# Diseño de procesadores educativos: alternativas de diseño multiciclo

*Julio Sahuquillo Borrás, Marina Alonso Díaz, María E. Gómez Requena*

*Departamento de Informática de Sistemas y Computadores*
*Escuela U. de Informática –Universidad Politécnica de Valencia–*

E_mail: { jsahuqui, malonso, megomez} @disca.upv.es

## Resumen

*La implantación de los nuevos planes de estudio en las carreras universitarias de informática ha incentivado a los autores al diseño de procesadores educativos, con el fin de que los alumnos los puedan implementar en el laboratorio y estudiar su funcionamiento. Este trabajo se ha visto recompensado no sólo por una mayor asistencia y motivación de los alumnos al laboratorio, sino también por un mejor aprendizaje de la asignatura. Implementar un procesador, aunque sólo sea con una herramienta de simulación digital y poder observar qué líneas de control se activan y cómo circulan los datos entre las unidades funcionales de la máquina implementada por ellos mismos es una de las mayores satisfacciones que obtiene un estudiante cuando cursa la asignatura de Estructura de Computadores I (ECI), que se imparte en el segundo cuatrimestre del primer curso en las titulaciones de Ingeniero Técnico de Informática de Sistemas, Ingeniero Técnico en Informática de Gestión e Ingeniero en Informática.*

*En [1] se presentan distintas alternativas de diseño de la unidad de control del procesador, como el control monociclo, el multiprogramado, y el multiciclo. Los dos primeros ya han sido abordados por los autores en publicaciones recientes así como libros de texto [2]. En el presente trabajo se aborda el diseño de la tercera alternativa, debido a los excelentes resultados pedagógicos obtenidos con los trabajos precedentes.*

## 1 Introducción

En [1] sólo se presenta el diseño de una unidad de control multiciclo. El hecho de que se estudie un único diseño conlleva a que el alumno no realice muchos ejercicios de diseño y por tanto no llegue a comprender la esencia del diseño multiciclo. Los autores consideran de vital importancia que el estudiante asimile este concepto, y por ello, para su estudio presentan implementaciones de una unidad de control multiciclo en dos, tres y cuatro fases.

Los autores se encuentran adscritos a la Universidad Politécnica de Valencia. En ésta, los alumnos estudian como modelo teórico en la asignatura de ECI el presentado por Patterson y Hennessy [1]. Para poder implementar el procesador en el laboratorio es necesario ceñirse a las limitaciones de la herramienta de diseño digital que se utilice. Como en los trabajos realizados se utiliza el CASCAD [3] que admite *buses* de hasta 16 líneas, los autores se han visto obligados a adaptar el modelo teórico del MIPS R2000 de 32 bits a los 16 bits de la herramienta. Esto se ha realizado reduciendo, entre otros, el número de registros y el número de bits del código de operación.

Para realizar el diseño se suponen retardos de tiempos de los componentes que se ajustan bastante a los reales, obteniendo una unidad de control donde la instrucción que más tiempo emplea en ejecutarse (la de carga) utiliza cinco ciclos. La elección del número de fases máximo adecuado para implementar una unidad de control es función de las características de las instrucciones a implementar y de los retardos de tiempo que incorporan las unidades funcionales que constituyen la ruta de datos. Como los retardos son función de la tecnología existente, es factible que una misma instrucción utilice distinto número de ciclos si se utiliza distinta tecnología en su implementación.

En el presente trabajo se analiza la implementación de una unidad de control multiciclo de dos, tres y cuatro fases. La estructura del presente trabajo es la siguiente: en el apartado dos se presenta el procesador; en el apartado tres se realiza un estudio detallado de la ruta de datos y el diseño de la unidad de control basado en un procesador multiciclo de dos fases; en el apartado cuatro se presenta una extensión de la ruta de datos y la unidad de control a un procesador multiciclo

de tres fases, y en el apartado cinco la extensión se realiza a cuatro fases. En el apartado seis se muestra un ejemplo de las simulaciones que se pueden realizar. Y por último, en el apartado siete aparecen las conclusiones.

## 2 El procesador

Para implementar un procesador educativo, la primera decisión a tomar siempre es el conjunto de instrucciones a implementar. Los autores eligieron un subconjunto pequeño del modelo teórico [1] con el objetivo de facilitar la comprensión de los conceptos. Añadir más instrucciones no aporta nuevos conceptos y dificulta más la comprensión del proceso, por ello se ha decidido utilizar únicamente ese subconjunto.

Las principales decisiones que realizaron los autores en el diseño fueron:

1. Qué instrucciones elegir.
2. Qué formatos de instrucción elegir.
3. Qué códigos de operación elegir.

Las dos primeras decisiones determinan la ruta de datos, por lo tanto deben tener la mayor similitud posible con el modelo teórico. Con este fin, los autores respetaron, con ciertas restricciones, el nombre y orden de los campos del modelo teórico (32 bits) en el diseño del procesador con la herramienta de simulación (16 bits). Como la ruta práctica es muy similar a la expuesta en la Figura 1, donde se muestra la ruta de datos del procesador de dos ciclos estudiada en el modelo teórico, cualquier comentario se realizará sobre ésta.

Respecto a los códigos de operación, hay que elegirlos cuidadosamente para facilitar la implementación del circuito de control, de forma que se pueda simplificar lo máximo posible.

La Tabla 1, muestra las instrucciones elegidas para la implementación de la máquina. Estas se pueden clasificar en tres grupos: de carga (lw) y almacenamiento (sw), de salto (beq) y aritmético–lógicas. En la tabla se exponen las instrucciones elegidas y para cada una de ellas se indican las reglas de escritura en lenguaje ensamblador o sintaxis, el tipo de formato que utiliza la instrucción para codificarse en lenguaje máquina y la función que realiza la instrucción descrita en lenguaje simbólico.

| *Instrucción* | *Sintaxis* | *Descripción* | *Formato* |
|---|---|---|---|
| Carga | lw Rt, desp(Rs) | Rt ← M[desp+Rs] | I |
| Almacena-miento | sw Rt, desp(Rs) | M[desp+Rs] ← Rt | |
| Salto | beq Rs, Rt, etiqueta | Salta a la dirección referenciada por etiqueta si Rs=Rt CP←CPactualizado+d esplazamiento | |
| Aritmético-Lógicas | add Rd, Rs, Rt | Rd ← Rs+Rt | R |
| | sub Rd, Rs, Rt | Rd ← Rs–Rt | |
| | and Rd, Rs, Rt | Rd ← Rs AND Rt | |
| | or Rd, Rs, Rt | Rd ← Rs OR Rt | |

Tabla 1: Sintaxis de las instrucciones.

En la Tabla 2 se exponen los formatos utilizados para codificar las distintas instrucciones. Las instrucciones elegidas se pueden agrupar en dos tipos, en función del formato que utilizan: las que necesitan codificar las direcciones de 3 registros utilizan el formato de tipo R y las que necesitan codificar las direcciones de 2 registros utilizan el formato de tipo I. Como sólo se dispone de 8 registros, se necesitan tres bits para poder codificar cada uno de los registros en la instrucción.

| | *C.O.* | *Rs* | *Rt* | *Rd* | *Func* |
|---|---|---|---|---|---|
| add | 0000 | *rs* | *rt* | *rd* | 001 |
| sub | 0000 | *rs* | *rt* | *rd* | 010 |
| or | 0000 | *rs* | *rt* | *rd* | 100 |
| and | 0000 | *rs* | *rt* | *rd* | 101 |
| sw | 1011 | *rs* | *rt* | Despl. | |
| lw | 1110 | *rs* | *rt* | Despl. | |
| beq | 1001 | *rs* | *rt* | Despl. | |
| | *15 12* | *11 9* | *8 6* | *5 3* | *2 0* |

Tabla 2: Formato de las instrucciones.

En las instrucciones que utilizan formato de tipo R (las aritmético–lógicas) el valor del código de operación es 0, al igual que en el modelo teórico. En consecuencia, es el contenido del campo función (***Func***), a través de conexiones directas a la ALU, el que indica cuál es la operación que ésta debe realizar.

Las conexiones directas suponen una simplificación respecto al modelo teórico.

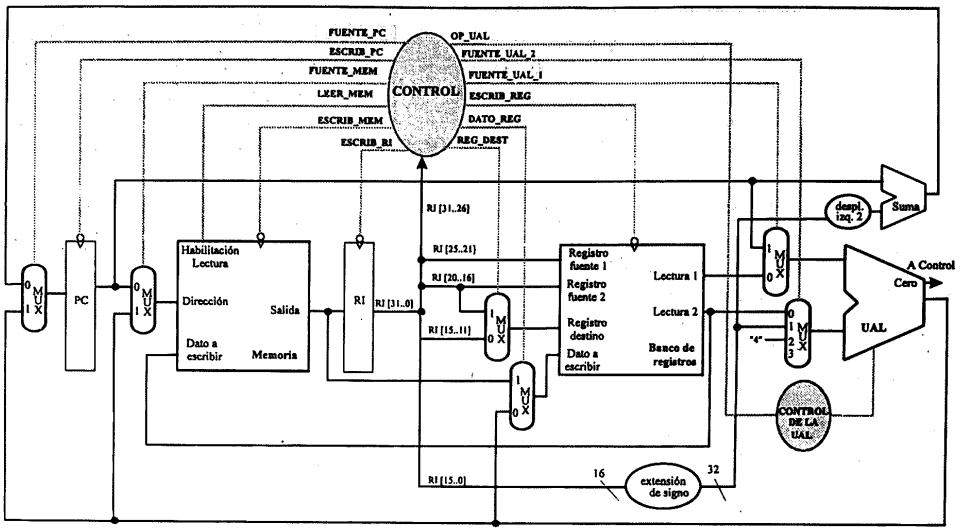


Figura 1: Ruta de datos y unidad de control de dos fases.

## 3 Ruta de datos y control en dos fases

Las actividades a realizar por el alumno se centran en los dos óvalos sombreados que se presentan en la Figura 1, es decir, en la implementación de los circuitos de control. Para ello, el alumno dispone un fichero donde tiene que implementar el nuevo contenido de estos *custom-chips*. La Figura 2 muestra el contenido de este fichero con la ruta de datos. La primera tarea que debe realizar el alumno es establecer la relación entre los componentes de la ruta de datos teórica con la implementada en el CASCAD.

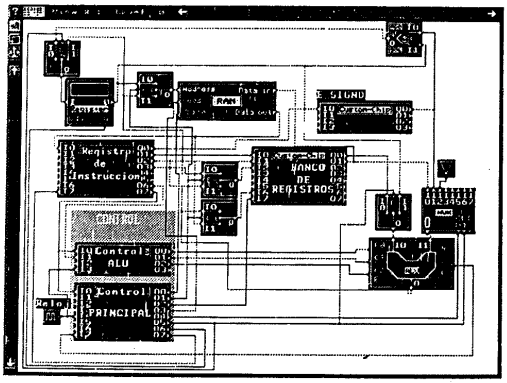


Figura 2: Ruta de datos y unidad de control implementadas en el fichero *ruta.cas*.

### 3.1 Diseño de la unidad de control principal

En primer lugar, nos centraremos en el diseño de la ruta de dos fases para posteriormente analizar la de tres y cuatro. Para poder realizar la unidad de control la primera tarea es identificar las señales de control que intervienen en la ruta. A continuación se realiza el diagrama de la máquina de estados para poder determinar cuándo se activa cada señal y de este modo poder codificarlas. La máquina de estados del procesador de dos fases aparece en la Figura 3.

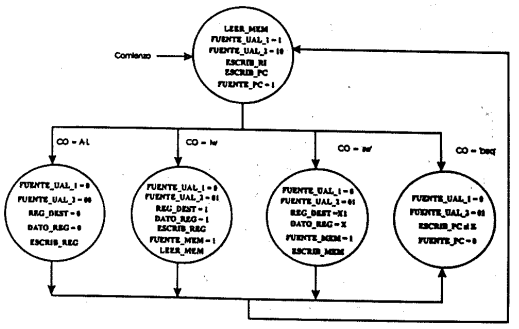


Figura 3: Máquina de estados para la unidad de control de dos fases.

Una vez se sabe cuándo se debe activar cada una de las señales de control, la siguiente etapa consiste en realizar la simplificación de las señales de la unidad de control de forma que resulte el circuito más simple posible. En la Tabla 3 aparece la simplificación resultante.

| *Señal de control* | | *Función* | *Simplificación* |
|---|---|---|---|
| LEER_MEM | | F0+F1*lw | F0+F1*CO2 |
| ESCRIB_MEM | | F1*sw*reloj | F1*CO0*CO1*reloj |
| REG_DEST | | F1*(lw+sw) | F1*CO1 |
| DATO_REG | | F1*(lw+sw) | F1*CO1 |
| ESCRIB_REG | | F1*(A-L +lw) | F1*/CO0*reloj |
| FUENTE_PC | | F0 | F0 |
| FUENTE_MEM | | F1*(lw+sw) | F1*CO1 |
| FUENTE_UAL_1 | | F0 | F0 |
| FUENTE _UAL_2 | L | F1(lw+sw) | F1*C01 |
| | M | F0 | F0 |
| FUENTE_PC | | F0 | F0 |
| ESCRIBE_PC | | F0+(F1*beq*Z) | (F0+(F1*(/CO1*CO0)*Z))*reloj |
| ESCRIBE_RI | | F0 | F0*reloj |

Tabla 3: Simplificación de las señales del control principal.

Una vez el alumno ha realizado la simplificación de las señales de control, ya puede proceder a la implementación del *custom-chip* correspondiente a la unidad de control principal. La Figura 4 muestra el contenido de dicho *custom-chip*. En el procesador implementado siempre se escribe, tanto en los registros como en la memoria, en los flancos de bajada. Por ello el reloj gobierna las operaciones de escritura, es decir, la señal de reloj aparece ligada mediante una puerta AND a cada una de estas señales.

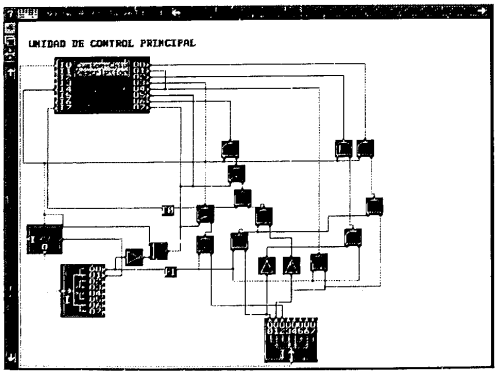


Figura 4: *Custom-chip* del control principal.

En la Tabla 4 se puede ver la asignación de patillas del *custom-chip* de la unidad de control principal a cada una de las señales de control. Como se puede apreciar en la tabla, se ha utilizado una misma salida para varias señales cuando el resultado de la simplificación era el mismo.

| *Señal de control* | | *Patilla salida custom-chip* |
|---|---|---|
| ESCRIB_MEM | | O0 |
| FUENTE_MEM | | O1 |
| REG_DEST | | O1 |
| DATO_REG | | O1 |
| ESCRIB_REG | | O2 |
| LEER_MEM | | O3 |
| FUENTE_ UAL_2 | L | O4 |
| | M | O4 |
| FUENTE_UAL_1 | | O5 |
| FUENTE_PC | | O5 |
| ESCRIBE_PC | | O6 |
| ESCRIBE_RI | | O7 |

Tabla 4: Patillaje de las señales de control.

## 3.2 Diseño de la unidad de control de la ALU

Al igual que la precedente, la unidad de control de la ALU debe de tener en cuenta el tipo de instrucción de que se trata, así como la fase en la que se encuentra el procesador. La Tabla 5 muestra como quedan las nuevas señales de control de la ALU (Cx') tomando como referencia las del procesador monociclo (Cx).

| *Señal de control* | *Simplificación* |
|---|---|
| C2' | C2*F1 |
| C1' | C1*F1 |
| C0' | C0*F1+F0 |

Tabla 5: Señales de control de la ALU.

La Figura 5 muestra la implementación del *custom_chip* de la unidad de control de la ALU.

## 4 Ruta de datos y control en tres fases

Si realizamos una adaptación de la ruta de datos anterior para implementar una unidad de control multiciclo de tres fases, podemos resolver alguno

de los inconvenientes que aparecían en el diseño estudiado para una unidad de control multiciclo de dos fases.

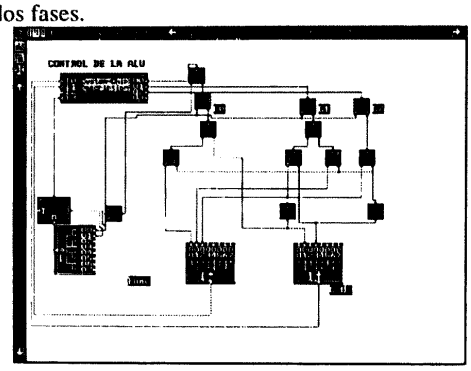


Figura 5: *Custom_chip* de control de la ALU.

La principal ventaja es la desaparición del sumador necesario en la unidad multiciclo, vista en apartado anterior, que era necesario para calcular la dirección de salto en la instrucción beq. Este sumador está justificado en el diseño basado en dos fases debido a que el cálculo de la dirección de salto se hace en la misma fase que el cálculo para comprobar si debe o no realizar el salto, y por lo tanto la ALU no puede realizar dos operaciones al mismo tiempo. Por otra parte, se hace necesaria la introducción de un registro temporal que denominaremos TEMP, donde se almacenará la dirección de salto calculada. Aparece asociado a este registro una señal de control que hemos denominado ESCRIB_TEMP.

En la Figura 6 aparece el diseño de la máquina de estados del procesador para tres fases.

## 5 Ruta de datos y control en cuatro fases

En el caso de una adaptación a cuatro fases de la unidad de control vista en el punto anterior, las modificaciones a realizar son mínimas. Realmente las dos primeras fases (fase 0 y fase 1) se mantienen comunes, y también lo hace la fase 2 para las instrucciones aritmético-lógicas, beq y sw. La única instrucción que se ve alterada respecto al diseño de la unidad de control multiciclo en tres fases es la instrucción lw, que necesita una fase más, fase 3, para realizar la escritura en el registro destino.

La Figura 7 muestra la máquina de estados multiciclo con un diseño de cuatro fases.

Como la implementación de estos tres procesadores llevaría muchísimo tiempo, los autores facilitan a los alumnos dos ficheros con los elementos que constituyen el procesador para las versiones de 2, 3 y 4 ciclos. Los alumnos simplemente tienen que rellenar el *custom-chip* correspondiente al circuito de control para cada ruta y verificar su funcionamiento tras la inclusión de un sencillo programa en la memoria de instrucciones.

## 6 Simulaciones

Sin duda, la experiencia más gratificante de las prácticas para el alumno es comprobar que el diseño que han realizado funciona. Es decir que la unidad de control realiza correctamente la ejecución de las instrucciones del programa, activando las señales adecuadas en el momento correspondiente.

Al alumno se le prepara un programa que debe ejecutar para comprobar el correcto funcionamiento del procesador que han diseñado previamente. A continuación se muestra uno de los ejercicios propuestos.

☞ *Establézcanse todos los registros inicialmente a cero, y en las posiciones cuatro y cinco de la memoria de datos, un seis y un tres respectivamente (M[4]=6 y M[5]=3). El programa se cargará a partir de la dirección 10 de memoria.*

La Tabla 6 presenta el programa escrito en código ensamblador así como el código máquina asociado que deberá introducirse en la memoria de instrucciones para su ejecución.

| Código ensamblador del programa: | Código máquina: |
|---|---|
| lw $3, 4($0) # $3=6 | 0xE0C4 |
| lw $2, 5($0) # $2=3 | 0xE085 |
| add $4, $3, $2 # $4=6+3=9 | 0x06A1 |
| sub $5, $3, $2 # $5=6–3= 3 | 0x06AA |
| sw $4, 4($0) # M[4]=9 | 0xB104 |
| sw $5, 6($0) # M[6]=3 | 0xB146 |

Tabla 6: Código ensamblador y código máquina del programa propuesto.

## Conclusiones

Como el funcionamiento del procesador sólo puede verificarse después de que haya sido todo implementado, es decir, es una variable del tipo "todo o nada", los alumnos se sienten incentivados a construir su propia máquina y motivados al

estudio de la asignatura, lo que es un gran aliciente y estimula a los autores a seguir trabajando e innovando en este campo temático tan gratificante.

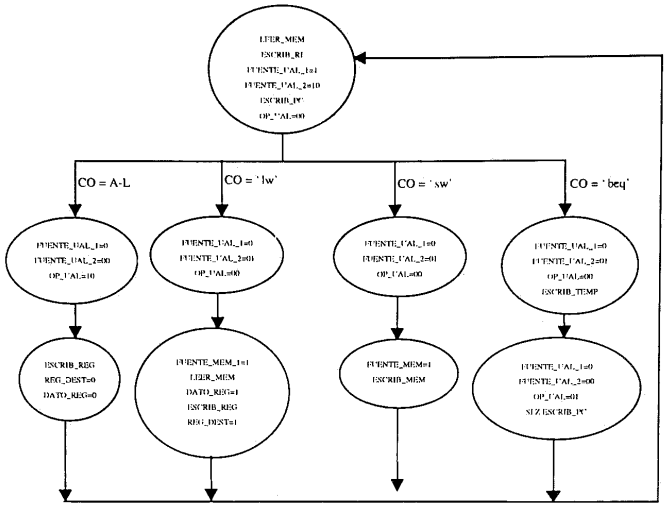


Figura 6: Máquina de estados para la unidad de control de tres fases.

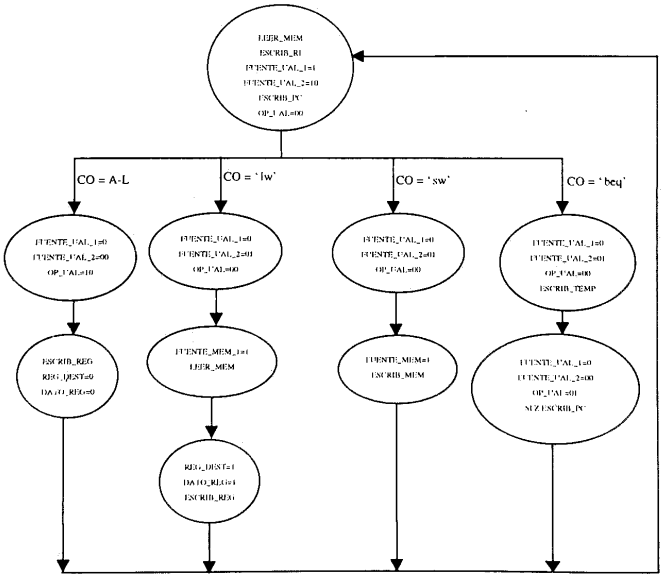


Figura 7: Máquina de estados para la unidad de control de cuatro fases.

Implementar distintas rutas, habilita al alumno para comprender el flujo de los datos de unas unidades a otras.

Implementar distintos circuitos de control, es un ejercicio ideal para la comprensión del funcionamiento de dichos circuitos.

## Referencias

[1] Patterson, D.A., Hennesssy, J.L. *Organización y diseño de computadores. La interfaz hardware/ software.* Editorial Mc-Graw-Hill. Segunda edición en español, 1995.

[2] Grup Tera. *Experiencias en Estructura de Computadores.* Servicio de publicaciones de la U.P.V. Ref.005.

[3] *Computer Assisted Simulation for Circuit Analysis & Design (CASCAD). Reference Manual.* Edusoft.