TP N°7: ENTRADAS /SALIDAS AVANZADAS

DRIVERS

Ejercicio 1.7

Realice una función que devuelva el código generado por teclado teclado matricial **4 x 1** de la figura sin, tener en cuenta la eliminación del rebote.

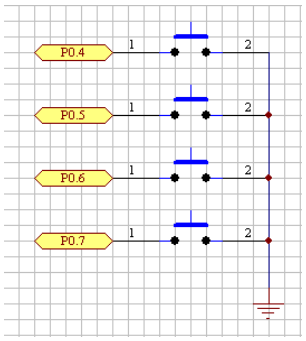


Figura 1

// Prototipo
unsigned char TecladoHW(void);

Ejercicio 2.7

Realice una función que devuelva el código generado por el teclado matricial **4 x 2** de la figura, sin tener en cuenta la eliminación del rebote.

// Prototipo
unsigned char TecladoHW(void);

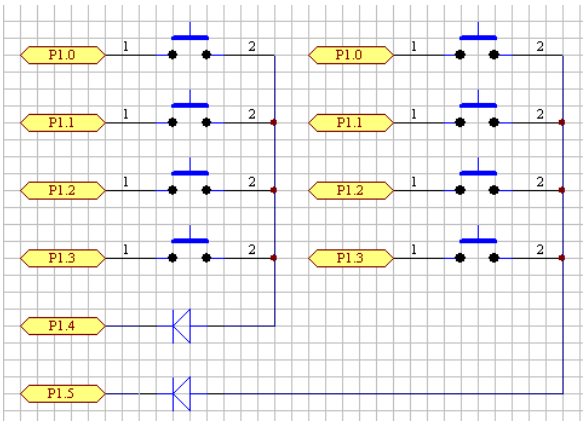


Figura 2

Ejercicio 3.7

Realice una función que devuelva el código generado por el teclado matricial **3 x 3** de la figura, sin tener en cuenta la eliminación del rebote.

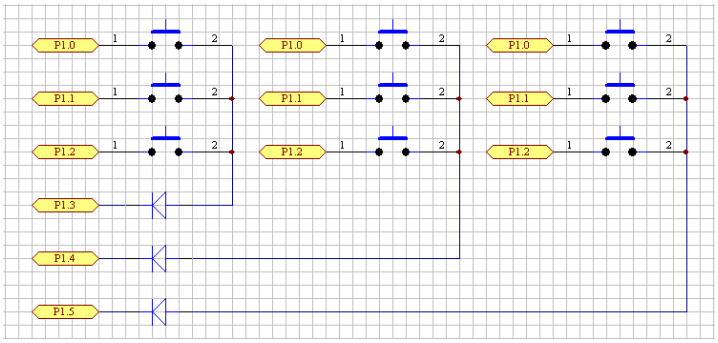


Figura 3

// Prototipo
unsigned char LecturaTecla(void);

Ejercicio 4.7

Realice una función portable que reciba el código generado por cualquiera de las realizadas en los **ejercicios 1.7 ,2.7 y 3.7** (la cual será invocada desde una interrupción de timer), capaz de eliminar su rebote y dejar el código filtrado en un buffer global.

// Buffer Global
unsigned char key;
// Prototipo
void CodigoTecla(unsigned char);

1) Aceptando el código si se repite 20 ms después de ser detectado
2) Aceptando el código si se repite durante cuatro lecturas consecutivas cada 5 ms después de ser detectado

Circuito de display multiplexado

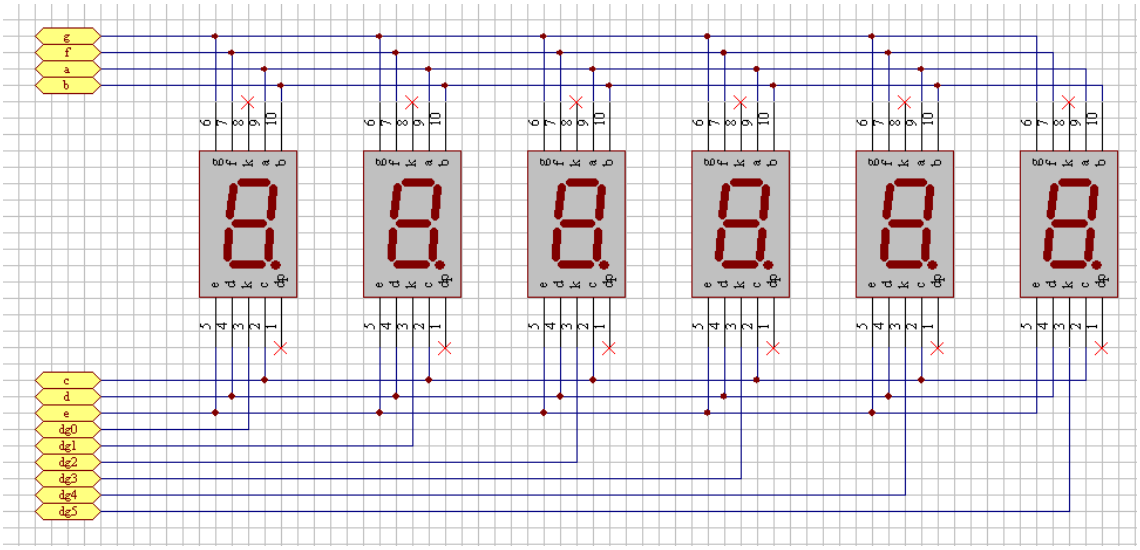


Figura 4

Ejercicio 5.7

Realice una función (la cual será invocada desde una interrupción de timer) que realice el multiplexado del display de la **figura 4** según la siguiente tabla tomando los datos a presentar desde un buffer global dimensionado apropiadamente.

// Prototipo
Void Mux7Seg(void);

// Buffer
unsigned char digitos[6];

Puerto	Función	Puerto	Función
P1.0	Habilitación dígito 0	P0.0	Segmento dp
P1.1	Habilitación dígito 1	P0.1	Segmento a
P1.2	Habilitación dígito 2	P0.2	Segmento b
P1.3	Habilitación dígito 3	P0.3	Segmento c
P1.4	Habilitación dígito 4	P0.4	Segmento d
P1.5	Habilitación dígito 5	P0.5	Segmento e
		P0.6	Segmento f
		P0.7	Segmento g

Ejercicio 6.7

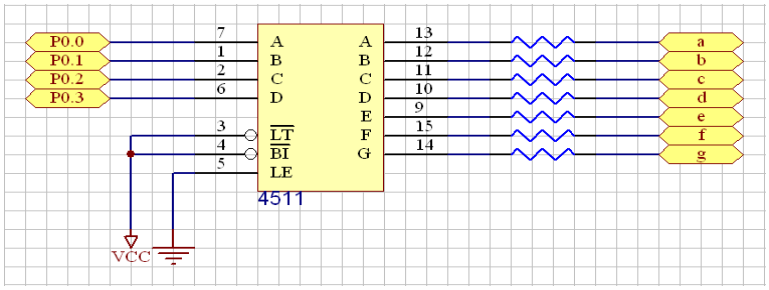
Realice una función (la cual será invocada desde una interrupción de timer) que realice el multiplexado del display de la **figura 4** acompañado de la Figura 5 según la siguiente tabla tomando los datos a presentar desde un buffer.
No considere para este ejercicio el dp. El CI 4511 es un decodificador BCD a 7 segmentos.

// Prototipo
Void Mux7Seg(void);

// Buffer
unsigned char digitos[6];

Puerto	Función
P0.0	Habilitación dígito 0
P0.1	Habilitación dígito 1
P0.2	Habilitación dígito 2
P0.3	Habilitación dígito 3
P0.4	Habilitación dígito 4
P0.5	Habilitación dígito 5
P0.6	BCD A
P3.7	BCD B
P3.7	BCD C
P3.9	BCD D

Figura 5



Ejercicio 7.7

Realice una función (la cual será invocada desde una interrupción de timer) que realice el multiplexado del display de la **figura 4** acompañado de la **figura 6** según la siguiente tabla tomando los datos a presentar desde el buffer ***unsigned char digitos[6]***

.

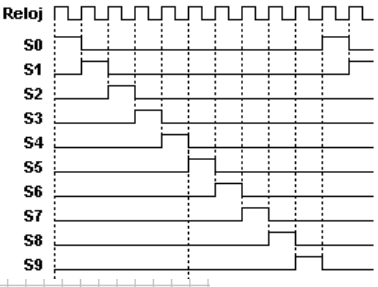
// Prototipo
Void Mux7Seg(void);

// Buffer
unsigned char digitos[6];

Puerto	Función
P0.0	CLK
P0.1	RESET
P0.2	Segmento dp
P0.3	Segmento a
P0.4	Segmento b
P0.5	Segmento c
P0.6	Segmento d
P0.7	Segmento e
P0.7	Segmento f
P0.9	Segmento g

El CI 4017 es un decodificador de 10 salidas. Con las entradas "CLKEN" y "RST" a tierra, el contador avanza una etapa a cada transición positiva de la señal de entrada (CLK).

El CI ULN2003 es un amplificador para las salidas.



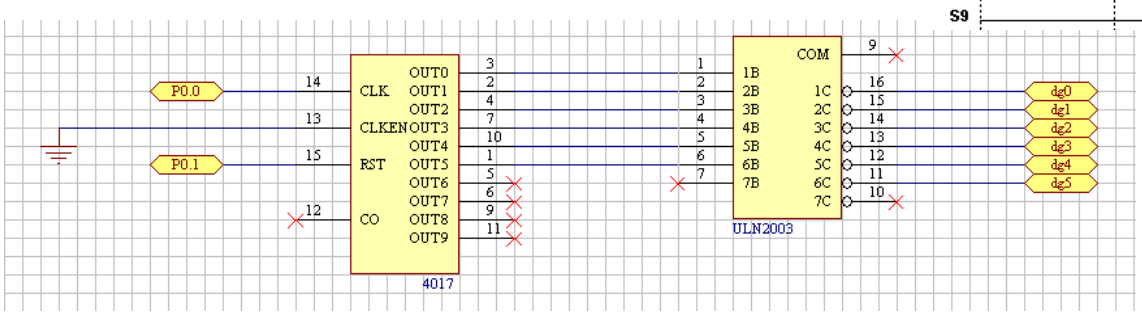


Figura 6

Circuito de display LCD

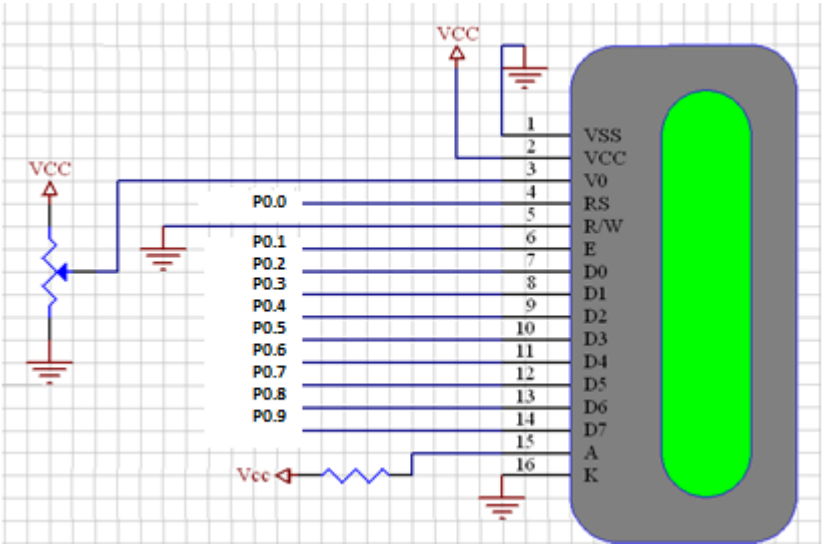


Figura 7

Ejercicio 8.7

Realice una función que inicialice el display de dos líneas por veinte caracteres de la **figura 7** para ser utilizado con ocho bits de datos **figura 8**. Considere para su desarrollo que esta función será llamada por única vez al iniciarse la ejecución de su programa.

// Prototipo
void InicLCD8d(void);

Ejercicio 9.7

Realice una función que inicialice el display de dos líneas por veinte caracteres de la **figura 7** para ser utilizado con cuatro bits de datos **figura 9**. Considere para su desarrollo que esta función será llamada por única vez al iniciarse la ejecución de su programa.

// Prototipo
void InicLCD4d(void);

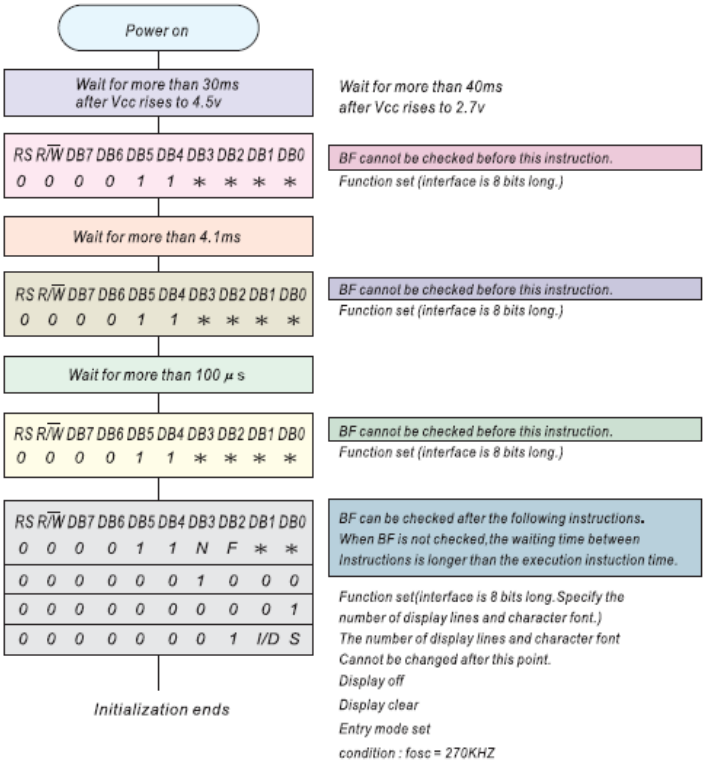


Figura 8

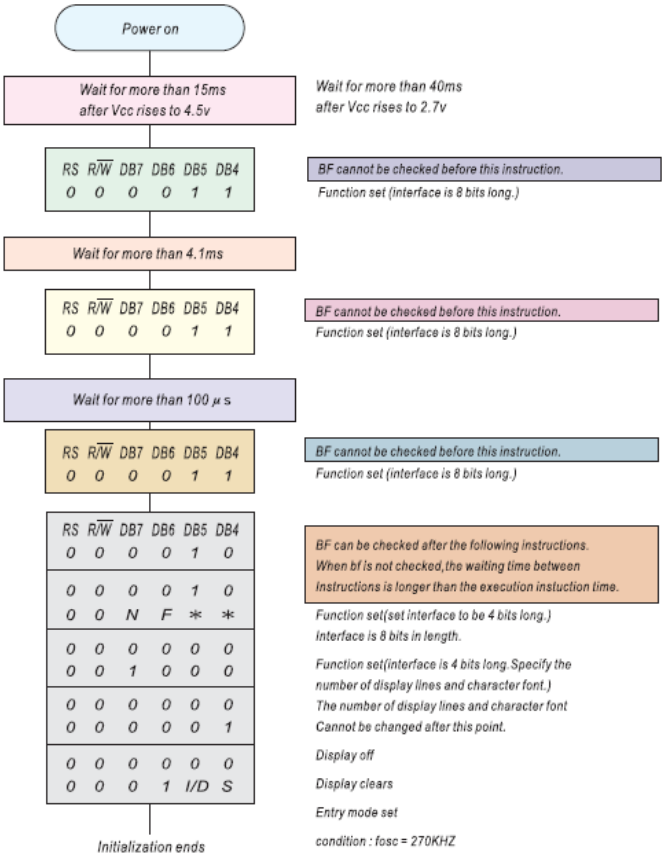


Figura 9

Ejercicio 10.7

Realice una función que elimine el ruido producido en la lectura de ocho entradas digitales conectadas a finales de carrera , dejando el valor filtrado en un buffer global.

// Buffer Global
unsigned char Entradas;

// Prototipo
void RebotеEntradas(unsigned char);

Command	Code										Description	Execution Time
	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0		
Clear Display	0	0	0	0	0	0	0	0	0	1	Clear the display and return the cursor to the home position (address 0).	82 μs~1.64ms
Return Home	0	0	0	0	0	0	0	0	1	*	Return the cursor to the home position(adress 0).also return a shifted display to the home position.DDRAM contents remain unchanged.	40 μs~1.6ms
Entry Mode Set	0	0	0	0	0	0	0	1	I/D	S	Set the cursor's move direction and enable/disable the display.	40 μs
Display On/off Control	0	0	0	0	0	0	1	D	C	B	Turn the display ON/OFF(D),or the cursor ON/OFF(C),and blink of the character at the cursor position(B).	40 μs
Cursor & Display Shift	0	0	0	0	0	1	S/C	R/L	*	*	Move the cursor and shift the display without changing the DD RAM contents.	40 μs
Function Set	0	0	0	0	1	DL	N	F	*	*	Set the data width(DL),the number of lines in display(L), and the character font(F).	40 μs
Set CG RAM Address	0	0	0	1	ACG						Set the CG RAM address.CG RAM data can be read or altered after making this setting.	40 μs
Set DD RAM Address	0	0	1	ADD							Set the DD RAM address.Data may be written or read after making this setting.	40 μs
Read Busy Flag & Address	0	1	BF	AC							Read the busy flag(BF)indicating that an internal operation is being performed and read the address counter contents.	1 μs
Write Data to CG or DD RAM	1	0	Write Data								Write data into DD RAM or CG RAM.	43 μs
Read Data from CG or DD RAM	1	1	Read Data								Read data from DD RAM or CG RAM.	43 μs
	I/D=1:Increment I/D=0:Decrement S=1:Accompanies Display Shift. S/C=1:Display Shift S/C=0:Cursor Move R/L=1:Shift To The Right. R/L=0:Shift To The Left. DL=1:8 Bits DL=0:4 Bits N=1:2 Lines N=0:1 Line F=1:5x10 Dots F=0:5x7 Dots BF=1:Busy BF=0:Can Accept Data										DD RAM:Display data RAM CG RAM:Character generator RAM ACG:CG RAM Address ADD:DD RAM Address Corresponds to cursor address. AC:Address counter Used for both DD and CG RAM address.	

PRIMITIVAS

Ejercicio 11.7

Realice una función portable que lea el buffer de teclado tal que retorne el código de tecla depositado en este o bien un código de no tecla en caso que se encuentre vacio.

// Codigo de no tecla
#define NO_KEY (unsigned char) 0xFF

// Buffer Global
unsigned char key;

// Prototipo
unsigned char LeerTecla(void);

Ejercicio 12.7

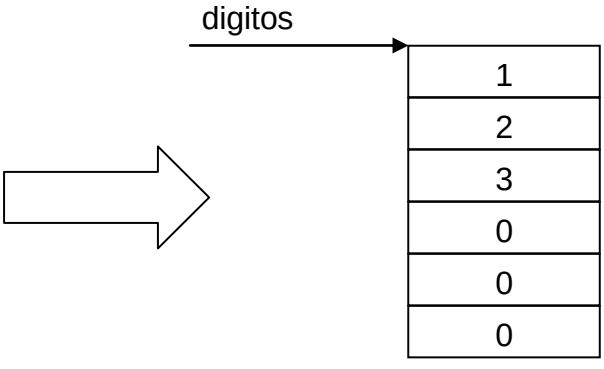
Realice una función portable que descomponga un número entero de 3 cifras según el teorema general de la numeración (TGN), es decir, unidades/decenas/centenas, y ubique cada cifra obtenida en un buffer (vector) de seis bytes, completando con ceros las posiciones del vector no utilizadas.

// Buffer Global
unsigned char digitos[6];

// Prototipo
void Display7seg (int val);

Ejemplo:

val = 123;



Ejercicio 13.7

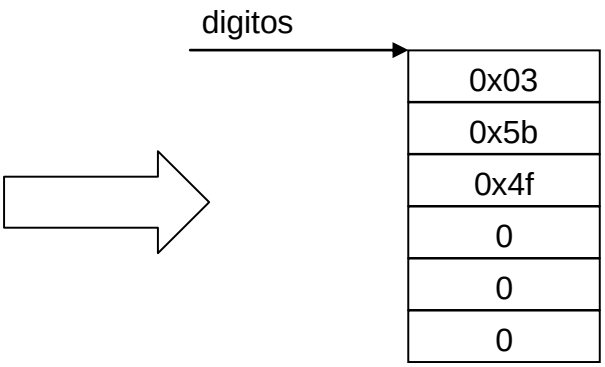
Realice una función portable que descomponga un número entero de 3 cifras según el mencionado TGN, y ubique sus correspondientes códigos siete segmentos en un buffer (vector) de seis bytes. Haga uso para la resolución del vector que realizó en el **Ejercicio 15.1**

// Buffer Global
unsigned char digitos[6];

// Prototipo
unsigned char Display7seg (int val);

Ejemplo:

val = 123;



Ejercicio 14.7

Realice una función portable que descomponga dos números enteros de 3 cifras según el mencionado TGN, y ubique sus correspondientes códigos siete segmentos en un buffer (vector) de seis bytes. Haga uso para la resolución del vector que realizó en el **Ejercicio 13.7**

// Buffer Global
unsigned char digitos[6];

// Prototipo
void Display7seg (int val1, int val2);

Ejercicio 15.7

Realice una función portable que reciba la dirección de comienzo de una string, el renglón y a partir de qué posición se desea imprimirla sobre el LCD de la **figura 7**

// Prototipo
void DisplayLCD (char msg , char renglon , char posicion);*

Ejercicio 16.7

Realice una función portable que reciba la dirección de comienzo de una string y el renglón donde se desea hacer ciclar un mensaje sobre el LCD de la **figura 7**
// Prototipo

void ShiftLCD(char msg , char renglon);*

APLICACION

Ejercicio 17.7

Utilizando las funciones resueltas en los ejercicios anteriores realice un programa que muestre en forma circular con una frecuencia de un segundo a un valor de 3 cifras en el display representado en el ejercicio 7.7

Figura 10

Ejercicio 18.7

Utilizando las funciones resueltas en los ejercicios anteriores realice un programa que haga parpadear cada un segundo a un valor de 3 cifras en el display representado en el ejercicio 6.7

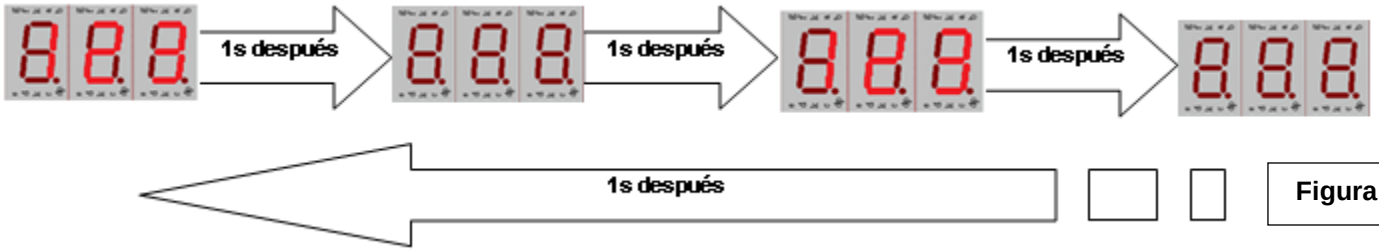


Figura 11

Ejercicio 19.7

Utilizando en lo posible las funciones resueltas en los ejercicios anteriores realice un programa que incremente la cantidad indicada en el display por cada vez que se pulse la tecla A y decremente por cada vez que se pulse la tecla B. Considere números negativos.

Figura 12

Ejercicio 20.7

Utilizando en lo posible las funciones resueltas en los ejercicios anteriores realice un programa que escriba en el display un número ingresado desde un teclado de 3 x 4 con la siguiente configuración.

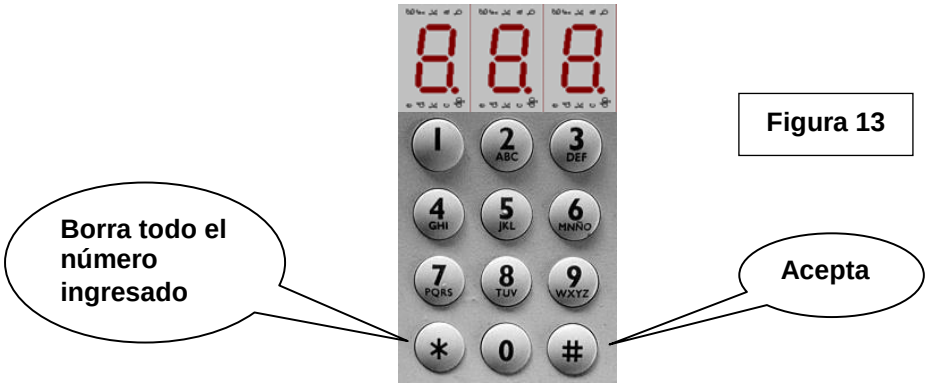


Figura 13

Una vez que el número es aceptado almacénelo en un buffer circular de 20 bytes.

Ejercicio 21.7

Utilizando en lo posible las funciones resueltas en los ejercicios anteriores realice un programa que incremente cuatro variables y muestre su contenido en diferentes zonas del LCD de la **figura 14**

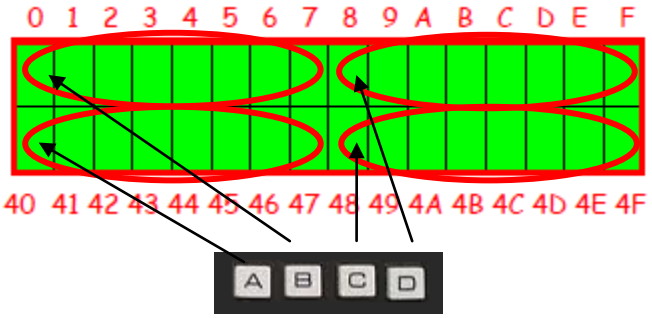


Figura 14

Ejercicio 22.7

Utilizando en lo posible las funciones resueltas en los ejercicios anteriores realice un programa que permita el ingreso de strings de 10 caracteres en un buffer circular (matriz) de 20 elementos desde un teclado de 4 x 1 de la siguiente forma.

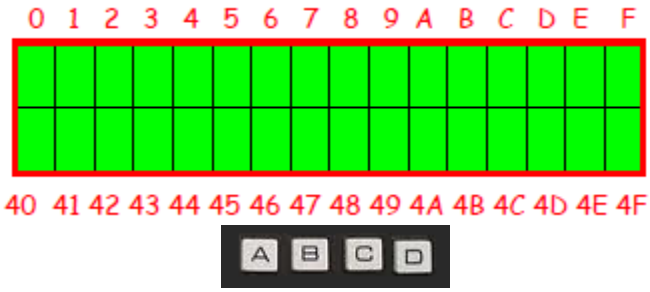


Figura 16

Pasa de un carácter visible a otro en forma cíclica
Avanza una posición en el display
Retrocede una posición en el display
Acepta la strig y la guarda en un bugffer circular (matriz)