UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Facultat d'Informàtica de Barcelona

Enunciat de la pràctica de laboratori

Lab 3:
Display 7 segments

L3. Display 7 segments

L3(A) Multiplexat en el temps del display

L3(B) Interrupcions generades per botons

1 Objectius

L'objectiu d'aquesta pràctica és controlar un grup de 4 displays de 7-segments de manera que el microcontrolador pugui presentar informació a través de les seves sortides.

Durant la primera part de la pràctica L3(A), farem servir botons d'entrada i consultarem el seu estat pel mètode d'enquesta (polling). Segons l'estat dels botons anirem manipulant els 4 displays per visualitzar un número mitjançant la tècnica de multiplexat en el temps.

Durant la segona part de la pràctica L3(B), l'objectiu serà la programació de les rutines de servei a les interrupcions (RSI). Això ens permetrà no haver de consultar constantment l'estat dels botons d'entrada. Amb aquest mètode es podrà generar una interrupció al prémer (o al deixar anar) el botó de tal manera que el microcontrolador executi codis específics. Per fer-ho, s'haurà de comprendre la configuració dels diferents paràmetres que intervenen en la gestió de les interrupcions (habilitació, prioritats, flancs d'activació, etc.).

En acabar la pràctica l'alumne serà capaç de:

- manegar informació tècnica donada pel fabricant de la placa de desenvolupament EasyPIC v7 que utilitzem a les pràctiques.
- cercar la informació necessària dins dels manuals de referència.
- aprofundir el coneixement dels PORTS d'E/S.
- controlar l'activació d'un display de 7-segments per visualitzar un valor desitjat.
- entendre el concepte de multiplexat en temps, per aconseguir mostrar un nombre de diversos dígits al conjunt de 7-segments.
- solucionar problemes d'implementació com el *ghosting* que no són visibles durant la simulació
- entendre la diferència entre enquesta o interrupció per la gestió dels botons
- utilitzar els botons de forma que executin codi diferent quan estan premuts, quan no estan premuts, quan es produeix un flanc de pujada o quan es produeix un flanc de baixada
- entendre com configurar interrupcions en el PIC18F45K22
- entendre com implementar les rutines de servei a la interrupció
- entendre el concepte de flag de la interrupció
- entendre la utilitat d'activar o desactivar interrupcions

2 Introducció

La placa de desenvolupament EasyPIC v7 (Fig.1) disposa de 4 displays de 7 segments que ens permetran mostrar informació de manera senzilla, emprant els ports d'E/S.

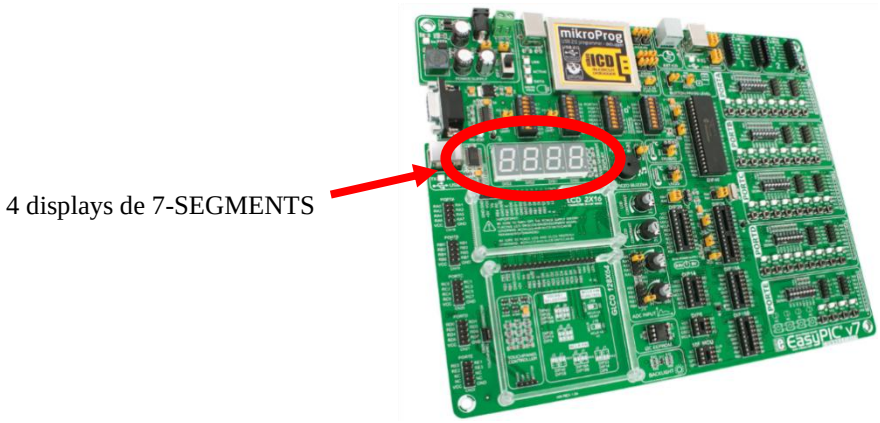

Fig. 1. Detall dels displays de 7 segments a la plataforma de desenvolupament EasyPIC v7.

Aquests 4 displays de 7 segments estan connectats amb una configuració anomenada "Càtode Comú", que vol dir que els 8 leds del display 7-segments (7 per crear la figura i un pel punt decimal) comparteixen una connexió a massa, i per tant són actius a "1". La figura 2 mostra la connexió dels quatre displays extreta del manual de la placa.

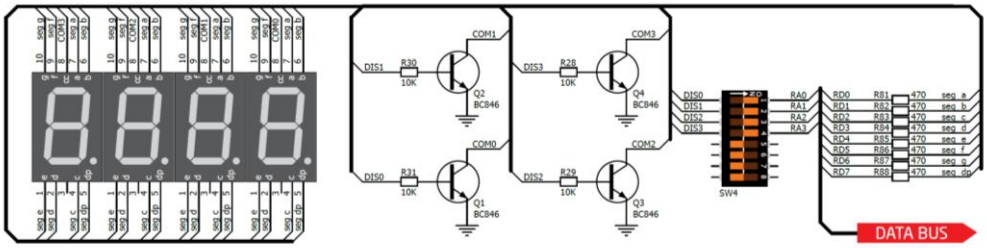

Fig. 2. Connexió a la placa dels 4 7-segments.

Observant les connexions a la placa podem concloure que:

- L'activació dels leds del display depèn del PORTD.
- El PORTD està connectat als quatre displays simultàniament.
- La selecció del display actiu es fa mitjançant el càtode comú, amb uns senyals anomenats DIS0, DIS1, DIS2 i DIS3.
- Els senyals DISx es poden connectar al PORTA mitjançant el switch SW4.

Finalment, si observem amb detall el circuit de selecció del display actiu, per selecció del càtode comú, veiem que el regeix un transistor tipus NPN i que s'activarà per "1", és a dir, quan posem un "1" a la senyal DISx s'activa el corresponent display.

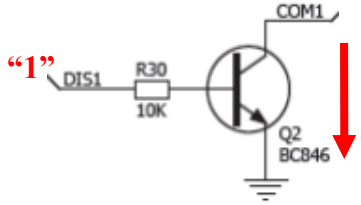

Fig. 3. Circuit de selecció del càtode.

Per visualitzar un valor en el conjunt dels 4 displays 7 segments, es fa de forma seqüencial. És a dir, es selecciona un únic display DIS0, DIS1, DIS2 o DIS3 amb el PORTA i tot seguit s'envia pel PORTD el valor a visualitzar al display actiu. A continuació s'espera una mica, es selecciona el següent display i se li envia el nou dígit i així successivament. Aquesta tècnica s'anomena "multiplexat en el temps". Si s'espera molt temps entre un dígit i el següent es veuran displays que s'encenen i s'apaguen donant una resultat poc atractiu. Si s'espera poc temps es barrejaran els valors pintats al conjunt. Teniu més informació disponible a Atenea a l'*Annex: info7Seg.pdf*. Llegiu-la amb atenció ja que serà una part molt important de la pràctica. En particular, mireu el punt 3 del document on es fa referència al problema del ghosting, que és un problema d'implementació que haureu de solucionar al laboratori.

A continuació s'inclouen una sèrie de passos per simular els 4 displays de 7-segments a Proteus:

2.1- Incloure l'arxiu `config.h` al projecte de Proteus. Aquest arxiu està disponible a Atenea.

2.2- Configurar la simulació del PIC18F45K22 a Proteus perquè funcioni a 8MHz de tal manera que la simulació i la placa EasyPicv7 funcionin a la mateixa velocitat de rellotge. Per fer-ho heu d'anar a la pestanya de l'esquemàtic i fer doble click sobre el microcontrolador. S'obrirà una finestra per editar el component i s'haurà de modificar el valor de Processor Clock Frequency a 8MHz.

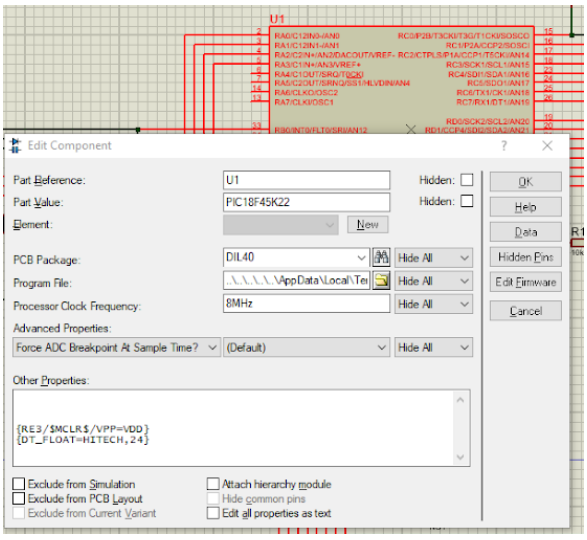

Fig. 4. Captura de pantalla pel canvi de la velocitat del clock.

2.3-Dissenyar l'esquema elèctric sobre Proteus. Farem el disseny amb Proteus amb el microcontrolador PIC18F45K22 i hi inclourem el bloc de 4 displays de 7-segments "7SEG-MPX4-CC" que funciona en càtode comú, tal com necessitem. Per gestionar la selecció dels displays, canviarem el circuit real amb els transistors NPN per portes NOT. A nivell lògic funcionen igual (al posar un "1" a la porta, surt un "0" del negador que fa que el display corresponent estigui connectat a "0" ó terra i per tant es tanca el circuit i es pot il·luminar el display) i estalviem que Proteus hagi de fer la simulació d'un circuit analògic amb els transistors, que ralentitzaria molt la simulació.

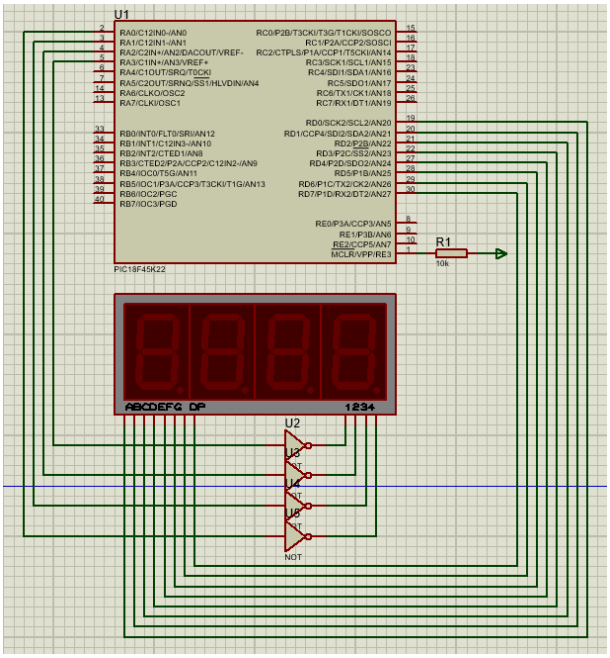

Fig. 6. Model per l'esquema de Proteus.

2.4-Podeu implementar l'espera entre un dígit i el següent amb `__delay_us(x)` on `x` és el temps d'espera en microsegons o `__delay_ms(x)` per fer esperar `x` milisegons (vigileu que té **2** guions baixos al davant). Per fer servir aquests delays necessiteu afegir `#define _XTAL_FREQ 8000000` al principi del vostre codi per indicar a la funció que la freqüència del clock és de 8MHz. Consulteu els arxius sobre la Freqüència Fosc del PIC *infoFosc.pdf* i *infoGraficIncludesXC8_PIC18F45K22.pdf* que trobareu a Atenea per a més informació.

3 Treball Previ Multiplexat en el temps L3 (A)

Fer un programa en llenguatge C amb les següents funcionalitats:

3.1- Connecteu un botó al pin RB7 i llegiu el seu estat per enquesta. Cada cop que es detecti un flanc de pujada en aquest pin es generarà un número aleatori entre 0 i 9999.

3.2-Connecetu un botó al pin RB0 i llegiu el seu estat per enquesta. Cada cop que es detecti un flanc de pujada en aquest pin es rotaran els dígits del display cap a la dreta de tal manera que si hi havia el número 7259, ara aparegui el 9725.

3.3-Connecetu un botó al pin RB2 i llegiu el seu estat per enquesta. Cada cop que es detecti un flanc de pujada en aquest pin es rotaran els dígits del display cap a l'esquerra de tal manera que si hi havia el número 7259, ara aparegui el 2597.

3.4-Connecetu un botó i llegiu l'estat del pin RB1 per enquesta. Cada cop que es detecti un flanc de **baixada** en aquest pin es farà una ordenació dels números, de tal manera que el número més baix del 4 displays s'ubiqui al display de l'esquerra del tot, el següent valor més baix al segon display començant per l'esquerra, el següent valor més baix al tercer display, i el valor més alt al display de la dreta.

3.5-Afegiu un comentari al vostre codi on s'indiqui el temps que triga el vostre codi en pintar els 4 displays de 7-segments.

Per generar números aleatoris podeu fer servir `srand()` i `rand()`, i necessitareu incloure la llibreria `#include <stdlib.h>`. Trobareu informació sobre aquestes i altres rutines i com utilitzar-les al manual de referència del compilador: "MPLAB XC8 C Compiler User's Guide", secció "Appendix A. Library Functions". Podreu trobar l'explicació sobre `rand`, `srand`, així com les rutines de `delay` `__delay_ms` i `__delay_us`. Trobareu aquest document a Atenea amb el nom *xc8.pdf*

Entregeu al Racó el projecte Proteus amb el diagrama esquemàtic i el codi en C realitzar

4 Rúbrica treball Previ Multiplexat en el temps L3 (A)

	Iniciat (0-2.5 punts)	En desenvolupament (2.5-5.0 punts)	Aconseguit (5.0-7.5 punts)	Exemplar (7.5-10 punts)
Botons (1 punt):	Hw mal dissenyat i funcionament erroni	Hw mal dissenyat o funcionament erroni	Funcionament correcte però mal ús de recursos	Funciona perfectament
Flancs (2 punts):	Mala detecció de flancs	Detecció de flancs correctament implementada però no fa servir funció independent o estructura	Detecció de flancs en una funció independent però codi mal estructurat per la gestió de múltiples situacions	Detecció de flancs en una funció independent amb el codi estructurat adequadament per la gestió de flancs de pujada o baixada
Números aleatoris (2 punts):	No hi ha cap generació de números aleatoris	La seqüència de números aleatoris és sempre la mateixa	La seqüència de números aleatoris és repeteix en algunes condicions	A cada execució apareix una sèrie diferent de números aleatoris
Ordenar (2 punts):	No s'ordena	L'ordenació no és correcta	S'ordenen correctament però el codi és molt poc eficient i poc modular	Es fan servir els recursos del microcontrolador perfectament
Temps execució (1 punt)	No hi ha valor ni informació dels temps d'execució de la rutina	La informació no és correcta	La informació és correcta però no hi ha informació que faci entendre el per què d'aquest valor	El valor i l'explicació corresponent són sintètics i correctes
Display 7-segments (2 punts):	Hw mal dissenyat	Visualització incorrecta als displays	Codi poc eficient amb especial èmfasis en com es mapegen els números als displays i el multiplexat en el temps	Funciona perfectament

5 Treball Previ Interrupcions per botons L3 (B)

El programa a desenvolupar en aquesta segona part de la pràctica consisteix en visualitzar al display de 7-segments un número mitjançant els botons associats a les interrupcions INT0, INT1 i INT2, per tant haureu d'escriure el codi d'atenció a la interrupció corresponent. El botó INT0 el farem servir per generar un número aleatori i ordenar els números. El botó de la INT1 el farem servir per rotar els números cap a la dreta, i el botó de la INT2 el farem servir per rotar-los cap a l'esquerra. El codi a implementar ha de complir els següents requeriments:

5.1- Configurar un polsador associat a la interrupció externa INT0 d'alta prioritat que quan es detecti un **flanc de pujada** generi un número aleatori entre 0 i 9999. Aquesta pulsació haurà d'habilitar les interrupcions de baixa prioritat INT1 i INT2.

5.2-Si es torna a prémer el polsador de la INT0, **s'ordenaran tots els números** que hi ha als displays posant el número més petit al display de l'esquerra i el més gran al display de la dreta de igual manera a com es va fer a la pràctica anterior L2(a). En aquest cas es deshabilitaran les interrupcions de la INT1 i INT2.

5.3-Si es torna a apretar el polsador associat a la INT0 es tornarà a generar i mostrar un nou número aleatori entre 0 i 9999 i s'habilitarà un altre cop la INT1 i INT2.

5.4- Configurar un polsador associat a la interrupció externa INT1 de baixa prioritat que quan es detecti un **flanc de baixada** desplaci els números una posició cap a la dreta, de tal manera que si hi havia el número 6490, i aparegui el 0649.

5.5- Configurar un polsador associat a la interrupció externa INT2 de baixa prioritat que quan es detecti un **flanc de baixada** desplaci els números una posició cap a l'esquerra, de tal manera que si hi havia el número 6490, i aparegui el 4906.

5.6- Les interrupcions són un punt crític a la programació de microcontroladors degut a que la seva natura asíncrona fa que sigui difícil debugar, tracejar i testejar el codi. Per tant, és necessari ser meticulós en la fase de desenvolupament per evitar errors crítics. En el nostre PIC farem algunes proves per veure que hem programat adequadament. En primer lloc posarem un software breakpoint a la rutina de servei a la interrupció que gestiona INT0 i mirarem que l'execució del codi s'atura correctament i s'executa el codi corresponent fent servir el botó F10 per avançar línia a línia mentre debuguem. Indiqueu al codi, al costat del codi d'atenció a la interrupció un comentari amb el millor i el pitjor temps d'execució del vostre codi de la interrupció. Podeu posar dos breakpoints diferents i amb el botó de continuar la simulació podreu avançar d'un a l'altre. A la part baixa de la finestra de Proteus veureu el temps transcorregut.

5.7- Posarem un hardware breakpoint al pin INT0 que s'activi amb el flanc de pujada. Podeu veure el vídeo tutorial *Breakpoints* i *Hardware Breakpoints* en la web de Proteus https://www.labcenter.com/tutorials/ o a youtube amb l'usuari Labcenter Electronics Ltd https://www.youtube.com/watch?v=qFkcELwgBuM.

5.8- Estresseu el sistema per a forçar l'aparició de bugs no detectats. Aquesta tècnica consisteix en provocar moltes més interrupcions per segon que les que s'esperarien en funcionament normal. Per a tal fi, substituïu els polsadors per un generador de funcions que generi interrupcions a una freqüència molt elevada o un generador de polsos que trobareu al menú Generators. Podeu seleccionar *Analogue Types = Pulse, Pulse (High) Voltage = 5V*, i anar canviant el valor de *Frequency*. Comproveu fins a quina freqüència màxima pot arribar el generador abans de que el vostre codi deixi de funcionar correctament. Indiqueu el valor en un altre comentari al vostre codi.

6 Rúbrica treball Previ Interrupcions per botons L3 (B)

	Iniciat (0-2.5 punts)	En desenvolupament (2.5-5.0 punts)	Aconseguit (5.0-7.5 punts)	Exemplar (7.5-10 punts)
Botò INT0 (1.5 punts):	Hw mal dissenyat i funcionament erroni	Hw mal dissenyat o funcionament erroni (flanc de pujada)	Funcionament correcte però mal ús de recursos o mala configuració	Funciona perfectament i les interrupcions associades estan ben gestionades
Botò INT1 (1.5 punts):	Hw mal dissenyat i funcionament erroni	Hw mal dissenyat o funcionament erroni (flanc de pujada)	Funcionament correcte però mal ús de recursos o mala configuració	Funciona perfectament i les interrupcions associades estan ben gestionades
Botò INT2 (1.5 punts):	Hw mal dissenyat i funcionament erroni	Hw mal dissenyat o funcionament erroni (flanc de baixada)	Funcionament correcte però mal ús de recursos o mala configuració	Funciona perfectament i les interrupcions associades estan ben gestionades
Temps execució (1.5 punts)	No hi ha valor ni informació dels temps d'execució de la rutina	La informació no és correcta	Només hi ha informació del millor o el pitjor temps però no està justificat, o no es dona informació de quan deixa de funcionar el codi degut a interrupcions del generador de polsos	Els valors i les explicacions corresponents són sintètics, estan justificats i són correctes
Display 7-segments (2.0 punts):	Hw mal dissenyat	Visualització incorrecta als displays	Codi poc eficient	Funciona perfectament
Ordenar (2.0 punts):	No s'ordena	L'ordenació no és correcta	S'ordenen correctament però el codi és molt poc eficient	Es fan servir els recursos del microcontrolador perfectament