# ECE 376 - Test #1: Name _____

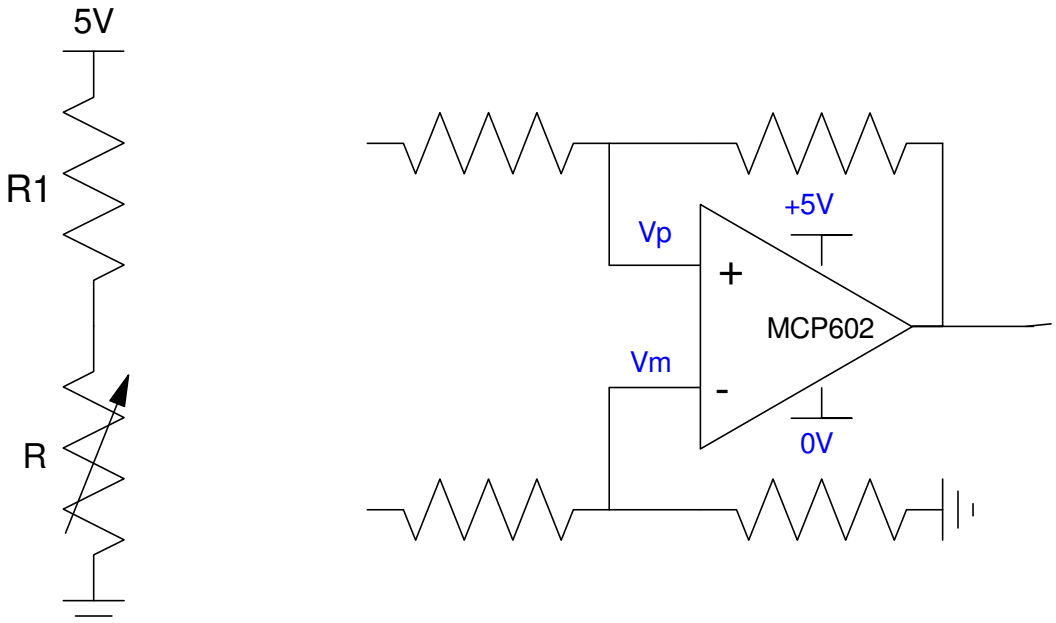
1) **Digital Inputs.** A light sensor has the following resistance vs lumens:

$$R = 10,000 \cdot (L)^{-0.6} \Omega$$

where L is the light level in lumens. Design a circuit which outputs:

- +5V when L < 10 lumens
- 0V when L > 15 lumens
- No change for 10 lumens < L < 15 lumens

| | |
|---|---|
| R1 = 900 + 100(Birth Month) + Birth Date<br>ex: May 14 = 1414 Ohms | R1 = |



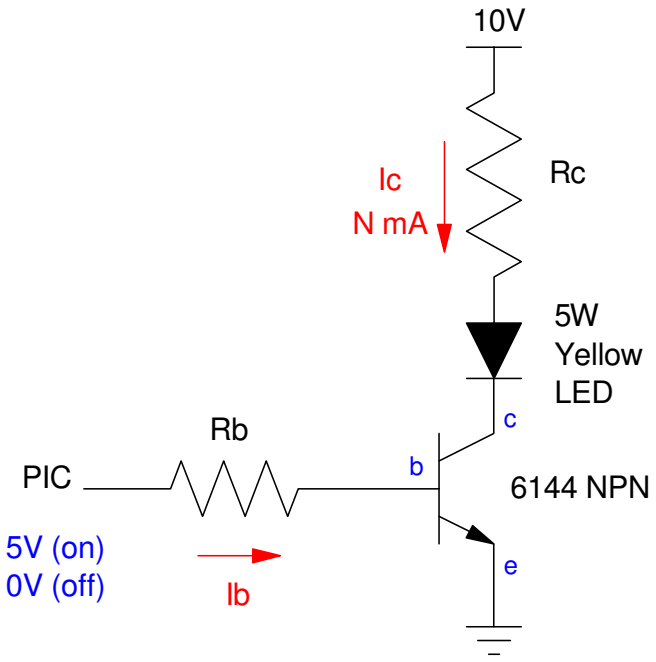
2) Digital Outputs: Determine Rb and Rc so that your PIC can drive a white 5W yellow LED at N mA where N is related to your birthday

- Vf = 2.4V @ 1200mA
- 600 Lumens @ 1200mA
- N = 900 + 100*(birth month) + (birth date).

Assume a 6144 NPN transistor

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

| N mA<br>900 + 100*(Birth Month) + Birth Date<br>ex: May 14th = 1414mA | Rb | Rc |
|---|---|---|
| | | |



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) | `0x23 = 35` | Birth Date (1..31) |
| `movf PORTC,W` | | | |
| `movff PORTB, PORTC` | | | |
| `movwf PORTB` | | | |
| `movlw 7` | | | |
| `xorwf PORTB,F` | | | |
| `btg  PORTC,0` | | | |
| `negf PORTB, W` | | | |

**4) Assembler & Timing:** Determine the number of clocks the following assembler subroutine takes to execute. Assume MONTH and DAY be your birth month and day.

| MONTH<br>(birth month: 1..12) | DAY<br>(birth day: 1..31) | N<br>Number of clocks Wait routine takes |
|---|---|---|
| | | |
| Find A / B / C for N = 22,000,000 +/- 500,000<br>( 2.2 seconds +/- 0.05 second ) | | |
| A = | B = | C = |

```
Wait:
     movlw     MONTH (A)
     movwf     CNT2
    nop
W2:
          movlw     DAY (B)
          movwf     CNT1
          nop
         nop
W1:
               movlw     177 (C)
               movwf     CNT0
               nop
               nop
               nop
               nop
W0:
                    nop
                    nop
                    nop
                    nop
                    nop
                    nop
                    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 a combination lock

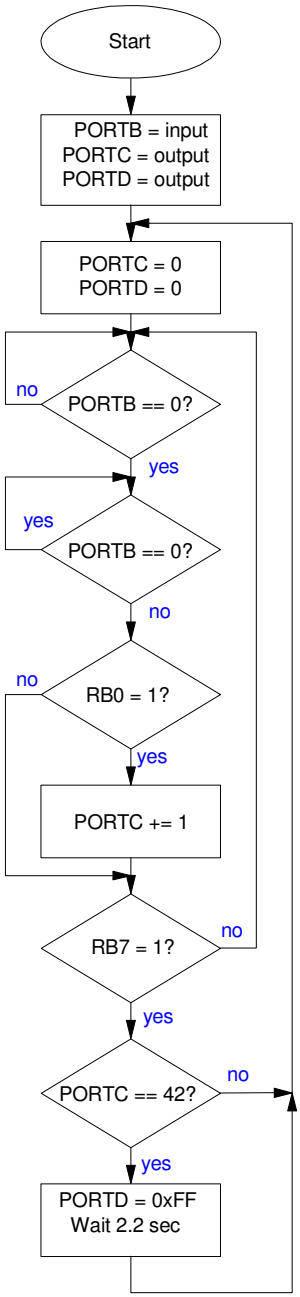
- Press RB0 (PORTB pin 0) N times, (number of presses is the combination) then
- Press RB7 (PORTB pin 7) one time to try to open the lock

If you pressed RB0 42 times, the lock opens (PORTD = 255)

Otherwise, the lock remains closed (PORTD = 0) and the count starts over (PORTC = 0)

note: Assume a Wait function exists that waits 2.2 seconds

- (a different problem on this test)




```
#include <pic18f4620.inc>

     org   0x800
     movlw 0x0F
     movwf ADCON1
```

| 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 | |
| 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 | |