PRÁCTICA 3: DISEÑO DE UNA PLATAFORMA INDEPENDIENTE

OBJETIVOS:

- El objetivo principal de esta práctica es finalizar el proyecto en el que habéis estado trabajando hasta conseguir un sistema empotrado autónomo de los ordenadores del laboratorio con el que se pueda jugar directamente. Para ello vais a:
    - Utilizar la pantalla LCD para visualizar el tablero.
    - Cargar vuestro código en la memoria Flash de la placa mediante el estándar JTAG, de forma que al encenderla se pueda jugar sin necesidad de conectarse ni descargar el programa.

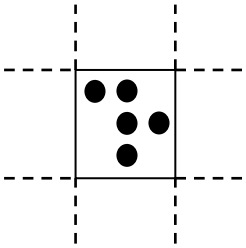
ESTRUCTURA DE LA PRÁCTICA:

La práctica consta de dos apartados obligatorios:

A. PANTALLA LCD

Debéis tener en cuenta los siguientes requisitos:

1. Primero se mostrará una **pantalla inicial** con las instrucciones para jugar y la leyenda “Pulse un botón para jugar”.
2. A continuación aparecerá el **tablero**, que será un cuadrado del máximo tamaño posible para que se vea entero en el LCD. Las columnas y las filas estarán numeradas del 1 al 9.
3. En la cuadrícula se deben distinguir claramente las celdas pista y aquéllas en las que el usuario ha introducido un valor.
4. En las celdas vacías se mostrará la lista de candidatos. Como el LCD es pequeño y de resolución limitada usaremos una notación ampliamente utilizada que consiste en dividir cada celda en una cuadrícula de 9 posiciones y usar una marca visible para señalar los candidatos (por ejemplo, un círculo), dejando en blanco los que no lo son. Por ejemplo, en la siguiente figura se observa una celda vacía donde el sistema indica tras el cálculo que son candidatos los números 1, 2, 5, 6 y 8, y no lo son 3, 4, 7 y 9.



5. Se mostrará en pantalla el **tiempo total** transcurrido en la partida y el tiempo invertido en los **cálculos** de vuestro programa (no incluye el tiempo dedicado por el usuario a pensar e introducir un valor).
6. En la pantalla aparecerá también la **leyenda**: "Introduzca Fila A para acabar la partida". Esta funcionalidad requiere la modificación de la máquina de estados. Sería interesante poder confirmar dicha acción.
7. El usuario irá introduciendo valores (fila, columna, dato). Tras cada inserción el sistema recalculará las listas de candidatos y comprobará la existencia de errores. Si hay algún **error** se mostrarán claramente todas las celdas involucradas en ese error (por ejemplo dibujando el recuadro de sus celdas con trazo más grueso o más oscuro o invirtiendo los colores). Se deben resaltar todas las celdas involucradas, incluidas las pistas, para facilitar que el usuario descubra la celda errónea. En todo caso se debe poder seguir distinguiendo pistas de valores de usuario.
8. Cuando se acabe la partida (tablero completado o terminada prematuramente por el usuario), se visualizará el resultado y la leyenda "Pulse un botón para jugar". Lógicamente si el usuario pulsa un botón comenzará una nueva partida.
9. Podéis añadir cualquier otra cosa que os parezca interesante, siempre que respetéis los puntos anteriores.

Para realizar este apartado proporcionamos un guión de prácticas en el que se explica cómo funciona el LCD y un ejemplo en el que se utiliza una biblioteca con funciones básicas del LCD. Como siempre, tratad de entender lo que uséis porque os podrá evitar muchos errores.

B. PLATAFORMA AUTÓNOMA

- **Queremos que nuestra plataforma sea autónoma** y se pueda utilizar sin conectarse a ningún PC. Para ello vamos a cargar nuestro programa en la memoria Flash de la placa utilizando *openOCD y JTAG*. Los pasos a seguir son los siguientes:

  1. Debemos generar el binario a escribir en la memoria Flash a partir del fichero `.elf`. Esto se hace con la utilidad de gcc `objcopy`. Lo podemos hacer desde la línea de comandos como en el siguiente ejemplo:

     ```
     arm-none-eabi-objcopy -O binary prueba.elf prueba.bin
     ```

  2. A continuación volcaremos el binario a la Flash con el siguiente comando (openocd está en C:\Program Files\EclipseARM\openocd-0.7.0\bin):

     ```
     openocd-0.7.0.exe -f test/arm-fdi-ucm.cfg -c "program
     prueba.bin 0x00000000"
     ```

  Al encender la placa de nuevo, nuestro código estará almacenado en la Flash, pero allí no podemos ejecutarlo correctamente porque no funcionarán las escrituras y porque las direcciones reales no cuadrarán con las que ha utilizado

el linker. Así que debemos copiarlo a la RAM y a partir de ahí podremos hacer el sudoku como antes. Para ello hay que añadir el código que: (1) inicialice el controlador de memoria (que antes estaba dentro de un script que se ejecutaba al conectarse a la placa), (2) copie el contenido de la ROM a la memoria RAM al comienzo de la ejecución, y por último, (3) salte al Main recién copiado en la RAM. Esta nueva funcionalidad estará dentro de la rutina de tratamiento del reset. En moodle encontraréis un ejemplo que muestra cómo se hacen estos pasos. Ese ejemplo incluye otras funcionalidades que no son necesarias porque las hace la función de inicio de nuestro proyecto que seguiremos utilizando. Los pasos a seguir son:

3. Estudiar el ejemplo que hemos puesto en moodle
4. Identificar el código que inicializa el controlador de memoria (**tanto las instrucciones como los datos guardados en "*SMRDATA*"**)
5. Identificar el código que copie el contenido de la ROM a la memoria RAM al comienzo de la ejecución. Este código utiliza dos etiquetas definidas en el linker script. **Debéis entender qué hace con ellas.**
6. Añadir esos códigos a vuestro `44binit`, justo en el mismo lugar en el que estaban en el ejemplo (en `Resethandler`, tras desactivar las interrupciones).
7. Después tenéis que eliminar la parte del `44binit` que actualizaba el controlador de memoria, porque ya lo hace el código que habéis añadido.
8. Cuando metáis vuestro código en la flash y reseteéis podéis observar cómo la ejecución empezará en las direcciones de la flash y al llevar al salto al Main pasaréis a las direcciones de la RAM. A partir de ahí vuestro código se ejecutará como siempre.

APARTADO OPCIONAL:

- La introducción de Fila, Columna y Dato con los botones es tediosa. Como apartado opcional podéis utilizar otros métodos para interactuar con el humano, tales como la pantalla táctil, el teclado o uno de los puertos serie de las placas para enviar los movimientos desde un PC del laboratorio.

  Tendréis disponibles todas las bibliotecas de la placa, y se valorará el uso de cualquier periférico que aporte valor al dispositivo.

- Según el método de entrada de movimientos, podría ser interesante que cuando el usuario introduzca un valor, éste aparezca en la celda adecuada de la pantalla parpadeando durante cinco segundos. Durante estos cinco segundos el usuario podría cancelar el movimiento pulsando cualquier botón o pinchando en cualquier parte de la pantalla.

MATERIAL DISPONIBLE:

En la página de la asignatura en Moodle podéis encontrar el siguiente material de apoyo:

- Un guión dónde se explica el funcionamiento del LCD.
- Códigos fuentes para usar el LCD con algunas funciones básicas.
- Un guión en el que se explica cómo programar la memoria flash.
- Códigos fuente con un ejemplo en el que se programa la flash de la placa.
- Para los apartados opcionales:
    - Un guión adicional que explica el puerto serie
    - Códigos fuente con las funciones básicas.

EVALUACIÓN DE LA PRÁCTICA

Está práctica se presentará el 22 de diciembre. La memoria el 15 de enero.
La evaluación de esta práctica y la presentación será individual (la memoria única).

ANEXO 1: REALIZACIÓN DE LA MEMORIA FINAL

Esta memoria incluirá el trabajo realizado en la práctica 2 y 3. La memoria de la práctica tiene diseño libre (recomendable revisar el documento Redacción de una Memoria Técnica, disponible en la web de la asignatura), pero es obligatorio que incluya los siguientes puntos:

1. Resumen ejecutivo (una cara como máximo). El resumen ejecutivo es un documento independiente del resto de la memoria que describe brevemente qué habéis hecho, por qué lo habéis hecho, qué resultados obtenéis y cuáles son vuestras conclusiones.
2. Descripción de la librería desarrollada para medir tiempos y resultados obtenidos al medir las funciones desarrolladas en la práctica 1. Comparar con las estimaciones realizadas en la práctica 1, utilizando el número de instrucciones ejecutadas y el tiempo de simulación.
3. Explicación de la estructura de vuestro proyecto.
4. Descripción de la gestión de la entrada/salida de la versión final de vuestro proyecto incluyendo los diagramas de estados pertinentes en el que se vea cómo funciona, detallando que acciones se realizan en cada estado y como se genera la salida correspondiente en la pantalla.
5. Describe la gestión de excepciones.
6. Explicar cómo conseguís que vuestro código se cargue en la memoria RAM.
7. Código fuente comentado (sólo el que habéis desarrollado nuevo para la práctica 2 y 3). Como siempre, cada función debe incluir una cabecera en la que se explique qué hace, qué parámetros recibe...
8. Descripción de los problemas encontrados en la realización de la práctica y sus soluciones.
9. Conclusiones valorando vuestro proyecto final.

Se valorará que el texto sea **claro y conciso**. Incluir en la memoria las partes de vuestro código que habéis desarrollado para cada sección. Cuánto más fácil sea entender el funcionamiento del código y vuestro trabajo, mejor.

ANEXO 2: FORMA DE ENTREGA DE LA MEMORIA

La forma de entrega de la memoria será idéntica a las anteriores. Debéis entregar la memoria en formato pdf junto con las fuentes originales (texto plano) en un fichero comprimido en formato zip. Se mandará un único fichero por pareja con el siguiente nombre:

p2-3_NIP-Apellidos_Alumno1_NIP-Apellidos_Alumno2.zip

Por ejemplo: p2-3_345456-Gracia_Esteban_45632-Arribas_Murillo.zip