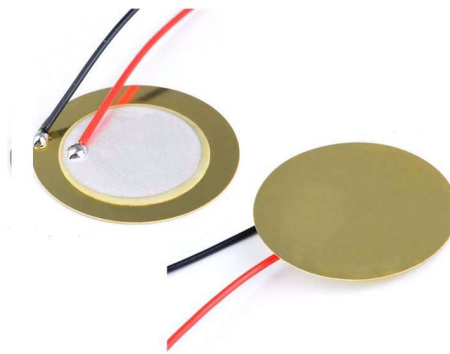


SISTEMAS BASADOS EN MICROPROCESADOR
PRÁCTICA 6. ZUMBADOR CERÁMICO
Grado en Ingeniería Electrónica Industrial (3^{er} curso)
Curso 2025/2026

1 INTRODUCCIÓN

Un zumbador (en inglés *buzzer*) es un transductor electroacústico que produce un sonido o zumbido continuo o intermitente de un mismo tono (generalmente agudo). Sirve como mecanismo de señalización o aviso y se utiliza en múltiples sistemas, como en automóviles o en electrodomésticos, incluidos los despertadores. Está basado en las propiedades de los cristales piezoeléctricos (poliéster o cerámica), que se deforman cuando se les aplica una tensión entre sus caras y que, actuando como transductor electroacústico, es utilizado para la reproducción de sonido. Si se une a una de sus caras un cono abocinado, éste sufrirá desplazamientos capaces de producir una presión oscilante dentro de un rango de frecuencia audible, es decir, baja frecuencia. Estos altavoces son sencillos, baratos y capaces de radiar con muy poca potencia eléctrica. Aunque su respuesta es óptima a la hora de reproducir altas frecuencias, resultan incapaces de reproducir rangos de baja frecuencia. En esta práctica dispondremos de la lámina zumbadora desnuda para reducir la emisión de sonido y facilitar el desarrollo de la práctica dentro del laboratorio.



Estos módulos se conectan al *Launchpad* mediante los puertos de expansión laterales. En concreto, se usará el puerto P1.0¹, disponible en el puente J7 como salida de audio digital. Es necesaria una conexión adicional a masa o alimentación, que se puede tomar del J5.

1.1 Objetivos

- Entender el funcionamiento de los zumbadores y conceptos básicos de música.
- Hacer una librería para poder generar notas musicales con el MSP430.

2 ESTUDIO TEÓRICO²

Un sonido no es más que una vibración del aire que nuestros oídos pueden captar. Un sonido que tiene un determinado tono, depende de la frecuencia a la cual vibra el aire. Las notas musicales son vibraciones de frecuencias determinadas. No obstante, una vibración sinusoidal a una frecuencia concreta, produce un sonido puro que nosotros percibimos como

1. El puerto se ha elegido por tener disponible la salida de algún timer y por no ser usado por el módulo de teclado.

2. Fuente: <http://latecladeescape.com/t/Frecuencia+de+las+notas+musicales>

un pitido de un determinado tono. En el sistema musical occidental, se ha acordado utilizar sólo unas frecuencias concretas, a las cuales llamamos notas.

Dividimos las posibles frecuencias en porciones que llamamos *octavas* y cada octava en 12 porciones que llamamos *notas*. Cada nota de una octava tiene exactamente la mitad de frecuencia que la misma nota en la octava superior. El oído humano capta solamente frecuencias que estén por encima de los 20Hz y por debajo de los 20kHz (muy aproximadamente). Así pues, y con mucha suerte, sólo podemos oír unas 10 octavas como mucho, con doce notas cada una.

La nota *La* sirve como referencia para todas las demás. A menudo se denomina "nota de afinar". Se produce un *La* de afinar cuando el aire vibra 440 veces por segundo, es decir a 440 Hz. Por convención, a la octava que contiene esta nota *La* se le suele considerar la cuarta. Hay otra nota *La*, de una "octava" superior (la quinta octava) cuando el aire vibra a 880 Hz, y otra más cuando vibra a 880*2 (sexta octava), y otra a 880*4 (séptima octava), etc, del mismo modo que hay un *La* que se produce cuando el aire vibra a 440/2 (tercera octava) y otra a 440/4 (segunda octava). Las demás notas de la octava están distribuidas uniformemente según la ley exponencial de la siguiente fórmula:

$$f(n, o) = 440e^{\left((o-4) + \frac{n-9}{12}\right) \ln 2}$$

Donde *o* es la octava (entre 0 y 10) y *n* es la nota (Do=0, Do#=1, Re=2, Re#=3, Mi=4, Fa=5, Fa#=6, Sol=7, Sol#=8, La=9, La#=10, Si=11).

2.1 Ejercicio 1

Busque en la documentación en qué función del puerto P1.0 está TA0.1 y anótela aquí _____.

2.2 Ejercicio 2

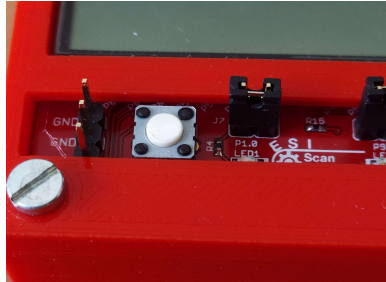
Obtenga las frecuencias de las 12 notas de las octavas cuarta a sexta, redondeando a unidades de Hz. Escriba en ensamblador del MSP430 una tabla con los valores necesarios para que la salida del `timerA` oscile con las frecuencias de dichas notas basándose en `SMCLK`³ como fuente de reloj y el temporizador en modo `CONTINUOUS` funcionando a una frecuencia de 1 MHz. Guarde esa tabla en un fichero llamado `snd.asm`.

3. Hay que irse a un reloj rápido como `SMCLK` para que las frecuencias generadas sean precisas. En caso de usar `ACLK`, los errores en las notas pueden llegar al 10%.

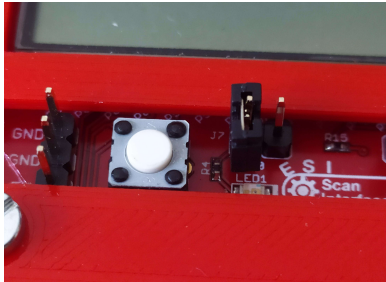
3 ESTUDIO PRÁCTICO

3.1 Conectando el zumbador

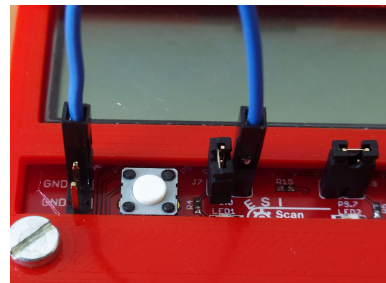
- Localice el puente J7 que está sobre el led1:



- Retírelo y vuelva colocarlo sobre el pin de la izquierda únicamente:



- Conecte uno de los terminales del zumbador en el pin de la derecha de J7 y el otro en uno de los pines del conector J5 (a la izquierda del botón):



- El zumbador debe quedar así:



3.2 Preparando el proyecto

Copie el proyecto de la práctica anterior y llámelo P6.1. Se va a reaprovechar la mayor parte del código, incluido el código C.

3.3 Módulo de sonido `snd.asm`

Añada el nuevo módulo `snd.asm` y su fichero de cabecera `snd.h`. En él se van a incluir funciones para manejar el zumbador. Las variables y servicios de este módulo son:

3.3.1 Tabla de frecuencias de notas `TabOct4`, `TabOct5`, `TabOct6`

Se trata de 3 vectores que recogen los valores a programar en el `TA0.1` para reproducir las 12 notas de cada una de las octavas. La entrada 0 será para `Do`, la 1 para `Do#`, etc. Dichos valores se obtendrán a partir de las frecuencias de cada nota halladas en el ejercicio teórico 1. Estos vectores sólo serán accesibles internamente, por lo que no se publicarán con la directiva `.global`.

3.3.2 Inicialización del módulo `sndIni`

```
void sndIni (void);
```

Inicializa el módulo de sonido. Para ello captura el `P1.0` en la función que encontró en **2.1 Ejercicio 1**, lo pone como salida y programa el `TA0.1` en modo `CONTINUOUS` con una frecuencia de reloj de 1MHz. Inicialmente la salida `TA0.1` es 0.

3.3.3 Control de sonido `sndMute`

```
void sndMute (int mute);
```

Si `mute=0` activa el sonido. En caso contrario lo desactiva. La activación/desactivación se hace poniendo `P1.0` en modo salida/entrada respectivamente. Activar el sonido no implica que se reproduzca el mismo. Es necesario generar una onda en la salida `TA0.1`.

3.3.4 Reproducir una nota `sndNota`

Reproduce una nota de una octava. También puede apagar el sonido:

```
int sndNota (uint8_t oct, uint8_t nota);
```

donde `oct` es la octava (valores posibles 0, 4, 5 y 6) y `nota` es un número entre 0 y 11.

- Cuando `oct` es 0, silencia el zumbador independientemente del valor de `nota` (haciendo `TA0.1 = 0`).
- Cuando `oct` está entre 4 y 6 inclusive, inicia la reproducción según las frecuencias de `TabOctX`.
- En caso de error devuelve -1 y 0 en caso contrario.

3.3.5 Subrutina de servicio de interrupción del timer `TA01ISR`



La generación de la señal cuadrada se hará por el puerto `P1.0`, que está conectado al `TA0.1`. Al tratarse de un CCR distinto del 0, obliga a que el modo del temporizador sea el `CONTINUOUS` y que, por tanto, se tenga que capturar la ISR correspondiente para actualizar el valor del CCR para producir una frecuencia arbitraria. Recuerde que no debe publicar la etiqueta en `.global`.

3.4 Programa principal en C

Cree una nueva tarea cooperativa llamada `Piano` que reproduzca las notas que se pulsen y las pinte en pantalla. Al ser una tarea interactiva, irá a la velocidad del `SystemTimer`, por lo

que no es necesario usar una variable `ProximaEjecucion` ni comprobar si tiene que ejecutarse o no. Simplemente lee la tecla, actúa en consecuencia y sale.

El teclado que se va a usar es el siguiente:

	C#	D#	+
C	D	E	-
	F#	G#	A#
F	G	A	B

Serigrafía del teclado

<i>Bel</i>	C	D	+
c	d	e	-
<i>CR</i>	F	G	A
f	g	a	b

Códigos ASCII de cada tecla

El programa empezará en la octava 5, sonido activado (mute = off) y animación de led activa. Las teclas disponibles serán⁴:

- *Bel* (código ASCII 7): Apaga/enciende el sonido. El icono ‘!’ del display LCD se encenderá si y solo si el sonido está apagado.
- ‘+’ y ‘-’: incrementan/disminuyen respectivamente la octava actual. Sólo se permitirán octavas de la 4 a la 6 y se reflejará en los segmentos de la batería encendiendo tantos segmentos como indique el número de la octava. Ignore ‘-’ si la octava es la 4 y ‘+’ para si la octava es la 6.
- *CR* (código ASCII 13): desactiva/activa la animación de led2 de usuario.
- *Notas* (ver códigos ASCII en la figura de arriba)⁵: mientras se mantiene pulsada, reproduce un tono de esa nota para la octava actual y se muestra su nombre en la pantalla con hasta 4 caracteres (rellene con espacios si es necesario).
- *ETX*. Borra la línea inferior de la pantalla y silencia la nota actual.

La animación del led2 consistirá en hacer parpadear el mismo con una velocidad dependiente de la octava actual: 1 Hz para la octava 4, 16 Hz para la octava 5 y 32Hz para la octava 6.

4. Cambie la tabla de códigos ASCII de teclado.asm para que se ajuste a esta nueva codificación.

5. Estamos usando la notación anglosajona de notas: c Do, d Re, e Mi, f Fa, g Sol, a La y b Si. Las letras en mayúsculas indican la nota en sostenido; es decir C es Do#, D Re#, F Fa#, G Sol# y A La#.