Elementos para la práctica

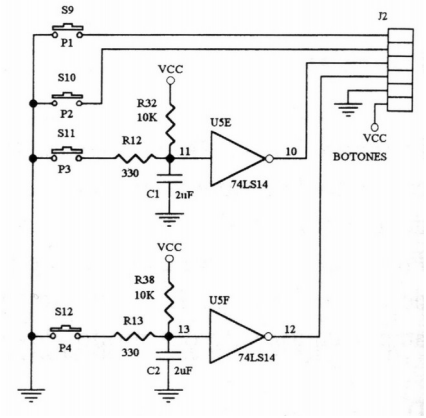
A continuación se comentan algunos de los elementos disponibles en la *Tarjeta de experimentos DISEN-EXP*.

Conmutadores sin rebote

Están disponibles 8 conmutadores que tienen la electrónica necesaria para suprimir los rebotes inherentes a cualquier apertura de circuito mecánica. Además están acompañados de leds para indicar su estado.

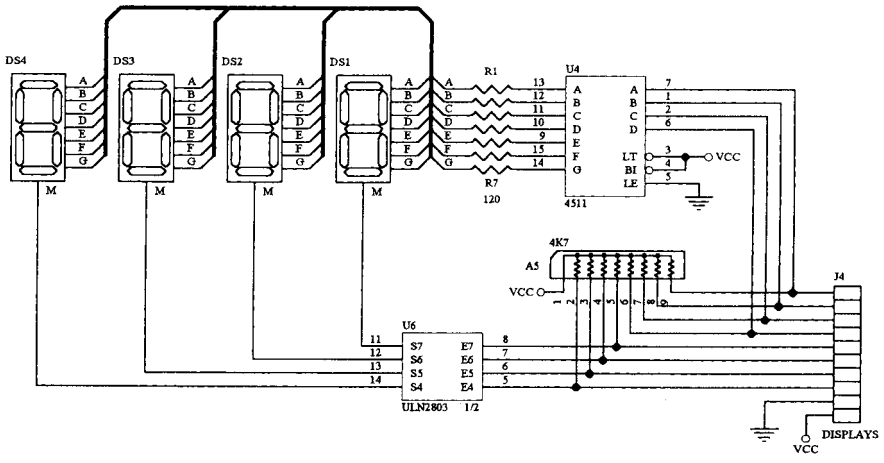
Pulsadores

Están disponibles cuatro pulsadores, situados sobre el display LCD dos de ellos con rebote y dos con los rebotes suprimidos como se muestra en el esquema siguiente.



7 segmentos

El esquema de los 7 segmentos con que cuenta la *Tarjeta de experimentos DISEN-EXP* es el siguiente:



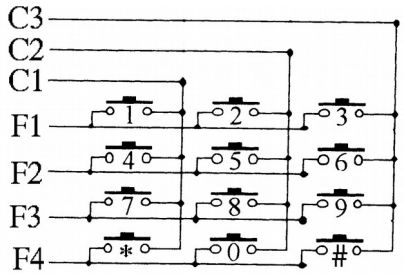
Como puede verse existe un único decodificador con lo cual, para poder representar un número con distintos dígitos, sólo es posible encender un 7 segmentos en cada instante, por lo que es necesario realizar una multiplexación temporal. La frecuencia de refresco ha de ser superior a los 100 Hz. Sin embargo el valor representado no debe cambiar a demasiada frecuencia (no más de 2Hz) para que puede ser correctamente interpretado por el usuario.

Potenciómetro

Consiste en un cursor que se desplaza sobre una resistencia cuyos extremos están conectados a 0V y 5V respectivamente. Por ello el voltaje varía linealmente entre los 0 y 5 V según se va girando el mismo. Este voltaje se podrá leer a través de una entrada analógica del HC12

El teclado numérico

El esquema del teclado disponible en *Tarjeta de experimentos DISEN-EXP* es el que se muestra en la siguiente figura:



La disposición de las teclas es la de una teclado telefónico: números del 1 al 0 y símbolos * y #. Como se ve, el teclado es una matriz de 4 filas y 3 columnas de pulsadores. La disposición del conector físico es la siguiente:

C2	F1	C1	F4	C3	F3	F2

Lectura del teclado

En el funcionamiento es necesario realizar lo que se denomina "escaneado del teclado" ya que no se puede determinar directamente la tecla pulsada: 12 teclas pero sólo 7 líneas. Inicialmente hay que poner todas los pines conectados a las filas como salida y escribir un 0 (tensión 0 V). Los pines conectados a los las columnas se configurarán como entrada con resistencia de pull-up activada. Mientras no se pulse ninguna tecla, la lectura de las columnas dará todas a 1. Cuando se pulsa una tecla, una de las columnas se pondrá a baja (un 0). Una vez detectada la pulsación es necesario dejar pasar un cierto tiempo para que la señal se estabilice, ya que en la conmutación de cualquier interruptor o pulsador siempre se producen una serie de rebotes, es decir, pasos de 0 a 5V. El tiempo de duración de estos rebotes, en el peor caso, es inferior a 20msg.

Después de haber esperado este tiempo, podemos comenzar el proceso de escaneado. Sabemos la columna en que se encuentra la tecla pulsada. Para conocer su fila se colocan todas a 1 y se va poniendo un 0 sólo en una de ellas. En el momento en que se vuelva a detectar un 0 por la columna inicial sabremos también la fila a la que pertenece la tecla y por tanto la habremos determinado exactamente.

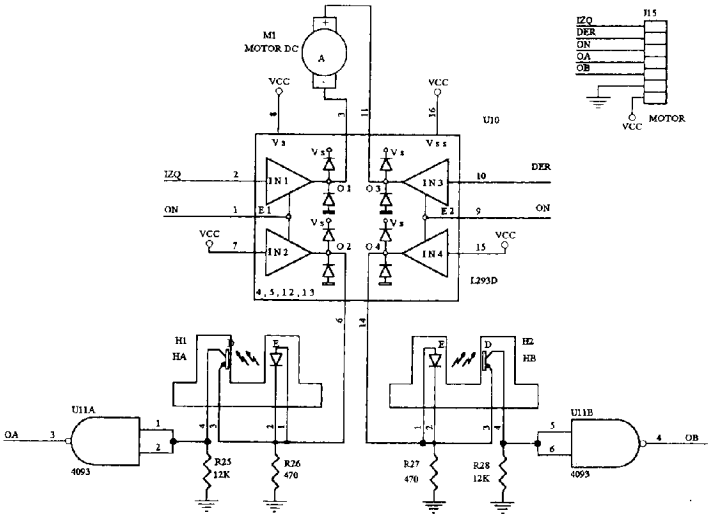
Si queremos permitir la pulsación simultánea de dos teclas el proceso tendrá que continuar para ver si aparecen más 0.

Ya que físicamente las filas no tienen nada distinto de las columnas, en toda la explicación anterior es posible cambiar fila por columna y viceversa, es decir, poner la filas como entradas y las columnas como salida, etc.

De la misma manera que es necesario esperar los rebotes debidos a la pulsación, es necesario detectar cuando se suelta la tecla y esperar un tiempo (20 msg) para que pasen los rebotes y evitar repetición de pulsaciones. Como la función que detecta las pulsaciones debe devolver el carácter en cuanto esta se produzca, y no cuando se suelte la tecla, debe ser al principio de dicha función cuando se determine si hay alguna tecla pulsada. Si es así, debe esperar a que se suelte, que pasen los rebotes y a continuación ponerse a esperar la nueva pulsación.

Motor codificador

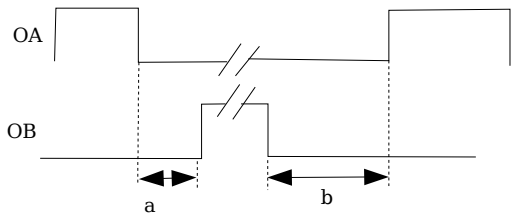
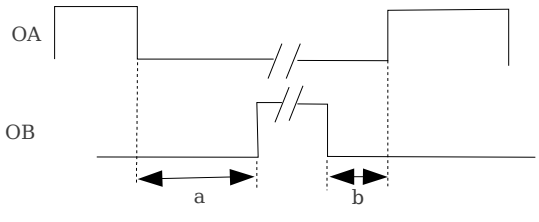
Es esquema del conjunto motor-codificador se muestra en la siguiente figura.



El control de la velocidad del motor se realiza aplicando una PWM, de ciclo de trabajo adecuado, en las entrada DER o IZQ, según en sentido de giro deseado, mientras se mantiene la otra a baja. La frecuencia máxima que permite el L293D es de 5kHz.

Cada detector infrarrojo emitirá un pulso (líneas OA y OB) cuando una de las 4 aspas, solidarias con el eje del motor, pase por él. Dada su posición relativa, los pulsos tendrán un desfase distinto dependiendo de si el motor gira a derecha o izquierda, con lo que se puede determinar el sentido de giro.

El método más fiable para detectar el sentido es utilizando la comparación entre los tiempos *a* y *b* que se muestran en el siguiente gráfico.

Girando en un sentido: (a<b)	Girando en el otro: (a>b)
	

No se garantiza que sean iguales el tamaño de dos pulsos sucesivos ni la distancia entre ellos. Es posible que en ocasiones falten alguno de esos pulsos o no se cumpla esta relación, por lo que es conveniente ir acumulando y quedarse con la condición que se de el mayor número de veces durante un cierto intervalo de tiempo.

Práctica 2

La 2ª práctica de la Parte III consistirá en realizar funciones y programas para el manejo de distintos elementos disponibles en la *Tarjeta de experimentos DISEN-EXP*. Se utilizarán las librerías desarrolladas en la Práctica 1.

Parte 1: 7-segmentos

Se harán las siguientes funciones para el manejo de los 7-segmentos:

- `void sieteSeg_init()`: realiza las inicializaciones necesarias.
- `void sieteSeg_digitos(uint8_t*)`: recibirá puntero a array de al menos 4 bytes. A partir de ese momento se mostrará en cada 7-segmentos valor correspondiente a las primeras 4 posiciones del array pasado.
- `void sietesSeg_valor(uint16_t)`: recibirá entero y hará que se muestre su valor en decimal en los 7-segmentos.

Se crearan los ficheros `sieteseg.h` y `sieteseg.c` y un fichero `prueba_sieteseg.c` con un programa principal que pruebe adecuadamente estas funciones.

Parte 2: Potenciometro y 7-segmentos

Hacer un programa que lea el valor de uno de los potenciómetros y represente en los 7- segmentos el valor de la conversión con una resolución de 10 bits.

Parte 3: Teclado numérico

Se harán las siguientes funciones para el manejo del teclado numérico:

- `void teclado_init()`: realiza las inicializaciones necesarias.
- `char teclado_getch():` espera a que se realice una pulsación en el teclado y devuelve código ASCII de la teclas pulsada.
- `char teclado_getch_timeout(uint32_t milis):` espera a que se realice una pulsación en el teclado y devuelve código ASCII de la tecla pulsada. Si no hay pulsación en `milis` milisegundos devuelve el carácter `'T'`.

Se crearan los ficheros `teclado.h` y `teclado.c` y un fichero `prueba_teclado.c` con un programa principal que pruebe adecuadamente estas funciones.

Parte 4: Teclado, 7-segmentos y motor

Hacer un programa que lea por teclado numérico un número de 0-100, lo represente en los 7-segmentos (a medida que el usuario lo va introduciendo) y haga que el motor gire a una velocidad proporcional al número introducido. Usar las teclas * y # para confirmar y cancelar la introducción del número, respectivamente.

Guía de estilo

Al escribir el código fuente se debe de respetar las siguientes reglas:

- Indicar autor y fecha en las primeras líneas del fichero fuente.
- Usar dos espacios para cada nivel de sangrado, utilizando caracteres espacio.
- En cada línea no deberá haber más de una instrucción; aunque una instrucción puede ocupar varias líneas, en ese caso las líneas de continuación deberán tener dos niveles de sagrado más que la inicial (4 espacios más).
- Dejar un espacio antes y después de cada operador, en especial de los de asignación (=, +=,

-=, …) y comparación (>, <, ==, …).

- Las líneas deberán tener, preferiblemente, menos de 72 caracteres y nunca superar los 80.
- Ya que se compila con el estándar C99, declarar las variables justo antes de ser utilizadas y, si es necesario, hacer la declaración y la inicialización en la misma instrucción.
- No declarar más de una variable por línea.
- Se utilizarán comentarios Doxygen para documentar el módulo y las funciones.