

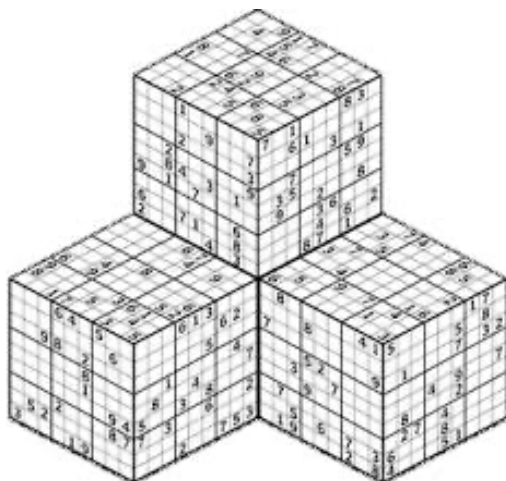
# PROYECTO HARDWARE

3º DE GRADO EN INGENIERÍA INFORMÁTICA, CURSO 2025/2026

UNIVERSIDAD DE ZARAGOZA

## PRÁCTICA 3

### DISEÑO DE UNA PLATAFORMA INDEPENDIENTE



## PRÁCTICA 3: DISEÑO DE UNA PLATAFORMA INDEPENDIENTE

|  |   |
|--|---|
| Objetivos.....                           | 3 |
| Estructura de la práctica .....          | 3 |
| A.    Juego completo y pantalla LCD..... | 3 |
| B.    Plataforma autónoma .....          | 5 |
| C.    Pantalla Táctil .....              | 6 |
| Material disponible.....                 | 6 |
| Evaluación de la práctica .....          | 7 |
| Anexo I. Realización de la memoria ..... | 7 |
| Anexo II. Entrega de la memoria .....    | 7 |

## OBJETIVOS

El objetivo principal de esta práctica es finalizar el proyecto en el que habéis estado trabajando hasta conseguir un sistema empotrado autónomo, completamente independiente de vuestros ordenadores, 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.
- Utilizar la pantalla táctil como mecanismo alternativo de entrada.

## ESTRUCTURA DE LA PRÁCTICA

La práctica consta de tres apartados:

### A. JUEGO COMPLETO Y PANTALLA LCD

Vamos a dotar al juego de una interfaz gráfica. Para ello, vamos a introducir un periférico nuevo: la pantalla LCD para mostrar información.

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 la pantalla LCD. Las columnas y las filas estarán numeradas del 1 al 9.
3. En la cuadrícula se deben distinguir claramente las celdas de pista y aquellas en las que el usuario ha introducido un valor.
4. En las celdas vacías se mostrará la lista de candidatos. Como la pantalla LCD es pequeña y de resolución limitada, usaremos una notación que consistirá 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.

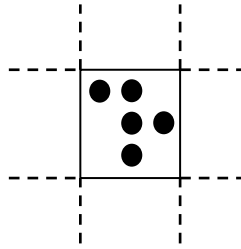


Figura 1. Celda vacía

5. Además, se mostrará en pantalla el **tiempo total** transcurrido en la partida.
6. En la pantalla aparecerá también la **leyenda**: "Introduzca Fila 0 para acabar la partida". Es decir, ahora sí que se podrá poner un 0 en Fila.
7. El usuario irá introduciendo valores (fila, columna, valor). 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, para facilitar que el usuario descubra la celda errónea. En todo caso, se debe poder seguir distinguiendo las pistas de los valores de usuario.
8. Cuando se acabe la partida (el tablero se haya completado o la partida haya sido terminada prematuramente por el usuario), se visualizará el resultado y la leyenda "Pulse un botón para jugar". Lógicamente, si el usuario pulsa cualquier 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, tenéis disponible en moodle un guion 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. Además, os facilitamos un fichero **main.c** con ejemplo de uso del LCD, y se suministran ejemplos y librerías de la placa. Hay que tener en cuenta que son los ficheros facilitados por el fabricante, pero que pueden tener bugs o malas prácticas. Podéis estudiarlos con sumo cuidado y decidir qué hacéis. Como siempre, tratad de entender lo que uséis, porque os podrá evitar muchos errores y ahorro de tiempo.

**Mostrar información en la pantalla:** las bibliotecas del LCD dibujan sobre una zona de memoria RAM principal a la que llamaremos **frame buffer**. Cuando queráis actualizar el contenido en el LCD, será necesario transferir del *frame buffer* a la memoria interna del LCD. Esta operación se realiza programando un nuevo dispositivo denominado **DMA** (*Direct Memory Access*). El procesador programa la operación del DMA (@origen y destino y la cantidad de memoria a transferir) y queda libre para realizar otras tareas a la espera de que el DMA acabe. Al acabar, el DMA interrumpe la ejecución del procesador para

notificárselo. Los ficheros del LCD de bajo nivel tienen primitivas genéricas para ayudar a pintar. Si lo consideráis necesario, podéis añadir primitivas nuevas a ese fichero.

## B. PLATAFORMA AUTÓNOMA

- **Queremos que nuestra plataforma sea autónoma** y que 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 el *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:

1. Estudiar el ejemplo que hemos puesto en moodle
2. Identificar el código que inicializa el controlador de memoria (**tanto las instrucciones, como los datos guardados en "SMRDATA"**)
3. Identificar el código que copia 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.**

4. Añadir esos códigos a vuestro `44binit`, justo en el mismo lugar en el que estaban en el ejemplo (en `Resethandler`, tras desactivar las interrupciones).
5. Después tenéis que eliminar la parte del `44binit` que actualizaba el controlador de memoria, porque ya lo va a hacer el código que habéis añadido.
6. Cuando metáis vuestro código en la Flash y reseteéis, podéis observar cómo la ejecución empezará en las direcciones de la Flash, y al llegar al salto al main pasaréis a las direcciones de la RAM. A partir de ahí vuestro código se ejecutará como siempre.
7. **Después de presentar vuestra práctica al profesor, debéis dejar la placa con el programa de demostración, es decir, tal y como estaba al comienzo del curso.** Para ello, debéis **restaurar el fichero `backup_flash.bin` original**, disponible en moodle.

### C. PANTALLA TÁCTIL

En este apartado se va a introducir otro periférico: la pantalla táctil o TouchPad (TP) para validar la jugada.

Dado que la introducción de Fila, Columna y Valor con los botones puede ser tediosa, o algunas placas pueden tener problemas con alguno de los botones, en este apartado vamos a habilitar otro método para interactuar con el juego: la pantalla táctil.

No obstante, dado que son dispositivos antiguos, las pantallas no son tan precisas como las actuales, por lo que será necesario que establezcáis un método de calibración inicial para hacer que funcione algo mejor.

Para que funcione bien todo, aun con poca precisión, quizá sea buena idea elegir primero la región, mostrarla aumentada para que el usuario pueda elegir la celda y posteriormente el candidato. No obstante, en este apartado tenéis mucha libertad para presentar una solución funcional.

### MATERIAL DISPONIBLE

En el moodle de la asignatura podéis encontrar el siguiente material de apoyo:

- Un guion donde se explica el funcionamiento del LCD.
- Códigos fuente para usar el LCD y el TP con algunas funciones básicas.
- Un guion en el que se explica cómo programar la memoria Flash para el apartado de la plataforma autónoma.
- Código fuente con un ejemplo en el que se programa la Flash de la placa.

## EVALUACIÓN DE LA PRÁCTICA

La fecha límite de entrega de la práctica será el próximo miércoles 14 de enero (hasta las 23:59), mientras que la demostración será o bien el jueves 15 o el viernes 16 a las 12:00 en el laboratorio Informática 2, en función del día que os venga mejor a cada grupo.

La demostración y la memoria se realizarán por grupo, aunque finalmente, la nota de esta práctica será **individual**.

## ANEXO I. REALIZACIÓN DE LA MEMORIA

En la memoria debéis seguir las pautas de la guía proporcionada por la asignatura. Es obligatorio que incluya:

1. Resumen ejecutivo (una cara máximo).
2. Código fuente comentado (sólo el que habéis desarrollado nuevo para la práctica). Como siempre, cada función debe incluir una cabecera en la que se explique qué hace, qué parámetros recibe...
3. Explicar cómo conseguís que vuestro código se transfiera desde la memoria Flash a la RAM para ejecutarse finalmente.
4. Descripción de los problemas encontrados en la realización de la práctica y sus soluciones.
5. 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. Cuanto más fácil sea entender el funcionamiento del código y vuestro trabajo, mejor.

## ANEXO II. ENTREGA DE LA MEMORIA

La forma de entrega de la memoria será similar a las anteriores, aunque no es necesario hacer una presentación pública, sino una demostración del funcionamiento de vuestro proyecto. No obstante, sí que debéis entregar la memoria en formato PDF junto con las fuentes originales (texto plano).

Se debe mandar un único fichero por pareja. El fichero se nombrará de la siguiente manera:

p3\_NIP-Apellidos\_Estudiante1\_NIP-Apellidos\_Estudiante2.zip

Por ejemplo: p3\_345456-Gracia\_Esteban\_45632-Arribas\_Murillo.zip