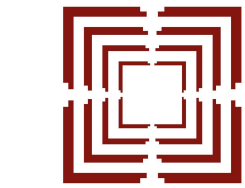


Ahmedabad University

Embedded Systems and Design

Lab 2 - Report

Section 1

Submitted to

Faculty: Anurag Lakhlani

Teaching Assistants: Yagnik Bhavsar, Harsh Patel, Pratik Rajpara

Date of Submission: 27th October 2021

Student Details

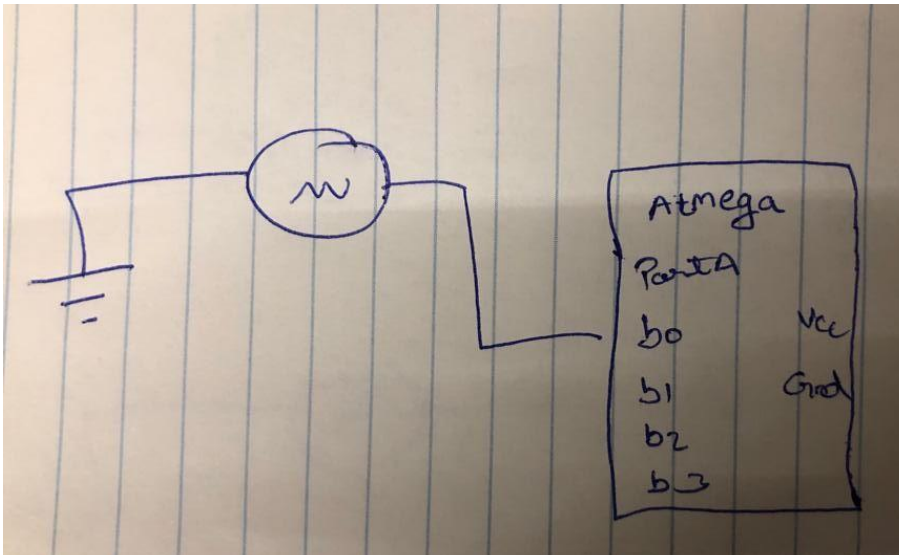
Roll No.	Name of the Student	Name of Program
AU1940065	Parth Shah	BTech (CSE)

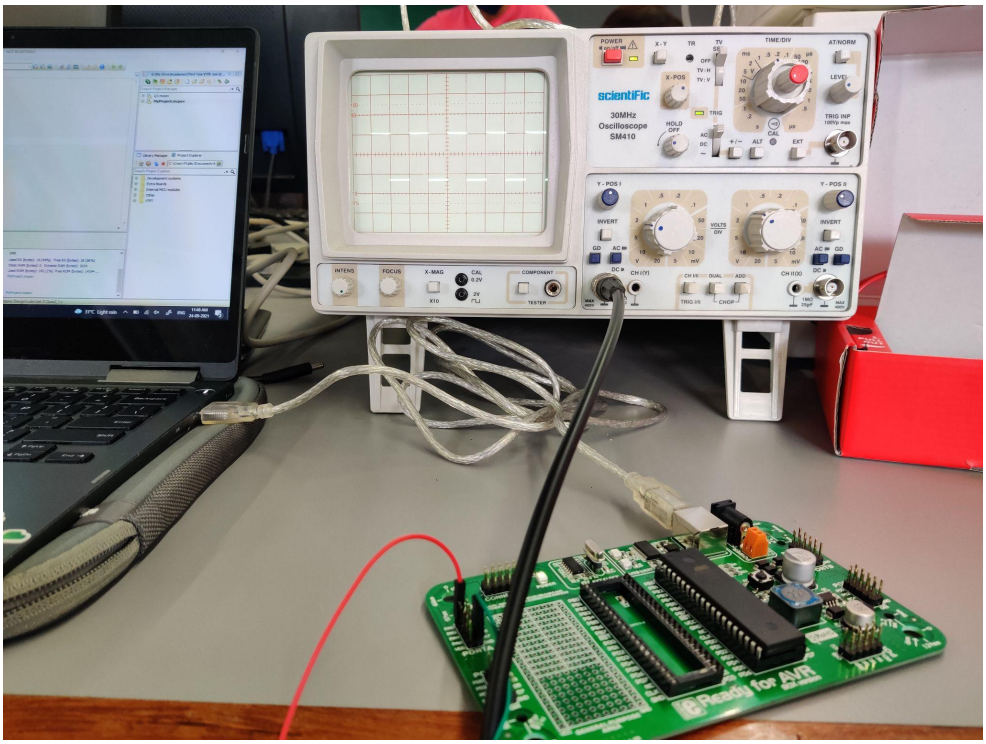
2021-2022 (Monsoon Semester)

Index

<u>Question 1:</u> Generate a square wave of 500 Hz (50% duty Cycle) by toggling the 0th pin of Port A. Observe this wave on CRO. *(Hint: Use delay_ms in-build function to create appropriate delay, toggle the port at 1ms (frequency = 500 Hz. So, the time delay for the whole wave is 2 ms).)*

Video | Code





```
1  void main() {
2       DDRA=0x01;
3       while(1){
4       PORTA=0x01;
5       Delay_ms(1);
6       PORTA=0x00;
7       Delay_ms(1);
8       }
9  }
```

In the first question, we Generate a square wave of 500 Hz (50% duty Cycle) by toggling the 0th pin of Port A and then observe this wave in CRO. We use the delay_ms in-build function to create appropriate delay, which is of 1 ms

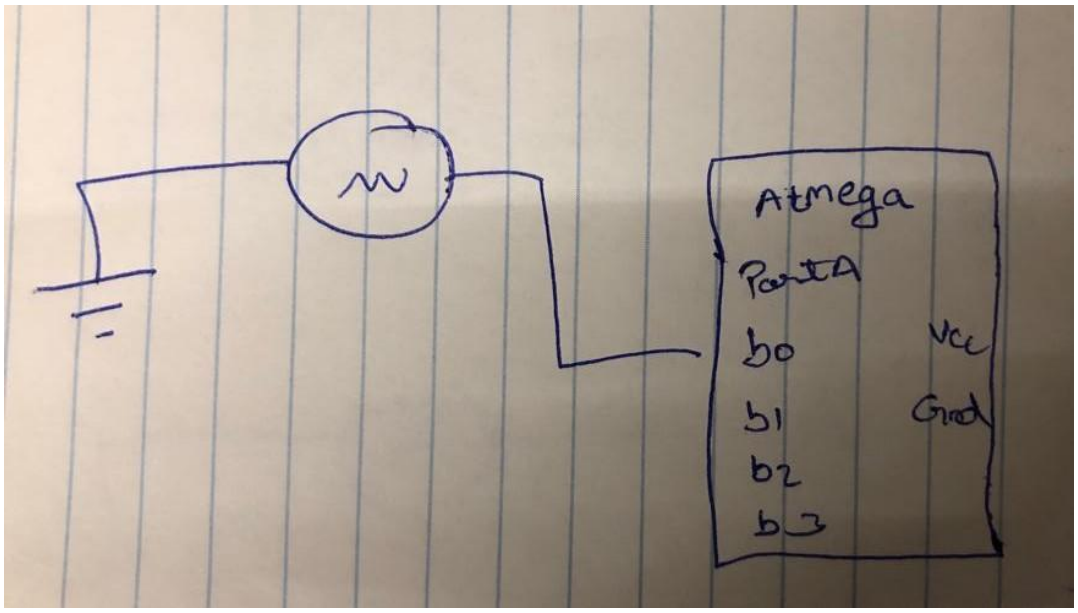
---

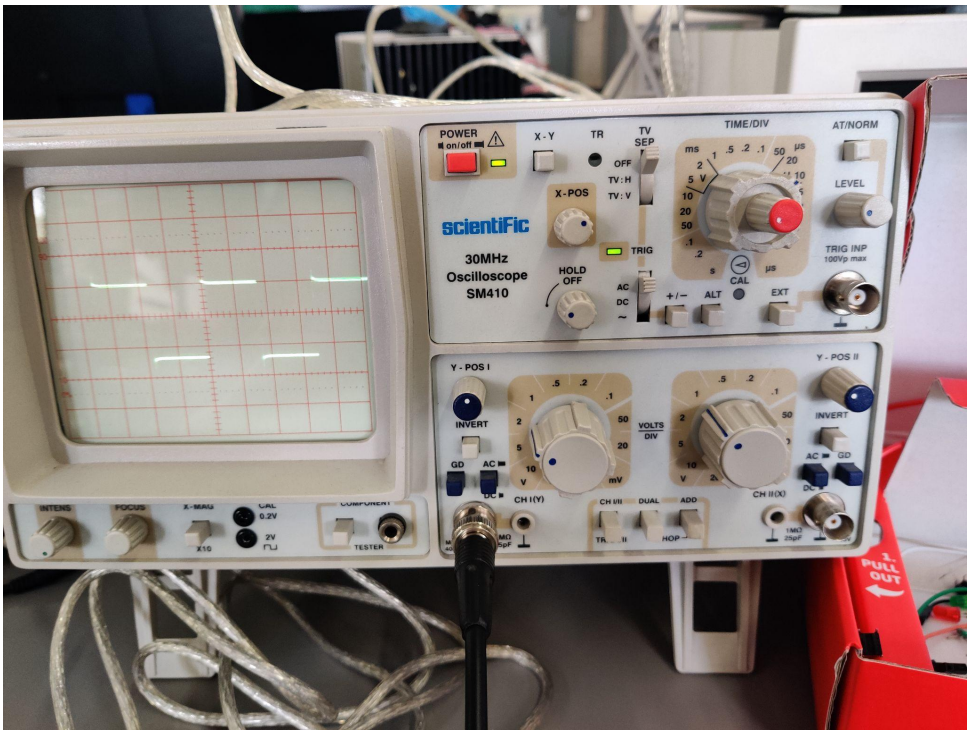
<u>Question 2:</u> Toggle a port pin upon stated conditions:

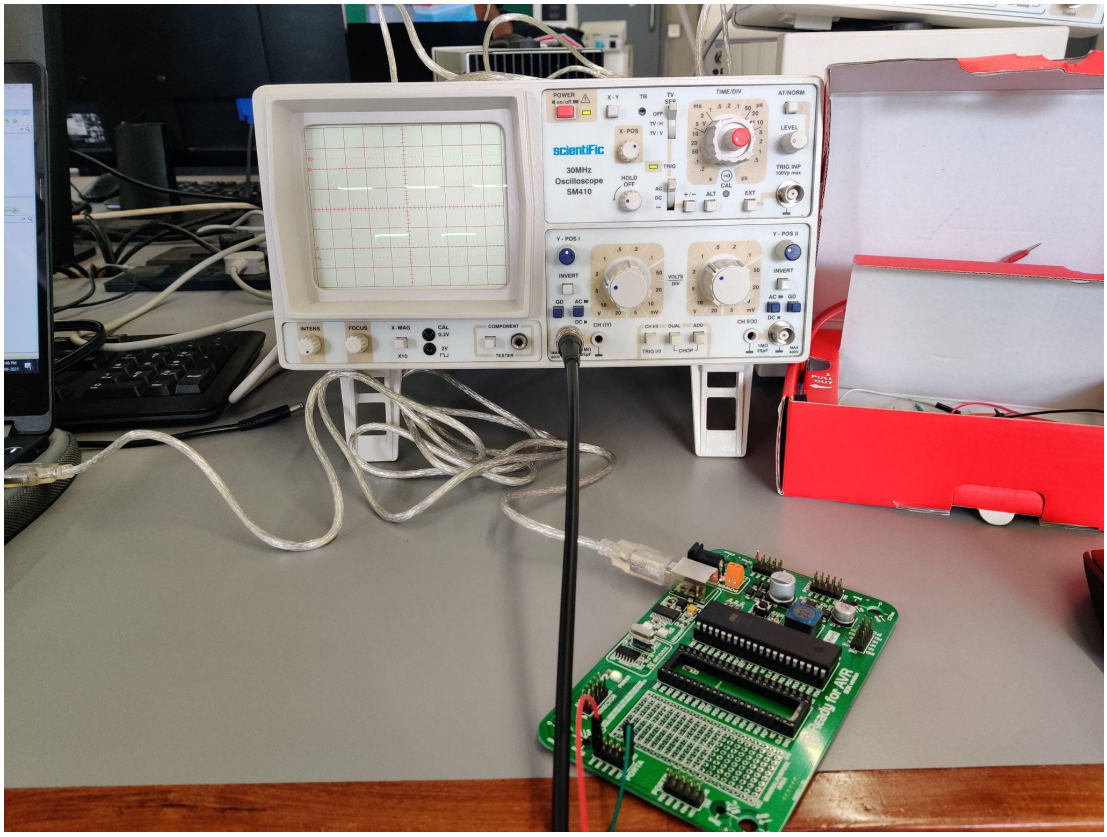
Calculate and verify the frequency for each case. You may write the values of these frequencies at the start of your program (in comment) for submission.

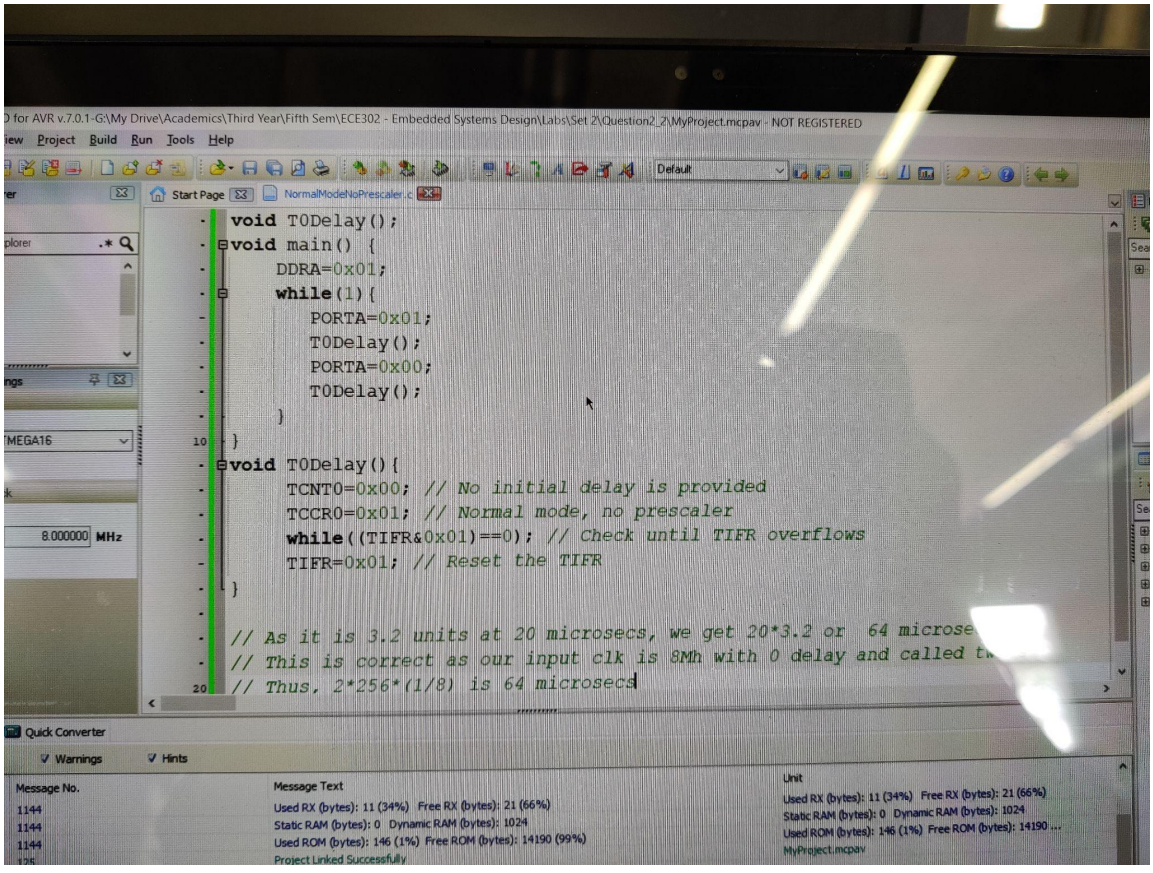
(A). Timer 0 Normal mode

Video | Code






scientific
30MHz
Oscilloscope
SM410
Ready for AVR

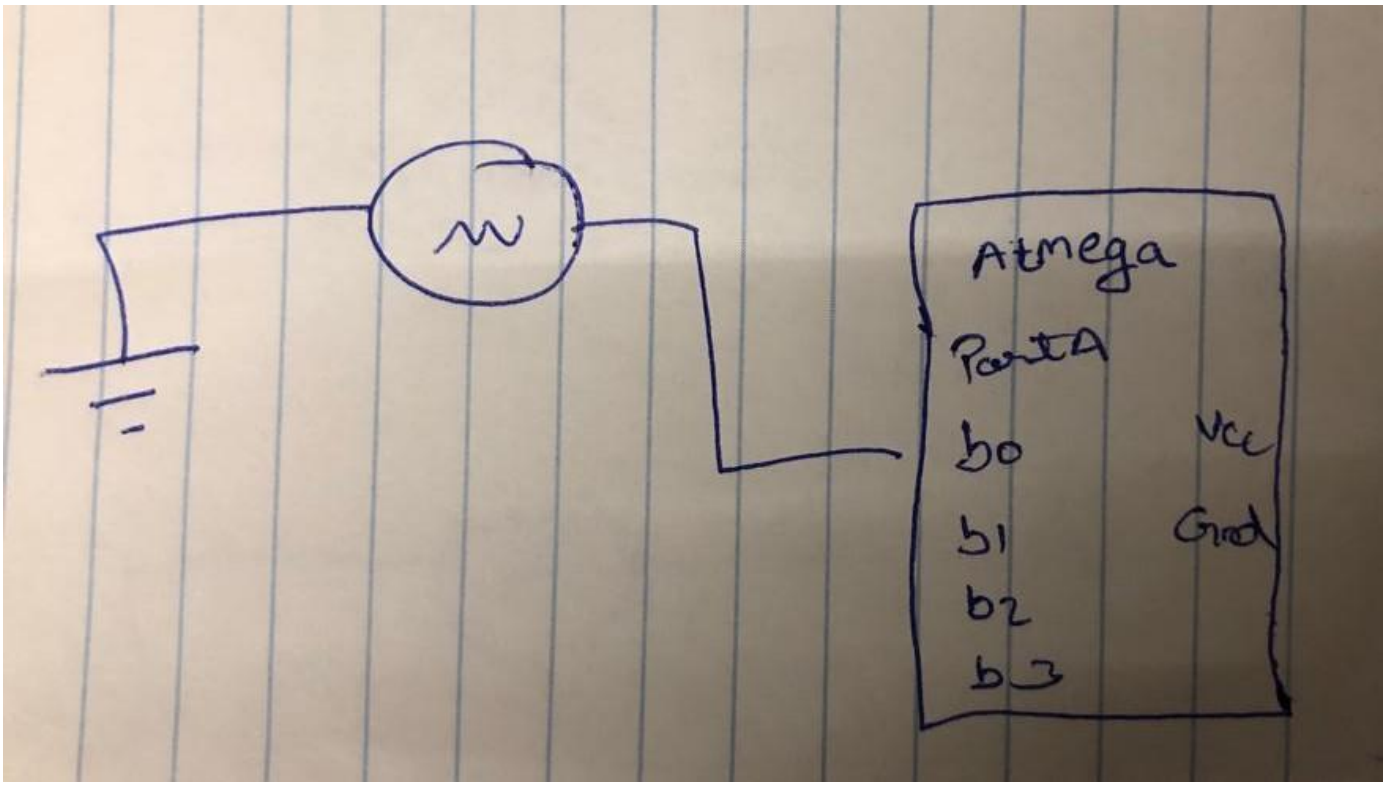

for AVR v.7.0.1-G:\My Drive\Academics\Third Year\Fifth Sem\ECE302 - Embedded Systems Design\Labs\Set 2\Question2_2\MyProject.mcpav - NOT REGISTERED
Project Build Run Tools Help
Start Page
NormalModeNoPrescaler.c
void T0Delay();
void main() {
DDRA=0x01;
while(1){
PORTA=0x01;
T0Delay();
PORTA=0x00;
T0Delay();
}
}
void T0Delay(){
TCNT0=0x00; // No initial delay is provided
TCCR0=0x01; // Normal mode, no prescaler
while((TIFR&0x01)==0); // Check until TIFR overflows
TIFR=0x01; // Reset the TIFR
}
// As it is 3.2 units at 20 microsecs, we get 20*3.2 or 64 microse
// This is correct as our input clk is 8Mh with 0 delay and called t
// Thus, 2*256*(1/8) is 64 microsecs
8.000000 MHz
Quick Converter
Warnings
Hints
Message No.
Message Text
Used RX (bytes): 11 (34%) Free RX (bytes): 21 (66%)
Static RAM (bytes): 0 Dynamic RAM (bytes): 1024
Used ROM (bytes): 146 (1%) Free ROM (bytes): 14190 (99%)
Project Linked Successfully
Unit
Used RX (bytes): 11 (34%) Free RX (bytes): 21 (66%)
Static RAM (bytes): 0 Dynamic RAM (bytes): 1024
Used ROM (bytes): 146 (1%) Free ROM (bytes): 14190 ...
MyProject.mcpav

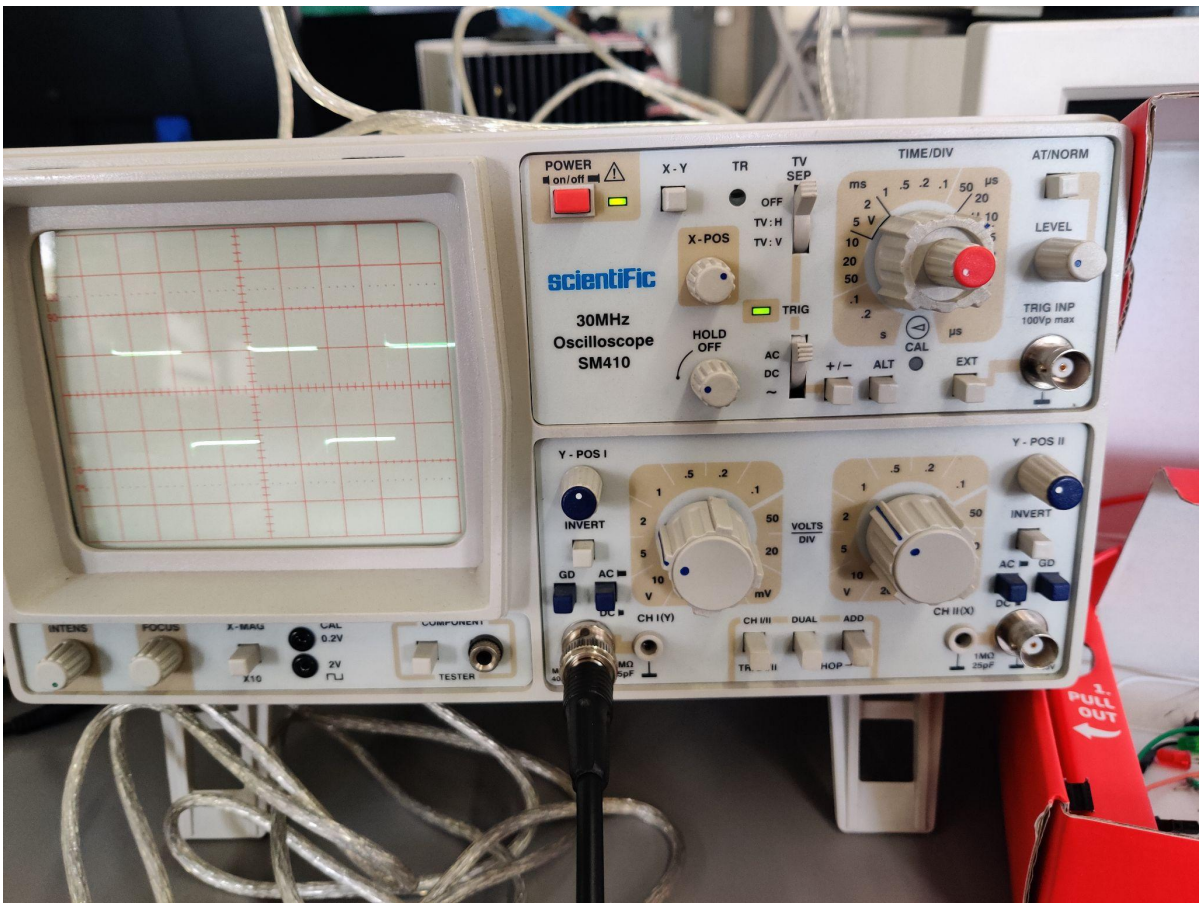
```
void T0Delay();
void main() {
    DDRA=0x01;
    while(1){
       PORTA=0x01;
       T0Delay();
       PORTA=0x00;
       T0Delay();
    }
}
void T0Delay(){
     TCNT0=0x00; // No initial delay is provided
     TCCR0=0x01; // Normal mode, no prescaler
     while((TIFR&0x01)==0); // Check until TIFR overflows
     TIFR=0x01; // Reset the TIFR
}

/*
 As it is 3.2 units at 20 microsecs, we get 20*3.2 or  64 microsecs.
 This is correct as our input clk is 8Mh with 0 delay and called twice
 Thus, 2*256*(1/8) is 64 microsecs
 */
```

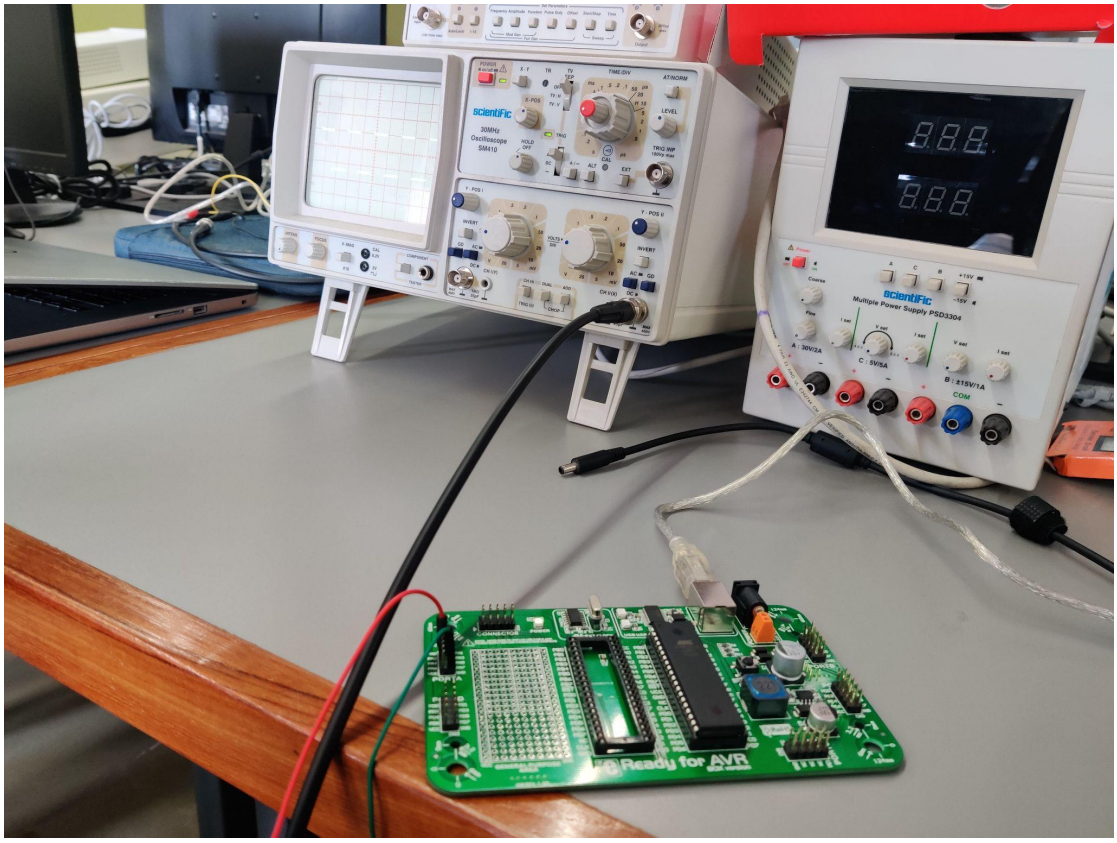
Here, we needed to generate a square wave by toggling PORTA.b0 using the timer0 circuit in normal mode. The connections are similar to the previous question. Furthermore, as we can see in the code below, it is evident that the timer counter register runs for a full cycle i.e from 0 to 255, and rolls over at 256. This means that the delay is 64 microseconds. This means that the theoretical frequency is 15.625 kHz. Now as we can see in the image above, the time delay for 2 blocks in CRO is approximately 20 microseconds each which means 40 microseconds. This means the time delay for the whole square wave is 80 microseconds. This means that the calculated frequency is 12.5 kHz.

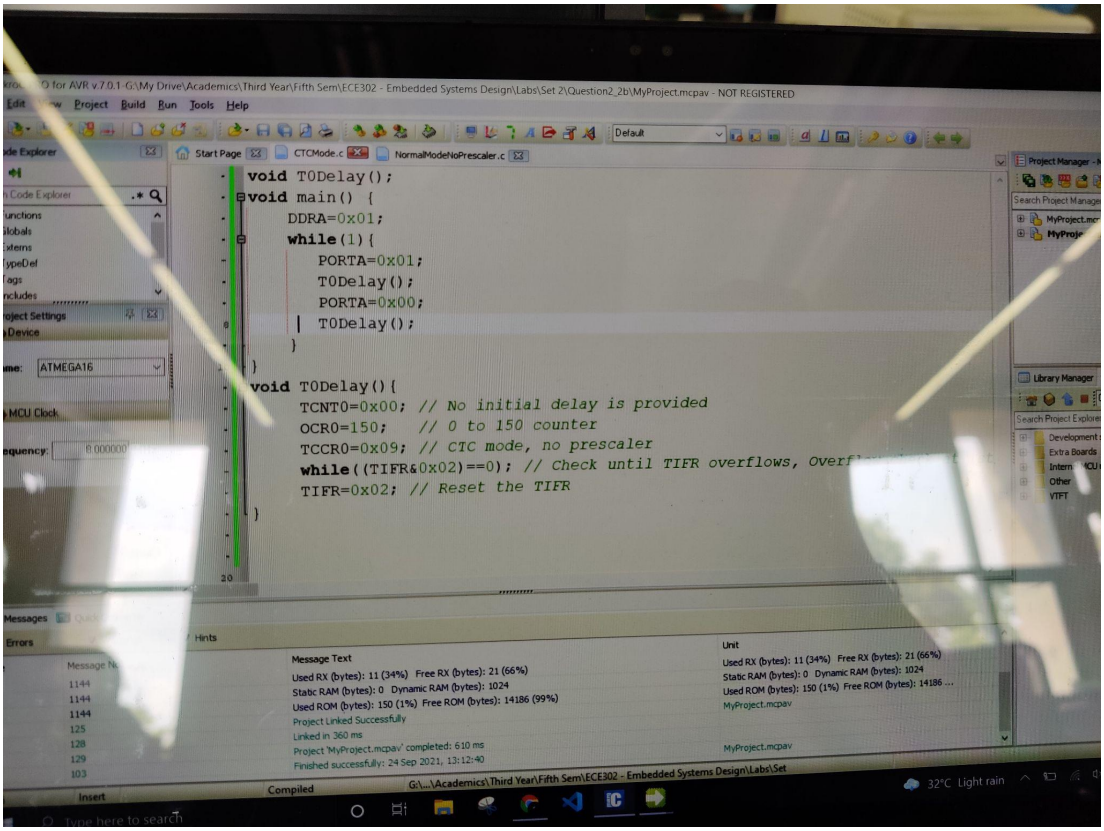
(B). Timer 0 CTC mode

Video | Code




POWER
on/off
X-Y
TR
TV SEP
OFF
TV:H
TV:V
TIME/DIV
AT/NORM
LEVEL
scientific
30MHz
Oscilloscope
SM410
X-POS
TRIG
HOLD OFF
AC
DC
+/−
ALT
CAL
EXT
TRIG INP
100Vp max
Y - POS I
INVERT
VOLTS DIV
CH I (Y)
Y - POS II
INVERT
CH II (X)
DUAL
ADD
INTENS
FOCUS
X-MAG
X10
0.2V
2V
TESTER
1. PULL OUT


scientific
Multiple Power Supply PSD3304
Ready for AVR


Start Page
CTCMode.c
NormalModeNoPrescaler.c
void T0Delay();
void main() {
DDRA=0x01;
while(1){
PORTA=0x01;
T0Delay();
PORTA=0x00;
T0Delay();
}
}
void T0Delay(){
TCNT0=0x00; // No initial delay is provided
OCR0=150; // 0 to 150 counter
TCCR0=0x09; // CTC mode, no prescaler
while((TIFR&0x02)==0); // Check until TIFR overflows,
TIFR=0x02; // Reset the TIFR
}
ATMEGA16
Project Linked Successfully
Compiled

```
void T0Delay();
void main() {
    DDRA=0x01;
    while(1){
       PORTA=0x01;
       T0Delay();
       PORTA=0x00;
       T0Delay();
    }
}
void T0Delay(){
     TCNT0=0x00; // No initial delay is provided
     OCR0=150;   // 0 to 150 counter
     TCCR0=0x09; // CTC mode, no prescaler
     while((TIFR&0x02)==0); // Check until TIFR overflows
                            //Overflow check at 1st bit and not 0th
     TIFR=0x02; // Reset the TIFR
}

/*
 2*150*(1/8) is thereotical = 37.5 microsecs
 Real is 2.1*20 microsecs or 42 microsecs
 Real freq is 1/42 or 0.0238 Mhz or 23.8 Khz and theoretical is 0.0266 Mhz or 26.67 Khz
*/
```

Here the timer0 counter runs from 0 to 150 and rolls over. For this, the theoretical time delay is 37.5 microseconds and the theoretical frequency is 26.67 kHz.

As visible in the photo, 1 block of square wave has a 20 seconds delay and thus the whole wave has a 40 seconds delay. Thus, the calculated frequency becomes 25 kHz.

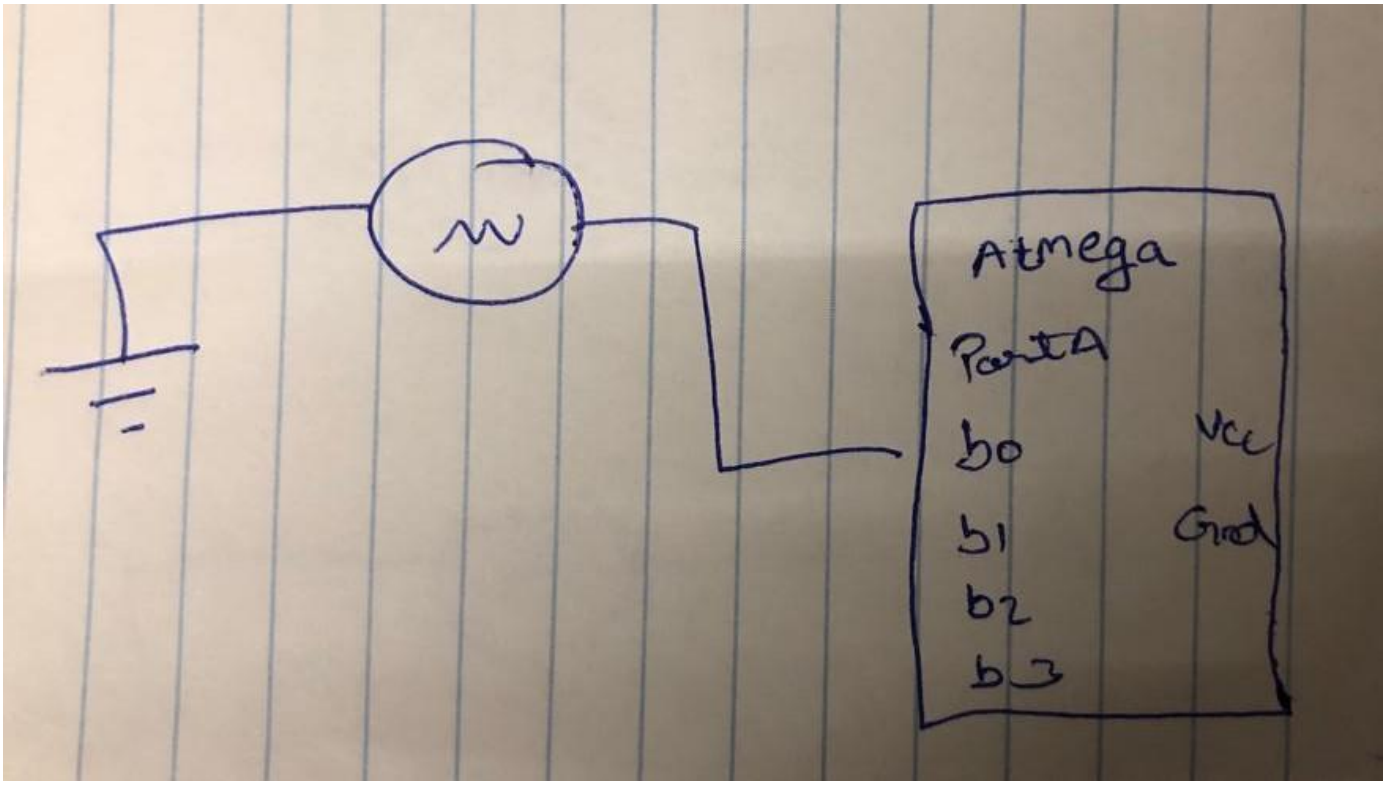
<u>Question 3:</u> Generate and verify square wave of the following frequency at pin

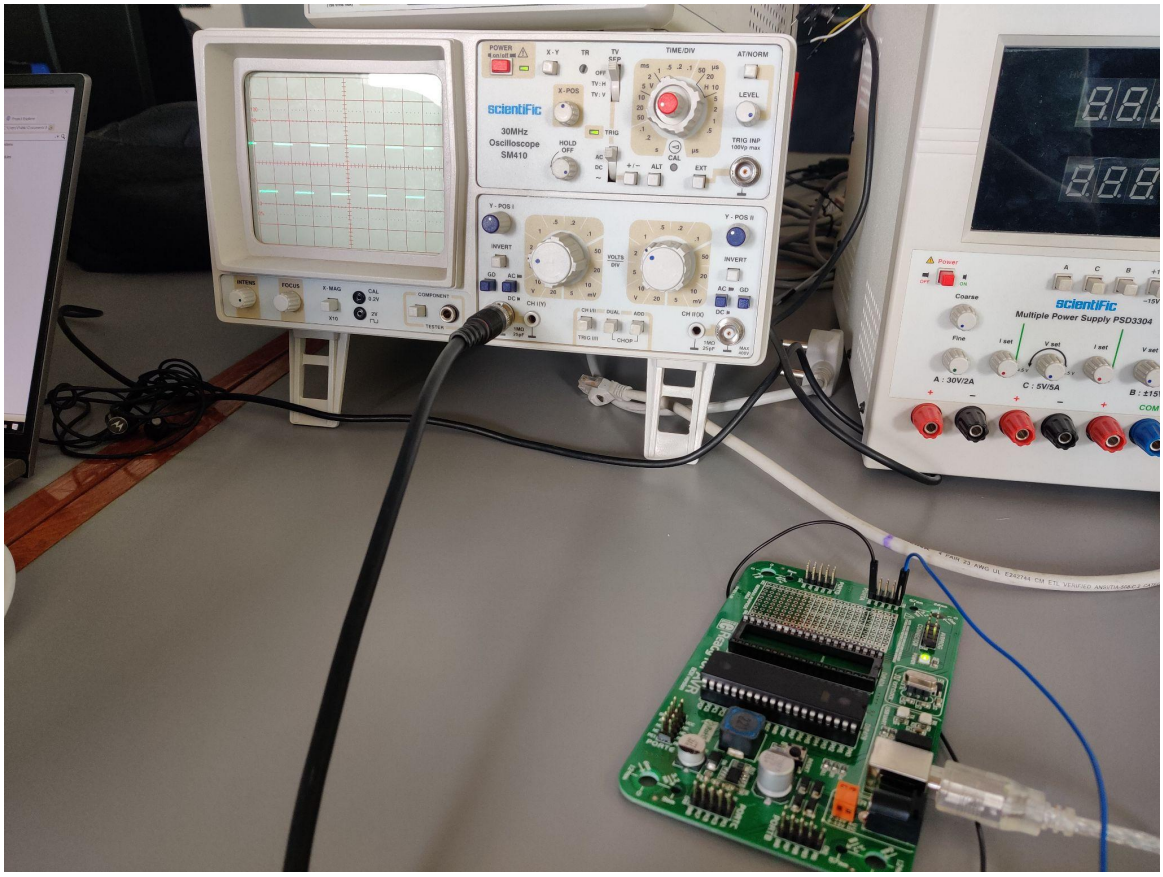
PORTA.B0:

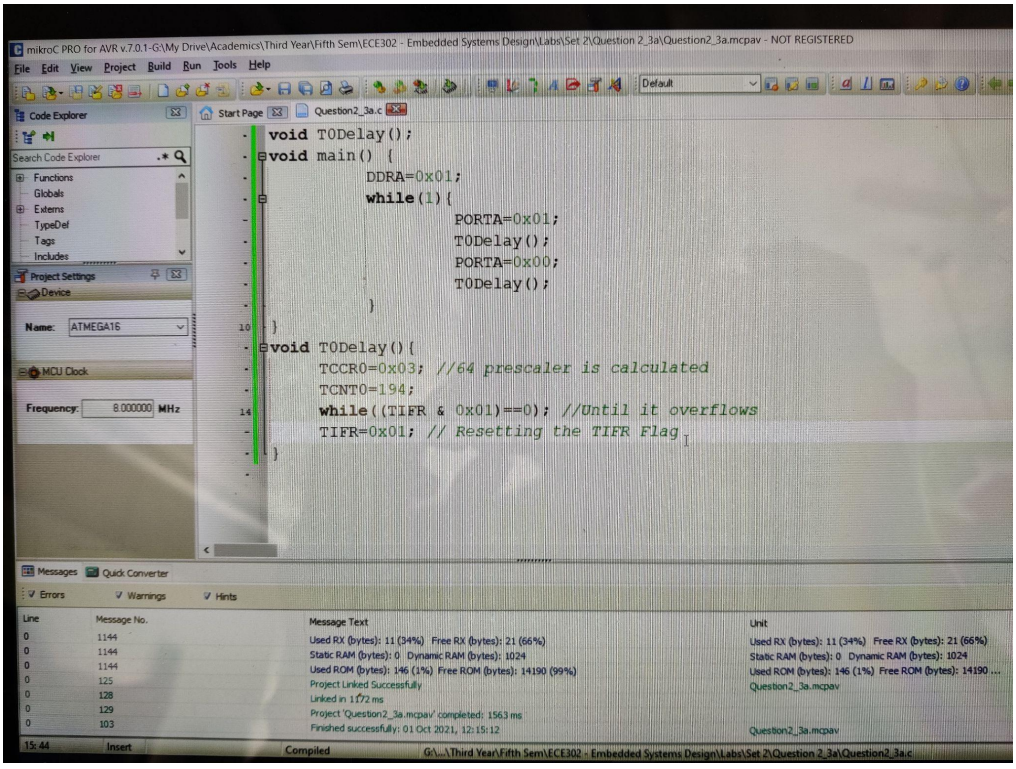
(A) 1 kHz

(Hint: Setup Pre-scaler)

Video | Code








```
void T0Delay();
void main() {
        DDRA=0x01;
        while(1){
                PORTA=0x01;
                T0Delay();
                PORTA=0x00;
                T0Delay();
        }
}
void T0Delay(){
    TCCR0=0x03; //64 prescaler is calculated
    TCNT0=194;
    while((TIFR & 0x01)==0); //Until it overflows
    TIFR=0x01; // Resetting the TIFR Flag
}
```

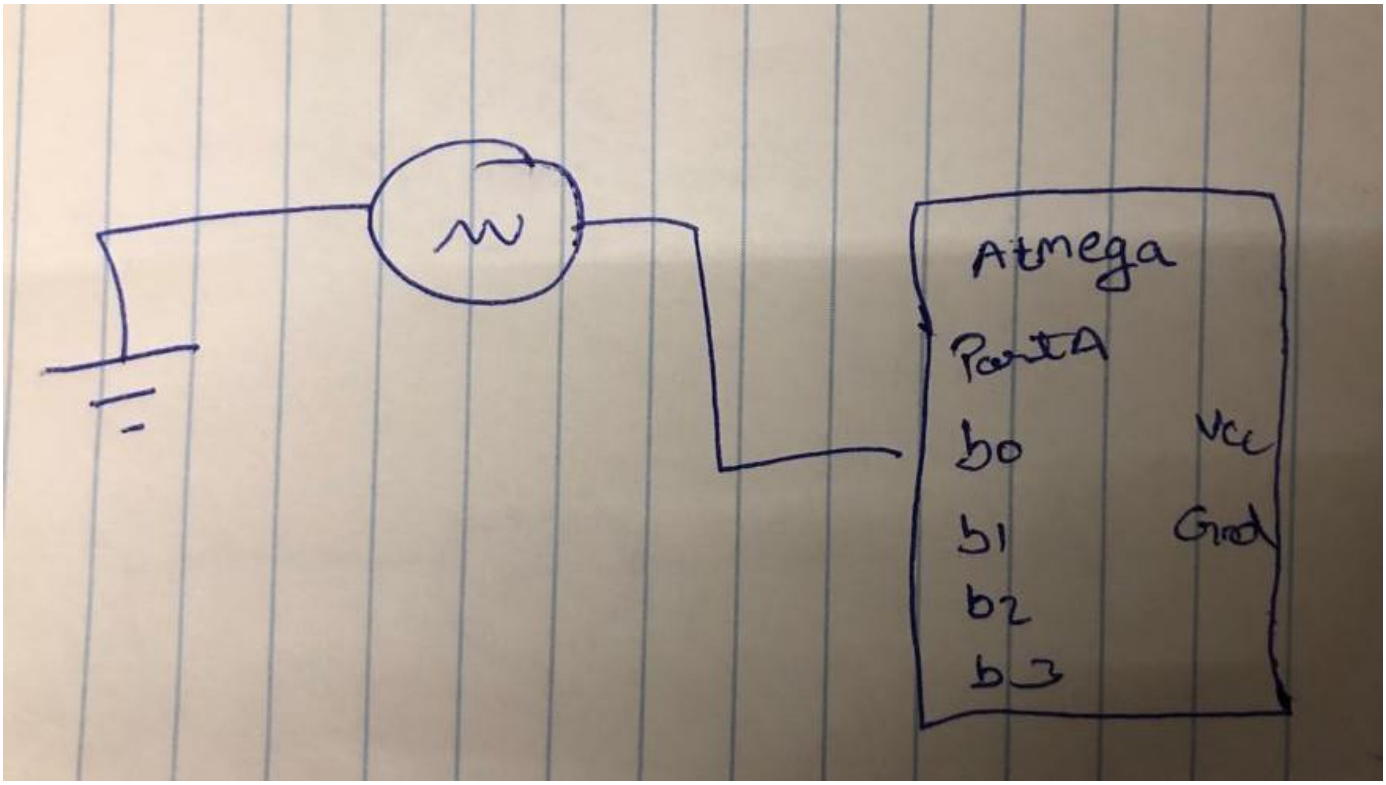
```
void T0Delay();
void main() {
        DDRA=0x01;
        while(1){
                PORTA=0x01;
                T0Delay();
                PORTA=0x00;
                T0Delay();
        }
}
void T0Delay(){
     TCCR0=0x03; //64 prescaler is calculated
     TCNT0=194;
     while((TIFR & 0x01)==0); //Until it overflows
     TIFR=0x01; // Resetting the TIFR Flag
}
```

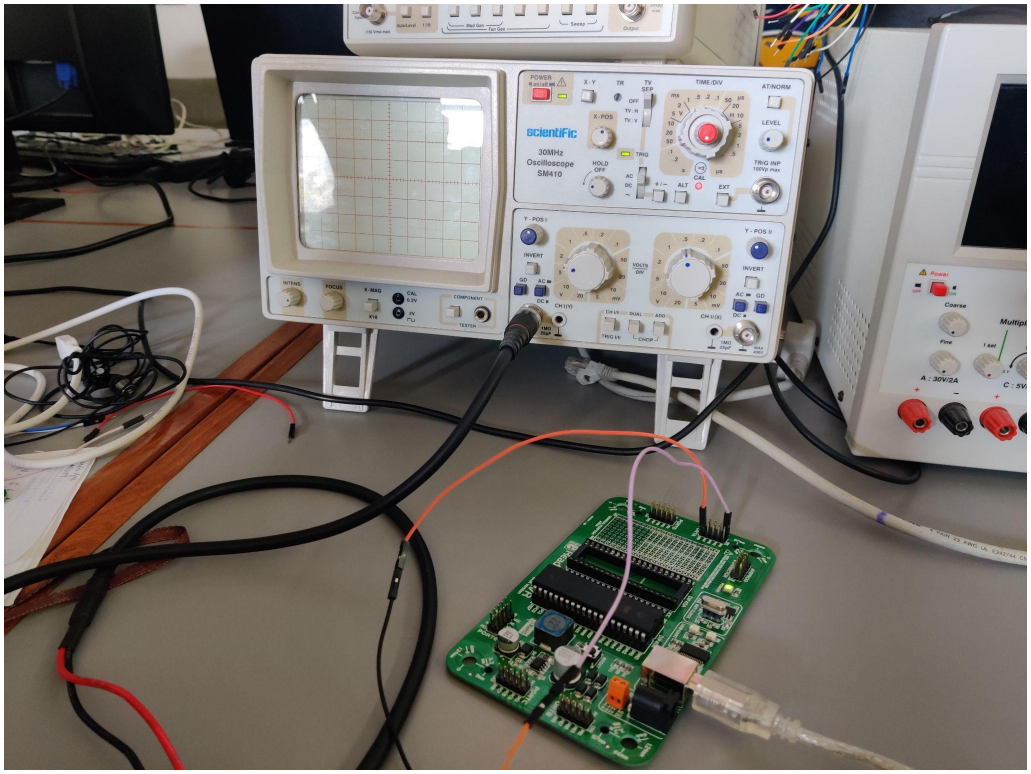
In this experiment, we had to use a timer0 circuit with an appropriate prescaler. Creating a square wave of frequency 1Khz lies beyond the range of no prescaler, thus we had to use CLK/64 prescaler. When the program was loaded into the microcontroller, it showed the result stated above. As seen in the image above, the delay for half part of the square wave is 0.5ms and thus the delay for the whole wave becomes 1ms which is the required delay.

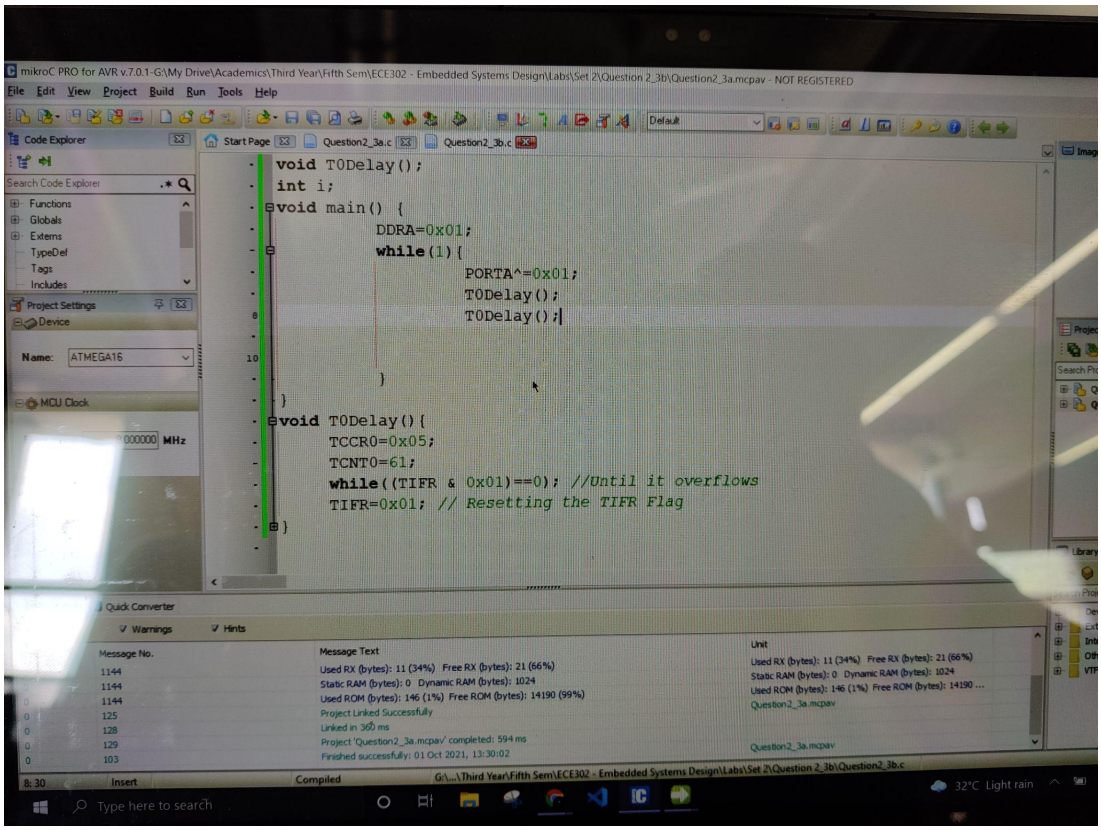
(B)10 Hz

(Hint: If Pre-Scaler is not sufficient to generate the required delay, one could call the same delay multiple times)

Video | Code




scientific
30MHz
Oscilloscope
SM410

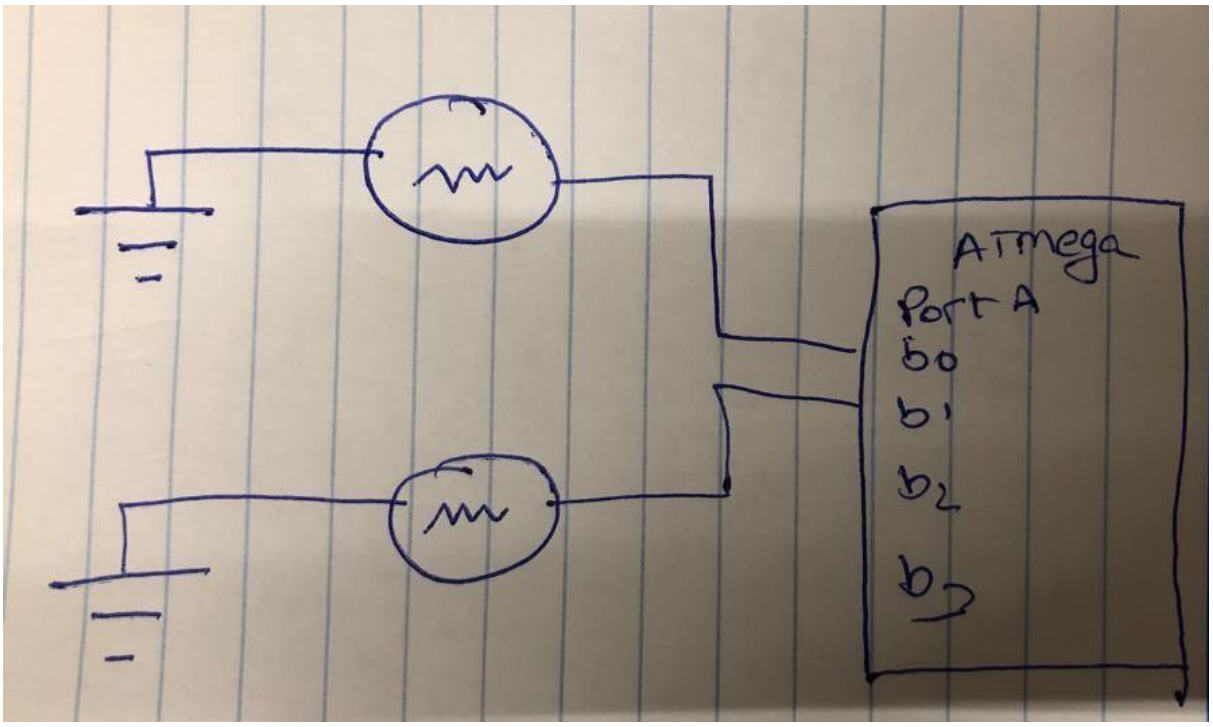

mikroC PRO for AVR v.7.0.1-G:\My Drive\Academics\Third Year\Fifth Sem\ECE302 - Embedded Systems Design\Labs\Set 2\Question 2_3b\Question2_3a.mcpav - NOT REGISTERED
File Edit View Project Build Run Tools Help
Code Explorer
Start Page
Question2_3a.c
Question2_3b.c
void T0Delay();
int i;
void main() {
DDRA=0x01;
while(1){
PORTA^=0x01;
T0Delay();
T0Delay();
}
}
void T0Delay(){
TCCR0=0x05;
TCNT0=61;
while((TIFR & 0x01)==0); //Until it overflows
TIFR=0x01; // Resetting the TIFR Flag
}
Project Settings
Device
Name: ATMEGA16
MCU Clock
Quick Converter
Warnings
Hints
Message No.
Message Text
Used RX (bytes): 11 (34%) Free RX (bytes): 21 (66%)
Static RAM (bytes): 0 Dynamic RAM (bytes): 1024
Used ROM (bytes): 146 (1%) Free ROM (bytes): 14190 (99%)
Project Linked Successfully
Linked in 360 ms
Project 'Question2_3a.mcpav' completed: 594 ms
Finished successfully: 01 Oct 2021, 13:30:02
Compiled
Type here to search
32°C Light rain

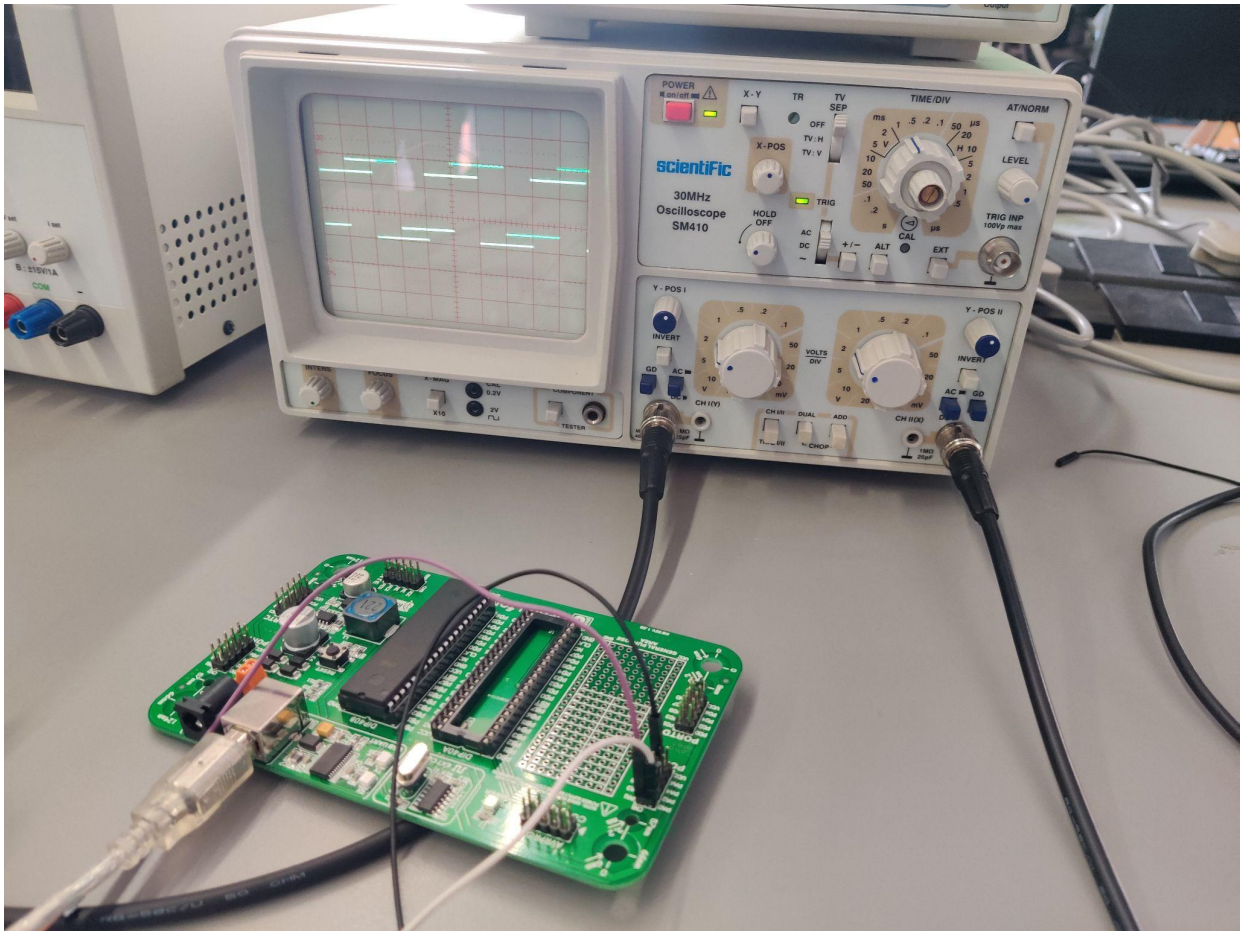
```
void T0Delay();
int i;
void main() {
          DDRA=0x01;
          while(1){
                    PORTA^=0x01;
                    T0Delay();
                    T0Delay();
          }
}
void T0Delay(){
     TCCR0=0x05;
     TCNT0=61;
     while((TIFR & 0x01)==0); //Until it overflows
     TIFR=0x01; // Resetting the TIFR Flag
}
```

Here, we are asked to generate a square wave of 10Hz frequency which implies a delay of 0.1 seconds to the whole wave and 0.05 seconds to the half-wave. For this, even if we used the highest prescaler of CLK/1024, it was not possible to obtain the required time delay. Thus, we had to use multiple time delays. So we obtained the time delay of 0.025 seconds using CLK/1024 prescaler and applied it 2 times making the overall delay for half-wave to be 0.05 seconds. Even though the image is not clear we can see that the delay for half-wave is actually 0.5 seconds.

Question 4: Generate two square waves with a frequency of 1 kHz that are 90 phases shifted.

Video | Code




POWER
X-Y
TR
TV SEP
TIME/DIV
AT/NORM
scientific
30MHz
Oscilloscope
SM410
X-POS
LEVEL
TRIG
HOLD OFF
TRIG INP
CAL
EXT
Y-POS I
Y-POS II
INVERT
VOLTS DIV
CH I(Y)
CH II(X)
DUAL
ADD
CHOP

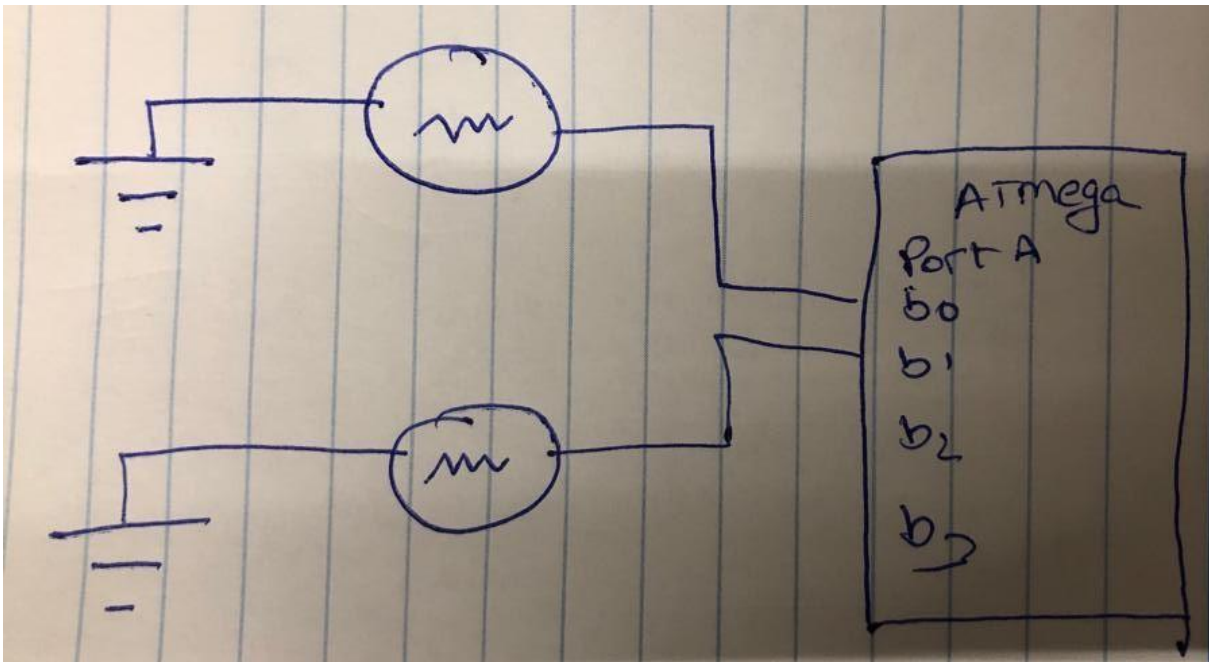
```c
// Laboratory 1
// Question 4
// Parth Shah AU1940065

#include <avr/io.h>

// Declaring global variables
int i, j;
void main()
{
    DDRA = 0xF0;
    while (1)
    {
        // Checks if the b4 pin is pressed
        if (PINA.b4 == 1)
        {
            // Loop from 0 to 15 Binary Display
            for (i = 0; i < 16; i++)
            {
                PORTA = i;
                Delay_ms(1111);
                if (PINA.b4 == 0)
                {
                    // Resetting PORTA to 0
                    PORTA = 0x00;
                    break;
                }
            }
        }
    }
}
```

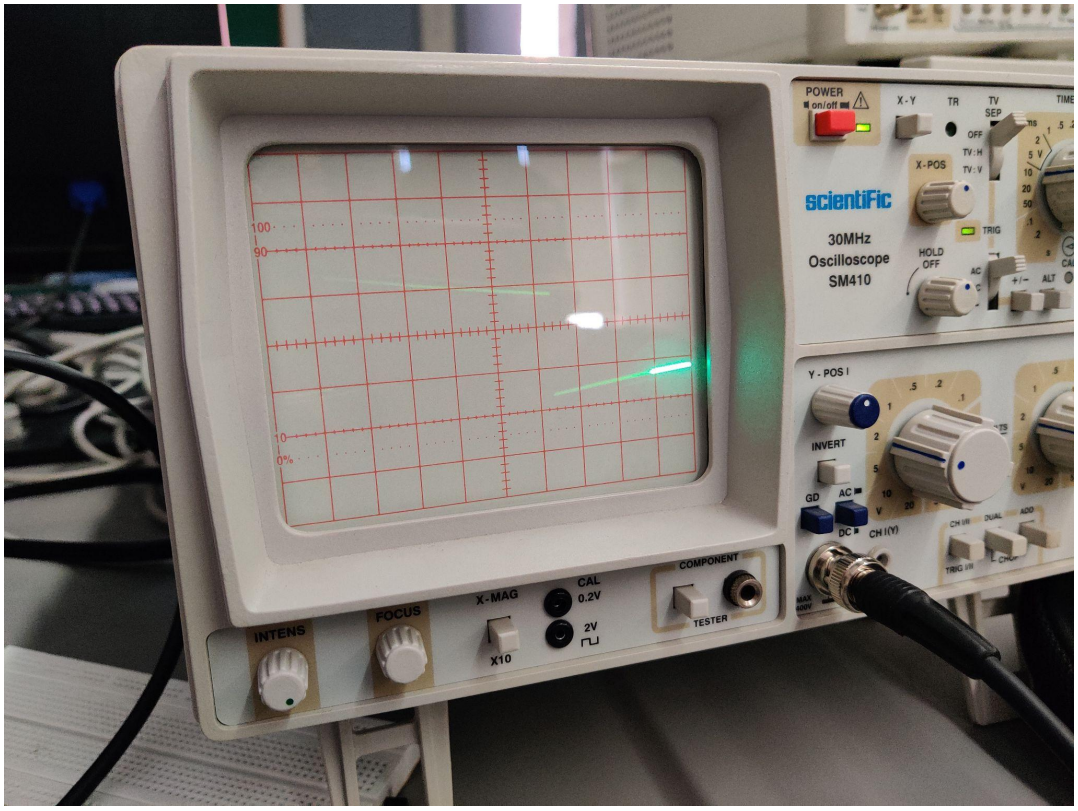
The question required to generate two square waves of frequency 1Khz that are shifted by 90 degrees. For this, we added a delay before toggling the PORTA.b0. Thus, it gave a proper 90 degrees shift. The time delay function remains the same as that in experiment 3a. The connections are quite simple with a modification that we need two CRO cables and operate the CRO in dual mode to get the desired result. As it can be seen in the image above, the 90 degrees phase shifting is clear.

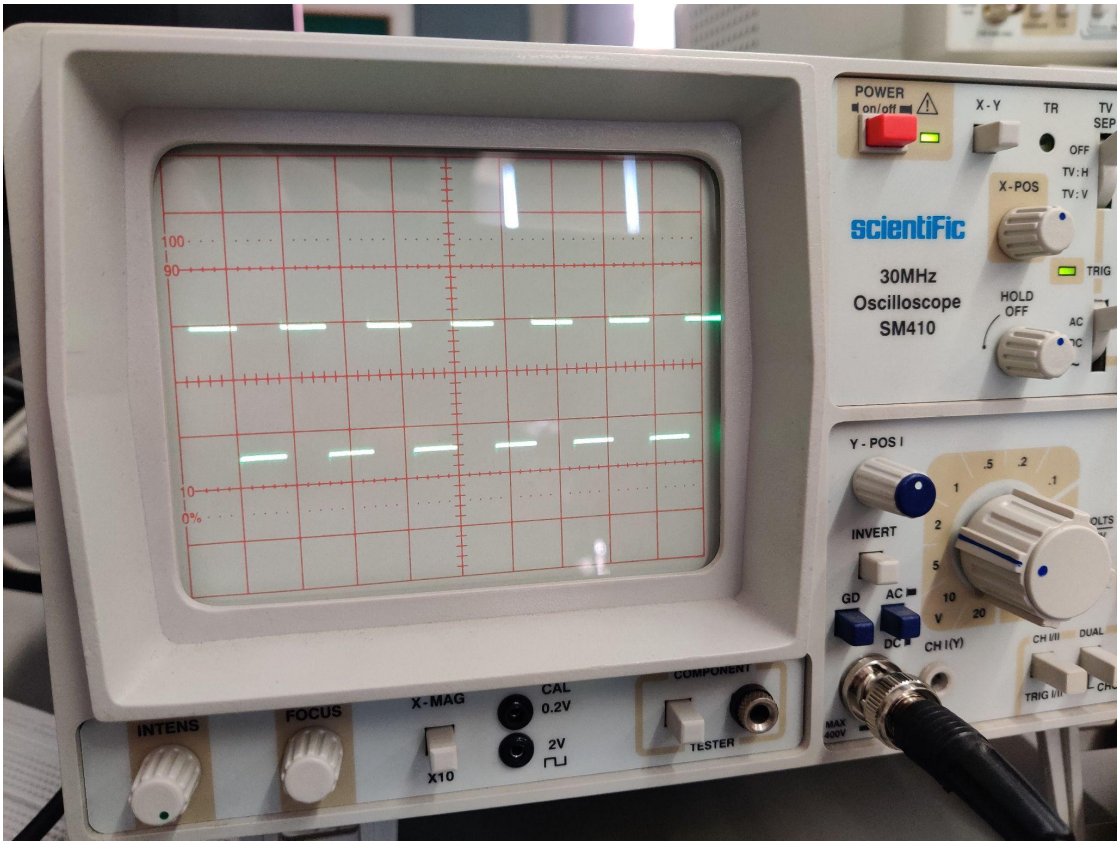
<u>Question 5:</u> (Extra Problem)

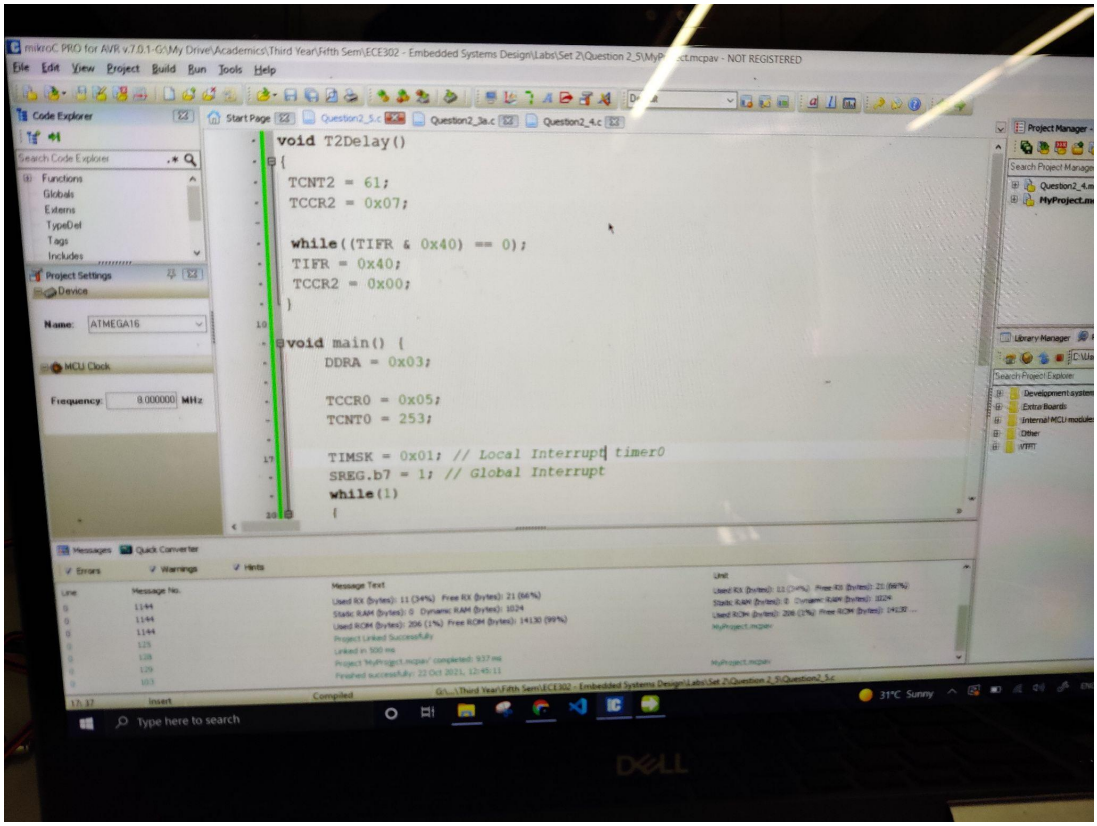
Generate both square waves of problem-3 simultaneously at two different port pins. (Hint: You may use Timer 0 and Timer 2 together)

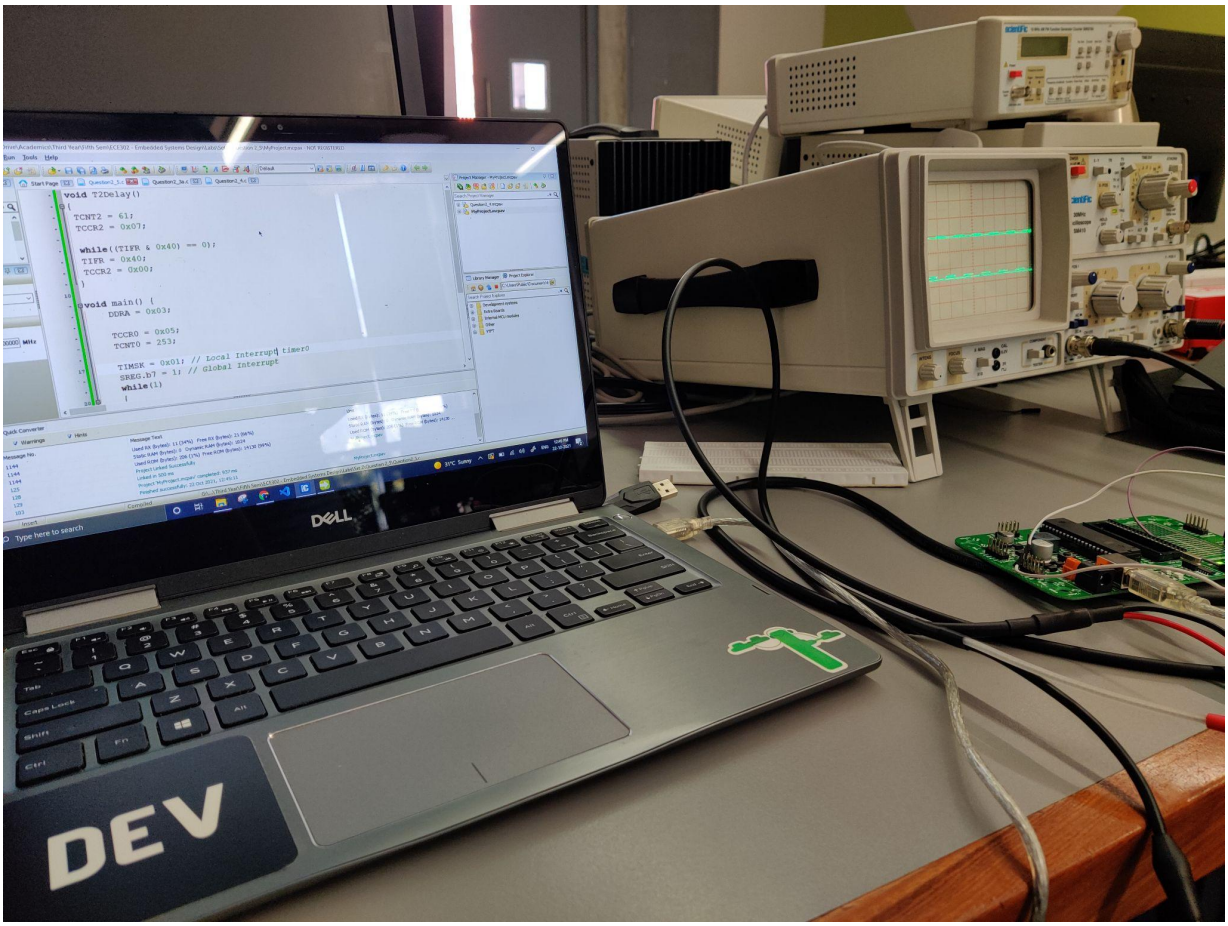
Video | Code




POWER
on/off
X-Y
TR
TV SEP
OFF
TV:H
TV:V
X-POS
scientific
30MHz
Oscilloscope
SM410
TRIG
HOLD OFF
AC
Y-POS I
INVERT
GD
AC
DC
CH I(Y)
DUAL
ADD
INTENS
FOCUS
X-MAG
X10
CAL
0.2V
2V
COMPONENT
TESTER


POWER
on/off
X-Y
TR
TV SEP
OFF
TV:H
TV:V
X-POS
scientific
30MHz
Oscilloscope
SM410
TRIG
HOLD OFF
AC
Y-POS I
INVERT
GD
AC
DC
CH I(Y)
DUAL
INTENS
FOCUS
X-MAG
X10
CAL
0.2V
2V
TESTER


void T2Delay()
{
TCNT2 = 61;
TCCR2 = 0x07;
while((TIFR & 0x40) == 0);
TIFR = 0x40;
TCCR2 = 0x00;
}
void main() {
DDRA = 0x03;
TCCR0 = 0x05;
TCNT0 = 253;
TIMSK = 0x01; // Local Interrupt timer0
SREG.b7 = 1; // Global Interrupt
while(1)
{


DEV

```
void T2Delay()
{
 TCNT2 = 61;
 TCCR2 = 0x07;

 while((TIFR & 0x40) == 0);
 TIFR = 0x40;
 TCCR2 = 0x00;
}

void main() {
    DDRA = 0x03;

    TCCR0 = 0x05;
    TCNT0 = 253;

    TIMSK = 0x01; // Local Interrupt timer0
    SREG.b7 = 1; // Global Interrupt
    while(1)
    {
     PORTA ^= 0x01;
     T2Delay();
     T2Delay();
    }
}

void timer0_ovf_isr(void) org 0x012
{
 PORTA ^= 0x02;
 TCNT0=253;
}
```

For the above experiment, we just had to replicate the results that we obtained in question 3 simultaneously at two different port pins. Thus we used PORTA.b0 and PORTA.b1. Furthermore, it was required to use an interrupt because two delays are two be applied using timer0 and timer2. Thus, timer0 is used in interrupt mode and timer2 is used in regular mode. Thus, we obtain the above result which is not clear because the time delay is very large.