Guía de Problemas

VHDL Avanzado

Elías Todorovich – Octubre 2012

DESCRIPCIÓN DEL DISEÑO 2

EJERCICIOS 4

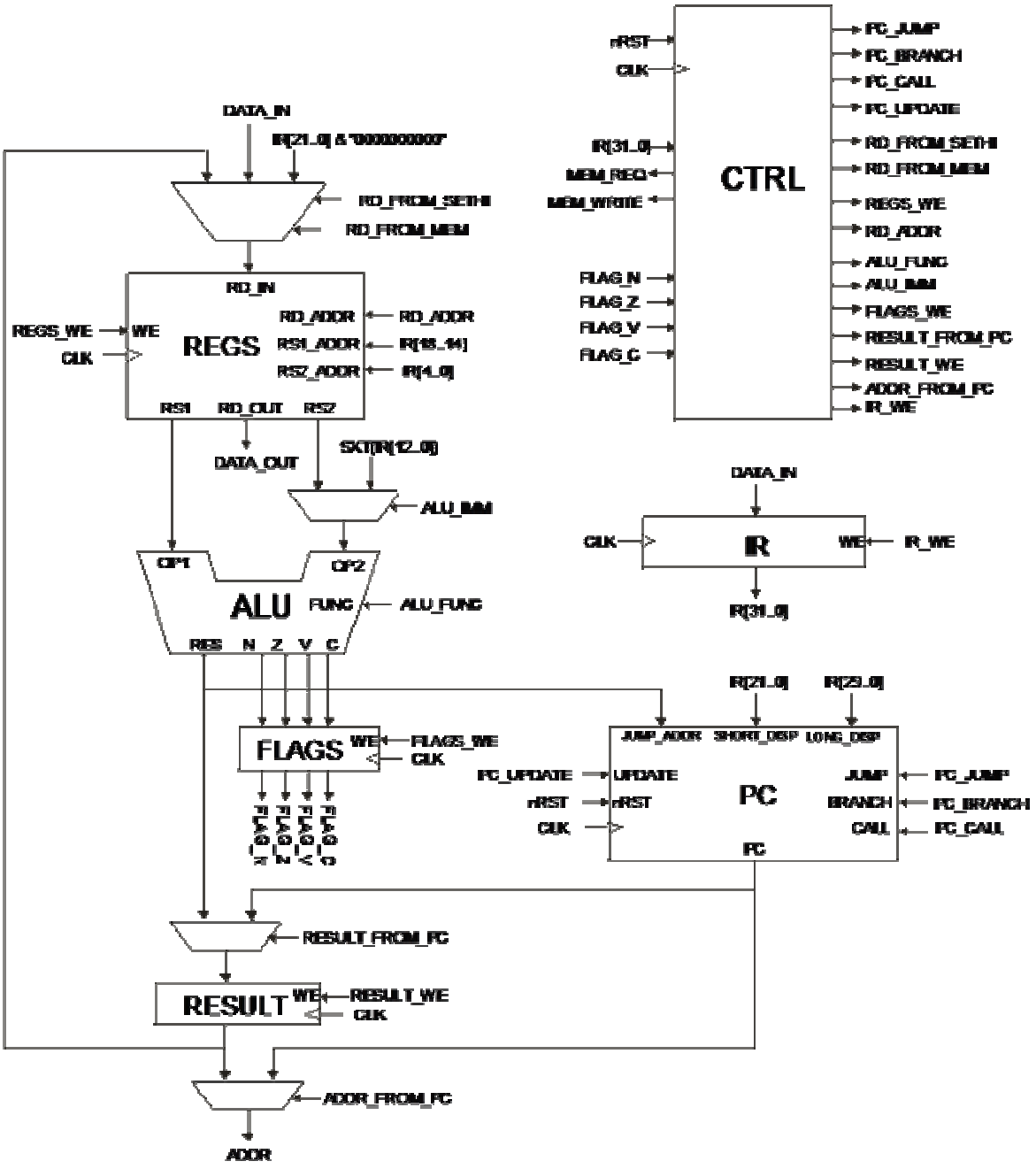
A. MEJORAR LA LEGIBILIDAD: CONSTANTES Y ALIAS 4
B. MEJORAR LA LEGIBILIDAD: RECORDS 5
C. MODULARIZACIÓN: FUNCIONES Y PROCEDIMIENTOS 5
D. PARÁMETROS GENÉRICOS 6

Descripción del Diseño

Se trata de un microprocesador que implementa un subconjunto muy reducido de la arquitectura SPARC llamado ARC. Las características principales que presentará este microprocesador son:

- Arquitectura de 32 bits: Los datos que va a manejar el microprocesador tendrán una longitud de 32 bits.
- Arquitectura LOAD-STORE. Los accesos a la memoria de datos se harán sólo con estas instrucciones.
- 32 registros internos de propósito general, cada uno de 32 bits, más un contador de programa (PC) y un registro de próxima instrucción (IR), ambos de 32 bits.
- Registro de estado del procesador con las 4 banderas aritméticas para realizar los saltos condicionales (Z: cero, N: negativo, C: acarreo, V: desbordamiento).
- 15 instrucciones básicas cuyos mnemónicos son:
    - **ld**, **st**: acceso a memoria
    - **sethi**: Carga inmediata los 22 bits más significativos a un registro
    - **be**, **bcs**, **bneg**, **bvs**, **ba**: Saltos condicionales y salto incondicional (ba)
    - **call**, **jmpl**: llamada y retorno de subrutina
    - **addcc**: suma
    - **andcc**, **orcc**, **orncc**: operaciones lógicas, afectando los flags
    - **srl**: desplazamiento lógico a derecha

ARC es un diseño jerárquico de manera que el módulo ARC instancia otros que lo componen:



Para diseñar ARC debe tenerse presente el formato de las instrucciones:

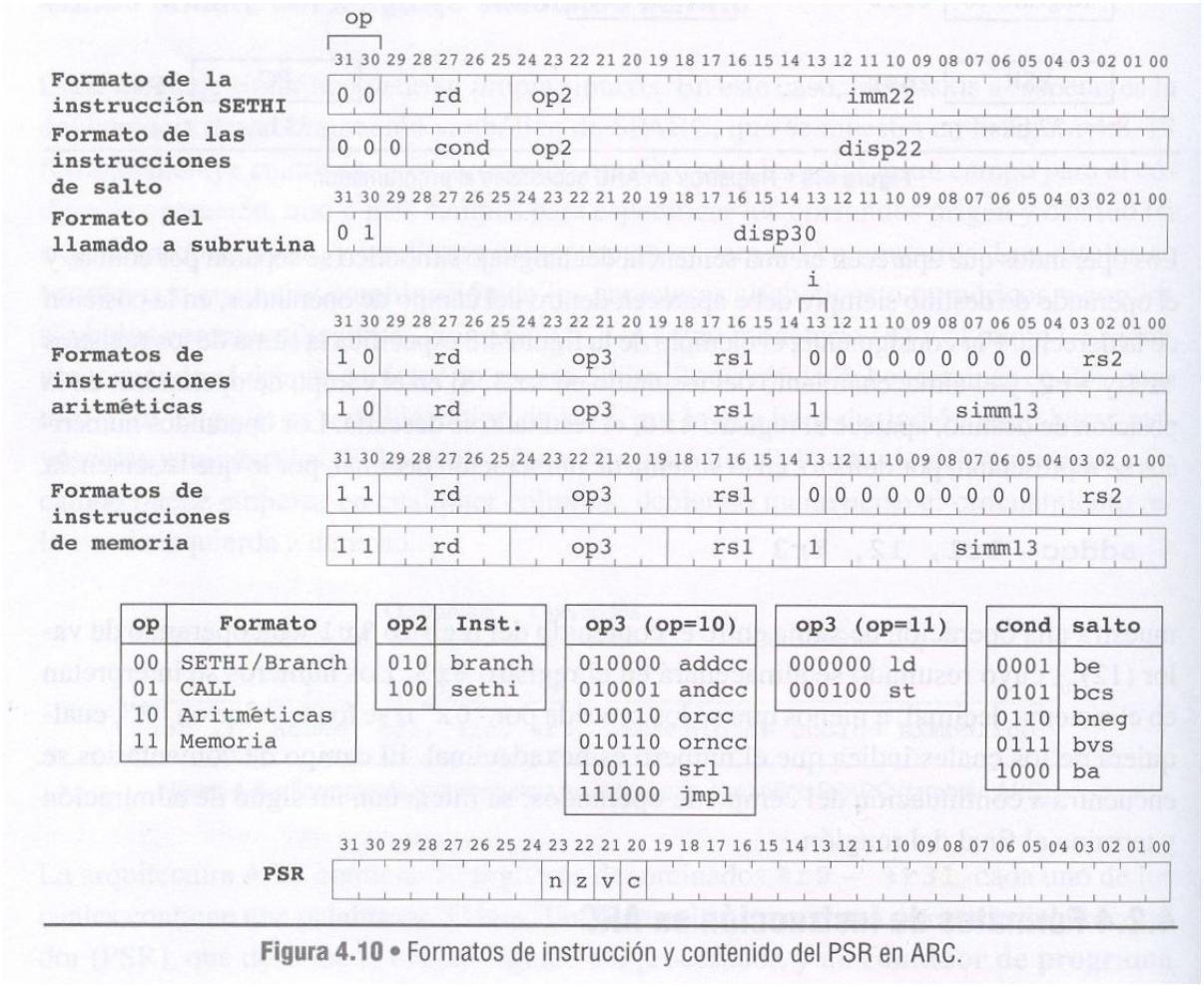


Figura 4.10 • Formatos de instrucción y contenido del PSR en ARC.

Ejercicios

A. Mejorar la legibilidad: Constantes y Alias

I) La unidad de control (CTRL.VHD) en su versión primera, muestra un número de literales que representan *arrays* de bits. Utilizar constantes y reemplazar los literales.

II) La unidad de control también utiliza partes de vectores de manera que se forman expresiones algo complejas. Definir nombres alternativos (Alias) para estas referencias complejas y reemplazarlos en el código.

III) Verificar si la calidad del código ha mejorado. Simular el modelo nuevamente con el testbench dado para comprobar que no se cometió ningún error.

IV) Intentar sintetizar el modelo. ¿Admite la herramienta de síntesis el uso de alias?

B. Mejorar la legibilidad: Records

La interfaz de la unidad de control presenta una cantidad de puertos para señales control del diseño del microprocesador. Algunos de estos puertos podrían agruparse en *records*:

- Flags (FLAG_X)
- Acceso a memoria (MEM_XX)
- Control de escritura de registros (XX_WE)
- Control del contador de programa (PC_XX)

I) Crear un *package* y definir los *records* correspondientes.

II) Utilizar los records para simplificar las interfaces y conexiones del diseño.

III) Verificar si la calidad del código ha mejorado. Simular el modelo nuevamente con el testbench dado para comprobar que no se cometió ningún error.

IV) Intentar sintetizar el modelo. ¿Admite la herramienta de síntesis el uso de records?

C. Modularización: Funciones y Procedimientos

La unidad de control debe generar la señal de salida PC_BRANCH que se debe activar cuando haya una instrucción de branch que sea efectiva. Esto es, el valor de los *flags* se corresponden con el esperado por la instrucción en particular, p.e. para **be** esta señal (PC_BRANCH) se activará solo si **FLAG_Z** vale uno.
PC_Branch podría calcularse mediante:

```
PC_BRANCH <= '1' when (OP=I_SEB and OP2=CO_BR and
BR_EFECTIVO='1') else '0';
```

donde BR_EFECTIVO podría ser:

```
BR_EFECTIVO <= '1' when (COND=BE   and FLAG_Z='1') or
                        (COND=BCS  and FLAG_C='1') or
                        (COND=BNEG and FLAG_N='1') or
                        (COND=BVS  and FLAG_V='1') or
                        COND=BA
     else '0';
```

Así como se indica arriba esta hecho en la versión básica del diseño dado.

I) Cambiar esta solución por otra donde se defina una *función* `BR_EFECTIVO` que dependa del campo COND y de los FLAGS. Recordar que COND es un alias y los FLAGS están agrupados en un record.

II) Simular el modelo nuevamente con el testbench dado para comprobar que no se cometió ningún error.

III) Intentar sintetizar el modelo. ¿Admite la herramienta de síntesis el uso de funciones?

D. Parámetros Genéricos

La unidad aritmético lógica de ARC es de 32 bits. Parametrizar esta ALU en el ancho de los operandos.

I) Se debe declarar el parámetro genérico en la unidad aritmético lógica.

II) Utilizar el parámetro genérico cuando se instancia la ALU: GENERIC MAP

III) Implementar la unidad aritmético-lógica de manera que el uso de operandos de ancho parametrizable sea completamente efectivo: editar la arquitectuta.

IV) Verificar que la compilación sigue siendo correcta.

V) Simular la nueva versión del multiplicador.

VI) Intentar sintetizar el modelo. ¿Admite la herramienta de síntesis el uso de parámetros genéricos?