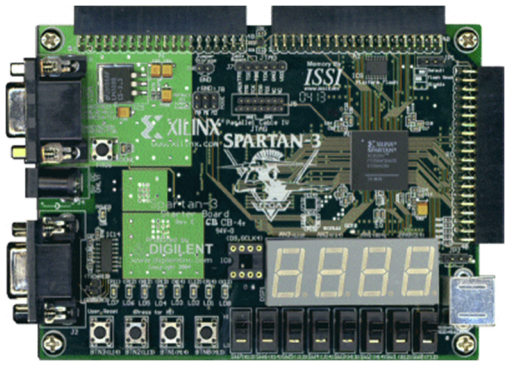
Representación de números en display 7 segmentos

El objetivo de la práctica es mostrar el valor de un dato de 8 bits sin signo en el *display* 7 segmentos de 4 dígitos incluido en la placa *Spartan-3 Starter Board* de Digilent. Con este propósito, debe definirse la entidad con los puertos de entrada/salida y describir en código VHDL la funcionalidad que se detalla a continuación.

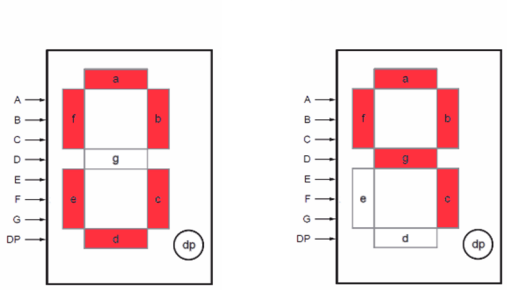


Entradas

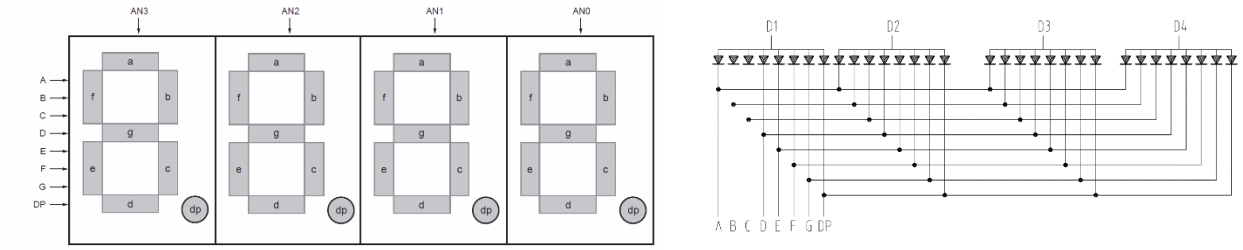
- Señal de reloj **clk** común a todos los componentes secuenciales del diseño.
- Señal de reset asíncrono **reset** conectada a los componentes secuenciales indicados en el esquema RTL simplificado.
- Señal de entrada **dato**, de 8 bits, cuyo valor, considerado entero sin signo, se representa en el *display*.

Salidas

- La señal **display7seg**, de 8 bits, que determina el estado apagado o encendido de los LEDs de los cuatro *displays* disponibles en la placa: cada *display* puede estar totalmente apagado o visualizar cualquier número entre 0 y 9. Por tanto, los valores posibles de **display7seg** son los valores de la codificación 7-segmentos de los números del 0 al 9 y el valor que corresponde a apagar el *display*. El bit más significativo de **display7seg** debe corresponder al segmento *A* del *display* y el menos significativo al punto *DP* (que estará siempre apagado). Para encender un LED es necesario dar valor '0' al bit correspondiente de **display7seg**.



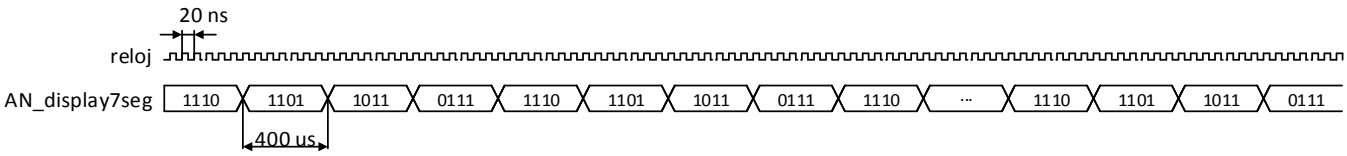
- La señal **AN_display7seg**, de 4 bits, que seleccionará de manera sucesiva y cíclica cada uno de los cuatro *displays* de 7 segmentos disponibles en la placa *Spartan-3 Starter Board*.



Como representa la figura anterior, este *display* está formado por 4 *displays* de 7 segmentos de ánodo común en el que los cátodos del mismo segmento de todos los *displays* están conectados entre sí: si, por ejemplo, el pin A vale '1', se iluminará el LED correspondiente de todos los *displays* cuyo ánodo AN valga '0'. De acuerdo con esto, la manera de representar un valor diferente en cada *display* es mediante la habilitación de uno de ellos –asignando valor '0' a su ánodo– y la deshabilitación de los restantes –con valor '1' en su ánodo–. La sucesiva y cíclica habilitación individual de cada uno de ellos permite, con la temporización adecuada, visualizar cualquiera de los códigos posibles en cada *display*. La señal **AN_display7seg** se encarga de la habilitación de los *displays* y por ello debe tomar los valores indicados en la tabla siguiente y repetirlos cíclicamente con la secuencia que muestra el cronograma.

AN_display7seg	*Display*	
1110	AN0	Unidades
1101	AN1	Decenas
1011	AN2	Centenas
0111	AN3	apagado

El tiempo que cada uno de los *displays* debe permanecer seleccionado para asegurar una correcta visualización es de 400 µs. Inicialmente, se seleccionará el *display* AN0 dando el valor "1110" a **AN_display7seg**. Transcurridos 400 µs, debe habilitarse el *display* AN1 actualizando el valor de **AN_display7seg**. Y así sucesivamente. Cuando el *display* AN3 haya estado habilitado durante 400 µs, se volverá a seleccionar el *display* AN0. Este proceso se repite indefinidamente. La activación de la señal de reset da a **AN_display7seg** su valor inicial "1110".



Para generar la temporización indicada debe tenerse en cuenta que la señal de entrada de reloj **CLK** tiene una frecuencia de 50 MHz.

Durante el tiempo que cada *display* está habilitado, el valor en el puerto de salida **display7seg** debe ser el del código que se pretende mostrar en dicho *display*. Por tanto, el valor de la señal **display7seg** en cada instante depende no sólo del valor a visualizar sino también de cuál es el *display* activo, es decir, de **AN_display7seg**.

La Figura 1 muestra el esquema RTL simplificado del diseño. La parte **Selección *display*** representa la funcionalidad descrita para la selección de los *displays*. La conversión del valor de **dato** de binario natural a código de 7 segmentos se describe a continuación.

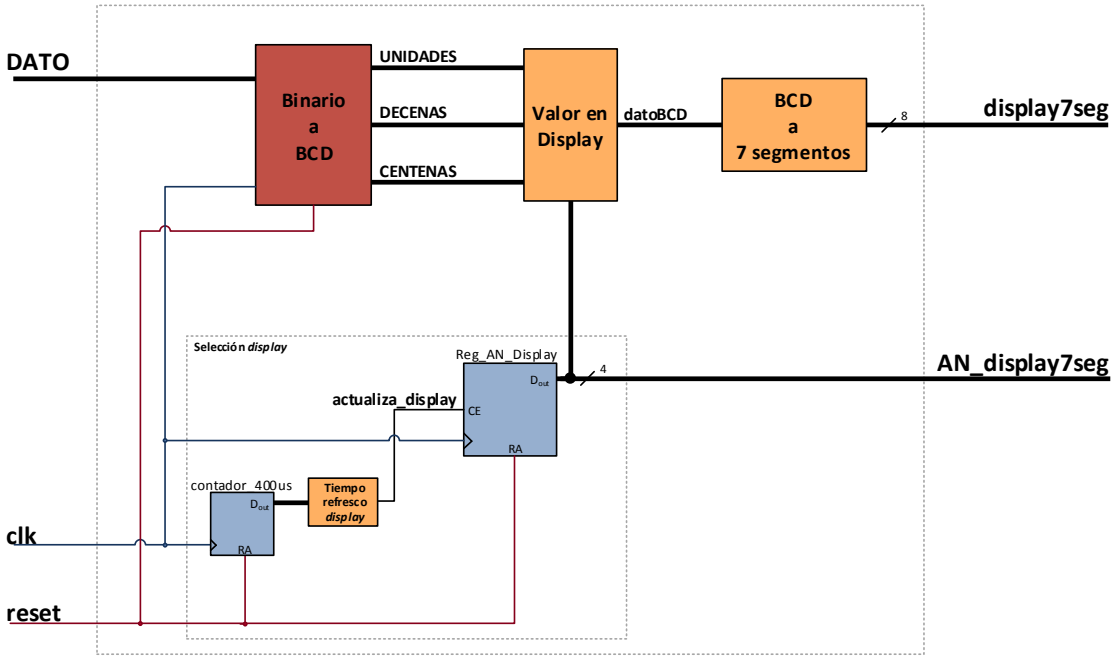


Figura 1. Esquema RTL simplificado

Verifica mediante simulación el correcto funcionamiento con los valores y los tiempos indicados en la siguiente tabla. Al iniciar la simulación, activa el reset asíncrono durante los primeros 85 ns y a continuación desactívalo. Considera una señal de reloj de 50 MHz.

t (ns)	Dato
0	131
1600	76
3200	0
4800	0
6400	15
8000	255
9600	1
11200	148

Conversión de binario a 7 segmentos

Con frecuencia resulta útil visualizar los valores de los datos que pueda producir cualquier dispositivo digital. La manera probablemente más familiar de hacerlo es mediante el uso de *displays* de siete segmentos. Para determinar si debe activarse o no cada uno de los segmentos de los diferentes *displays* necesarios para mostrar los dígitos que representan el valor de un dato, se suele recurrir como paso intermedio a la representación del valor en decimal codificado binario (BCD, *Binary-Coded Decimal*). Obtenida la codificación BCD, la representación en los *displays* 7 segmentos es directa.

1. Conversión de binario natural a BCD

La codificación de un número decimal en binario BCD es la forma más natural de expresar un número en base decimal mediante bits. Consiste en una traducción directa de cada uno de los dígitos del número a binario.

Para representar los diez posibles valores (del 0 al 9) que puede tomar cada dígito hacen falta cuatro bits y la codificación coincide con la binaria natural hasta el valor 1001, tal y como muestra la tabla adjunta. Las combinaciones desde la 1010 hasta la 1111 no se emplean en BCD, puesto que corresponden a valores decimales de dos dígitos y por ello se representan con dos palabras BCD.

Dígito decimal	BCD
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

Para la conversión a BCD de un valor codificado en binario con k bits se consideran tantas palabras de cuatro bits como dígitos tenga el valor 2^k-1 y se va a realizar el siguiente algoritmo de dos pasos que se repiten k veces:

1. Sumar 3 a todas y cada una de las palabras de cuatro bits del código BCD cuyo valor sea mayor o igual que 5.
2. Desplazar hacia la izquierda simultáneamente el código binario y las palabras de cuatro de bits, de manera que:
   - el bit más significativo del código binario pase a ser el menos significativo de la palabra de cuatro bits BCD correspondiente al dígito de menor peso.
   - El bit más significativo de la palabra i de cuatro bits BCD pasa a ser el menos significativo de la palabra i+1.

Las siguientes tablas muestran la evolución de las palabras BCD y del valor binario tomando como ejemplo un dato de 4 bits con el valor 1101. Puesto que con cuatro bits se pueden expresar en binario natural números de dos dígitos decimales, es necesario considerar dos

palabras BCD de cuatro bits: *dígito1* y *dígito2*. Inicialmente todos los bits de estas palabras BCD toman valor '0'.

dígito2				*dígito1*				binario				
0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**0**	**1**	**Estado inicial**

El primer paso es sumar 3 a todas las palabras BCD que tengan un valor mayor o igual que 5. Como todas están a 0, los bits del BCD permanecerán igual.

dígito2				*dígito1*				binario					
0	0	0	0	0	0	0	0	1	1	0	1		
0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**0**	**1**	**Paso1**	**Iteración1**

En el segundo paso se lleva a cabo el desplazamiento hacia la izquierda descrito anteriormente. En el bit menos significativo del dato binario entra un '0'.

dígito2				*dígito1*				binario					
0	0	0	0	0	0	0	0	1	1	0	1		
0	0	0	0	0	0	0	0	1	1	0	1		
0	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**0**	**1**	**0**	**Paso2**	**Iteración1**

Estos dos pasos del proceso se repiten *k* veces. Para el ejemplo de 4 bits, tras realizar el paso 2 por cuarta vez, en *dígito2* y *dígito1* se obtiene la representación BCD del valor binario.

dígito2				*dígito1*				binario					
0	0	0	0	0	0	0	0	1	1	0	1	estado inicial	
0	0	0	0	0	0	0	0	1	1	0	1	Paso1	Iteración1
0	0	0	0	0	0	0	1	1	0	1	0	Paso2	
0	0	0	0	0	0	0	1	1	0	1	0	Paso1	Iteración2
0	0	0	0	0	0	1	1	0	1	0	0	Paso2	
0	0	0	0	0	0	1	1	0	1	0	0	Paso1	Iteración3
0	0	0	0	0	1	1	0	1	0	0	0	Paso2	
0	0	0	0	1	0	0	1	1	0	0	0	Paso1	**Iteración4**
0	**0**	**0**	**1**	**0**	**0**	**1**	**1**	**0**	**0**	**0**	**0**	**Paso2**	
	1				**3**								

2. Conversión BCD a 7 segmentos

Para la conversión de BCD a 7 segmentos se emplea un decodificador que tiene como entrada cuatro bits con el valor BCD y ocho salidas para activar cada uno de los diodos LED que forman el *display*. Como se indicó anteriormente, el bit más significativo de la señal de salida **display7seg** corresponderá al segmento *A* del *display* y el menos significativo el punto *DP* (permanentemente apagado). Para encender un LED del *display* es necesario dar valor '0' al bit correspondiente de **display7seg**.

La siguiente tabla recoge los valores en los sucesivos pasos del algoritmo en las tres palabras BCD necesarias para representar el dato binario de 8 bits “11001111”.

digito3	*digito2*	*digito1*	binario		
0 0 0 0	0 0 0 0	0 0 0 0	1 1 0 0 1 1 1 1	Estado inicial	
0 0 0 0	0 0 0 0	0 0 0 0	1 1 0 0 1 1 1 1	Paso1	Iteración 1
0 0 0 0	0 0 0 0	0 0 0 1	1 0 0 1 1 1 1 0	Paso2	
0 0 0 0	0 0 0 0	0 0 0 1	1 0 0 1 1 1 1 0	Paso1	Iteración 2
0 0 0 0	0 0 0 0	0 0 1 1	0 0 1 1 1 1 0 0	Paso2	
0 0 0 0	0 0 0 0	0 0 1 1	0 0 1 1 1 1 0 0	Paso1	Iteración 3
0 0 0 0	0 0 0 0	0 1 1 0	0 1 1 1 1 0 0 0	Paso2	
0 0 0 0	0 0 0 0	0 1 1 0	0 1 1 1 1 0 0 0	Paso1	Iteración 4
0 0 0 0	0 0 0 0	1 1 0 0	1 1 1 1 0 0 0 0	Paso2	
0 0 0 0	0 0 0 1	0 0 1 0	1 1 1 1 0 0 0 0	Paso1	Iteración 5
0 0 0 0	0 0 1 0	0 1 0 1	1 1 1 0 0 0 0 0	Paso2	
0 0 0 0	0 0 1 0	0 1 0 1	1 1 1 0 0 0 0 0	Paso1	Iteración 6
0 0 0 0	0 1 0 0	1 0 1 1	1 1 0 0 0 0 0 0	Paso2	
0 0 0 0	0 1 0 1	0 0 0 1	1 1 0 0 0 0 0 0	Paso1	Iteración 7
0 0 0 0	1 0 1 0	0 0 1 1	1 0 0 0 0 0 0 0	Paso2	
0 0 0 1	0 0 0 0	0 0 1 1	1 0 0 0 0 0 0 0	Paso1	Iteración 8
0 0 1 0	0 0 0 0	0 1 1 1	0 0 0 0 0 0 0 0	Paso2	
2	0	7			

Implementación del algoritmo de conversión de binario a BCD

La arquitectura que implementa el algoritmo de conversión de binario a BCD se muestra en el esquema RTL simplificado de la Figura 2.

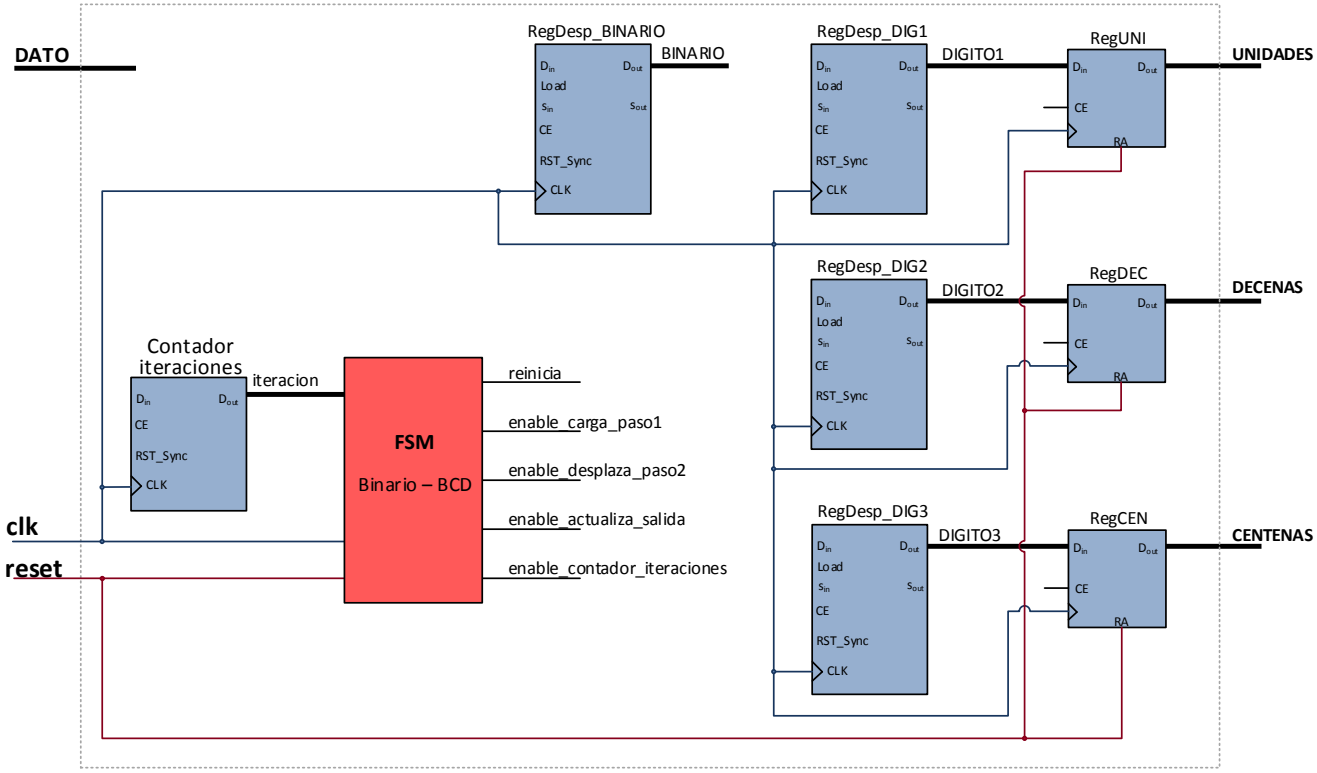


Figura 2. Esquema RTL simplificado del circuito para implementar el algoritmo de conversión de binario a BCD

El control de los pasos del proceso de conversión se realiza mediante una Máquina de Estados Finitos. La Figura 3 muestra el diagrama de la FSM que se propone, en la que se consideran 4 estados: `START`, que representa el estado inicial del proceso de conversión, `PASO1` y `PASO2` , que representan los dos pasos del algoritmo, y `FIN`, que representa el estado final. Las señales a las que da valor la FSM controlan el funcionamiento de los registros de datos, registros de desplazamiento y el contador en el esquema de la Figura 2.

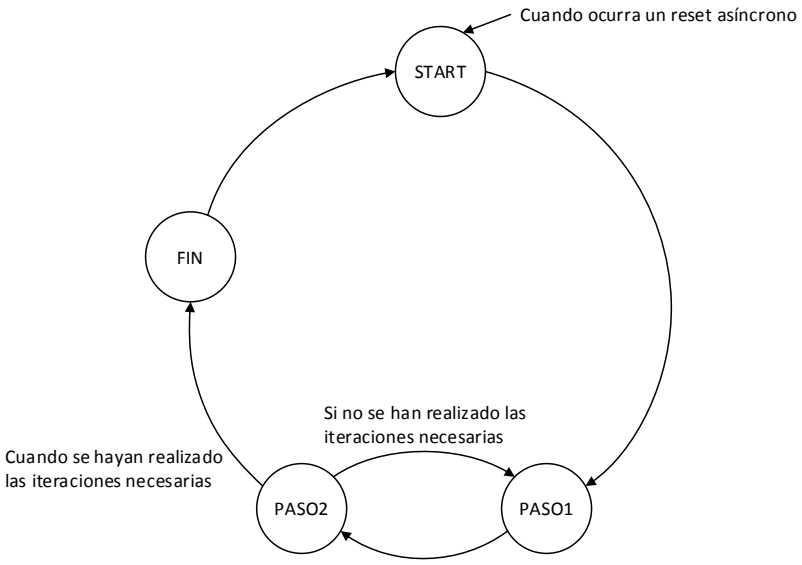


Figura 3. Diagrama de estados de la FSM que controla el proceso de conversión de binario a BCD

La señal `reinicia` se encarga de que el contador de iteraciones y los registros de desplazamiento que implementan el algoritmo de conversión tomen el valor del estado inicial del proceso de conversión.

La señal `enable_carga_paso1` controla la carga de los registros de desplazamiento que se realiza en el paso 1, y la señal `enable_desplaza_paso2` el desplazamiento que se realiza en el paso 2.

La señal `enable_actualiza_salida` controla la carga de los registros RegUNI, RegDEC y RegCEN con el valor de RegDesp_DIG1, RegDesp_DIG2 y RegDesp_DIG3 cuando el algoritmo ha terminado.

La señal `enable_contador_iteraciones` controla el incremento del contador necesario para saber cuándo el algoritmo ha llegado al final.