Ana Cristina Ballesteros Herrera
A01637832

Evidencia 2. Diseño y descripción de un circuito que utilice memoria(s).

Contexto

Mediante la tarjeta FPGA DE10-LITE y un display de 16x2 se desea llevar a cabo la visualización de cinco mensajes, con la programación a través del software Quartus. Y también que con el uso de un potenciómetro se pueda modificar la intensidad del brillo para el mensaje proyectado.
Los mensajes proyectados se pueden visualizar mediante los switches que vienen en la tarjeta.

Análisis y proceso

Primeramente se realizó el código para poder definir a la tarjeta lo que debía hacer. Para esto, se definieron las librerías, entradas y salidas. En este caso, se utilizaron cinco switches, para proyectar cinco mensajes diferentes en el display. Un CLK, para que los cambios en el display se efectuaran cuando el CLK cambiara de forma ascendente, cabe mencionar que el CLK viene integrado con la tarjeta FPGA. Y también se definieron las salidas “rw”, que es read o write, “en” de enable, “rs” que define si la expresión en el arreglo es un comando o un dato y “data” que son los datos que se proyectarán en el display. A continuación se puede observar lo mencionado anteriormente.

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;
use IEEE.STD_LOGIC_ARITH.all;
use IEEE.STD_LOGIC_UNSIGNED.all;

entity Evidencia2 is
port (clk: in std_logic;                          --clock i/p
      sw1: in std_logic;
      sw2: in std_logic;
      sw3: in std_logic;
      sw4: in std_logic;
      sw5: in std_logic;
      rw: out std_logic;                         --read & write control
      en: out std_logic;                         --enable control
      rs: out std_logic;                         --data or command control
      data: out std_logic_vector(7 downto 0));     --data line
end Evidencia2;
```

Posteriormente, se inicia la arquitectura en donde se define lo que se desea desplegar en el display y los comandos. Esto se hizo a través de arreglos, en donde primeramente se define el número total de elementos dentro de estos, que en este caso fue 39.
Y todos los comandos se obtuvieron de la tablas presentadas:

CONTROL Y DATO	SEÑAL DE CONTROL		DATO / DIRECCIÓN								DECRIPCIÓN	TIEMPO DE EJEC
INSTRUCCIÓN	RS	RW	D7	D6	D5	D4	D3	D2	D1	D0	DESCRIPCION	
Borrar pantalla	0	0	0	0	0	0	0	0	0	1	Limpia todo el display y retorna el cursor a la posición de inicio, dirección 0	
Cursor a casa	0	0	0	0	0	0	0	0	1	-	Retorna el cursor a la posición inicio (dirección 0). También retorna el display, desplazando a la posición original. Los contenidos de la DDRAM permanecen sin cambios.	
Seleccionar modo	0	0	0	0	0	0	0	1	I/D	S	Configura la dirección de movimiento y si se desplaza o no el display. Esta operación es realizada durante operaciones de lectura escritura.	
Encender/ apagar pantalla	0	0	0	0	0	0	1	D	C	B	Configura el estado ON/OFF de todo el display (D), el cursor (C) y el parpadeo del caracter en la posición del cursor.	
Desplazar Cursor / Pantalla	0	0	0	0	0	1	S/C	R/L	-	-	Mueve el cursor y desplaza el display sin cambiar los contenidos de la DDRAM.	
Activar función	0	0	0	0	1	D/L	N	F	-	-	Configura el tamaño de la interfase (DL), el número de líneas del display (N) y la fuente del carácter (F) . N=0 es 1 línea. N=1 es 2 líneas.	
CG RAM	0	0	0	1	Dirección generador de RAM						Ajusta la dirección del generador de caracteres. El dato CG RAM es enviado y recibido después de este ajuste.	
DD RAM	0	0	1	Dirección de datos RAM							Ajusta la dirección de la DDRAM. La dirección es enviado y recibido después de este ajuste.	
Bandera de ocupado	0	0	BF	AC							Lectura de la bandera Busy Flag. Indicando que operaciones internas son realizadas y lectura de los contenidos del contador de direcciones.	
Escritura CG RAM/DD RAM	1	0	Escritura de dato								Escribe datos en la DDRAM o en la CGRAM	
Lectura CGRAM/ DDRAM	1	1	Lectura de dato								Lectura de datos desde la DDRAM o la CGRAM	

(Bolaños, s.f).

SIGNIFICADO DE LAS ABREVIATURAS	
I/D	= 1 incremental = 0 decrementa
S	= 1 desplaza el mensaje en la pantalla = 0 mensaje fijo en la pantalla
D	= 1 encender (activar) la pantalla = 0 apagar la pantalla (desactivar)
C	= 1 activar cursor = 0 desactivar cursor
B	= 1 parpadea caracter señalado por el cursor = 0 no parpadea el caracter
S/C	= 1 desplaza pantalla = 0 mueve el cursor
RL	= 1 desplazamiento a la derecha = 0 desplazamiento a la izquierda
DL	= 1 datos de 8 bits = 0 datos de 4 bits
BF	= 1 durante operación interna del módulo = 0 finaliza la operación interna

(Bolaños, s.f).

Y los comandos que se utilizaron fueron los siguientes,

Comando (hex)	Comando (binario)	Descripción	Abreviaturas elegidas

X ”38”	0011 1000	Activar función	DL = 1 (datos de 8 bits) N = 1 (dos líneas) F = 1 (fuente del carácter)
X ”0c”	0000 1100	Encender/Apagar pantalla	D = 1 (Estado ON del display) C = 0 (desactivar el cursor) B = 0 (no parpadea el carácter)
X ”06”	0000 0110	Seleccionar modo	I/D = 1 (incremental) S = 0 (mensaje fijo)
X ”01”	0000 0001	Borrar pantalla	Limpiar todo el display y retorna el cursor a la posición de inicio.
X ”80”	1000 0000	Dirección de datos RAM	Ajusta la dirección de la DDRAM.

Y todos los caracteres se obtuvieron mediante la siguiente tabla:

(N.A, 2017).

En este caso se seleccionaron los mismos comandos para los cinco mensajes que se quisieron proyectar, que se colocaron en las primeras cinco posiciones y en la posición 22 de cada arreglo. Los elementos restantes fueron los carácteres, que en este caso se decidió desplegar los nombres de las integrantes del equipo. También, el tipo de arreglo va de 0 a 7 ya que cada comando/carácter se conforma por ocho bits, y se representa en formato hexadecimal.

```
architecture funcionamiento of Evidencia2 is
 constant N: integer := 39;
 type arr is array (1 to N) of std_logic_vector(7 downto 0);
 constant info1: arr :=
(X"38",X"0c",X"06",X"01",X"80",X"41",x"6E",x"61",x"20",x"20",x"20",x"20",x"20",x"20",x"20",x"20"
,x"20",x"20",x"20",x"20",x"C0",x"43",x"72",x"69",x"73",x"20",x"20",x"20",x"20",x"20",x"20",x"20"
,x"20",x"20",x"20",x"20",X"53"); --command and data to display
```

```
 constant info2: arr :=
(X"38",X"0c",X"06",X"01",X"80",X"41",x"6E",x"61",x"20",x"20",x"20",x"20",x"20",x"20",x"20",x"20",x"20"
,x"20",x"20",x"20",x"20",x"C0",x"50",x"61",x"75",x"20",x"20",x"20",x"20",x"20",x"20",x"20",x"20",x"20"
,x"20",x"20",x"20",x"20",X"53");
 constant info3: arr :=
(X"38",X"0c",X"06",X"01",X"80",X"4D",x"61",x"72",x"69",x"73",x"20",x"20",x"20",x"20",x"20",x"20",x"20"
,x"20",x"20",x"20",x"20",x"C0",x"20",x"20",x"20",x"20",x"20",x"20",x"20",x"20",x"20",x"20",x"20",x"20"
,x"20",x"20",x"20",x"20",X"53");
 constant info4: arr :=
(X"38",X"0c",X"06",X"01",X"80",X"4D",x"6F",x"6E",x"74",x"73",x"65",x"20",x"20",x"20",x"20",x"20",x"20"
,x"20",x"20",x"20",x"20",x"C0",x"20",x"20",x"20",x"20",x"20",x"20",x"20",x"20",x"20",x"20",x"20",x"20"
,x"20",x"20",x"20",x"20",X"53");
 constant info5: arr :=
(X"38",X"0c",X"06",X"01",X"80",X"52",x"61",x"71",x"75",x"65",x"6C",x"20",x"20",x"20",x"20",x"20",x"20"
,x"20",x"20",x"20",x"20",x"C0",x"20",x"20",x"20",x"20",x"20",x"20",x"20",x"20",x"20",x"20",x"20",x"20"
,x"20",x"20",x"20",x"20",X"53");
 constant NA: arr :=
(X"38",X"0c",X"06",X"01",X"80",X"53",X"65",X"6C",X"65",X"63",x"63",x"69",x"6F",x"6E",x"61",x"72",x"20"
,x"20",x"20",x"20",x"20",x"C0",x"20",x"20",x"20",x"20",x"20",x"20",x"20",x"20",x"20",x"20",x"20",x"20"
,x"20",x"20",x"20",x"20",X"53");
```

Después se definieron algunas variables, que fueron “i” e “j”, que se utilizarán más adelante, y se inicializó “rs”.

```
begin
rw <= '0';  --lcd write

process(clk)
 variable info: arr;
 variable i: integer := 0;
 variable j: integer := 1;
```

Cuando se comienza con el procedimiento, se define la información que se quiere proyectar en cada caso tomando en cuenta los switches. Y si la situación no entra en ninguno de los casos se proyectará “Seleccionar”.

```
 begin
 if sw1 = '1' and sw2 = '0' and sw3 = '0' and sw4 = '0' and sw5 = '0' then
  info := info1;
 elsif sw2 = '1' and sw1 = '0' and sw3 = '0' and sw4 = '0' and sw5 = '0' then
  info := info2;
 elsif sw3 = '1' and sw2 = '0' and sw1 = '0' and sw4 = '0' and sw5 = '0' then
  info := info3;
 elsif sw4 = '1' and sw2 = '0' and sw3 = '0' and sw1 = '0' and sw5 = '0' then
  info := info4;
 elsif sw5 = '1' and sw2 = '0' and sw3 = '0' and sw4 = '0' and sw1 = '0' then
  info := info5;
 else
  info := NA;
 end if;
```

Cuando se tienen definidos los casos mediante los switches de la tarjeta, se definirá lo que hará el CLK cuando cambia de manera ascendente. En este se define primeramente la condición, y se hace un tipo de contador, para que cada letra aparezca cada cierto tiempo después de la anterior. Como se puede observar en la parte inferior, en el código se define una condición con la variable “i” que en este caso será un contador, que de cierta manera funcionará como un delay. Ya que al definir que “i” va de 0 a 10,000 y que enable sea 1

cuando i es menor igual a 5,000, para escribir entre letra y letra, se debe esperar a que el contador de 5,000 llegue a 10,000, para así reiniciarse y comenzar a contar nuevamente.

```
if clk'event and clk = '1' then
 if i <= 5000 then
       i := i + 1;
       en <= '1';
       data <= info(j)(7 downto 0);
 elsif i > 5000 and i < 10000 then
      i := i + 1;
      en <= '0';
 elsif i = 10000 then
   j := j + 1;
   i := 0;
 end if;
```

Y por otra parte, como se mencionó anteriormente, los primeros cinco elementos y el elemento 22 del arreglo son para comandos, por lo tanto, en estos “rs” debe ser igual a 0, y en los demás debe ser 1. Por lo tanto se hace lo siguiente en el código, y cuando “j” llegue al número máximo de elementos en un arreglo, en este caso 39, se reinicia al número inicial de carácteres.

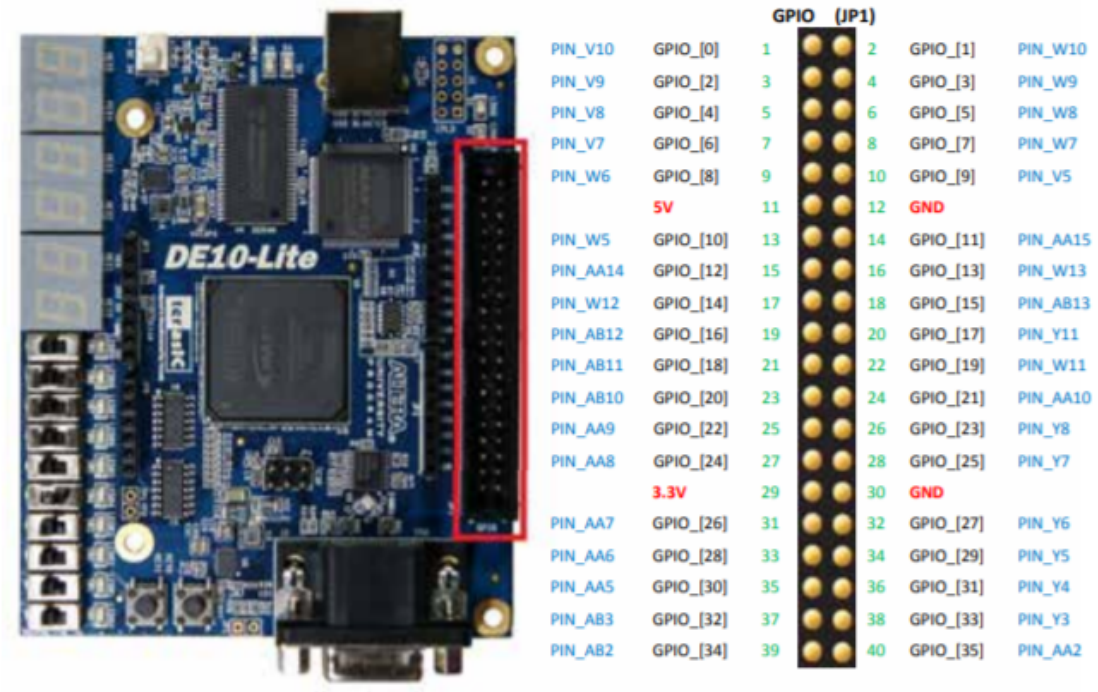
```
if j <= 5 then
 rs <= '0';    --command signal
     elsif j = 22 then
      rs <= '0';
else
 rs <= '1';   --data signal
end if;

if j = 39 then  --repeated display of data
 j := 5;
```

Desarrollo en tarjeta FPGA

Al realizar el código para introducirlo en Quarts, se realizaron las conexiones de los pines para la tarjeta FPGA. Los cuáles se acomodaron de la siguiente manera.

Node Name	Direction	Location	I/O Bank	VREF Group	Fitter Location	I/O Standard	Reserved	Current Strength	Slew Rate	Differential Pair	Strict Preservation
clk	Input	PIN_P11	3	B3_N0	PIN_P11	2.5 V		12mA (default)			
data[7]	Output	PIN_AB10	4	B4_N0	PIN_AB10	2.5 V		12mA (default)	2 (default)		
data[6]	Output	PIN_AB11	4	B4_N0	PIN_AB11	2.5 V		12mA (default)	2 (default)		
data[5]	Output	PIN_AB12	4	B4_N0	PIN_AB12	2.5 V		12mA (default)	2 (default)		
data[4]	Output	PIN_W12	4	B4_N0	PIN_W12	2.5 V		12mA (default)	2 (default)		
data[3]	Output	PIN_AA14	4	B4_N0	PIN_AA14	2.5 V		12mA (default)	2 (default)		
data[2]	Output	PIN_W5	3	B3_N0	PIN_W5	2.5 V		12mA (default)	2 (default)		
data[1]	Output	PIN_W6	3	B3_N0	PIN_W6	2.5 V		12mA (default)	2 (default)		
data[0]	Output	PIN_V7	3	B3_N0	PIN_V7	2.5 V		12mA (default)	2 (default)		
en	Output	PIN_V8	3	B3_N0	PIN_V8	2.5 V		12mA (default)	2 (default)		
rs	Output	PIN_V10	3	B3_N0	PIN_V10	2.5 V		12mA (default)	2 (default)		
rw	Output	PIN_V9	3	B3_N0	PIN_V9	2.5 V		12mA (default)	2 (default)		
sw1	Input	PIN_C10	7	B7_N0	PIN_C10	2.5 V		12mA (default)			
sw2	Input	PIN_C11	7	B7_N0	PIN_C11	2.5 V		12mA (default)			
sw3	Input	PIN_D12	7	B7_N0	PIN_D12	2.5 V		12mA (default)			
sw4	Input	PIN_C12	7	B7_N0	PIN_C12	2.5 V		12mA (default)			
sw5	Input	PIN_A12	7	B7_N0	PIN_A12	2.5 V		12mA (default)			

Cabe mencionar que se tomó en cuenta el siguiente diagrama que se tomó de las especificaciones de la tarjeta. También, el display se alimentó con un voltaje de 3.3 V, proveniente de la tarjeta.



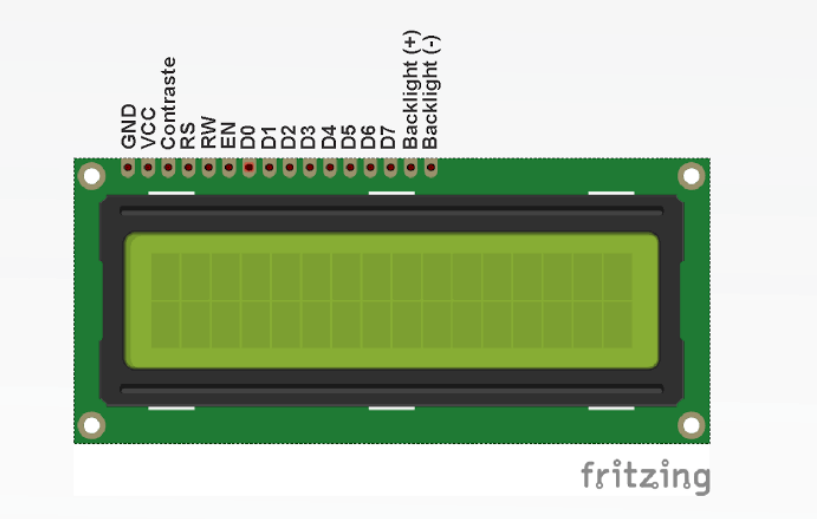
(Terasic, 2016)

Para los switches se tomaron los pines específicos para esta función, que también se obtuvieron de las especificaciones de la tarjeta, como se puede observar en la siguiente tabla.

Signal Name	FPGA Pin No.	Description	I/O Standard
SW0	PIN_C10	Slide Switch[0]	3.3-V LVTTL
SW1	PIN_C11	Slide Switch[1]	3.3-V LVTTL
SW2	PIN_D12	Slide Switch[2]	3.3-V LVTTL
SW3	PIN_C12	Slide Switch[3]	3.3-V LVTTL
SW4	PIN_A12	Slide Switch[4]	3.3-V LVTTL
SW5	PIN_B12	Slide Switch[5]	3.3-V LVTTL
SW6	PIN_A13	Slide Switch[6]	3.3-V LVTTL
SW7	PIN_A14	Slide Switch[7]	3.3-V LVTTL
SW8	PIN_B14	Slide Switch[8]	3.3-V LVTTL
SW9	PIN_F15	Slide Switch[9]	3.3-V LVTTL

(Terasic, 2016)

Por otro lado, para poder conectar el display de la manera correcta, se tomó de base los siguientes pines:



(Irineo, 2015).

Y la conexión de cada pin de la LCD a la tarjeta se puede ver en el pin planner, presentado anteriormente. El que no se conectó a la tarjeta directamente fue el contraste, porque este fue conectado a la salida del potenciómetro, para que así, modificara la intensidad del brillo.

Simulaciones

Teniendo todo esto en cuenta, se comenzó a conectar el display con la tarjeta, con los pines definidos. Cabe mencionar que se conectó un potenciómetro y cuando este se giraba se iba modificando la intensidad del display. Al unir todo estos elementos se obtuvieron los siguientes resultados, que se encuentran en la siguiente carpeta, junto con el código.
Evidencia 2

Conclusión

El primer acercamiento con la LCD fue en esta evidencia. Gracias a esto, se pudo comprender cómo funciona la tarjeta, y cómo poder programarla en VHDL. Para esto, se tuvieron que comprender los comandos de esta y como poder activarlos, los cuales se encontraban en las especificaciones del LCD.
Con esta evidencia también se pudo utilizar la tarjeta FPGA, para que así se pueda entender mejor con cada acercamiento.

Los resultados sí fueron los esperados, ya que se pudieron imprimir en total 6 mensajes, siendo: Ana Cris, Ana Pau, Maris, Montse, Raquel y Seleccionar. Y esto se realizaba a través del cambio en los switches.

Referencias:

Bolaños, D (s.f). MANEJO DE DISPLAY LCD (Versión 1.1). TEORIALCDV1. Recuperado de. https://www.bolanosdj.com.ar/SOBRELCD/TEORIALCDV1.pdf

Irineo, R (2015). Librería LCD con Phyton. botboss. Recuperado de. https://www.google.com.mx/amp/s/botboss.wordpress.com/2015/07/16/libreria-lcd-con-python/amp/

N.A (2017). LCD controller and User Logic in VHDL and Programming a FPGAs. LCD_Display. Recuperado de. https://openlab.citytech.cuny.edu/wang-cet4805/files/2017/04/LCD-controller-and-User-Logic-in-VHDL-and-Programming-a-FPGAs_posted.pdf

Terasic (2016). DE10-LITE User Manual. Terasic. Recuperado de. https://www.intel.com/content/dam/www/programmable/us/en/portal/dsn/42/doc-us-dsnbk-42-2912030810549-de10-lite-user-manual.pdf