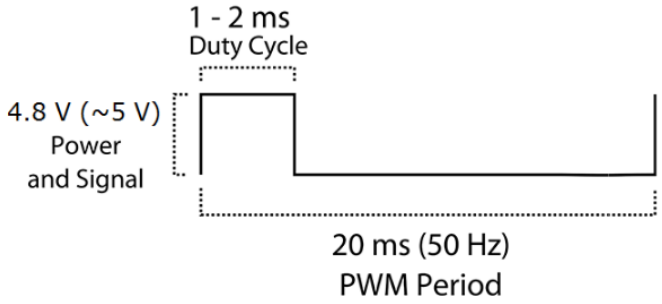
Computer Interfacing (CI) FIB. Segon parcial **10/06/2022**

Cognoms, Nom _____ **DNI** _____

Tota resposta sense justificar es considerarà nul·la !

P1. (1 punt)

Es vol controlar un servo motor mitjançant un PWM. Al datasheet del servomotor trobem la següent informació:



Position "0" (1.5 ms pulse) is middle, "90" (~2 ms pulse) is all the way to the right, "-90" (~1 ms pulse) is all the way to the left.

Podem fer servir el mòdul PWM del nostre micro per controlar aquest motor si Fosc=4MHz?

Hi han diverses maneres de demostrar que no es pot fer servir. Per exemple podem calcular quin és el període màxim de PWM que podem generar amb el CCP del nostre PIC i veure que no és prou gran per controlar el servomotor:

$$Periode_{PWM_{MAX}} = (PRX_{MAX} + 1) * 4 * T_{osc} * PRE_{MAX}$$

Si PRX_{MAX}= 255 i PRE_{MAX}= 16, el període màxim del PWM serà 0,004sec ($F_{PWM_{MIN}} = 244{,}1Hz$) insuficient per controlar el servomotor.

P2 (1 punt)

Hem connectat el CCP en mode capture cada flanc ascendent associat a un sensor que capta les voltes d'una roda. Sabem que Fosc=8MHz i T1CON=0x03 i que tot el necessari està ben configurat
El codi programat a la interrupció de CCP i a la interrupció del seu timer associat és:

```
void interrupt captura(){
      if (TMR1IE&&TMR1IF){
            ++num_overflows;
            TMR1IF=0;
      }
      if (CCP1IE&&CCP1IF){
            milisegons = calcula_temps_en_ms();
            num_overflows=0;
            CCPR_anterior=CCPR;
            CCP1IF=0;
      }
}
```

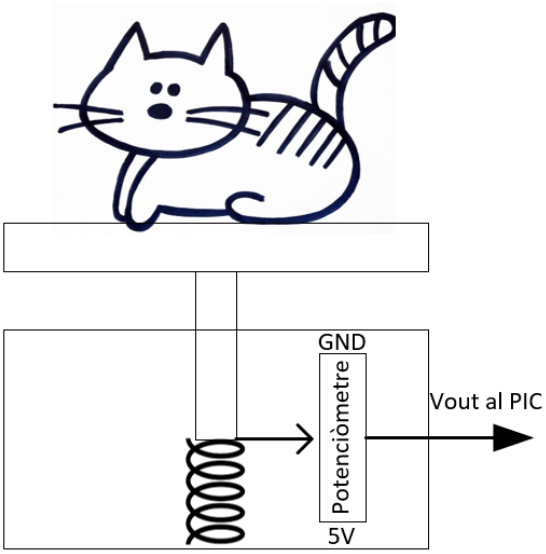
2.1 (0,5 punts) Quant de temps ha passat si en el primer flanc hem llegit un valor a CCPR1 de 1234 i en el següent flanc llegim un valor de 50 i s'ha produït un overflow al timer 1 (`num_overflows=1`)?
Amb la configuració del timer (PRE=1 i Fosc/4) sabem que un tic de timer serà 4/8M=0,5µs
Entre el primer flanc i el segon han passat $2^{16} - 1234 + 50$ tics = 64352 tics = 32,18ms

2.2 (0,5 punts) Quant de temps ha passat si en el primer flanc hem llegit un valor a CCPR1 de 1234 i en el següent flanc llegim un valor de 1235 i s'ha produït un overflow al timer 1 (`num_overflows=1`)?

Entre el primer flanc i el segon ha passat un overflow de timer i un tic: $2^{16} + 1$ tics = 65537 tics = 32,77ms

P3 (2,5 punts)

Volem fabricar una bàscula basada en un potenciòmetre per a pesar gats (entre 0g i 5kg). La idea de funcionament es pot veure a l'esquema adjunt. En repós el potenciòmetre estarà al mínim del seu recorregut, i si posem un gat de 5Kg, el potenciòmentre estarà al màxim del seu recorregut. Configurem l'AD del noste micro com ACQT<2:0>=0b100, ADCS<2:0>=0b101, VREF+=5V i VREF-=0V.



3.1 (0,75 punts) Quina resolució tenim en grams per step d'AD?

Fem servir tot el recorregut de l'AD (de 10 bits) llavors:

$$Resolució = \frac{5000g}{2^{10} - 1} = 4{,}88\ g/step$$

3.2 (0,75 punts) Quants gats podem pesar per segon si Fosc=16MHz?

Segons la configuració de l'AD:
ADCS = Fosc/16
TAD = 16/16MHz = 1us
TACQ = 8TAD = 8us

Assumint que una mostra és Tacq+12Tad = 20us, **teòricament** podem pesar 50000 gats per segon. De forma evident, a la pràctica seran molts menys.

3.3 (1 punt) Dóna una configuració de l'AD que permeti millorar la resolució suposant que només pesarem gats de fins a 2Kg (i no volem modificar la bàscula). Quina resolució aconsegueixes amb aquesta configuració?

Per millorar la resolució cal modificar els valors de referència per ajustar-los als valors de sortida que tindrà el potenciòmetre: de 0 a 2V (de 0g a 2Kg). Per fer-ho podem fer servir, per exemple, els valors fixes de referència del PIC18, configurant-los a Vref+ = 2.048V i Vref-=0V. Aquesta configuració l'aconseguim amb els registres:
ADCON1bits.PVCFG=0b10
ADCON1bits.NVCFG=0b00
VREFCON0bits.FVREN=1
VREFCON0bits.FVRS=0b10
La resolució obtinguda amb aquesta configuració serà de $\frac{2048g}{2^{10}-1} = 2g/step$

Computer Interfacing (CI) FIB. Segon parcial **10/06/2022**

Cognoms, Nom _____ **DNI** _____

Tota resposta sense justificar es considerarà nul·la !

P4. (2 punts)

Volem utilitzar el Timer6 per cridar una RSI cada milisegon de forma exacta.
El nostre micro treballa amb una Fosc=12 MHz. Considereu que les interrupcions ja estan ben configurades.

4.1 (1 punt) Un alumne proposa configurar el Timer6 ajustant els registres següents d'aquesta forma:

```
PR6 = 499;
T6CONbits.T6OUTPS=5;
T6CONbits.T6CKPS=0;
T6CONbits.TMR6ON=1;
```

¿Seria correcta aquesta configuració? *(recorda justificar la teva resposta)*

T6OUTPS=5 ⇒ El postscaler està configurat a 1:6
T6CKPS=0 ⇒ El prescaler és 1

Usem la fórmula per calcular el temps entre RSI amb Timer6:

$$T_{RSI} = \frac{4}{F_{OSC}} \cdot PRE \cdot (PR6 + 1) \cdot POST = \frac{4}{12MHz} \cdot 1 \cdot 500 \cdot 6 = 1ms$$

Podria semblar que s'aconsegueix 1ms de forma exacta. Però compte! Tenim un greu problema: el Timer6 és de 8 bits, i estem intentant posar un valor de 499 a PR6 que admet com a màxim 255.

Per tant, la configuració és INCORRECTA (no calia ni fer els números previs; només veient que el PR6 excedeix el valor màxim ja es podia contestar la pregunta).

4.2 (1 punt) Suposeu que volem mantenir el **Prescaler configurat a 1:1**. ¿Quantes possibles configuracions del Timer6 podem trobar que compleixin amb els requisits de l'enunciat? Detalla com seria cada configuració, si és que n'hi ha.

Segons la fórmula de T_{RSI}, hauríem de complir el següent:

$$0.001s = \frac{4}{12MHz} \cdot 1 \cdot (PR6 + 1) \cdot POST \Longrightarrow \boxed{3000 = (PR6 + 1) \cdot POST}$$

Tenim dues expressions a ajustar ([PR6+1] i POST), de tal forma que la seva multiplicació doni 3000.
Com que hem d'intentar trobar un número petit per a que càpiga en el registre PR6 de 8 bits, comencem provant amb els números més grans de POST.

POST=16 ⇒ PR6+1=$\frac{3000}{16}$=187,5 Dóna decimals, no podrem ajustar a 1ms de forma exacta.
POST=15 ⇒ PR6+1=$\frac{3000}{15}$=200 Es pot ajustar a 1ms de forma exacta, posant POST=15 i PR6=199
POST=14 ⇒ PR6+1=$\frac{3000}{14}$=214,29 Dóna decimals, no podrem ajustar a 1ms de forma exacta.
POST=13 ⇒ PR6+1=$\frac{3000}{13}$=230,77 Dóna decimals, no podrem ajustar a 1ms de forma exacta.
POST=12 ⇒ PR6+1=$\frac{3000}{12}$=250 Es pot ajustar a 1ms de forma exacta, posant POST=12 i PR6=249
POST=11 ⇒ PR6+1=$\frac{3000}{11}$=272,73 Dóna decimals, no podrem ajustar a 1ms de forma exacta, i a més acabem la cerca, doncs ja estem obtenint números de PR6 per sobre dels 8 bits.

Conclusió: només tenim dues possibles configuracions, $\boxed{\text{POST} = 15 \text{ i PR6} = 199}$, i $\boxed{\text{POST} = 12 \text{ i PR6} = 249}$

P5. (2 punts)

Un alumne ha de programar la unitat UART1 del PIC18F45K22 per tal de transmetre dades en mode asíncron. La Fosc és de 10MHz. Algunes de les configuracions que fa al seu codi són les següents:

```
BAUDCON1bits.BRG16 = 1;
SPBRGH1 = 1;
SPBRG1 = 3;
TXSTA1bits.TXEN = 1;
TXSTA1bits.SYNC = 0;
TXSTA1bits.BRGH = 0;
RCSTA1bits.SPEN = 1;
```

5.1 (0,8 punts) Quin és el baudrate obtingut amb aquesta configuració?

Donat que SYNC=0, BRG16=1 i BRGH=0, podem anar a la taula de les fórmules de càlcul de Baudrate i saber que s'està usant la següent: $Baudrate = \frac{F_{OSC}}{16 \cdot (n+1)}$

Com que BRG16=1, estem usant un Baudrate Generator (el valor *n* de la fórmula) de 16 bits. Es a dir, hem de concatenar els dos registres de 8 bits anomenats: <SPBRGH:SPBRG>
$n = SPBRGH \cdot 256 + SPBRG = 1 \cdot 256 + 3 = 259$

Per tant:
$$Baudrate = \frac{10MHz}{16 \cdot (259 + 1)} = 2403{,}85\ bauds$$

5.2 (0,8 punts) Si desitgéssim treballar amb una velocitat de 115200 bauds, quins canvis s'haurien de fer a les línies de codi de l'enunciat per tal que funcionés adequadament la comunicació sèrie?

Provem les diferents fórmules, per veure quina ens dóna la millor aproximació a 115200.
Cas 1 (divisor 64):
$$n = \frac{F_{OSC}}{64 \cdot Baudrate} - 1 = \frac{10MHz}{64 \cdot 115200} - 1 = 0{,}356 \cong 0$$
$$Baudrate = \frac{10MHz}{64 \cdot (0 + 1)} = 156250\ bauds$$

Cas 2 (divisor 16):
$$n = \frac{F_{OSC}}{16 \cdot Baudrate} - 1 = \frac{10MHz}{16 \cdot 115200} - 1 = 4{,}425 \cong 4$$
$$Baudrate = \frac{10MHz}{16 \cdot (4 + 1)} = 125000\ bauds$$

Cas 3 (divisor 4):
$$n = \frac{F_{OSC}}{4 \cdot Baudrate} - 1 = \frac{10MHz}{4 \cdot 115200} - 1 = 20{,}701 \cong 21$$
$$Baudrate = \frac{10MHz}{4 \cdot (21 + 1)} \cong 113636\ bauds$$

La millor aproximació és el darrer cas, amb 113636 bauds. Això ens dóna un % d'error $= \left(\frac{BR_{real}}{BR_{ideal}} - 1\right) \cdot 100 = \left(\frac{113636}{115200} - 1\right) \cdot 100 = -1{,}36\%$. Com que és menor que ±5%, donem aquesta configuració per bona.

El codi s'hauria de canviar per tal que s'utilitzi el divisor 4, i la única possibilitat en mode asíncron és: BRG16=1, BRGH=1.

Cognoms, Nom _____ **DNI** _____

Tota resposta sense justificar es considerarà nul·la !

```
BAUDCON1bits.BRG16 = 1;
SPBRGH1 = 0;
SPBRG1 = 21;
TXSTA1bits.TXEN = 1;
TXSTA1bits.SYNC = 0;
TXSTA1bits.BRGH = 1;
RCSTA1bits.SPEN = 1;
```

5.3 (0,4 punts) Amb la vostra configuració trobada a la pregunta 5.2, quin % d'error estaríem cometent respecte al baudrate ideal de 115200?

Forma part dels càlculs fets a la pregunta anterior. Com que no ens ha donat el valor de 115200 exacte, estem cometent un cert error. Si ho expressem en % respecte el baudrate ideal desitjat, llavors tenim:

% d'error $=\left(\frac{BR_{real}}{BR_{ideal}}-1\right)\cdot 100=\left(\frac{113636}{115200}-1\right)\cdot 100=-1{,}36\%.$

P6. (1,5 punts)

Comprensió dels protocols de comunicacions: UART, SPI, I2C i USB.

Indica amb una X quina de les afirmacions és certa per cada plantejament (sols hi ha una correcta).

Encert suma $\frac{1{,}5}{4}$. Error resta $\frac{1{,}5}{12}$. Blanc no afecta. En aquesta pregunta, no cal justificar les respostes.

6.1 Quins protocols disposen de transmissió diferencial?

	UART (amb els cables Tx i Rx), i USB (amb els cables D+ i D-)
X	USB (amb els cables D+ i D-)
	Cap dels quatre protocols permet transmissió diferencial
	Totes les proposicions anteriors són falses

6.2 Quin és l'avantatge que tenim si un protocol admet comunicacions full duplex?

X	Es duplica la taxa de transferencia d'informació per segon, perquè es poden transmetre i rebre dades a la vegada
	Es duplica la taxa de transferencia d'informació per segon, perquè els bits s'envien a més velocitat
	Que podem interconnectar els diferents dispositius amb una línia de dades comú
	Totes les proposicions anteriors són falses

6.3 Si en SPI, I2C i USB els dispositius s'interconnecten entre ells formant un bus…

	… qualsevol dispositiu pot parlar en qualsevol moment. Si més d'un parla a la vegada i s'interfereixen els missatges, caldrà una retransmissió per a corregir errors
	… si posem una resistència de Pull-Up a cada línia del bus, ja no tindrem problemes d'interferències entre els dispositius
X	… cal que hi hagi un Master que indiqui qui pot parlar en cada moment
	Totes les proposicions anteriors són falses

6.4 En el bus SPI cal posar resistències de Pull-Up?

	Sí, només a la línia SDI, per si ningú transmet res, que no es quedi l'input a l'aire
	Sí, una a SDI i una altra a SDO
	Cal una a cadascuna de les tres línies del bus: SDI, SDO i SCK
X	Totes les proposicions anteriors són falses