



UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH

Facultat d'Informàtica de Barcelona

Enunciat de la pràctica de laboratori

Lab 2:

Entrada-Sortida

L2. Entrada-Sortida Assembler (A) + Entrada-Sortida XC8 (B)

1 Introducció

En aquesta pràctica es prendrà contacte amb la plataforma de desenvolupament EasyPIC v7 (Fig.1) que utilitzarem al llarg de tot el curs. A partir de la documentació proporcionada, identificareu components en esquemes electrònics, coneixereu l'entorn de programació, veureu com descarregar els programes a la plataforma de desenvolupament i fareu unes proves senzilles per agafar pràctica amb el conjunt.

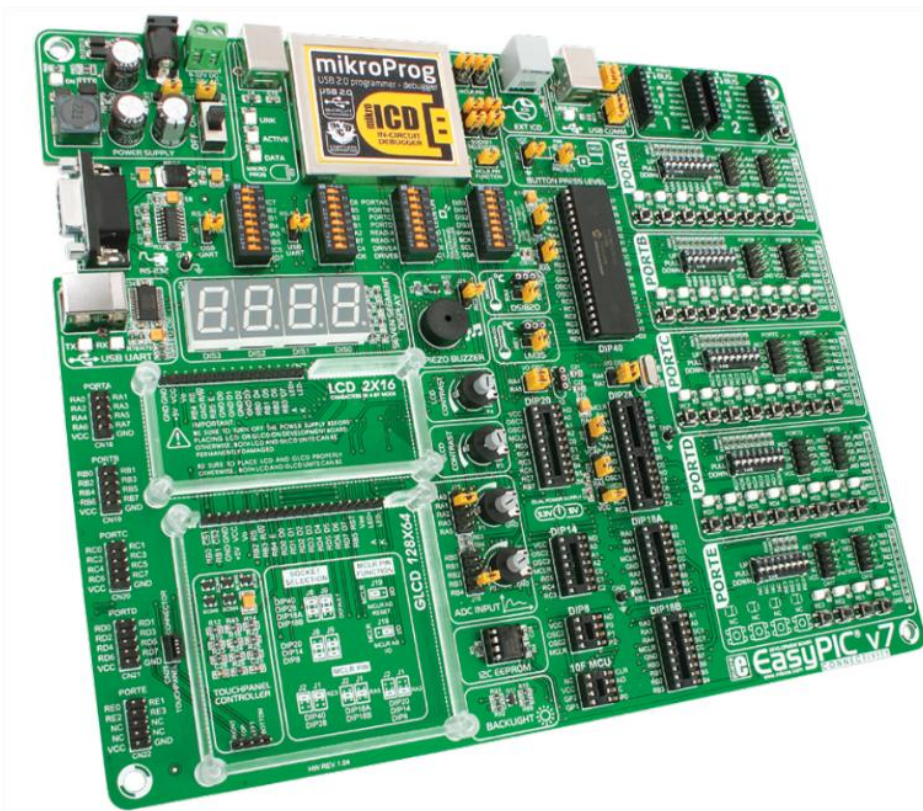


Fig. 1. Plataforma de desenvolupament EasyPIC v7.

El *Integrated Development Environment* (IDE) utilitzat a les pràctiques serà el Proteus. Proteus integra un editor d'esquemàtics, editor de codi, compilador, assemblador i simulador en temps real. A l'hora de crear un nou projecte amb Proteus i seleccionar el *firmware* amb el que treballareu, també haureu d'indicar el tipus de compilador.

A la primera part de la pràctica L2 (A) programarem en assembler amb el MPASM.

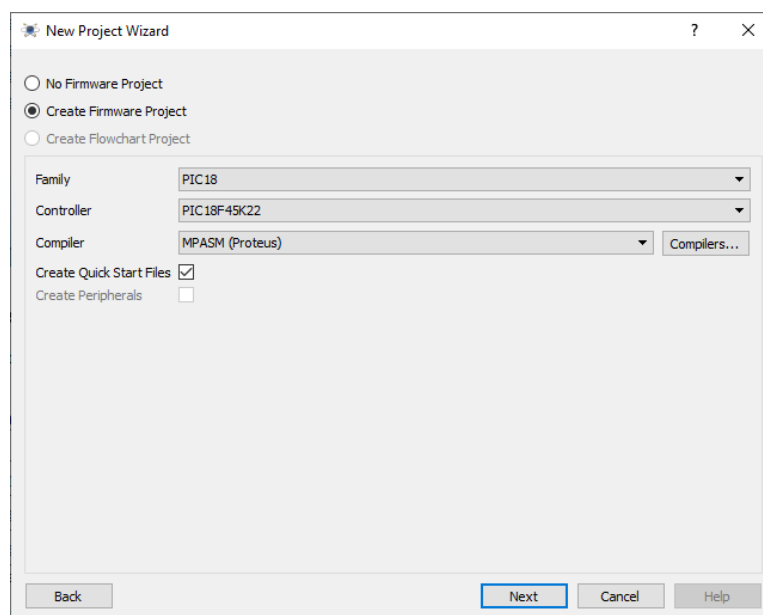


Fig. 2. Captura de pantalla per la generació d'un nou projecte d'assembler amb el PIC18F45K22 a Proteus.

A la segona part de la pràctica L2 (B) i fins a acabar el quadrimestre, es farà servir el **compilador XC8** (llenguatge C). Aquest compilador ja està instal·lat als laboratoris de la FIB però no ve per defecte amb Proteus. El compilador està disponible a:

<https://www.microchip.com/en-us/tools-resources/archives/mplab-ecosystem#xc8>

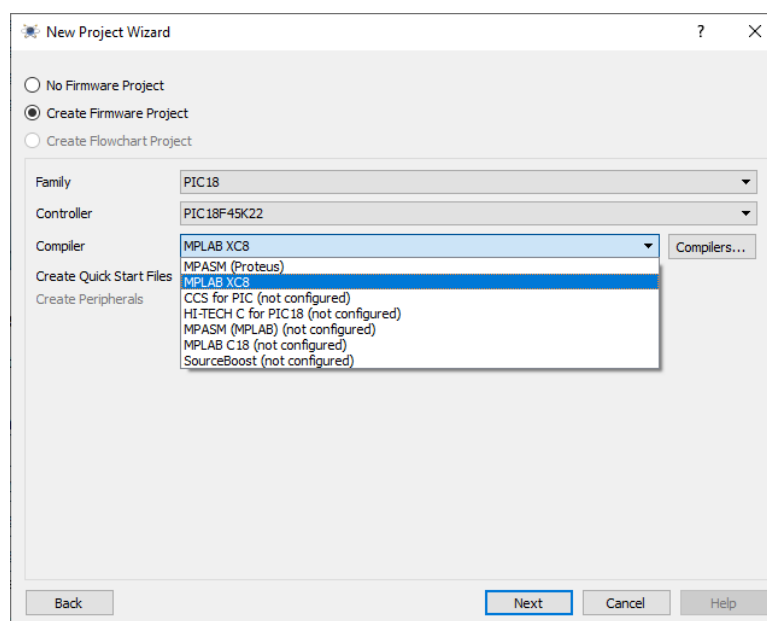


Fig. 3. Captura de pantalla per la generació d'un nou projecte en C amb el compilador XC8 pel PIC18F45K22 a Proteus.

Amb el software Proteus podreu crear interfícies entre el microcontrolador i el món exterior, programar el codi que executa el microcontrolador i simular-ho tot conjuntament. Un cop el simulador funcioni, podreu provar realment el funcionament del vostre programa a la placa de desenvolupament. Els dos escenaris de simulació i execució a la placa necessiten compilar de forma diferent a Proteus:

- **SIMULACIÓ:** haureu de compilar el codi en “mode *Debug*” (a la finestra “*Source Code*” del Proteus). Fent això, el procés de compilació generarà un arxiu amb extensió “.cof” i permetrà simular l’execució del PIC en el vostre esquemàtic *hardware* i també podreu debugar el contingut de les variables, registres, etc.
- **EXECUCIÓ A LA PLATAFORMA EASYPIC:** haureu de compilar el codi en “mode *Release*”. Fent això, el procés de compilació generarà a la vostra carpeta de treball un arxiu amb extensió “.hex” que és executable sobre el PIC. Per executar el programa a la EasyPIC, haureu de descarregar aquest fitxer “.hex” a la placa a través del programador connectat al bus USB. La descàrrega del *firmware* a la placa la fareu a través del programa **mikroProg**. Un cop carregat el *firmware*, el programador provocarà un reset i executarà la vostra aplicació.

En aquesta pràctica L2 es treballaran les entrades i sortides del PIC18F45K22. Per entendre el seu funcionament s’haurà de comprendre tant la part de software a programar com la part de hardware sobre la que funciona el codi programat. Per tant:

- Cal entendre la configuració interna dels I/O PORTS A, B, C, D i E del microcontrolador (pàgines 127-136 del *PIC18F45K22 Data Sheet*). En particular la configuració del PORTx com a port digital o analògic mitjançant els registres ANSELx, així com la configuració dels ports com a IN o OUT mitjançant els registres TRISx.
- Cal dissenyar sobre Proteus el circuit basat en l’esquemàtic que trobareu a la documentació lliurada sobre la placa EasyPIC v7 (i a la figura 4). Tingueu sempre present que heu de copiar de l’esquemàtic només aquells components que necessiteu per a fer la pràctica de forma similar als circuits que es demanaven a la pràctica anterior (**NO intenteu dissenyar la placa sencera**).
- D’altra banda, vigileu que l’entrada MCLR (Master Clear/Reset) estigui a 1, per evitar que hi hagi resets aleatoris si queda a l’aire.

Tri-state DIP switches, like **SW7** on **Figure 10-2**, are used to enable 4K7 pull-up or pull-down resistor on any desired port pin. Each of these switches has three states:



Figure 10-2: Tri-state DIP switch on PORTC

1. **middle position** disables both pull-up and pull-down feature from the PORT pin
2. **up position** connects the resistor in pull-up state to the selected pin
3. **down position** connects the resistor in pull-down state to the selected PORT pin.

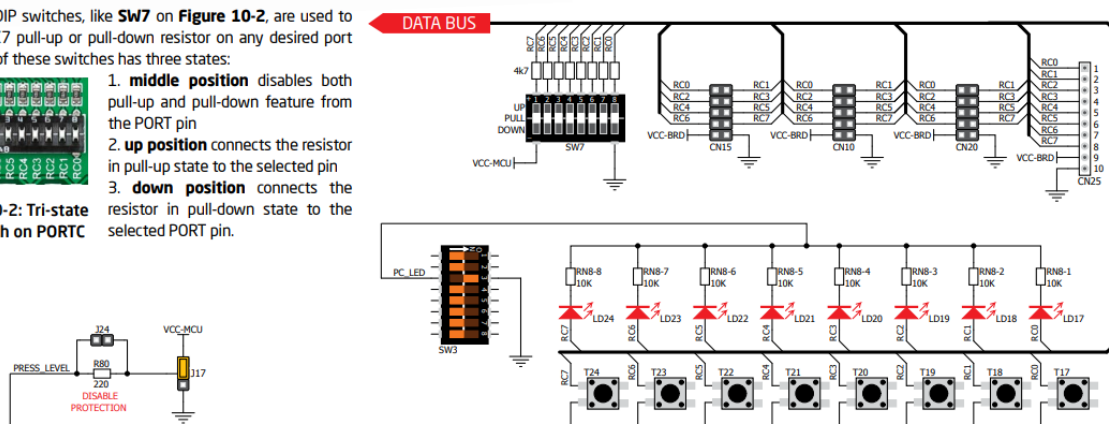


Fig. 4. Esquema dels polsadors i LEDs del PORT C a la placa EasyPIC v7.

2 Objectius

L'objectiu d'aquesta pràctica és encendre i apagar uns LEDs connectats a un port (que configurarem com a sortida) a partir dels senyals dels pulsadors connectats al mateix o a un altre port (que configurarem com a entrada).

En acabar la pràctica l'alumne serà capaç de:

- manegar informació tècnica donada pel fabricant de la placa de desenvolupament que utilitzem a les pràctiques.
- cercar la informació necessària dins dels manuals de referència.
- controlar el temps i la seva relació amb la freqüència d'oscil·lació de la EasyPIC $F_{osc}=8\text{MHz}$
- identificar els elements 'pulsador' i 'LED' i com es connecten al microcontrolador.
- crear i configurar un projecte.
- *debugar* el programa, utilitzar *breakpoints* i *tracejar* els valors de les diferents variables i registres.
- descarregar els fitxers executables ja generats a la placa de desenvolupament.
- entendre el funcionament del conjunt, veient la relació entre el codi i el comportament del sistema.
- gestionar adequadament senyals d'entrada per nivell i per flanc, i utilitzar variables d'estat.

3 Treball Previ Entrada-Sortida Assembler (A)

Aquesta primera part de la pràctica L2 es farà en assembler.

Fer un programa amb Proteus que realitzi la següent funcionalitat:

- Connecteu adequadament un botó al pin RD0. Configureu el pin RD0 com a entrada digital.
- Connecteu adequadament un botó al pin RD1. Configureu el pin RD1 com a entrada digital.
- Connecteu adequadament un led al pin RD2. Configureu el pin RD2 com a sortida digital.
- Llegiu l'estat de l'entrada RD0 i RD1. Si ambdues són 1 llavors enceneu el led connectat al RD2, altrament el led haurà d'estar apagat.
- Connecteu adequadament un led al pin RD3. Configureu el pin RD3 com a sortida digital.
- Afegiu una rutina al final del bucle infinit que canviï l'estat del pin RD3 cada 200 cicles.

```

// A proposal for your code... Just to inspire you
// Hexadecimal numbers start with 0x or without prefix
// Binary numbers start with B'
// Decimal numbers start with D'

#include p18f45k22.inc
config FOSC = HSHP

;=====
; VARIABLES
;=====
var1 equ 0x20
var2 equ D'15'

;=====
; RESET and INTERRUPT VECTORS
;=====
RST    code    0x0
        goto    Start
;=====
=====
; CODE SEGMENT
;=====
PGM    code
Start
        movlw B'00000001'; W=1
        ; Configurar PORTs
        ...
Loop
        ; Llegir RD0, RD1 i actualitzar RD2
        ...
        ; Posar el valor a RD3 i esperar uns 200 cicles
        ...
        goto Loop

delay_200cicles
        ...

```

Fig. 5. Plantilla de codi que podeu fer servir.

Entregueu pel Racó, ABANS de la vostra sessió de pràctiques, el projecte PROTEUS (versió actual als laboratoris, v8.13 SP0).

4 Rúbrica treball Previ Entrada Sortida Assembler (A)

	Iniciat (0-2.5 punts)	En desenvolupament (2.5-5.0 punts)	Aconseguit (5.0-7.5 punts)	Exemplar (7.5-10 punts)
Esquemàtic circuit entrada (1.5 punts):	No funciona	Funciona en alguns casos. Hi han errors o falten connexions	Funciona però la selecció dels components no és apropiada	Funciona perfectament
Esquemàtic circuit sortida (1.5 punts):	No funciona	Funciona en alguns casos. Hi han errors o falten connexions	Funciona però la selecció dels components no és apropiada	Funciona perfectament
Codi configuració (2 punts):	No funciona	Mala configuració de ANSELx i TRISx	Mala configuració de ANSELx o TRISx	Ben configurat
Codi led RD2 (2 punts):	No funciona	No té en compte l'estat del bit sinó del PORT sencer	No funciona en alguns casos	Funciona perfectament
Codi delay_200cicles (3 punts):	No funciona	El codi està molt mal estructurat	Algunes parts del codi són difícils de justificar o no es donen suficients proves de comprovació	Codi net, ben estructurat i fàcil d'entendre, comprovat mitjançant un oscil·loscopi correctament configurat

5 Treball Previ Entrada-Sortida XC8 (B) 7-segments

Aquesta segona part de la pràctica L2 es farà en llenguatge C fent servir el compilador XC8 que podeu trobar a la pàgina web de microchip.

5.1 Preparació del projecte

Abans de començar és necessari que afegiu l'arxiu "config.h" al projecte de Proteus per a una correcta configuració del nostre micro. Trobareu aquest arxiu a Atenea. Per afegir-lo teniu diferents opcions, una d'elles és anar a la finestra de Proteus anomenada Source Code, a la part esquerra trobareu l'estructura d'arxius del projecte, cliqueu amb el botó dret i afegiu un arxiu nou amb Add New File, anomeu l'arxiu com a config.h, obriu-lo i copieu el codi que hi ha a l'arxiu d'Atenea.

També heu de configurar Proteus perquè el PIC18F45K22 funcioni a 8MHz de tal manera que la simulació i la placa EasyPicv7 funcionin a la mateixa freqüència 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.

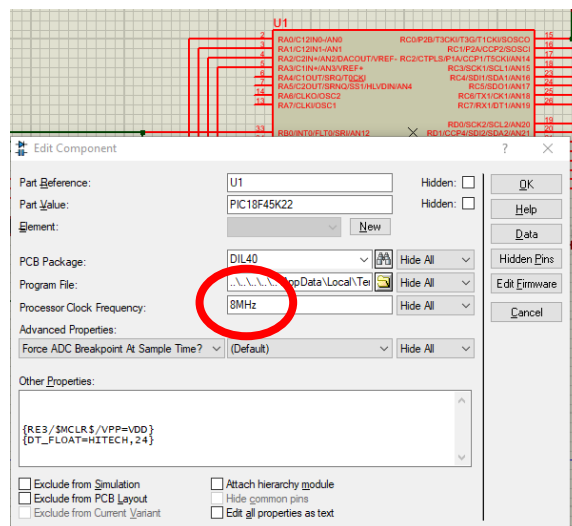


Fig. 6. Captura de pantalla pel canvi de la velocitat del clock.

5.2 Introducció als displays de 7 segments

Un cop explicada les modificacions necessàries d'ara endavant en els projectes de Proteus, explicarem l'objectiu d'aquesta part de la pràctica. En aquesta part haureu de fer un programa en llenguatge C en el que es mostri un número entre el 0 i el 9 en un display de 7 segments.

La placa de desenvolupament EasyPIC v7 (Fig. 7) disposa de 4 displays de 7 segments que ens permetran mostrar informació de manera senzilla, emprant els ports d'E/S. **En aquesta pràctica només farem servir un d'aquest displays.**

Essencialment un display de 7 segments és un component electrònic format per 8 leds (7 per crear la figura i un pel punt decimal) disposats d'una manera adequada per mostrar informació, principalment números.

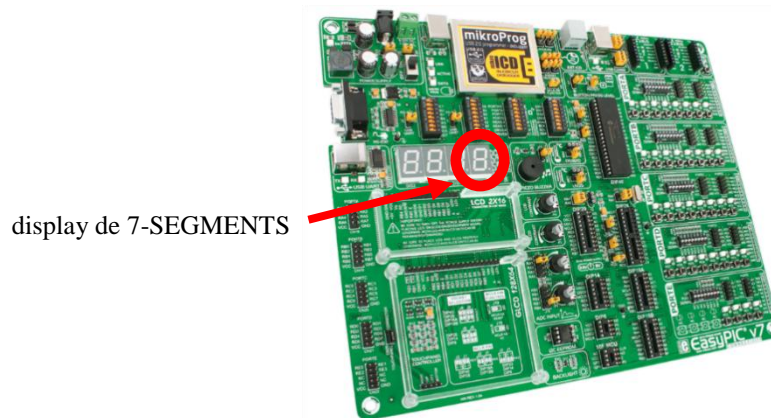


Fig. 7. Detall dels displays de 7 segments a la plataforma de desenvolupament EasyPIC v7.

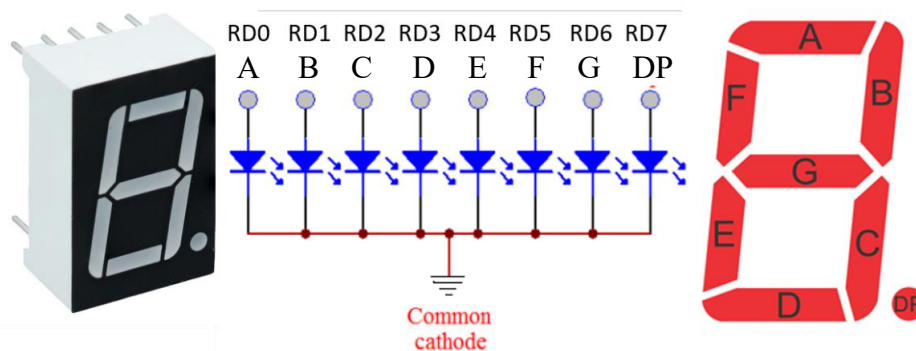


Fig. 8. Esquema dels 7-segments.

A la figura 8 es pot veure la ubicació dels leds i la seva connexió al PORTD de l'EasyPIC.

En el disseny del vostre circuit haureu d'afegir un display de 7-segments anomenat 7SEG-COM-CATHODE que trobareu a la llibreria DISPLAY de Proteus.

Aquest display té 8 connexions, 7 de les quals aniran al PORTD i serviran per encendre els leds, i una connexió addicional anomenada càtode comú que serveix per activar/desactivar el display. El led del DP no existeix en aquest model i per tant el pin RD7 no es connecta enlloc.

Per explicar el funcionament del càtode comú s'ha afegit la figura 9, on apareix l'esquemàtic dels displays de 7 segments de l'EasyPIC i la figura 10 on s'explica els dos estats de funcionament dels transistors.

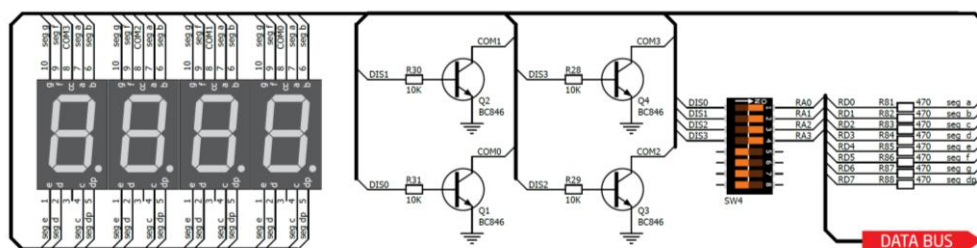


Fig. 9. Connexió a la placa dels 4 7-segments segons l'esquemàtic del fabricant.

D'aquesta figura 9, podem distingir 4 parts, d'esquerra a dreta trobem els displays, els transistors, el selector i les resistències limitadores de corrents als leds. En aquest punt és necessari destacar que el selector SW4 de l'esquemàtic serveix per connectar/desconnectar físicament la base del transistor dels pins RA0...RA3 quan no es vol fer servir algun dels displays de 7-segments. En aquesta pràctica només farem servir un dels displays, un dels transistors i un dels 4 selectors.

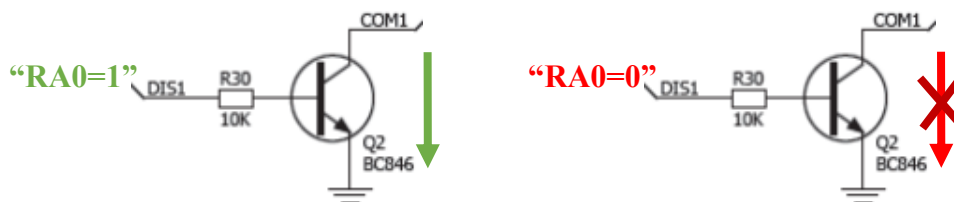


Fig. 10. Circuit d'activació del display de 7-segments, quan al pin RA0 hi ha un "1" el transistor condueix i el display s'encén, quan al RA0 hi ha un 0 el transistor no condueix i el display resta apagat.

Observant amb detall els esquemes anteriors podem dir que el dibuix que haurà als leds del display de 7-segments dependrà del contingut del PORTD del nostre microcontrolador. D'altra banda, el pin RA0 determinarà si el transistor condueix o no, i per tant si el display associat estarà activat o no.

A continuació disposeu d'una captura de pantalla amb l'esquema a realitzar amb Proteus:

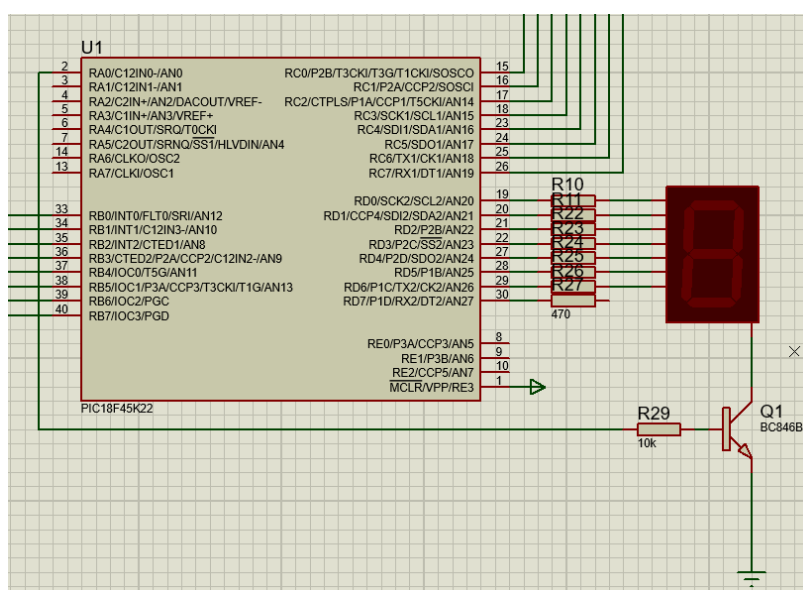


Fig. 11. Captura de pantalla del circuit a realitzar

Segons l'esquemàtic, el transistor connectat al càtode comú és del tipus NPN, en concret és el BC846 que trobareu a la llibreria ZETEX de Proteus, que es connectarà com s'indica a continuació:

- la base del transistor es connecta al pin RA0 a través d'una resistència de $10k\Omega$
- el col·lector del transistor es connecta al càtode comú del display de 7-segments
- l'emissor del transistor es connecta a terra

Amb aquestes connexions podrem activar o desactivar el display. Si posem el pin RA0 a "1" llavors el display estarà activat i mostrarà la informació del PORTD. Si posem el RA0 a "0" llavors el display estarà totalment apagat sigui quin sigui el contingut del PORTD.

D'altra banda, entre els leds del display i el PORTD del microcontrolador s'han posat unes resistències de 470Ω per limitar el corrent a través dels leds segons l'esquemàtic de l'EasyPIC.

5.3 Requeriments del programa a realitzar

A continuació programeu el vostre codi perquè faci el següent:

- Feu dos circuits amb dos botons als pins RB7 i RB6 i configureu-los com a entrades.
- Programeu una variable que augmenti quan es detecta un flanc de pujada al botó RB7 i que disminueixi quan es detecta un flanc de pujada al botó RB6. El valor mínim d'aquesta variable serà 0 i el màxim serà 9.
- Representeu el valor d'aquesta variable al display de 7-segments. Haureu de seleccionar quins leds encendre per visualitzar cada número (mireu la Fig. 8). Per cada un dels deu dígit {0,1,2,..9} creareu un mapa de 8 bits que s'ha d'escriure al PORTD com a sortida.

Sigueu eficients a l'hora de programar aquest punt.

- Afegeu un botó al pin RB5 que si està apretat apagui el display de 7-segments, i que si no està apretat l'encengui.

Entregueu pel Racó, ABANS de la vostra sessió de pràctiques, el projecte PROTEUS amb els circuits i els codis que es demanen. Per a garantir compatibilitats de versions, lliureu sempre el vostre treball salvat en una versió compatible de PROTEUS (versió actual als laboratoris, v8.13 SP0).

```

// A proposal for your code... Just to inspire you

#include <xc.h>
#include "config.h"

void configPIC(){
    ANSELA=...; // All pins as digital
    ANSELB=...;
    ANSELC=...;
    ANSELD=...;
    ANSELE=...;

    TRISA=...;
    TRISB=...;
    TRISC=...;
    TRISD=...;
    TRISE=...;
}

void main(void)
{
    configPIC();

    while(1)
    {
        // your code here
    }
}

```

Fig. 12. Plantilla de codi que podeu fer servir.

6 Rúbrica treball Previ Entrada Sortida XC8 (B)

	Iniciat (0-2.5 punts)	En desenvolupament (2.5-5.0 punts)	Aconseguit (5.0-7.5 punts)	Exemplar (7.5-10 punts)
Proteus circuit botons entrada (1 punt):	No hi ha circuit	El circuit no funciona	No hi ha resistències de pull-up o de pull-down	Circuit perfectament implementat
Proteus circuit display (1 punts):	Més de dos errors al disseny del circuit del display	Dos errors al disseny del circuit del display	Un error al disseny del circuit del display	Circuit perfectament implementat
Configuració Entrada/Sortida (1 punts):	La configuració és inexistent	Alguna part de la configuració és errònia	La configuració o no és estructurada, o no és modular, o no està comentada o té algun error	La configuració és estructurada, modular, està comentada i no té cap error
Detecció flancs (2.5 punts):	La detecció és inexistent	Alguna part de la detecció és errònia	La detecció o no és estructurada, o no és modular, o no està comentada o té algun error	La detecció és estructurada, modular, està comentada i no té cap error
Programació display (4.0 punts)	El codi és molt bàsic o inexistent	Alguna part del codi té un error greu	El codi o no és estructurat, o no és modular, o no està comentat o té un temps de càlcul innecessàriament gran degut a la ineficiència	El codi és estructurada, modular, està comentat i és eficient