UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH

Facultat d'Informàtica de Barcelona

Enunciat de la pràctica de laboratori

Interrupcions

Objectius

L'objectiu d'aquesta pràctica és familiaritzar-se amb la programació de les rutines de servei a les interrupcions (RSI). Dels diferents tipus d'interrupcions existents, en aquesta pràctica treballarem sobre les interrupcions externes, en particular farem servir polsadors externs que al apretar-los o deixar-los anar 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). Es treballarà amb les interrupcions del microcontrolador PIC18F45K22 i la placa de desenvolupament EASYPIC7 de MikroElectronica.

Coneixements previs de l'alumne:

L'alumne ja coneix:

- L'arquitectura del PIC18F45K22
- El simulador PROTEUS
- La programació del PIC en C
- El funcionament dels ports d'E/S del PIC
- Les particularitats i instruccions pròpies del compilador XC8
- El funcionament de les interrupcions d'alta i baixa prioritat
- Les opcions de la paca EASYPIC7 referents a la connexió dels polsadors a VDD, VSS, configuracions de *pull-up* i *pull-down*
- Les opcions de la placa EASYPIC7 referents a la connexió dels leds als ports del micro

Treball previ:

Temps estimat: 4 hores

L'objectiu del treball previ és programar un petit joc que simuli 4 tirades de daus. Per fer-ho s'hauran de generar números aleatoris entre l'1 i el 6. També s'hauran de connectar 3 botons als pins associats a les interrupcions INT0, INT1 i INT2.

Els passos a seguir per a realitzar la pràctica són:

- Estudiar l'apartat 5.9 del *User's Guide* del compilador XC8 referent a la programació de rutines de servei a interrupcions.
- Consultar el Capítol 9 (*Interrupts*) del *DataSheet* del microcontrolador per resoldre el que es demana en aquesta pràctica.
- Esbrinar l'adreça del vector d'interrupció pel cas d'alta i baixa prioritat.
- Entendre com configurar si la interrupció es produirà al flanc de pujada o baixada.
- Estudiar la manera d'assignar la prioritat a les interrupcions externes.
- Esbrinar en el *datasheet* del microcontrolador per quins ports i per quins bits entren les interrupcions externes INT0, INT1 i INT2, així com els corresponents pins del microcontrolador.
- Dissenyar l'esquema electrònic sobre PROTEUS usant els components que calguin per a poder simular la pràctica respectant, lògicament, els ports i els pins per on entren les interrupcions externes INT0, INT1 i INT2.
- Generar números aleatoris entre l'1 i el 6. Per fer-ho teniu la funció `rand()` i `srand()` que són accessibles al incloure la llibreria `#include <stdlib.h>`
- Implementar el programa descrit a l'apartat ***Lògica del Joc***.
- Realitzar l'execució, test i debugat del vostre programa sobre PROTEUS. Les interrupcions són una font contínua de bugs. Aquests són molt difícils de tracejar, doncs degut a la natura asíncrona de les interrupcions, els errors de programa succeeixen de forma imprevisible. Degut a la complicació que suposa debugar el programa en fase d'execució, és recomanable ser meticulós en la fase de desenvolupament del codi. Per treballar de forma correcta, cal que seguiu els següents passos:

– Posar breakpoints a les RSI. Comproveu que sempre que feu saltar una interrupció, s'executa la RSI que toca. Comproveu que els bits de control implicats dins i fora de la RSI (flags, prioritats, emmascarament) són els esperats.
– Poseu 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/), i un software breakpoint a la primera instrucció de la RSI d'alta prioritat.
– Comproveu que no salten més interrupcions que les provocades pels polsadors.
– 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. Comproveu fins a quina freqüència màxima pot arribar el generador abans de que el vostre codi deixi de funcionar correctament.

Lògica del joc:

La lògica del joc és la següent:

Al arrencar el sistema es pinta al display de 7-segments el text “Go..” En aquest estat inicial només les interrupcions de INT0 estan activades.

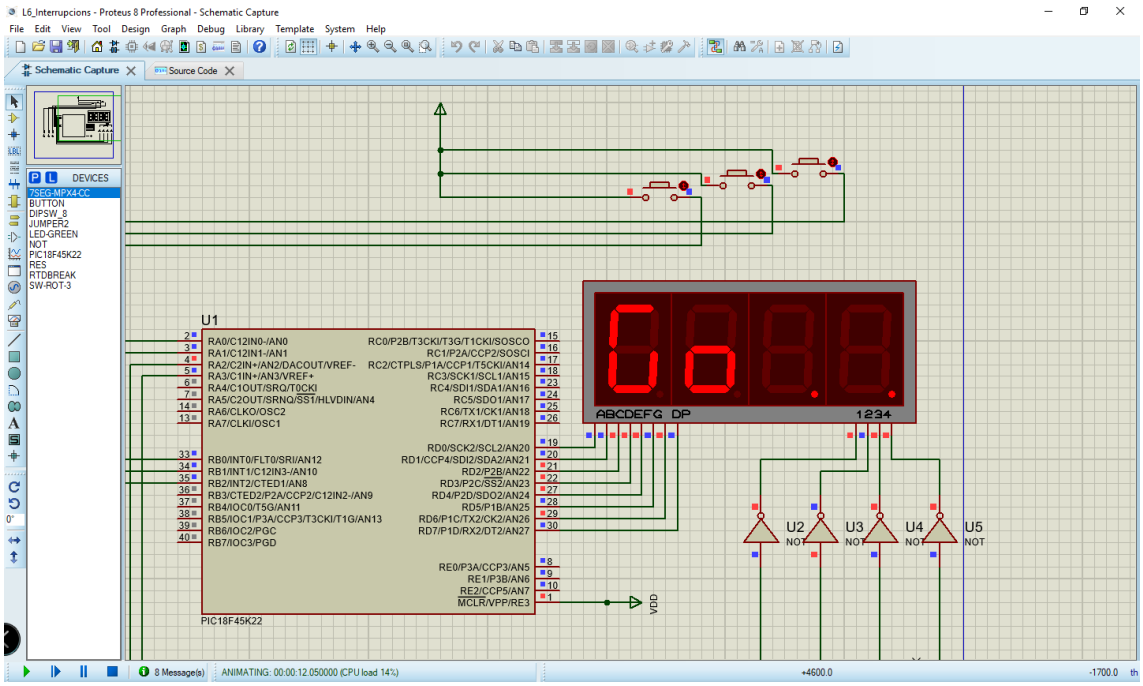


Figura 1. Estat inicial

Al apretar el botó associat a INT0 es genera una interrupció per flanc de pujada que pinta el text “_ _ _ _” com es mostra a la Fig. 2.

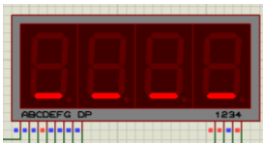


Figura 2. Estat al apretar el botó associat a INT0

Cada cop que s'apreta el botó associat a INT1, es genera un número aleatori entre 1 i 6, la primera vegada es pinta el numero generat al display de la dreta, la segona vegada al segon display i així successivament fins a arribar a les 4 tirades.

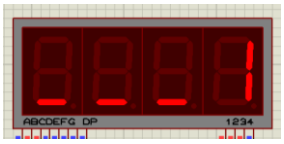


Figura 3. Primera tirada de dau al apretar INT1

Figura 4. Segona tirada de dau al apretar INT1

Figura 5. Tercera tirada de dau al apretar INT1

Figura 6. Quarta tirada de dau al apretar INT1

Un cop fetes les 4 tirades, s'innabilita la interrupció INT1 i s'habilita la interrupció INT2.

En aquest estat, si s'apreta el botó associat a INT2 que funciona per flanc de **baixada** es pintarà per pantalla el valor de sumar els 4 números aleatoris generats anteriorment amb el text "X=" + suma dels números generats, tal i com es pot veure a la Fig. 7.

Figura 7. Text corresponent a la suma de les 4 tirades
de daus "X=16" al apretar el botó associat a INT2

Un cop mostrat el resultat anterior, s'inhabiliten les interrupcions associades a INT2.

En qualsevol moment, apretar el botó associat a INT0 comença una nova partida.

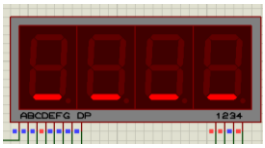


Figura 8. Partida nova al apretar el botó associat INT0

Entregueu el full de respostes al final d'aquest document i el projecte PROTEUS (inclou diagrama esquemàtic, codi font implementat i el fitxer executable ".hex") pel Racó, ABANS de la vostra sessió de pràctiques, tot en un arxiu *.zip. Per a garantir compatibilitats de versions, lliureu sempre el vostre treball salvat en una versió compatible de PROTEUS (versió actual als laboratoris, v8.4 SP0).

Pràctica al laboratori

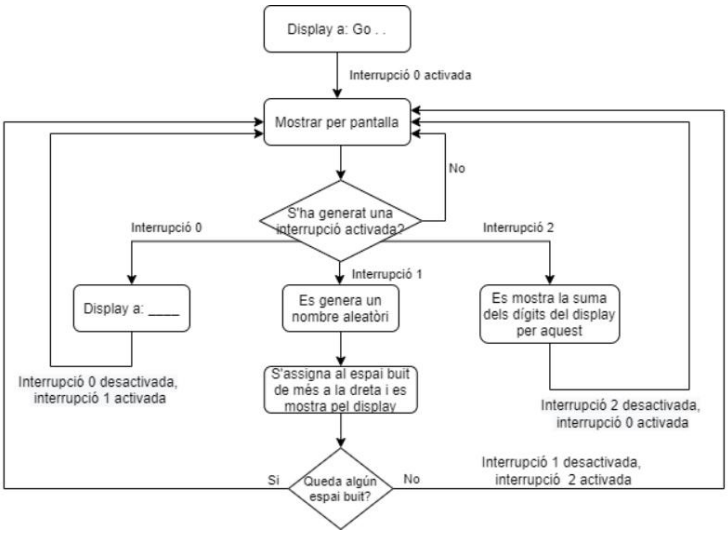
- En iniciar la sessió de pràctiques al laboratori, haureu de mostrar el programa que heu implementat funcionant en PROTEUS.
- A continuació, haureu de passar la implementació a la placa EASYPIC7, per la qual cosa cal:
  - Esbrinar a l'esquema electrònic de la placa EASYPIC7 quins polsadors son els que arriben a les interrupcions externes INT0, INT1 i INT2.
  - Configurar els polsadors tal com els teniu a l'esquema de PROTEUS a partir dels *jumpers* i *switches* corresponents de la placa.
  - Connectar un extrem del polsador a VCC o GND mitjançant el *jumper* J17.
  - Connectar una resistència de *pull-up* o *pull-down* a l'altre extrem mitjançant el SW del port corresponent.
  - Programar el microcontrolador amb el *firmware* implementat i provat sobre PROTEUS.
  - Comprovar el correcte funcionament del programa en la placa.
- Comprovareu que es pot produir una lleugera diferència sobre la placa quan, a cada pulsació, s'activa més d'una interrupció. Implementar la corresponent estratègia anti-rebots.
- A continuació s'haurà de modificar el programa per a que realitzi la funció que determini el professor. Aquesta part de la pràctica s'haurà de realitzar "in situ" al laboratori, i no disposeu d'enunciat. L'enunciat us el proposarà el professor en la mateixa sessió, un cop hagueu entregat el treball previ.

Noms:____Bernat Borràs Civil i Miquel Torner Viñals _____GRUP: __13___

Qüestionari d'interrupcions

A ENTREGAR PEL RACÓ ABANS DE L'INICI DE LA SESSIÓ DE LABORATORI

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



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 Treball Previ d'aquest enunciat).

Hardware breakpoint:
INTCON = 0b11010010
INTCON2= 0b11110101
INTCON3 = 0b11000000

Software breakpoint:
INTCON = 0b01010010
INTCON2= 0b11110101
INTCON3 = 0b11000000

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 INT0 fins al software breakpoint en la primera línia de la RSI? Justifica aquest retard.

Hi ha una diferència de 9 microsegons desde que es detecta el hardware breakpoint fins al software breakpoint. Això es dona degut a que el PIC ha de detectar que s'ha generat una interrupció i saltar fins a la RSI corresponent, en aquest cas la de alta prioritat.

4) Quan estresseu el sistema, quina és la freqüència màxima a la que podem generar interrupcions sense perdre'n cap?

A partir de 12 kHz el PIC18 no és capaç de gestionar totes les interrupcions rebudes.

5) Quina és la diferència entre un generador de números pseudo-aleatoris i un generador de números aleatoris real.

Un generador de números pseudo-aleatoris fórmules matemàtiques mentre que un generador de números hauria d'emprar l'atzar com a fórmula.