

# ECE 376 - Homework #9

Timer 0/1/2/3 Interrupts - Due Wednesday, April 3rd

1) Write a C routine using Timer0 interrupts to measure time to 100ns. Using this routine, determine how long a the following operations in C take:

a) Integer operations

```
int A, B, C;  
A = 5;  
B = 7;  
  
C = 2*A + 3*B + 4;
```

**time 197 clocks (19.7us)**

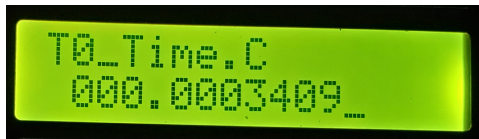


T0\_Time.C  
000.0000197\_

b) Floating Point Operations

```
float A, B, C;  
A = 3.14159;  
B = 2.71718;  
  
C = 2.1*A + 3.7*B + 4.16;
```

**time = 3409 clocks (340.9us)**



T0\_Time.C  
000.0003409\_

c) The time it takes you to press and release RB0 ten times

```
TRISB = 0xFF;  
  
for(i=0; i<10; i++) { // start  
    while(!RB0);  
    while(RB0);  
} // end
```

**time = 9,111,380 clocks (911.1380ms)**



T0\_Time.C  
000.9111380\_

2) Write a C routine using Timer0 / Timer1 / Timer2 / Timer3 interrupts to play 4 notes at the same time when you press button RB0.. RB3 at the same time (each note plays if its input button is pressed)

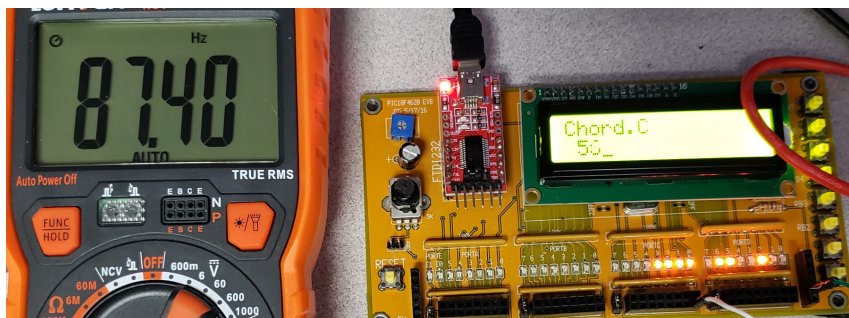
Input Pin	RB0	RB1	RB2	RB3
Output Pin	RC0	RC1	RC2	RC3
Note	F2	G2	A2	B2
Frequency (Hz)	87.307 Hz	97.999 Hz	110.000 Hz	123.471 Hz
Interrupt	Timer0	Timer1	Timer2	Timer3
N	57,269.18	51,020.93	45,454.54 (A = 12, B = 237, C = 16)	40,495.33
Measured Frequency	87.40	98.11	110.0	123.6
Error (%)	+0.10645%	+0.1133%	0%	+0.1045%

T2CON = 0x5F

7	6	5	4	3	2	1	0
0	1	0	1	1	1	1	1
A = 12				C = 16			

Interrupt Service Routine:

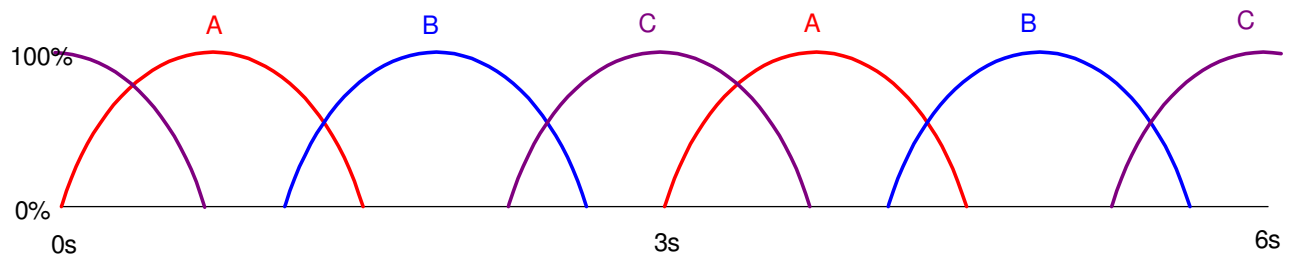
```
void interrupt IntServe(void)
{
    if (TMR0IF) {
        TMR0 = -57269 + 25;
        RC0 = !RC0;
        TMR0IF = 0;
    }
    if (TMR1IF) {
        TMR1 = -51020 + 32;
        RC1 = !RC1;
        TMR1IF = 0;
    }
    if (TMR2IF) {
        RC2 = !RC2;
        TMR2IF = 0;
    }
    if (TMR3IF) {
        TMR3 = -40495 + 41;
        RC3 = !RC3;
        TMR3IF = 0;
    }
}
```



## Three-Phase Sine Wave

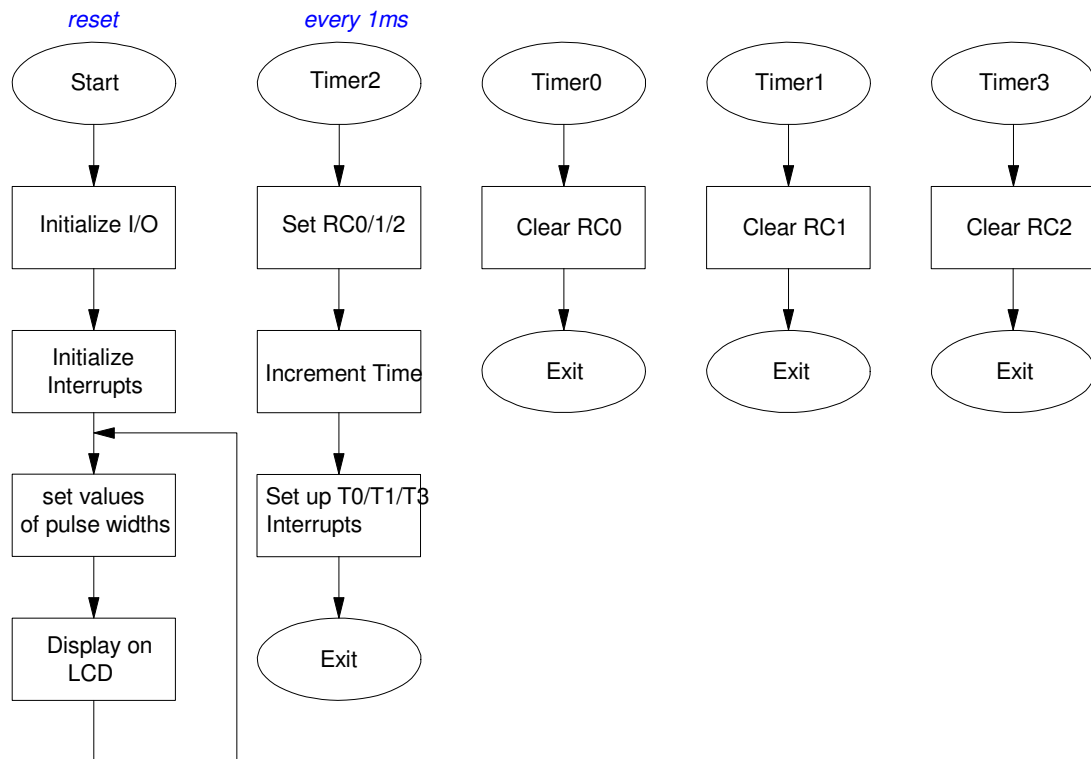
Write a program to output the positive voltage for a 3-phase sine wave using Timer interrupts

- Timer2 interrupt triggers every 1ms and sets pins RC0 (phase A), RC1 (B), and RC2 (C)
- When Timer2 triggers, it sets up a Timer0/1/3 interrupt nA/nB/nC clocks in the future
  - Timer0 interrupt then clears RC0 (setting the pulse width of phase A)
  - Timer1 interrupt then clears RC1 (setting the pulse width of phase B)
  - Timer3 interrupt then clears RC2 (setting the pulse width of phase C)
- The pulse width is determined by nA / nB / nC
  - 100 = 1%
  - 9900 = 99%
- The main routine is responsible for setting the values of NA, NB, and NC



3) Give a flow chart for this program

- There should be five flow charts (one for each interrupt and one for the main routine)



#### 4) Write the corresponding C code

For phase A

- The zero crossings are at 0 and 1500ms
- The height is 9999 (99.99%)

$$N_a = 0.0178(t)(1500 - t)$$

$$N_a = 1.778\left(\frac{t}{10}\right)\left(150 - \frac{t}{10}\right)$$

Phase B is 120 degrees delayed

Phase C is 240 degrees delayed

**Interrupt Service Routine:**

```
void interrupt IntServe(void)
{
    if (TMR0IF) {
        RC0 = 0;
        TMR0IF = 0;
    }
    if (TMR1IF) {
        RC1 = 0;
        TMR1IF = 0;
    }
    if (TMR2IF) {
        if(Na > 0) RC0 = 1;
        if(Nb > 0) RC1 = 1;
        if(Nc > 0) RC2 = 1;
        TMR0 = -Na;
        TMR1 = -Nb;
        TMR3 = -Nc;
        Time += 1;
        if(Time >= 3000) Time = 0;
        TMR2IF = 0;
    }
    if (TMR3IF) {
        RC2 = 0;
        TMR3IF = 0;
    }
}
```

**Main Routine (option #1: using formulas for a parabolic sine wave)**

```
while(1) {
    Xa = 0.1*Time;
    if(Xa < 150) Na = 1.6*Xa*(150-Xa);
    else Na = 0;
    Xb = Xa - 100;
    if(Xb < 0) Xb += 300;
    if(Xb < 150) Nb = 1.6*Xb*(150-Xb);
    else Nb = 0;
    Xc = Xa - 200;
    if(Xc < 0) Xc += 300;
    if(Xc < 150) Nc = 1.6*Xc*(150-Xc);
    else Nc = 0;
}
```

## Option #2: Using a look-up table.

Step 1: Generate a 1/2 rectified sine wave with 30 entries (30 is an arbitrary number)

```
>> N = [0:29]';
>> Y = sin(2*pi*N/30);
>> Y = max(0,Y);
>> Y = floor(9900*Y);
```

In Code, walk through the table (without interpolation):

```
TABLE = [0, 2058, 4026, 5819, 7357, 8573, 9415, 9845, 9845, 9415, 8573, 7357, 5819,
4026, 2058, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0];

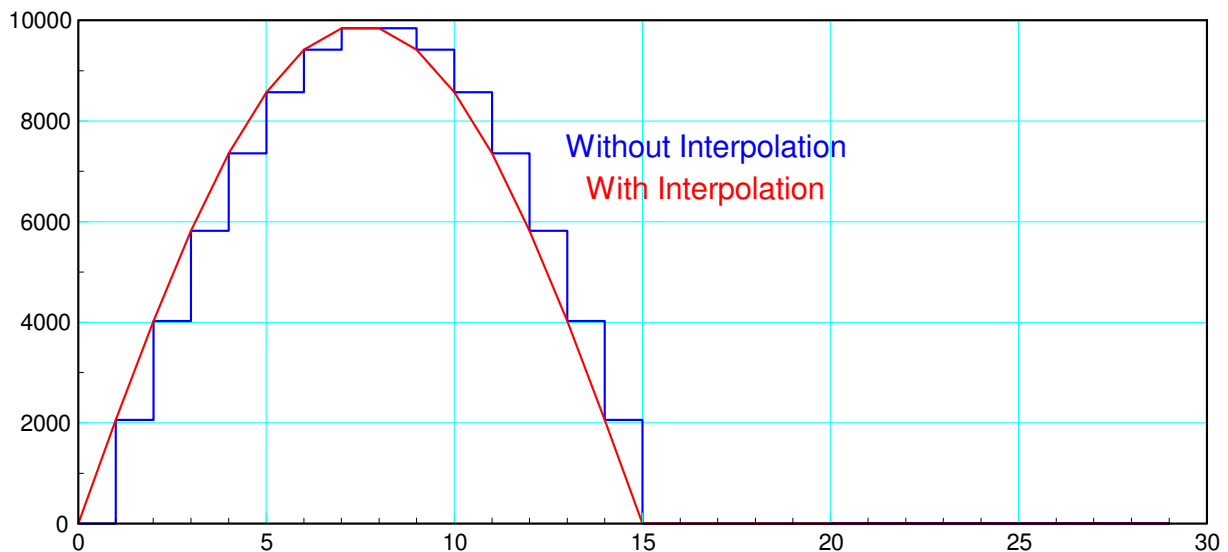
while(1) {
    N = (0.01 * Time) % 30;
    Na = TABLE(N)
    Nb = TABLE( (N+10)%30 );
    Nc = TABLE( (N+20)%30 );
}
```

Walk through the table (with interpolation)

- note: It would be more efficient to stick to powers of 2 (64 entries in the table, 2048 points per cycle, etc)*

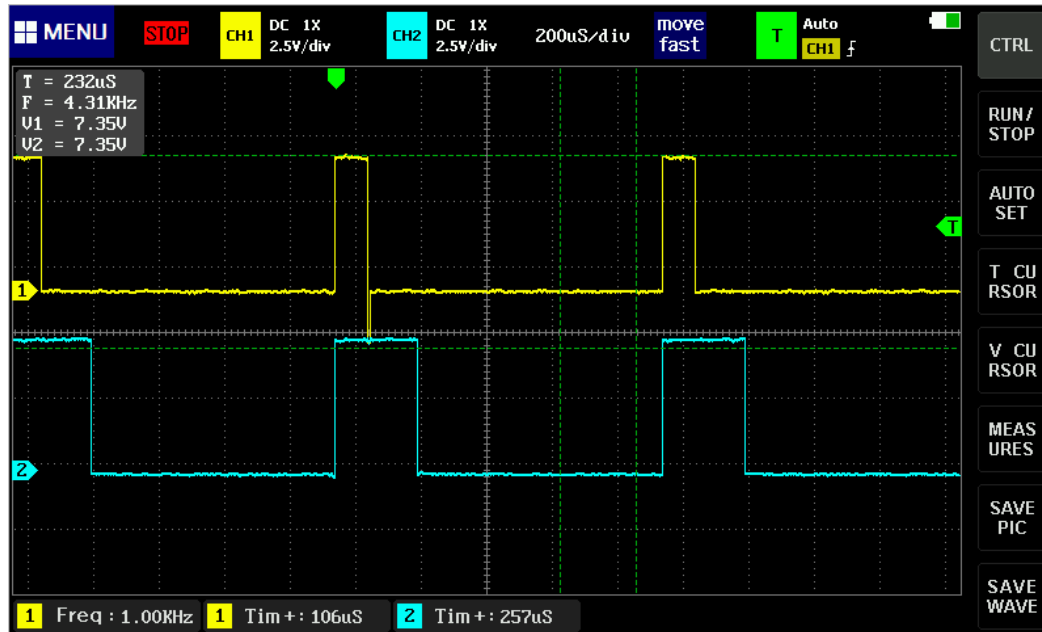
```
int Interpolate(unsigned int Time) {
    unsigned int N, PWM;
    float k

    // N goes 0..29
    N = ( 0.01 * Time ) % 30;
    // k is the fraction 0..99
    k = ( Time % 100 ) / 100;
    PWM = (1-k)*TABLE[N] + k*TABLE[ (N+1)%30 ];
    return(PWM)
}
```



5) Verify the interrupts are working

- If  $nA = 1000$  (10%), you read 0.50V on RC0 with multimeter (or 10% on an oscilloscope)
- If  $nB = 2500$  (25%), you read 1.25V on RC1
- If  $nC = 8000$  (80%) you read 4.00V on RC2
- Timer2 kicks in every 1.00ms



Channel A (yellow)

- Frequency = 1.000kHz (set by Timer2)
- Pulse Width = 106µs (should be 100µs)

Channel B (blue)

- Frequency = 1.000kHz (set by Timer2)
- Pulse Width = 257µs (should be 250µs)

Looks like it's working, but the pulse width is 6µs or 7µs too long (60 or 70 clocks)

6) Demo: Demonstrate a 3-phase rectified sine wave with a period of 3 seconds

- Phase A cycles from 0% to 100% then back to 0%
- Phase B lags phase A by 120 degrees
- Phase C lags phase A by 240 degrees

