Un computador didáctico elemental (CODE-2)

A.Prieto F.J. Pelayo F.Gómez Mula J.Ortega

A.Cañas A.Martínez F.J.Fernández

Departamento de Arquitectura y Tecnología de Computadores
E.T.S.I. Informática
Universidad de Granada
E-18071 Granada. e-mail: aprieto@ugr.es

Resumen

Se describe en los niveles de lenguaje máquina, micromáquina y usuario un computador didáctico elemental (CODE-2) que reúne las características deseables para la enseñanza de las asignaturas de introducción a los computadores. Este computador ha sido diseñado completamente, con unidades de control cableada y microprogramada. También se dispone de un entorno completo de simulación que incluye emulador, ensamblador, desensamblador y visualización del comportamiento dinámico de todos los elementos internos del computador.

1. Introducción

La arquitectura de computadores es una disciplina típica de ingeniería y para preparar material didáctico para ella se debe realizar un laborioso trabajo de generalización de las diversas técnicas utilizadas en computadores concretos, y no limitarse a recopilar información detallada sobre ellos. Este material debe presentar al alumno abstracciones de equipos reales, de forma que le capaciten no sólo a entender los computadores actuales sino también los futuros, cuando éstos vean la luz. Este concepto es especialmente relevante en un ámbito tan cambiante y en expansión como el de los computadores. Esta idea es de gran importancia en los cursos de iniciación, donde hay que lograr que el estudiante fije su atención en las cuestiones esenciales, y no se pierda en los detalles, la mayoría de ellos fugaces, de las máquinas reales.

Dentro de este contexto la mayoría de libros de texto que tratan de la estructura o arquitectura de computadores suelen presentar máquinas ficticias que muestran cruda y claramente los conceptos más básicos y generales; sin perjuicio de que el conocimiento de éstos posteriormente se reafirme y particularice considerando casos de máquinas concretas y reales. Chu [1] fue uno de los primeros autores que describió con detalle la estructura y diseño de un computador didáctico; con posterioridad Gorsline [2] presenta el "G-1", Karam y Bryant [3] el "CUSP" *(Carlenton's Utterly Simple Processor),* Mano [4,5] el "Basic Computer" y posteriormente [6] otros dos prototipos (uno RISC y el otro CISC), Foster [7] el "Blue" (nombre coincidente con el color de su chasis), Knuth [8] el "MIX", Lee [9] el "SDC" *(Simple Didactic Computer),* Tanenbaum [10] el "Mac-1" *(Macroarchitecture-1),* Hennessy y Patterson [11] una máquina RISC genérica, el "DLX" (*DeLuXe*), Carpinelli [12] el *Relatively Simple Microprocessor* y Patt [13] el "LC-2" *(Little Computer-2).*

Otros autores utilizan para apoyar sus explicaciones subconjuntos o modelos simplificados de máquinas reales. Así, Tannebaum [14] presenta el "IJVM" *(Integer Java Virtual Machine),* que contiene el subconjunto de instrucciones para manejar enteros de la JVM, Murdocca [15] el "ARC" *(A RISC Computer)* que es un subconjunto del SPARC, y Patterson y Hennesy [16] utilizan como modelo didáctico una versión simplificada del MIPS R2000/R3000.

Los autores españoles de textos de introducción a la estructura de computadores también suelen considerar máquinas ideales. Este es el caso del procesador descrito por de Miguel [17], el "ODE" (Ordenador Didáctico Elemental) de Prieto, Lloris, Pelayo y Gómez-Mula [18-21], la "MS" (Máquina Sencilla) de Ayguadé, Navarro y Valero-García [22], y la Máquina Rudimentaria de Hermida, del Corral, Pastor y Sánchez [23,24], también descrita por Ruz [25]. Gregorio Fernández, con una gran sentido pedagógico, presenta su curso de arquitectura y sistemas operativos [26] partiendo de una máquina muy sencilla "Simplez(+i4)", que va ampliando en máquinas virtuales sucesivas conforme va introduciendo conceptos más complejos: "Algorítmez", "Regístrez", "Monoalgorítmez" y "Multialgorítmez".

Ante esta pléyade de máquinas didácticas, Fermín Sánchez presentó en la edición anterior de estas jornadas [27] un interesante trabajo en el que definió las características deseables en un procesador pedagógico para la enseñanza básica de arquitectura de computadores. En él, aparte de describir las características indicadas, se comparaban tres de los procesadores citados anteriormente: el MS, el DLX y la Máquina Rudimentaria. Estas tres máquinas se acercaban en mayor o menor grado al modelo preconizado en [27], pero ninguna de ellas lo seguía plenamente.

En el presente trabajo presentamos un Computador Didáctico Elemental (CODE-2) que consideramos satisface plenamente las especificaciones y prestaciones propuestas en [27]. De este computador en la actualidad se dispone del siguiente material:

- Definición de la estructura accesible por el programador y del lenguaje máquina.
- Definición de un lenguaje ensamblador
- Implementación de un emulador para un entorno MS-Windows
- Implementación de un ensamblador cruzado, para un entorno MS-Windows.
- Implementación de herramientas integradas para trabajar con CODE-2 (contiene además del ensamblador y emulador, depurador, desensamblador, visualización del funcionamiento dinámico de cada una de los elementos del procesador, etc.)
- Diseño completo de CODE-2 con unidad de control cableda.
- Diseño completo de CODE-2 con unidad de control microprogramada.
- Diseño de CODE en VHDL
- Construcción de un entrenador CODE-2 con circuitos lógicos programables (FPGA).

El resto del trabajo se estructura de la siguiente forma: la Sección 2 presenta las características generales de CODE-2 cotejándolas con las indicadas en [27] para un procesador pedagógico ideal; la Sección 3 presenta someramente el repertorio de instrucciones máquina. El computador CODE-2, para ser utilizado por los alumnos como entrenador, se ha montado en un bastidor y su panel frontal contiene visualizadores de los principales elementos internos, en la forma que se describe en la Sección 4. La Sección 5 considera las pautas seguidas para el diseño de la máquina; por último, la Sección 6 presenta las conclusiones.

2. Características de CODE-2

Obviamente un procesador didáctico ideado para introducir la estructura y el funcionamiento de un computador debe tener los elementos indispensables para que el alumno adquiera los conocimientos que se pretenden. En un curso básico estos conocimientos se centran en conceptos tales como: memoria principal, puertos de entrada/salida (E/S), unidad de procesamiento de datos, unidad de control, buses, etc., así como los relacionados con la estructura de dichos elementos a nivel de micromáquina (RTL): contador de programa (PC), puntero de pila (SP), banco de registros (RF), unidad aritmético-lógica, biestables indicadores de la ALU, etc. También deben introducirse los conceptos básicos inherentes al funcionamiento dinámico del procesador: fases y estados en la ejecución de una instrucción, activación de los biestables indicadores, circulación de la información por los buses, sincronización, etc.

En la Figura 1 se muestra un esquema de los elementos de CODE-2 a los que se tiene acceso desde el lenguaje máquina; es decir, esta figura presenta los elementos visibles al programador. Las principales características de CODE-2 son:

1. Su arquitectura es sencilla y ortogonal. En efecto, dispone de tan sólo 16 instrucciones

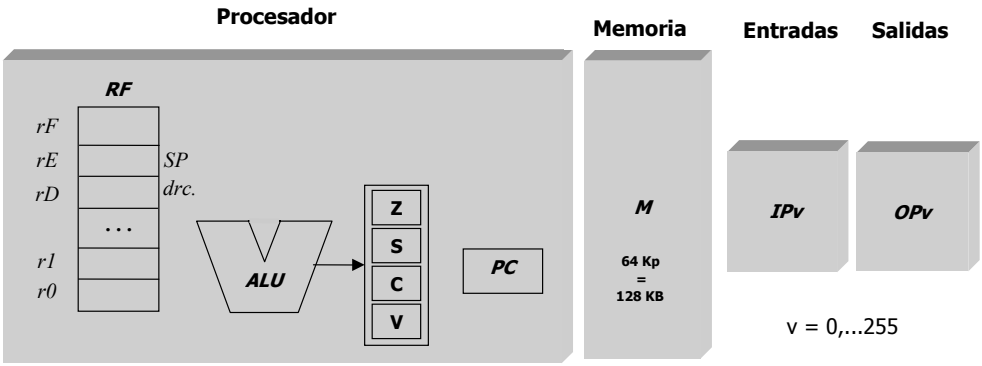

Figura 1. Elementos de CODE-2 accesibles al programador

máquina, coincide el tamaño de los datos y las instrucciones (16 bits) con lo que se evitan problemas de alineamiento y se facilita el diseño del procesador, la memoria es de 128 Kbytes haciéndose su direccionamiento por palabras y no por bytes. La unidad de procesamiento de datos dispone de un banco de 16 registros (*RF*) de 16 bits. El formato de las instrucciones es completamente ortogonal (Figura 2), estando ubicado el código de operación (*codop*) siempre en los cuatro primeros bits. En la Figura 2, *rx, rs* y *ra* representan registros de RF, *cnd* la condición de salto, y *v*, un valor inmediato de 8 bits.

2. También sigue la tendencia RISC, en cuanto es una arquitectura de carga-almacenamiento. En efecto, todas las operaciones de la ALU se hacen entre registros de RF, cuyos contenidos pueden cargarse o memorizarse con dos instrucciones específicas, una de carga desde memoria (*LD*) y otra de almacenamiento en memoria (*ST*), respectivamente. Hay cuatro espacios de direcciones (Figura 1): direcciones de registros (16), direcciones de memoria (64 Kp), direcciones de puertos de entrada (256), y direcciones de puertos de salida (256). El registro *rE* del banco de registros (Figura 1) actúa como puntero de pila (*SP*), y, al igual que el contador de programa (*PC*), es de 16 bits. El registro *rD* se utiliza para almacenar direcciones (direccionamiento indirecto).
3. La ALU realiza operaciones muy sencillas: suma y resta de enteros en complemento a 2, operación lógica NAND, y desplazamientos a derecha e izquierda. Se dispone de cuatro biestables indicadores: cero (*Z*), signo (*S*), acarreo (*C*), y desbordamiento (*V*).

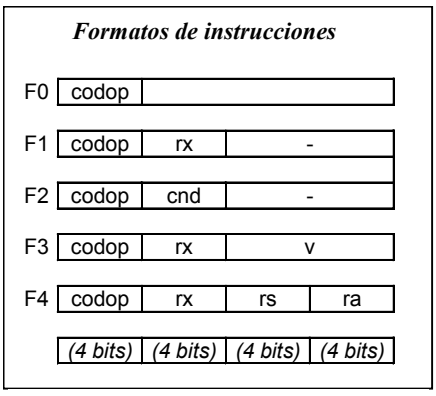

Figura 2: Formatos de las instrucciones de CODE-2

Especial cuidado se ha tenido en la planificación de los direccionamientos a memoria, que intervienen sólo en las instrucciones de carga (*LD*) y de almacenamiento (*ST*). En efecto, los direccionamientos se realizan sumando el contenido del registro *rD* y el campo *v* de la instrucción; siendo, por tanto, la dirección efectiva *d=rD+v*. De esta forma se puede tener, como casos particulares: direccionamiento directo (si *rD=H'0000*), direccionamiento indirecto a través de registro (si *v=H'00*), direccionamiento indexado (si *rD* hace las funciones de índice) y direccionamiento relativo a base (si *rD* hace las funciones de base y *v* las de desplazamiento).

En consecuencia consideramos que CODE-2 reúne todos los requisitos establecidos en [27] para un procesador pedagógico.

3. Repertorio de instrucciones máquina de CODE-2

El repertorio de instrucciones máquina de CODE-2 se muestra en la Tabla 1. Los nemotécnicos que se utilizan son los propuestos en el estándar IEEE 694 [28]. Aparte de las instrucciones de carga y almacenamiento (comentadas en la sección anterior) hay dos instrucciones de carga inmediata de registros, que utilizan el formato F3 (Figura 2): una de ellas (*LLI*) carga la parte baja del registro especificado en la instrucción y otra (*LHI*) la parte alta del registro. También se han incluido instrucciones específicas de entrada y salida, lo cual no impide que el profesor explique la posibilidad de realizar entradas y salidas utilizando direcciones del mapa de memoria. Las instrucciones de salto y de llamada a subrutina utilizan el formato F2, de forma que el segundo campo de 4 bits especifica si la bifurcación es incondicional *(cnd=0000)* o, en caso contrario la condición de salto (especificada con otros valores de *cnd*). La dirección de salto hay que almacenarla previamente en el registro *rD* del banco de registros.

Debido a la ortogonalidad de CODE-2, que divide todas las instrucciones en campos de 4 bits (1 cifra hexadecimal) es inmediato pasar de nemónicos a código hexadecimal; en efecto, la primera cifra hexadecimal identifica el *codop*, la segunda uno de los 16 registros de RF (0, 1,..., E,F) o la condición de salto, y los otros dos campos dos registros o un valor inmediato (*v*), en hexadecimal

Cuando se realiza un programa en lenguaje máquina para CODE-2, se recomienda al alumno cargar en los registros r0 y r1 los valores 0 y 1, respectivamente, con dos instrucciones *LLI*. Una vez almacenados estos valores, resulta inmediato llevar el contenido de un registro a otro (con la instrucción de suma, sumando el contenido de r0 al registro origen, y llevando el resultado al registro destino), o incrementar o decrementar el contenido de un registro (sumándole o restándole el contenido de r1, respectivamente).

Además del lenguaje máquina, se ha definido un lenguaje ensamblador, siguiendo el estándar IEEE694 [28] tanto para la definición de los nemónicos (que coinciden con los de la cuarta columna de la Tabla 1) como de las directivas que se han implementado (*ORG, EQU, DW, DR* e *INCLUDE*).

4. Montaje de CODE-2

El CODE-2, para ser utilizado por los alumnos como entrenador, se encuentra montado en un pequeño bastidor cuyo panel frontal se muestra en la Figura 2, y que contiene los siguientes elementos:

Tabla 1. Repertorio de instrucciones de CODE-2

Codop		**Nombre**	**Nemó-nico**	**Pára-metros**	**For.**	**Explicación**
binario	*Hex*					
0000	0	Cargar	LD	rx,[v]	F3	rx←M(rD+v)
0001	1	Almacenar	ST	[v],rx	F3	M(rD+v)←rx
0010	2	Carga inme.baja	LLI	rx,v	F3	rx(15:8)←H'00; rx(7:0)←v
0011	3	Carga inme. alta	LHI	rx,v	F3	rx(15:8)←v
0100	4	Entrada	IN	rx,IPv	F3	rx←IPv
0101	5	Salida	OUT	OPv,rx	F3	OPv←rx
0110	6	Suma	ADDS	rx,rs,ra	F4	rx←rs+ra
0111	7	Resta	SUBS	rx,rs,ra	F4	rx←rs-ra
1000	8	NAND	NAND	rx,rs,ra	F4	rx←(rs·ra)'
1001	9	Desplaza izquierda	SHL	rx	F1	C←rx(15), rx(i)←rx(i-1), i=15,...,1; rx(0)←0
1010	A	Desplaza derecha	SHR	rx	F1	C←rx(0), rx(i)←rx(i+1), i=0,...,14; rx(15)←0
1011	B	Desplaza arit. dch.	SHRA	rx	F1	C←rx(0), rx(i)←rx(i+1), i=0,...,14
1100	C	Salto	B-	cnd	F2	Si cnd se cumple, PC←rD
1101	D	Subrutina	CALL-	cnd	F2	Si cnd se cumple, rE←rE-1, M(rE)←PC, PC←rD
1110	E	Retorno	RET	-	F0	PC← M(rE); rE←rE+1
1111	F	Parar	HALT	-	F0	Parar

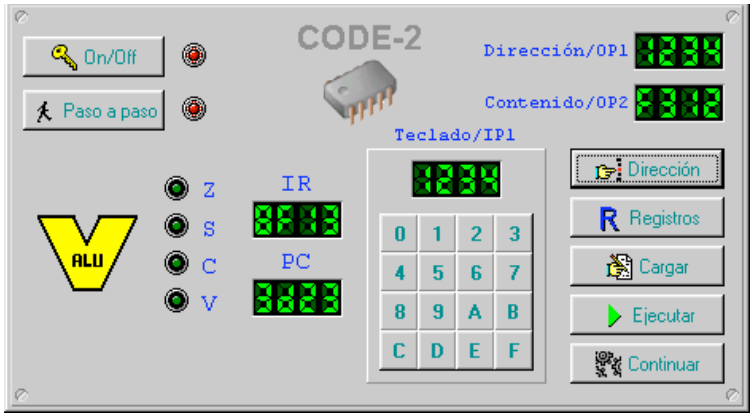

Figura 3. Panel de control de CODE-2

- *Interruptores* ON/OFF, para funcionar en Modo Paso a Paso, del que se puede salir con el pulsador CONTINUAR.
- *Teclado hexadecimal*, para introducción de programas y datos.
- *Puertos de salida 1 (OP1) y 2 (OP2).* La información se visualiza en hexadecimal.
- *Teclas de órdenes*: Dirección, Registros, Cargar, Ejecutar, y Continuar.
- *Pilotos* indicando el valor de los biestables indicadores de la ALU (*Z, S, C* y *V*) y visualizadores de los contenidos de los registros *IR* y *PC*.

Una vez activado el interruptor ON/OFF, bajo el control de las teclas de órdenes, pueden realizarse las siguientes tareas:

- *Seleccionar una posición de memoria previamente tecleada.* Con lo que en *OP1* aparece la dirección y en *OP2* su contenido.
- *Cargar un valor en una posición de memoria.*
- *Seleccionar los registros.*
- *Cargar un valor en un registro.*
- *Ejecutar un programa*.

Para gestionar el funcionamiento de las teclas de órdenes, la memoria de CODE-2 contiene un pequeño monitor, que se ubica en la zona ROM de la memoria principal.

5. Estructura y diseño de CODE-2

La Figura 4 muestra la estructura del procesador de CODE-2, cuyo funcionamiento se determina por medio de 29 señales de control. Se han diseñado dos unidades de control, una cableada y otra microprogramada.

El diseño de la unidad de control cableada se realiza partiendo de un diagrama de flujo del repertorio de instrucciones. Por otra parte realizamos una tabla con una lista de todas las microoperaciones (40) que hay que generar, y que se muestra parcialmente en la Tabla 2. A partir de las columnas segunda y tercera de la tabla resulta inmediato obtener las funciones de conmutación correspondientes a cada señal de control. Por ejemplo, para implementar la señal de control *b* se suman todos los términos que aparecen en la columna tercera correspondientes a las filas donde aparezca *b=1* en la segunda columna. A título de ejemplo, algunas de las 29 funciones de conmutación obtenidas se muestran en la Tabla 3.

La unidad de control se implementa según el modelo que se incluye en la parte derecha de la Figura 4. Un reloj actúa sobre un contador de 4 bits ("estado", en la figura) que va generando los distintos ciclos de cada instrucción.

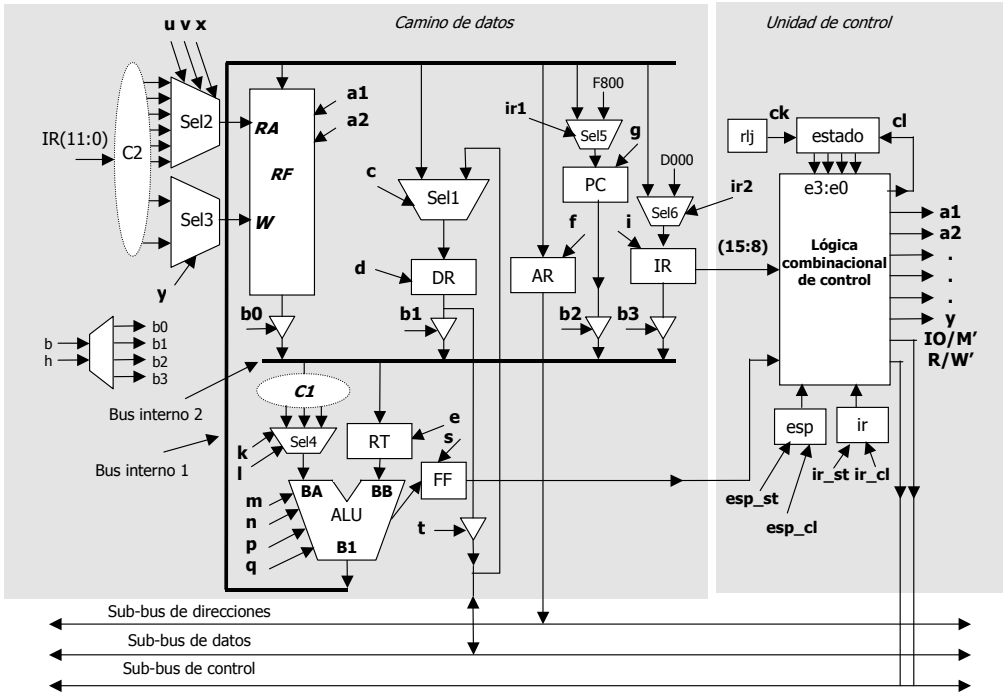

Figura 4. Esquema completo del procesador de CODE-2

En el ciclo en que finaliza una instrucción se hace *cl=1*, con lo que el contador pasa a generar otra secuencia de ciclos: c0, c1, c2, ... La Figura 5 muestra un esquema de la lógica combinacional de control, que contiene un decodificador de instrucciones, un decodificador de estado y un selector de condición, que, en las instrucciones de bifurcación, proporciona a su salida (FFc) un valor que es 1 si efectivamente hay que generar el salto. Con ayuda de este esquema, y la Tabla 2 es inmediato sintetizar las distintas señales de control. En la Figura 5, a título de ejemplo, se muestran los esquemas de las señales *k, c, R/W'* e *y*.

Más fácil de diseñar y describir resulta la unidad de control microprogramada, que sigue un esquema completamente clásico, y cuyos detalles pueden verse en [29].

7. Conclusiones

Un computador es un sistema de gran complejidad y cuya estructura y funcionamiento requiere el conocimiento de un gran número de conceptos. Debido a ello la mayoría de profesores de esta materia optan por utilizar máquinas ficticias donde explicar con claridad los conceptos más generales y básicos que se aplican a cualquier máquina real. El problema fundamental a la hora de seleccionar una máquina ideal estriba en buscar un compromiso adecuado entre cualidades didácticas (presentar sólo cuestiones esenciales) y complejidad (entrar en excesivos detalles).

En el presente trabajo proponemos un Computador Didáctico Elemental (CODE-2), que consideramos (Sección 2) reúne todas las características deseables en un procesador pedagógico para la enseñanza básica de la arquitectura de computadores [27]. Detalles adicionales sobre CODE-2 se encuentran en [29], y en la dirección web

http://atc.ugr.es/intro_info_mcgraw.html

se incluye un entorno didáctico completo (emulador, ensamblador, desensamblador, etc.), así como su diseño VHDL. En la actualidad se

Tabla 2. Lista parcial de las microoperaciones a implementar

Microoperaciones	Señales a generar	Situación en que debe generarse la microoperación
AR←H'00##IR(7:0)	b=h=1,k=0,l=1,m=n=p=q=0,f=1	c5·I4+c5·I5
AR←00'H##IR(7:0) +RT	b=h=1,k=m=0, n=p=1, q=0,f=1	c6·I0+c6·I1
AR←alu←PC	b=1,h=k=j=m=n=p=q=0, f=1	c1·ir'
AR←RF	a1=1,b=h=0, k=0,I=0,f=1	c6·ID·FFc+c5·IE
DR← alu ←PC	b=1,h=0, m=n=p=q=0, c=0, d=1	c7·ID·FFc+c7·IE
.....	.....	.....
RF←RT+1	m=p=q=0, n=1, a2=1	c6·IE
RT←PC	b=1,h=0,e=1	c2·ir'
RT←RF	a1=1,b=h=0,e=1	c5·(I0+I1+I6+I7+I8+IE)+c6·I3
WA←E'H	y=1	c5·ID·FFc+c6·IE
WA←rx	y=0	c5·(I2+I3+I9+IA+IB)+c6(I6+I7+I8) +c7·(I3+I4)+ c8·I0

esta procediendo al montaje de varios prototipos usando circuitos lógicos programables (FPGA).

CODE-2 se presenta en cursos de Ingeniería Informática y de Ingeniería Electrónica de la Universidad de Granada, según distintos niveles de complejidad y abordándose su diseño en profundidad de forma completa y utilizando diversas metodologías de implementación. Concretamente, en la primera de las titulaciones citadas, se considera a CODE-2 en las siguientes asignaturas:

- *"Introducción a los computadores"* (1.1, primer curso, primer cuatrimestre): Programación en lenguajes máquina y ensamblador, y utilización de CODE-2.
- *"Estructura de computadores"* (2.1): diseño completo de CODE-2 al nivel de micromáquina. Hay que hacer notar que en un cuatrimestre previo (1.2) el alumno ha cursado la asignatura *"Tecnología de Computadores I",* donde ha estudiado todos los elementos que intervienen en el diseño, que son los siguientes: decodificador binario, selector de datos, adaptador triestado, ALU, registro, biestable asíncrono, banco de registros, reloj y contador binario de 4 bits.
- *"Diseño de circuitos microelectrónicos"* (3.2): diseño VHDL de CODE-2
- *"Síntesis automática de arquitecturas VLSI"* (4.2): implementación de CODE-2 utilizando circuitos lógicos programables.

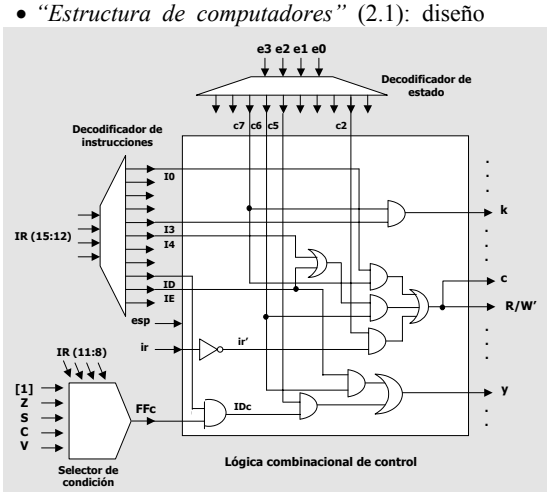

Figura 5. Esquema simplificado de la lógica combinacional de la unidad de control

En consecuencia, consideramos a CODE-2 como una potente herramienta didáctica, utilizable como caso práctico para presentar conceptos de distinto nivel de complejidad.

Referencias

[1] Y.Chu. *Digital Computer Design Fundamentals*. McGraw-Hill, 1962.

[2] G.W.Gorsline. *Computer Organization: Hardware/Software.* Prentice Hall, 1986.

[3] G.M.Karam, J.C.Bryant. *Principles of Computer Systems.* Prentice Hall, 1992.

[4] M.M. Mano. *Computer System Architecture.* Prentice Hall, 1976.

[5] M.M.Mano. *Arquitectura de computadores.* 3ª Ed. Prentice Hall, 1994.

[6] M.M.Mano, C.Kime. *Logic and computer design fundamentals.* Prentice Hall, 1997.

Tabla 3. Algunas de las 29 funciones de conmutación que sintetizan las señales de control de CODE

```
b= c1·ir'c2·ir'+ c3·ir'+ c5·(I2+I4+I5) + c6(I0+I1)+ c7·(I3+IDc+IE)
c= c2·ir'+c6·(I4+IE) + c7·I0
cl= c5·(I2+I9+IA+IB+IC+IF)+c6·(I6+I7+I8)+c7·(I3+I4+I5+IE)+c8·(I0+I1)+c9·ID
d= c2·ir'+ c6·(I5+I4+IE) + c7·(I0+IDc+IE+c7·IE)
ir_cl= c9·IDc·ir
k= c7·I3
l= c5·(I2+I3+I4 +I5)
x= c5·(I0+I1+IC·FFc)+ c6·(I6+I7+I8)+c9·IDc·ir'
y= c5·IDc+c6·IE
IO/M'= c6·I4+ c7·I5
R/W'= c
```


[7] C.C.Foster, *Computer Architecture,* 2nd Ed. Van Nostrand, 1976.

[8] D.Knuth. *The art of computer programming,* Volume I, 3rd Ed. Addison Wesley, 1997.

[9] G.Lee. *From Hardware to Software. An Introduction to Computers*. McMillan, 1982.

[10] A.S.Tanenbaum. *Structured Computer Organization.* 3rd Ed., Prentice Hall, 1990.

[11] J.L.Hennessy, D.Patterson. *Computer Architecture: A Quantitative Approach,* 2nd Ed. McGraw-Hill, 1996.

[12] J.D.Carpinelli. *Computer Systems. Organization & Architecture,* Addison Wesley, 2001.

[13] Y.N.Patt, S.J.Patel. *Introduction to computing systems,* McGraw-Hill, 2001.

[14] A.S. Tanenbaum, *Organización de computadoras. Un enfoque estructurado,* 4ª Ed. Pearson Educación, 2000.

[15] M.J. Murdocca, V.P. Heuring. *Principles of Computer Architecture,* Prentice Hall, 2000.

[16] D.A. Patterson, J.L.Hennessy. *Estructura y diseño de computadores*, (3 volúmenes) Editorial Reverté, 2000.

[17] P.M.Miguel, *Fundamentos de los computadores*. Paraninfo, 1999.

[18] A.Prieto, A.Lloris, J.A.Romera. *Realización de un ordenador didáctico elemental,* V Congreso de Informática y Automática. A.E.I.A; pp. 611-615. Madrid. 1982.

[19] F.J.Pelayo, A.Prieto, A.Lloris, F.Gómez-Mula. *Emulador y ensamblador de un ordenador didáctico elemental,* Rev. de Informática y Automática. Vol. 17, Nº 161, pp. 7-17. 1984.

[20] A.Prieto, A.Lloris, J.C.Torres. *Introducción a la Informática*. 1ª Ed. McGraw-Hill, 1989.

[21] A.Prieto, F.J.Pelayo, A.Lloris, F.Gómez-Mula. *Description and use of a simple didactic computer,* EC Newsletter (Education in Computing Computers in Education). Vol.2; N1 1, Jan-Apr 19-90, pp.17-29. 1990.

[22] E.Ayguadé, J.J.Navarro, M.Valero-García. *La màquina senzilla. Introducció a l'estructura básica d'un computador*. Col·lecció Aula. CPET, 1992.

[23] R.Hermida, A.M. del Corral, E.Pastor, F.Sánchez, *Fundamentos de computadores,* Síntesis, 1998.

[24] E.Pastor, F.Sánchez. *La Máquina Rudimentaria: un procesador pedagógico.* III Jornadas de Enseñanza Universitaria sobre Informática (JENUI'97), págs. 394-402, Junio 1997.

[25] J.Ruz. *De la tecnología a la arquitectura de computadores,* Síntesis, 1997.

[26] G.Fernández. *Conceptos básicos de arquitectura y sistemas operativos.* Sistemas y Servicios de Comunicación, 1998.

[27] F.Sánchez. *Características deseables en un procesador pedagógico para la enseñanza básica de la Arquitectura de Computadores.* Actas de las VII Jornadas de Enseñanza Universitaria de la Informática, pp. 68-73, Palma de Mallorca, 16-18 Julio 2001.

[28] *IEEE Standard for Microprocessor Assembly Language.* IEEE Std. 694-1985. The Institute of Electrical and Electronics Engineers, Inc. New York., June 30, 1985.

[29] A.Prieto, A.Lloris, J.C.Torres. *Introducción a la Informática*. 3ª Ed. McGraw-Hill, 2002.