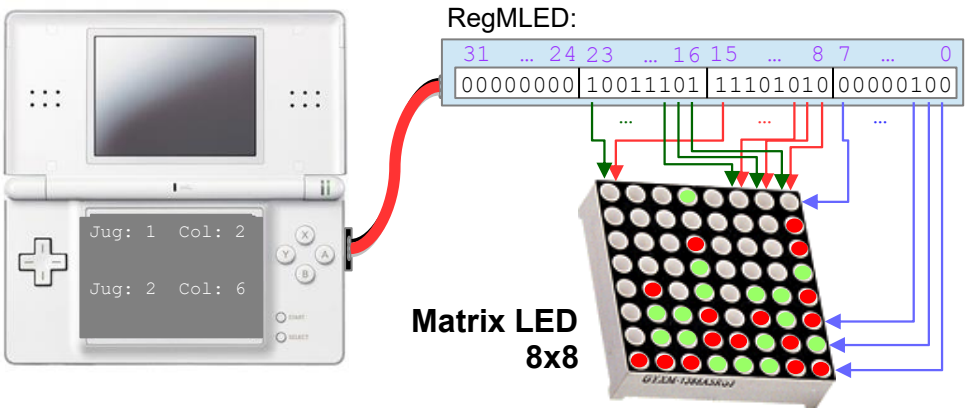
Problema 23: Matriz 8x8 LEDs (Ex. 1ª Conv. 2017-18)

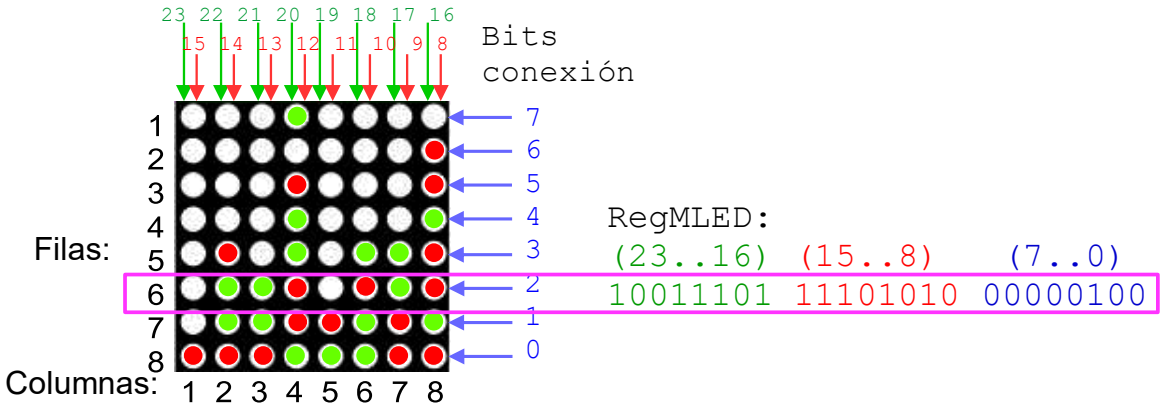
Se propone conectar una interfaz para controlar una matriz de 8x8 LEDs bicolor (1588ABEG-5) con la NDS, con el fin de implementar el juego del 4 en ralla:



La interfaz dispone de un único registro de Entrada/Salida de 32 bits, de nombre simbólico RegMLED, pero solo se utilizarán los 24 bits de menor peso, los cuales están divididos en los siguientes campos:

Campo	*Bits*	*Función*
`Columnas en Verde`	23..16	Activan (=0) o desactivan (=1) el color verde de los 8 LEDs de la fila seleccionada por el campo de Filas
`Columnas en Rojo`	15..8	Activan (=0) o desactivan (=1) el color rojo de los 8 LEDs de la fila seleccionada por el campo de Filas
`Filas`	7..0	Activan (=1) la fila que se tiene que iluminar en cada momento

El problema de gestionar 64 LEDs, cada uno con 2 colores, es que resulta muy costoso conectar los 128 cables necesarios para el control individual del color de cada LED. Por este motivo, el dispositivo dispone de un cable por cada fila más dos cables por cada columna, de manera que se pueda aplicar la técnica del "barrido" por filas, que consiste en enviar corriente eléctrica sobre un único cable de fila y utilizar el voltaje de los 16 cables de las columnas para indicar el estado del color de los 8 LEDs de la fila seleccionda. El siguiente esquema muestra un ejemplo de cómo se controla el color de los LEDs de la fila 6 (bit 2 = 1):



Si se recorren todas las filas secuencialmente, fijando el color de las 8 columnas de cada fila durante un breve periodo de tiempo (> 25 ms), repitiendo el proceso a una frecuencia suficientemente alta (≥ 30 Hz), el ojo humano no detecta el hecho de que solo hay una fila que está emitiendo luz en todo momento, sino que tiene la sensación de que la matriz está totalmente iluminada (fenómeno de persistencia visual).

La interfaz no genera interrupciones, de modo que se pide utilizar la RSI del *timer* 0, programada a una frecuencia de 240 Hz, para realizar el barrido periódico de la matriz.

Por otro lado, para implementar la dinámica del juego del 4 en ralla, se pide utilizar la RSI del *timer* 1, programada a una frecuencia de 3 Hz, que se encargará de generar el efecto de caída de las fichas (ver más adelante).

El programa a implementar controlará las fichas de dos jugadores. El jugador 1 llevará las rojas y el jugador 2 las verdes. El tablero consistirá en 7 filas (de la 2 a la 8) por 8 columnas. Por turnos, cada jugador tendrá que seleccionar la columna sobre la que quiere soltar su ficha. Para esto, sobre la fila superior (la 1) se mostrará una ficha del color del jugador que tiene el turno, posicionada inicialmente en la columna 4. El jugador podrá cambiar la posición (columna) de la ficha con las teclas `KEY_RIGHT` y `KEY_LEFT`. El jugador podrá soltar la ficha pulsando `KEY_SELECT`, si la columna actual no está llena (si no hay ficha en la fila inferior). Cuando se realice la selección de la columna, aparte de escribir dicha selección por la pantalla inferior de la NDS, la ficha empezará a caer hacia las filas inferiores hasta que llegue a la fila 8 o hasta que encuentre otra ficha en la posición inferior. En el momento que la ficha se detiene, el programa debe comprobar si hay cuatro en ralla; en caso negativo la partida continúa, en caso positivo se declara el vencedor por la pantalla de la NDS y la partida finaliza; el programa ya no hace nada más, hasta que se reinicie la NDS (no hay tareas independientes).

Se dispone de las siguientes rutinas, ya implementadas:

Rutina	*Descripción*
`inicializaciones()`	Inicializa el *hardware* (pantalla, interrupciones, etc.)
`scanKeys()`	Captura el estado actual de los botones de la NDS
`int keysDown()`	Devuelve un patrón de bits con los botones activos
`int comprobar_4(unsigned char *tablero, int nfil, int ncol)`	Comprueba si existe 4 en ralla en un tablero de `nfil` x `ncol` posiciones, devolviendo 1 en caso afirmativo y 0 en caso negativo (tiempo de ejecución > 0,0001 s)
`mostrar_ganador(int jugador)`	Muestra por pantalla el mensaje de resultado de la partida
`swiWaitForVBlank()`	Espera hasta el próximo retroceso vertical
`printf(char *format,...)`	Escribe un mensaje en la pantalla inferior de la NDS

En la pantalla inferior de la NDS de la figura inicial se muestran diversos ejemplos de la salida de texto que indican la selección de cada columna, que siguen el siguiente formato:

```
Jug: i (tabulador) Col: j          (1 ≤ i ≤ 2)  (1 ≤ j ≤ 8)
```

Se sugiere el uso de las siguientes variables globales:

```
unsigned char matriz[8][8];   // valores de los LEDs, en cada
                              // posición:
                                   // = 0 → vacía (apagado)
                                   // = 1 → ficha jugador 1 (rojo)
                                   // = 2 → ficha jugador 2 (verde)
unsigned char turno = 1;           // ident. del jugador actual
unsigned char fil = 1, col = 4;    // fila y columna de la ficha
                                   // que se está colocando
```

Para la implementación del programa principal y la RSI del *timer* 1, puede ser conveniente usar una variable que indique en qué fase se encuentra el juego en cada instante (seleccionar columna, caída ficha, comprobar ganador, final partida).

A cada activación de la RSI del *timer* 1, si el programa se encuentra en la fase de caída, la ficha solo bajará una posición (fila inferior), en el caso de que sea posible, obviamente. Si la

ficha se encuentra en la fila 8 o si se detecta otra ficha en la fila inferior a la actual, se habrá terminado la fase de caída.

A cada activación de la RSI del *timer* 0 habrá que acceder a una de las filas de la matriz y generar los 24 bits de información para el refresco de esa fila. Obviamente, este proceso se debe realizar periódicamente para todas las filas de la matriz.

Se pide:

Programa principal y variables globales en C, RSIs de los *timers* en ensamblador.