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 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. Amb aquesta informació anirem seleccionant cada un dels 4 displays per pintar 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 apretar (o al deixar anar) el botó de tal manera que s'executin codis específics a dintre del microcontrolador. 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 fer-hi un dibuix 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 apretats, quan no estan apretats, 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.

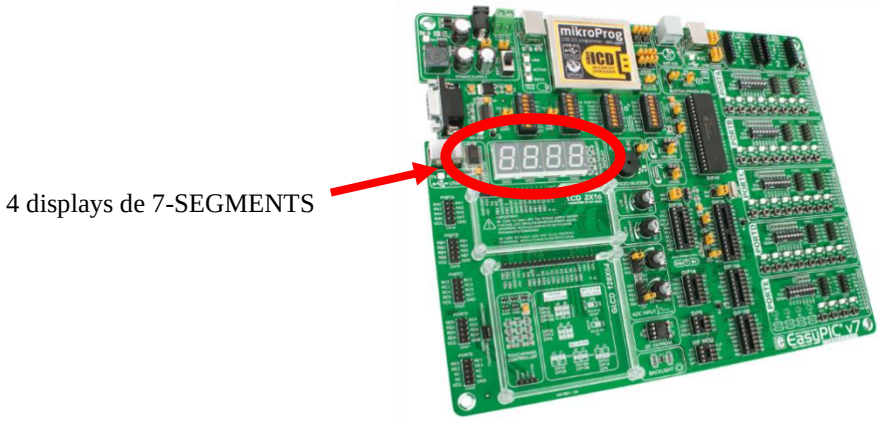

Fig. 1. Detall dels displays de 7 segments a la plataforma de desenvolupament EasyPIC v7.

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.

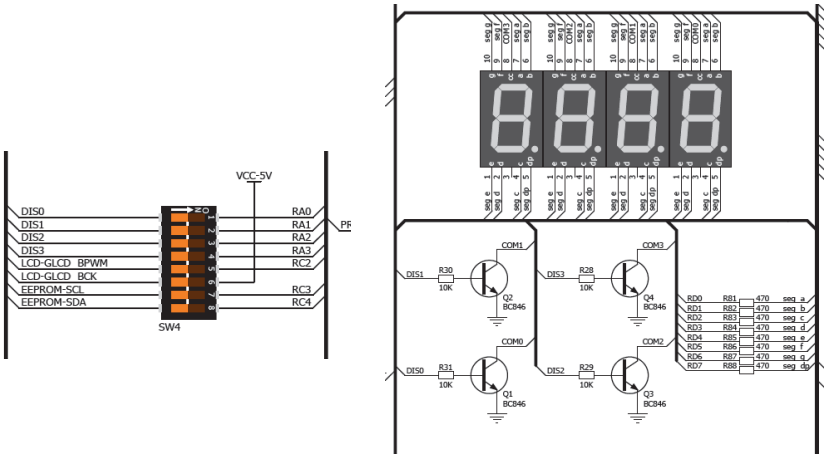

Fig. 2. Connexió a la placa dels 4 7-segments.

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 es selecciona el corresponent display.

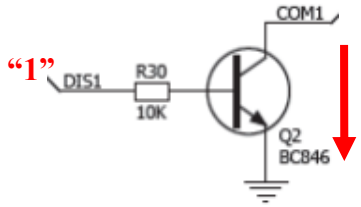

Fig. 3. Circuit de selecció del càtode.

Per pintar en el display s'ha de seleccionar un únic display DIS0, DIS1, DIS2 o DIS3 amb el PORTA i tot seguit enviar pel PORTD el valor a pintar. A continuació s'espera una mica, es selecciona el següent display i se li envia el nou número i així successivament. 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: 7Seg Recull Info addicional*. 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.

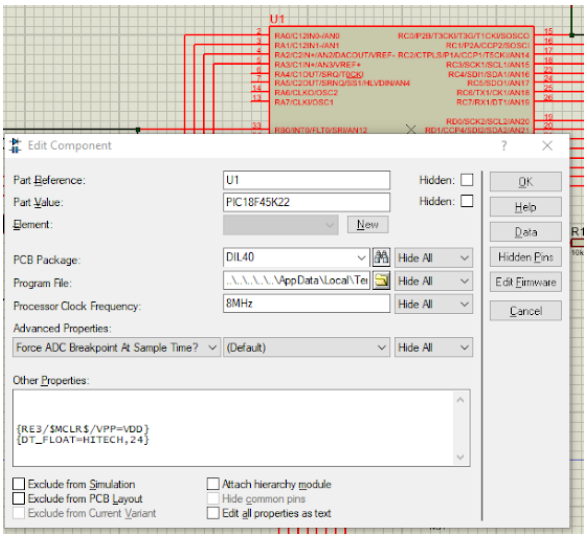

Fig. 4. Captura de pantalla pel canvi de la velocitat del clock.

2.3-Dissenyar els nombres a pintar als 7-segments; per cada un dels deu dígits {0..9} crearem un mapa de 8 bits (que posarem al PORTD) perquè el representi.

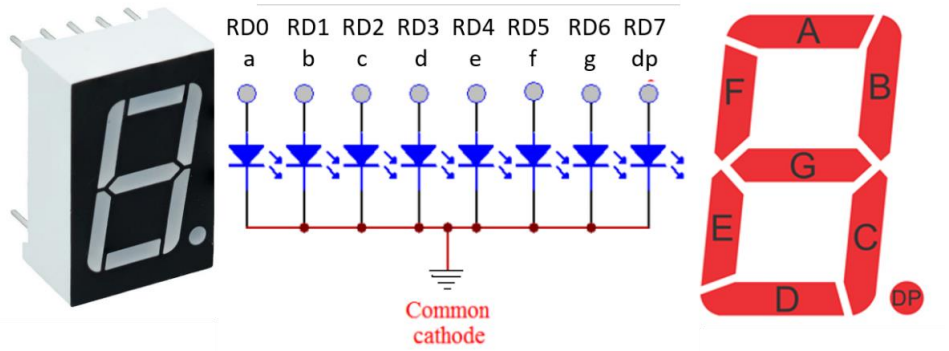

Fig. 5. Esquema dels 7-segments.

Sigueu eficients a l'hora de programar el punt 2.3.

2.4-Disseny de l'esquema elèctric sobre Proteus. Farem el disseny amb Proteus amb el microcontrolador PIC18F45K22 i hi inclourem el bloc de 4 7-segments "7SEG-MPX4-CC" que funciona en càtode comú i inclou 4 displays, 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" que selecciona el càtode comú d'un display) i estalviem que Proteus hagi de fer la simulació d'un circuit analògic amb els transistors, que ralentitzaria la simulació.

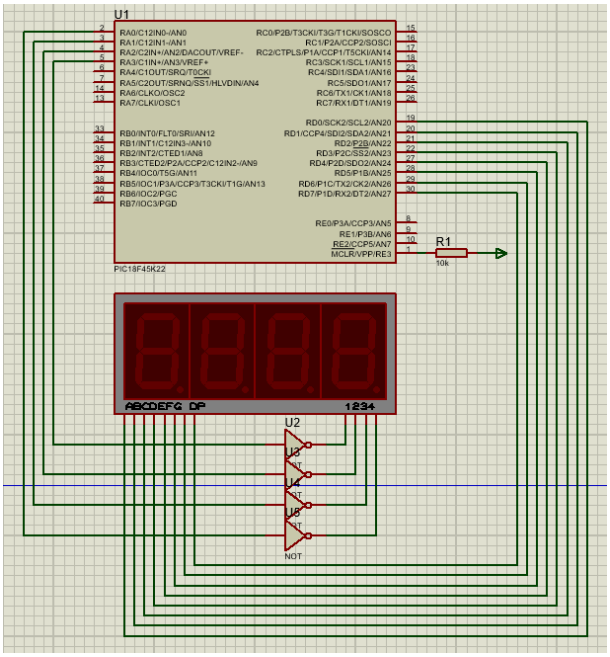

Fig. 6. Model per l'esquema de Proteus.

2.5-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 Freqüència Fosc del Pic i sobretot Esquema gràfic Headers XC8 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-Configurar 4 botons RC4, RC5, RC6, RC7 com a entrades digitals.

3.2-Si es detecta un **flanc de pujada** al botó RC7, es genera un número aleatori entre 0 i 99 que es mostrarà als dos displays de 7-segments de la dreta.

3.3-Si es detecta un **flanc de pujada** al botó RC5, es genera un número aleatori entre 0 i 99 que es mostrarà al dos displays de 7-segments de l'esquerra.

3.4-Si ja s'han generat els dos números aleatoris anteriors i es detecta un **flanc de baixada** al botó RC6 es multiplicaran els dos números aleatoris i es mostraran als 4 displays de 7-segments.

3.5-Si es detecta un **flanc de baixada** al botó RC4 s'apagaran tots els displays de 7-segments. Si es torna a detectar un flanc de baixada es tornaran a activar els displays.

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 Compilador XC8 al link:
https://atenea.upc.edu/pluginfile.php/5581776/mod_resource/content/2/xc8.pdf

No hi ha qüestionari per la pràctica L3(A) però sí per la L3(B).

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
Display 7-segments (5 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 pintar 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 serveix per generar un número aleatori i per resetejar-lo. El botó de la INT1 serveix per modificar el número que surt a pantalla. El botó de la INT2 serveix per canviar el valor que es sumarà. El codi a implementar ha de complir els següents requeriments:

5.1- Al principi del programa hi apareixerà el número "0000" als displays.

5.2- Al apretar la INT0 (configurada per detectar **flancs de pujada**) es generarà un número aleatori entre 0 i 9999 que es representarà als displays. En qualsevol moment que apretem aquest botó apareixerà un nou número aleatori. Al apretar aquest botó s'habiliten la INT1 i la INT2 que han d'estar deshabilitades fins aquest moment.

5.3- Al apretar la INT2 (configurada per detectar **flancs de baixada**) canviarem un comptador. Aquest comptador valdrà {1, 10 ,100 ó 1000} segons el número de vegades que haguem apretat la INT2. Per defecte el comptador valdrà 1, si apreto 1 vegada el comptador valdrà 10, si apreto 2 vegades serà 100, si apreto 3 vegades valdrà 1000, i si torno a apretar es torna a començar valent novament 1.

5.4- Al apretar la INT1 (configurada per detectar **flancs de pujada**) serà quan es canviï el número a pintar pels displays. El valor a mostrar per pantalla serà el valor actual més el comptador seleccionat al punt 5.3.

A mode d'exemple:

-Apretem INT0 i es pinta per exemple el número aleatori 7849.
-Apretem per exemple INT1 i pintarem el número 7849+1=7850 al nostre display.
-Apretem ara INT2 i seleccionem +10 al nostre comptador.
-Apretem una altre cop INT1 i pintarem el número 7850+10=7860 al nostre display.
-Apretem per exemple INT2 i seleccionem +100 al nostre comptador.
-Apretem un altre cop INT2 i seleccionem +1000 al nostre comptador.
-Apretem ara INT1 i pintarem el número 8860+1000=9860 al nostre display.
-Tornem a apretar INT1 i el resultat seria 9860+1000=10860 però pintarem 9999 que és el màxim número representable amb 4 dígits.
-Si en qualsevol moment apretem INT0 tornem a generar un nou número aleatori.

5.5- 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 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.

5.6- 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.

5.7- 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 la el valor de *Frequency*. Comproveu fins a quina freqüència màxima pot arribar el generador abans de que el vostre codi deixi de funcionar correctament.

5.8- Omplir el qüestionari que trobareu a la última pàgina.

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 (2.0 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 (2.0 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 (2.0 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
Display 7-segments (2.0 punts):	Hw mal dissenyat	Visualització incorrecta als displays	Codi poc eficient	Funciona perfectament
Qüestionari (2 punts)	Pregunta 1 = 0.6 punts. Pregunta 2 = 0.1 punts. Pregunta 3 = 0.1 punts. Pregunta 4 = 0.1 punts. Pregunta 5 = 0.1 punts. Pregunta 6 = 0.25 punts. Pregunta 7 = 0.25 punts. Pregunta 8 = 0.25 punts. Pregunta 9 = 0.25 punts.			

QÜESTIONARI–L3 (B) Display 7 segments

(s'ha d'entregar en format electrònic com a treball previ de la L3 (B))

Nom i Cognoms: _____Grup LAB: _____

_____

1) Dibuixa un diagrama de flux amb els estats i les transicions del programa descrit a l'apartat **5**

2) Indiqueu el contingut dels següents registres (en binari) just després d'haver saltat el hardware i el software breakpoint en INT0 (segons us demanem a la secció **5.6 i 5.7** d'aquest enunciat).

Hardware breakpoint:	Software breakpoint:
INTCON =	INTCON =
INTCON2=	INTCON2=
INTCON3 =	INTCON3 =

3) Quin és el elapsed time que us indica Proteus (temps d'execució entre dos breakpoints consecutius, indicat en la barra inferior), des de l'instant en que salta el hardware breakpoint al apretar el botó associat a INT1 fins al software breakpoint en la primera línia de la RSI? Justifica aquest retard.

4) Quin és el temps que es triga a pintar un número de 4 xifres als displays de 7-segments?

5) Quin és el temps que triga el vostre codi de la RSI de la INT0?

Un client ens pregunta quin és el límit de funcionament del nostre sistema. Li hem de contestar de forma raonada estressant el sistema tal i com es menciona al punt 5.7. Decidim dividir la resposta en dues parts, la primera consisteix en simular quan la informació dels displays deixa de ser útil. La segona consisteix en trobar el punt on ja no hi ha manera de que el codi evolucioni de forma correcta.

6) Quin és el límit de funcionament (freqüència o temps de la senyal d'estrès) on deixen de funcionar els displays.

7) Quin és el límit de funcionament (freqüència o temps de la senyal d'estrès) on deixa d'evolucionar el programa.

8) Escriu una breu explicació que justifiqui aquests valors tenint en compte que segurament tindreu dues parts clarament diferenciades al vostre codi, una part al bucle infinit que gestiona el display i fa servir delays, i una altra a la RSI o interrupció que gestiona el funcionament del conjunt.

9) Com podries millorar els límits de funcionament?