Universidad de Costa Rica | Facultad de Ingeniería | Escuela de Ingeniería Eléctrica

Anteproyecto III

Roberto Sánchez Cárdenas — B77059
Gabriel Jiménez Amador — B73895

San José, 30 de junio de 2021
IE0424 — Laboratorio de Circuitos Digitales

Índice

1. Introducción 1
2. Objetivos 2
3. Anteproyecto 2
   3.1. LEDs de 7 segmentos en la tarjeta Nexys 4 DDR: Pasos y tiempos para desplegar números 2
   3.2. Bancos en las FPGAs 3
   3.3. Esquemático Nexys 4 DDR 3
4. Diseño 5
   4.1. Ejercicio 1 7
   4.2. Ejercicio 2 7
   4.3. Ejercicio 3 8
   4.4. Ejercicio 4 9
Referencias 11

1. Introducción

En este laboratorio se diseñan dos módulos para controlar el display de 7 segmentos de la tarjeta de desarrollo Nexys 4 DDR, un módulo para desplegar números en decimal y otro para números en hexadecimal. Para esto se debe controlar los pines conectados a los ánodos y cátodos de cada LED del display. En este documento se presentan los cambios necesarios al archivo de restricción de pines para la tarjeta y los diseños de hardware. Los diseños de hardware se mantienen principalmente en un alto nivel por medio de diagramas.

2. Objetivos

Introducir el manejo de las entradas y salidas en una FPGA

- Investigar el funcionamiento de la tarjeta de desarrollo FPGA Nexys 4.
- Utilizar las herramientas del Xilinx Vivado.
- Conocer y aplicar el flujo de diseño para sistemas basados en FPGA.

3. Anteproyecto

3.1. LEDs de 7 segmentos en la tarjeta Nexys 4 DDR: Pasos y tiempos para desplegar números

Los LEDs de 7 segmentos en la Nexys 4 DDR estan configurados de forma que se comportan como un *display* continuo de 8 dígitos. Donde cada uno de estos contiene 7 segmentos que se pueden iluminar independientemente, cada dígito entonces puede contener 2^7 = 128 patrones diferentes [2].

Los ánodos de los 7 segmentos en cada *display* son comunes mientras que los cátodos se mantienen separados como se muestra en la Figura 1. Las 8 salidas de ánodos se verán como `DIGIT ENABLE` para cada dígito, por otro lado las salidas de los cátodos son comunes para todos los dígitos, es decir todos los segmentos A, B, C, D, ..., P son controlados por un sólo nodo: CA, CB, CD, ..., DP. Para poder desplegar números se debe variar tanto la activación del dígito con el ánodo común como los segmentos con los cátodos. Los anódos y cátodos ambos se activan en bajo en el Nexys 4 DDR [2].

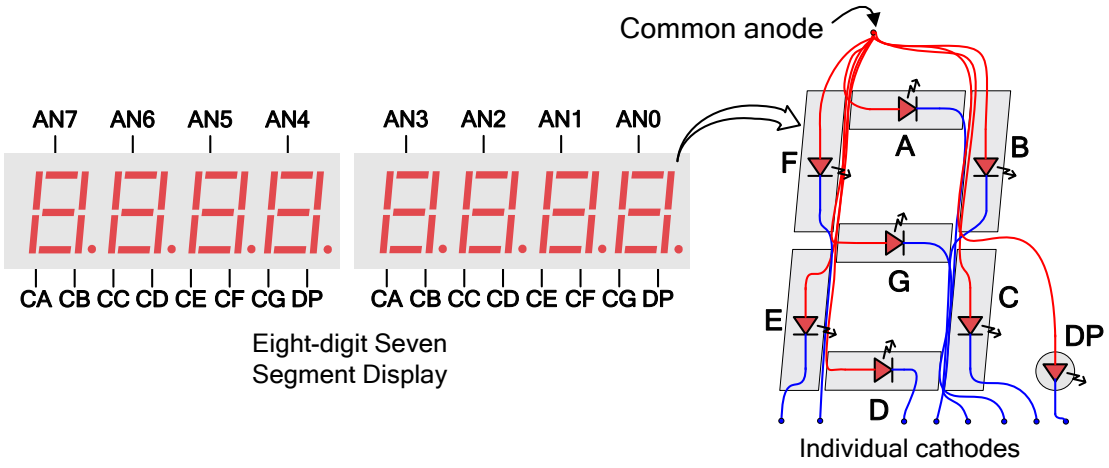

Figura 1: Conexiones del display de 7 segmentos del Nexys 4 DDR [2]

Para lograr desplegar números de forma continua en el display de 7 segmentos se debe implementar un controlador que refresque los números a una tasa más rápida que lo que notaría

el ojo humano que es por lo general mayor a 60 Hz. De forma que si por ejemplo se elige que todos los 8 dígitos cambien en un periodo de 16 ms, se refrescan a una tasa de 62.5 Hz y cada dígito estaría iluminado en una fracción 1/8 del total, es decir por 2 ms.

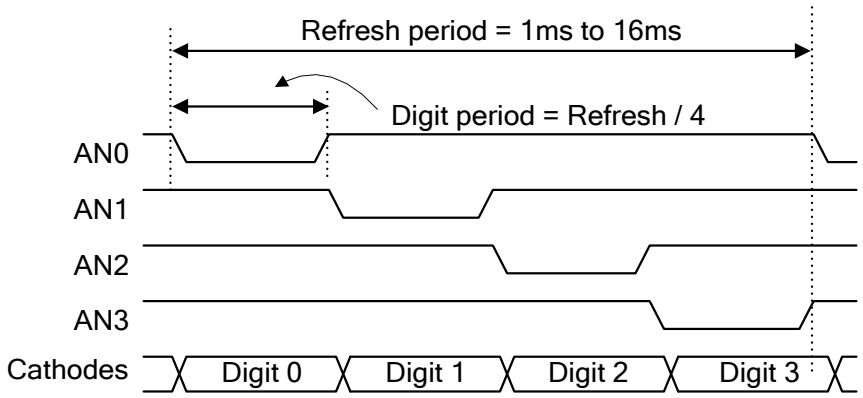

Figura 2: Diagrama de timing para un display de 7 segmentos con 4 dígitos [2]

3.2. Bancos en las FPGAs

En los FPGAs, un banco corresponde a un conjunto de pines de I/O que comparten un recurso tal y como una fuente de poder o una referencia de corriente de salida. En estos circuitos usualmente se especifican varios niveles de voltaje para el I/O, el voltaje seleccionado se dicta por el circuito digital externo. Para brindar flexibilidad, los FPGAs generalmente proporcionan múltiples bancos de I/Os que se pueden alimentar por separado. [1]

Estos bancos permiten que tenga una interfaz para varias familias lógicas. Esto sirve para hacer que la FPGA sea más fácil de fabricar (menos costosa) y se pueda reducir la cantidad de pines del dispositivo. [1]

3.3. Esquemático Nexys 4 DDR

Siguiendo el esquemático de la tarjeta Nexys 4 DDR [3]:

- El número de parte de los LEDs de 7 segmentos: KW4-281ASB.
- Los números de bancos a los que están conectados cada uno de los ánodos y cátodos de los LEDs de 7 segmentos:

- AN0: Banco 15
- AN1: Banco 15
- AN2: Banco 14
- AN3: Banco 15
- AN4: Banco 14
- AN5: Banco 14
- AN6: Banco 35
- AN7: Banco 14
- CA: Banco 14
- CB: Banco 14
- CC: Banco 15
- CD: Banco 15
- CE: Banco 14
- CF: Banco 14
- CG: Banco 14
- DP: Banco 15

4. Diseño

Para los ejercicios que ocupen el controlador del display de 7 segmentos (ejercicio 2 en adelante) se debe modificar el archivo de restricciones de localización de pines para poder desplegar las salidas de este en la tarjeta de desarrollo.

Siguiendo los pines de acuerdo a los bancos que están conectados cada uno de los ánodos y cátodos de los LEDs de 7 segmentos [3], el archivo de constraints `system.xdc` quedaría como está representado en la Figura 3. En este se relaciona cada pin de la tarjeta de desarrollo con un puerto de salida del módulo principal `system.v`. `an_out` y `c_out` serían entonces los pines a conectar con los LEDs del display de 7 segmentos, ánodos y cátodos respectivamente.

```
set_property CFGBVS VCCO [current_design]
set_property CONFIG_VOLTAGE 3.3 [current_design]

# clk
set_property -dict { PACKAGE_PIN E3 IOSTANDARD LVCMOS33 } [get_ports { clk }];
create_clock -add -name sys_clk_pin -period 10.00 [get_ports {clk}];

# switches
set_property -dict { PACKAGE_PIN V10 IOSTANDARD LVCMOS33 } [get_ports { resetn }];

# leds
set_property -dict { PACKAGE_PIN H17 IOSTANDARD LVCMOS33 } [get_ports { out_byte[0] }];
set_property -dict { PACKAGE_PIN K15 IOSTANDARD LVCMOS33 } [get_ports { out_byte[1] }];
set_property -dict { PACKAGE_PIN J13 IOSTANDARD LVCMOS33 } [get_ports { out_byte[2] }];
set_property -dict { PACKAGE_PIN N14 IOSTANDARD LVCMOS33 } [get_ports { out_byte[3] }];
set_property -dict { PACKAGE_PIN R18 IOSTANDARD LVCMOS33 } [get_ports { out_byte[4] }];
set_property -dict { PACKAGE_PIN V17 IOSTANDARD LVCMOS33 } [get_ports { out_byte[5] }];
set_property -dict { PACKAGE_PIN U17 IOSTANDARD LVCMOS33 } [get_ports { out_byte[6] }];
set_property -dict { PACKAGE_PIN U16 IOSTANDARD LVCMOS33 } [get_ports { out_byte[7] }];

set_property -dict { PACKAGE_PIN V12 IOSTANDARD LVCMOS33 } [get_ports { out_byte_en }];
set_property -dict { PACKAGE_PIN V14 IOSTANDARD LVCMOS33 } [get_ports { trap }];

#7 segment display
# anodes
set_property -dict { PACKAGE_PIN T10   IOSTANDARD LVCMOS33 } [get_ports { an_out[0] }];
set_property -dict { PACKAGE_PIN R10   IOSTANDARD LVCMOS33 } [get_ports { an_out[1] }];
set_property -dict { PACKAGE_PIN K16   IOSTANDARD LVCMOS33 } [get_ports { an_out[2] }];
set_property -dict { PACKAGE_PIN K13   IOSTANDARD LVCMOS33 } [get_ports { an_out[3] }];
set_property -dict { PACKAGE_PIN P15   IOSTANDARD LVCMOS33 } [get_ports { an_out[4] }];
set_property -dict { PACKAGE_PIN T11   IOSTANDARD LVCMOS33 } [get_ports { an_out[5] }];
set_property -dict { PACKAGE_PIN L18   IOSTANDARD LVCMOS33 } [get_ports { an_out[6] }];
set_property -dict { PACKAGE_PIN H15   IOSTANDARD LVCMOS33 } [get_ports { an_out[7] }];
# cathodes
set_property -dict { PACKAGE_PIN J17   IOSTANDARD LVCMOS33 } [get_ports { c_out[0] }];
set_property -dict { PACKAGE_PIN J18   IOSTANDARD LVCMOS33 } [get_ports { c_out[1] }];
set_property -dict { PACKAGE_PIN T9    IOSTANDARD LVCMOS33 } [get_ports { c_out[2] }];
set_property -dict { PACKAGE_PIN J14   IOSTANDARD LVCMOS33 } [get_ports { c_out[3] }];
set_property -dict { PACKAGE_PIN P14   IOSTANDARD LVCMOS33 } [get_ports { c_out[4] }];
set_property -dict { PACKAGE_PIN T14   IOSTANDARD LVCMOS33 } [get_ports { c_out[5] }];
set_property -dict { PACKAGE_PIN K2    IOSTANDARD LVCMOS33 } [get_ports { c_out[6] }];
set_property -dict { PACKAGE_PIN U13   IOSTANDARD LVCMOS33 } [get_ports { c_out[7] }];
```

Figura 3: Archivo de restricciones de localización de pines a utilizar.

4.1. Ejercicio 1

En este ejercicio se planea simular con el firmware dado para obtener `out_byte`, medir los tiempos que se piden y responder las preguntas planteadas.

Los cambios en el firmware para observer la cuenta ascendente correspondería a cambiar el valor de LOOP_WAIT_LIMIT para que las transiciones de los LEDs puedan ser notadas fácilmente por el ojo humano.

4.2. Ejercicio 2

Se debe implementar el módulo controlador del display de 7 segmentos, para la creación de este se debe considerar que en cada display sólo se puede mostrar números de 0 a 9. Se plantea convertir el número binario a BCD, como muestra la Tabla 1 y luego activar los segmentos necesarios para representar el número en el display de 7 segmentos siguiendo la Tabla 2.

Tabla 1: Decimal a BCD

Decimal	BCD			
	Bit 3	Bit 2	Bit 1	Bit 0
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1

Tabla 2: Tabla de verdad para el display de 7 seg.

Segmentos activos							Número
a	b	c	d	e	f	g	
×	×	×	×	×	×		0
	×	×					1
×	×		×	×		×	2
×	×	×	×			×	3
	×	×			×	×	4
×		×	×		×	×	5
×		×	×	×	×	×	6
×	×	×					7
×	×	×	×	×	×	×	8
×	×	×	×		×	×	9

La utilidad de convertir el número a BCD primero viene en el hecho en que el número original de 32 bits se representaría como 8 números BCD de 4 bits. Cada uno de estos representaría al final un display de 7 segmentos del Nexys 4 DDR.

Además se tiene que mantener un contador para desplegar los 8 números de forma continua, siguiendo un diagrama de timing similar al de la Figura 2. Este contador controlaría el valor de los pines de los cátodos y ánodos y debe ser ajustado para ser incrementado en una tasa apropiada para el ojo humano (≥60 Hz).

Resumiendo el diseño del módulo controlador del display de 7 segmentos, `7_segment_dec`, se seguiría el diagrama de bloques de la Figura 4.

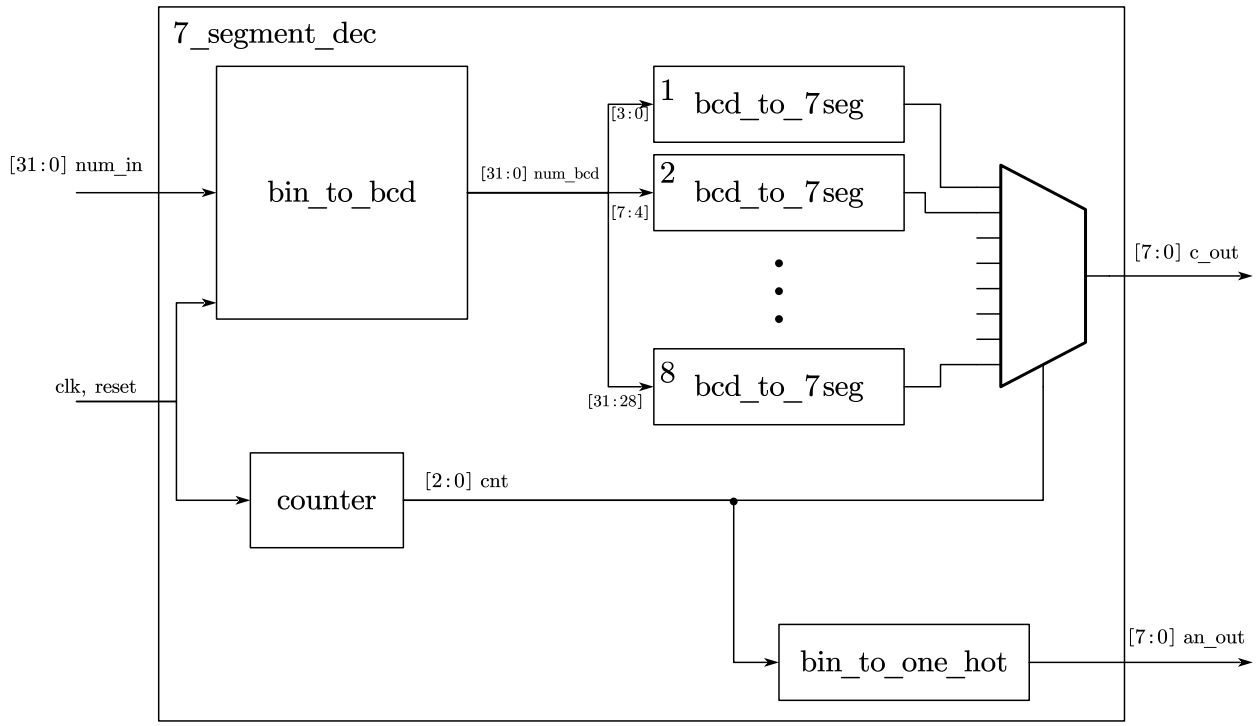

Figura 4: Diagrama de bloques del controlador del display de 7 segmentos de la tarjeta Nexys 4 DDR

También se debe considerar dos cosas en este diseño:

- Tanto los cátodos como ánodos de los LEDs de 7 segmentos se activan en bajo.
- El display de 7 segmentos puede mostrar un número máximo de 99 999 999, se debe incluir un caso adicional que considere números mayores a este y que sólo muestre el número máximo.

4.3. Ejercicio 3

Ahora se debe implementar un controlador del display de 7 segmentos, `7_segment_hex` pero incluyendo números hexadecimales. Se sigue un diseño similar al del ejercicio anterior, sólo que ahora como se conoce que el número entrante es hexadecimal, no se debe convertir a BCD puesto que en binario, cada bloque de 4 bits corresponde a un número en hexadecimal. Se puede omitir el bloque de conversión a BCD como se muestra en el diagrama de bloques diseñado en la Figura 5.

También se debe modificar el bloque `bcd_to_7seg` para incluir las representaciones faltantes de `0xa, 0xb, ... 0xf`, siguiendo la Tabla 3.

Además, ya no se debe considerar el caso de números mayores a 99 999 999; cualquier número de 32 bits será posible representar en hexadecimal.

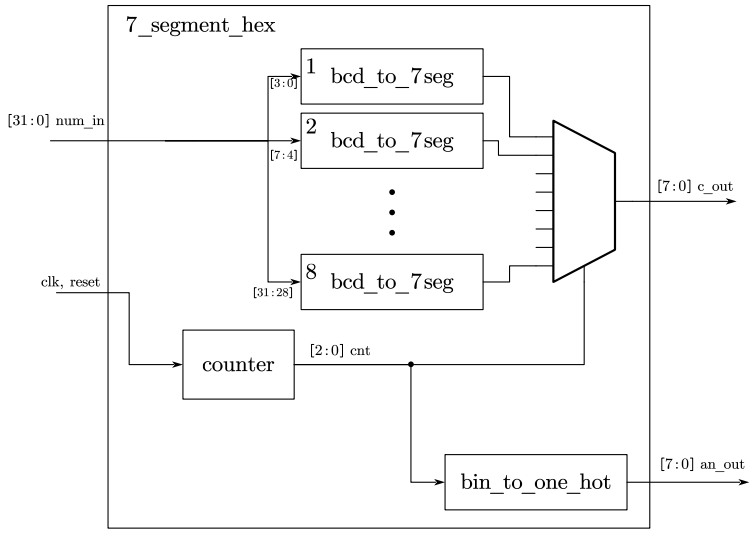

Figura 5: Diagrama de bloques del controlador del display de 7 segmentos de la tarjeta Nexys 4 DDR para números hexadecimales

Tabla 3: Tabla de verdad para el display de 7 seg.

Segmentos activos							Número
a	b	c	d	e	f	g	
×	×	×	×	×	×		0
	×	×					1
×	×		×	×		×	2
×	×	×	×			×	3
	×	×			×	×	4
×		×	×		×	×	5
×		×	×	×	×	×	6
×	×	×					7
×	×	×	×	×	×	×	8
×	×	×	×		×	×	9
×	×	×		×	×	×	A
		×	×	×	×	×	b
×			×	×	×		C
	×	×	×	×		×	d
×			×	×	×	×	E
×				×	×	×	F

4.4. Ejercicio 4

En este ejercicio se implementa un selector que permite elegir si se quere cambiar entre el despliegue del número binario en decimal o en hexadecimal en el display de 7 segmentos de la tarjeta Nexys 4 DDR. Para poder controlar el *switch* se debe agregar la siguiente línea en el archivo de restricción de pines de la Figura 3:

```
set_property -dict { PACKAGE_PIN J15 IOSTANDARD LVCMOS33 } [get_ports { base_sel }];
```

Con el valor del selector `base_sel` en el *switch* sólo se elige la salida deseada, con muxes o en Verilog un `parallel case`.

El diseño del módulo de este ejercicio se observa en el diagrama de bloques de la Figura 6.

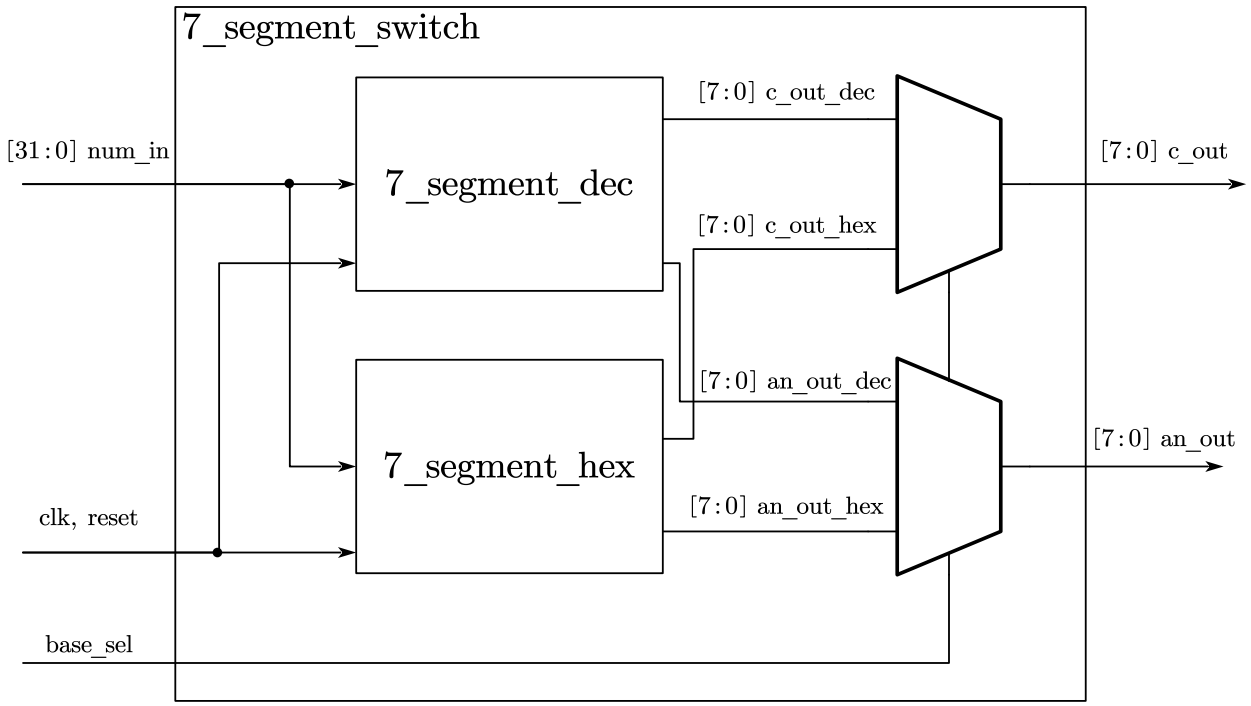

Figura 6: Diagrama de bloques para el switch de controladores del display de 7 segmentos de la tarjeta Nexys 4 DDR

Referencias

[1] National Semiconductor Dennis Hudgins. *Power Supply Design Considerations for Modern FPGAs*. 2010. URL: https://www.ti.com/lit/an/snoa864/snoa864.pdf.

[2] Inc. Digilent. *Nexys 4 DDR Reference Manual*. 2016. URL: https://reference.digilentinc.com/_media/reference/programmable-logic/nexys-4-ddr/nexys4ddr_rm.pdf.

[3] Inc. Digilent. *Nexys 4 DDR Schematic*. 2014. URL: https://reference.digilentinc.com/_media/reference/programmable-logic/nexys-4-ddr/nexys-4-ddr_sch.pdf.