Computer Interfacing (CI) FIB. Segon parcial **19/06/2023**

Cognoms, Nom _____ DNI _____

Tota resposta sense justificar es considerarà nul·la !

P1. (1,5 punts) Per controlar la velocitat de rotació d'un motor incorporem un sensor que produirà un flanc de pujada a cada volta del motor. Connectarem aquest flanc a un CCP configurat en mode capture cada flanc ascendent. Sabem que Fosc=8MHz i T1CON=0x03 i que tot el necessari està ben configurat.
Quina serà la velocitat mínima en RPM (Revolucions per Minut) que podem capturar si no controlem els overflows de Timer1?

P2. (1.5 punts) Quin és el període màxim i mínim de PWM que podem generar amb el módul CCP del nostre PIC18 si Fosc=4MHz? Quants duty cycles diferents podem crear quan configurem el módul amb el període màxim? I amb el mínim?

P3. (1 punt) Comprensió dels protocols de comunicacions: UART, SPI, I2C.
Indica amb una X quina de les afirmacions és certa per cada plantejament (sols hi ha una correcta). Encert suma 1/4. Error resta 1/12 . Blanc no afecta. En aquesta pregunta, no cal justificar les respostes.

3.1 El bus 1-wire és

	sèrie asíncron full-duplex
	sèrie asíncron half-duplex
	sèrie síncron full-duplex
	sèrie síncron half-duplex

3.2 El bus i2c és

	sèrie asíncron full-duplex
	sèrie asíncron half-duplex
	sèrie síncron full-duplex
	sèrie síncron half-duplex

3.3 El bus spi és

	sèrie asíncron full-duplex
	sèrie asíncron half-duplex
	sèrie síncron full-duplex
	sèrie síncron half-duplex

3.4 La técnica de bit-banging consisteix en

	En la transmissió d'informació entre dispositius utilitzant perifèrics específics
	En la transmissió d'informació per programari enlloc d'utilitzar un maquinari dedicat
	Generar un clock fent servir algun dels timers present al microcontrolador
	Demanar a una IA que ens doni el codi per programar el nostre microcontrolador

P4. (1 punts) En una transmissió de dades USB 1.1 calcula quant triga a enviar-se un data packet si volem enviar 100 Bytes de dades i el bus és Low Speed (1.5Mb/s)

DATA (Data to be sent/received)

SOP	DATA0/1	DATA	CRC	EOP
8b	8b	0 a 1203B	16b	2b

Computer Interfacing (CI) FIB. Segon parcial **19/06/2023**

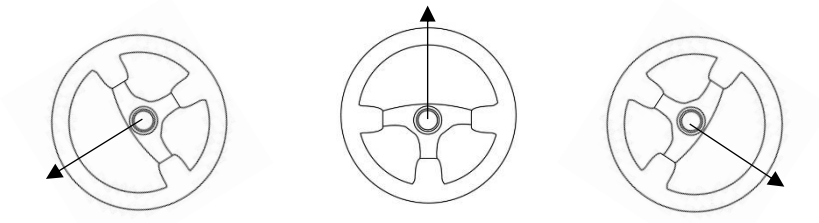
Cognoms, Nom _____ DNI _____

Tota resposta sense justificar es considerarà nul·la !

P5. (1 punts) Una marca de cotxes vol llegir el gir del volant usant un potenciòmetre. Amb el volant girat al màxim a l'esquerra (-120º) el pote donarà 1V. Girant al màxim a la dreta (+120º) donarà 4V.

Volem llegir els graus de gir del volant amb una resolució de 0.2º.

Quants bits haurà de tenir el nostre conversor A/D com a mínim per complir amb els requeriments? Les tensions de referència del ADC seran V_{REF-}=0V i V_{REF+}=5V.



Volant màxim esquerra (-120º) Pote = 1V

Volant centrat

Volant màxim dreta (+120º) Pote = 4V

P6. (1 punts) El registre ADCON2 del ADC del micro està configurat al valor hexadecimal 0xAA. El nostre PIC treballa amb una F_{OSC}=8 MHz.

P6.1. Quin és el temps total de mostreig d'una senyal d'entrada analògica?

P6.2. Amb aquest temps de mostreig, quina serà la freqüència màxima que podrem convertir amb l'ADC sense tenir problemes d'aliasing?

P7. (2 punts) Necessitem usar el perifèric UART del PIC18F45K22 per tal de transmetre dades sèrie en mode asíncron, sense paritat, amb 8 bits de dades, 1 bit d'stop i amb Baudrate=57600. La Fosc és de 4MHz.
Configura els bits necessaris dels registres TXSTA, RCSTA, BAUDCON, SPBRGH i SPBRG per a que funcioni la transmissió correctament.
Especifica clarament els càlculs que justifiquen les teves decisions i el % d'error que cometem en el Baudrate amb la configuració triada.

P8. (1 punts) Contesta breument les següents preguntes sobre la línia sèrie UART.

1) Disposa la línia sèrie d'algun sistema de detecció d'errors en els bits de dades? Si existeix aquest sistema, ¿està implementat en el perifèric del nostre PIC18F45K22?

2) Volem connectar dos PIC parlant entre ells per línia sèrie (es transmetran dades en els dos sentits). Especifica les connexions que haurem de fer entre els dos micros.

3) Què significa que la línia sèrie sigui un protocol de comunicacions Full-Duplex?

4) Quan es transmet la informació per línia sèrie, quin bit de les dades comença a enviar-se primer: el LSbit o el MSbit?

5) Quants bits en total s'envien en una trama de línia sèrie, necessaris per enviar 1 byte de dades?

Computer Interfacing (CI) FIB. Segon parcial **19/06/2023**

Cognoms, Nom _____ **DNI** _____

Tota resposta sense justificar es considerarà nul·la !

P1. (1,5 punts) Per controlar la velocitat de rotació d'un motor incorporem un sensor que produirà un flanc de pujada a cada volta del motor. Connectarem aquest flanc a un CCP configurat en mode capture cada flanc ascendent. Sabem que Fosc=8MHz i T1CON=0x03 i que tot el necessari està ben configurat.
Quina serà la velocitat mínima en RPM (Revolucions per Minut) que podem capturar si no controlem els overflows de Timer1?

Segons la configuració el timer està funcionant sense PRE i amb un CS=Fosc/4.
Com que no controlem els overflows de timer, sabem que com a màxim es pot calcular la velocitat del motor quan el temps d'una volta és més petit que el temps en el que el timer (de 16 bits) fa l'overflow:

$$Temps_{max} = 2^{16}\frac{4}{Fosc} * PRE = 32768\mu s$$

Si la volta més lenta que podem controlar triga 32768µs, això es correspon a una velocitat de:

$$\frac{1}{32768 \cdot 10^{-6}/60} = 1831\ RPM$$

P2. (1.5 punts) Quin és el període màxim i mínim de PWM que podem generar amb el mòdul CCP del nostre PIC18 si Fosc=4MHz? Quants duty cycles diferents podem crear quan configurem el mòdul amb el període màxim? I amb el mínim?

El període màxim s'obté amb el PR màxim i el PRE més gran possible:

$$T_{max} = (255+1)\cdot\frac{4}{4M}\cdot 16 = 4096\mu s$$

$$Resolution = \frac{\log[4(255+1)]}{\log(2)} = 10bits$$

Amb un PR de 255 tenim una resolució de 10 bits per fer duty cycles i es poden fer fins a 1024 dutys diferents.
El mínim s'obté amb el PR mínim i sense PRE

$$T_{min} = (0+1)\cdot\frac{4}{4M}\cdot 1 = 1\mu s$$

$$Resolution = \frac{\log[4(0+1)]}{\log(2)} = 2bits$$

Amb un PR=0 teinm dos bits de resolució i es poden fer fins a 4 dutys diferents.

P3. (1 punt) Comprensió dels protocols de comunicacions: UART, SPI, I2C.
Indica amb una X quina de les afirmacions és certa per cada plantejament (sols hi ha una correcta). Encert suma 1/4. Error resta 1/12 . Blanc no afecta. En aquesta pregunta, no cal justificar les respostes.

3.1 El bus 1-wire és

	sèrie asíncron full-duplex
X	sèrie asíncron half-duplex
	sèrie síncron full-duplex
	sèrie síncron half-duplex

3.2 El bus i2c és

	sèrie asíncron full-duplex
	sèrie asíncron half-duplex
	sèrie síncron full-duplex
X	sèrie síncron half-duplex

3.3 El bus spi és

	sèrie asíncron full-duplex
	sèrie asíncron half-duplex
X	sèrie síncron full-duplex
	sèrie síncron half-duplex

3.4 La técnica de bit-banging consisteix en

	En la transmissió d'informació entre dispositius utilitzant perifèrics específics
X	En la transmissió d'informació per programari enlloc d'utilitzar un maquinari dedicat
	Generar un clock fent servir algun dels timers present al microcontrolador
	Demanar a una IA que ens doni el codi per programar el nostre microcontrolador

P4. (1 punts) En una transmissió de dades USB 1.1 calcula quant triga a enviar-se un data packet si volem enviar 100 Bytes de dades i el bus és Low Speed (1.5Mb/s)

DATA (Data to be sent/received)

SOP	DATA0/1	DATA	CRC	EOP
8b	8b	0 a 1203B	16b	2b

Podem enviar els 100 Bytes de dades encapsulats en un únic Data Packet. Llavors, el packet que enviem conté els següents bits:

8bits SOP + 8bits Data0/1 + 100Bytes dades (800bits) + 16bits CRC + 2bits EOP = 834bits

En enviar-ho es triga:

$$834bits \cdot \frac{1s}{1{,}5Mbits} = 556\mu s$$

Computer Interfacing (CI) FIB. Segon parcial **19/06/2023**

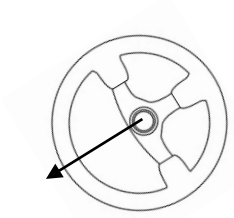
Cognoms, Nom _____ **DNI** _____

Tota resposta sense justificar es considerarà nul·la !

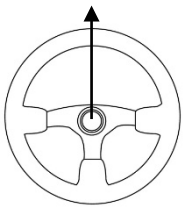
P5. (1 punts) Una marca de cotxes vol llegir el gir del volant usant un potenciòmetre. Amb el volant girat al màxim a l'esquerra (-120º) el pote donarà 1V. Girant al màxim a la dreta (+120º) donarà 4V.

Volem llegir els graus de gir del volant amb una resolució de 0.2º.

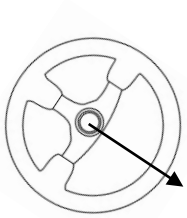
Quants bits haurà de tenir el nostre conversor A/D com a mínim per complir amb els requeriments? Les tensions de referència del ADC seran V_{REF-}=0V i V_{REF+}=5V.



Volant màxim esquerra (-120º) Pote = 1V



Volant centrat



Volant màxim dreta (+120º) Pote = 4V

Entre els dos extrems de gir del volant, tenim un rang de 120º-(-120º) = 240º. Com que es vol una resolució de 0.2º/step, llavors el número de steps que caldran en el rang de voltatge [1V…4V] serà de:

$$Num\ steps_{[1..4]} = \frac{240º}{0.2º/\text{step}} = 1200\ steps\ .$$

Com que el ADC treballa amb tensions de referència en el rang [0V... 5V], necessitarem calcular quants steps d'ADC calen en el rang complet de tensions de referència.

$$Num\ steps_{[0..5]} = \frac{1200}{3} \cdot 5 = 2000\ steps$$

Per tenir 2000 steps, calen 2001 codis. Calculem quants bits necessitem per tenir 2001 codis:

$$nBits = log_2(2001) = 10.97 bits$$

Per tant, necessitarem com a mínim un conversor ADC de 11 bits.

P6. (1 punts) El registre ADCON2 del ADC del micro està configurat al valor hexadecimal 0xAA. El nostre PIC treballa amb una F_{OSC}=8 MHz.

P6.1. Quin és el temps total de mostreig d'una senyal d'entrada analògica?

El valor 0xAA en binari és: 10101010. Si mirem la definició del ADCON2 al formulari, veurem que el camp ACQT = 0b101. I el camp ADCS = 0b010.

Aquests valors indiquen que tindrem un temps d'adquisició de 12 Tads. I un clock de conversió = Fosc/32.

$$Tad = \frac{32}{Fosc} = \frac{32}{8MHz} = 4\mu s$$

El temps total de mostreig està format pel temps d'adquisició, més 11 Tad de la conversió, més 1 Tad extra per a la descàrrega del condensador de hold.

$$Tmostreig = Tacq + 12Tad = 12 \cdot 4\mu s + 12 \cdot 4\mu s = 96\mu s$$

P6.2. Amb aquest temps de mostreig, quina serà la freqüència màxima que podrem convertir amb l'ADC sense tenir problemes d'aliasing?

Segons criteri de Nyquist, per evitar problemes d'aliasing cal que: Fmostreig > 2·Fsenyal.
Sabem que Fmostreig = 1/96us = 10416.67 Hz.

Per tant, Fsenyal < $\frac{Fmostreig}{2} \approx 5208Hz$

P7. (2 punts) Necessitem usar el perifèric UART del PIC18F45K22 per tal de transmetre dades sèrie en mode asíncron, sense paritat, amb 8 bits de dades, 1 bit d'stop i amb Baudrate=57600. La Fosc és de 4MHz.
Configura els bits necessaris dels registres TXSTA, RCSTA, BAUDCON, SPBRGH i SPBRG per a que funcioni la transmissió correctament.
Especifica clarament els càlculs que justifiquen les teves decisions i el % d'error que cometem en el Baudrate amb la configuració triada.

Primer de tot provem les 3 fòrmules per saber quina s'ajusta millor al Baudrate desitjat.
Cas divisor 64:

$$n = \frac{Fosc}{64 \cdot BR} - 1 = \frac{4MHz}{64 \cdot 57600} - 1 = 0.085 \Rightarrow arrodonim\ a\ 0.\ BR_{real} = \frac{4MHz}{64 \cdot (0+1)} = 62500$$

$$\%\ Error = \frac{BRreal - BRideal}{BRideal} \cdot 100 = \frac{62500 - 57600}{57600} \cdot 100 = 8.5\% \ \Rightarrow error\ excessiu\ (supera\ 5\%)$$

Cas divisor 16:

$$n = \frac{Fosc}{16 \cdot BR} - 1 = \frac{4MHz}{16 \cdot 57600} - 1 = 3.34 \Rightarrow arrodonim\ a\ 3.\ BR_{real} = \frac{4MHz}{16 \cdot (3+1)} = 62500$$

$$\%\ Error = 8.5\%\ (mateix\ cas\ d'abans) \ \Rightarrow error\ excessiu\ (supera\ 5\%)$$

Cas divisor 4:

$$n = \frac{Fosc}{4 \cdot BR} - 1 = \frac{4MHz}{4 \cdot 57600} - 1 = 16.36 \Rightarrow arrodonim\ a\ 16.\ BR_{real} = \frac{4MHz}{4 \cdot (16+1)} = 58823.53$$

$$\%\ Error = \frac{58823.53 - 57600}{57600} \cdot 100 = 2.12\% \ \Rightarrow error\ acceptable!\ (no\ supera\ 5\%)$$

Per tant, ens quedem amb divisor 4, que implica BRGH=1 i BRG16=1.
Bits a configurar al TXSTA: TX9=0, TXEN=1, SYNC=0, BRGH=1
Bits a configurar al RCSTA: SPEN=1
Bits a configurar al BAUDCON: BRG16=1
SPBRGH=0 i SPBRG=16

P8. (1 punts) Contesta breument les següents preguntes sobre la línia sèrie UART.

1) Disposa la línia sèrie d'algun sistema de detecció d'errors en els bits de dades? Si existeix aquest sistema, ¿està implementat en el perifèric del nostre PIC18F45K22?

Sí, el bit de paritat permet detectar errors en la transmissió de dades. No està implementat per hardware en el nostre PIC, tot i així per software el podríem calcular i enviar-lo usant el 9è bit de dades.

2) Volem connectar dos PIC parlant entre ells per línia sèrie (es transmetran dades en els dos sentits). Especifica les connexions que haurem de fer entre els dos micros.

Haurem de connectar el pin Tx del primer micro amb el pin Rx del segon. I viceversa (el Tx del segon amb el Rx del primer). Així com connectar els pins GND (Vss) per a que tots dos micros comparteixin els 0V.

3) Què significa que la línia sèrie sigui un protocol de comunicacions Full-Duplex?

Que es poden fer transmissions i recepcions de dades simultàniament.

4) Quan es transmet la informació per línia sèrie, quin bit de les dades comença a enviar-se primer: el LSbit o el MSbit?

Es comença enviant el Least Significant bit.

5) Quants bits en total s'envien en una trama de línia sèrie, necessaris per enviar 1 byte de dades?

Cal enviar 1 bit d'start + 8 bits de dades + 1 bit d'stop. Total: 10 bits

Computer Interfacing (CI) FIB. Segon parcial **13/01/2023**

Cognoms, Nom _____ DNI _____

Tota resposta sense justificar es considerarà nul·la !

P1. (2 punts)
El següent codi genera una ona quadrada amb el CCP1 d'una freqüència tal que, si la reproduïm amb un altaveu, espanta a una guineu que sempre vol robar les coses a una bona amiga nostra.

```
#define XTAL_FREQ 8000000
int semiperiod;
void main (){
        ANSELC=0;
        TRISCbits.TRISC2=0;
        CCP1CONbits.CCPxM=0b0010;
        CCPTMRS0=0;
        semiperiod=2000;
        CCPR1=semiperiod;
        TMR1GE=0;
        T1CON=0x03;
        PEIE=1; CCP1IE=1; GIE=1;
        while(1);
}
```

```
void interrupt no_robis(){
        if(CCP1IE && CCP1IF){
                CCPR1+=semiperiod;
                CCP1F=0;
        }
}
```

1.1 Pots indicar quina és la freqüència que genera aquest codi? (1p)

1.2 Una persona d'una altra facultat vol reproduir el mateix so fent servir CCPxCONbits.CCPxM=0b1100 com a configuració del mòdul CCP1. Creus que aconseguirà reproduir la mateixa freqüència que el codi del primer apartat? (1p)

P2. (1 punts)

Hem descobert que alguns pokemons són més fàcils de capturar quan la seva veu té una freqüència superior a 100KHz. Per millorar les nostres estadístiques de captura volem digitalitzar el senyal provinent d'un micròfon amb un PIC18F45K22 (FOSC= 1 MHz) i així llançar pokeballs només quan les nostres opcions de capturar el pokemon siguin prou bones. Si suposem que TAD>=1μs i TACQ>7.5μs, Quin és el millor temps de mostreig d'AD que pots aconseguir? Amb quin valor del registre ADCON2 ho aconsegueixes? Aquesta configuració de l'AD permet capturar sense aliasing el so en el rang de freqüències que ens interessa?

P3. (1 punts)

Configurem un conversor AD de 10 bits amb Vref-=1V i amb Vref+4V i ADFM=1. Quin valor en volts hi ha a Vin si després de la conversió trobem que ADRESH=0x01 i ADRESL=0x01?

P4. (1 punts)

Quants bits d'AD són necessaris si volem mesurar la distància a un objecte mitjançant un sensor que ens dóna tensions entre 2V i 4V, corresponents a distàncies d'entre 1 i 4 metres (de manera lineal) i necessitem una resolució de 0.01 metres. Les tensions de referencia són VREF– = 0V i VREF+ = 5V.

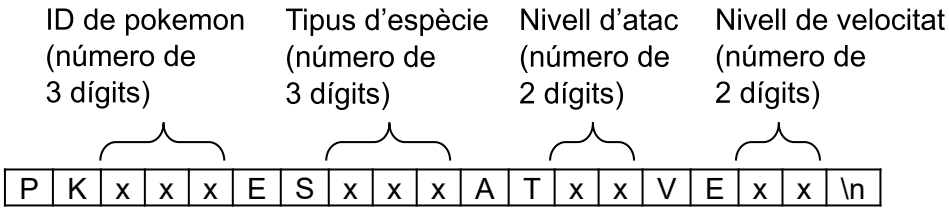
Computer Interfacing (CI) FIB. **Segon parcial** **13/01/2023**

Cognoms, Nom _____ **DNI** _____

Tota resposta sense justificar es considerarà nul·la !

P5. (2 punts)

El mateix microcontrolador PIC18F45K22 de la pregunta P2 (Fosc=1MHz) dedicat a la captura de pokemons necessita enviar la informació de cada pokemon capturat a través d'una línia sèrie UART a un ordinador central que emmagatzema les dades.
Per cada pokemon, enviarem la següent trama formada per caràcters ASCII:



Exemple:

P	K	1	2	2	E	S	0	2	1	A	T	1	1	V	E	3	9	\n

Taula ASCII *(apartat 5.2)*

048	0	064	@	080	P	096	`	112	p
049	1	065	A	081	Q	097	a	113	q
050	2	066	B	082	R	098	b	114	r
051	3	067	C	083	S	099	c	115	s
052	4	068	D	084	T	100	d	116	t
053	5	069	E	085	U	101	e	117	u
054	6	070	F	086	V	102	f	118	v
055	7	071	G	087	W	103	g	119	w
056	8	072	H	088	X	104	h	120	x
057	9	073	I	089	Y	105	i	121	y
058	:	074	J	090	Z	106	j	122	z
059	;	075	K	091	[	107	k	123	{
060	<	076	L	092	\	108	l	124	|
061	=	077	M	093	]	109	m	125	}
062	>	078	N	094	^	110	n	126	~
063	?	079	O	095	_	111	o	127	⌂

La configuració de la línia sèrie serà asíncrona, sense paritat, amb 8 bits de dades, 1 bit d'stop i amb Baudrate=4800.

5.1 Configura els bits necessaris dels registres TXSTA, RCSTA, BAUDCON i SPBRG per a que puguem enviar les dades descrites abans amb el perifèric UART1 del micro. I especifica el % d'error que cometem en el Baudrate amb la configuració triada. (1p)

5.2 Dibuixa el cronograma dels bits que surten pel pin TX1 durant l'enviament dels primers tres caràcters de la trama d'**exemple** de l'enunciat. L'estat *idle* treu un '1' lògic. Ajuda't amb la taula ASCII que adjuntem (els números estàn en **base decimal**). (0.5p)

5.3 Calcula quants pokemons per segon podríem notificar amb aquesta comunicació sèrie, si enviéssim trames contínuament, sense pausa entre trames. (0.5p)

P6. (1 punt)

Volem enviar les trames de la pregunta P5 usant un bus SPI, configurat a una velocitat de 3 Mb/s.

¿Quant temps trigarem en enviar 1 trama?
¿Amb aquest sistema de comunicacions, es podran detectar errors en la transmissió?

Computer Interfacing (CI) FIB. Segon parcial **13/01/2023**

Cognoms, Nom _____ DNI _____

Tota resposta sense justificar es considerarà nul·la !

P7. (2 punts)

Observeu aquests dos codis, orientats a saber l'amplada d'un pols que arriba a un *PIN*. Per cada un dels casos teniu la versió en C i ASM per veure que la compilació ha estat òptima.

CODI 1

```
// versió C

        TMR1GE = 0;
        T1CON = 0x13;
        while( PIN == 0);
        T_START = TMR1;
        while( PIN == 1);
        T_END = TMR1;

// versió ASM

        BCF TMR1GE
        MOVLW 13h
        MOVWF T1CON
loop1   BTFSC PIN
        BRA loop1
        MOVFF  T_START_L, TMR1L
        MOVFF  T_START_H, TMR1H

loop2   BTFSS PIN
        BRA loop2
        MOVFF T_END_L, TMR1L
        MOVFF T_END_H, TMR1H
```

CODI 2

```
// versió C

        TMR1GE = 0;
        T1CON = 0x13;
        CCP1CON   = 0x05;
        CCP2CON   = 0x04;
        CCPTMRS0 = 0x00;
        while (CCP2IF==0);
        T_START = CCPR1;
        T_END = CCPR2;

// versió ASM

        BCF TMR1GE
        MOVLW 13h
        MOVWF T1CON
        MOVLW 5
        MOVWF CCP1CON
        MOVLW 4
        MOVWF CCP2CON
        CLRF CCPTMRS0
loop    BTFSC CCP2IF
        BRA loop
        MOVFF  T_START_L, CCPR1L
        MOVFF  T_START_H, CCPR1H
        MOVFF  T_END_L, CCPR2L
        MOVFF  T_END_H, CCPR2H
```

En ambdós casos tenim connectat al xip un oscil·lador de **10 MHz**. A la versió 1, el senyal amb el pols arriba al PIN i a la versió 2 l'hem connectat als pins CCP1 i CCP2. Considereu els pins ben configurats com a entrades. El pols serà un flanc de pujada seguit d'un flanc de baixada.

7.1 En quin dels dos casos (1 o 2) podrem detectar polsos amb més precisió? Per què? (0.5p)

7.2 Quina és l'amplada mínima de pols que podrem detectar en el cas millor? (1p)

7.3 Proposa un canvi senzill al codi en C per augmentar aquesta precisió (a una sola línia). (0.5p)

Computer Interfacing (CI) FIB. Segon parcial **13/01/2023**

Cognoms, Nom _____ DNI _____

Tota resposta sense justificar es considerarà nul·la !

P1. (2 punts)
El següent codi genera una ona quadrada amb el CCP1 d'una freqüència tal que, si la reproduïm amb un altaveu, espanta a una guineu que sempre vol robar les coses a una bona amiga nostra.

```
#define XTAL_FREQ 8000000
int semiperiod;
void main (){
        ANSELC=0;
        TRISCbits.TRISC2=0;
        CCP1CONbits.CCPxM=0b0010;
        CCPTMRS0=0;
        semiperiod=2000;
        CCPR1=semiperiod;
        TMR1GE=0;
        T1CON=0x03;
        PEIE=1; CCP1IE=1; GIE=1;
        while(1);
}
```

```
void interrupt no_robis(){
        if(CCP1IE && CCP1IF){
                CCPR1+=semiperiod;
                CCP1F=0;
        }
}
```

1.1 Pots indicar quina és la freqüència que genera aquest codi? (1p)

Veiem que el codi configura el CCP en mode compare toggle associat al timer 1, que està configurat sense prescaler i Fosc/4 com a clock source. El match entre timer1 i el CCPR1 serà cada 2000 tics de timer1, llavors el pin del CCP fa toggle cada 2000 tics (que serà el semiperiode de la senyal generada):

1 tic de timer 1 = 4/8M = 0,5µs
semiperiode = 2000 tics *0,5µs = 0,001s
període = 0,002s
Freqüència=1/0,002s=500Hz

1.2 Una persona d'una altra facultat vol reproduir el mateix so fent servir CCPxCONbits.CCPxM=0b1100 com a configuració del mòdul CCP1. Creus que aconseguirà reproduir la mateixa freqüència que el codi del primer apartat? (1p)

En aquest cas, la persona de l'altra facultat, està configurant el CCP en mode PWM. Recordem que en aquest cas el CCP treballarà amb un timer de 8 bits (2, 4 o 6). Podem demostrar si és correcte que pot generar una freqüència de 500Hz trobant una configuració que ho aconsegueixi o buscant les freqüències màximes i mínimes que es poden generar amb el PWM. En aquesta solució ho demostrarem trobant una configuració que genera una freqüència de 500Hz:

1/500Hz=(PRx+1)*4*1/8M*PRE

Es pot resoldre que amb PRE =16 i PRx=249 es generen el 500Hz (s'ha de configurar el duty cycle al 50% per aconseguir que l'ona generada sigui quadrada).

P2. (1 punts)

Hem descobert que alguns pokemons són més fàcils de capturar quan la seva veu té una freqüència superior a 100KHz. Per millorar les nostres estadístiques de captura volem digitalitzar el senyal provinent d'un micròfon amb un PIC18F45K22 (FOSC= 1 MHz) i així llançar pokeballs només quan les nostres opcions de capturar el pokemon siguin prou bones. Si suposem que TAD>=1µs i TACQ>7.5µs, Quin és el millor temps de mostreig d'AD que pots aconseguir? Amb quin valor del registre ADCON2 ho aconsegueixes? Aquesta configuració de l'AD permet capturar sense aliasing el so en el rang de freqüències que ens interessa?

La millor configuració que satisfà les condicions de l'enunciat és:
TAD=2/Fosc=2µs -> ADCS=0b000
TACQ=4TAD=8µs -> ACQT=0b010
ADCON2=X0010000

El temps d'una mostra serà 11TAD+TACQ+1TCY. En la correcció també s'accepta 12TAD+TACQ i també s'ha acceptat si no s'ha tingut en compte el temps de descàrrega del condensador del circuit de sample&hold, ja que per llegir el valor de l'ADRES no cal esperar a que s'hagi descarregat (tot i que sí s'ha d'esperar per demanar a l'AD la següent mostra).
Temps=12*2+8=32µs
F Mostreig=31,250KHz, insuficient per satisfer el criteri de Nyquist. No podem asegurar que no es produeixi aliasing.

P3. (1 punts)

Configurem un conversor AD de 10 bits amb Vref-=1V i amb Vref+4V i ADFM=1. Quin valor en volts hi ha a Vin si després de la conversió trobem que ADRESH=0x01 i ADRESL=0x01?

ADRES=ADRESH<<8+ADRESL=ADRESH*256+ADRESL=257

$$ADRES = 2^N - 1 * (V_{IN} - V_{REF-})/(V_{REF+} - V_{REF-})$$

Vin=1,75V

P4. (1 punts)

Quants bits d'AD són necessaris si volem mesurar la distància a un objecte mitjançant un sensor que ens dóna tensions entre 2V i 4V, corresponents a distàncies d'entre 1 i 4 metres (de manera lineal) i necessitem una resolució de 0.01 metres. Les tensions de referencia són VREF– = 0V i VREF+ = 5V.

Necessitem una resolució de 3m/0,01metres=300 steps d'AD entre 2V i 4V.
Com que els valors de referència d'AD no són entre 2V i 4V, sinó que es troben entre 0V i 5V sabem que necessitem 750 steps a resolució completa.
$2^N = 750$ -> N=9,55 -> necessitarem 10 bits d'AD.

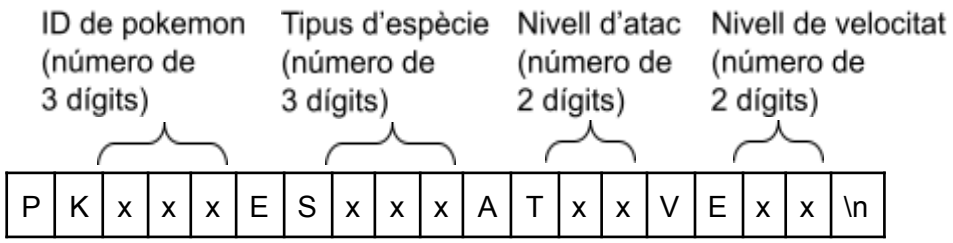
Computer Interfacing (CI) FIB. Segon parcial **13/01/2023**

Cognoms, Nom _____ **DNI** _____

Tota resposta sense justificar es considerarà nul·la !

P5. (2 punts)

El mateix microcontrolador PIC18F45K22 de la pregunta P2 (Fosc=1MHz) dedicat a la captura de pokemons necessita enviar la informació de cada pokemon capturat a través d'una línia sèrie UART a un ordinador central que emmagatzema les dades.
Per cada pokemon, enviarem la següent trama formada per caràcters ASCII:



Exemple:

P	K	1	2	2	E	S	0	2	1	A	T	1	1	V	E	3	9	\n

Taula ASCII *(apartat 5.2)*

048 0	064 @	080 P	096 `	112 p
049 1	065 A	081 Q	097 a	113 q
050 2	066 B	082 R	098 b	114 r
051 3	067 C	083 S	099 c	115 s
052 4	068 D	084 T	100 d	116 t
053 5	069 E	085 U	101 e	117 u
054 6	070 F	086 V	102 f	118 v
055 7	071 G	087 W	103 g	119 w
056 8	072 H	088 X	104 h	120 x
057 9	073 I	089 Y	105 i	121 y
058 :	074 J	090 Z	106 j	122 z
059 ;	075 K	091 [	107 k	123 {
060 <	076 L	092 \	108 l	124 |
061 =	077 M	093]	109 m	125 }
062 >	078 N	094 ^	110 n	126 ~
063 ?	079 O	095 _	111 o	127 ⌂

La configuració de la línia sèrie serà asíncrona, sense paritat, amb 8 bits de dades, 1 bit d'stop i amb Baudrate=4800.

5.1 Configura els bits necessaris dels registres TXSTA, RCSTA, BAUDCON i SPBRG per a que puguem enviar les dades descrites abans amb el perifèric UART1 del micro. I especifica el % d'error que cometem en el Baudrate amb la configuració triada. (1p)

Provem les fórmules per veure quina ens dóna un Baudrate real (BR_{real}) amb un % d'error acceptable.

Divisor 64: $n = \frac{F_{osc}}{64 \cdot 4800} - 1 \cong 2.26$	Divisor 16: $n = \frac{F_{osc}}{16 \cdot 4800} - 1 \cong 12.02$	Divisor 4: $n = \frac{F_{osc}}{4 \cdot 4800} - 1 \cong 51.08$
Arrodonim a **SPBRG=2**	Arrodonim a **SPBRG=12**	Arrodonim a **SPBRG=51**
$BR_{real} = \frac{F_{osc}}{64 \cdot (2+1)} \cong 5208.3$	$BR_{real} = \frac{F_{osc}}{16 \cdot (12+1)} \cong 4807.7$	$BR_{real} = \frac{F_{osc}}{4 \cdot (51+1)} \cong 4807.7$
$\%err = \frac{5208.3-4800}{4800} \cdot 100 \cong 8.5\%$	$\%err = \frac{4807.7-4800}{4800} \cdot 100 \cong 0.16\%$	$\%err = \frac{4807.7-4800}{4800} \cdot 100 \cong 0.16\%$
$\|\%err\|$ superior al 5%, **descartat**	$\|\%err\|$ inferior al 5%, **acceptable**	$\|\%err\|$ inferior al 5%, **acceptable**

Qualsevol dels divisors 16 ó 4 ens serveix. Escollim el divisor 16, per exemple. La taula 16-3 diu el valor que hem d'assignar a SYNC, BRG16 i BRGH. L'enunciat diu comunicació asíncrona, per tant **SYNC=0** ja està bé. Els altres dos bits podem escollir dos casos, ens quedem arbitràriament amb **BRG16=0, BRGH=1**.
A part, al registre TXSTA necessitem activar el **TXEN=1** (Transmit Enable). I assegurar-nos que TX9=0 (dades de 8 bits), tot i que aquest és el valor per defecte al SFR.
També al registre RCSTA necessitem activar el **SPEN=1** (Serial Port Enable). La resta del registre no ens afecta, doncs no estem fent recepció de dades.

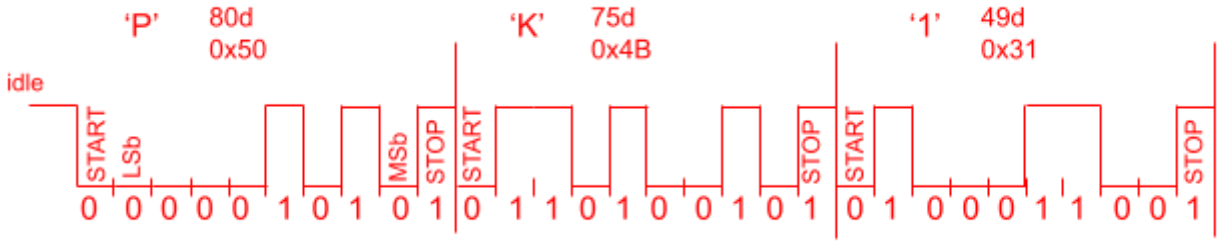
5.2 Dibuixa el cronograma dels bits que surten pel pin TX1 durant l'enviament dels primers tres caràcters de la trama d'**exemple** de l'enunciat. L'estat *idle* treu un '1' lògic. Ajuda't amb la taula ASCII que adjuntem (els números estàn en **base decimal**). (0.5p)

Primers tres caràcters: 'P' 'K' '1'. Per cada caràcter, anem a la taula ASCII, consultem el número que està en decimal com diu l'enunciat, i el convertim a binari per treure'l per la línia TX.
Per exemple, el tercer caràcter: volem enviar un caràcter '1'. Segons la taula ASCII és el valor 49 (decimal). En binari: 00110001. Compte: quan treiem les dades pel pin TX, primer surt el LSbit, i al final el

MSbit.

Recordem que per cada caràcter (byte de dades) transmès per UART, cal posar un START bit al principi, i finalitzar amb 1 STOP bit.

El cronograma quedaria així, amb un temps per cada bit de $t_{bit} = \frac{1}{4800} \cong 208.3\ \mu s$:



5.3 Calcula quants pokemons per segon podríem notificar amb aquesta comunicació sèrie, si enviéssim trames contínuament, sense pausa entre trames. (0.5p)

S'accepta que els càlculs els fem amb el baudrate ideal de 4800, o amb el real obtingut (4807.7).
Tenim un temps per cada bit de $t_{bit} = \frac{1}{4800} \cong 208.3\ \mu s$
Per cada caràcter de la trama, cal enviar 10 bits (1 start + 8 dades + 1 stop).
$t_{char} = 10 \cdot t_{bit} \cong 2083.3\ \mu s$
Per cada pokemon enviem una trama que té 19 caràcters (el '\n' del final és també un caràcter).
$t_{trama} = 19 \cdot t_{char} \cong 39.58\ ms$

Fem la inversa i sabrem el número de pokemons per segon:
$pokemons/s = \frac{1}{t_{trama}} = \frac{1}{39.58ms} \cong 25.26\ pokemons/s$

P6. (1 punt)

Volem enviar les trames de la pregunta P5 usant un bus SPI, configurat a una velocitat de 3 Mb/s.

¿Quant temps trigarem en enviar 1 trama?
¿Amb aquest sistema de comunicacions, es podran detectar errors en la transmissió?

En bus SPI surten pel pin SDO els bits de les dades, sense cap bit extra degut al protocol de transmissió.
Per tant, per enviar 1 trama sencera, caldran 19·8 bits = 152 bits.
La duració d'1 bit ve determinada per la velocitat:
$t_{bit} = \frac{1}{3 \cdot 10^6 bits/s} \cong 0.33\ \mu s$

$t_{trama} = 152\ bits \cdot t_{bit} \cong 50.67\ \mu s$

En una comunicació SPI no hi ha sistema de detecció d'errors en la transmissió.

Computer Interfacing (CI) FIB. Segon parcial **13/01/2023**

Cognoms, Nom _____ DNI _____

Tota resposta sense justificar es considerarà nul·la !

P7. (2 punts)

Observeu aquests dos codis, orientats a saber l'amplada d'un pols que arriba a un *PIN*. Per cada un dels casos teniu la versió en C i ASM per veure que la compilació ha estat òptima.

CODI 1

```
// versió C

        TMR1GE = 0;
        T1CON = 0x13;
        while( PIN == 0);
        T_START = TMR1;
        while( PIN == 1);
        T_END = TMR1;



// versió ASM

        BCF TMR1GE
        MOVLW 13h
        MOVWF T1CON
loop1   BTFSC PIN
        BRA loop1
        MOVFF  T_START_L, TMR1L
        MOVFF  T_START_H, TMR1H

loop2   BTFSS PIN
        BRA loop2
        MOVFF T_END_L, TMR1L
        MOVFF T_END_H, TMR1H
```

CODI 2

```
// versió C

        TMR1GE = 0;
        T1CON = 0x13;
        CCP1CON   = 0x05;
        CCP2CON   = 0x04;
        CCPTMRS0 = 0x00;
        while (CCP2IF==0);
        T_START = CCPR1;
        T_END = CCPR2;

// versió ASM

        BCF TMR1GE
        MOVLW 13h
        MOVWF T1CON
        MOVLW 5
        MOVWF CCP1CON
        MOVLW 4
        MOVWF CCP2CON
        CLRF CCPTMRS0
loop    BTFSC CCP2IF
        BRA loop
        MOVFF  T_START_L, CCPR1L
        MOVFF  T_START_H, CCPR1H
        MOVFF  T_END_L, CCPR2L
        MOVFF  T_END_H, CCPR2H
```

En ambdós casos tenim connectat al xip un oscil·lador de **10 MHz**. A la versió 1, el senyal amb el pols arriba al PIN i a la versió 2 l'hem connectat als pins CCP1 i CCP2. Considereu els pins ben configurats com a entrades. El pols serà un flanc de pujada seguit d'un flanc de baixada.

7.1 En quin dels dos casos (1 o 2) podrem detectar polsos amb més precisió? Per què? (0.5p)

En el cas 2: quan tinguem un pols ascendent la unitat CCP1 guardarà el valor del Timer1 al registre CCPR1 i el mateix pel flanc descendent (CCP2: Timer1 a CCPR2). Això es farà per hardware i quan poguem anirem a recollir les dades (CCPR1 i CCPR2) per copiar-les a les variables. Detectarem que ja estan les dades a punt amb el flag de CCP2IF (això no vol dir que hi hagi cap interrupció involucrada als codis).

En el cas 1 això es faria per software i presenta dos inconvenients:

- des del flanc de pujada (sortida del loop1), triguem uns quants cicles fins arribar a l'espera del flanc de baixada (entrada loop2). Si en aquest temps ja ha arribat el de baixada, no ho podrem detectar.
- en cas de que arribin interrupcions després de sortir dels loops, els timers seguiran corrent i no farem la seva còpia a les variables START i END, portant a errors.

En ambdós casos podem tenir el problema d'overflow del Timer!

7.2 Quina és l'amplada mínima de pols que podrem detectar en el cas millor? (1p)

Amb la configuració donada: Fosc=10 MHz, font del Timer1 a Fosc/4 i PREscaler=2, cada tick de timer serà de: 100ns x 4 x 2 = 800ns.
Com que el que capturem als CCPR són polsos del Timer1, el pols més petit que podem detectar seria un tick de timer (CCPR1 = K, CCPR2 = K+1).
Si heu suposat que Fosc era 8 MHz, el Timer1 faria cicles de 1us.

7.3 Proposa un canvi senzill al codi en C per augmentar aquesta precisió (a una sola línia). (0.5p)

Podríem tenir més precisió augmentant d'alguna manera la freqüència que arriba al Timer1, com que ens diuen fer-ho per codi el millor és canviar el valor al registre de configuració T1CON. Podem posar el Prescaler a 1 (és a dir no fer servir Prescaler) i augmentarem x2 la precisió.
També ens podem plantejar fer servir Fosc en comptes Fosc/4 (acceptat a la correcció), tot i que el manual diu que en modes CCP no és possible.

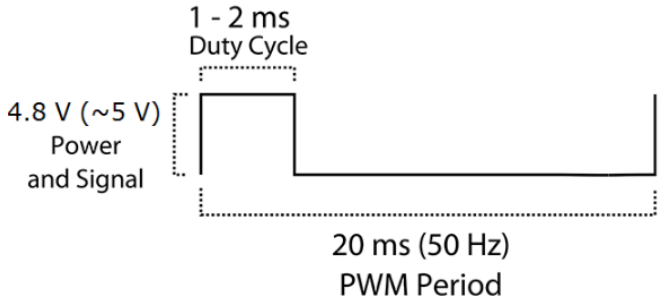
Computer Interfacing (CI) FIB. Segon parcial **10/06/2022**

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;
      }
}
```

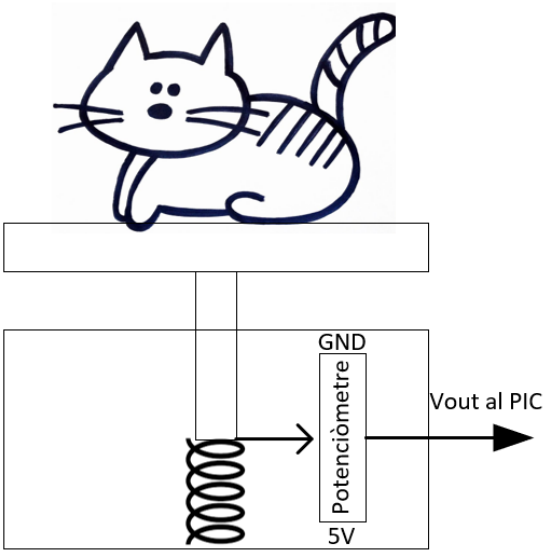
2.1 (0,5 punts) Quant de temps ha passat si en el primer flanc hem llegit un valor a CCPR1 de 1234 i en el següent flanc llegim un valor de 50 i s'ha produït un overflow al timer 1 (`num_overflows=1`)?
Amb la configuració del timer (PRE=1 i Fosc/4) sabem que un tic de timer serà 4/8M=0,5µs
Entre el primer 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

Computer Interfacing (CI) FIB. Segon parcial **13/01/2023**

Cognoms, Nom _____ DNI _____

Tota resposta sense justificar es considerarà nul·la !

P1. (2 punts)
El següent codi genera una ona quadrada amb el CCP1 d'una freqüència tal que, si la reproduïm amb un altaveu, espanta a una guineu que sempre vol robar les coses a una bona amiga nostra.

```
#define XTAL_FREQ 8000000
int semiperiod;
void main (){
        ANSELC=0;
        TRISCbits.TRISC2=0;
        CCP1CONbits.CCPxM=0b0010;
        CCPTMRS0=0;
        semiperiod=2000;
        CCPR1=semiperiod;
        TMR1GE=0;
        T1CON=0x03;
        PEIE=1; CCP1IE=1; GIE=1;
        while(1);
}
```

```
void interrupt no_robis(){
        if(CCP1IE && CCP1IF){
                CCPR1+=semiperiod;
                CCP1F=0;
        }
}
```

1.1 Pots indicar quina és la freqüència que genera aquest codi? (1p)

1.2 Una persona d'una altra facultat vol reproduir el mateix so fent servir CCPxCONbits.CCPxM=0b1100 com a configuració del mòdul CCP1. Creus que aconseguirà reproduir la mateixa freqüència que el codi del primer apartat? (1p)

P2. (1 punts)

Hem descobert que alguns pokemons són més fàcils de capturar quan la seva veu té una freqüència superior a 100KHz. Per millorar les nostres estadístiques de captura volem digitalitzar el senyal provinent d'un micròfon amb un PIC18F45K22 (FOSC= 1 MHz) i així llançar pokeballs només quan les nostres opcions de capturar el pokemon siguin prou bones. Si suposem que TAD>=1μs i TACQ>7.5μs, Quin és el millor temps de mostreig d'AD que pots aconseguir? Amb quin valor del registre ADCON2 ho aconsegueixes? Aquesta configuració de l'AD permet capturar sense aliasing el so en el rang de freqüències que ens interessa?

P3. (1 punts)

Configurem un conversor AD de 10 bits amb Vref-=1V i amb Vref+4V i ADFM=1. Quin valor en volts hi ha a Vin si després de la conversió trobem que ADRESH=0x01 i ADRESL=0x01?

P4. (1 punts)

Quants bits d'AD són necessaris si volem mesurar la distància a un objecte mitjançant un sensor que ens dóna tensions entre 2V i 4V, corresponents a distàncies d'entre 1 i 4 metres (de manera lineal) i necessitem una resolució de 0.01 metres. Les tensions de referencia són VREF– = 0V i VREF+ = 5V.

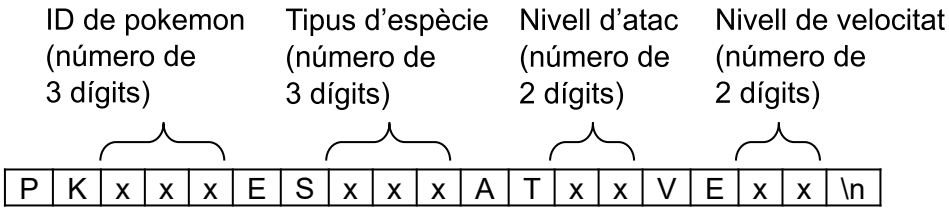
Computer Interfacing (CI) FIB. **Segon parcial** **13/01/2023**

Cognoms, Nom _____ **DNI** _____

Tota resposta sense justificar es considerarà nul·la !

P5. (2 punts)

El mateix microcontrolador PIC18F45K22 de la pregunta P2 (Fosc=1MHz) dedicat a la captura de pokemons necessita enviar la informació de cada pokemon capturat a través d'una línia sèrie UART a un ordinador central que emmagatzema les dades.
Per cada pokemon, enviarem la següent trama formada per caràcters ASCII:



Exemple:

P	K	1	2	2	E	S	0	2	1	A	T	1	1	V	E	3	9	\n

Taula ASCII *(apartat 5.2)*

048	0	064	@	080	P	096	`	112	p
049	1	065	A	081	Q	097	a	113	q
050	2	066	B	082	R	098	b	114	r
051	3	067	C	083	S	099	c	115	s
052	4	068	D	084	T	100	d	116	t
053	5	069	E	085	U	101	e	117	u
054	6	070	F	086	V	102	f	118	v
055	7	071	G	087	W	103	g	119	w
056	8	072	H	088	X	104	h	120	x
057	9	073	I	089	Y	105	i	121	y
058	:	074	J	090	Z	106	j	122	z
059	;	075	K	091	[	107	k	123	{
060	<	076	L	092	\	108	l	124	|
061	=	077	M	093	]	109	m	125	}
062	>	078	N	094	^	110	n	126	~
063	?	079	O	095	_	111	o	127	⌂

La configuració de la línia sèrie serà asíncrona, sense paritat, amb 8 bits de dades, 1 bit d'stop i amb Baudrate=4800.

5.1 Configura els bits necessaris dels registres TXSTA, RCSTA, BAUDCON i SPBRG per a que puguem enviar les dades descrites abans amb el perifèric UART1 del micro. I especifica el % d'error que cometem en el Baudrate amb la configuració triada. (1p)

5.2 Dibuixa el cronograma dels bits que surten pel pin TX1 durant l'enviament dels primers tres caràcters de la trama d'**exemple** de l'enunciat. L'estat *idle* treu un '1' lògic. Ajuda't amb la taula ASCII que adjuntem (els números estàn en **base decimal**). (0.5p)

5.3 Calcula quants pokemons per segon podríem notificar amb aquesta comunicació sèrie, si enviéssim trames contínuament, sense pausa entre trames. (0.5p)

P6. (1 punt)

Volem enviar les trames de la pregunta P5 usant un bus SPI, configurat a una velocitat de 3 Mb/s.

¿Quant temps trigarem en enviar 1 trama?
¿Amb aquest sistema de comunicacions, es podran detectar errors en la transmissió?

Computer Interfacing (CI) FIB. Segon parcial **13/01/2023**

Cognoms, Nom _____ DNI _____

Tota resposta sense justificar es considerarà nul·la !

P7. (2 punts)

Observeu aquests dos codis, orientats a saber l'amplada d'un pols que arriba a un *PIN*. Per cada un dels casos teniu la versió en C i ASM per veure que la compilació ha estat òptima.

CODI 1

```
// versió C

        TMR1GE = 0;
        T1CON = 0x13;
        while( PIN == 0);
        T_START = TMR1;
        while( PIN == 1);
        T_END = TMR1;

// versió ASM

        BCF TMR1GE
        MOVLW 13h
        MOVWF T1CON
loop1   BTFSC PIN
        BRA loop1
        MOVFF  T_START_L, TMR1L
        MOVFF  T_START_H, TMR1H

loop2   BTFSS PIN
        BRA loop2
        MOVFF T_END_L, TMR1L
        MOVFF T_END_H, TMR1H
```

CODI 2

```
// versió C

        TMR1GE = 0;
        T1CON = 0x13;
        CCP1CON   = 0x05;
        CCP2CON   = 0x04;
        CCPTMRS0 = 0x00;
        while (CCP2IF==0);
        T_START = CCPR1;
        T_END = CCPR2;

// versió ASM

        BCF TMR1GE
        MOVLW 13h
        MOVWF T1CON
        MOVLW 5
        MOVWF CCP1CON
        MOVLW 4
        MOVWF CCP2CON
        CLRF CCPTMRS0
loop    BTFSC CCP2IF
        BRA loop
        MOVFF  T_START_L, CCPR1L
        MOVFF  T_START_H, CCPR1H
        MOVFF  T_END_L, CCPR2L
        MOVFF  T_END_H, CCPR2H
```

En ambdós casos tenim connectat al xip un oscil·lador de **10 MHz**. A la versió 1, el senyal amb el pols arriba al PIN i a la versió 2 l'hem connectat als pins CCP1 i CCP2. Considereu els pins ben configurats com a entrades. El pols serà un flanc de pujada seguit d'un flanc de baixada.

7.1 En quin dels dos casos (1 o 2) podrem detectar polsos amb més precisió? Per què? (0.5p)

7.2 Quina és l'amplada mínima de pols que podrem detectar en el cas millor? (1p)

7.3 Proposa un canvi senzill al codi en C per augmentar aquesta precisió (a una sola línia). (0.5p)

Computer Interfacing (CI) FIB. Segon parcial **12/01/2022**

Cognoms, Nom _____ DNI _____

Tota resposta sense justificar es considerarà nul·la !

P1. (1 punt)

Estudiant exàmens d'altres anys hem trobat aquesta espantosa funció i la volem provar amb el nostre PIC 18F45K22 amb l'oscil·lador de 8MHz. Calcula (justificadament) quant de temps triga a executar-se.
Suposeu que la cridem al principi del main( ) sense haver configurat res que l'afecti. Podeu ignorar els temps de crida, retorn, etc.

```
void Patata (void)
{
        int i = 0;
        T0CON = 0x86;
        while (i < 25)
        {
                if (TMR0L == 250)
                {
                        TMR0L = 0;
                        i++;
                }
        }
}
```

Amb T0CON=0x86 "Engeguem" el TIMER0 amb un PRESCALER de 128 i entrada de Fosc/4 que són 500ns, per tant s'incrementarà cada 64us.
Per tant suposem que el programa farà 25 vegades una espera de 250 tics de 64us, això dóna que el temps serà de 25x250x64us=400000us=400ms=0,4s.

Caldria tenir en compte però que el TIMER0 ja estarà engegat quan s'executa el programa. Veieu al formulari que per defecte el bit TMR0on està a 1 després del RESET, per tant el primer cop, trobarem el TMR0L amb un valor entre 0 i 255, per tant el temps de la funció PATATA, serà, i d'aquí el seu nom:
1 cop (0..255)x64 us `+ 24 cops x 250 x 64 us = (384000 us .. 400320us) un valor dins d'aquest rang.

P2. (1,5 punt)

Uns companys volen usar la unitat 1 de Compare del PIC i el TIMER1 per generar una funció Delay que comptarà **microsegons** fent una espera activa. Tenen clar com serà la funció (i que de moment només servirà per valors de 0 a 65535). Suposeu Fosc = 8MHz.

```
void Delay_us (unsigned int temps)
{
        TMR1=0;
        CCPR1= temps;
        while (! CCP1IF);
        CCP1IF=0;
}
```

Creuen però que s'han oblidat inicialitzar algunes coses… Els pots ajudar omplint aquesta funció que caldrà cridar un cop al principi del main( ) per configurar-ho tot bé?

Cal arrancar el Timer1, configurar-lo a 1 us, deixar el Gate Enable a 0, associar el Timer1 a la unitat de CCP1 i posar la unitat CCP1 en mode COMPARE generant interrupció.

```
void Init_Delay_us (void)
{
        T1CON = 0b01110011; (o T1CON=0b00010011; o T1CON=0x73 o T1CON=0x13 )
        TMR1GE=0;
        CCPTMRS= 0bxxxxxx00; (o CCPTMRS=0x00 o CCPCCPTMRS &= 0xFC);
        CCP1CON= 0bxxxx1010; (o CCP1CON=0x0A)
}
```

No cal activar el IE ni res d'interrupcions perquè només usem el FLAG no la RSI. La funció Delay_us hauria de posar el flag a 0 al principi per si de cas estava ja a 1.

P3. (1 punt)

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

Encert suma 1/4. Error resta 1/12. Blanc no afecta. Només en aquesta, no cal justificar les respostes.

3.1 Pel PIC 18F45K22, amb Fosc=8MHz, el període del senyal generat amb la unitat de PWM…

	Serà com a mínim de 2048 us (microsegons).
X	Serà com a màxim de 2048 us (microsegons).
	Serà com a mínim de 125 ns (nanosegons).
	Totes les proposicions anteriors són falses.

El periode es determina amb (PR+1)x4xTosc**x**PRESCALER = (255+1)x4x125nsx16=2048us
Tosc és 125ns, el PRESCALER més gran 16, i PR major 255.
El mínim no pot ser ni 125 ni 2048 us.

3.2 Per generar una sortida en un PIN del micro amb la màxima freqüència possible…

	Usarem la unitat Capture perquè és la única lligada als Timers de 16 bits (1/3/5) i tindrem més resolució (fins a 65536 valors).
	Usarem la unitat Capture perquè els seus Timers associats (1/3/5) permeten connectar Fosc (no Fosc/4) i així tenir la freqüència base més alta.
	Podem usar les cinc unitats de Capture sincronitzadament, CCPTMRS0 = CCPTMRS1 = 0x00, així totes aniran amb el TIMER1.
X	Totes les proposicions anteriors són falses.

La unitat de Capture és per gestionar entrades, no sortides. Totes són falses.

Computer Interfacing (CI) FIB. Segon parcial **12/01/2022**

Cognoms, Nom _____ DNI _____

Tota resposta sense justificar es considerarà nul·la !

3.3 Amb el PIC 18F45K22, amb Fosc=8MHz, la unitat de PWMx farà una interrupció CCPxIF…

	Cada cop que el comparador amb PR indica que s'ha de fer reset al Timer.
	Cada cop que actualitza el Duty Cycle (còpia dels 8+2 bits de CCPRxL:CCPxCON a CCPRxH...).
	Només si el valor de Duty Cycle és menor que el de Període, per avisar de l'error.
X	Totes les proposicions anteriors són falses.

La unitat de PWM no pot generar interrupcions, mirar esquema. La idea és que queda funcionant autònomament, no té res a "comunicar" al programa principal.

3.4 Amb un PIC18F45K22 (Fosc max = 64MHz), volem generar un senyal PWM de període 10us (microsegons). La resolució (nombre de senyals diferents que podem generar a la sortida)…

X	Podrà ser major quan més gran sigui la freqüència Fosc que triem pel processador .
	Podrà ser major quan més petita sigui la freqüència Fosc que triem pel processador.
	Serà independent de la freqüència Fosc que tingui el processador.
	Totes les proposicions anteriors són falses.

Un cop determinat el període de 10 us, tindrem 10 us = (PR+1) x 4 x Tosc x PRESCALER
Aïllem (PR+1) i ens queda (PR+1) = 10 us / (4 x Tosc x PRESCALER)
Per tant (PR+1) = 10 us x Fosc / 4 x PRESCALER
Resolució és: Log2 (4 x (PR+1)) = Log2 (10us x Fosc / PRESCALER) =
Log2(10) + Log2(Fosc) - Log2(PRESCALER) = ct + Log2 (Fosc) + ct i serà creixent amb la Fosc.

P4. (1,5 punt)

Tenim la següent funció per rebre el caràcters al port sèrie d'un PIC18F45K22 funcionant amb una Fosc=4Mhz

```
void interrupt RSI (void)
{
        if (RC1IF && RC1IE){
                caracter_llegit=RCREG1;  // RC1IF bit is read-only. It gets cleared automatically
                                         //when received character is read
        }
}
```

Configura tot el que sigui necessari per a que la funció anterior rebi caràcters a 115200 bauds

La millor configuració disponible és amb BRG16=1, BRGH=1 i SPBRG=8 que s'obté un baudrate de 111111bauds. Això suposa un error d'un -3,55%. La configuració és:

```
SPEN=1;
TRISCbits.RC7=TRISCbits.RC6=1;
SPBRG=8;
BRGH=1;
BRG16=1;
SYNC=0;
RX9=0;
CREN=1;
RC1IE=1;
PEIE=1;
GEIE=1;
```

Creus que podries configurar-ho si enlloc d'una Fosc de 4MHz tinguéssim una d'1MHz?

Amb una Fosc = 1MHz la millor configuració disponible és 125000bauds. Aquest baudrate suposa un error de 8,5% respecte al baudrate desitjat. Amb errors tan grans es produeix framing error i no es pot fer servir.

P5. (1,5 punt)

Contesta breument a les següents preguntes sobre busos de comunicació:

Computer Interfacing (CI) FIB. Segon parcial **12/01/2022**

Cognoms, Nom _____ DNI _____

Tota resposta sense justificar es considerarà nul·la !

Quins fils componen un cable USB 2.0? L'USB és síncron o asíncron?

En un cable USB 2.0 podem trobar: GND (negre), 5V (vermell), D+ (verd) i D- (blanc).
La transmissió USB és asíncrona

Quines senyals (fils) podem trobar en una comunicació SPI? L'SPI té control d'errors?

A l'SPI trobem els senyals
MOSI - Master Output, Slave Input
MISO - Master Input, Slave Output
SCLK - Clock
SS - Slave select. Opcional segons el tipus de connexió

L'SPI no disposa de control d'errors

Com que el PIC18F45k22 no té hardware específic per a implementar el 1-wire, no podem fer servir el nostre micro amb aquest tipus de comunicacions. Hi estàs d'acord amb aquesta afirmació?

Fals. Tot i no tenir hardware específic es pot implementar mitjançant bit banging, és a dir, fent servir el firmware per controlar els pins implicats en la comunicació.

Què vol dir que un protocol de comunicació fa servir una senyal diferencial per enviar les dades? Quin és el principal avantatge de la senyal diferencial?

En transmissió diferencial s'envien les dades de forma complementaria fent servir un parell de senyals (les dades per un fil i les dades negades per un altre). El circuit de recepció respon a la diferència entre les dos senyals enviades.
La immunitat al soroll és el principal avantatge del senyal diferencial. Si s'introdueix soroll durant la transmissió, aquest quedará molt atenuat al realitzar la diferència al punt de recepció

Una comunicació I2C sempre comença amb l'enviament per part del master de la condició d'start seguida de 8 bits. Per a què serveixen aquest 8 bits?

Són els 7 bits d'adreça i el bit de Read/Write

P6. (1 punt)
Hem configurat el conversor AD del PIC amb les següents tensions de referència: V_{REF-} = 1V i V_{REF+} = 5V. Fosc és 12MHz, i el bit ADFM està a 0 (justificat a l'esquerra). Li posem un voltatge constant a l'entrada, el convertim (respectant les restriccions de Tad i Tacq) i obtenim als registres de resultat els següents valors: ADRESH = 0xCB, i ADRESL = 0xC0.
Quina tensió analògica (en Volts) tenim a l'entrada del conversor AD?

El primer que fem és obtenir el valor de 10 bits de la conversió. Com que sabem que el valor està justificat a l'esquerra, obtenim els 10 bits més alts entre els registres ADRESH i ADRESL.

ADRESH	ADRESL
11001011	**11**000000

Per tant, el valor de 10 bits és: 1100101111, equivalent a 815 en decimal.
Segons la fórmula per calcular la dada obtinguda per un convertidor AD, si aïllem Vin ens dóna:
Vin = [Dout / (2^N - 1)] * (Vref+ – Vref-) + Vref-

Sabem que el nostre PIC 18F45K22 té un ADC de 10 bits, i que V_{REF-} = 1V i V_{REF+} = 5V. Per tant:
Vin = [815 / 1023] * (5 – 1) + 1 = 4.187V

P7. (1,5 punts)
La revista "*Distorsió*" ha decidit emprendre una nova idea de negoci i fabricar guitarres elèctriques. Cadascuna de les 6 cordes té un sensor per a donar-nos informació de l'àudio de la corda. Cada corda es llegeix per un canal AD diferent, i es va fent multiplexació en el temps per a llegir tots els canals cíclicament amb igual temps de mostreig per cadascun (ch1-ch2-ch3-ch4-ch5-ch6-ch1-ch2- etc.).
Suposeu que la freqüència més elevada que genera l'instrument és de 4kHz.
S'ha utilitzat el PIC18F45K22 per aquesta aplicació. La Fosc és de 16MHz. Han configurat ACQT<2:0>=0b100, ADCS<2:0>=0b010. Si us calen, assumiu uns requisits de Tacq ≥ 7,45us i Tad ≥ 1us.

7.1) Calcula el temps total de mostreig d'un valor analògic.

Sabem que ADCS<2:0>=0b010, per tant Tad = 32 / Fosc = 32 / 16MHz = 2us (compleix amb mínim de 1us)
Sabem que ACQT<2:0>=0b100, per tant Tacq = 8 Tad = 8 * 2us = 16us (compleix amb mínim de 7,45us)

El temps total de mostreig d'1 valor engloba la fase d'adquisició, la de conversió (11 Tad) i després tenim 1 Tcy (descàrrega del condensador). Però habitualment simplifiquem arrodonint a temps d'adquisició + 12 Tad. Per tant:

Tmostra = Tacq + Tconv = 8 Tad + 12 Tad = 20 * 32 / 16MHz = 40us

7.2) Si ens fixem en una corda en concret qualsevol, calcula quantes mostres per segon llegirem d'aquesta corda.

La freqüència de mostreig total, comptant les mostres de totes les cordes, seria:
Fmostreig = 1 / Tmostreig = 1 / 40us = 25000 Hz

Ara bé, ens demanen per les mostres per segon només d'una corda en concret. Com que hi ha sis cordes, les mostres per segon s'han de repartir entre les 6 cordes. Per tant:
Fmostreig_1corda = Fmostreig / 6 ≈ 4167 Hz
7.3) Aquesta configuració permetrà adquirir l'àudio de la guitarra sense que es produeixi *aliasing*? Si no ho permet, podries fer algun canvi de configuració del perifèric per tal que funcionés bé?

Segons el criteri de Nyquist, per evitar *aliasing* hem d'adquirir mostres a una freqüència com a mínim del

Cognoms, Nom _____ DNI _____

Tota resposta sense justificar es considerarà nul·la !

doble de la freqüència més alta que genera l'instrument. L'enunciat ens diu que la més alta serà de 4kHz.

Per tant, hauríem de mostrejar cada corda a una freqüència de 8kHz. La freqüència de mostreig de cada corda hem calculat que serà de 4,17kHz aprox. Podem concloure que l'àudio mostrejat tindrà *aliasing*. Per tant la guitarra sonarà distorsionada, i la revista "*Distorsió*" haurà de demanar ajut a un/a Fiber per a dissenyar bé el seu producte.

Per tal que funcionés bé, caldria que cada corda agafés mostres almenys a 8kHz. Per tant, entre totes les cordes, hauríem de tenir:
Fmostreig ≥ 8kHz * 6 = 48kHz
Tmostra ≤ 1 / 48kHz = 20,83us
20 * x / 16MHz ≤ 20,83us x ≤ 16,67

Per exemple, la opció **ADCS<2:0>=0b101** compleix el requisit doncs divideix Fosc entre 16.

Amb el nou ADCS, tindrem Tad = 16 / Fosc = 16 / 16MHz = 1us (compleix amb mínim de 1us)
Comprovem que l'anterior ACQT segueixi sent vàlid: Tacq = 8 Tad = 8 * 1us = 8us (compleix amb mínim de 7,45us).
Tmostra = 8 Tad + 12 Tad = 20 * 16 / 16MHz = 20us Fmostreig = 1 / 20us = 50000 Hz
Fmostreig_1corda = Fmostreig / 6 ≈ 8333 Hz

La nova freqüència de mostreig de cada corda serà de 8,33kHz aprox, complint amb el criteri de Nyquist.

P8. (1 punt)
Una prestigiosa farmacèutica catalana, ha dissenyat una vacuna contra el Covid-19. Com a control de qualitat, cada vial és mesurat per saber quant líquid s'hi ha introduït. Disposem d'un sensor que ens dóna tensions entre 2V i 4V, corresponents a volums d'entre 0 i 3 mil·lilitres (de manera lineal). El senyal analògic provinent d'aquest sensor és mostrejat amb l'AD d'un PIC18F45K22. Les tensions de referencia són V_{REF-} = 0V i V_{REF+} = 5V.
Necessitem obtenir una resolució de 0.01 mil·lilitres en els valors llegits amb el AD.

¿Quants bits són necessaris per al conversor AD?
¿Són suficients els bits del conversor del nostre PIC, o caldria un conversor AD de més bits?

Primer calculem el número d'esglaons necessaris en la escala de mil·lilitres:
esglaons = (valor_max – valor_min) / resolució = (3ml – 0ml) / 0.01ml = 300 esglaons

Això vol dir que entre 2V i 4V hem de mesurar 300 esglaons amb el ADC.
Amb una regla de tres, podem calcular quants esglaons necessitem entre les tensions de referència mínima i màxima:

esglaons entre V_{REF-} i V_{REF+} = [300 / (4V – 2V)] * (5V – 0V) = 750 esglaons

El número de codis diferents necessaris serà de 751 codis.

Per tant, podem calcular el número de bits:

Nbits = $\log_2 751 \approx 9{,}55$ bits

Com que el nostre PIC18F45K22 té un ADC de 10 bits, sí que serà adequat per a obtenir la resolució demanada.