UNIVERSIDAD DE COSTA RICA

Plan de Pruebas para el Proyecto II

IE0523 CIRCUITOS DIGITALES II

Freddy Zúñiga Cerdas
A45967

Alexander Calderón Torres
B61325

Profesor

JORGE SOTO

June 29, 2021

1 Plan de pruebas

1.1 Pruebas de la memoria

Para la prueba de la memoria se debe crear un probador que envíe entradas de datos random y acto seguido se debe levantar la orden de write para empezar a llenar la memoria, para esto el addr_write irá aumentando. Para comprobar que funciona, luego de la primera escritura se irá leyendo los datos ya guardados y aumentando el addr_read.

1.2 Pruebas del FIFO

Cuando se hable de un **campo borrado** esto hará referencia a que el puntero read ya se desplazó de esta posición a la siguiente, cuando el puntero read se desplaza de una posición a otra lo que queda es un campo con un dato que básicamente es basura, al que se le puede escribir encima sin provocar un error.

Cuando se hable de un **campo vacío** en la memoria del FIFO, nos referimos a que el dato en la posición o bien no ha sido aún escrito o bien es un **campo borrado**.

Cuando se hable de un **campo ocupado** nos referimos a que se ha escrito un dato en dicho campo por lo que el puntero de write se ha desplazado a la siguiente posición.

1.2.1 Prueba de Funcionamiento Normal

Para esta prueba simplemente se da unas cuantas órdenes de write antes de empezar a dar órdenes de read, una vez que se han ocupado unos cuantos espacios del FIFO se darán órdenes write/read para mantener el flop en un nivel estable.

1.2.2 Prueba de Errores

Para esta prueba se debe producir los dos errores posibles del FIFO, error de read y de write, el error de read se da cuando se trata de leer un dato de un **campo vacío**, mientras que el error de write se da cuando se intenta escribir un dato en un **campo ocupado**. Cuando el FIFO se integre con el sistema no debería ocurrir ninguno de estos errores.

1.2.3 Prueba de Full Constante

Para esta prueba se llena el FIFO y una vez que esté lleno se envían órdenes write/read, el FIFO entonces debe mantenerse lleno y estar escribiendo y leyendo datos sin que aparezca un error. Este sería el caso extremo del funcionamiento normal.

1.2.4 Prueba de Almost Empty y Almost Full

Para esta prueba se deben cambiar los parámetros HIGH y LOW del probador a los parámetros deseados y estos funcionan así: si el FIFO tiene ocupados LOW campos o menos de LOW, se levanta el flag fifo_empty_out lo que claramente indica que el FIFO esta muy cerca de estar vacío. Si el FIFO tiene ocupado HIGH campos o más de HIGH campos se levanta el flag fifo_full_out lo que indica que el FIFO está casi lleno.

Si se desea que los límites del FIFO sean todos los campos vacíos y todos los campos llenos entonces se debe escoger los siguientes valores para los parámetros del probador: HIGH = LOW = 0.

La prueba propiamente dicha es simplemente llenar el FIFO Y luego vaciarlo sin incurrir en un error y ver que sucede con las flags.

1.3 Prueba de Arbitro

Estas son pruebas tentativas, posiblemente una vez creado el módulo se pueden agregar pruebas más espefícas que complementen las que se citarán a continuación.

1.3.1 FIFOS de entrada empty

Se harán varias pruebas teniendo algunas combinaciones de FIFOS empty para ver como se comporta el arbitro y se dejará trabajar hasta que todos los FIFOS de cada una de las pruebas estén vacíos. Así comprobaremos si el árbitro para de hacer pop como se supone que debe hacerlo.

1.3.2 FIFOS de salida full

La prueba consiste en dejar que se llene el primer FIFO, y ver el funcionamiento del árbitro, que debe deterner el push, y repetir la prueba para los restantes FIFOS, como se recomendó se llevarán los 4 FIFO's hasta que falte un dato para estar almost full y se harán convenientemente los pops del probador para tener un resultado en secuencia.

1.4 Pruebas de contadores

Esta será una prueba sencilla y consta de hacer pasar datos a través de los FIFO's de salida e ir llevando un conteo de cuantos paquetes salen de estos flops, se da por válida la prueba si el conteo de datos de salida es igual al conteo de datos de entrada.

1.5 Pruebas del bloque completo

Al diseño final y completamente integrado de la capa de transacción se le aplicará el siguiente conjunto de pruebas, dado en el enunciado del proyecto:

1. Saque el bloque de `RESET`, manteniendo el estado de `INIT` (señal `init`).
2. Modifique 2 veces los umbrales altos y bajos de los *FIFOs* (son el mismo umbral bajo y alto para todos). Libere la señal `init`.
3. Provoque un *Almost Full* en todos los *FIFOs* de **salida**, el árbitro no le permitirá hacerlo de forma simultánea. Desde el probador, haga la menor cantidad de *POPs*. Verifique que las palabras que salieron son las mismas que entraron y que salieron por la salida correcta en la prioridad correcta.
4. Provoque un *Almost Full* en todos los *FIFOs* de **entrada**. Luego, usando *POPs* del probador, deje todos los *FIFOs* vacíos. Verifique que las palabras que salieron son las mismas que entraron y que salieron por la **salida correcta en la prioridad correcta**.
5. Lea los contadores de palabras.
6. Envíe 16 palabras, 4 a cada *FIFO* de entrada, y cada set de 4 palabras por *FIFO* de entrada estén con destino a cada *FIFO* de salida (las $4 * 4 = 16$ combinaciones posibles). Deje todos los *FIFOs* vacíos. Verifique que las palabras que salieron son las mismas que entraron y que salieron por la salida correcta en la prioridad correcta.
7. Lea los contadores de palabras y valide un aumento de 4 palabras por contador.