INGENIERÍA MECATRÓNICA

DI_CERO

DIEGO CERVANTES RODRÍGUEZ

ELECTRÓNICA DIGITAL: CIRCUITOS LÓGICOS, LENGUAJE VHDL Y VERILOG

XILINX (64-BIT PROJECT NAVIGATOR) & ADEPT

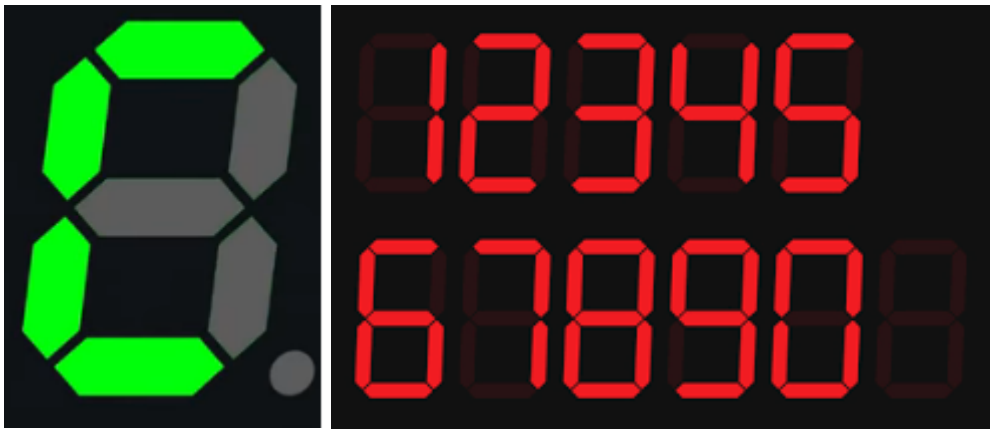
Display de 7 Segmentos

Contenido

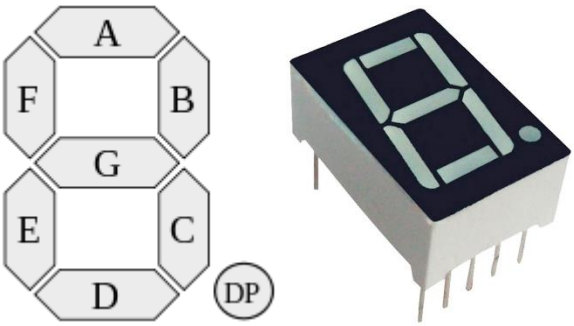
Display de 7 segmentos 2
Configuraciones de Ánodo y Cátodo Común 3
Displays de 7 Segmentos en la Placa de Desarrollo NEXYS 2 4
Código de Ejemplo en Verilog y VHDL: 7
Código Verilog: 7
Código UCF: 8
Código VHDL: 9
Código UCF: 10

Display de 7 segmentos

El nombre completo del display de 7 segmentos es decodificador BCD de 7 segmentos y son simplemente varios leds ordenados en una forma que muestren números del cero al nueve y un punto.

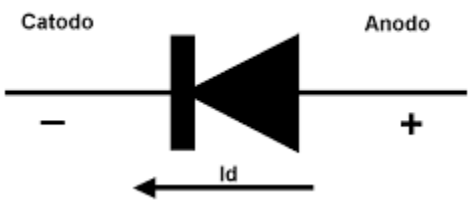


Cada uno de sus 7 segmentos tiene nombre y con segmento nos referimos a cada led que enciende una parte del display.



Se nombran con letras y su orden va en sentido horario, para mostrar el número 1 debemos prender el segmento B y C, para mostrar el número 2 debemos encender los segmentos A, B, G, E y D y así respectivamente para mostrar todos los demás números.

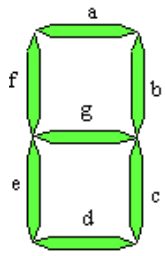
Dentro del display hay 7 leds, todos están conectados a un mismo punto y recordemos que solo puede conducir corriente y prenderse el led si el voltaje positivo entra en el cátodo y el negativo (osea tierra) está conectado al cátodo.

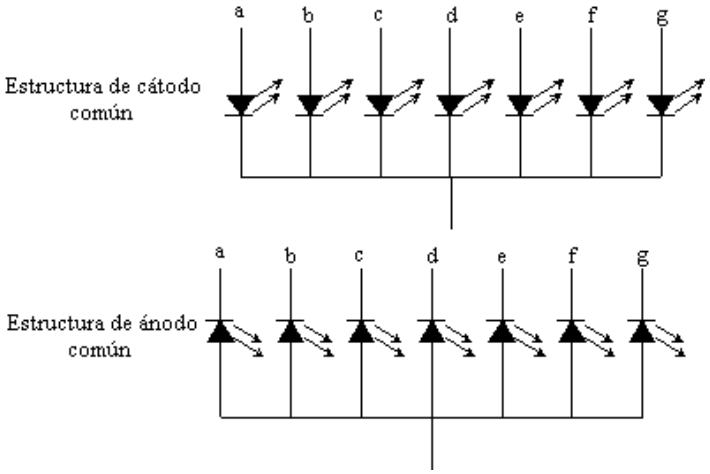


Configuraciones de Ánodo y Cátodo Común

Hay 2 configuraciones en los displays de 7 segmentos, una llamada ánodo común y otra llamada cátodo común.

- **En los displays de cátodo común, todos los cátodos están conectados a tierra:** Para prender los leds a, b, c, d, e, f, g o el punto debo meter alimentación de 5 o 3.3 Volts en cada patita del display.
- **En los displays de ánodo común, todos los ánodos están conectados a alimentación:** Para prender los leds a, b, c, d, e, f, g o el punto debo meter un voltaje de 0 Volts en cada patita del display.

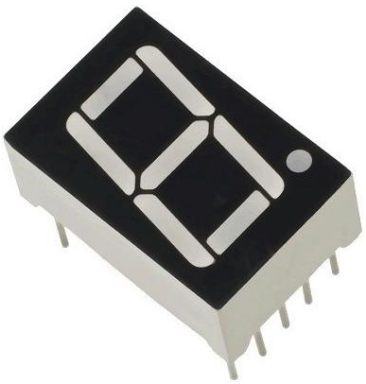


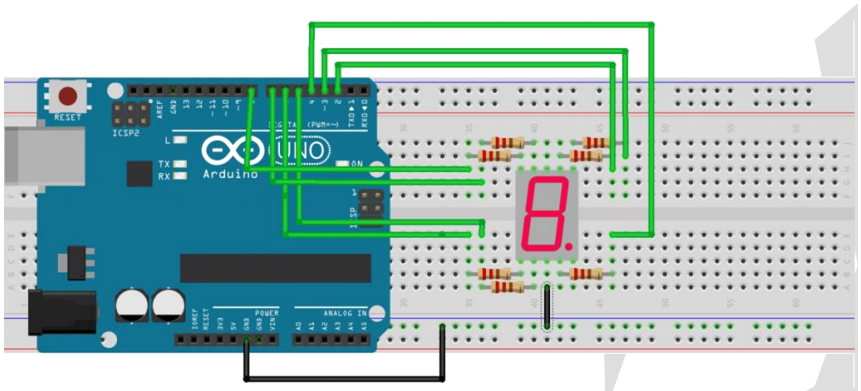


En electrónica digital esto significa que:

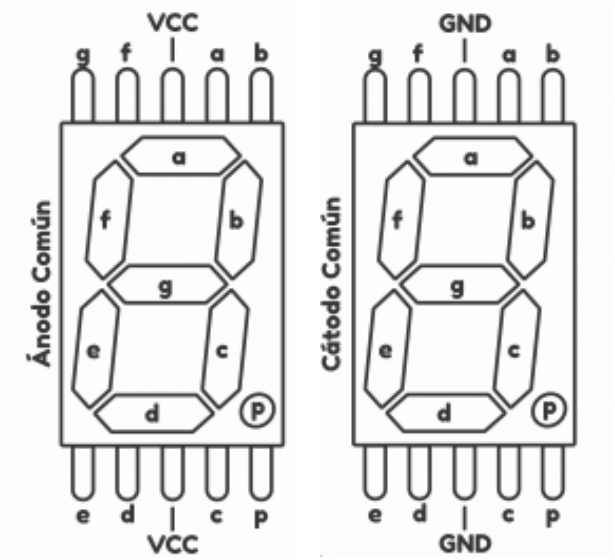
- Para prender un led en la configuración C.C. (**cátodo común**) debo mandar un **1 lógico** al display.
- Para prender un led en la configuración A.C. (**ánodo común**) debo mandar un **0 lógico** al display.

En los displays de 7 segmentos que estén incluidos en la NEXYS 2 no es necesario poner resistencias de protección porque ya las tiene incluidas la tarjeta de desarrollo, pero si uso un display de 7 segmentos externo como el siguiente, si es necesario que ponga una resistencia de protección ya que si los leds del display reciben 5 o 3.3 Volts directamente, hay posibilidad de que se quemen, ya que los leds solo aguantan aproximadamente 2 Volts, por lo que se debe reducir el voltaje por medio de una resistencia.



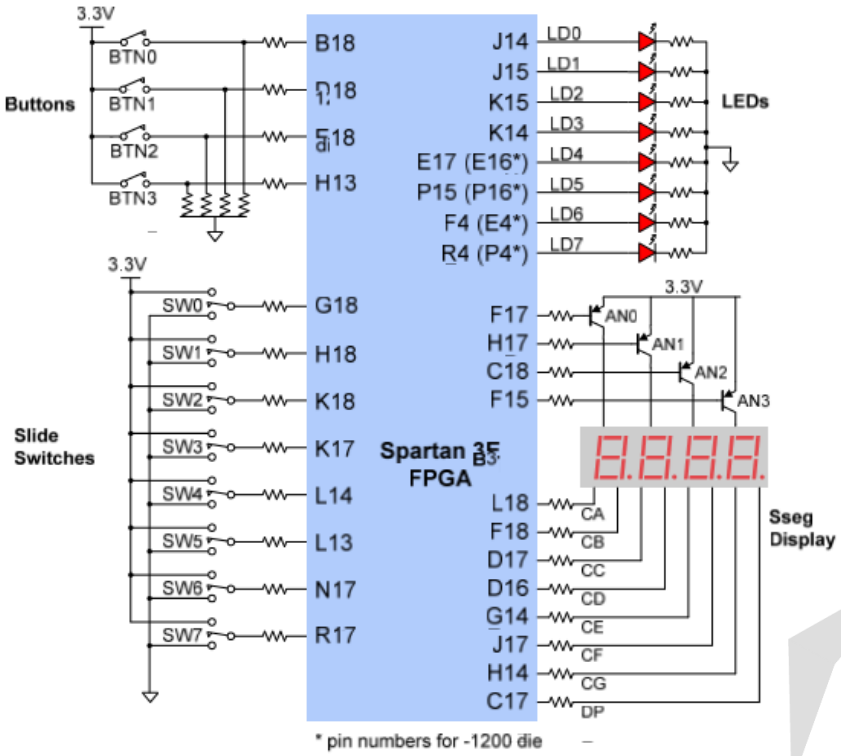


Para saber exactamente cuales leds corresponden a cada patita debo ver el datasheet del display, pero usualmente están acomodadas así:



Displays de 7 Segmentos en la Placa de Desarrollo NEXYS 2

En el manual de la NEXYS 2 se nos muestra que estos son los nombres y códigos para que enviemos nuestras salidas a los displays de 7 segmentos a través del archivo .ucf con el programa Adept:



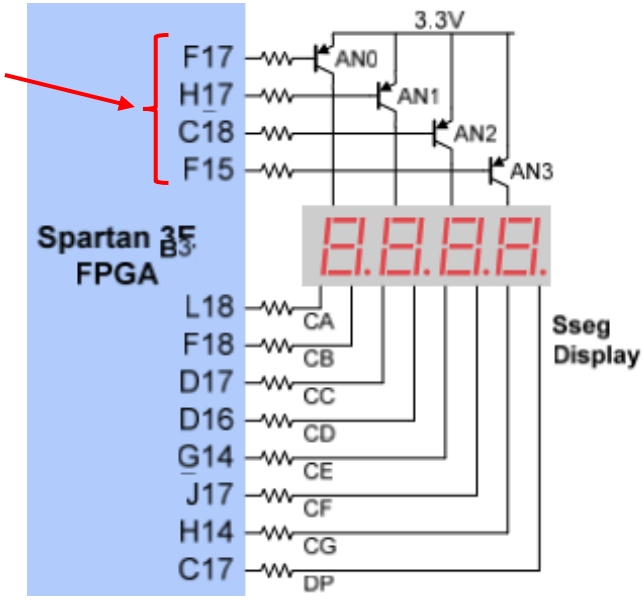
Las salidas para los **ánodos comunes** de los displays de 7 segmentos son de 1 bit:

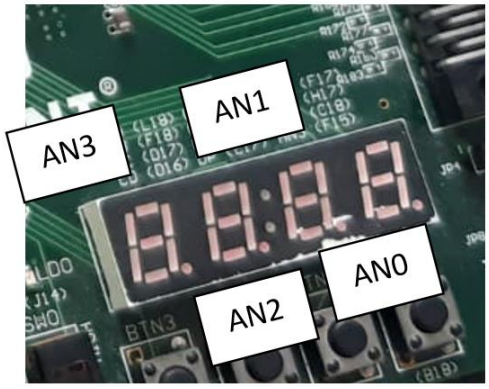
- ✓ **El 0 lógico prende cada uno de los displays.**
- ✓ **El 1 lógico apaga cada display.**

- Los nombres de los **ánodos comunes de los 4 displays de 7 segmentos** en la NEXYS 2 que se programan en el archivo .ucf para prender cada uno de los 4 displays de 7 segmentos disponibles en la placa de desarrollo son los siguientes:
  - **Salidas de 1 bit:**

AN0 (nombre) - **F17 (código).**
AN1 (nombre) - **H17 (código).**
AN2 (nombre) - **C18(código).**
AN3 (nombre) - **F15 (código).**

Otro punto importante de los 4 ánodos comunes de los displays de 7 segmentos es que, si creo un vector en el código y quiero prender en ese orden los diferentes displays de 7 segmentos, cuando asigne cada salida a los códigos de los displays en el archivo UCF, debo ponerlos en un orden que vaya del bit menos significativo al más significativo.
Ya que **AN3 es el 1er display, AN2 es el 2do, AN1 es el 3ro y AN0 es el último** si los veo todos de izquierda a derecha, aunque en el manual no lo parece.





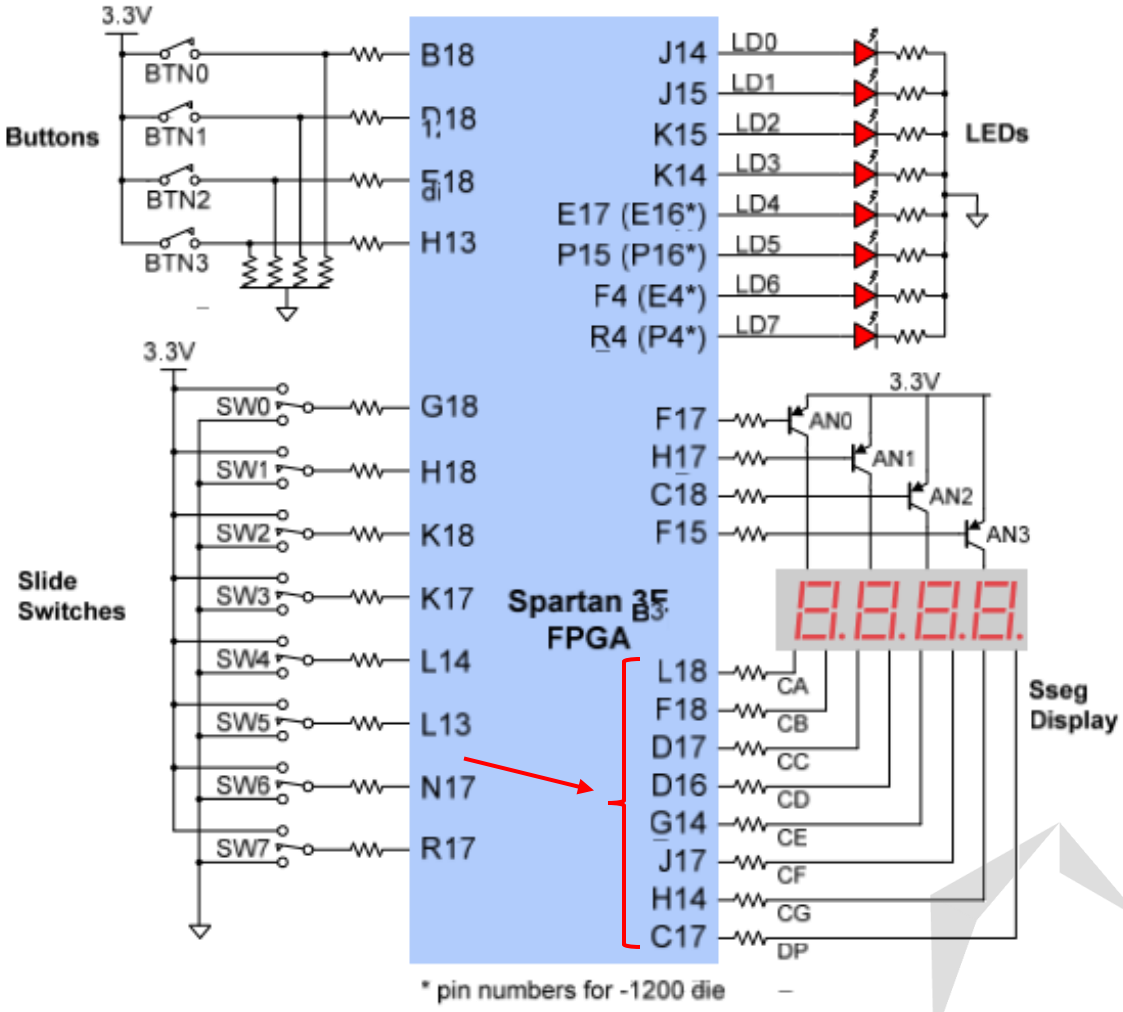
Las salidas para los diferentes leds de los displays de 7 segmentos son de 1 bit:

- ✓ **El 0 lógico prende cada led** A, B, C, D, E, F, G o DP.
- ✓ **El 1 lógico apaga cada led** A, B, C, D, E, F, G o DP.

Las salidas CA, CB, …, CG y DP son de 1 bit y prenden cada led A, B, C, D, E, F, G o los Puntos de todos los displays de la tarjeta, los 4 displays de la NEXYS 2 muestran siempre la misma figura o número.

- **Leds de los displays de 7 segmentos** en la NEXYS 2:
  - **Salidas de 1 bit:**

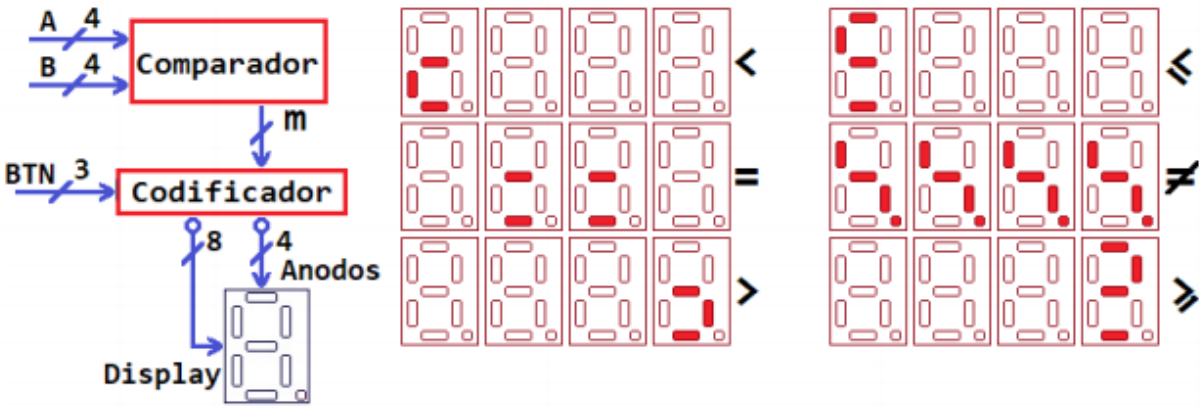
CA (nombre) - **L18 (código).**
CB (nombre) - **J15 (código).**
CC (nombre) - **K15 (código).**
CD (nombre) - **K14 (código).**
CE (nombre) - **E17 (código).**
CF (nombre) - **P15 (código).**
CG (nombre) - **F4 (código).**
DP (nombre) - **R4 (código).**



Código de Ejemplo en Verilog y VHDL:

Implementar un comparador de dos vectores A y B de 4 bits (cada bit del número binario debe estar conectado a interruptores y/o botones de la tarjeta de desarrollo NEXYS 2), las comparaciones serán:

- A > B, en VHDL o Verilog, **se realiza esta operación relacional con el signo >**
- A < B, en VHDL o Verilog, **se realiza esta operación relacional con el signo <**
- A = B, en VHDL o Verilog, **se realiza esta operación relacional con el signo ==**
- A ≥ B, en VHDL o Verilog, **se realiza esta operación relacional con el signo >=**
- A ≤ B, en VHDL o Verilog, **se realiza esta operación relacional con el signo <=**
- A ≠ B, en VHDL o Verilog, **se realiza esta operación relacional con el signo /=**



Como todavía no se está usando una señal de reloj en el programa, solamente se verá un display encendido a la vez.

Código Verilog:

```
//Ver el código del archivo demuxMuxComparador porque aquí se va a hacer una comparación.
module display7Segmentos(
    output reg [3:0] AN,
           //El vector AN sirve para elegir cuales de los 4 displays de 7 segmentos que tiene la
           //FPGA se van a activar. Los displays se prenden mandando un 0 lógico.
    output reg CA,
    output reg CB,
    output reg CC,
    output reg CD,
    output reg CE,
    output reg CF,
    output reg CG,//En verilog cuando use una salida en un condicional, lo debo declarar como reg
           //Estas salidas lo que van a hacer es prender los mismos leds en todos los displays activados,
           //osea mostrar el mismo número o letra.
    output reg DP,
           //Esto va a prender el led del punto de todos los displays activados.
           //Todos los leds de los displays se prenden mandando un 0 lógico porque son de nodo común.
    input [3:0] A,
    input [3:0] B
           //Estos son los vectores con números binarios (o bits) que vamos a comparar y van a ser
           //entradas por medio de switches.
    );

//Dentro del always (que sirve para poder usar condicionales) debo poner las entradas que vaya a usar.
always@(A or B)
          begin//always tiene su propio begin y end.
                    if(A>B)
                              begin//En los condicionales se pone begin y end cuando se vaya a dar valor a más de una salida.
                                        AN=4'b1110;//Solo el 4to display está encendido
                                        CA=1'b1;//Led A del display apagado
                                        CB=1'b1;//Led B del display apagado
                                        CC=1'b0;//Led C del display ENCENDIDO
                                        CD=1'b0;//Led D del display ENCENDIDO
                                        CE=1'b1;//Led E del display apagado
                                        CF=1'b1;//Led F del display apagado
                                        CG=1'b0;//Led G del display ENCENDIDO
```

```
                                        //Todos estos leds prendidos están representando un signo de mayor que >.
                                        DP=1;//El punto del display esta apagado.
                                end
                        else if(A<B)
                                begin
                                        AN=4'b0111;//Solo el 1er display está encendido
                                        CA=1'b1;//Led A del display apagado
                                        CB=1'b1;//Led B del display apagado
                                        CC=1'b1;//Led C del display apagado
                                        CD=1'b0;//Led D del display ENCENDIDO
                                        CE=1'b0;//Led E del display ENCENDIDO
                                        CF=1'b1;//Led F del display apagado
                                        CG=1'b0;//Led G del display ENCENDIDO
                                        //Todos estos leds prendidos están representando un signo de menor que <.
                                        DP=1'b1;//El punto del display esta apagado.
                                end
                        else if(A==B)
                                begin
                                        AN=4'b1001;//El 2do y 3er display están encendidos
                                        CA=1'b1;//Led A del display apagado
                                        CB=1'b1;//Led B del display apagado
                                        CC=1'b1;//Led C del display apagado
                                        CD=1'b0;//Led D del display ENCENDIDO
                                        CE=1'b1;//Led E del display apagado
                                        CF=1'b1;//Led F del display apagado
                                        CG=1'b0;//Led G del display ENCENDIDO
                                        //Todos estos leds prendidos están representando dos signos de igual ==.
                                        DP=1'b1;//El punto del display esta apagado.
                                end
                        else if(A>=B)
                                begin
                                        AN=4'b0111;//Solo el 1er display está encendido
                                        CA=1'b0;//Led A del display ENCENDIDO
                                        CB=1'b1;//Led B del display apagado
                                        CC=1'b1;//Led C del display apagado
                                        CD=1'b0;//Led D del display ENCENDIDO
                                        CE=1'b1;//Led E del display apagado
                                        CF=1'b0;//Led F del display ENCENDIDO
                                        CG=1'b0;//Led G del display ENCENDIDO
                                        //Todos estos leds prendidos están representando un signo de mayor o igual que >=.
                                        DP=1'b1;//El punto del display esta apagado.
                                end
                        else if(A!=B)
                                begin
                                        AN=4'b0000;//Todos los displays están encendidos
                                        CA=1'b1;//Led A del display apagado
                                        CB=1'b1;//Led B del display apagado
                                        CC=1'b0;//Led C del display ENCENDIDO
                                        CD=1'b1;//Led D del display apagado
                                        CE=1'b1;//Led E del display apagado
                                        CF=1'b0;//Led F del display ENCENDIDO
                                        CG=1'b0;//Led G del display ENCENDIDO
                                        //Todos estos leds prendidos están representando un signo de diferente que !=.
                                        DP=1'b0;//El punto del display está encendido.
                                end
                        else//La última condición siempre debe ser un else
                                begin
                                        AN=4'b1110;//Solo el 4to display está encendido
                                        CA=1'b0;//Led A del display ENCENDIDO
                                        CB=1'b0;//Led B del display ENCENDIDO
                                        CC=1'b1;//Led C del display apagado
                                        CD=1'b0;//Led D del display ENCENDIDO
                                        CE=1'b1;//Led E del display apagado
                                        CF=1'b1;//Led F del display apagado
                                        CG=1'b0;//Led G del display ENCENDIDO
                                        //Todos estos leds prendidos están representando un signo de menor o igual que <=.
                                        DP=1'b1;//El punto del display esta apagado.
                                end
        end
endmodule
```

Código UCF:

```
net "AN[3]" loc="F17";//Esta es la coordenada del bit menos significativo
net "AN[2]" loc="H17";
net "AN[1]" loc="C18";
net "AN[0]" loc="F15";//Esta es la coordenada del bit más significativo
//Para prender los diferentes displays de 7 segmentos de los 4 disponibles en la NEXYS 2

net "CA" loc="L18";
net "CB" loc="F18";
net "CC" loc="D17";
net "CD" loc="D16";
net "CE" loc="G14";
net "CF" loc="J17";
net "CG" loc="H14";
net "DP" loc="C17";
//Los diferentes Leds A,B,C,D,E,F,G y el punto de los displays

net "A[3]" loc="R17";
net "A[2]" loc="N17";
net "A[1]" loc="L13";
net "A[0]" loc="L14";
```

```
//Switches para los 4 bits de la entrada A

net "B[3]" loc="K17";
net "B[2]" loc="K18";
net "B[1]" loc="H18";
net "B[0]" loc="G18";
//Switches para los 4 bits de la entrada B
```

Código VHDL:

```
--COMPARADOR DE 2 NÚMEROS BINARIOS CON RESULTADO MOSTRADO EN DISPLAY DE 7 SEGMENTOS
--Para hacer este comparador se usar un condicional IF
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
--Las librerias IEEE y IEEE.STD_LOGIC_1164 sirven para que pueda solo usar el lenguaje VHDL

--ENTDIDAD: Aqu se declaran las entradas/salidas del programa
entity comparador is
    Port ( AN : out  STD_LOGIC_VECTOR (3 downto 0);
                     --El vector AN sirve para elegir cuales de los 4 displays de 7 segmentos que tiene
                     --la FPGA se van a activar. Los displays se prenden mandando un 0 lógico.
           CA : out  STD_LOGIC;
           CB : out  STD_LOGIC;
           CC : out  STD_LOGIC;
           CD : out  STD_LOGIC;
           CE : out  STD_LOGIC;
           CF : out  STD_LOGIC;
           CG : out  STD_LOGIC;
           DP : out  STD_LOGIC;
                     --Leds individuales de todos los displays de 7 segmentos, se prende cada uno con 0 logico
                     a : in  STD_LOGIC_VECTOR (3 downto 0);        --Numeros binarios A y B de 4 bits a comparar.
           b : in  STD_LOGIC_VECTOR (3 downto 0));
end comparador;

--ARQUITECTURA: Aqu se les dice a las entradas/salidas lo que van a hacer
architecture nombreArquitectura of comparador is--La arquitectura tiene su propio begin y end nombreArquitectura;
begin
         --process se declara siempre que vaya a usar condicionales if o case y dentro de sus parentesis van las entradas
         --que voy a usar dentro del condicional.
         process(A, B)
         begin--process tiene sus propios begin y end process;
                  if (A>B) then
                           AN <= "1110";--Solo el 4to display esta encendido
                           --Para que este orden se respete en el UCF debo asignar cada display partiendo desde el bit
                           --menos significativo hasta el mas significativo.
                           CA <= '1';--Led A del display apagado
                           CB <= '1';--Led B del display apagado
                           CC <= '0';--Led C del display ENCENDIDO
                           CD <= '0';--Led D del display ENCENDIDO
                           CE <= '1';--Led E del display apagado
                           CF <= '1';--Led F del display apagado
                           CG <= '0';--Led G del display ENCENDIDO
                           --Todos estos leds prendidos estan representando un signo de mayor que >.
                           DP <= '1';--El punto del display esta apagado.
                  elsif (A<B) then
                           AN <= "0111";--Solo el 1er display esta encendido
                           --Para que este orden se respete en el UCF debo asignar cada display partiendo desde el bit
                           --menos significativo hasta el mas significativo
                           CA <= '1';--Led A del display apagado
                           CB <= '1';--Led B del display apagado
                           CC <= '1';--Led C del display apagado
                           CD <= '0';--Led D del display ENCENDIDO
                           CE <= '0';--Led E del display ENCENDIDO
                           CF <= '1';--Led F del display apagado
                           CG <= '0';--Led G del display ENCENDIDO
                           --Todos estos leds prendidos estan representando un signo de menor que <.
                           DP <= '1';--El punto del display esta apagado.
                  elsif (A=B) then
                           AN <= "1001";--Los dos displays de en medio estan encendidos
                           --Para que este orden se respete en el UCF debo asignar cada display partiendo desde el bit
                           --menos significativo hasta el mas significativo.
                           CA <= '1';--Led A del display apagado
                           CB <= '1';--Led B del display apagado
                           CC <= '1';--Led C del display apagado
                           CD <= '0';--Led D del display ENCENDIDO
                           CE <= '1';--Led E del display apagado
                           CF <= '1';--Led F del display apagado
                           CG <= '0';--Led G del display ENCENDIDO
                           --Todos estos leds prendidos estan representando un signo de igual que =.
                           DP <= '1';--El punto del display esta apagado.
                  elsif (A>=B) then
                           AN <= "1011";--Solo el 2do display esta encendido
                           --Para que este orden se respete en el UCF debo asignar cada display partiendo desde el bit
                           --menos significativo hasta el mas significativo
                           CA <= '0';--Led A del display ENCENDIDO
                           CB <= '0';--Led B del display ENCENDIDO
                           CC <= '1';--Led C del display apagado
                           CD <= '0';--Led D del display ENCENDIDO
                           CE <= '1';--Led E del display apagado
                           CF <= '1';--Led F del display apagado
```

```
                        CG <= '0';--Led G del display ENCENDIDO
                        --Todos estos leds prendidos estan representando un signo de mayor o igual que >=.
                        DP <= '1';--El punto del display esta apagado.
                elsif (A<=B) then
                        AN <= "1101";--Solo el 3er display esta encendido
                        --Para que este orden se respete en el UCF debo asignar cada display partiendo desde el bit
                        --menos significativo hasta el mas significativo.
                        CA <= '0';--Led A del display ENCENDIDO
                        CB <= '1';--Led B del display apagado
                        CC <= '1';--Led C del display apagado
                        CD <= '0';--Led D del display ENCENDIDO
                        CE <= '1';--Led E del display apagado
                        CF <= '0';--Led F del display ENCENDIDO
                        CG <= '0';--Led G del display ENCENDIDO
                        --Todos estos leds prendidos estan representando un signo de mayor o igual que >=.
                        DP <= '1';--El punto del display esta apagado.
                else
                        AN <= "1001";--Los dos displays de en medio estan encendidos
                        --Para que este orden se respete en el UCF debo asignar cada display partiendo desde el bit
                        --menos significativo hasta el mas significativo.
                        CA <= '1';--Led A del display apagado
                        CB <= '1';--Led B del display apagado
                        CC <= '0';--Led C del display ENCENDIDO
                        CD <= '1';--Led D del display apagado
                        CE <= '1';--Led E del display apagado
                        CF <= '0';--Led F del display ENCENDIDO
                        CG <= '0';--Led G del display ENCENDIDO
                        --Todos estos leds prendidos estan representando un signo de menor o igual que <=.
                        DP <= '0';--El punto del display esta ENCENDIDO.
                end if;
        end process;
end nombreArquitectura;
--Este cadigo no se puede hacer asi porque como los displays muestran una sola imagen todos a la vez
--Aunque se cumplan varias condiciones del if a la vez, no se veran todos, necesitara 2 displays externos
--Para que se pudieran ver todas las condiciones a la vez.
```

Código UCF:

```
//Después de net se pone el nombre de la variable y su coordenada si es un vector
//y luego de loc se pone el código del elemento al que va dirigido de la NEXYS 2
//ENTRADAS:
net "a[3]" loc = "R17";
net "a[2]" loc = "N17";
net "a[1]" loc = "L13";
net "a[0]" loc = "L14";

net "b[3]" loc = "K17";
net "b[2]" loc = "K18";
net "b[1]" loc = "H18";
net "b[0]" loc = "G18";

//SALIDAS:
//Leds de todos los displays de 7 segmentos
net "CA" loc = "L18";
net "CB" loc = "F18";
net "CC" loc = "D17";
net "CD" loc = "D16";
net "CE" loc = "G14";
net "CF" loc = "J17";
net "CG" loc = "H14";
net "DP" loc = "C17";

//Para encender cada uno de los 4 displays de 7 segmentos en el orden en que fueron
//declararos en los vectores del código debo ponerlos desde el bit menos significativo
//hasta el más significativo.
net "AN[0]" loc = "F17";
//Bit menos significativo del vector AN al display AN3 que es el primero viéndolo de izq a der.
net "AN[1]" loc = "H17";
//Bit que le sigue al display AN2 que es el segundo viéndolo de izq a der
net "AN[2]" loc = "C18";
//Bit que le sigue al display AN1 que es el penúltimo viéndolo de izq a der
net "AN[3]" loc = "F15";
//Bit más significativo del vector AN al display AN0, que es el último viéndolo de izq a der.
```