ECE 376 - Test #1: Name _____

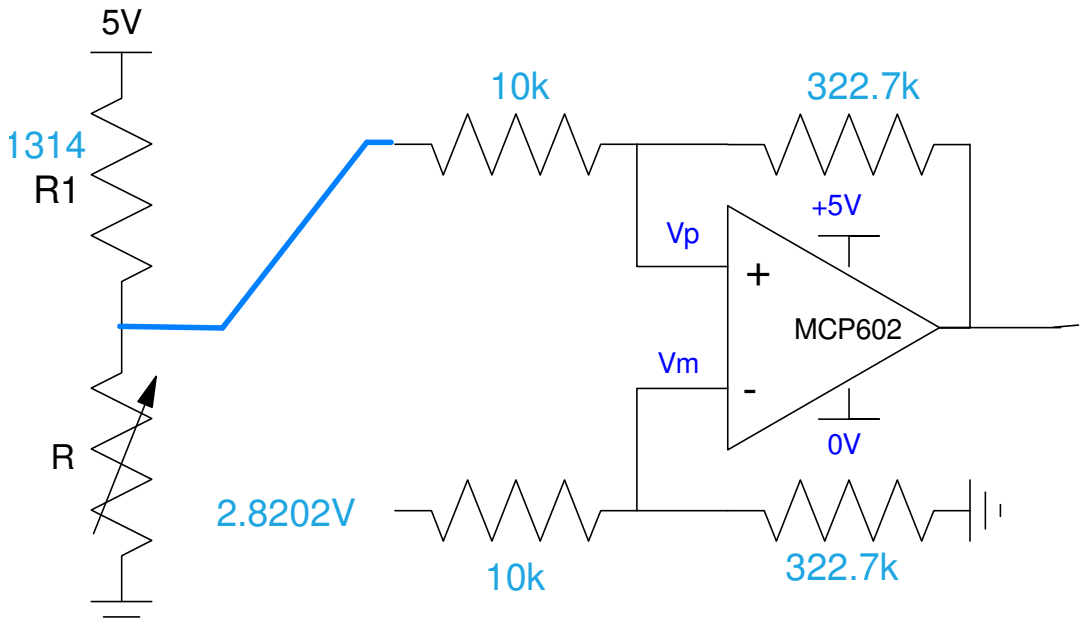
Spring 2023

1) **Digital Inputs.** Design a circuit which outputs

- 0V when R < 1500 Ohms
- 5V when R > 1700 Ohms

Assume

- R1 = 800 + 100*(your birth month) + (your birth date).
- May 14th, for example, gives R1 = 1314 Ohms



R = 1500 (off)

$$V_a = \left(\frac{1500}{1500+1314}\right)5V = 2.6652V$$

R = 1700 (On)

$$V_a = \left(\frac{1700}{1700+1314}\right)5V = 2.8202V$$

V(on) > V(off) so connect to the plus input

Output turns on at 2.82V - make the offset 2.82V

$$gain = \left(\frac{5V-0V}{2.8202V-2.6652V}\right) = 32.27$$

2) Digital Outputs: Design a circuit which allows your PIC to drive a 100W LED at N mA

- N = 800 + 100*(your birth month) + (your birth date)
- N = 1314 mA for May 14th, for example

Assume a 20W LED has the following characteristics

- Vf = 10V @ 2000mA
- 2,000 Lumens @ 2000mA

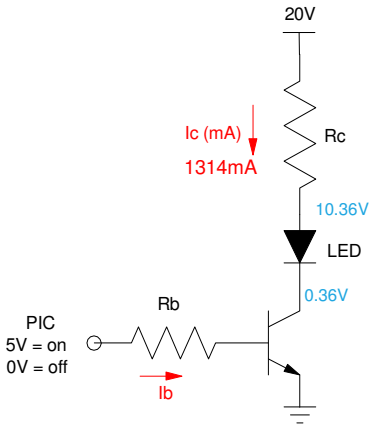
Assume a 6144 NPN transistor

- Vbe = 700mV
- Vce(sat) = 360mV
- Current gain = β = 200

Ic (mA) 800 + 100*Month + Day	Lumens Light output when on	Rb	Rc
1314mA	**1314**	**172 .. 654**	**7.33**

$$Light = \left(\frac{1314mA}{2000mA}\right) 2000Lumens = 1314Lumens$$

$$R_c = \left(\frac{20V - 10.36V}{1314mA}\right) = 7.33\Omega$$



Range of Rb

$$\min(I_b) = \left(\frac{I_c}{\beta}\right) = \left(\frac{1314mA}{200}\right) = 6.57mA$$

$$R_b = \left(\frac{5V - 0.7V}{6.57mA}\right) = 654\Omega$$

$$\max(I_b) = 25mA$$

$$R_b = \left(\frac{5V - 0.7V}{25mA}\right) = 172\Omega$$

3) **Assembler:** Determine the contents of the W, A, and B after each operation. Assume

- A and B are 8-bit registers (spots in memory).
- Default is decimal

	W	A	B
Start:	**13**	Birth Month (1..12) **5**	Birth Date (1..31) **14**
`incf A,W`	**6**	5	14
`decf B,W`	**13**	5	14
`addlw 5`	**18**	5	14
`addwf A,F`	18	**23**	14
`subwf B,W`	**252** -4 is also correct	23	14
`movf A,W`	**23**	23	14
`movff A,B`	23	23	**23**
`andlw 7`	**7**	23 0001 0111	23
`btg A,1`	7	**21** 0001 0101	23
`movwf B`	7	21	**7**

4) Assembler & Timing:

a) Determine the number of clocks the following assembler subroutine takes to execute.

- Assume MONTH and DAY be your birth month and day.

b) Modify this routine (change A, B, and C) so that it takes 2,500,000 clocks (250ms) to execute

- +/- 50,000 clocks

A	Month birth month 1..12	Day birth date 1..31	N number of clocks Wait takes
150	**5**	**14**	**67,808** 6ABC + 5AB + 7A + 8
A	B	C	N 2,500,000 +/- 50,000
7	**232**	**255**	**2,492,897**

Other values also work.

Limitations: A, B, C are all integers, in the range of 1..255

```
Wait:
     movlw     150 (A)
     movwf     CNT2
      nop
      nop
      nop
      nop
W2:
          movlw     MONTH    (B)
          movwf     CNT1
          nop
          nop

W1:
               movlw     DAY (C)
               movwf     CNT0
W0:
                    nop
                    nop
                    nop
                    decfsz CNT0,F
                    goto   W0

               decfsz  CNT1,F
               goto    W1

          decfsz    CNT2,F
          goto W2

     return
```

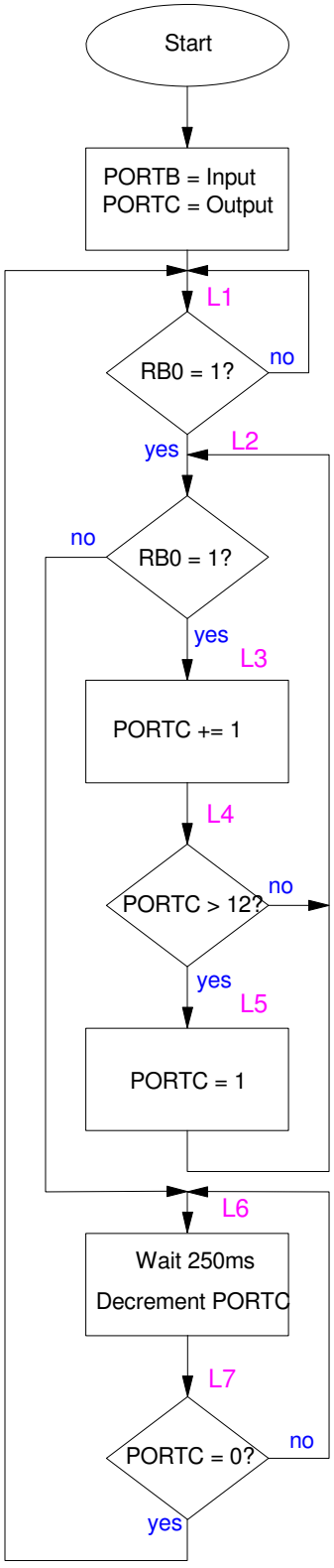
5) Assember & Flow Charts. Write an assembler program to turn your PIC processor into random count-down timer

- When RB0 is pressed, PORTC counts from 1..12 really fast
- When RB0 is released, PORTC then contains a random number from 1..12
- PORTC then counts down every 250ms
- When PORTC reaches zero, it then goes back to the beginning and waits for RB0.

Assume a 250ms wait routine exists (call Wait)

```
     org 0x800
     movlw     0x0F
     movwf     ADCON1

     movlw     0xFF
     movwf     TRISB
     clrf      TRISC
L1:
     btfss     PORTB,0
     goto L1
L2:
     btfss     PORTB,0
     goto L6
L3:
     incf PORTC,F
L4:
     movlw     12
     cpfsgt    PORTC
     goto      L5
     goto      L2
L5:
     movlw     1
     movwf     PORTC
     goto      L2
L6:
     call      Wait
     decf      PORTC,F
L7:
     movlw     0
     cpfseq    PORTC
     goto      L6
     goto      L1
```



Bonus: (Due Monday 2pm): Program and demonstrate problem #5 on yor PIC board

Memory Read & Write		
MOVWF PORTA	memory write	w g PORTA
MOVFF PORTA PORTB	copy	PORTA g PORTB
MOVF PORTA,W	memory read	PORTA g W
MOVLW 234	Move Literal to WREG	123 g W
Memory Clear, Negation		
CLRF PORTA	clear memory	0x00 g PORTA
COMF PORTA, W	toggle bits	!PORTA g W (bit toggle)
NEGF PORTA, W	negate	-PORTA g W (2's compliment)
Addition & Subtraction		
INCF PORTA,F	increment	PORTA + 1 g PORTA
ADDWF PORTA, F	add	PORTA + W g PORTA
ADDWFC PORTA, W	add with carry	PORTA + W + carry g W
ADDLW	Add Literal and WREG	
DECF PORTA,F	decrement	PORTA -1 g PORTA
SUBFWB PORTA,F	subtract with borrow	PORTA - W - c g PORTA
SUBWF PORTA,F	subtract no borrow	PORTA - W g PORTA
SUBWFB PORTA,F	subtract with borrow	PORTA - W - c g PORTA
SUBLW 223	Subtract WREG from #	223 - W g W
Shift left (*2), shift right (/2)		
RLCF PORTA,F	rotate left through carry (9-bit rotate)	
RLNCF PORTA,F	rotate left no carry	
RRCF PORTA,F	rotate right through carry	
RRNCF PORTA,F	rotate right no carry	
Bit Operations		
BCF PORTA, 3	Bit Clear f	clear bit 3 of PORTA
BSF PORTA, 4	Bit Set f	set bit 4 of PORTA
BTG PORTA, 2	Bit Toggle f	toggle bit 2 of PORTA
Logical Operations		
ANDWF PORTA, F	logical and	PORTA = PORTA and W
ANDLW 0x23	AND Literal with WREG	W = W and 0x23
IORWF PORTA,F	logical or	PORTA = PORTA or W
IORLW 0x23	Inclusive OR Literal	W = W or 0x23
XORWF PORTA,F	logical exclusive or	PORTA = PORTA xor W
XORLW 0x23	Exclusive OR Literal	W = W xor 0x23
Tests (skip the next instruction if...)		
CPFSEQ PORTA	Compare PORTA to W, skip if PORTA = W	
CPFSGT PORTA	Compare PORTA to W, Skip if PORTA > W	
CPFSLT PORTA	Compare PORTA to W, Skip if PORTA < W	
DECFSZ PORTA,F	decrement, skip if zero	
DCFSNZ PORTA,F	decrement, skip if not zero	
INCFSZ PORTA,F	increment, skip if zero	
INFSNZ PORTA,F	increment, skip if not zero	
BTFSC PORTA, 5	Bit Test f, Skip if Clear	
BTFSS PORTA, 1	Bit Test f, Skip if Set	
Flow Control		
GOTO Label	Go to Address 1st word	
CALL Label	Call Subroutine 1st word	
RETURN	Return from Subroutine	
RETLW 0x23	Return with 0x23 in WREG	