Problema 5: Generador de sonido

Se propone interactuar con el *hardware* de generación de sonido de la NDS. Concretamente, se trata de implementar unas funciones para activar notas musicales con una determinada frecuencia y duración.

Las rutinas a implementar son las siguientes:

Rutina	*Descripción*
`activar_nota(char canal, short frec, short vol)`	Activa la reproducción de una nota por un canal de sonido, a una determinada frecuencia (en Hz) y a un determinado volumen (127..0) durante un tiempo indefinido
`RSI_timer0()`	Rutina de servicio de interrupciones del *timer* 0: se encargará de controlar la duración de la nota actual y de activar la nota siguiente

Para accionar la nota en cada canal hay que acceder al registro de control (`0400 04X0`) y al registro de *timer* (`0400 04X4`) del canal especificado, donde `X` es el número de canal como dígito hexadecimal (de 0 a F). Los campos de dichos registros significan lo siguiente:

SOUND_X_CNT – SOUND Channel X Control Register (32 bits)

Bits	*Campo*	*Descripción*
6..0	`Volume`	Nivel de volumen, de 0 a 127 (0 es silencio)
28..27	`Repeat Mode`	01: bucle infinito, 10: una vez
31	`Start / Stop`	0: Parar, 1: iniciar

SOUND_X_TMR – SOUND Channel X Timer Register (16 bits)

Bits	*Campo*	*Descripción*
15..0	`Timer value`	Divisor de frecuencia de entrada para que la frecuencia de salida sea igual a `frec`: `Timer value = -(33513982/2) / frec`

Se dispone de las siguientes rutinas, ya implementadas:

Rutina	*Descripción*
`inicializaciones()`	Realiza inicializaciones del *hardware*

`tareas_independientes()`	Tareas independientes de la generación de sonido (ej. gestión de un juego)
`swiWaitForVBlank()`	Espera retroceso vertical
`printf(const char * format,...)`	Imprime por pantalla un mensaje de texto con formato

Para cada nota se debe utilizar una estructura de información con los siguientes campos:

```
typedef struct {
     short freq;    // frecuencia de la nota (Hz)
     short time;    // tiempo de la nota (centésimas de segundo)
     short vol;     // volumen de la nota (0..127)
} info_note;
```

Todas las notas a tocar se encuentran en un vector con un número de posiciones igual a una constante `MAX_NOTAS`:

```
info_note musica[MAX_NOTAS];
```

El funcionamiento del programa principal tiene que ser el siguiente:

- inicializaciones
- leer primera nota
- activar primera nota
- bucle principal
- tareas independientes
- sincronización de pantalla
- escribir el índice de la nota actual (sólo cuando haya un cambio de nota)
- fin de bucle principal

Mientras tanto, la RSI del *timer* 0 se activará a 100 Hz y se encargará de controlar el tiempo de cada nota y de cargar la siguiente. Cuando llegue a la última nota, volverá a empezar desde el principio.

Para realizar la división para el cálculo del divisor de frecuencia del controlador de sonido se llamará a una función de la BIOS con la instrucción de lenguaje máquina `swi 9`, pasando el numerador en R0 y el denominador en R1: la función devuelve el cociente (con signo) en R0, el resto en R1 y el valor absoluto del cociente en R3

Se pide:

Programa principal en C y la RSI del *timer 0* y la rutina `activar_nota()` en ensamblador. Se utilizará el canal de sonido 0.