Aplicaciones docentes del diseño de un pico-procesador

Jiménez, Carlos J.;Baena, Carmen; Parra, Pilar; Valencia, Manuel
Dpto. de Tecnología Electrónica, Universidad de Sevilla / Instituto de Microelectrónica de Sevilla (Universidad de Sevilla / CSIC)
Sevilla, España
cjesus@us.es

Lopéz-Hinojo, Antonio A.
Universidad de Sevilla
Sevilla, España

***Abstract*—El conocimiento de la estructura interna y del mecanismo de funcionamiento de microprocesadores es una parte muy importante en la formación de ingenieros en electrónica e informática. Este conocimiento puede profundizarse con experiencias de diseño de procesadores, que reúnen además muchos aspectos vinculados a otros conocimientos básicos. Sin embargo, debido a su complejidad, el diseño de procesadores comerciales no es efectivo desde un punto de vista docente. En la presente comunicación presentamos una experiencia de diseño en VHDL de un procesador muy sencillo que demuestra los múltiples aprendizajes que suponen para el alumno.**

Keywords—VHDL; diseño de procesadores; enseñanza de diseño digital.

I. INTRODUCCIÓN

La Electrónica Digital es un campo tan amplio que su enseñanza se lleva a cabo en diferentes asignaturas. Las primeras asignaturas suelen dedicarse al diseño lógico, a partir de puertas lógicas y biestables, y a partir de ahí otras asignaturas se dedican a la enseñanza de diseño desde unos niveles de abstracción más altos, como puede ser el nivel RT. También existen asignaturas dedicadas a la enseñanza del funcionamiento de los procesadores, siguiendo para ello el proceso de programación de algún procesador concreto. Otras asignaturas más avanzadas se dedican a la realización de diseño digital utilizando lenguajes de descripción de hardware. Es bastante frecuente que asignaturas dedicadas a la enseñanza del funcionamiento de los procesadores no tengan mucha relación con asignaturas dedicadas al diseño a partir de Lenguajes de Descripción de Hardware (HDL).

Por todo esto resulta muy provechoso buscar ejemplos de aplicación que aúnen los contenidos impartidos en varias de las asignaturas, de forma que proporcionen al alumno una visión conjunta del proceso de diseño de circuitos digitales. Desde este punto de vista, el diseño de procesadores en un ejemplo muy válido, puesto que puede aunar los contenidos de asignaturas vinculadas a la utilización de procesadores y a asignaturas vinculadas al diseño utilizando HDLs. Sin embargo la complejidad de los procesadores actuales, aunque sean de los más sencillos, lo hacen imprácticos para su realización como un diseño completo.

Con estos antecedentes, en esta comunicación se presenta una experiencia de diseño de un procesador académico que llamaremos pico-procesador por su sencillez. Esta experiencia ha consistido en el diseño y documentación de una librería de celdas básicas, necesarias para el diseño de procesadores, su interconexión para la construcción del pico-procesador y su verificación. Esta experiencia ha sido llevada a cabo como Proyecto Fin de Carrera [1].

La descripción de las celdas básicas y del pico-procesador se ha llevado a cabo utilizando el lenguaje de descripción hardware VHDL. Estas descripciones han sido hechas a nivel RT y cumpliendo con las restricciones de síntesis para que los circuitos obtenidos puedan ser sintetizados e implementados utilizando librerías tecnológicas.

En este trabajo se ha dado gran importancia a la documentación. Que un diseño esté bien documentado es un aspecto sumamente importante, tanto en su desarrollo, como en su mantenimiento posterior. La documentación de un diseño empieza a la vez que la construcción del mismo y finaliza justo antes de la entrega al cliente. Una vez concluido el diseño, los documentos que se deben entregar son una guía técnica, una guía de uso y de instalación.

Son muchos los problemas derivados por una mala documentación o por ausencia de ella. Cualquier cambio en el diseño puede derivar en tener que volver a revisarlo por completo, esto conlleva una pérdida de tiempo considerable y hace que su mantenimiento se haga prácticamente imposible.

Por ello para cada una de las celdas desarrolladas se ha creado una hoja de características que incluye todos los aspectos de funcionamiento e interconexión necesarios para poder utilizarla. De esta forma la librería de celdas puede ser utilizada con facilidad en otros diseños, tanto de otros microprocesadores como de otro tipo de circuitos.

El esquema de esta presentación es la siguiente: en el siguiente apartado se realiza una descripción en detalle del procesador que se va a diseñar, en el apartado III se presentan las celdas diseñadas, en el apartado IV se explica la

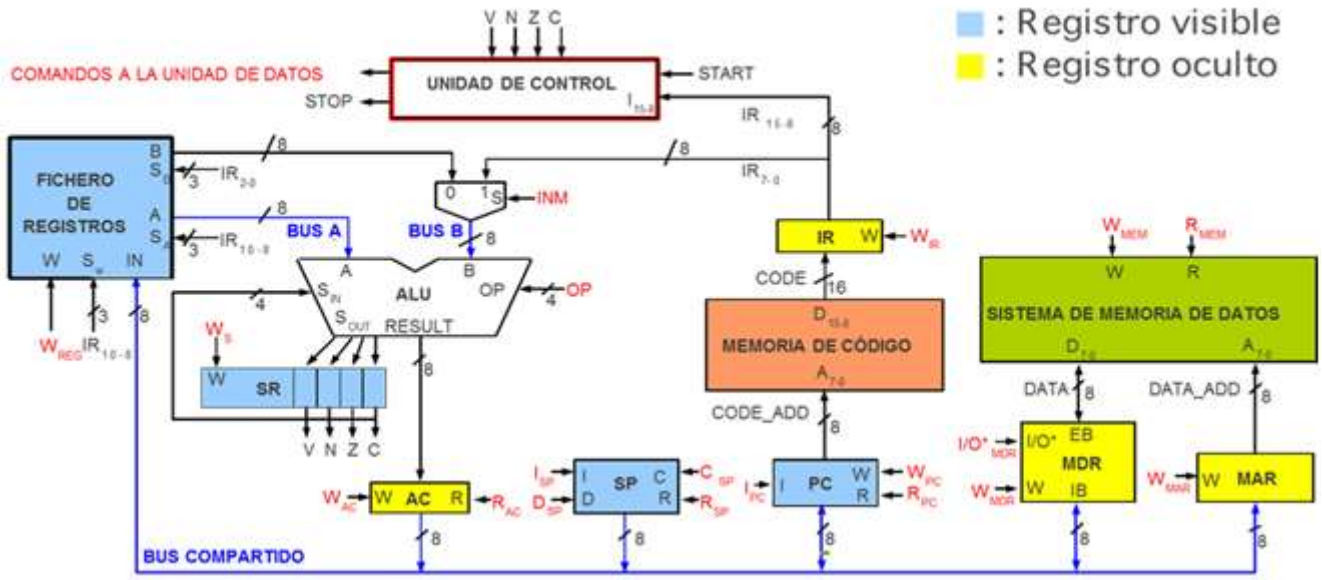

Figura 1. Arquitectura del pico-procesador diseñado.

construcción y verificación del procesador y finalmente se extraen algunas conclusiones.

Descripción del procesador a diseñar

El procesador que se va a diseñar tiene una arquitectura del tipo Harvard, puesto que va a tener una memoria para los datos y otra memoria para las instrucciones y un conjunto de instrucciones reducido (RISC). Este procesador, al ser muy sencillo no va a tener segmentación, es decir, no va a disponer de etapas de pipeline para ir ejecutando instrucciones en paralelo. A continuación se describen con detalle la arquitectura y el conjunto de instrucciones del procesador.

A. Arquitectura del procesador

El diseño a realizar es el de un procesador "académico" [2] pero compatible en el juego de instrucciones con un microcontrolador real (AVR de Atmel). La arquitectura de este procesador se muestra en la figura 1.

Esta arquitectura tiene los siguientes componentes básicos:

- Un bloque Fichero de Registros que contiene 8 registros de datos de propósito general. Este bloque es el que interacciona con la ALU suministrando hasta dos datos fuente, uno a través de su bus A y otro por el bus B. Por otra parte, también recibe datos a través del bus interno compartido. El dato B procede del registro seleccionado por los tres bits menos significativos del registro de instrucciones (IR, $IR_{2:0}$) mientras que el A procede del del registro seleccionado por $IR_{10:8}$, que también seleccionan el registro destino.
- Seis registros de propósito específico: el Contador de Programa (PC), que contiene la dirección de la próxima instrucción que a ejecutar; el Registro de Instrucciones (IR), que contiene la instrucción en curso; el Registro de Estado (SR), que da información sobre el resultado de la última operación realizada en la ALU; el Puntero de Pila (SP) que es un registro que contiene la dirección de la primera posición libre de la pila; el Registro de Datos de Memoria (MDR) que contiene la palabra que se va escribir o se acaba de leer de la memoria de datos y el Registro de Dirección de Memoria (MAR) que contiene la dirección a la que se accede en la memoria de datos.
- Una memoria de datos, de 256 palabras de 8 bits a la que se accede mediante un bus de direcciones y un bus de datos bidireccional. La lectura o escritura en la memoria se controla por dos señales de control, W_{MEM} y R_{MEM}.
- Una Unidad Aritmético Lógica (ALU) con operaciones de suma, resta, desplazamiento a derecha o izquierda de la entrada A y transferencia de las entradas a la salida. También tiene salidas de estado que dan información sobre el resultado de la última operación realizada: Z si es cero, V si ha habido desbordamiento en operaciones de números con signo, N si es negativo y C si se ha producido carry o borrow.
- Una Unidad de Control que secuencia adecuadamente las microoperaciones a realizar para cada instrucción del procesador.

B. Conjunto de instrucciones

El conjunto de instrucciones de este procesador incluye operaciones de acceso a registro y a memoria, de salto, de estado y aritméticas o lógicas. La figura 2 muestra la lista de las instrucciones ordenadas según su código máquina e incluyendo sus descripciones en mnemónico y a nivel RT.

Conjunto de instrucciones (ordenadas por el código de operación, COP)					
COP IR_{15} : IR_{11}	For-mato	Mnemónico y sintaxis	Tipo	Efecto	V N Z C [1]
0 0000	A	ST (Rb), Rf	Mem	M(Rb) ← Rf	- - - -
0 0001	A	LD Rd, (Rb)	Mem	Rd ← M(Rb)	- - - -
0 0010	B	STS dir, Rf	Mem	M(dir) ← Rf	- - - -
0 0011	B	LDS Rd, dir	Mem	Rd ← M(dir)	- - - -
0 0100	C	CALL dir	Salto	M(SP)←PC; SP←SP-1; PC←dir	- - - -
0 0101	-	RET	Salto	PC←M(SP+1); SP←SP+1	- - - -
0 0110	C	BRxx dir	Salto	Xx: PC ← dir	- - - -
0 0111	C	JMP dir	Salto	PC ← dir	- - - -
0 1000	A	ADD Rd, Rf	Arit/Log	Rd ← Rd + Rf	* * * *
0 1010	A	SUB Rd, Rf	Arit/Log	Rd ← Rd - Rf	* * * *
0 1011	A	CP Rd, Rf	Estado	NOP [Rd –Rf ⇒ VNCZ]	* * * *
0 1111	A	MOV Rd, Rf	Movim	Rd ← Rf	- - - -
1 0010	-	CLC	Estado	NOP [⇒ C←0]	- - - 0
1 0011	-	SEC	Estado	NOP [⇒ C←1]	- - - 1
1 0100	A/B	ROR Rd	Despl	Rd ← SHR(Rd, C) [⇒ C←Rd0]	* * * Rd0
1 0101	A/B	ROL Rd	Despl	Rd ← SHL(Rd, C) [⇒ C←Rd7]	* * * Rd7
1 0111	-	STOP	Especial		- - - -
1 1000	B	ADDI Rd, dato	Arit/Log	Rd ← Rd + dato	* * * *
1 1010	B	SUBI Rd, dato	Arit/Log	Rd ← Rd - dato	* * * *
1 1011	B	CPI Rd, dato	Estado	NOP [Rd –dato ⇒ VNCZ]	* * * *
1 1111	B	LDI Rd, dato	Mem	Rd ← dato	- - - -
COPs sin utilizar en CS3: ${9, C, D, E, 10, 11, 16, 19, 1C, 1D, 1E}				[1] El carácter "-" denota "no modificado"; el carácter "*" denota "modificado de forma definida"; otros COP, no documentados.	

Figura 2. Conjunto de instrucciones del pico-procesador

Las instrucciones solo poseen una palabra de código máquina y constan de un total de 16 bits. Los cinco más significativos contienen el código de operación y los once restantes contienen información para la localización de los operandos. Esta información puede presentar diversos formatos que se muestran en la figura 3, en la que se detallan los campos y la función de cada uno de ellos.

Los modos de direccionamiento admitidos son los siguientes:

- Directo de registro: se referencia en la instrucción al registro que contiene el operando.

Figura 3. Formatos de las instrucciones.

- Directo de memoria (o absoluto): se referencia en la instrucción la dirección de memoria del operando.
- Indirecto de registro: se referencia en la instrucción un registro que contiene la dirección de memoria del operando.
- Direccionamiento inmediato: el operando es suministrado en la propia instrucción.

Hay tres tipos de instrucciones de salto, estos son:

- Incondicional.
- Condicional, en las que el salto se realizará dependiendo de algunos valores de los bits de estado. En este pico-procesador se han incluido las condiciones de resultado cero, menor que tanto para magnitudes como para signo, y de desbordamiento tras operar con números con signo;
- De subrutina, de salto a y de retorno de ella.

C. *Unidad de control*

Finalmente el procesador tiene una unidad de control encargada de gobernar y temporizar las distintas micro-operaciones que hay que realizar en la ejecución de cada una de las instrucciones. La descripción de esta unidad de control está

realizada mediante una carta ASM [3] Sin embargo su complejidad es bastante grande, por lo que no va a ser reproducida aquí de forma completa. A título de ejemplo, en la figura 4 se muestra la carta ASM de la selección y ejecución de las instrucciones a partir de la memoria de programa. En esta carta ASM se diferencian tres tipos de instrucciones, las de manejo de datos, las de acceso a memoria y las de salto. Cada uno de estos tipos de instrucciones se ejecuta siguiendo una descripción en forma de carta ASM que por su prolijidad no se han incluido aquí.

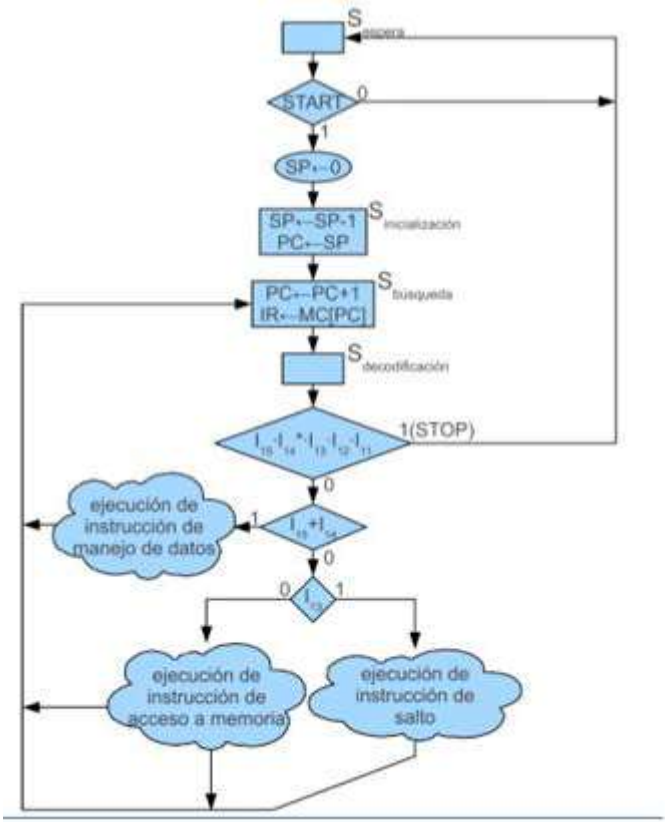

Figura 4. Parte de la carta ASM de la unidad de control.

II. Desarrollo y documentación de la librería de celdas

En el trabajo de diseño del procesador, la primera parte del trabajo consistió en la selección, descripción y verificación de las celdas necesarias para la construcción de la arquitectura del pico-procesador. Esta selección se hizo en base a los bloques que presenta la arquitectura y que se mostraron en la figura 1. En la Tabla I se muestra el listado de las celdas desarrolladas.

Cada una de las celdas fue descrita utilizando el lenguaje de descripción de hardware VHDL y simuladas utilizando el programa Modelsim. Como ya puede concluirse viendo la arquitectura del procesador, no existe una dificultad especial en el diseño y verificación de las celdas. Gran parte de las celdas se corresponden con registros con pequeñas variaciones en su

TABLE I. Celdas descritas

Nombre	Descripción
PC	Registro contador de programa
MEMCOD	Memoria de programa
IR	Registro de instrucciones
ALU	Unidad Aritmético Lógica
REGISTRO	Banco de registros
MULTIPLEXOR	Multiplexor de entrada a la ALU
MEMDAT	Memoria de datos
MDR	Registro de datos de la memoria
MAR	Registro de direcciones de la memoria
AC	Acumulador de salida de la ALU
SR	Registro de estado
SP	Puntero de pila

funcionamiento. Hay un banco de registros (con ocho registros de 8 bits con dos accesos de lectura y uno de escritura), pero el resto son registros de 8 bits, que se diferencian en su forma de acceso y en si tienen capacidad de incremento. La única situación no demasiado común es el empleo de un bus compartido, que obligó a la utilización de líneas de entrada/salida en algunos registros y por lo tanto el empleo de señales de alta impedancia.

Como ya se ha comentado en la introducción, uno de los aspectos importantes de este trabajo es la creación de una documentación que permitiera la reutilización de las celdas diseñadas. Para ello se creó una hoja de características para cada una de las celdas desarrolladas. Para seleccionar la información que debían de contener estas hojas de características, se consultó con hojas de características de celdas de distintas librerías tecnológicas. Muchas de estas hojas incluyen información relativa a aspectos tecnológicos (tiempos de retraso, comportamiento para distintas tensiones, etc.). Estos aspectos no son de aplicación en nuestro caso, puesto que las descripciones que presentamos son independientes de la tecnología. En la documentación realizada se incluye información relativa a aspectos de interfaz y funcionales, entre ellos:

- Símbolo.
- Introducción (descripción textual de su funcionamiento).
- Descripción formal de su funcionamiento a nivel RT.
- Puertos de entrada/salida y su descripción.

Además de la descripción de cada una de estas celdas se realizó la descripción de la unidad de control. Este es el bloque más complejo de todos. Para su diseño se han seguido las pautas existentes para el diseño de máquinas de estado en los lenguajes de descripción de hardware.

III. Diseño del pico-procesador

El siguiente paso en el proceso de diseño consistió en la construcción del pico-procesador completo. Debido a la metodología seguida, de descripción de cada uno de los bloques que componen el procesador como celdas básicas, el diseño del procesador consistió en la colocación e interconexión de todos los componentes.

El proceso de verificación se llevó a cabo en dos fases: en una primera fase se probó la correcta ejecución de todas las instrucciones: paso de datos de memoria a registros y viceversa, realización de operaciones por parte de la ALU, etc. En una segunda fase se realizó una verificación más completa comprobando la ejecución de programas. En esta comunicación vamos a presentar una de las pruebas realizadas. El ejemplo que presentamos consiste en una pequeña rutina para guardar en un registro el mayor de los datos almacenados en otros dos registros, para el caso de que los datos sean números con signo.

La operación a realizar es la siguiente a nivel RT:

$$R0 \leftarrow \text{Mayor}(R1, R2)$$

La rutina que realiza esta operación es la siguiente:

```
            LDI R1, $48 ; dato a R1
            LDI R2, $D1 ; dato a R2
MAX:        CP R1, R2   ; Comparar
            BRLT R2MAYOR; ¿R1 menor?
            MOV R0, R1  ; No. Devolver R1
            RET         ; Retornar
R2MAYOR:    MOV R0, R2  ; Devolver R2
            RET         ; Retornar
```

En esta rutina, en primer lugar se cargan los valores $48 en el registro R1 y $D1 en el registro R2. Después comienza la subrutina (etiqueta MAX) con la instrucción CP para hacer la comparación. Esta instrucción realiza la resta R1 – R2 pero no guarda el resultado, sólo actualiza el registro de estado. La siguiente instrucción (BRLT, *branch if less than*) realiza el salto a la etiqueta si el registro de estado indica que el resultado de la resta ha sido negativo (N⊕V). Sin embargo ha de tenerse en cuenta que para este resultado se entiende que los datos restados son números con signo (codificados en complemento a 2). En función del resultado de la comparación se guarda en R0 el valor de R1 (si no se salta) o de R2 si se produce el salto condicional, y se retorna.

Una vez realizada esta rutina hay que pasarla al código máquina que hay que introducir en la memoria de programa. El resultado se muestra en la tabla II.

TABLE II. Programa de prueba

Posición	Código Máquina
$00	11111 001 01001000
$01	11111 010 11010001
$02	01011 000 00000001
$03	00110 011 00000110
$04	01111 000 00000001
$05	00101 000 00000000
$06	01111 000 00000010
$07	00101 000 00000000

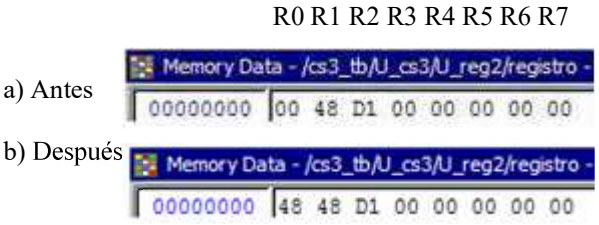

Figura 5. Datos en registros internos, a) antes y b) después de ejecutar la subrutina.

En la figura 5 se muestran los contenidos de los registros antes de ejecutarse la subrutina MAX y después de su ejecución. Como puede comprobarse en los resultados mostrados, en la parte (a) de la figura se muestran los datos almacenados en los registros tras cargar R1 y R2 (R0 permanece a $00), y en la parte (b) se muestran los datos almacenados en los registros una vez terminada la ejecución de la rutina. En el registro R0 aparece almacenado el valor $48 ($48 = $72_{(10)}$), que es el mayor de los dos, puesto que como están codificados en complemento a 2, el dato $D1 se corresponde con un número negativo ($D1 = - $47_{(10)}$).

IV. Conclusiones

En esta comunicación se presenta una experiencia de diseño de un pico-procesador para aplicaciones docentes. La experiencia, que se ha llevado a cabo como Proyecto Fin de Carrera, ha consistido en el diseño, verificación y documentación de una librería de celdas, que son la base para la construcción de procesadores sencillos. Además, y utilizando esta librería de celdas se ha construido un pico-procesador, que ha sido también verificado.

Las descripciones, tanto de las celdas como del procesador, han sido llevadas a cabo en VHDL a nivel RT y cumpliendo con las restricciones de síntesis, de forma que el diseño pueda ser sintetizado e implementado en tecnologías tanto ASIC como FPGA.

Este trabajo ha supuesto un aprendizaje en diferentes disciplinas: por una parte ha sido necesario conocer aspectos de diseño digital a nivel RT utilizando HDLs, pero también incluye aspectos de arquitectura de procesadores. Finalmente la importancia dada a la documentación permite la reutilización de la librería de celdas desarrollada.

Referencias

[1] Antonio Alberto López Hinojo, "Diseño y docuementación de una librería de celdas para el diseño de procesadores", Proyecto Fin de Carrera, Ingeniería Técnica Industrial, especialidad en Electrónica Industrial, Escuela Politécnica Superior, Universidad de Sevilla, septiembre de 2015.

[2] D. Guerrero y otros profesores del Departamento de Tecnología Electrónica de la Universidad de Sevilla. "Computador simple 2010". Apuntes de "Estructura de computadores" de los Grados en Ingeniería Informática. 2010.

[3] D. Green: "Modern Logic Design". Addison-Wesley. 1986.