The Abdus Salam
International Centre
for Theoretical Physics

smr2384

Joint ICTP-TWAS Latin-American Advanced Course on FPGA Design for Scientific Instrumentation

19 November 2012 – 07 December 2012

Havana – Cuba

LABORATORY EXERCISES II

SEQUENTIAL LOGIC DESCRIBED in VHDL

ICTP Multidisciplinary Laboratory
MLAB

Objetivo

- Utilización de las instrucciones secuenciales, concurrentes y paquetes aprendidos en clase.
- Interpretación de información especificada en hojas de datos o especificaciones de diseño.
- Familiarizarse con el kit de desarrollo *Nexys 2 Spartan-3E*. Uso de archivos de restricciones (*constraint file*) para asignación de pines de E/S, pulsadores, *switches* y *LEDs*.
- Configuración de la *Spartan-3E FPGA* del kit *Nexys 2*.
- Uso de código VHDL genérico para inferir una memoria de solo lectura tipo ROM.
- Simulación *post-place&route*.
- Comunicación entre la entidad descripta e implementada en la *Nexys 2* y una PC a través del puerto RS-232.

Parte A

a) Realizar la descripción en VHDL de un contador tipo BCD (0...9). El contador debe tener las siguientes E/S:

   - *Clk*: entrada de reloj.
   - *Reset*: entrada de *reset* asincrónico.
   - *Enable*: entrada de habilitación de cuenta. Cuando *enable* tiene valor lógico 1, el contador cuenta, de lo contrario mantiene su último valor.

   Verifique su funcionalidad con un *testbench* ejecutando simulación funcional y simulación *post-place&route*.

b) Utilizar el contador descripto en la Parte A pero agregar un pin de control de cuenta ascendente o descendente de acuerdo a la siguiente descripción:

   - *Up_Down*: cuando *up_down* tenga valor lógico '1', el contador debe contar en forma ascendente, de lo contario en forma descendente.
     Nota: esta señal también es controlada por la señal *enable*, de modo que si *enable* es inactivo la cuenta es mantenida.

Compruebe el correcto funcionamiento del componente mediante un *testbench* y su correspondiente simulación.

Nota: Recuerde utilizar el paquete ***ieee.numeric_std.all***

Parte B (opcional)

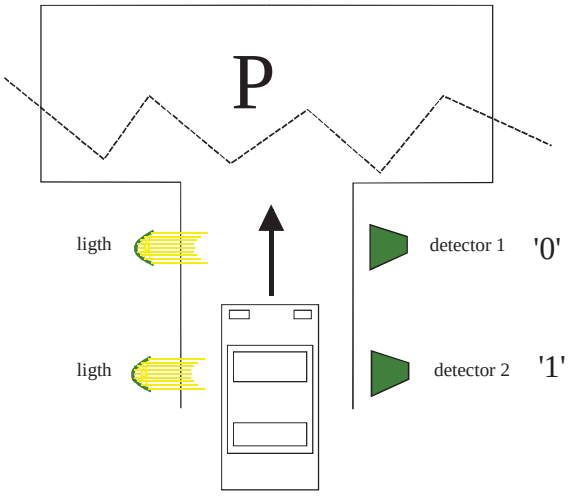
Realizar la descripción en VHDL de un contador tipo Hexadecimal (0...F). El contador debe tener las siguientes E/S:

- *Clk*: entrada de reloj.
- *Reset*: entrada de *reset* asincrónico.
- *Enable*: entrada de habilitación de cuenta. Cuando *enable* tiene valor lógico 1, el contador cuenta, de lo contrario mantiene su último valor.

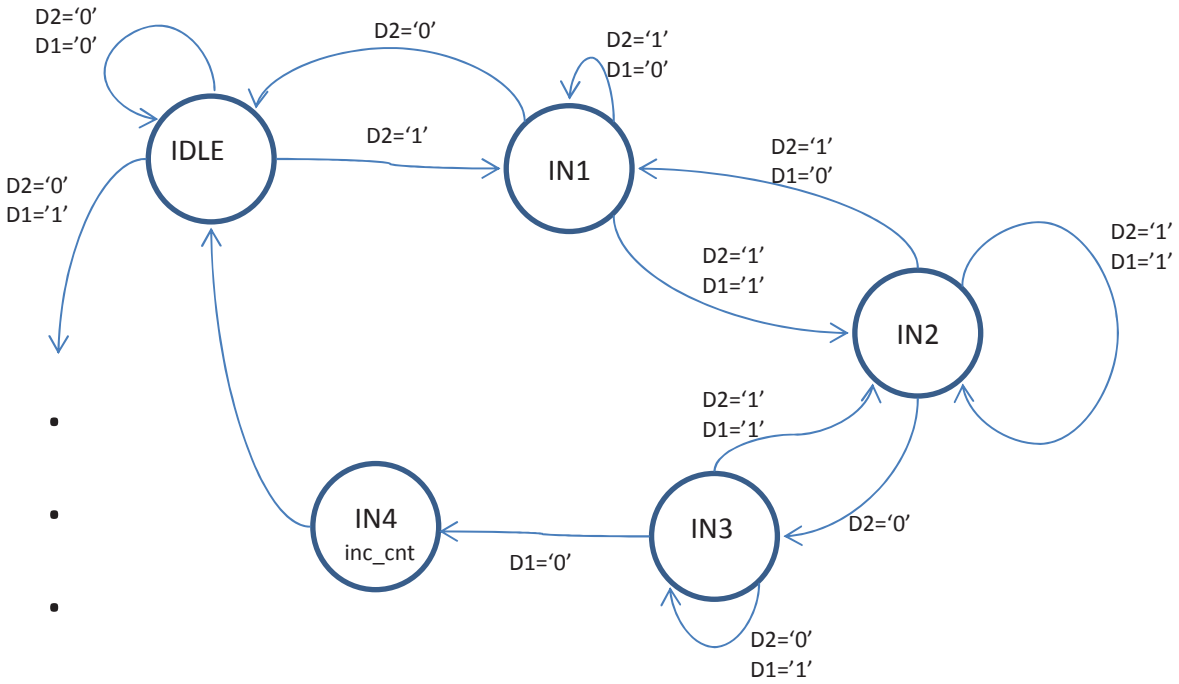
Compruebe el correcto funcionamiento del componente mediante un *testbench* y su correspondiente simulación.

Parte C

El parque de estacionamiento mostrado en la siguiente figura tiene espacio para 10 autos y un sólo un ingreso, por lo tanto, solo un auto a la vez puede entrar o salir del parque de estacionamiento. Existen dos sensores, a una distancia de un metro uno del otro, para detectar si el auto está entrando o saliendo; y un semáforo indicando si el parque está completo (FULL) o no (FREE).



a) Realice y simule un diseño en VHDL para controlar el semáforo de acuerdo a la cantidad de autos presentes en el parque de estacionamiento (ver ejemplo de máquina de estado describiendo la entrada de un auto al estacionamiento).
b) Implemente el diseño en *el board Nexys 2* (opcional).



Ej: Vehículo entrando al parque de estacionamiento

Parte D

El *board Nexys 2* tiene un oscilador de 50MHz como entrada de reloj a la *FPGA Spartan-3E*. Realice un divisor de frecuencias de modo que teniendo como entrada los 50MHz obtenga como frecuencia de salida 1Hz.
Escriba el código usando constante el valor máximo del divisor. Asocie la salida del divisor con uno de los *LEDs* disponible en el *board* (ver página 5 del *Reference Manual de la Nexys 2*). Compruebe su funcionamiento con un *testbench*. Analice que pasa con el tiempo real y el tiempo de simulación cuando simule su circuito. Escriba el correspondiente *constraint file*. Genere el *bitstream* y programe la FPGA. Verifique el correcto funcionamiento.

Parte E

Ahora está en condiciones de completar la Parte D del Laboratorio de Lógica Combinacional. Reemplace las 'cajas negras' de los contadores BCD por contadores tipo BCD o Hexadecimal descripto en los puntos anteriores.

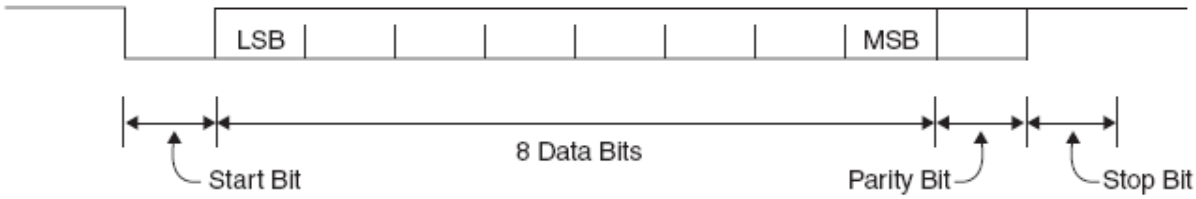
1) Describa el módulo *Top Level* con todos los componentes necesarios, recuerde que este módulo debe contener sólo los componentes que se instancian, no colocar ningún tipo de lógica en este módulo. Utilice una entrada común de *reset* para inicializar los contadores, y una entrada de *up/down* para definir el modo de cuenta de cada contador. Para estas entradas utilice los *switches* disponibles en el *board*, y para la visualización de la cuenta utilice el *display* 7 segmentos.
2) Compruebe el sistema descripto mediante la adecuada simulación del mismo. Opcionalmente: investigue el *timing report* y descubra cuál es el camino más crítico de este diseño. Realice un dibujo esquemático del mismo. Cada componente de este esquemático debe tener su valor de retardo.
3) Escriba un *contraint file* (.ucf) donde especifique la frecuencia de funcionamiento de su diseño. Asigne los *I/O pads* respectivos. Compruebe la correcta asignación de las señales de entrada/salida con los respectivos *I/O pads* de la FPGA revisando el *PAD report*. Investigue el *report* de *timing* estático generado por el proceso *Generate Post-Place&Route Static Timing*. Detalle en el informe sus conclusiones.
4) Genere el archivo de configuración de la FPGA.
5) Configure la *Spartan-3E FPGA de la Nexys board*, y verifique el correcto funcionamiento de su diseño.

Parte F

a) Realizar la descripción en VHDL de un sistema digital que transmita datos en forma serie según el protocolo RS-232.

El dato a transmitir debe tener el formato detallado en la figura a continuación, con la siguiente configuración:

- 1 bit de *start*
- 8 bits de datos
- Paridad par
- 1 bit de *stop*
- Frecuencia de recepción/transmisión por defecto es de 9600 *bauds*, con la posibilidad de cambiar la velocidad de Rx/Tx a 4800, 38400, 115200 *bauds*, seleccionadas por las llaves disponibles en el *board*.



RS-232 formato de transmisión

Los datos a transmitir por RS-232 deben ser leídos de una memoria ROM que debe ser inferida desde el código VHDL escrito en forma genérica.

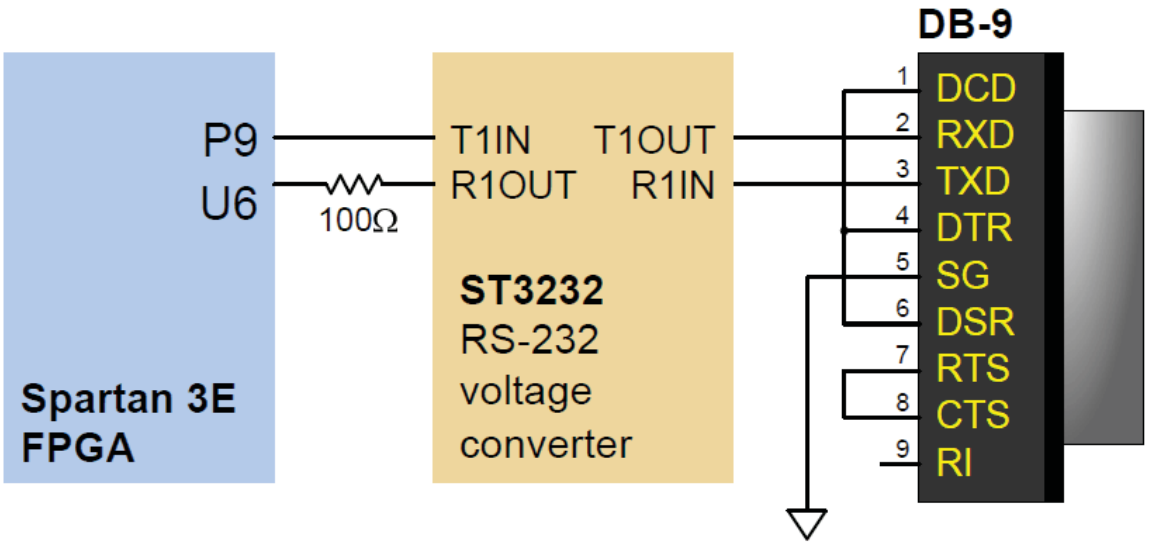
La memoria debe ser implementada usando los bloques de memoria RAM (BRAM) disponibles en la *Spartan-3E*.

Los datos almacenados en la memoria son 4 mensajes diferentes (use la función de conversión *to_unsigned*):

1) "Laboratorio: Completado por <Nombre_del_Alumno>"
2) "ICTP 2012 Cuba"
3) "<frase_a_elección_del_alumno>"
4) "<frase_a_elección_del_alumno>"

El mensaje a transmitir será seleccionado por los pulsadores BTN3, BTN2, BTN1, BTN0 del *board*. Así, cuando se presione el pulsador BTN3 debe transmitir el mensaje 4, BTN2 mensaje 3, BTN1 mensaje 2 y BTN0 mensaje 1.

b) Implemente el diseño en la *FPGA Spartan-3E del board Nexys 2*. Compruebe el correcto funcionamiento con un programa tipo Hyper-Terminal. Utilice la siguiente figura como referencia para las entradas y salidas del sistema:



Requerimientos

- A fines de familiarizarse con las diferentes herramientas de VHDL el proyecto deberá tener por lo menos un paquete para:
  a) La definición de constantes y componentes del sistema.
  b) Definir una función para el cálculo del bit de paridad.
- El código del componente de mayor jerarquía (*top level*) debe poseer **sólo** instrucciones de instanciación (**no lógica)**.
- Escriba un *contraint file* (.ucf) donde especifique la frecuencia de funcionamiento de su diseño. Asigne los *I/O pads* respectivos.
- Constatar con el *PAD report* la correcta asignación de I/Os, teniendo en cuenta la asignación de pines I/O realizada en la *Nexys 2 board*.
- Verifique que no haya problemas de *timing*, al revisar el informe de *timing* elaborado por el ISE. Recuerde detallar en el *constraint file* la frecuencia de trabajo de su sistema. Encuentre cual es el camino crítico (*critical path*) y dibuje el correspondiente esquemático detallando los distinto retardos.
- Especifique en un *constraint file* (.ucf) los pines de I/O con sus respectivos estándares I/O, *slew y drive*, la(s) frecuencias de funcionamiento, etc.
- Compruebe el sistema descripto mediante la adecuada simulación del mismo. Genere el modelo de simulación *post-place&route* (proceso *Generate Post-Place&Route Simulation Model*). Ejecute una simulación *post-place&route*.

Parte G (opcional)

Descripción

Realizar la descripción en VHDL de un sistema digital que reciba datos en forma serie según el protocolo RS-232. Implementar el diseño en el *board Nexys 2*.
Se puede seguir los mismos lineamientos detallados para el Tx RS-232, sin embargo, como el protocolo RS-232 es asincrónico, el receptor conoce solo la velocidad de transmisión y no posee información del reloj asociado al dato a recibir.
Una técnica de diseño para este tipo de problemas es conocida como sobre-muestreo mediante el cual el receptor obtiene el bit recibido en un punto adecuado a fin de recibir el bit correcto. Este esquema utiliza una señal de muestreo de alta frecuencia para estimar el punto medio del bit a recuperar. Usualmente un muestreo de 16 veces la velocidad de transmisión (*baud rate*).
Para recuperar el dato de entrada se pueden seguir los siguientes pasos:

1) Cuando comienza el bit de *start*, es decir la línea de dato pasa a '0', se inicia el contador de muestras.
2) Cuando el contador alcanza 7, se reinicia el contador. En este punto, la señal de entrada alcanza la mitad del bit de *start*.
3) Cuando el contador alcanza 15, se reinicia al contador. En este punto, la señal de entrada alcanza la mitad del primer bit recibido. Entonces este dato, '0' o '1', debería ser capturado en un registro.
4) Repita el paso anterior 7 veces más para recuperar los otros 7 bits.
5) Repita el paso 3 una vez más para recuperar el bit de paridad (si es que es transmitido).
6) Repita el paso 3, esta vez para verificar que el bit recibido es el bit de *stop* y debería tener valor lógico '1'.