

PRÁCTICA 1

Ejercicio 1.1: Parpadeo de 2 LEDs

El objetivo de esta práctica es tomar contacto con el entorno UVI-51 Win con ejemplos muy sencillos. El primer programa hace parpadear un par de LEDs conectados al terminal P1.3 del microcontrolador.

Una vez que el PC ha arrancado, buscar la carpeta del turno de prácticas que corresponda y abrirla. Ejecutar UVI51. Aparece el gestor de proyectos de UVI51, al que llamaremos *entorno* de UVI51.

APERTURA DE UN SISTEMA YA DEFINIDO:

- 1) Seleccionar File-Abrir CNX. En el cuadro de selección de archivos que se presenta localizar el archivo EJ_1_1.CNX. Hacer click sobre él para abrirlo. Los archivos CNX definen el hardware de un sistema formado por uno o varios micros y un conjunto de periféricos externos (leds, pulsadores, etc.) interconectados. En el manual de UVI51 se denominan archivos de configuración (de sistemas).
- 2) Pulsar el botón [EJ_1_1.CNX]. Se entra en el editor de texto que permite visualizar y modificar el archivo CNX. Este ejemplo define un microcontrolador y 2 LEDs conectados como indica la figura. Se define además un analizador lógico, con el que se registra en el archivo pr1 la forma de onda generada en los nodos del circuito, en este caso el nodo 2. El nodo 0 representa masa. El nodo 1 representa +Vcc a través de una resistencia.

* Archivo de configuración para el ejercicio 1.1 de SDPI

* UVI51

Uej_1_1.src 12.0 16 4096

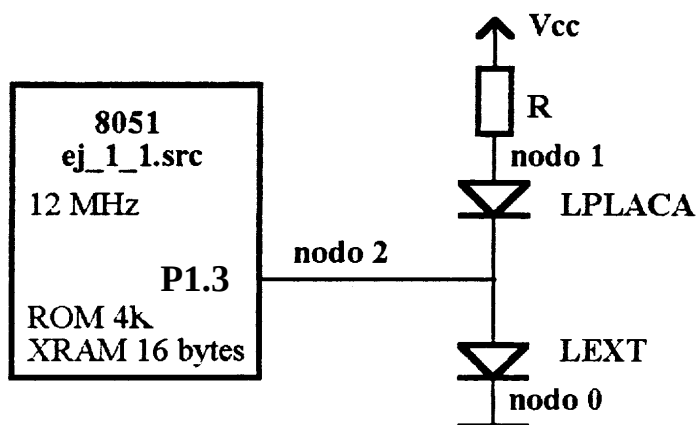
P13=2

LPLACA 1 2 1

LEXT 2 0 2

Apr1

N2 led



La sintaxis empleada por los archivos CNX puede verse en el Manual del Simulador UVI51. El micro ejecutará el programa que aparece después de la letra U, es decir, ej_1_1.src.

- 3) Salir del editor mediante: Archivo-Salir.

4) Pulsar el botón [EJ_1_1.SRC]. Se entra nuevamente en el editor de texto EDIT que permite en este caso editar el archivo del programa fuente en ensamblador (SRC). Escribimos el programa:

; Programa fuente en ensamblador del ejercicio 1.1 de sdp1

```
led    BIT    p1.3           ;define el puerto del LED
org 0   ;actualiza el contador de programa (CP) a 0

      ajmp    inicio         ;salta al inicio del programa
org 100h ;actualiza el CP a 100H
inicio:
      mov     dptr,#-600      ;carga -600 en DPTR para frec. de parpadeo,  $T/2 = 5 + 600 * 6 \mu s = 3605 \mu s$ 
      cpl     led             ;invierte estado LED
bucle: ;6  $\mu s$  cada iteración
      inc     dptr            ;incrementa valor de DPTR
      mov     a, dph          ;guarda en A la parte alta del DPTR
      orl     a, dpl          ;OR de la parte alta con la parte baja del DPTR
      jnz     bucle           ;salto a bucle si DPTR no es cero
      sjmp    inicio          ;salta al inicio del programa

end
```

El programa hace parpadear los LEDs manteniéndose en cada estado 3605 μs , tiempo que se puede ajustar cambiando el valor cargado en DPTR. Se emplea el método del *retardo software*.

5) Salir del editor con Archivo-Salir.

6) Pulsar el botón [--E->) para ensamblar (y enlazar) el archivo fuente. El ensamblador utilizado abre una nueva ventana con una serie de mensajes en la pantalla y provoca un desplazamiento de la pantalla que no afecta al funcionamiento de UVI51. Si se producen errores de ensamblado, el botón [ej_1_1.LST] adquiere el color rojo.

7) Pulsar el botón [ej_1_1 .LST] para visualizar el informe del ensamblador. El archivo LST muestra el código máquina generado para cada línea del programa fuente. Si hay errores al ensamblar se pueden ver en este fichero. Tras localizarlos salir del editor y volver al paso 4 para corregirlos.

8) Salir del editor.

9) Pulsar el botón [SIMULAR] para entrar en el simulador. La pantalla está dividida en dos ventanas. La superior se denomina ventana del sistema e incluye una tabla con los periféricos externos definidos en el archivo CNX. En este ejemplo se visualizan 2 LEDs. La inferior se denomina ventana del micro y muestra el estado del microcontrolador indicando el contenido de la RAM, registros de la CPU, etc. en forma de subventanas.

10) Pulsar el botón [PASO] para ejecutar una instrucción. Pulsar varias veces hasta que se ejecute la instrucción cpl led. Observar como ambos LEDs cambian de estado.

11) Pulsar el botón [RESET] para inicializar el micro. Este botón equivale a aplicar la señal de reset en el micro real.

12) Pulsar el botón [EJ/DET] para ordenar la ejecución en modo continuo. Observar el parpadeo de los leds.

Pulsar otra vez [EJ/DET] para detener la ejecución.

13) Cerrar la ventan del Sistema para salir del simulador.

14) Cambiar el valor de recarga en el programa fuente a -300, ensamblar y simular de nuevo. Observar que el parpadeo es ahora más rápido.

15) Salir del simulador y pulsar el botón [ANALIZA] del entorno. Se entra en el programa visualizador del analizador lógico de UVI51. Se puede observar la evolución temporal (cronograma) del nodo 2 (denominado led).

16) Salir del (visualizador del) analizador.

Ejercicio 1.2: Manejo de tablas (memoria de programa y de datos)

A partir del fichero EJ_1_1.CNX crear un archivo EJ_1_2.CNX que defina únicamente un microcontrolador y un programa fuente de nombre EJ_1_2.SRC (no se requiere ningún periférico externo).

Editar el archivo fuente EJ_1_2.SRC con el siguiente programa:

; Programa del ejercicio 1.2 de SDP1

```
contador      data    20H    ;definición de variable de RAM interna "contador"

                org 0
                ajmp  inicio    ;salta a la primera instrucción

tabla: db 0,1,2,3,4,5,6,7,8,9 ;define una tabla en la mem de programa con los nº de 0 al 9

                org 100H
inicio: mov     dptr, #tabla    ;DPTR apunta al comienzo de la tabla
          mov     contador,#10 ;guarda el valor 10 en la dirección 20H
bucle:  clr      a              ;pone el acumulador a cero
          movc    a,@a+dptr     ;guarda el dato de la tabla apuntado por a+dptr en el acumulador
          add     a,#2          ;incrementa el dato en dos unidades
          movx    @dptr,a       ;guarda el dato incrementado en memoria RAM externa
          inc     dptr          ;apunta a la siguiente posición de la tabla
          djnz    contador,bucle ;decrementa contador y salta a bucle si contador≠0
          sjmp    inicio        ;salto a la primera instrucción

end
```

En este programa se define una tabla de constantes en memoria de programa (ROM) mediante la directiva db. Al ejecutarse el programa, se entra en un bucle infinito en el que se cargan el registro dptr, y la posición de memoria ram interna *contador* con los valores 2 (tabla) y 10 respectivamente.

Dentro de este bucle indefinido existe un bucle que se itera 10 veces, bajo el control de la variable contador. Cada iteración del bucle copia el contenido de una posición de la tabla de constantes (ROM) en la RAM externa, después de sumarle 2.

Ensamblar el programa y entrar en el simulador. Ejecutar paso a paso observando cómo se realizan las copias (+2) de las constantes en la memoria externa (Xram). La evolución de la variable contador se puede observar en la posición 20h de la RAM interna o en la ventana de símbolos utilizados.

Ejercicios opcionales

Ejercicio 1.3: Manejo de RAM EXTERNA (datos)

Crear un archivo EJ_1_3.CNX que defina únicamente un microcontrolador y un programa fuente de nombre EJ_1_3.SRC (no se requiere ningún periférico externo).

Editar el archivo fuente EJ_1_3.SRC. Escribir un programa que inicialice una tabla de 10 valores en **RAM externa** con los números 0 al 9. El programa debe entrar después en un bucle indefinido que incremente los 10 valores en cada iteración. Para lograrlo, se empleará un bucle interno que en cada iteración incrementará una de las posiciones de la tabla. Se recomienda utilizar, entre otras, las instrucciones: INC DPTR, INC A y DJNZ Ri,etiqueta

Cuando el micro recibe alimentación, o se produce un reset, no inicializa la RAM externa. Para inicializar la RAM externa existen dos alternativas. La primera consiste en definir una tabla de constantes en ROM y trasladarla a la RAM externa. La segunda consiste en escribir los valores directamente, uno a uno o mediante un bucle. En este ejercicio, como los valores siguen una secuencia de 0 a 9 es fácil emplear un bucle (buclel).

Comprobar el funcionamiento del programa con el simulador. Después de haber ejecutado y detenido el programa, hacer un reset del micro [RESET] y modificar manualmente el contenido de la RAM externa. Para modificar la RAM externa basta con hacer click con el ratón sobre el valor a modificar. Se introduce el nuevo valor con el teclado (en hexadecimal) y se termina con Intro.

Ejecutar paso a paso para comprobar que la tabla se inicializa correctamente.

Ejercicio 1.4: Tabla en ROM (constantes)

Crear un archivo EJ_1_4.CNX que defina únicamente un microcontrolador y un programa fuente de nombre EJ_1_4.SRC (no se requiere ningún periférico externo).

Pulsar [ej_1_4.src] para editar el programa. El programa debe definir en memoria ROM (constantes) el texto *ALARMA*. Después debe trasladar estos caracteres a la pila que debe ubicarse a partir de la dirección 30h. Finalmente se debe entrar en un bucle muerto. Aunque la pila se puede utilizar para almacenar variables, de forma temporal, en la memoria de datos interna, su mayor utilidad radica en poder realizar (y anidar) llamadas a subrutinas.

Ensamblar el programa y simular para comprobar su funcionamiento.

Ejecutar de forma continua. Detener la ejecución. Observar que la posición de RAM interna apuntada por *SP* aparece en color verde. Es necesario tener en cuenta que la pila no debe invadir zonas de la RAM utilizadas por variables.

Ejercicio 1.5

Modificar el programa para que el texto se traslade a los registros R0 ... R5 del banco de registros 1 en lugar de copiarse en la pila. El banco seleccionado se muestra en color amarillo. Ensamblar y comprobar con el simulador. Utilizar en el programa la instrucción MOVC A, @A+DPTR y direccionamiento indirecto.