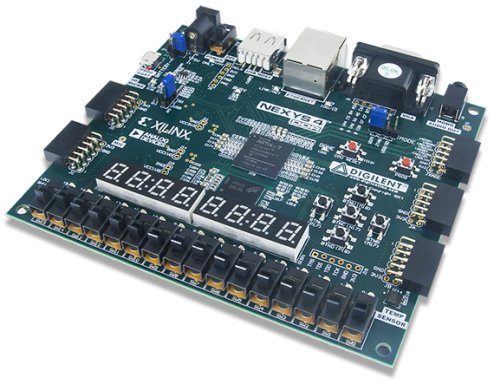
UNIVERSIDAD NACIONAL DE SAN AGUSTÍN

FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS

ESCUELA PROFESIONAL DE INGENIERÍA ELECTRÓNICA

Microelectrónica

Guía de Prácticas



Autor:
PhD. Alexander B. HILARIO TACURI

Contenido

Práctica 1

(Fecha de entrega: M' 04-A: 07 de mayo, M' 04-B: 08 de mayo, M' 17: 06 de mayo)

1. Objetivos

- Introducción de programación VHDL para FPGAs
- Aprender a escribir $test - benches$ en VHDL
- Aprender el flujo de diseño con el Vivado: Síntesis, simulación y generación del *Bitstream*
- Aprender como asignar pins de entrada y salida del FPGA y cargar el *Bitstream* en el Nexys-4 DDR Artix-7 FPGA Board.

2. Programación en VHDL

- Revise el material referente al curso.

3. Desarrollo de la práctica

- **Problema 1:** Escriba el código en VHDL de un restador completo de 1 bit utilizando ecuaciones lógicas. Escriba un código en VHDL de un restador de 4 bits utilizando el modulo definido del restador completo de 4 bits.
- **Problema 2:** Diseñe una unidad aritmética y lógica (ALU) que implemente las 8 funciones descritas en la Tabla 1.

Control	Instrucción	Operación
000	Add	Suma de A+B+Cin
001	Sub	Resta de A-B-Cin
010	Or	A or B
011	And	A and B
100	Shl	Desplazamiento a la izquierda
101	Shr	Desplazamiento a la derecha
110	Rol	Rotación a la izquierda
111	Ror	Rotación a la derecha

Tabla 1: Instrucciones del ALU

Las entradas A y B son de 4 bits. La salida también es de 4 bits y se tiene los acarreos de entrada y salida de 1 bit cada uno.

4. Flujo de diseño

- Cree un nuevo proyecto Vivado. Seleccione el dispositivo FPGA Artix-7
- Escriba el código VHDL que implementa la expresión Booleana simplificada. Sintetice su circuito.
- Escriba el *test-bench* VHDL para probar todas las combinaciones posibles de las entradas.
- Realice la *Functional Simulation* (Run simulation → Run Behavioral Simulation).
- Asignación de entradas y salidas: Crear un archivo XDC. Use los switchs para las entradas A, B y Cin. Use los leds para las salidas.

- Implemente su diseño. (Run implementation)
- Realice la *Timing Simulation*. (Run simulation → Run Post-Implementation Timing Simulation).
- Generar el archivo *bitstream* (Generate Bitstream)
- Descargar el *bitstream* en el FPGA. (Open Hardware Manager)

5. Presentación del informe

- Cuando el docente lo autorice, es decir luego de presentar la práctica funcionando, puede mandar un correo electrónico a: ahilariot@unsa.edu.pe con las siguientes especificaciones:
  - Asunto: Microelectrónica X - Informe de práctica 1, Grupo Y.
  - El correo electrónico debe contener: Informe en pdf (Escrito en LaTeX), y un enlace para un Drive personal donde estén los programas escritos en Vivado.

Práctica 2

(Fecha de entrega: M' 04-A: 21 de mayo, M' 04-B: 22 de mayo, M' 17: 20 de mayo)

1. Objetivos

- Usar la descripción estructural en VHDL.
- Probar circuitos secuenciales en un FPGA.

2. Programación en VHDL

- Revise el material referente al curso.

3. Desarrollo de la práctica

- **Problema 1:** Un circuito secuencial tiene una entrada X, una señal de reloj CLK y dos salidas S y V. Escriba un código en VHDL que implemente la tabla de estados presentada en la Tabla 1 e indique el objetivo de esta maquina de estados.

Estado actual	**Siguiente estado**		**Salida**	
	x=0	x=1	x=0	x=1
S0	S1	S2	1,0	0,0
S1	S3	S4	1,0	0,0
S2	S4	S4	0,0	1,0
S3	S5	S5	0,0	1,0
S4	S5	S6	1,0	0,0
S5	S0	S0	0,0	1,0
S6	S0	S0	1,0	0,1

Tabla 2: Tabla de estados problema 1

- **Problema 2:** Escriba un código en VHDL para Implementar un contador BCD. Este contador debe tener la opción de escoger si es ascendente o descendente y ademas debe contar con un pulsador de reset.

4. Flujo de diseño

- Cree un nuevo proyecto Vivado. Seleccione el dispositivo FPGA Artix-7
- Escriba el código VHDL para los dos problemas usando la descripción estructural: Crear un archivo separado para cada una de las operaciones y un archivo principal.
- Escriba el *test-bench* VHDL para probar los casos más significativos.
- Realice la *Functional Simulation* y la *Timing Simulation*.
- Asignación de entradas y salidas: Utilice los switches y pulsadores de la placa como entradas y los leds y display de 7 segmentos como salidas.
- Generar el archivo *bitstream* (Generate Bitstream)
- Descargar el *bitstream* en el FPGA.

5. Presentación del informe

- Cuando el docente lo autorice, es decir luego de presentar la práctica funcionando, puede mandar un correo electrónico a: ahilariot@unsa.edu.pe con las siguientes especificaciones:
  - Asunto: Microelectrónica X - Informe de práctica 2, Grupo Y.
  - El correo electrónico debe contener: Informe en pdf (Escrito en LaTeX), y un enlace para un Drive personal donde estén los programas escritos en Vivado.

Práctica 3

(Fecha de entrega: M' 04-A: 04 de junio, M' 04-B: 05 de junio, M' 17: 03 de junio)

1. Objetivos

- Utilizar la descripción estructural en VHDL.
- Diseñar sistemas digitales con circuitos combinacionales y secuenciales en el FPGA.

2. Programación en VHDL

- Revise el material referente al curso.

3. Desarrollo de la práctica

- **Problema:** Diseñe un clasificador de paquetes para clasificar los paquetes en función de sus pesos y realizar un seguimiento de los paquetes de diferentes categorías. El clasificador tiene un restablecimiento asíncrono activo en alto y realizará un seguimiento de los paquetes desde el último restablecimiento. Los paquetes deben clasificarse en 6 grupos:

  1. Entre 1 y 200 gramos
  2. Entre 201 y 500 gramos
  3. Entre 501 y 800 gramos
  4. Entre 801 y 1000 gramos
  5. Entre 1001 y 2000 gramos
  6. Mayores que 2000 gramos

  Necesita decodificar las medidas de peso y clasificarlas en varios grupos. La entrada al circuito será un número binario sin signo de 12 bits (que indica el peso del paquete), una señal de reloj CLK y un reset. Una de las salidas será *currentGrp*, un número sin signo de 3 bits que representa el número de grupo actual. También habrá seis salidas sin signo de 8 bits $Grp1 - Grp6$ que representan el número de elementos pesados en cada categoría desde el último reinicio. El reset permitira borrar todos los conteos.

  Las líneas de salida tienen la siguiente funcionalidad: *currentGrp*: Genera el número de grupo para el peso que se está aplicando actualmente al clasificador. Cuando se aplica un peso de cero, debería generar un cero. Esto debería actualizarse tan pronto como cambie el peso de un paquete y no necesariamente refleje el último grupo al que se asignó un paquete. $Grp1 - Grp6$: Genera el número de objetos que se han pesado en cada grupo desde el último reinicio. Estas salidas deben ser cero cuando reset = "1".

  Tenga en cuenta que la funcionalidad de las dos salidas es tal que la descripción de *currentGrp* será puramente combinacional ya que no depende de ninguna entrada previa. Pero la descripción de $Grp1 - Grp6$ será secuencial, ya que depende no solo de la entrada actual sino también de las entradas anteriores.

  Cualquier salida secuencial debería cambiar en el flanco descendente del reloj. Tenga en cuenta que la señal de CLK será significativamente más rápida que la duración de la señal de peso. Como tal, debe asegurarse de que el recuento solo se actualice una vez para un peso de entrada dado. En segundo lugar,

los nuevos objetos solo pueden detectarse y clasificarse si se permite que el peso llegue a cero. Esto es para asegurar que cualquier fluctuación en el peso después de haber sido muestreado no se considere un elemento nuevo. Solo el primer peso después de 0 actualiza un conteo grupal.

Ejemplo de secuencia de entrada
Reset →Poner 250 gramos →Quitar el paquete →Poner 300 gramos →Quitar el paquete →Poner 501 gramos →Poner 512 gramos más.

Al final de esta secuencia, las salidas deben ser:
grp1 = grp4 = grp5 = 0x00
grp2 = 0x02
grp3 = 0x01
currentGrp = 0x5

Tenga en cuenta que después de muestrear 501 gramos en grp3, agregar 512 gramos solo actualiza el grupo actual y no el recuento de grp5.

4. Flujo de diseño

- Cree un nuevo proyecto Vivado. Seleccione el dispositivo FPGA Artix-7
- Escriba el código VHDL para el problema propuesto.
- Escriba el *test-bench* VHDL para probar el circuito y pruebe el ejemplo presentado.
- Realice la *Functional Simulation* y la *Timing Simulation*.
- Asignación de entradas y salidas: Use los switches y pulsadores para las entradas y los led y displays de 7 segmentos para las salidas.
- Generar el archivo *bitstream*.
- Descargar el *bitstream* en el FPGA.

5. Presentación del informe

- Cuando el docente lo autorice, es decir luego de presentar la práctica funcionando, puede mandar un correo electrónico a: ahilariot@unsa.edu.pe con las siguientes especificaciones:
  - Asunto: Microelectrónica X - Informe de práctica 3, Grupo Y.
  - El correo electrónico debe contener: Informe en pdf (Escrito en LaTeX), y un enlace para un Drive personal donde estén los programas escritos en Vivado.

Práctica 4

(Fecha de entrega: M' 04-A: 25 de junio, M' 04-B: 26 de junio, M' 17: 24 de junio)

1. Objetivos

- Diseño e implementación de un sistema digital completo en FPGA.

2. Programación en VHDL

- Revise el material referente al curso.

3. Desarrollo de la práctica

- **Problema:** Diseñar una máquina de estados finitos que simulará la operación de un parquímetro. Los botones de la tarjeta representarán diferentes denominaciones de monedas y por lo tanto de segundos (ver Tabla 1) y la pantalla LED de siete segmentos mostrará la cantidad total de segundos restantes antes de que expire el medidor.

Pulsador	Tiempo
Pulsador 1	Agregar 30 segundos
Pulsador 2	Agregar 120 segundos
Pulsador 3	Agregar 180 segundos
Pulsador 4	Agregar 300 segundos
Switch 1	Reiniciar a 15 segundos
Switch 2	Reiniciar a 185 segundos

Tabla 3: Tabla de tiempos por pulsador

Tan pronto como se presiona un botón, se debe agregar el tiempo inmediatamente. Cuando quedan menos de 180 segundos, la pantalla debe parpadear con un período de 2 segundos y un ciclo de trabajo del 50% (encendido durante 1 segundo y apagado durante 1 segundo; por lo que verá recuentos alternativos en la pantalla, por ejemplo, 185, en blanco, 183, en blanco, 181 ...). Cuando el tiempo ha expirado, la pantalla debe parpadear con un período de 1 segundo y un ciclo de trabajo del 50% (encendido durante 0,5 segundos y apagado durante 0,5 segundos).

Por ejemplo, cuando se inicia el tablero, debe estar en el estado restante del tiempo 0 y parpadear 0000 a una velocidad de 0,5 segundos. Si se presiona el botón 4, la pantalla debería leer 300 segundos y comenzar la cuenta regresiva. Cuando el tiempo cuenta hasta 200 segundos y se presiona el botón 2, la pantalla debería leer 380 segundos (200 + 180). Si el interruptor 1 se pone alto, el tiempo debería cambiar a 15segundos y parpadear.

El valor máximo de tiempo será 9999 y cualquier intento de aumentar más allá de 9999, debería dar como resultado que el contador se establezca por defecto en 9999 y una cuenta atrás desde allí.

4. Flujo de diseño

- Cree un nuevo proyecto Vivado. Seleccione el dispositivo FPGA Artix-7.

- Escriba el código VHDL para el problema.
- Escriba el *test-bench* VHDL para probar el circuito y pruebe los casos mas representativos.
- Realice la *Functional Simulation* y la *Timing Simulation*.
- Asignación de entradas y salidas: Crear un archivo XDC. Use los pulsadores y switches para las entradas y los display de 7 segmentos para las salidas.
- Generar y Descargar el *bitstream* en el FPGA.

5. Presentación del informe

- Cuando el docente lo autorice, es decir luego de presentar la práctica funcionando, puede mandar un correo electrónico a: ahilariot@unsa.edu.pe con las siguientes especificaciones:
  - Asunto: Microelectrónica X - Informe de práctica 4, Grupo Y.
  - El correo electrónico debe contener: Informe en pdf (Escrito en LaTeX), y un enlace para un Drive personal donde estén los programas escritos en Vivado.

Práctica 5

(Fecha de entrega: M' 04-A: 13 de julio, M' 04-B: 10 de julio, M' 17: 13 de julio)

1. Objetivos

- Aprender a usar el standard VGA del FPGA.
- Aprender a usar el protocolo PS/2 del FPGA.

2. Programación en VHDL

- Revise el material referente al curso.

3. Desarrollo de la práctica

- **Problema:** Crear un controlador simple de teclado y VGA. El controlador del teclado permitirá la comunicación desde el teclado. El controlador VGA se usará para mostrar algunos patrones gráficos simples en el monitor de la computadora conectado a la placa.

  Parte 1. Diseño de interfaz de teclado: Los valores enviados desde el teclado se mostrarán en la pantalla de siete segmentos en la placa.

  El protocolo PS2 es un esquema simple de dos hilos que utiliza transmisión en serie para transmitir los datos a la placa. Si bien el bus de dos hilos tiene un diseño bidireccional, solo lo usaremos como entrada para el FPGA. [Normalmente, la escritura en el teclado se usa para restablecer, encender las diversas luces indicadoras, etc.]

  Cuando se presiona una tecla, se envía una secuencia de bytes en serie a través del bus de dos hilos. Cada tecla del teclado recibe un "scancode" único (consulte el manual del usuario de la placa). Para detectar cuando las teclas se presionan inicialmente y luego se sueltan, el teclado enviará una secuencia de bytes por cada pulsación de tecla. El primer byte enviado por el teclado generalmente se llama "make code" y representa la tecla que se presiona. El último byte enviado por el teclado es el "break code" que representa la tecla que se libero.

  Por ejemplo, considere la situación en la que un usuario presiona la letra "a":

  1. El usuario presiona la tecla "a"
  2. El teclado envía "make code" (que es "1C" para la tecla "a") en serie. El teclado sigue enviando el "make code" cada 100 ms hasta que el usuario suelte la tecla.
  3. El usuario suelta la tecla "a".
  4. El teclado envía el código "key up" "F0" en serie
  5. El teclado envía el "break code" (que es "1C" para la tecla "a") en serie

  Solo necesitaremos buscar los bytes del "break code". Por lo tanto, simplemente podemos monitorear los bits para el código de escaneo "key up" que indica que la tecla ha sido liberada. Cuando se ha enviado este byte, se enviará el "break code" para la tecla liberada.

  Para transmitir la secuencia de bytes, el teclado primero fuerza la línea DATA a un nivel bajo para crear el bit de inicio. Los bits se transmiten utilizando el flanco descendente de CLK para la sincronización.

Esto se ilustra en la Figura 1. La señal de DATOS cambia de estado cuando la señal de CLK es alta, y los DATOS son válidos para leer en el flanco descendente de CLK.

Entonces, la operación básica de su diseño es la siguiente:

- En el flanco descendente de CLK, use un registro de desplazamiento para capturar cada bit de datos
- Cuando se hayan enviado los 11 bits (star, scancode, parity, stop), puede mirar el scancode y decidir qué hacer.
- Si el scancode es una "key up", es decir, "F0", sabrá que la siguiente secuencia de datos enviada por el teclado es el Scancode final que necesita.
- Capture el scancode final siguiendo los mismos pasos que antes y envíe este valor desde su controlador de teclado.

Muestre los 2 dígitos hexadecimales inferiores del código de escaneo recibido por el controlador en los displays de siete segmentos (tenga en cuenta que algunos scan tienen 2 dígitos y algunos tienen 4 dígitos, consulte el Manual del usuario de la placa). También debe tener una señal estroboscópica para indicar que el controlador del teclado está emitiendo una nueva pulsación de tecla. Una señal estroboscópica es un pulso corto en uno de los LED de la placa.

Parte 2. Diseño de interfaz de VGA: Lea el material del curso y muestre en una pantalla VGA un fondo de pantalla rojo y un rectángulo amarillo.

Parte 3. Diseño del juego SNAKE: La pantalla está en blanco al principio. Al presionar “S” en el teclado se inicia el juego SNAKE: un gráfico de serpiente en el borde izquierdo de la pantalla que automáticamente comienza a desplazarse hacia la derecha.

Este gráfico de desplazamiento responderá a las pulsaciones de teclas de flecha de la siguiente manera:

Orientación original	**Tecla**	**Cambio**
Horizontal	Flecha arriba	Voltear verticalmente y desplazarse hacia arriba
Horizontal	Flecha abajo	Voltear verticalmente y desplazarse hacia abajo
Horizontal	Flecha izquierda	Sin cambio
Horizontal	Flecha derecha	Sin cambio
Vertical	Flecha arriba	Sin cambio
Vertical	Flecha abajo	Sin cambio
Vertical	Flecha izquierda	Voltear horizontalmente y desplazarse a la derecha
Vertical	Flecha derecha	Voltear horizontalmente y desplazarse a la izquierda

Tenga en cuenta que la serpiente gira desde su cabeza delantera en lugar de la cola. Al presionar el botón “P” en el teclado se detiene el juego (se congela la pantalla) y al presionar “R” se reanuda el juego desde su estado de pausa. Al presionar “ESC” se sale del juego (deja la pantalla en blanco).

Otras propiedades del juego:

- Parámetros del juego

Color de fondo	Blanco
Color de la serpiente	Azul
Tamaño de la serpiente	40×10
Velocidad de la serpiente	50 pixeles por segundo

- En un juego terminado, asegúrate de que solo se vea la parte requerida de la serpiente. Por ejemplo, si la serpiente se mueve hacia la dirección correcta muy cerca del borde superior, y el usuario presiona una tecla de flecha hacia arriba, el juego terminará y solo una parte de la serpiente debe ser visible.

  - Cuando la serpiente toca cualquier borde de la pantalla, toda la pantalla debe congelarse y dejar de responder a las pulsaciones de las flechas R/P. Pero aún debería responder a las teclas "ESC" y "S".
  - Presionar "R" en el estado sin pausa no hace nada. Presionar "P" en el estado de pausa no hace nada. Presionando "ESC" en cualquier momento sale del juego (deja en blanco la pantalla) y al presionar "S" en cualquier momento se inicia el juego.

4. Flujo de diseño

- Cree un nuevo proyecto Vivado. Seleccione el dispositivo FPGA Artix-7.
- Escriba el código VHDL para el problema dado.
- Escriba el *test-bench* VHDL para probar el circuito y pruebe los casos mas representativos.
- Realice la *Functional Simulation* y la *Timing Simulation*.
- Asignación de entradas y salidas.
- Generar y Descargar el *bitstream* en el FPGA.

5. Presentación del informe

- Cuando el docente lo autorice, es decir luego de presentar la práctica funcionando, puede mandar un correo electrónico a: ahilariot@unsa.edu.pe con las siguientes especificaciones:
  - Asunto: Microelectrónica X - Informe de práctica 5, Grupo Y.
  - El correo electrónico debe contener: Informe en pdf (Escrito en LaTeX), y un enlace para un Drive personal donde estén los programas escritos en Vivado.

Práctica 6

(Fecha de entrega: M' 04-A: 30 de julio, M' 04-B: 05 de agosto, M' 17: 03 de agosto)

1. Objetivos

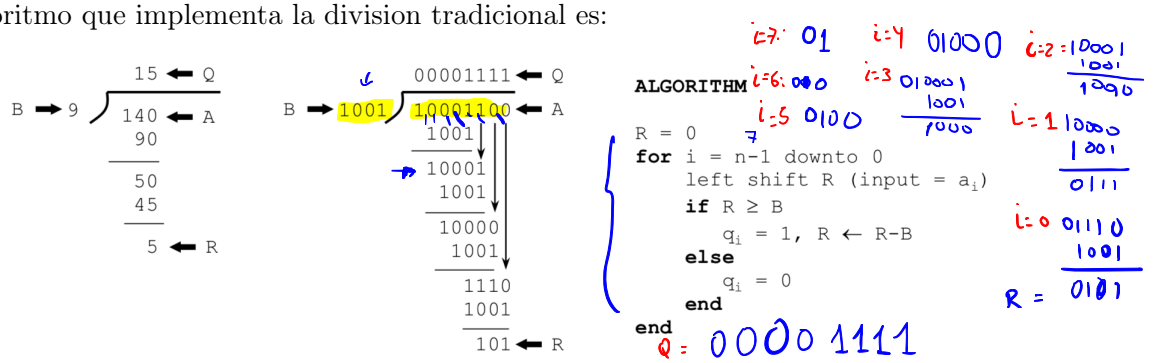
- Describir maquinas de estado finito en VHDL.
- Implementar un sistema digital: Unidad de ruta de datos y de control

2. Programación en VHDL

- Revise el material referente al curso.

3. Desarrollo de la práctica

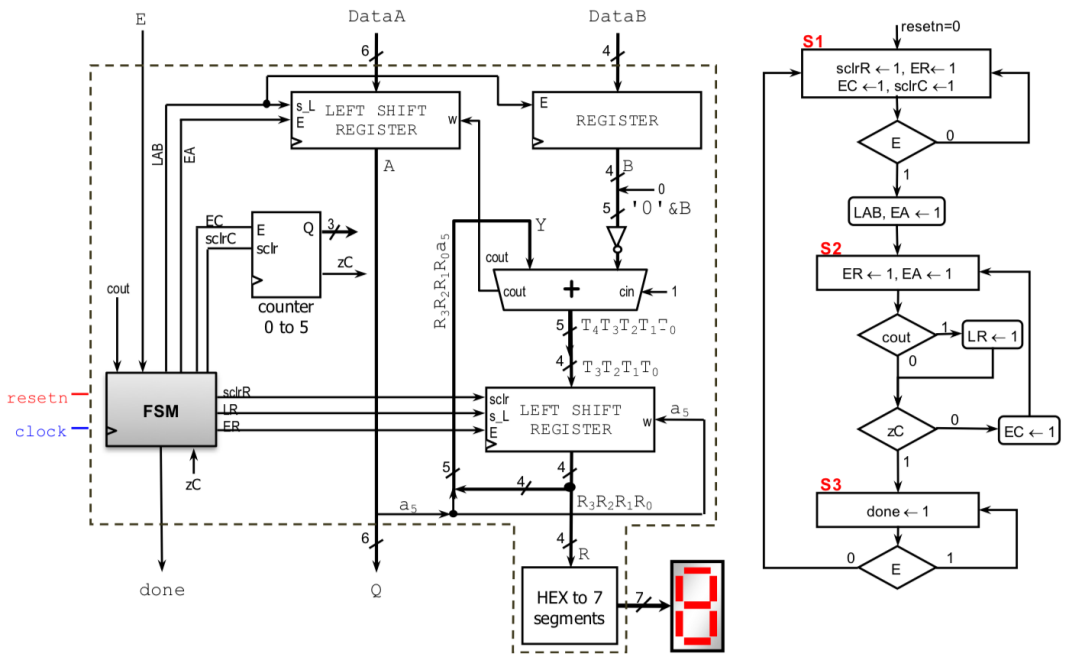
- **Problema:** Dado dos números unsigned A y B, diseñe un circuito que produce el cociente Q y el residuo R. El algoritmo que implementa la division tradicional es:



- Una arquitectura iterativa es mostrada en la siguiente Figura para A con 6 bits y B con 4 bits. El registro R guarda el residuo. En este diseño, una operación de division comienza cuando $E = 1$ (Donde los valores A y B son capturados). Entonces, a cada ciclo del reloj se ejecuta i) Desplazamiento en el siguiente bit de A o ii) Desplazamiento en el siguiente bit de A y restamos B. La señal *done* es creada para indicar que la operación ha sido completada y el resultado aparece en Q y R.
- El contador de modulo 6 incluye una entrada síncrona *sclr* que limpia el contador cuando $E = sclr = 1$ y una salida zC que es activada cuando el contador llega a 5.
- $Left - shit\ register$: Note que uno de los registros de desplazamiento incluye una entrada síncrona *sclr* que limpia la salida de los registros cuando $E = sclr = 1$.
- Cada componente secuencial tiene las entradas *resetn* y *clock*.
- Este circuito es un ejemplo de un sistema digital: incluye un circuito de ruta de datos y un circuito de control (FSM). El circuito de ruta de datos es hecho de componentes combinacionales y secuenciales.

4. Flujo de diseño

- Cree un nuevo proyecto Vivado. Seleccione el dispositivo FPGA Artix-7.
- Escriba el código VHDL para el circuito dado: Crear archivos separados para i) Contador modulo 6, ii) Registro de dezplazamiento, iii) Registro de desplazamiento con entrada *sclr*, iv) Registro, v) Sumador, vi) decodificador de 7 segmentos y vii) Archivo principal.



- Escriba el *test-bench* VHDL para probar los siguientes casos:

  - DataA = 010011 (19), DataB = 0100 (4)
  - DataA = 100111 (39), DataB = 1000 (8)
  - DataA = 110011 (51), DataB = 1110 (14)
  - DataA = 111100 (60), DataB = 1101 (13)
  - DataA = 011100 (28), DataB = 1001 (9)
  - DataA = 110101 (53), DataB = 0011 (3)

- Realice la *Functional Simulation* y la *Timing Simulation*.
- Asignación de entradas y salidas: Crear un archivo XDC. Use $SW0$ a $SW10$ para las entradas, CLK100MHZ para la entrada *clock*, el pulsador BTN_RES para *resetn*, un LED para *done*, seis LEDs para Q y un display de 7 segmentos para R.
- Generar y Descargar el *bitstream* en el FPGA.

5. Presentación del informe

- Cuando el docente lo autorice, es decir luego de presentar la práctica funcionando, puede mandar un correo electrónico a: ahilariot@unsa.edu.pe con las siguientes especificaciones:

  – Asunto: Microelectrónica X - Informe de práctica 6, Grupo Y.
  – El correo electrónico debe contener: Informe en pdf (Escrito en LaTeX), y un enlace para un Drive personal donde estén los programas escritos en Vivado.