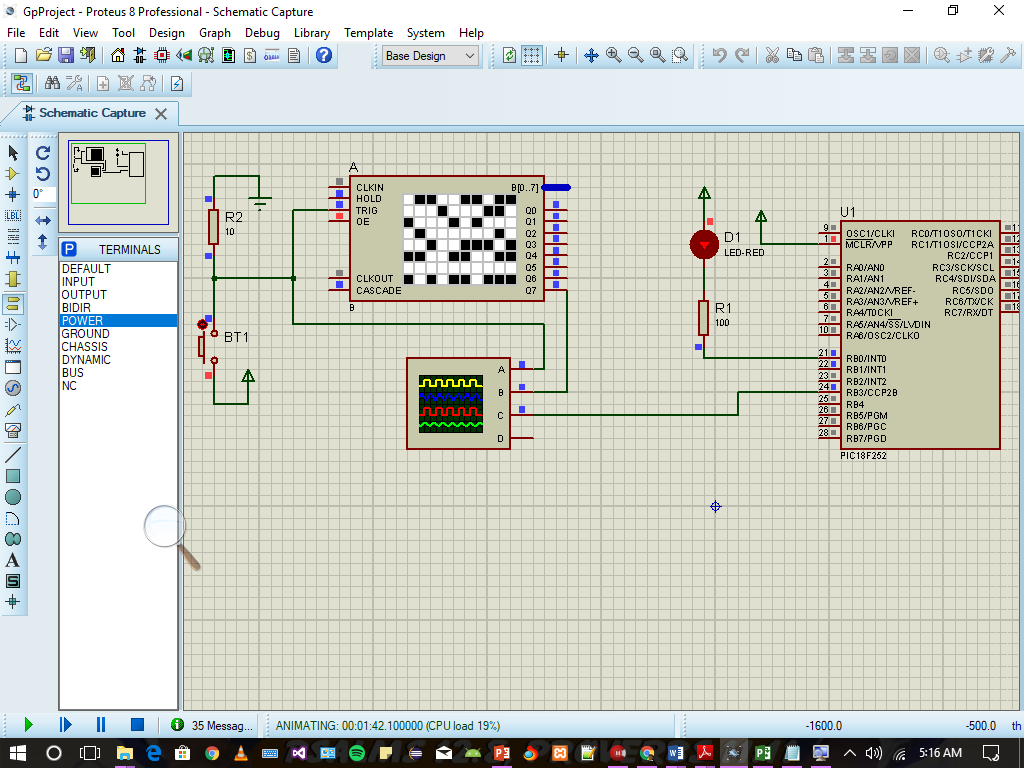


Correct Pulse- 20uS



Task C

Add on your test circuit a PIC18F252 (uC) and a Blinkalive led. On this second test circuit write a program code with 0.1 ms timebase that can implement blinkalive function, and build the simulation of the circuit to test it in ISIS environment.



Timer Setting Calculation:

For blinking 1 time every 10s

Count= 10/0.0001 = 100 000 CC

🡪 = 1.53 < PS

PS=2

Nc= Ncc/PS 🡪 100 000/2 = 50 000

To count 50 000 Prescaled clock cycles we set TMR0H = - Nc/256 & TMR0L = - Nc%256

Code:

void BlinkAlive(void)

{

char bcount;

if(bcount==0) PORTB.0=0; //LED ON

else PORTB.0=1; //LED OFF

bcount++;

if(bcount>9) bcount=0;

}

....

BlinkAlive();

Do{ T0CON = 0b100000000; TMR0H = - 50 000/256; TMR0L = - 50 000%256;

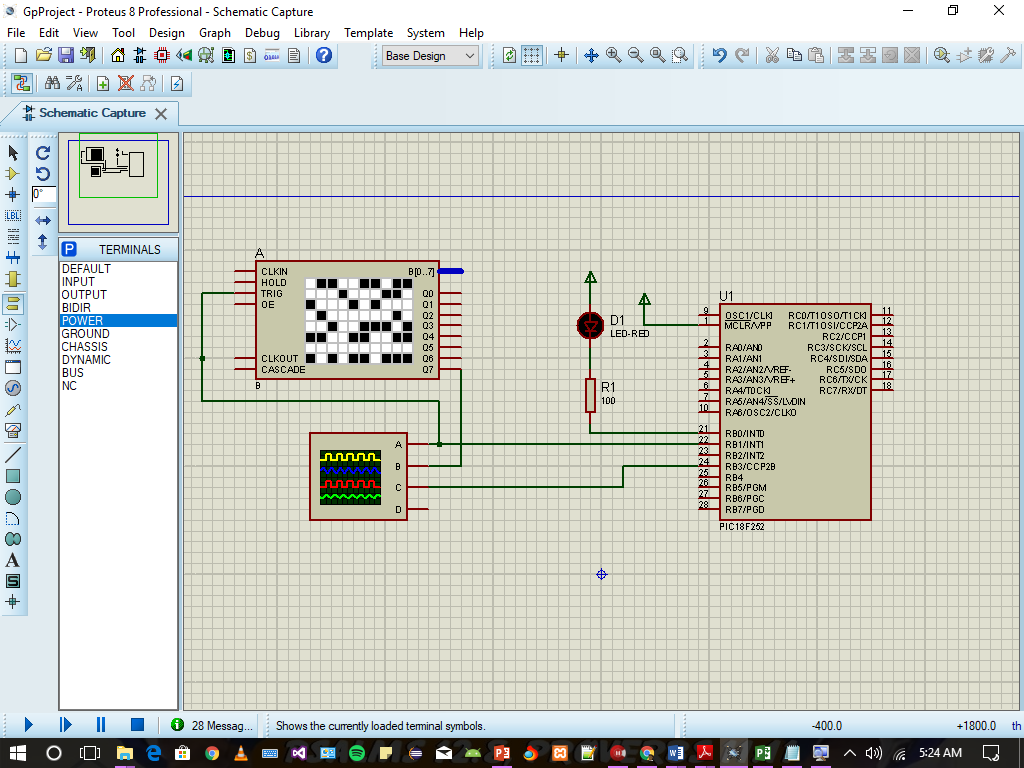
.......

}while(!TMR0IF);

Task D:

On your next test circuit remove the TRIG button, and connect TRIG to your uC. Modify the main program by adding a counter to count 0.1s periods for 10 seconds. Add a code to generate an approximately 20us trig pulse once at every 10 seconds. Use lst file to determine the period of TRIG pulse. Determine the exact period of the TRIG pulse by oscilloscope.

TRIG connected to Micro-Controller Schema



Code Tasks:

Adds counter to count 0.1s periods for 10 seconds.

Generates an 20Us TRIG pulse once at every 10 seconds

#define ECHOPin PORTB.2

#define ECHOTrs TRISB.2

#define TXPin PORTB.3

#define TXTrs TRISB.3

#define NC 50000

uns16 t0,t1,tp;

char T100m;

void BlinkAlive(void)

{ char bcount; if (bcount==0) PORTB.0=0; //LED ON

else PORTB.0=1; //LED OFF

bcount++;if (bcount>9) bcount=0; }

void ECHO\_WAIT\_UP()

{ECHOTrs=1; // Input for ECHO at RB2

do{}while(!ECHOPin);// portb.2=0}

void ECHO\_WAIT\_DN()

{ ECHOTrs=1; // Input for ECHO at RB2

do{}while(ECHOPin); //portb.2=1 }

void main(void)

{//Configuration

TRISB.0=0; //BLINKALIVE OUTPUT AT RB0

TRISB.1=0; TRISB.3=0;

TRISB.1=0; // OUTPUT TRIG (PULSE IS HIGH)

do{

T0CON=0b10000000; // set TMR0 for 0.1 sec

TMR0H=-NC/256; TMR0L=-NC%256;TMR0IF=0;

BlinkAlive();

++T100m; //increment by one to count 10s time period.

if(T100m>100)//true, then 10s is over

{

T100m=0; //clear to start count again

//TRIG passed 20us pulse

PORTB.1=1; //Output high for Trig

char i=20/3;

do{}while(--i);

PORTB.1=0;

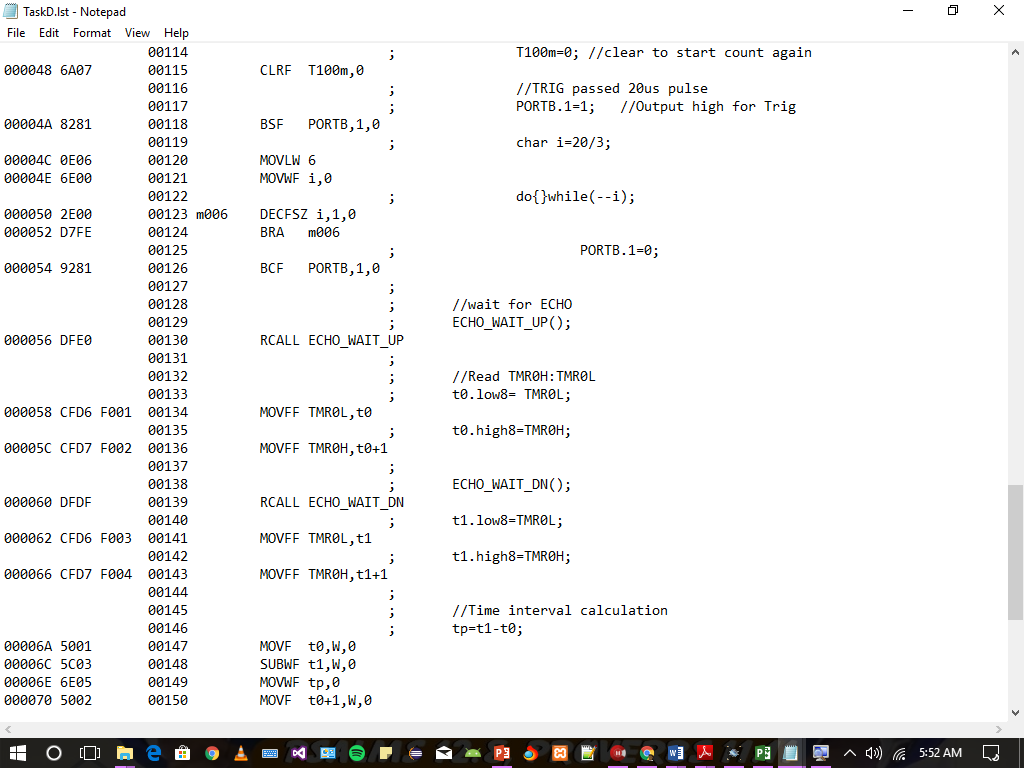
ECHO\_WAIT\_UP(); //wait for ECHO

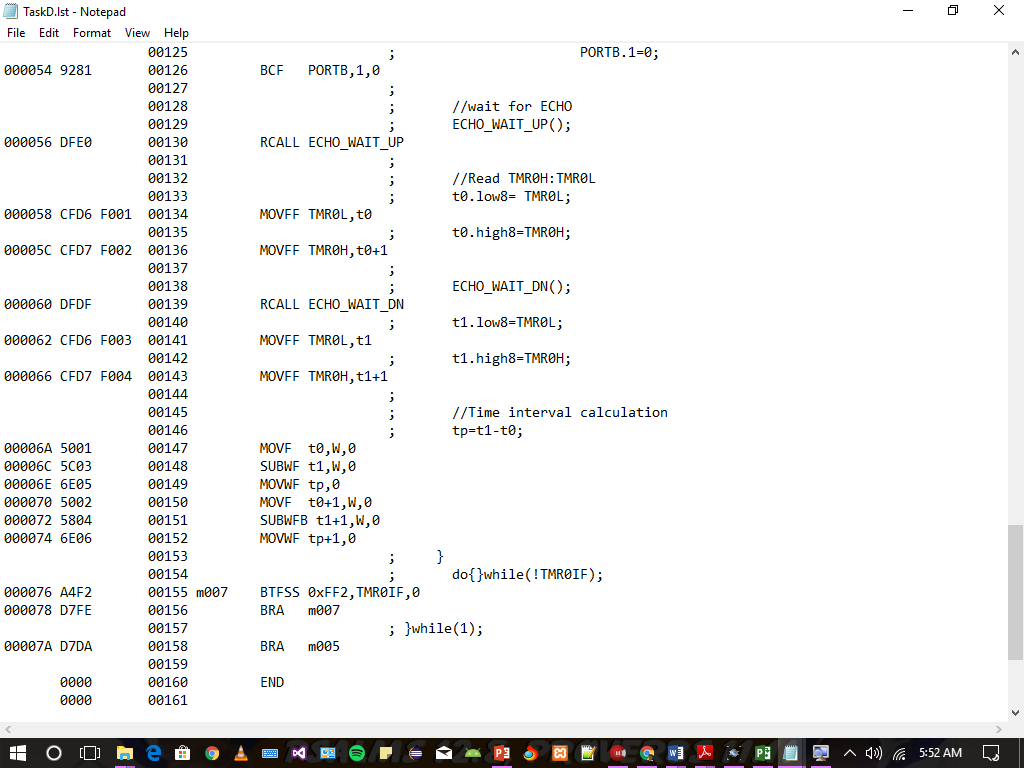
//Read TMR0H:TMR0L t0.low8= TMR0L; t0.high8=TMR0H;

ECHO\_WAIT\_DN(); t1.low8=TMR0L; t1.high8=TMR0H;

//Time interval calculation tp=t1-t0;} do{}while(!TMR0IF); }while(1); }

**Read 20uS Pulse from Lst file:**

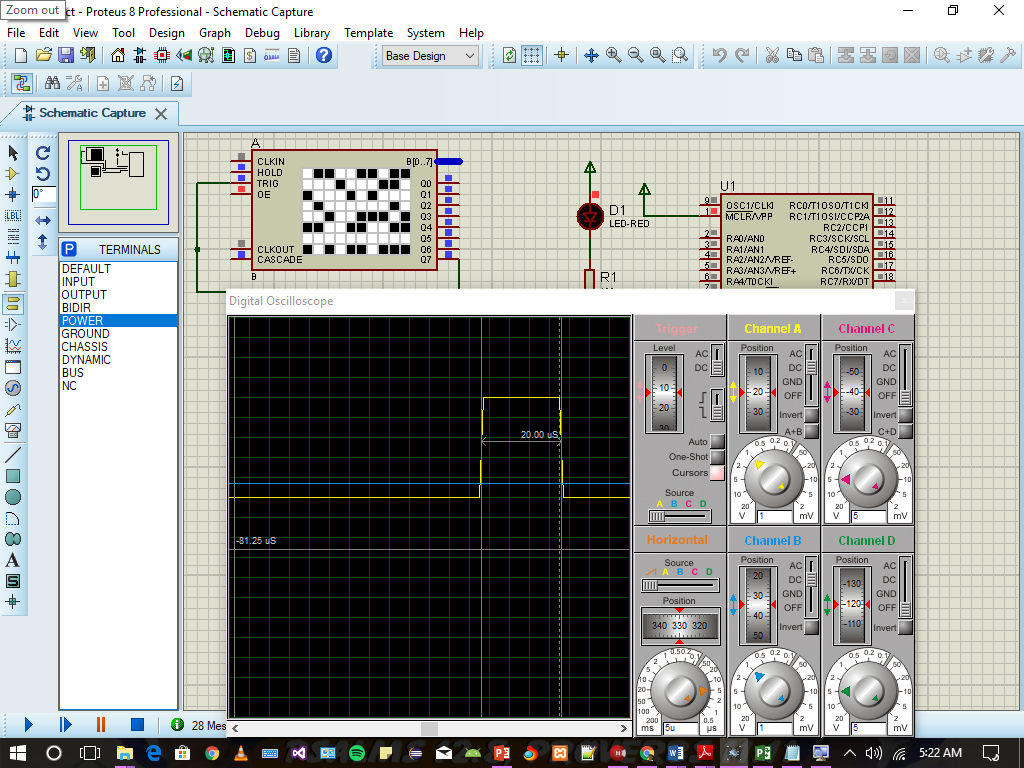




**Pulse count starts from BSF PORTB,1,0 till BRA m007 with every instruction being 1CC and every “BRA” instruction is 2 CC.**

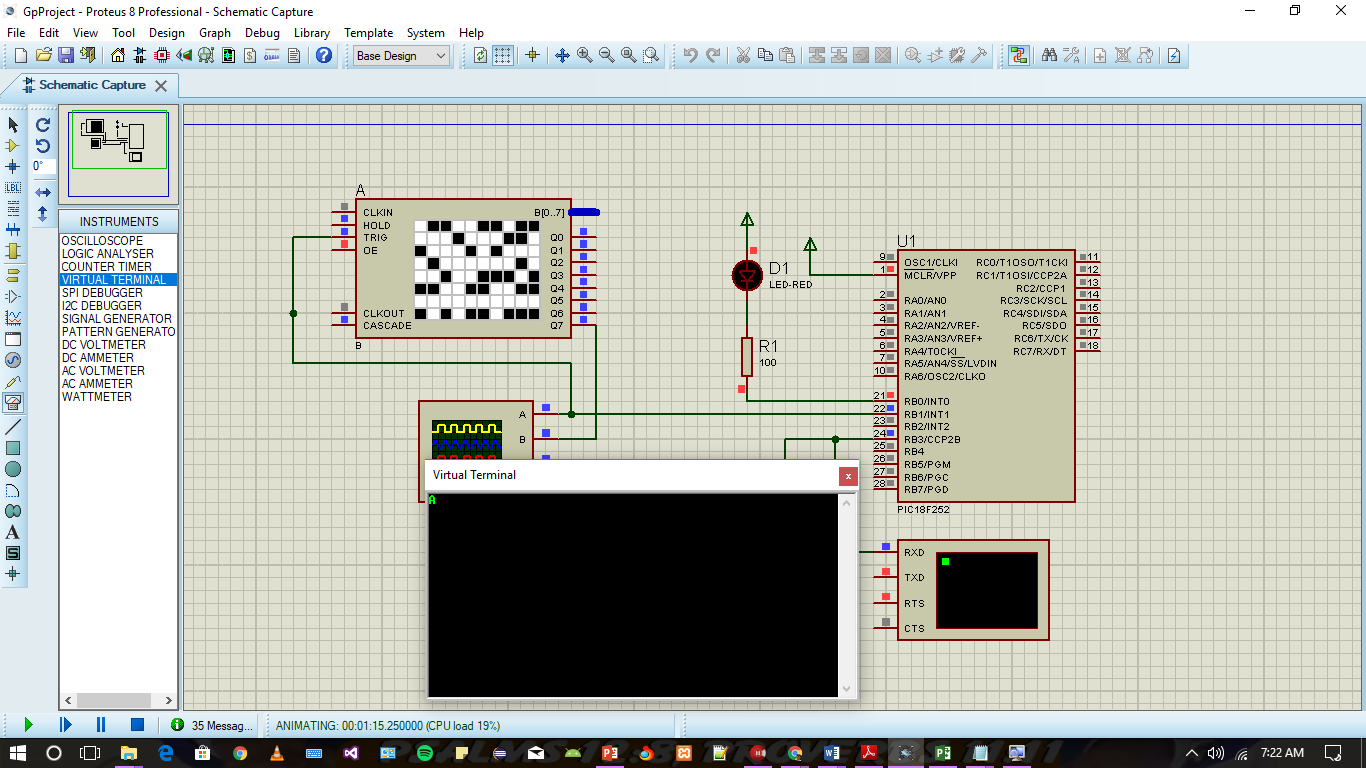
**Total Clock Cycles =20 CC**

**Read The Exact Period (20Us) of TRIG PULSE from Oscilloscope :**



Task E

On your next text circuit connect a virtual terminal. Write a transmit-character procedure in a complete C program that transmits a character at every second, starting from 0x30 to 0x39, at 9600 baud with 4MHz oscillator. Test it in ISIS environment using the virtual terminal



Task E

Write necessary code to read the timer at the start and at the end of the ECHO pulse. Develop a solution to detect failed sensors when ECHO takes longer than 40 ms

void ECHO\_WAIT\_UP()

{ECHOTrs=1; // Input for ECHO at RB2

do{}while(!ECHOPin);// portb.2=0}

void ECHO\_WAIT\_DN()

{ECHOTrs=1; // Input for ECHO at RB2

do{}while(ECHOPin); //portb.2=1}

....

....

....

//wait for ECHO

ECHO\_WAIT\_UP();

//Read TMR0H:TMR0L

t0.low8= TMR0L;

t0.high8=TMR0H;

ECHO\_WAIT\_DN();

t1.low8=TMR0L;

t1.high8=TMR0H;

//Time interval calculation

tp=t1-t0;

//TO DETECT FAILED SENSOR Check if Tp is greater than 40ms if its true BLINKALIVE SHOULD STOP WORKING

if(tp>40)

TRISB.0=1; //TURN OFF BLINKALIVE

**Task G**

Write necessary code that converts the ECHO time-count to distance in millimeters. At this point we have no test facilities on our system for the result of calculation..

Using #include <math24F.h>

//wait for ECHO

ECHO\_WAIT\_UP();

//Read TMR0H:TMR0L

t0.low8= TMR0L;

t0.high8=TMR0H;

ECHO\_WAIT\_DN();

t1.low8=TMR0L;

t1.high8=TMR0H;

//Time interval calculation

tp=t1-t0;

//timer calculation

TXhex(t0.high8);TXhex(t0.low8);TXchar('/');

TXhex(t1.high8);TXhex(t1.low8);

TXchar('/');

dmm=tp\*0.34;

WREG=84;multiply(tp.high8);

dmm.low8=PRODL;dmm.high8=PRODH;

multiply(tp.low8);

dmm.low8+=PRODH;

if(Carry)++dmm.high8;

Task H:

Write a procedure that converts a 16-bit integer to ascii codes and puts them in global char variables a4...a0. Add a test code that converts numbers n={10510, 12345, 909, 1} to ascii, and sends them using transmit-character function. Do not forget to put end of line (0x0D) after transmitting the number.

char a5,a4,a3,a2,a1;

void u16toASCII(char \*ap , uns16 dmm) {

uns16 i=dmm;

\*ap='0'; while(i>9999){++\*ap; i-=10000;} ++ap;

\*ap='0'; while(i>999){++\*ap; i-=1000;} ++ap;

\*ap='0'; while(i>99){++\*ap; i-=100;} ++ap;

\*ap='0'; while(i>9){++\*ap; i-=10;} ++ap; \*ap='0'+i;

++ap; \*ap=0 ;

}

void Int16AsciiConvert(uns16 n)

{

uns16 i=n;

a5='0'; while(i>9999){++a5; i-=10000;}

a4='0'; while(i>999){++a4; i-=1000;}

a3='0'; while(i>99){++a3; i-=100;}

a2='0'; while(i>9){++a2; i-=10;}

a1='0'+ i;

}

void TXASCII()

{

TXchar(a5);TXchar(a4);TXchar(a3);

TXchar(a2);TXchar(a1); TXchar(0x0D);

}

void main(void)

{

//Configuration

TRISB.0=0; //OUTPUT AT RB0

TRISB.1=0;

TRISB.3=0;

TRISB.1=0; // OUTPUT TRIG

TRISB.4=1;

do{

T0CON=0b10000000; // set TMR0 for 0.1 sec

TMR0H=-NC/256; TMR0L=-NC%256;TMR0IF=0;

BlinkAlive();

++T100m;

//Button Tasks

if(T100m>100)

{

T100m=0;

//TRIG passed 20us pulse

PORTB.1=1; //Output high for Trig

char i=20/3;

do{}while(--i);

PORTB.1=0;

//wait for ECHO

ECHO\_WAIT\_UP();

//Read TMR0H:TMR0L

t0.low8= TMR0L;

t0.high8=TMR0H;

ECHO\_WAIT\_DN();

t1.low8=TMR0L;

t1.high8=TMR0H;

//Time interval calculation

tp=t1-t0;

//timer calculation

TXhex(t0.high8);TXhex(t0.low8);TXchar('/');

TXhex(t1.high8);TXhex(t1.low8);

TXchar('/');

dmm=tp\*0.34;

WREG=84;multiply(tp.high8);

dmm.low8=PRODL;dmm.high8=PRODH;

multiply(tp.low8);

dmm.low8+=PRODH;

if(Carry)++dmm.high8;

arrypoint=&arry[0];

u16toASCII(arry,dmm);

Int16AsciiConvert(dmm);

\*arrypoint=13;

++arrypoint;

\*arrypoint=0;

arrypoint=&arry[0];

while(\*arrypoint){TXchar(\*arrypoint); ++arrypoint;};

TXASCII();

}

do{}while(!TMR0IF);

}while(1);

}

Task I:

Merge all components to your system (in hardware and software) and test the overall system function for the ECHO pulse durations {1ms, 5ms, 10ms, 15ms, 20ms, 25ms, 30ms, 40ms, always high}

