Para la fase 3 de nuestro proyecto se nos pidió que implementáramos los módulos correspondientes para poder realizar instrucciones de tipo jump. Las modificaciones que tuvimos que hacer en el código de nuestro módulo completo fue agregar un multiplexor, al que llamé multiplexor 5, otro módulo de shift left y las salidas correspondientes de la unidad de control, que fue sólo una salida llamada jump.

Tabla

Descripción generada automáticamente con confianza media

Figura 1

En la figura 1 podemos ver el fragmento de código que se le agregó al módulo unidad de control para que se lleven a cabo las instrucciones de tipo jump, se agregó un nuevo caso para la variable opcode, esto quiere decir que al recibir el opcode “000010”, la unidad de control enviará las señales que vemos en la figura 1 y se realizará el jump. Cabe aclarar que aunque aquí no se vea, se añadió un output llamado *jump* al módulo.

El nuevo módulo de shift left fue agregado después del buffer 1, recibe 26 bits que son los primeros bits de la instrucción (del bit 25 al bit 0), y la salida del shift left es de 28 bits que entran al buffer 2, en el buffer 2 se agregó una entrada para estos 28 bits, otra entrada para la señal de jump y una salida de 32 bits, esta salida esta dada por los 28 bits del shift left concatenados con 4 bits (del bit 31 al bit 28) de la salida del sumador 1. Esta salida de 32 bits es la dirección a la que se hará el jump. La señal de jump entra desde la unidad de control y sale hacia el próximo buffer.

En el buffer 3 también se agregaron dos entradas, una de 32 bits y otra para la señal de jump, también dos pares de salidas para la dirección de jump y la señal de jump, que son de 32 bits y 1 bit respectivamente. Estas dos salidas ya no pasan al próximo buffer, las dos se van al multiplexor 5.

En el módulo multiplexor 5 entra la dirección de jump y entra también el valor de salida del multiplexor 4, es decir, el multiplexor 4 ya no está conectado directamente al PC, ahora está conectado el multiplexor 5, entonces si la señal de jump está “encendida”, sale la dirección del jump (y entra al PC) y si está en 0, sale la dirección de la próxima instrucción (que viene de mux 4).

Esto es básicamente, una breve descripción de los cambios que se tuvieron que hacer en el módulo de la fase 2 para que se pudieran realizar instrucciones de tipo jump.

Para comprobar que funciona, probamos el módulo con una serie de instrucciones que son las instrucciones que se requieren para realizar la sucesión de Fibonacci.

Imagen que contiene Texto

Descripción generada automáticamente

En la figura 2 podemos ver las instrucciones que se probaron, primero tenemos una suma de los registros 0 y 1 que se almacena en el registro 2, la próxima instrucción es un addi, pero lo único que hace es guardar el valor del registro 1 en el registro 0, la próxima también es un addí que guarda el valor del registro 2 en el registro 1, la próxima instrucción aumenta en 1 el registro que funciona como contador, la próxima es un beq que compara el registro contador (que es el 9) con el registro 10, cuando el valor del registro 9 se haya aumentado 9 veces se saltará una instrucción y por último tenemos la operación de jump, que regresa a la dirección 0.

Figura 2

Entonces los resultados esperados son los siguientes, que podemos ver en nuestra tabla 1.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Vuelta | Operación 1  add $0, $1, $2 | Operación 2  addi $1, $0, #0 | Operación 3  addi $2, $1, #0 | Operación 4  addi $9, $9, #1 | Operación 5  beq $9, $10, #1 | Operación 6  Jump #0 |
| 1 | $2 = 1 | $0 = 1 | $1 = 1 | $9 = 2 | No | Sí |
| 2 | $2 = 2 | $0 = 1 | $1 = 2 | $9 = 3 | No | Sí |
| 3 | $2 = 3 | $0 = 2 | $1 = 3 | $9 = 4 | No | Sí |
| 4 | $2 = 5 | $0 = 3 | $1 = 5 | $9 = 5 | No | Sí |
| 5 | $2 = 8 | $0 = 5 | $1 = 8 | $9 = 6 | no | Sí |
| 6 | $2 = 13 | $0 = 8 | $1 = 13 | $9 = 7 | No | Sí |
| 7 | $2 = 21 | $0 = 13 | $1 = 21 | $9 = 8 | No | Sí |
| 8 | $2 = 34 | $0 = 21 | $1 = 34 | $9 = 9 | No | Sí |
| 9 | $2 = 55 | $0 = 34 | $1 = 55 | $9 = 10 | Sí | No |

Tabla 1

Imagen que contiene Gráfico

Descripción generada automáticamente

Figura 3

Al correrlo por primera vez sólo 400 nanosegundos, podemos ver los valores iniciales del banco de registros en la figura 3. Como los valores que nos interesan se encuentran entre los registros 0 y 10, sólo muestro los registros del 0 al 10.

Texto

Descripción generada automáticamente con confianza media

Figura 4

En la figura 4 podemos ver que después de correrlo una vez más (400 nanosegundos), se guarda la suma de los registros 0 y 1 en el registro 2.

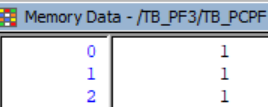


Figura 5

En la figura 5 podemos ver que también ya se cambiaron los valores de los registros 0 y 1, el registro 0 tomo el valor que tenía el registro 1 y el registro 1 tomó el valor del registro 2, esto sucede al correrlo otras veces más.

Tabla

Descripción generada automáticamente con confianza media

Figura 6

En la figura 6 podemos ver que al seguir corriendo el código se aumenta en 1 el valor del registro 9, hasta este momento no se ha realizado ninguna operación de jump o beq, en las próximas figuras podremos ver que la instrucción de jump se lleva a cabo exitosamente y se vuelven a sumar los valores de los registros 0 y 1.

Tabla

Descripción generada automáticamente con confianza baja

Figura 7

En la figura 7 podemos ver que ya se realizó el jump y se volvieron a sumar los registros 0 y 1.

Imagen que contiene Tabla

Descripción generada automáticamente

Figura 8

En la figura 8 ya tenemos los nuevos valores de los registros 0 y 1.

Tabla

Descripción generada automáticamente

Figura 9

En la figura 9 se vuelve a aumentar el valor del registro 9 que nos va a servir como contador.

Tabla

Descripción generada automáticamente con confianza media

Figura 10

En la figura 10 se vuelve a saltar hasta la instrucción 0 y se vuelven a sumar los registros 0 y 1.

Voy a dejar hasta aquí la demostración para no extenderme tanto en el espacio y en el tiempo.

**Conclusión**

Fue una fase fácil y sencilla a comparación de la fase 2 donde tuvimos que añadir 4 módulos e interconectar todo otra vez. Esta fase fue sencilla porque solo fueron dos módulos y no hubo necesidad de añadir mucho código, así que fue fácil. Con estas nuevas instrucciones que añadimos podremos realizar nuestro algoritmo planeado.