Computer Interfacing (CI) FIB. Primer parcial **28/10/2022**

Cognoms, Nom _____ DNI _____

Tota resposta sense justificar es considerarà nul·la !

P1. (2 punts)

1.1 Quants cicles triga a executar-se el següent codi si Fosc=4MHz:

```
   LFSR FSR2, 000 ; 2 cicles
 loop
   CLRF INDF2, A ; 1 cicle
   INCF FSR2L, F,  A ; 1 cicle
   BTFSC FSR2L, 6, A ; 1 cicle si no salta, 3 si salta una instrucció 2-word
   GOTO final; 2 cicles
   GOTO loop ; 2 cicles
 final
```

El codi itera sobre el banc 0 de la memòria de dades fins que el bit 6 de l'FSR2L és 1. En total realitza 64 iteracions.

LFSR (2c) + 63*(CLRF(1c) + INCF(1c) + BTFSC fa skip (3c) + GOTO (2 cicles)) + CLRF(1c) + INCF(1c) + BTFSC no fa skip (1c) + GOTO (2 cicles) = 448 cicles

1.2 Proposa una alternativa que tingui la mateixa funcionalitat però que sigui més eficient en el número de cicles que triga en executar-se. Quant triga la teva proposta?

Es pot millorar l'eficiència del codi canviant GOTOs per BRA (estalvia un cicle al BTFSx quan fa skip), També es pot reorganitzar la seqüència de salts per no fer skip a cada iteració i fer servir POSTINC per incrementar l'FSR2 sense necessitar l'INCF

Una proposta pot ser:

```
   LFSR FSR2, 000 ; 2 cicles
loop
   CLRF POSTINC2, A ; 1 cicle
   BTFSS FSR2L, 6, A ; 1 cicle si no salta, 2 cicles si salta
   BRA loop; 2 cicles
final
```

LFSR (2c) + 63*(CLRF(1c) + BTFSS no fa skip (1c) + BRA (2 cicles)) + CLRF(1c) + BTFSS fa skip (2c) = 257 cicles

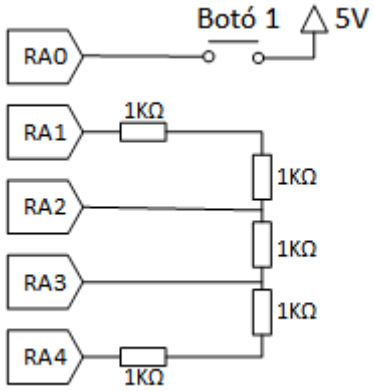
P2. (2 punts)

Dos alumnes han decidit fer una part d'una pràctica de laboratori per separat i just abans de la classe unir el que han fet. Aquest és el resultat:

Codi

```
MOVLB 0x0F
CLRF ANSELA, B
BSF TRISA, 0, B
BCF TRISA, 1, B
BCF TRISA, 4, B
BSF TRISA, 2, B
BSF TRISA, 3, B
loop
BSF LATA, 1, B
BCF LATA, 4, B
BRA loop
end
```

Esquemàtic



2.1 Si suposem que VDD=5V, quin serà l'estat dels següents bits:
Tenim una caiguda de tensió entre RA1 i RA4 de 5V. Com que totes les resistències tenen el mateix valor, cau el mateix voltatge a cada resistència: 1V

Bit	Estat (i justificació)
PORTAbits.RA0 amb el botó 1 premut	RA0=1. Els 5V arribaran al pin.
PORTAbits.RA0 amb el botó 1 sense polsar	RA0= indeterminat. El pin queda a l'aire al no disposar d'una R de pull-down
PORTAbits.RA1	RA1=1. Pin configurat com a output amb LAT configurat a 1.
PORTAbits.RA2	RA2=1. Al pin arriben 3V que és més gran que VIH
PORTAbits.RA3	RA3=1. Al pin arriben 2V que és més gran que VIH. Si suposem un VOH menor a 5V pot ser indeterminat.
PORTAbits.RA4	RA4=0. Pin configurat com a output amb LAT configurat a 0.

2.2 Circula corrent pel pin RA1? Quants ampers? Això suposa algun problema?
Sí que circula corrent entre RA1 i RA4.
Si suposem que RA1=5V i RA4=0V:
I=V/R=5V/5000Ω=1mA
Com que el límit dels pins és de 20mA no suposa cap problema.

Computer Interfacing (CI) FIB. Primer parcial **28/10/2022**

Cognoms, Nom _____ DNI _____

Tota resposta sense justificar es considerarà nul·la !

P3. (3 punts)

Mireu aquest codi d'una funció pel 18F45K22

```
Calcula:    MOVLB 3                 ; Suposem Bank 3 disponible
            CLRF 00, B
            MOVWF 01, B
            MOVLW 08
            MOVWF 02, B
            MOVF 01, W, B

loop:       ANDLW 01
            BZ   no
            INCF 00, F, B
no:         RRNCF 01, F, B          ;RR Rotate Right, com shiftar un bit
            MOVF 01, W, B
            DECFSZ 02, F, B
            BRA loop

            MOVF 00, W, B
            RETURN
```

Si cridem la funció d'aquesta manera

```
            MOVLW 36h
            CALL Calcula
```

3.1 Quin valor tindrem al WREG quan retorni? (0.5p)

El WREG valdrà 4, que és el nombre de 1's que tenia el paràmetre i que s'ha anat calculant a l'adreça 300h.

3.2 Què tindrem a les adreces 300h, 301h i 302h després de la seva execució? (0.75p)

300h tindrà també 4, 301h després de la rotació completa tindrà el valor original del paràmetre, 36h i 302h és el comptador que s'ha anat decrementant fins quedar a 0.

3.3 Quants Bytes ocupa a memòria de programa la funció "Calcula" ? (0.5p)

La funció va des de l'etiqueta calcula fins al return (sinó no seria una funció) i consta de 15 instruccions codificades amb 1 word, per tant 30 Bytes en total.
El codi de cridar a una funció no es compta com a codi de funció. Voldria dir que una funció que es crida més vegades ocupa més que una que es crida poques vegades?

3.4 Quina funcionalitat té el codi englobat a “Calcula” ? (0.5p)

El codi compta el nombre de bits a 1 que té el valor rebut a WREG. Ho fa rotant-lo cap a la dreta 8 vegades i mirant si el bit que queda a la posició 0 és o no un 1. Si ho és incrementa el comptador de la posició 300h.

3.5 El temps d’execució és independent del contingut de WREG a la crida? En cas de que no ho sigui, calcula el cas millor i el cas pitjor. (0.75p)

El temps és independent del valor.
El nombre d’iteracions és constant, 8, perquè fem un recorregut sobre 8 elements.
Dins de cada iteració tenim una estructura alternativa, amb la instrucció BZ després de la AND.
Tant si salta (BZ branch farà dos cicles) com si no salta (BZ farà un cicle i INCF un cicle) el codi triga el mateix, per tant no hi ha cap dependència del valor inicial a WREG.

Computer Interfacing (CI) FIB. Primer parcial **28/10/2022**

Cognoms, Nom _____ **DNI** _____

Tota resposta sense justificar es considerarà nul·la !

P4. (3 punts)

4.1 (1,5 punts) Disposeu del següent codi que usa el Timer0 per a generar una interrupció periòdica cada cert temps. Dins de la RSI fa que un pin canviï periòdicament de nivell lògic (*toggle*).
Completa els 2 espais emmarcats amb el codi necessari per a que funcioni bé el programa.

```
void interrupt low_priority lowRSI()
{
   if (TMR0IF==1 && TMR0IE==1)
   {
      TMR0L=131;

      TMR0IF=0;  // cal netejar flag, per tal que no es
                 // torni a cridar RSI quan sortim

      // a cada RSI faig un canvi de nivell
      // ... al pin de Output:
      LATAbits.LATA0=!LATAbits.LATA0;
   }
}

void main(void)
{
   initOutputPinRA0();  // doneu per fet que aquesta
                        // subrutina esta ben programada

   // inicialitzacio Timer0
   T0CONbits.TMR0ON=0;
   T0CONbits.T08BIT=1;
   T0CONbits.T0CS=0;
   T0CONbits.PSA=0;
   T0CONbits.T0PS=4;
   TMR0L=131; // ajustarem periodicament aquest valor inicial de comptatge

   //inicialitzacio interrupcions

   TMR0IP=0; // establim prioritat a Low (veure capçalera RSI)
   TMR0IF=0; // netegem possible Flag que hi hagués previ (poc probable,
             // doncs s'havien executat molt poques línies de codi i
             // segurament no havia donat temps a fer cap overflow encara)
   TMR0IE=1; // habilitem font d'interrupció TMR0

   IPEN=1;   // habilitem sistema de nivells de prioritat a les interrupcions
   GIEL=1;   // habilitem el Global Interrupt de les Low
             // (la nostra interrupció és Low)
   GIEH=1;   // habilitem el Global Interrupt de les High. Cal habilitar-lo
             // també, si no no es cridaria a cap RSI (ni High ni Low)


   //start Timer0
   T0CONbits.TMR0ON=1;

   while (1);

}
```

Ens diu que el comptatge s'inicia periòdicament des d'un cert valor: aquesta és la estratègia per tal que la interrupció periòdica del Timer 0 arribi cada cert número desitjat de ticks.

Per tant, després de cada overflow, cal restablir dins la RSI el valor inicial de 131.

4.2 (1 punt) Executem el codi de la pregunta P4 en un micro que funciona amb Fosc=16 MHz. ¿Cada quant temps s'executa la interrupció de Timer0?

La fórmula per calcular el període entre RSI és:

$$T_{RSI} = T_{CS} \cdot PRE \cdot nTicks$$

T_{CS} és el període del Clock Source escollit. En el cas del codi anterior es posa el bit T0CS a 0, per tant selecciona usar el clock d'instruccions Fcy. Sabem que Fosc=16MHz, per tant Fcy=16MHz/4 = 4MHz.

El bit PSA a 0 indica que s'utilitza Prescaler. El valor del Prescaler surt del número gravat al camp T0PS=4. El valor 4 (100 en binari), segons la descripció del SFR, correspon al Prescaler 1:32 (és a dir: divideix freqüència d'entrada entre 32, o bé multiplica període per 32).

Per tant:

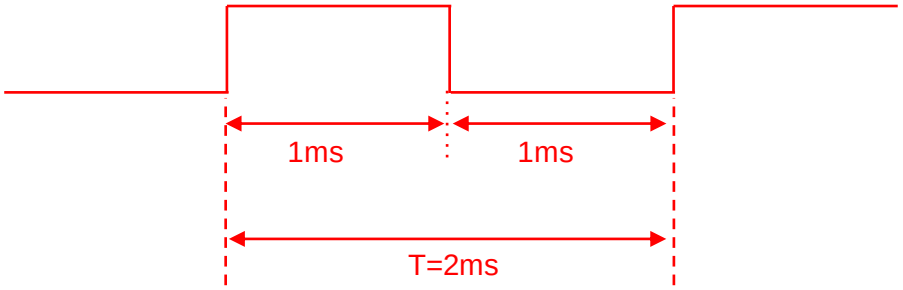
$$T_{RSI} = T_{CY} \cdot PRE \cdot nTicks = \frac{1}{4MHz} \cdot 32 \cdot nTicks$$

Per saber el número de Ticks que transcorre entre interrupcions, cal saber el valor inicial al qual restablim el Timer a cada overflow: 131 (el codi ens dóna aquest valor).
Com que el timer està configurat a 8 bits (T08BIT és 1), vol dir que es comptarà sempre de 131 fins al overflow (256). Per tant, comptem un número de ticks = 256-131 = 125.

$$T_{RSI} = \frac{1}{4MHz} \cdot 32 \cdot 125 = 1ms$$

4.3 (0,5 punts) ¿Quina és la freqüència de la senyal que surt pel pin RA0?

Sabem que la RSI s'executa cada 1ms. Dins del codi hi ha una instrucció per fer un Toggle del pin RA0. Per tant, cada 1ms la senyal de RA0 canvia de valor.
Veiem-ho gràficament:



Si la senyal té un període de T=2ms, llavors la freqüència és de $\frac{1}{2ms} = 500Hz$