

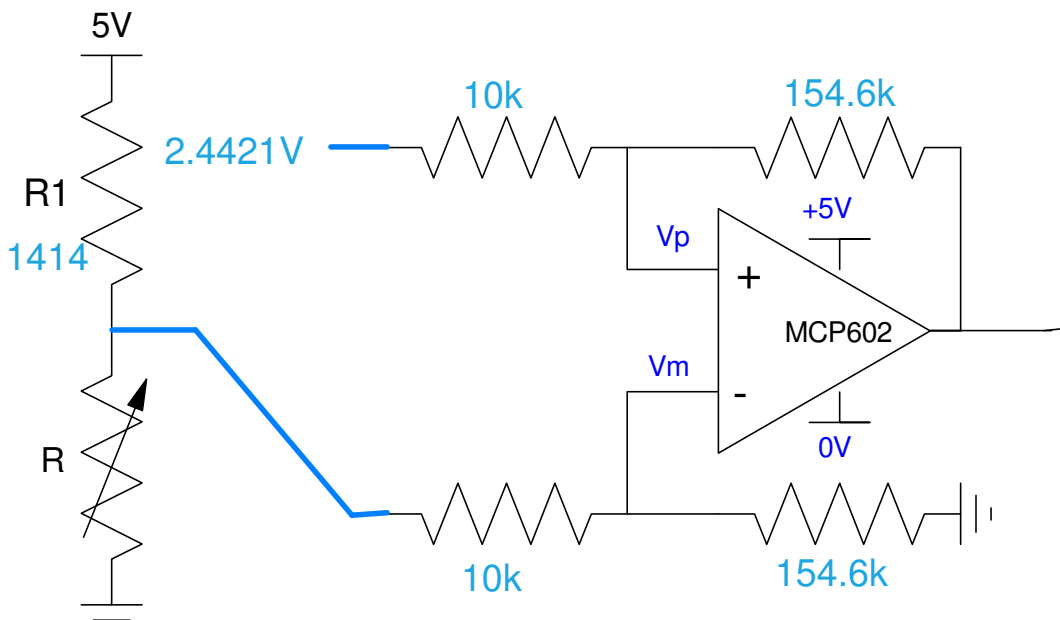
# ECE 376 - Test #1: Name \_\_\_\_\_

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

- 0V when  $R > 1750$  Ohms
- 5V when  $R < 1350$  Ohms

Assume

- $R1 = 900 + 100 \cdot (\text{your birth month}) + (\text{your birth date})$ .
- May 14th, for example, gives  $R1 = 1414$  Ohms



$$R(\text{on}) = 1350$$

$$V_x = \left( \frac{1350}{1350+1414} \right) 5V = 2.4421V$$

$$R(\text{off}) = 1750$$

$$V_x = \left( \frac{1750}{1750+1414} \right) 5V = 2.7655V$$

$$V(\text{on}) < V(\text{off})$$

- connect to the minus input

$$V(\text{on}) = 2.3950V$$

- Make the offset 2.3950V

$$\text{gain} = \left( \frac{5V-0V}{2.7655V-2.4421V} \right) = 15.46$$

**2) Digital Outputs:** Design a circuit which allows your PIC to turn on and off a 50W LED

- $R_c = 900 + 100 \cdot (\text{your birth month}) + (\text{your birth date})$  Ohms
- $R_c = 1414$  Ohms for May 14th, for example

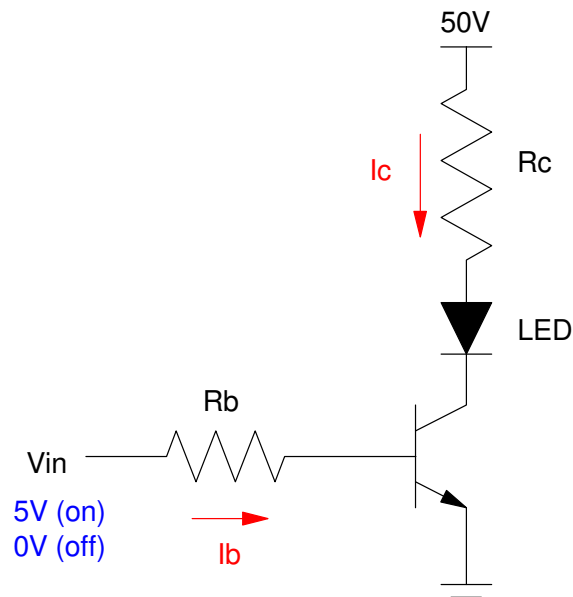
Assume a 50W LED has the following characteristics

- $V_f = 25V @ 2A$
- 5000 Lumens @ 2A

Assume a 6144 NPN transistor

- $V_{be} = 700mV$
- $V_{ce(sat)} = 200mV$
- Current gain =  $\beta = 300$

Lumens	$I_c$ (mA)	$R_b$	$R_c$ 900 + 100*Month + Day
43.85 Lumens	17.54 mA	4.3k Ohms	1414 Ohms



$$I_c = \left( \frac{50V - 25V - 0.2V}{1414\Omega} \right) = 17.54mA$$

$$I_b > \left( \frac{I_c}{300} \right) = 58.5\mu A$$

Let  $I_b = 1mA$

$$R_b = \left( \frac{5V - 0.7V}{1mA} \right) = 4.3k\Omega$$

$$Light = \left( \frac{17.54mA}{2000mA} \right) 5000 \text{ Lumens} = 43.85 \text{ Lumens}$$

3) **Assembler:** Determine the contents of the W, PORTB, and PORTC registers after each operation.

Assume

- PORTB and PORTC are output.
- Default is decimal

	W	PORTB	PORTC
Start:	Birth Month (1..12) <b>5</b>	Birth Date (1..31) <b>15</b>	<b>15</b>
addlw 2	7	15	15
subwf PORTC,F	7	15	8
addwf PORTB,F	7	22	8
movf PORTB,W	22	22	8
iorlw 0x15	23	22	8
movwf PORTB	23	23	8
btg PORTC,0	23	23	9
iorlw 0x0F (not graded - typo)	31	23	9
negf PORTB,F	31	-23	9
comf PORTC,F	31	-23	-10

#### 4) Assembler & Timing:

a) Convert the following C code to assembler.

- Assume A and B are 8-bit numbers

b) How long does your program take to execute when A = 25?

# Clocks 16 clocks
-----------------------

```
; unsigned char A, B;

A equ 0
B equ 1

; if(A < 10)
;   B = 0
    movlw    10    1
    cpfslt   A     2
    goto    L2    3,4
    clrf    B
    goto    L5

; else if(A < 20)
;   B = 1
L2: movlw    20    5
    cpfslt   A     6
    goto    L3    7,8
    movlw    1
    movwf   B
    goto    L5

; else if(A == 25)
;   B = 2
L3: movlw    25    9
    cpfseq   A    10,11
    goto    L4
    movlw    2    12
    movwf   B    13
    goto    L5    14,15

; else
;   B = 3
L4: movlw    3
    movwf   B

L5: nop      16
```

**5) Assembler & Flow Charts.** Write an assembler program that corresponds to the following flow chart.

This program gives you points based upon which button you press:

- Button RB0: Usually one point to PORTC, sometimes adds 3 points to PORTC
- Button RB1: Usually adds two points to PORTD, sometimes adds 3 points to PORTC

Test #1: (due Friday) Write the assembler code

Bonus (due Monday): Demonstrate your program on your PIC board

```
#include <p18f4620.inc>
```

```
; Start of Program
org 0x800
```

```
L1:  movlw    0xFF
     movwf   TRISB
     clrf    TRISA
     clrf    TRISC
     clrf    TRISD
     clrf    PORTC
     clrf    PORTD

L2:  movlw    0
     cpfseq  PORTB
     goto    L4

L3:  incf     PORTA,W
     andlw   0x0F
     movwf   PORTA
     goto    L2

L4:  movlw    3
     cpfseq  PORTA
     goto    L6

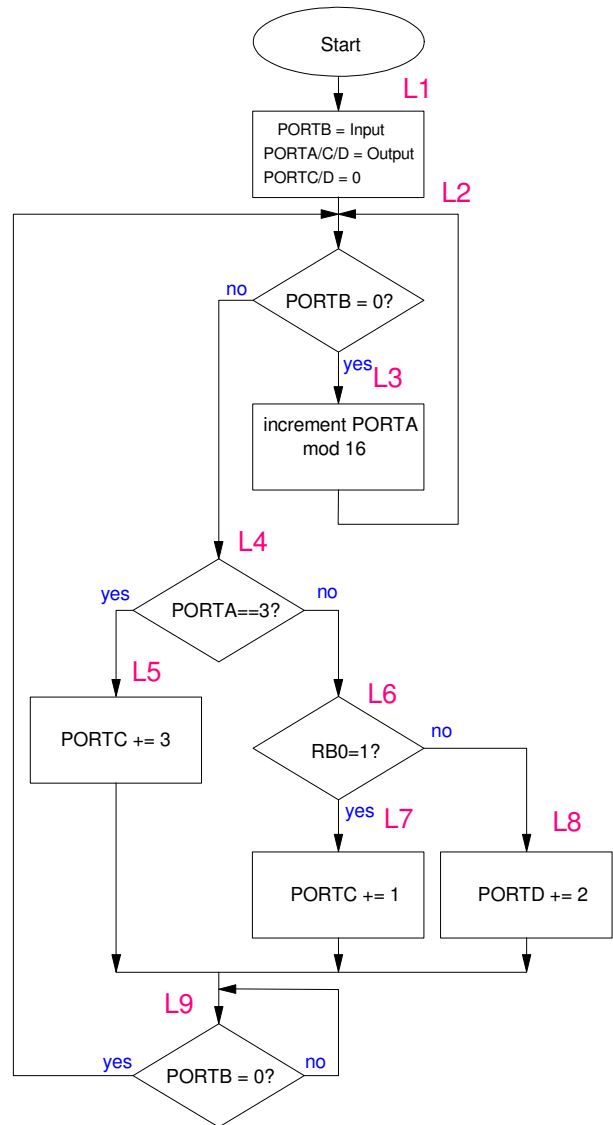
L5:  movlw    3
     addwf   PORTC,F
     goto    L9

L6:  btfss   PORTB,0
     goto    L8

L7:  incf     PORTC,F
     goto    L9

L8:  movlw    2
     addwf   PORTD,F

L9:  movlw    0
     cpfseq  PORTB
     goto    L9
     goto    L2
```







Memory Read & Write			
MOVWF	PORTA	memory write	w → PORTA
MOVFF	PORTA PORTB	copy	PORTA → PORTB
MOVF	PORTA,W	memory read	PORTA → W
MOVLW	234	Move Literal to WREG	123 → W
Memory Clear, Negation			
CLRF	PORTA	clear memory	0x00 → PORTA
COMF	PORTA, W	toggle bits	!PORTA → W (bit toggle)
NEGF	PORTA, W	negate	-PORTA → W (2's compliment)
Addition & Subtraction			
INCF	PORTA,F	increment	PORTA + 1 → PORTA
ADDWF	PORTA, F	add	PORTA + W → PORTA
ADDWFC	PORTA, W	add with carry	PORTA + W + carry → W
ADDLW		Add Literal and WREG	
DECF	PORTA,F	decrement	PORTA - 1 → PORTA
SUBFWB	PORTA,F	subtract with borrow	PORTA - W - c → PORTA
SUBWF	PORTA,F	subtract no borrow	PORTA - W → PORTA
SUBWFB	PORTA,F	subtract with borrow	PORTA - W - c → PORTA
SUBLW	223	Subtract WREG from #	223 - W → 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	
BTFS	PORTA, 5	Bit Test f, Skip if Clear	
BTFS	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	