Guía de ejercicios
Verilog

Materia: Programación de softcores en FPGAs
Programa de Profesoras/es Visitantes

Material disponible en: `https://github.com/fokerman/fpgaSoftcoreProgrammingCourse`

Los siguientes ejercicios están pensados como referencia para practicar la implementación de módulos en `verilog`. Se recomienda seguir los ejercicios en orden implementando los módulos pedidos y un *testbench* para probar su correcto funcionamiento. Adicionalmente, para los casos más simples pueden agregar marcas de tiempo en el código para simular diferentes temporizaciones.

Ejercicio 1
Módulos combinacionales básicos:

1. Implementar un multiplexor de 16 entradas y 4 líneas de control.
2. Usando la implementación anterior construir un multiplexor de 32 entradas y 5 líneas de control.
3. Construir un circuito verificador de paridad de 8 bits.

Ejercicio 2
Módulos aritméticos básicos:

1. Armar el circuito de un sumador simple usando compuertas como primitivas.
2. Usando el sumador anterior armar un sumador completo.
3. Usando los circuitos anteriores armar un sumador de 8 bits.
4. Construir un sumador de 8 bits usando descripción de comportamiento. Comprar el resultado con la implementación anterior.
5. Extender el sumador de 8 bits anterior a 32 bits. Comparar el resultado entre las dos implementaciones.
6. Construir un circuito multiplicador con entradas de 8 bits y salida de 16 bits usando múltiples sumadores.
7. Construir un circuito multiplicador con entradas de 8 bits y salida de 16 bits usando registros de desplazamiento.

Ejercicio 3
Módulos aritméticos generales:

1. Implementar una ALU de 8 bits que permita realizar las siguientes operaciones: suma, resta, not and, or, xor, shift lógico y aritmético a derecha e izquierda.
2. Modificar el circuito anterior para agregar las Flags: Carry, Zero, Overflow, Negative, Parity.
3. Usando el código anterior proponer una ALU vectorial, que soporte las mismas operaciones pero sobre varios datos al mismo tiempo. La misma debe soportar cuatro datos de 8 bits, dos datos de 16 bits o un dato de 32 bits.

Ejercicio 4
Módulos contadores:

1. Construir un circuito contador de 8 bits, tal que por cada ciclo de clock aumente en uno el valor del contador.
2. Modificar el circuito anterior para extenderlo a 32 bits y que aumente de a 3 unidades.
3. Modificar el circuito anterior para agregarle una señal de reset asincrónica y un valor de inicio arbitrario.
4. Implementar un contador gray de 5 bits.

Ejercicio 5
Módulos registros:

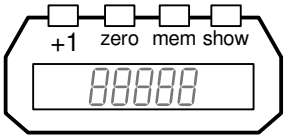
1. Implementar un registro de 8 bits con `reset` sincrónico y salida controlada por un buffer de tres estados.
2. Modificar el circuito anterior y agregarle una señal de set sincrónico.
3. Modificar el circuito anterior para que las señales de `reset` y set sean asincrónicas.
4. Extender el diseño para soportar un registro de 32 bits.
5. Agregar una señal para extender el signo de los 16 bits menos significativos del registro.

Ejercicio 6
Módulos memorias:

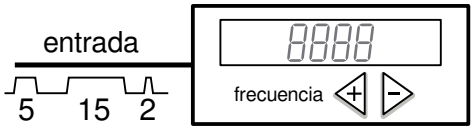
1. Implementar un circuito que lea una memoria de 32 palabras de 8 bytes y exponga byte a byte en un bus de 8 bits. La memoria debe estar cargada inicialmente con datos leídos directamente del código. Proponer luego una variante que lea los datos de un archivo.
2. Modificar el circuito anterior para exponer los bytes de forma serial desde el más significativo al menos significativo.
3. Modificar el circuito y agregarle una señal de control para indicar el orden en que deben ser expuestos los bits de cada byte.
4. Modificar el circuito anterior para agregar una nueva señal de control que indique el orden en que se leen los bytes, desde la dirección 0 a 31 o a la inversa.
5. Modificar el circuito anterior para agregar una señal de `reset`.
6. Modificar el circuito anterior para agregar un conjunto de señales que permitan setear el primer byte desde donde se va a leer.
7. Modificar el circuito anterior para soportar que la indicación del primer byte que leer se tome del forma serial.

Ejercicio 7
Módulos integradores:

1. Implementar un contador manual en decimal. Este dispositivo posee 4 botones denominados “+1”, “zero”, “mem” y “show”. Además de una salida de 5 dígitos decimales, es decir 5 valores en BCD, que corresponden a 20 bits. El comportamiento de los botones será el siguiente: “+1”, incrementa en 1 el valor del contador; “zero”, resetea el contador a cero; “mem”, memoriza el valor actual; y “show”, muestra el último valor memorizado en el display mientras se lo este presionando.



2. Implementar un dispositivo que detecta el tamaño de los pulsos que llegan a la entrada y los expone el tamaño de cada uno en un display de 16 bits. El dispositivo posee dos botones, uno para aumentar y otro para disminuir la frecuencia de muestreo de la señal de entrada.



3. Implementar el juego `Simon`, este 4 botones y 4 leds como salidas en cada botón; las salidas reproducen una secuencia de encendido, donde en cada turno del juego una sola salida se enciende. El jugador debe seleccionar el mismo botón que se encendido para pasar de turno. En el próximo turno, se agrega un nuevo valor a la secuencia que el jugador debe reproducir. Si el jugador no presiona el botón correcto, el jugador pierde y los 4 leds se deben quedar titilando hasta que se presiona el botón de reset R, comenzando nuevamente el juego.