ECE 376 - Test #3: Name _____

Fall 2023

1a) Single Interrupt. The following C code sets up a Timer2 interrupt to output a square wave on RC0. Determine the frequency that appears on pin RC0.

T2CON = 0x73 = b0111 0011

A = 15, C = 16

PR2 = 46

B = 47

In the interrupt, RC0 toggles every 13th interrupt, so

N = 13*A*B*C

N = 146,640

$$f = \left(\frac{10{,}000{,}000}{2 \cdot N}\right) = 34.097 Hz$$

b..f) If the following sections of code are deleted, what frequency will you see on pin RC0?

Section of Code	Frequency on RC0 if this section is deleted

`// Global variable`
`unsigned int COUNT`
 	
`void interrupt IntServe(void) {`
	code doesn't compile

`N = (N + 1) % 13;` `if(N == 0)`	b) RC toggles every interrupt, resulting in 13x the frequency **f = 443.26Hz**
`RC0 = !RC0;`	c) RC0 never toggles. **f = 0Hz**
`TMR2IF = 0;`	d) Stuck inside the interrupt interrupt every 50 clocks N = 13 * 50 **f = 7692Hz**

`}`
`void main(void) {`
`TRISC = 0;`
`ADCON1 = 0x0F;`
	code doesn't compile

`T2CON = 0x73;` `PR2 = 46;`	e) A,B,C are something, I just don't know what (they are not initialized incode) **frequency = unknown**

`TMR2ON = 1;`
`TMR2IE = 1;`
`TMR2IP = 1;`
`PEIE = 1;`
`GIE = 1;`
	f) Interrupts are not called
	f = 0Hz

`while(1) {`
`RC1 = !RC1;`
`}`
`}`
	code doesn't compile

2) Multiple Interrupts: Give the interrupt service routine and interrupt initialization code for a Hungry Hungry Hippo game where each player has a hadicap:

- Player A gets one point for every rising edge on RA4 (Timer0 external input)
- Player B gets one point for every 3rd rising edge on RC0 (Timer1 external input)
- Player C gets one point for every 5th rising edge on RC1 (Timer3 external input)

Interrupt Initialization

	Timer0 count every rising edge on RA4	Timer1 count every 3rd rising edge on RC0	Timer3 count every 5th rising edge on RC1
Initialization	`T0CS = 1;` `T0CON = 0x88;`	`TMR1CS = 1` `T1CON = 0x81`	`TMR3CS = 1` `T3CON = 0x81`
PS	1	1	1

Interrupt Service Routines (Option 1)

Timer0 — count every rising edge on RA4

```
if(TMR0IF) {

    TMR0 = -1;
    A = A + 1;

    TMR0IF = 0;

    }
```

Timer1 — count every 3rd rising edge on RC0

```
if (TMR1IF) {

   TMR1 = -3;
   B = B + 1;

   TMR1IF = 0;
   }
```

Timer3 — count every 5th rising edge on RC1

```
if (TMR3IF) {

   TMR3 = -5;
   C = C + 1;

   TMR3IF = 0;

   }
```

(Option 2)

Timer0 — count every rising edge on RA4

```
if(TMR0IF) {

    TMR0 = -1;
    A = A + 1;

    TMR0IF = 0;

    }
```

Timer1 — count every 3rd rising edge on RC0

```
if (TMR1IF) {

   TMR1 = -1;
   Nb = (Nb + 1) % 3;

   if(Nb == 0)
      B = B + 1;

   TMR1IF = 0;
   }
```

Timer3 — count every 5th rising edge on RC1

```
if (TMR3IF) {

   TMR3 = -1;
   Nc = (Nc + 1) % 5;

   if(Nc == 0)
      C = C + 1;

   TMR3IF = 0;

   }
```

3) Bathroom Fan: Write a C program which uses interrupts to control the light and fan in a bathroom:

- RB0 On: Turn on the light and the fan when RB0 is pressed
- RB1 Off: Turn off the light when RB1 is pressed and the fan remains on.
  - 10.00 seconds later, the fan is then turned off

Assume

- RC0 controls the bathroom light (1 = on, 0 = off)
- RC1 controls the fan (1 = on, 0 = off)

a) Interrupt Initialization: (affects the interrupt service routine)

INT0 rising or falling edge	INT1 rising or falling edge	Timer0 prescalar for Timer0 (1,2,4,8,16,32,64,128,256)
rising edge	rising edge	PS = 256 (interrupt every second)

b) Interrupt service Routines

INT0 — trigger on RB0

```
if(INT0IF) {
   RC0 = 1;
   RC1 = 1;

   INT0IF = 0;
   }
```

INT1 — trigger on RB1

```
if(INT1IF) {
   RC0 = 0;
   N = 10;
   TMR0 = -39062;

   INT1IF = 0;
   }
```

Timer0 — Called every 1.00 second

```
if(TMR0IF) {
   TMR0 = -39062;
   if(N) N = N - 1;

   if(RC0)
      RC1 = 1;
   else
      if(N == 0)
      RC1 = 0;

   TMR0IF = 0;

   }
```

4) Digital Filters: Assume X and Y are related by

$$Y = \left(\frac{2(s+3)(s+5)}{(s+1)(s+10)}\right) X$$

a) Give the transfer funciton for a digital filter, G(z), which has

- Approximately the same gain vs. frequency,
- With a sampling rate of T = 0.01 seconds

Convert from the s-plane to the z-plane as $z = \exp(sT)$

$s = -3$ $\quad z = e^{sT} = 0.9704$

$s = -5$ $\quad z = e^{sT} = 0.9512$

$s = -1$ $\quad z = e^{sT} = 0.9900$

$s = -10$ $\quad z = e^{sT} = 0.9048$

so G(z) is of the form

$$G(z) = k\left(\frac{(z-0.9704)(z-0.9512)}{(z-0.9900)(z-0.9048)}\right)$$

Pick k to match the DC gain

$$\left(\frac{2(s+3)(s+5)}{(s+1)(s+10)}\right)_{s=0} = 3.00$$

$$k\left(\frac{(z-0.9704)(z-0.9512)}{(z-0.9900)(z-0.9048)}\right)_{z=1} = 3.00$$

$$k = 1.9708$$

$$G(z) = 1.9708\left(\frac{(z-0.9704)(z-0.9512)}{(z-0.9900)(z-0.9048)}\right)$$

b) Write a C program to implement this filter assuming a sampling rate of T = 0.01 seconds.

Multiply out

$$Y = 1.9708\left(\frac{z^2-1.9217z+0.9231}{z^2-1.8949z+0.8958}\right)X$$

Cross multiply

$$(z^2 - 1.8949z + 0.8958)Y = 1.9708(z^2 - 1.9217z + 0.9231)X$$

Convert to a difference equation

y(k+2) - 1.8949 y(k+1) + 0.8958 y(k) = 1.9708(x(k+2) - 1.9217 x(k+1) + 0.9231 x(k))

Time shift by 2 (change of variable)

y(k) - 1.8949 y(k-1) + 0.8958 y(k-2) = 1.9708(x(k) - 1.9217 x(k-1) + 0.9231 x(k-2))

Sovle for y(k)

y(k) = 1.8949 y(k-1) - 0.8958 y(k-2) + 1.9708(x(k) - 1.9217 x(k-1) + 0.9231 x(k-2))

That's essentially your program

```
while(1) {
   x2 = x1;
    x1 = x0;
    x0 = A2D_Read(0);

    y2 = y1;
    y1 = y0
    y0  = 1.8949*y1 - 0.8958*y2 + 1.9708*( x0 - 1.9217*x1 + 0.9231*x2 );

     D2A(y0);

     Wait_10ms();

     }
```