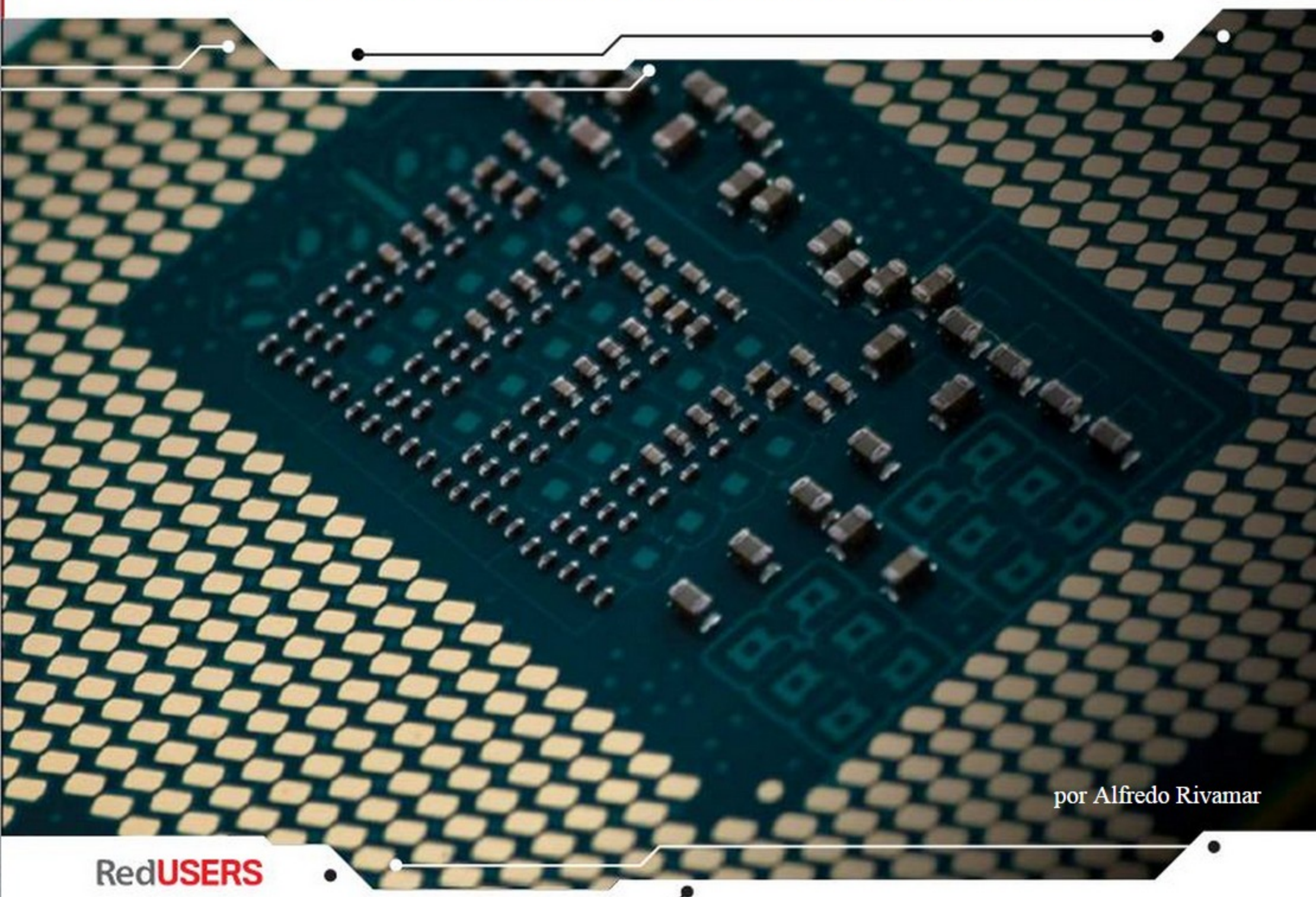
USERS

Fundamentos de microprocesadores

CONOCIMIENTOS Y HERRAMIENTAS INDISPENSABLES

Bases numéricas + Compuertas y circuitos lógicos + Microprocesadores de 8 y 16 bits +
Interfaces y direccionamiento + Set de instrucciones + Programación en lenguaje ensamblador

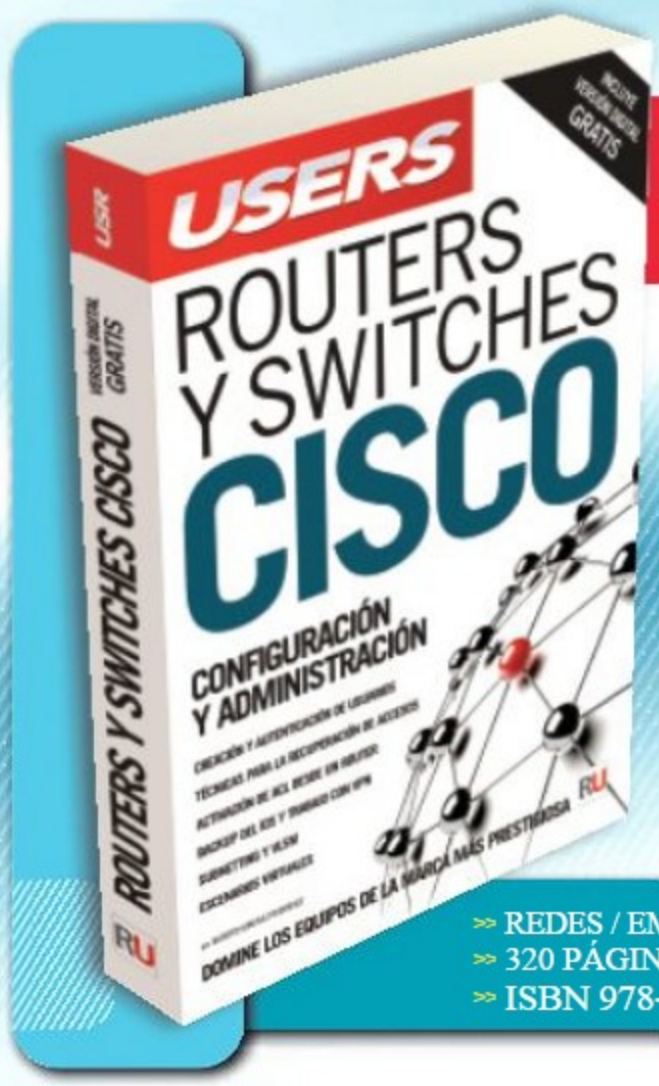


por Alfredo Rivamar

RedUSERS

COLECCIÓN HARDWARE AVANZADO

CONÉCTESE CON LOS MEJORES
LIBROS DE COMPUTACIÓN

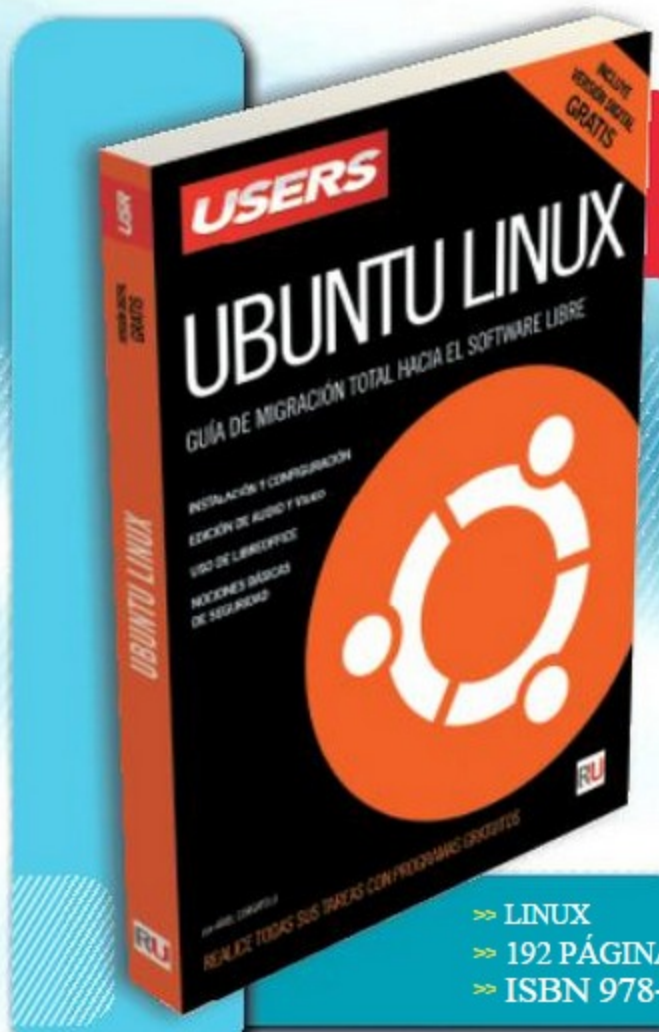


DOMINELOS EQUIPOSDELA MARCAMÁS PRESTIGIOSA

>> REDES / EMPRESAS
>> 320 PÁGINAS
>> ISBN 978-987-1949-34-2

CONOZCALOS SECRETOSDEL MUNDODELA ELECTRÓNICA

>> ELECTRÓNICA
>> 320 PÁGINAS
>> ISBN 978-987-1949-54-0



REALICETODAS SUSTAREASCON PROGRAMAS GRATUITOS

>> LINUX
>> 192 PÁGINAS
>> ISBN 978-987-1949-63-2

CONOZCALOS SECRETOSDEL MUNDODELA ELECTRÓNICA

>> ELECTRÓNICA
>> 320 PÁGINAS
>> ISBN 978-987-1949-55-7

LLEGAMOS A TODO EL MUNDO VÍA OCA* Y DHL**

MÁS INFORMACIÓN / CONTÁCTENOS

usershop.redusers.com +54 (011) 4110-8700 usershop@redusers.com

* SÓLO VÁLIDO EN LA REPÚBLICA ARGENTINA // ** VÁLIDO EN TODO EL MUNDO EXCEPTO ARGENTINA

FUNDAMENTOS DE MICROPROCESADORES

CONOCIMIENTOS Y HERRAMIENTAS INDISPENSABLES

por Alfredo Rivamar

RedUSERS

TÍTULO: Fundamentos de microprocesadores

AUTORES: Alfredo Rivamar

COLECCIÓN: Seriada

FORMATO: 24 x 17 cm

PÁGINAS: 192

ISBN 978-987-734-003-7

Rivamar, Alfredo

Fundamentos de microprocesadores. - 1a ed. - Ciudad Autónoma de Buenos Aires : Fox Andina; Buenos Aires: Dalaga, 2014.

192 p. ; 24x17 cm. - (Seriada; 13)

ISBN 978-987-734-003-7

1. Informática. I. Título

CDD 005.3

VISITENUESTRAWEB

EN NUESTRO SITIO PUEDE OBTENER, DE FORMA GRATUITA, UN CAPÍTULO DE CADA UNO DE LOS LIBROS EN VERSIÓN PDF Y PREVIEW DIGITAL. ADEMÁS, PODRÁ ACCEDER AL SUMARIO COMPLETO, LIBRO DE UN VISTAZO, IMÁGENES AMPLIADAS DE TAPA Y CONTRATAPA Y MATERIAL ADICIONAL.

RedUSERS
COMUNIDAD DE TECNOLOGIA

redusers.com

Nuestros libros incluyen guías visuales, explicaciones paso a paso, recuadros complementarios, ejercicios, glosarios, atajos de teclado y todos los elementos necesarios para asegurar un aprendizaje exitoso y estar conectado con el mundo de la tecnología.

LLEGAMOS A TODO EL MUNDO VÍA OCA* Y DHL**

* SÓLO VÁLIDO EN LA REPÚBLICA ARGENTINA / ** VÁLIDO EN TODO EL MUNDO EXCEPTO ARGENTINA

usershop.redusers.com usershop@redusers.com +54 (011) 4110-8700

Alfredo Rivamar

Desde niño, Alfredo Gabriel Rivamar es un apasionado admirador del fascinante mundo de la electrónica. Es ingeniero electrónico (UTN) y profesor de grado universitario en Ingeniería Electrónica (UNCuyo), licenciado en Administración de Empresas (UTN), especialista en Docencia Universitaria (UNCuyo) y MBA (UFV, España). Actualmente desarrolla su tesis de maestría en Teleinformática (UM-ITBA).

Es coautor de dos publicaciones y autor de otras dos relacionadas con la electrónica, las telecomunicaciones y la informática, incluyendo la obra que hoy usted tiene en sus manos.

Durante 10 años se ha desempeñado en empresas multinacionales y PyMes en el sector de las telecomunicaciones. Es, y ha sido por más de 20 años, docente en el nivel superior universitario y no universitario. En la actualidad, investiga sistemas basados en microcontroladores, y le interesan las redes de sensores inalámbricas (WSN) y la robótica aplicada a la educación.

Email: rivamara@yahoo.com

Dedicatoria

Este libro está dedicado a mis hijos Gabriel y María Angelina, espejos en los que me veo reflejado a diario, y quienes me enseñaron que ser padre es una tarea de todos y cada uno de los días en mi vida. A Martincito, en donde quiera que esté.

Agradecimientos

A todas y cada una de las personas que transitaron por mi vida, porque todas colaboraron a forjar este camino. A mis padres y a mi propia familia por guiarme, compartir los buenos momentos y contenerme a lo largo de todos estos años. Para Ana y Paula, de USERS, por su dosis de paciencia y por ofrecerme la oportunidad de contribuir como autor en esta prestigiosa editorial. A todos ellos, gracias.

Prólogo

Uno de los temas más importantes y complejos que enfrenté durante mi etapa de estudiante de Ingeniería Electrónica se relacionaba precisamente con el objeto de esta publicación, los microprocesadores. Este libro propone un acercamiento a los sistemas basados en estos dispositivos, analizando modelos de 8 y 16 bits como paso previo a la comprensión de microprocesadores más avanzados, de 32 y 64 bits. Aunque utilizamos un microprocesador genérico para presentar los conceptos y facilitar que el lector alcance sólidas bases respecto de los principios fundamentales que rigen el funcionamiento de estos dispositivos, también se ofrecen ejemplos específicos en 8 y 16 bits.

Entre los microprocesadores de 8 bits detallados: Zilog Z80, Motorola MC6800 e Intel 8085, se hace hincapié en este último por considerarlo suficientemente representativo, disponer de un set de instrucciones amplio, varios modos de direccionamiento, la sencillez suficiente para no complicar las simulaciones propuestas y porque, a partir de este, se comprenden con facilidad las arquitecturas de 16 bits expuestas.

Como no es necesario que el lector posea conocimientos previos en tecnología digital, en el capítulo 1 se presenta la arquitectura de los sistemas de numeración y, en el capítulo 2, el álgebra de Boole y los dispositivos básicos a partir de los cuales se construyen los microprocesadores. En el capítulo 3, se identifican los conceptos fundamentales para comprender cómo funciona un microprocesador y, en el siguiente, la arquitectura y las características de las interfaces microprocesador-memorias y E/S. El capítulo 5 presenta los fundamentos de la programación en lenguaje ensamblador y algunos ejemplos de aplicación. En los dos últimos capítulos, los conocimientos adquiridos se aplican a microprocesadores de 8 y de 16 bits, respectivamente.

Como autor, espero que estudiantes, aficionados y profesionales en sistemas que deseen comprender el funcionamiento interno de un microprocesador encuentren en este libro los conceptos fundamentales, reforzados mediante actividades de autoevaluación y ejercicios prácticos propuestos al final de cada capítulo.

Alfredo Gabriel Rivamar

El libro de un vistazo

Este libro está enfocado hacia quienes deseen ingresar en el campo de los microprocesadores más sencillos, de 8 y 16 bits, y tener una comprensión de los más avanzados, de 32 y 64 bits. En cada uno de los capítulos se presentan los conocimientos necesarios, se describe la lógica subyacente, se desarrollan los aspectos básicos de su programación en lenguaje ensamblador y se proponen ejemplos de aplicación para microprocesadores de 8 y 16 bits específicos.

*01 ARITMÉTICA DE MICROPROCESADORES

Los sistemas basados en microprocesadores utilizan números binarios mediante un sistema de numeración que emplea solamente los dígitos 0 y 1 llamados bits. En el primer capítulo conoceremos y detallaremos las bases numéricas, códigos y aritmética que sustentan el funcionamiento lógico de los microprocesadores.

*02 DISPOSITIVOS DIGITALES

En este capítulo conoceremos y comprenderemos los circuitos digitales básicos sobre los cuales se construyen los microprocesadores. Analizaremos el álgebra de Boole, las compuertas lógicas básicas, y las combinaremos para obtener nuevos circuitos digitales.

*03 DISPOSITIVOS BASADOS EN MICROPROCESADORES

En este apartado identificaremos los conceptos básicos para comprender cómo funciona un microprocesador analizando la organización de una computadora basada en microprocesador, sus partes, funcionamiento y aspectos característicos.

*04 INTERFACES DEL MICROPROCESADOR

En este apartado del libro presentaremos la arquitectura y las características de las interfaces del microprocesador con los dispositivos de memoria, el direccionamiento de la memoria, los puertos de E/S y las características eléctricas de un microprocesador.

*05 PROGRAMACIÓN EN LENGUAJE ENSAMBLADOR

En este capítulo presentaremos los fundamentos de la programación en lenguaje ensamblador y algunas aplicaciones básicas. Se presenta el modelo de programación, los modos de direccionamiento, el conjunto de instrucciones típicas, algunas técnicas y herramientas para el desarrollo de programas así como distintos ejemplos de programación en este lenguaje.

*06 MICROPROCESADORES DE 8 BITS

Aquí comenzaremos a aplicar los conocimientos adquiridos en los capítulos anteriores en microprocesadores de 8 bits específicos, y describiremos las tecnologías básicas en microprocesadores de 8 bits, su arquitectura y las familias de microprocesadores de 8 bits más populares.

*07 MICROPROCESADORES DE 16 BITS

En este capítulo continuaremos aplicando los conocimientos adquiridos en los capítulos anteriores en microprocesadores de 16 bits específicos, y describiremos las tecnologías básicas en microprocesadores de 16 bits, su arquitectura y las familias de microprocesadores de 16 bits más populares.

INFORMACIÓN COMPLEMENTARIA

A lo largo de este manual, podrá encontrar una serie de recuadros que le brindarán información complementaria: curiosidades, trucos, ideas y consejos sobre los temas tratados. Para que pueda distinguirlos en forma más sencilla, cada recuadro está identificado con diferentes iconos:

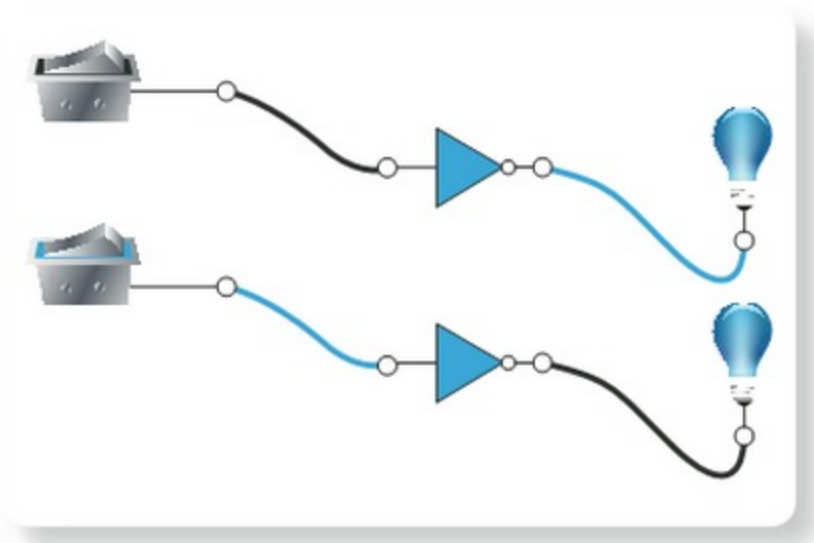

CURIOSIDADES E IDEAS


ATENCIÓN


DATOS ÚTILES Y NOVEDADES


SITIOS WEB

Contenido



*02

Dispositivos digitales

*03

Dispositivos basados en microprocesadores

*04

Interfaces del microprocesador

*05

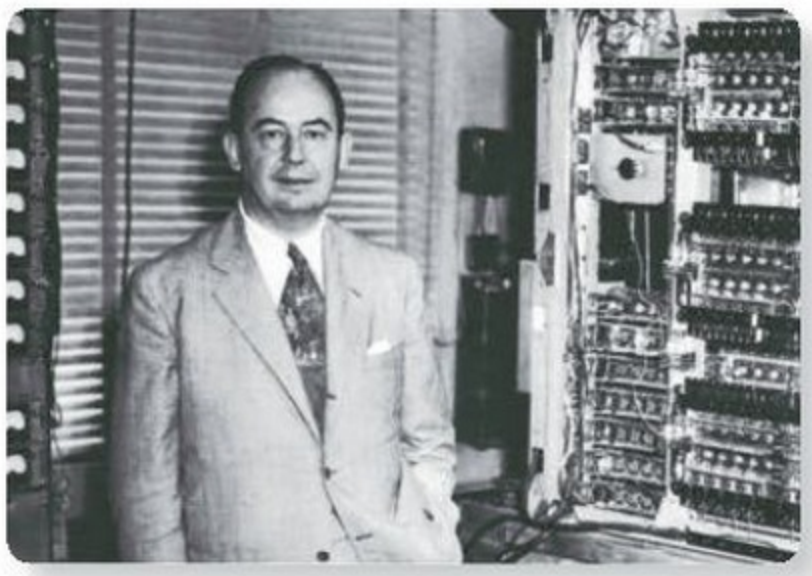
Programación en lenguaje ensamblador

*06

Microprocesadores de 8 bits

*07

Microprocesadores de 16 bits



RedUSERS

MEJORA TUPC

La reddeproductos sobre tecnologíamás importante del mundo de habla hispana

Libros

Desarrollos temáticos en profundidad

Coleccionables

Cursos intensivos con gran desarrollo visual

Revistas

Las últimas tecnologías explicadas por expertos

RedUSERS

redusers.com

Noticias al día
downloads, comunidad

Newsletters

El resumen de noticias
que te mantiene actualizado
Regístrate en redusers.com

RedUSERS PREMIUM

premium.redusers.com

Nuestros productos en versión digital con contenido ampliado y a precios increíbles

Usershop

usershop.redusers.com

El ecommerce de RedUSERS, revistas, libros y fascículos a un clic de distancia. Entregas a todo el mundo

Introducción

Toda tecnología suficientemente avanzada no se distingue de la magia.

Arthur C. Clarke, *Perfiles del futuro* .

Este libro ofrece una introducción al hardware y software de sistemas basados en microprocesadores. El microprocesador (CPU) es un dispositivo programable de propósito general que integra diversos productos: desde simples juguetes para niños hasta automóviles, desde robots y equipamiento industrial hasta equipos de ingeniería biomédica, y desde computadoras y dispositivos móviles inteligentes hasta sofisticados sistemas de control en energía nuclear e ingenios que intentan descubrir los secretos de otros planetas, satélites y estrellas. Luego, el desarrollo tecnológico dio lugar a dispositivos de propósito específico, los microcontroladores, que integran en un chip la CPU, y los circuitos necesarios para funcionar y conectarse con el mundo exterior.

Esta obra constituye una prolongación de los apuntes de clase que el autor desarrolló y mejoró durante varios años en relación con las ideas fundamentales sobre arquitectura de computadoras. Si bien se presentan los microprocesadores más simples, de 8 y 16 bits, su aprendizaje facilitará comprender el funcionamiento de los poderosos microprocesadores de 32 y 64 bits actuales con los que comparten arquitectura (Von Neumann).

Se organiza alrededor de varios conceptos: aritmética de microprocesadores, álgebra de Boole, fundamentos e interfaces de los microprocesadores, programación en lenguaje ensamblador y tecnologías básicas de microprocesadores de 8 y 16 bits, comunicados de manera clara y sencilla, según la secuencia de capítulos propuesta. También se recurre al uso de simuladores para entender el funcionamiento de un determinado microprocesador.

Comprender las funcionalidades de un microprocesador constituye una base de conocimientos útil para aprender sobre microcontroladores, y perfeccionar al lector en el diseño de aplicaciones muy variadas que faciliten su trabajo y, por qué no, mejoren la calidad de vida de las personas.

*01

Aritmética de microprocesadores

En este primer capítulo, abordaremos información general sobre números, códigos y aritmética de microprocesadores. Es necesario apropiarse de estos conocimientos para comenzar a comprender los sistemas comerciales basados en microprocesador, ya que estos son sumamente complejos.

Servicio de atención al lector: usershop@redusers.com

Sistema de numeración

Un sistema de numeración es el conjunto de símbolos y reglas que se utilizan para representar los datos numéricos o las cantidades. Se caracteriza fundamentalmente por su **base**, que es el número de símbolos distintos que utiliza, y el coeficiente que determina cuál es el valor de cada símbolo dependiendo de la posición (columna) que ocupe en el número. Los sistemas de numeración actuales son **posicionales**. Un ejemplo de sistema de numeración no posicional es el sistema de numeración egipcio, en el que los dígitos tienen el valor del símbolo utilizado, que no depende de la posición (columna) que ocupan en el número. En un sistema de numeración posicional, el valor relativo que representa cada cifra o cantidad depende de su valor absoluto y de la posición relativa (columna) que ocupa dicha cifra en el número. En cuanto a la **notación** , en algunos casos usaremos la notación númeroB para indicar que el número está expresado en base B. Por ejemplo, 1961_{10}.

El **sistema decimal** es un sistema posicional que utiliza la **base 10**, que corresponde al número de símbolos disponibles para la representación de cantidades; estos símbolos (o dígitos) son: **0**, **1**, **2**, **3**, **4**, **5**, **6**, **7** **8**, **9**.

Cada carácter se denomina **dígito**. Así, un determinado número decimal **N** se puede expresar de la siguiente forma:

$N = \sum (dígito_i * base_i)$

Para: i desde n hasta -m

Donde:

Base: 10.
i: posición respecto de la coma.
m: número de dígitos a la derecha de la coma.
n: número de dígitos a la izquierda de la coma, menos 1.
dígito: cada uno de los que componen el número.

Por ejemplo:

$23{,}1416_{10} = 2 * 10^1 + 3 * 10_0 + 1 * 10^{-1} + 4 * 10^{-2} + 1 * 10^{-3} + 6 * 10^{-4}$

La expresión matemática superior se basa en el **Teorema Fundamental de la Numeración** (**TFN**), que relaciona una cantidad expresada en cualquier sistema de numeración con la misma cantidad expresada en sistema decimal. Para comprender este teorema,

supongamos una cantidad expresada en un sistema cuya base es **B**. Representamos por Xi cada uno de los dígitos que contiene dicha cantidad, donde el subíndice indica la posición del dígito respecto de la coma decimal, posición que hacia la izquierda de la coma se numera desde 0 con un incremento de 1, y hacia la derecha se numera desde –1 con un incremento de –1.

El TFN dice que el valor decimal de una cantidad, expresada en otros sistemas de numeración, está dado por la expresión:

$$\ldots X_4 * B^4 + X_3 * B^3 + X_2 * B^2 + X_1 * B^1 + X_0 * B^0 + X_{-1} * B^{-1} + X_{-2} * B^{-2} + X_{-3} * B^{-3} \ldots$$

Donde el número en base B es:

$$\ldots X_4\ X_3\ X_2\ X_1\ X_0\ .\ X_{-1} X_{-2} X_{-3} \ldots$$

Numeración binaria

El **sistema binario** es un sistema en **base 2** y está formado por dos dígitos o símbolos: 0 y 1. A cada símbolo lo llamamos **dígito binario**, en inglés **bit** (*binary digit*). Del mismo modo que en el sistema decimal, combinando adecuadamente estos dos dígitos, podremos representar nuevas cantidades. Se trata de un sistema de numeración posicional en el que, dependiendo de la posición que ocupe el bit dentro del número binario, resultará su valor o peso. En un número binario entero, los pesos crecen de derecha a izquierda en potencias de 2. Así, el bit a la derecha es el bit menos significativo (**LSB**, *Least Significant Bit*) y tiene un peso de $2^0 = 1$.

El bit del extremo izquierdo es el bit más significativo (**MSB**, *Most Significant Bit*) y tiene un peso dependiente del tamaño del número binario. En números fraccionarios, el bit a la izquierda de la coma es el

MSB, y su peso es de 2^{-1}. Los pesos decrecen de izquierda a derecha en potencias negativas de 2.

Peso de la base: $2^{n-1}.... 2^4 2^3 2^2 2^1 2^0 , 2^{-1} 2^{-2} 2^{-3}..... 2^{-n}$

La ventaja de este sistema de numeración es su aritmética muy sencilla, por el hecho de que utiliza solo dos símbolos. Por otra parte, requiere de más posiciones para representar el mismo número que en el sistema decimal. Por ejemplo, el número 2 en decimal necesita una posición, mientras que el mismo número en binario, 10, requiere dos posiciones.

El sistema binario es el utilizado por las computadoras digitales, y sus dos valores lógicos posibles, 0 y 1, pueden ser representados físicamente de distintas maneras. Por ejemplo:

- Tensión eléctrica, alto y bajo.
- Interruptor, cerrado o abierto.
- Verdadero o falso.
- Corriente eléctrica, alta o baja.

El sistema de numeración binario produce números con muchas cifras que luego complejizarían demasiado la electrónica de los microprocesadores. Para evitarlo, se utilizan códigos intermedios de bases mayores y relacionados con la binaria, que permiten fácilmente transformar un número en base 2 a otra base que sea una potencia de 2 ($2^2 = 4$; $2^3 = 8$; $2^4 = 16$, etcétera), y viceversa. Por lo general, se utilizan como códigos intermedios los sistemas de numeración en base 8 (u octal) y en base 16 (o hexadecimal).

Numeración hexadecimal

La notación hexadecimal es muy utilizada para trabajar con los microprocesadores como método "resumido" para representar números binarios y, debido a que 16 es una potencia de 2 ($2^4 = 16$), resulta muy sencilla la conversión de los números del sistema binario al hexadecimal, y viceversa.

Al igual que el sistema decimal y el binario, el **sistema hexadecimal** es un sistema posicional de base 16. Los 16 símbolos que se utilizan para la representación de cantidades son: **0**, **1**, **2**, **3**, **4**, **5**, **6**, **7**, **8**, **9**, **A**, **B**, **C**, **D**, **E**, **F**.

BASE 16	
▼ SÍMBOLO	▼ VALOR ABSOLUTO
A	10
B	11
C	12
D	13
E	14
F	15

Tabla 1. Se asignan los siguientes valores (decimales) a los símbolos A-F.

Como ejemplos de números hexadecimales tenemos: 1AF, 3D, 17, 70B. La aritmética en este sistema de numeración es análoga a la de los anteriores.

Conversión entre sistemas de numeración

Como se explicó en la sección anterior, los sistemas de numeración son conjuntos de símbolos usados para representar cantidades; así se tienen los sistemas de numeración decimal, binario, octal, hexadecimal, etcétera. Estos se caracterizan por tener una base, que es un número de dígitos diferentes: diez, dos, ocho, dieciséis, respectivamente. Para el sistema decimal, se tienen 10 dígitos; en el sistema binario, 2 dígitos; en el sistema octal, 8 dígitos, y en el hexadecimal, 16 dígitos.

A continuación analizaremos de qué manera se realiza la conversión entre los sistemas numéricos más utilizados en las aplicaciones basadas en microprocesadores: decimal, binario y hexadecimal.

REPRESENTACIÓN DE SÍMBOLOS

¿Cuántos bits se necesitan para representar una cierta cantidad de símbolos diferentes? La cantidad es **2n**, donde el exponente **n** es el número de bits. Para representar 16 números diferentes necesitaremos 4 bits, ya que $2^4 = 16$. Así, para representar el equivalente binario a los números 0 a 15 en decimal, 6 números distintos, necesitará 4 bits.

Conversión binario-decimal

Para convertir un valor expresado en binario a decimal, basta con aplicar el TFN. Por ejemplo:

$1101_2 = 1*2^3 + 1*2^2 + 0*2^1 + 1*2^0 = 1*8+1*4+0*2+1*1=8+4+0+1=13_{10}$

Observamos que el valor en decimal de un número expresado en binario se obtiene sumando las potencias de 2 correspondientes a las posiciones de todos sus dígitos cuyo valor es 1.

Conversión decimal-binario

En la conversión de un número expresado en sistema decimal a otro número expresado en el sistema de numeración binario, se debe considerar tanto la parte entera del número decimal como su parte decimal, antes de realizar la conversión.

- **Parte entera del número** : se utiliza el método de las sucesivas divisiones por 2 para convertir un número entero en decimal a su respectivo número entero en binario. Se trata de dividir por 2 el número original y los sucesivos cocientes, hasta que el cociente de una división resulte con valor 0. La sucesión de todos los restos obtenidos, tomados en orden inverso, proporciona la representación binaria del número dado.

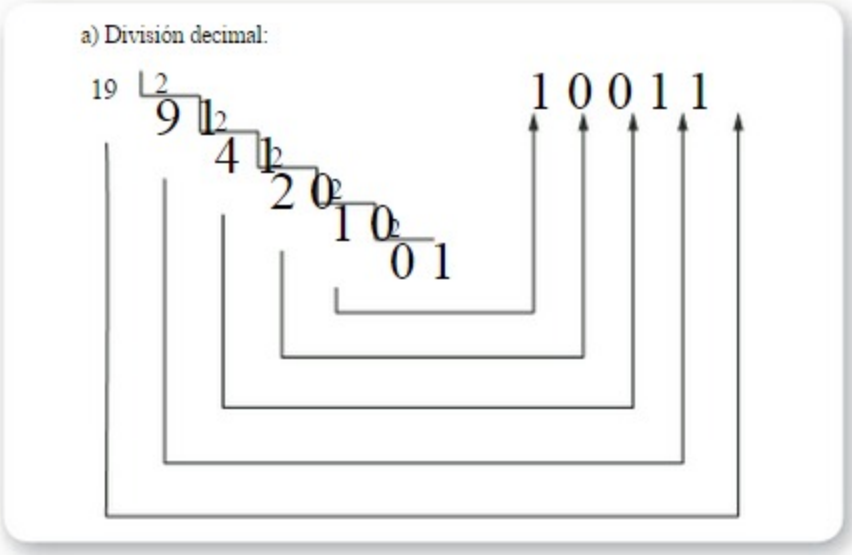


Figura 1. Al convertir el número 19_{10} a binario, resulta = 10011_2.

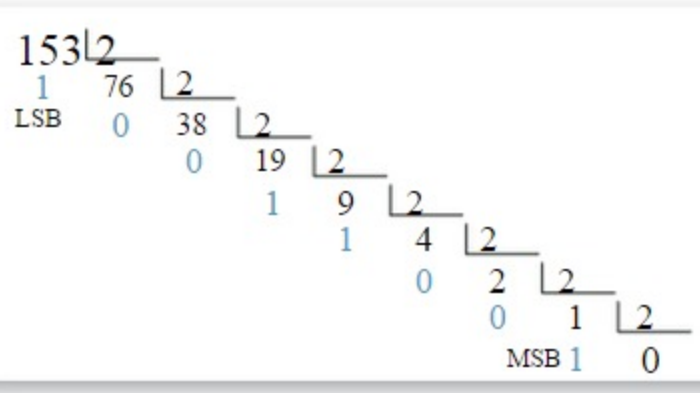


Figura 2. Al convertir el número 153_{10} a binario, resulta = 10011001_2.

- **Parte decimal del número** : el método de las sucesivas multiplicaciones por 2 se utiliza para convertir una fracción decimal en su equivalente fracción en binario. Consiste en multiplicar la fracción decimal dada por 2; la parte entera del resultado es el primer dígito binario de la fracción convertida. Luego se repite el proceso con la parte fraccionaria del resultado anterior, y se obtiene, en la parte entera del nuevo resultado, el segundo de los dígitos de la fracción buscada. Este proceso se repetirá hasta que se anule la parte fraccionaria de un resultado parcial o hasta obtener una cantidad de dígitos binarios que permitan no sobrepasar un error dado. Una fracción binaria, al igual que un decimal, puede tener un conjunto de dígitos que se repiten periódicamente.

				Verificación	
				Posición	Pot de 2
0,75 * 2 = 1,50	→	dígito 1	1	-1	0,50
0,50 * 2 = 1,00	→	dígito 2	1	-2	0,25
					0,75

Figura 3. Al convertir la fracción decimal 0,75 en fracción binaria, se obtiene 0.11_2.

				Verificación	
				Posición	Pot de 2
0,62 * 2 = 1,24	→	dígito 1	1	-1	0,5
0,24 * 2 = 0,48	→	dígito 2	0	-2	
0,48 * 2 = 0,96	→	dígito 3	0	-3	
0,96 * 2 = 1,92	→	dígito 4	1	-4	0,0625
0,92 * 2 = 1,84	→	dígito 5	1	-5	0,03125
0,84 * 2 = 1,68	→	dígito 6	1	-6	0,015625
0,68 * 2 = 1,36	→	dígito 7	1	-7	0,0078125
0,36 * 2 = 0,72	→	dígito 8	0	-8	
0,72 * 2 = 1,44	→	dígito 9	1	-9	0,0019531
0,44 * 2 = 0,88	→	dígito 10	0	-10	
0,88 * 2 = 1,76	→	dígito 11	1	-11	0,0004883
0,76 * 2 = 1,52	→	dígito 12	1	-12	0,0002441
					0,619873

Figura 4. Al convertir la fracción decimal 0,62 en fracción binaria con error $< 2^{-12}$, se obtiene 0.100111101011_2.

CONVERSIONES ONLINE

Además de utilizar la calculadora de las computadoras, podemos verificar el resultado de las conversiones en **Metricconversion.biz** (**http://metricconversion.biz/es/conversion-de-unidades.html**). Seleccionamos la opción Sistema de numeración, ingresamos el número por convertir, su sistema, el sistema al que deseamos convertir y hacemos clic en Conversión para obtener el resultado.

Para convertir un número con parte entera y fraccionaria, se deben aplicar los dos métodos anteriores, es decir:

- Convertir a binario la parte entera.
- Convertir a binario la parte decimal.
- Sumar ambos resultados.

Por ejemplo, convertir 13,25 en base 10 a su equivalente binario:

- Convertir 13 a binario: por lo visto anteriormente, $13_{10} = 1101_2$.
- Convertir 0,25 a binario: por lo que ya hemos visto, $0{,}25_{10} = 0.01_2$.
- Sumar los resultados de las conversiones parciales: $13{,}25_{10} = 1101.01_2$.

Para convertir $1101{,}11_2$ a decimal, la parte entera se resuelve como se ha visto antes y la parte decimal de la misma manera (como polinomio), pero utilizando exponentes negativos. Entonces:
$0{,}01_2 = 0 * 2^{-1} + 1 * 2^{-2} = 0 + 0{,}25 = 0{,}25_{10}$.

Conversión hexadecimal a decimal

También se aplica el TFN. Para convertir el número $2E7_{16}$ a decimal:
$2E7_{16} = 2 * 16^2 + E * 16^1 + 7 * 16^0 = 2 * 256 + 14 * 16 + 7 * 1 = 743_{10}$

Para los números en hexadecimal con parte fraccionaria, se emplean potencias de 16 con exponente negativo, de manera análoga a lo analizado para el sistema binario.

Conversión decimal a hexadecimal

Se emplea el método de las sucesivas divisiones por 16 para convertir números decimales enteros a hexadecimales. Se trata de dividir el número original y los sucesivos cocientes por 16 hasta que el cociente de una división resulte con valor 0. La sucesión de todos los restos obtenidos, tomados en orden inverso, proporciona la representación hexadecimal del número dado.

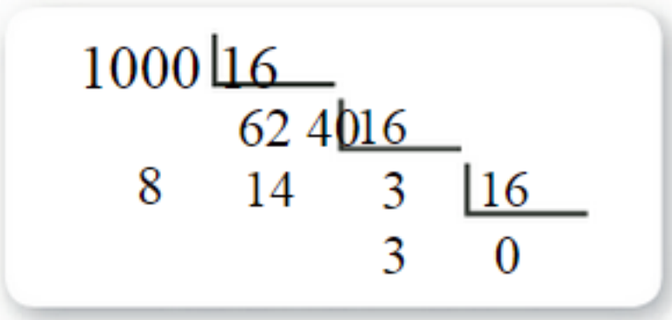


Figura 5. Al convertir el número decimal 1000 a hexadecimal, se obtiene $3E8_{16}$.

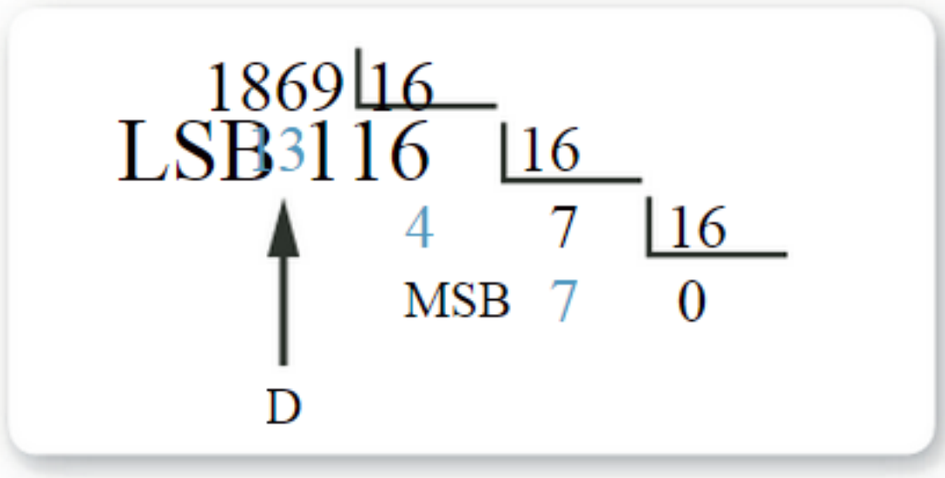


Figura 6. Al convertir el número 1869_{10} a hexadecimal, se obtiene $74D_{16}$.

Para convertir la fracción decimal en su equivalente fracción decimal, se emplea el método de las sucesivas multiplicaciones por 16. Esta consiste en multiplicar la fracción decimal dada por 16; la parte entera del resultado es el primer dígito de la fracción hexadecimal. Luego se repite el proceso con la parte fraccionaria del resultado anterior, y se obtiene en la parte entera del nuevo resultado el segundo de los dígitos de la fracción buscada. Este proceso se repite hasta que se anule la parte fraccionaria de un resultado parcial o hasta obtener una cantidad de dígitos que permita no sobrepasar un error dado.

Por ejemplo, convertir la fracción decimal 0,06640625 a hexadecimal:

0,06640625 * 16 = 1.0625

0,0625 * 16 = 1.00

Por lo tanto: $0{,}06640625_{10} = 0.11_{16}$

Combinando ambos métodos, es posible convertir números con parte entera y fraccionaria a hexadecimal.

Conversión hexadecimal a binario

Para convertir un número hexadecimal a binario, se reemplaza cada dígito hexadecimal por su representación binaria con 4 dígitos. En la tabla siguiente, se muestra la expresión binaria en cuatro dígitos de los dígitos hexadecimales.

b_{16} a b_2

DÍGITO HEXADECIMAL	DÍGITO BINARIO
0	0000
1	0001
2	0010
3	0011

▼ DÍGITO HEXADECIMAL	▼ DÍGITO BINARIO
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
A	1010
B	1011
C	1100
D	1101
E	1110
F	1111

Tabla 2. Equivalencia entre los sistemas hexadecimal y binario.

CONVERSIÓN BASE 16 A BASE 2				
▼ A	▼ 4	▼ .	▼ B	▼ 3
1010	0100	.	1011	0011

Tabla 3. Al convertir el número hexadecimal A4.B3 a binario, se obtiene 10100100.10110011_2.

Conversión binario a hexadecimal

Para convertir números binarios a hexadecimales, se realiza un proceso inverso al anterior. Se agrupan los dígitos binarios de 4 en 4 a partir del punto decimal hacia la izquierda y hacia la derecha, reemplazando cada cuarteto por su correspondiente valor hexadecimal.

CONVERSIÓN BASE 2 A BASE 16				
▼ 0001	▼ 0010	▼ 1100	▼ .	▼ 0110
1	2	C	.	6

Tabla 4. Al convertir el número 100101100.0112 a hexadecimal, se obtiene $12C.6_{16}$.

Observemos que, de ser necesario, se completan los cuartetos con ceros a izquierda o a derecha, según correspondan a la parte entera o fraccionaria, respectivamente.

EQUIVALENCIA ENTRE SISTEMAS

SISTEMA DECIMAL	SISTEMA BINARIO	SISTEMA HEXADECIMAL
0	0	0
1	1	1
2	10	2
3	11	3
4	100	4
5	101	5
6	110	6
7	111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F
16	10000	10
17	10001	11

Tabla 5. Tabla de equivalencias entre los primeros números de cada sistema de numeración.

Aritmética binaria

Si bien en un microprocesador se utiliza la numeración binaria dado que sus componentes electrónicos solo aceptan dos estados (encendido, 1; apagado, 0), en ocasiones es necesario emplear números con signo para representar tanto números positivos como negativos. Los números con signo tienen dos partes: **signo** y **magnitud**. Así, es necesario codificar el signo del número para representarlo mediante 0 y 1. En general, se utiliza el bit 0 para codificar el signo en números positivos y un 1 para números negativos.

De acuerdo a las diferentes formas de codificar la magnitud, se tienen tres sistemas de representación diferentes: **signo y magnitud** , **complemento a 1** y **complemento a 2** .

Notación en complemento a 2

A continuación describiremos el sistema de complemento a 2 porque permite simplificar los circuitos electrónicos del microprocesador. Cuando el número sea positivo, indicaremos su signo con un 1 y representará al número positivo igual que el número binario original (valor absoluto del número). Pero, si el número con signo es negativo, se utilizará la forma en complemento a 2 de ese número.

Para representar los números negativos en complemento a 2, en primer lugar se representa el número decimal dado en magnitud positiva; el número de magnitud positiva se representa en forma binaria positiva. Para obtener el complemento 1 del número binario logrado antes cambiamos los unos por ceros, y viceversa. Para encontrar el complemento a 2, se le suma uno al complemento 1.

Para simbolizar el número -5_{10} en binario utilizando el complemento a 2 y 4 bits, se requieren 3 bits para representar el número 5 y uno para representar el signo. Entonces, se representa el número decimal dado en magnitud positiva: –5 => 5; el número de magnitud positiva se representa en forma binaria positiva con 4 bits: 0101. El complemento 1 del número binario anterior se obtiene cambiando los unos por ceros y viceversa: 1010. Para el complemento a 2, se suma 1, y obtenemos el resultado: 1011.

Para decodificar un número representado en complemento a 2, teniendo en cuenta que el bit de más peso (MSB) representa el signo, debemos considerar lo siguiente. Si el primer bit de la izquierda es 0 y el número es positivo, el número representado es el equivalente del número binario que forma el resto de bits, pero, si el primer bit es 1 y el número es negativo, el número representado es el opuesto del equivalente decimal del número binario que forma su complemento a dos.

Aritmética en complemento a 2

El complemento a 2 de un número es importante ya que un microprocesador puede utilizarlo para complementar, incrementar y sumar números binarios. Los microprocesadores no tienen circuitos electrónicos para restar, en su lugar utilizan un sumador y números en complemento a 2 para realizar la sustracción. El proceso de sumar y restar es similar al utilizado para operar con números binarios, tal como se verá más adelante.

Suma en complemento a 2:

A-Sumar +5 y +3:

Sumando nº1	(+5)	00000101
Sumando nº2	+ (+3)	+ 00000011
Suma:	(+8)	00001000

B- Sumar +3 y -8:

Sumando nº1:	(+3)	00000011
Sumando nº2:	+ (-8)	+ 00001000
Suma:	(+5)	11111011

dado que:

C1 de (-8) = 11110111

C2 de (-8) = C1 + 1 = 11111000

Suma: 00000011 + 11111000 = 11111011

Figura 7. Ejemplo de suma en complemento a 2.

Resta utilizando suma en complemento a 2:

Minuendo: (-8) + 00001000

Sustraendo: (+5) = 00000101=> C2: 11111011

Diferencia: (+3) 1 00000011 ($=3_{10}$)

MSB: acarreo (se descarta)

Figura 8. Ejemplo de resta utilizando suma en complemento a 2.

Utilizando la representación en complemento a 2 y un sumador, el microprocesador puede realizar la sustracción.

Bits, nibbles y bytes

Como ya hemos explicado, el **bit** es una de las unidades de información junto con el **nibble** y el **byte**.

Un grupo de cuatro bits se denomina *nibble*. Con un nibble se presenta un número BCD, y también un nibble puede representar un dígito hexadecimal.

Un grupo de ocho bits u octeto se conoce como byte y es el agrupamiento de datos más importante para los microprocesadores, dado que cualquier referencia a una dirección de memoria no es menor a un byte y, por ende, se considera el dato localizable más pequeño. Los bits de un byte normalmente se numeran desde 0 hasta 7. Por otra parte, un byte está conformado por dos nibbles y, así, los bits 0, 1, 2 y 3 forman el nibble de menor orden, mientras que los bits 4, 5, 6 y 7 forman el nibble de mayor orden. Puesto que un byte está formado por dos nibbles, se puede representar cualquier valor mediante dos dígitos hexadecimales.

Operaciones aritméticas en el sistema binario

El sistema de numeración decimal dispone de dos operaciones básicas, la suma y la resta. A partir de estas se llevan a cabo la multiplicación y la división. La aritmética binaria es análoga a la aritmética decimal con la salvedad que este sistema se basa solo en dos dígitos (0 y 1).

Reglas suma binaria

+	0	1
0	0	1
1	1	0(*)

(*) "0" con acarreo "1"

Reglas resta binaria

-	0	1
0	0	1(*)
1	1	0

(*) "1" con préstamo

Figura 9. Reglas para sumar y restar en binario.

Para sumar en binario, las reglas básicas son las siguientes:
0 + 0 = 0
0 + 1 = 1
1 + 0 = 1
1 + 1 = 10 (1 es el acarreo)

El acarreo se produce porque, al sumar 1 + 1, el resultado en el sistema de numeración decimal sería 2_{10} que, en binario, se representa por 10_2, por lo que el resultado es 0 más el acarreo, que se indica con un 1 a la izquierda. Entonces: 1 + 1 = 10 (0 + acarreo 1).

PALABRA DE BIT

Una palabra es un grupo de 16 bits (el 0 es el bit de más bajo orden, y el 15, el más alto). Podemos dividir una palabra en dos bytes llamados igualmente de bajo y alto orden, y considerar a una palabra como un grupo de cuatro nibbles. Una palabra doble es un grupo de 32 bits. Para un número de bits mayor, la agrupación se denomina por su número de bits: de 64 bits, de 128 bits, etcétera.

a)
```
  1100
+ 0101
------
     1
```

b)
```
  1100
+ 0101
------
    01
```

c) acarreo 1
```
  1100
+ 0101
------
   001
```

d) (acarreo) 1
```
  1100
+ 0101
------
  0001
```

e)
```
  1100
+ 0101
------
 10001
```

Figura 10. Pasos y resultado de sumar los números binarios 1100 y 0101.

```
  1 0 0 1 0 0        1 1 0 0 1
+                  +
    1 0 0 1 0          1 0 0 1 1
-----------        -----------
  1 1 0 1 1 0        1 0 1 1 0 0
```

Figura 11. Dos ejemplos adicionales de suma binaria.

Si aplicamos el TFN, se podría calcular el valor decimal de estas cifras y comprobar que, en decimal, las operaciones planteadas son, respectivamente:

36+18=54 y 25+19=44

En una resta binaria (por ejemplo de 0 – 1), el resultado sería negativo por lo que se pide el préstamo de los siguientes dígitos, tal como se haría en una resta en decimal.

a)
```
    0
  1110
- 0101
------
     1
```

b)
```
  1100
- 0101
------
    01
```

c)
```
  1100
- 0101
------
   001
```

d)
```
  1100
- 0101
------
  1001
```

Figura 12. Pasos y resultado de restar los números binarios 1110 y 0101.

```
  1 1 1 1 1 1        1 1 1 1 0 0
-                  -
  1 0 1 0 1 0        1 0 1 0 1 0
-------------      -------------
  0 1 0 1 0 1        0 1 0 0 1 0
```

Figura 13. Dos ejemplos adicionales de resta binaria.

Al calcular el valor decimal de estas cifras se comprueba que, en decimal, las operaciones propuestas son:

63 – 42 = 21 y 60 – 42 = 18

Para la multiplicación y división de números binarios, se opera exactamente igual que con los números decimales, pero tomando en cuenta que las sumas y restas se harán en binario.

Multiplicación binaria

Para la multiplicación en binario se aplica el mismo procedimiento seguido para multiplicar números en el sistema decimal.

MULTIPLICACIÓN BINARIA

MULTIPLICANDO A	MULTIPLICADOR B	MULTIPLICACIÓN: A*B
0	0	0
0	1	0
1	0	0
1	1	1

Tabla 6. En multiplicación binaria es necesario considerar esta tabla de multiplicación binaria.

a) Multiplicación decimal:

```
        1   1
    x   1   3
    -------
        3   3
    1   1
    -------
    1   4   3
```

b) Multiplicación en binario:

```
            1   0   1   1
        x   1   1   0   1
        ---------------
            1   0   1   1
        0   0   0   0
    1   0   1   1
    ---------------------
1   0   0   1   1   1   1
```

Figura 14. Multiplicación de dos números en sistema decimal y en sistema binario. Se observa que: $143_{10} = 10001111_2$.

Para el caso de una multiplicación con signo, se representan los operandos en complemento 2, y el resultado también se obtiene en complemento a 2. El último multiplicando desplazado se niega.

División binaria

Con el análisis de esta operación, finaliza la presentación de las cuatro operaciones básicas en sistema binario. Para la división en binario, se aplica el mismo procedimiento seguido para dividir números en el sistema decimal, aunque se multiplica y resta en binario.

a) División decimal:

```
 7 | 2
   | 3 1
```

b) División en binario:

```
111 | 10
-10 | 11
 11
-10
  1
```

Figura 15. División de dos números en sistema decimal y en sistema binario.

Códigos binarios y códigos alfanuméricos

Se entiende por **código** a toda representación unívoca de cantidades, de forma tal que a cada una de ellas se le asigna una determinada combinación de símbolos, y viceversa. En este sentido, los sistemas de numeración analizados antes constituyen una forma de representación de cantidades o códigos.

SUMAR NÚMEROS DE VARIOS DÍGITOS

En el caso de sumar dos números de varios dígitos, se opera como en los números decimales, sumando dígito a dígito desde los de la derecha y con el acarreo en caso de haber un 1 + 1. El acarreo debe ser sumado en la siguiente columna a la izquierda.

Código BCD

En las aplicaciones que utilizan dispositivos microprocesadores o microcontroladores, resulta de particular interés desarrollar los cálculos en decimal y no en binario. Esto se debe a que el sistema de numeración decimal es el que utilizan los seres humanos, aunque internamente los microprocesadores y microcontroladores utilizan el sistema de numeración binario representado por dos dígitos binarios: 0 para la condición lógica cerrado y 1 para la condición lógica abierto. Para ello, debe disponerse de una representación interna, de manera tal que ciertos conjuntos sean entendidos como dígitos decimales, y también para tener la posibilidad de operar en forma aritmética con ellos.

Uno de los modos más comunes de representación de dígitos decimales se da mediante la utilización del código denominado **BCD** (en inglés, *Binary Coded Decimal*, o decimal codificado en binario), en el que cada dígito decimal tiene asignada una determinada combinación de 0 y 1. Los códigos BCD pueden ser **ponderados** o **no ponderados**. Un código es ponderado, cuando cada bit tiene un valor diferente dependiendo de la posición que ocupe, y es no ponderado en caso contrario. Dos ejemplos de códigos BCD ponderados son el código **BCD natural** y el código **BCD Aiken**, que se caracterizan porque, a la posición de cada bit, se le asigna un peso, y el número decimal equivalente a una combinación binaria se obtiene sumando los pesos de las posiciones que poseen valor 1.

En el caso del código BCD natural, se asocia a cada dígito su valor en binario puro, y se utiliza la representación binaria en cuatro bits para cada dígito, es decir, cada número decimal se convierte al código BCD sustituyéndolo por la correspondiente combinación binaria, de entre las que se muestran en la tabla siguiente:

EQUIVALENCIA

NÚMERO DECIMAL	CÓDIGO BCD NATURAL PESO			
	8	4	2	1
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0

NÚMERO DECIMAL	CÓDIGO BCD NATURAL PESO			
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1

Tabla 7. Equivalencia entre numeración decimal y código BCD natural (8421).

Observamos que los grupos de cuatro bits representan un solo dígito decimal. Es decir, si se tiene un número decimal de tres dígitos, puede expresarse como tres grupos de cuatro bits cada uno, en el que cada grupo representará el valor de cada dígito decimal. Por ejemplo, así se representa el número $327_{10} = 001100100111$.

Códigos de paridad

Estos códigos son sumamente útiles para detectar errores en la transmisión de datos. Se agrega a cualquiera de los códigos anteriores un **bit de paridad** que puede ser **par** o **impar**. En un código de paridad par, el bit de paridad será 1 y hará que la cantidad de bits de cada combinación sea par, mientras que, en un código de paridad impar, el bit de paridad será 1 para hacer que el número 1 de cada combinación sea impar.

PARIDAD			
NÚMERO DECIMAL	CÓDIGO BCD NATURAL PESO	BIT PARIDAD PAR	BIT PARIDAD IMPAR
	8 4 2 1		
0	0 0 0 0	0	0
1	0 0 0 1	1	0
2	0 0 1 0	1	0
3	0 0 1 1	0	1
4	0 1 0 0	1	0
5	0 1 0 1	0	1
6	0 1 1 0	0	1
7	0 1 1 1	1	0
8	1 0 0 0	1	0
9	1 0 0 1	0	1

Tabla 8. Códigos de paridad par e impar.

Códigos alfanuméricos

Son códigos que superan la limitación de los analizados antes, en el sentido de que, con aquellos, no es posible representar letras y signos, sino solamente números. Dos de los códigos para la representación interna de caracteres alfanuméricos son:

- El **código ASCII** (*American Standard Code for Information Interchange* o código estándar americano para intercambio de información).
- El **código EBCDIC** (*Extended Binary Coded Decimal Interchange Code* o código extendido de binario codificado decimal).

El código ASCII es un código binario ampliamente usado para la transmisión de información, para codificar los caracteres de un teclado, así como los que debe imprimir una impresora o mostrar una pantalla.

Es interesante recordar que, en el teclado de una computadora, los caracteres imprimibles que figuran son:

- 54 letras (27 mayúsculas y 27 minúsculas en español).
- 10 dígitos (0 al 9).
- Signos de puntuación y operación (coma, punto, punto y coma, menos, más, igual, barra de división).
- Caracteres especiales (por ejemplo, %, $, #, { }, [ ]).

Además, coexisten teclas de control de impresión para órdenes mecánicas, como la barra espaciadora (**SP**, por *Space*), la tecla **ENTER** de retorno de carro a un nuevo renglón (**CR**, por *Carry return*), etcétera. Estas teclas sirven para organizar la impresión de caracteres en los renglones de un papel o de una pantalla. También existen caracteres de control usados en teleprocesamiento, como **ACK** (*Acknowledge* o aviso de mensaje recibido), **BEL** (*Bell* o aviso por señal sonora), **ETX** (*End of text* o fin de texto), **STX** (*Start of text* o comienzo de texto), etcétera.

Los **códigos ASCII ampliados** o **extendidos** utilizan los ocho bits para codificar una serie de caracteres gráficos especiales (trazos), caracteres científicos (letras griegas y otros símbolos especiales) y caracteres con tilde (vocales con tilde, ñ, etcétera). Son redefinibles, no normalizados.

En el código ASCII, las letras minúsculas y las letras mayúsculas están codificadas en dos sucesiones ordenadas de números binarios, lo que permite realizar ordenamientos alfabéticos.

CARACTERES Y CÓDIGOS

CARÁCTER	ASCII 8 bits	HEXADECIMAL	DECIMAL
A	01000001	41	65
B	01000010	42	66
C	01000011	43	67
--------	--------	--------	--------
Z	01011010	39	90

Tabla 9. Distintos caracteres y sus equivalencias.

La tabla muestra el código ASCII normalizado, en el que cada carácter tiene dos coordenadas que, escritas en forma consecutiva, conforman el equivalente hexadecimal del código binario.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NUL	SC	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI
1	DLE	DC1	DC2	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US	
2	SP	!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[		]	^	_
6	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	DEL

Figura 16. Código ASCII normalizado.

Así, el código ASCII del carácter **A** estará dado por las coordenadas 4 (horizontal) y 1 (vertical), que forman $41_{16} = 0100\ 0001$.

RESUMEN

En este primer capítulo explicamos que las computadoras utilizan internamente el sistema de numeración binario. Aprendimos acerca de las operaciones en distintos sistemas de numeración y reconocimos el bit como símbolo fundamental del sistema de numeración binaria (base de distintos códigos y del **álgebra de Boole**, tema que se tratará en el próximo capítulo) así como distintos códigos ponderados, no ponderados y alfanuméricos para representar combinaciones de dígitos binarios.

Actividades

TEST DE AUTOEVALUACIÓN

1 ¿Qué es un sistema de numeración?

2 ¿Cuáles son las características del sistema de numeración binario?

3 Realice una tabla para los sistemas de numeración indicados en el punto 1, en los que relacione las primeras quince cantidades de cada uno a modo de equivalencia.

4 Dé un ejemplo de suma binaria y explique el procedimiento para realizarla.

5 Dé un ejemplo de resta binaria y explique el procedimiento para realizarla.

6 ¿En qué consiste la conversión de decimal a binario?

7 Explique el procedimiento para convertir una fracción decimal a su equivalente binario.

8 ¿En qué consiste la representación numérica por complemento a 2?

9 ¿Qué es el código ASCII extendido?

EJERCICIOS PRÁCTICOS

1 Convierta el número 467_{10} a su equivalente hexadecimal.

2 Convierta el número $57{,}3_{10}$ a su equivalente binario.

3 Convierta el número $01001110{,}11_{2}$ a su equivalente decimal.

4 ¿Cuál es el complemento a 1 de 010001011011_2 en hexadecimal?

5 ¿Cuántos bits se necesitan, al menos, para codificar 1450 símbolos distintos?

PROFESOR EN LÍNEA

Si tiene alguna consulta técnica relacionada con el contenido, puede contactarse con nuestros expertos: profesor@redusers.com.

*02

Dispositivos digitales

Presentaremos algunos dispositivos electrónicos básicos, como las compuertas y los bloques que conforman los circuitos lógicos. Estos últimos pueden considerarse un conjunto de dispositivos que manipulan, de una manera determinada, las señales electrónicas que les llegan (señales de entrada), y generan como resultado otro conjunto de señales (señales de salida).

Servicio de atención al lector: usershop@redusers.com

Álgebra de Boole

La relación que existe entre el álgebra de Boole y los sistemas de cómputo es uno a uno entre las funciones booleanas y los circuitos electrónicos digitales. Para cada función booleana es posible diseñar un circuito electrónico, y viceversa, y como las funciones booleanas solo requieren de los operadores AND, OR y NOT, podemos construir circuitos electrónicos digitales, base de los microprocesadores, utilizando exclusivamente estos operadores mediante las compuertas lógicas homónimas.

El inglés George Boole formalizó matemáticamente la lógica en 1854 cuando definió lo que hoy se conoce como **álgebra de Boole**. De hecho, en este capítulo se presenta el álgebra de Boole binaria, un caso particular; dado que, en general, el conjunto de un álgebra de Boole puede tener más de dos elementos. En 1938, Claude E. Shannon definió el modo de comportamiento de los circuitos lógicos sobre el fundamento del álgebra de Boole y creó el **álgebra de conmutación** . Al igual que en el álgebra se tienen variables y funciones, aunque ahora las variables representarán estados lógicos **0** o **1**.

Un álgebra de Boole es una entidad matemática formada por un conjunto que contiene dos elementos, unas operaciones básicas sobre estos elementos y una lista de axiomas que definen las propiedades que cumplen las operaciones.

Las siguientes son las operaciones booleanas básicas:

- negación o complementación o **NOT**;
- producto lógico **AND**;
- suma lógica **OR**.

La negación corresponde a la partícula **no** y se representa, en general, mediante una comilla simple ’. Por lo tanto, la expresión **x**’ denota la negación de la variable **x**, y se lee **no x**.

ELEMENTOS DE UN ÁLGEBRA DE BOOLE

Un álgebra de Boole o álgebra booleana está compuesta únicamente por dos elementos. Estos elementos se pueden denominar: **falso** y **verdadero** o, en general, **0** y **1**. De esta manera, una variable booleana o variable lógica puede tomar los valores 0 o 1.

El producto lógico que corresponde a la conjunción y de la lógica se representa con el símbolo ·. Entonces, si **x** e **y** son variables lógicas, la expresión x y denota su producto lógico y se lee **x** e **y**.

Finalmente, la suma lógica corresponde a la conjunción o y se representa con el símbolo +. Por lo que la expresión **x** + **y** denota la suma lógica de las variables **x** e **y**, y se lee **x** o **y**.

Todas las operaciones lógicas indicadas se pueden representar mediante una tabla, **tabla de verdad**, en la que, a la izquierda de una línea vertical, se indican todas las combinaciones posibles de las variables de entrada y, a su derecha, cada uno de los resultados de la operación para cada combinación. Como se verá más adelante, tanto los axiomas como los teoremas son muy importantes en la simplificación de expresiones algebraicas.

Postulados del álgebra de Boole

Los axiomas o postulados que describen el comportamiento de las operaciones booleanas permiten demostrar una serie de leyes o teoremas sumamente útiles para operar con expresiones algebraicas booleanas. Los postulados, dadas las variables lógicas **a**, **b** y **c**, son: las propiedades conmutativa, asociativa y distributiva; elementos neutros y complementación. Se enuncian de esta manera:

Ambas operaciones booleanas son conmutativas:

$a + b = b + a$

$a \cdot b = b \cdot a$

Se cumple la propiedad asociativa:

$a + (b + c) = (a + b) + c$

$a \cdot (b \cdot c) = (a \cdot b) \cdot c$

Se cumple la propiedad distributiva:

$a \cdot (b + c) = a \cdot b + a \cdot c$

$a + (b \cdot c) = (a + b) \cdot (a + c)$

Precaución: la segunda expresión no se cumple en la aritmética entera.

Existen dos elementos neutros 0 y 1 que cumplen la **propiedad de identidad** respecto de cada una de las operaciones booleanas básicas:

$a + 0 = a$
$a \cdot 1 = a$

Por la complementación se cumple que:
$a + a' = 1$
$a \cdot a' = 0$

Estas expresiones indican la imposibilidad de que las variables a y a' tomen simultáneamente el valor 0 en el primer caso y 1 en el segundo. De estas restricciones surge que si:

$a = 0 \Rightarrow a' = 1$

o si: $a = 1 \Rightarrow a' = 0$ (siendo a' la variable inversa de a.)

Teoremas del álgebra de Boole

Dadas las variables lógicas a, b y c en base a los postulados enunciados antes se verifican los siguientes teoremas:

1. Principio de dualidad : toda identidad deducida a partir de los axiomas continúa siendo válida si las operaciones + y ·, y los elementos **0** y **1** se intercambian entre sí.

2. Ley de idempotencia : para cada elemento a de un álgebra de Boole se verifica que:
$a + a = a$
$a \cdot a = a$

3. Ley de absorción :
$a + a \cdot b = a$
$a \cdot (a + b) = a$

4. Ley de dominancia : para cada elemento de un álgebra de Boole se verifica:
$a + 1 = 1$
$a \cdot 0 = 0$

5. Ley de involución : para todo elemento a'' de un álgebra de Boole se verifica:
$a'' = a$

6. Leyes de De Morgan: en toda álgebra de Boole se verifican:
$(a + b + c)' = a' \cdot b' \cdot c'$
$(a \cdot b \cdot c)' = a' + b' + c'$

Un buen ejercicio consiste en comprobar que se cumplan estas leyes, utilizando una tabla de verdad y analizando la salida en función de la combinación de tres variables de entrada.

Simulación de expresiones booleanas

Para simular y construir expresiones booleanas es posible utilizar varias aplicaciones, algunas de ellas libres y otras con versión de evaluación y funcionalidad reducida.

Por ejemplo, **Logic Friday** es un software libre para optimizar, analizar y sintetizar funciones lógicas. Una de sus características más interesantes es que permite minimizar funciones lógicas y sintetizar diagramas de compuertas lógicas. Se descarga desde el sitio web **http://sontrak.com/index.html** .

Otro software que enseña lógica de compuertas y circuitos digitales de una manera efectiva es **Logicly**. Ofrece una versión de evaluación que luego de finalizada hace necesario que se adquiera la licencia de uso. Se descarga desde el sitio web **http://logic.ly** .

NI Multisim , de *National Instruments* , es un entorno de simulación **SPICE** estándar en la industria. Es una herramienta muy interesante para la enseñanza de construcción de circuitos, para construir experiencia a través de la aplicación práctica del diseño, generación de prototipos, simulación y pruebas de circuitos eléctricos. Se descarga desde el sitio web **http://ni.com/multisim/esa** endistintas versiones.

Proteus es una compilación de programas de diseño y simulación electrónica desarrollados por Labcenter Electronics. Consta de dos

VALIDEZ DE LOS TEOREMAS

Es importante destacar que se puede demostrar la validez de cada teorema mediante la implementación de una tabla verdad y verificando que el resultado de cada combinación en su entrada entregue en la salida un resultado acorde con los postulados de cada teorema.

programas principales: **Ares** e **Isis**, y los módulos VSM de simulación y Electra. La versión demo de Proteus 8.1 se descarga desde **http://labcenter.com/index.cfm** .

Compuertas lógicas

El sistema de numeración binaria, o de base 2, solo utiliza los dígitos 0 y 1, conocidos como dígitos binarios, números binarios o bits.

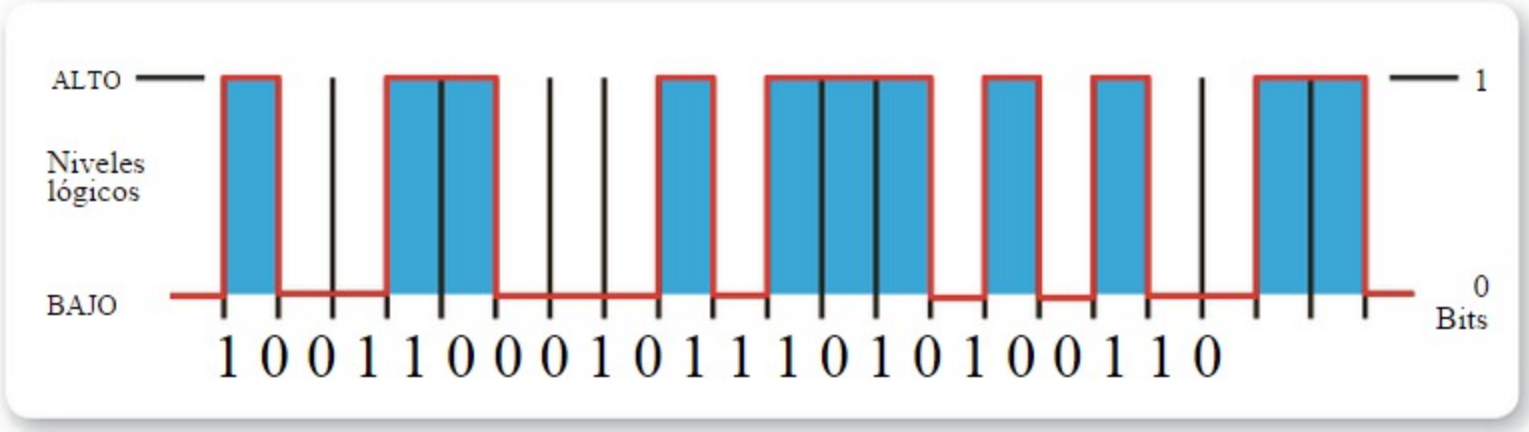


Figura 1. Tren de pulsos típico que representa los datos digitales donde se observan valores lógicos altos, 1, y valores lógicos bajos, 0.

En general, el estado lógico 1 representa una tensión alta, por ejemplo 5 volts, conocido como **nivel lógico** o **valor alto**. Por otro lado, el estado lógico 0 representa una tensión baja como 0 volt o tierra y nos referimos a él como **nivel lógico** o **valor bajo**. Cada uno de estos valores 0 o 1 se denomina **digito binario** (en inglés, *Binary digit*) o **bit**.

Definición, tipos y simbología

Todas las operaciones lógicas pueden ser realizadas por medio de elementos físicos de diferentes tipos (mecánicos, eléctricos, neumáticos o electrónicos), que admiten entradas binarias o lógicas y que devuelven una respuesta (salida) también binaria o lógica.

Los dispositivos con los cuales se implementan las funciones lógicas son llamados **puertas lógicas** o **compuertas lógicas** y están basados en **transistores**, los componentes electrónicos más elementales. Los tipos básicos de compuertas lógicas son tres: **OR** (**O**), **AND** (**Y**) y **NOT** (**inversor**), cada una representa una operación o función lógica básica.

La compuerta AND permite representar la misma función y se relaciona con la **conjunción lógica**. La compuerta OR permite representar la misma función y se relaciona con la **disyunción lógica**, mientras que la compuerta NOT representa la misma función y se relaciona con la **inversión** o negación.

Las compuertas AND, OR y NOT se conocen como **compuertas básicas**, dado que no se definen a partir de otras compuertas, sino que, a partir de ellas, se definen otros tipos de compuertas y circuitos digitales combinacionales y secuenciales que las incorporan hasta el límite de los microprocesadores. Para comprender el funcionamiento de las compuertas básicas, es interesante utilizar ejemplos provenientes del mundo de la electricidad. Esto implica que las funciones lógicas son de aplicación en otras áreas además de la electricidad y la electrónica: hidráulica, neumática, mecánica, informática, telecomunicaciones, transporte, etcétera. En este capítulo, se analizarán las compuertas digitales con comportamiento ideal, es decir, en las que no se produce **retardo de propagación** entre el cambio en alguna de las entradas y el cambio a la salida.

Compuerta AND (Y)

Se la puede representar en forma simple mediante un circuito eléctrico compuesto de una fuente de alimentación continua (pila, batería), dos interruptores uno a continuación de otro o en **serie**, y un elemento que emita luz (lámpara, led). Luego de analizar el circuito, se deduce que la lámpara **L** se encenderá solamente cuando el interruptor **I1** y el interruptor **I2** estén cerrados, permitiendo el paso de la corriente eléctrica desde la batería hasta la lámpara. Esta forma de representación se conoce como **diagrama de contactos eléctricos** .

MICROPROCESADOR

Se denomina **microprocesador** (o simplemente **procesador**) a un dispositivo digital programable, ampliamente utilizado en computadoras y también en sistemas electrónicos. El microprocesador emplea números binarios para representar cualquier tipo de información.

Desde el punto de vista lógico, es frecuente asignar a la lámpara L el estado 1 (alto) cuando está encendida y 0 (bajo) en caso contrario. En el caso de los interruptores, corresponde 1 para interruptor cerrado y 0 para interruptor abierto. Todas las combinaciones posibles se pueden representar en una tabla de verdad.

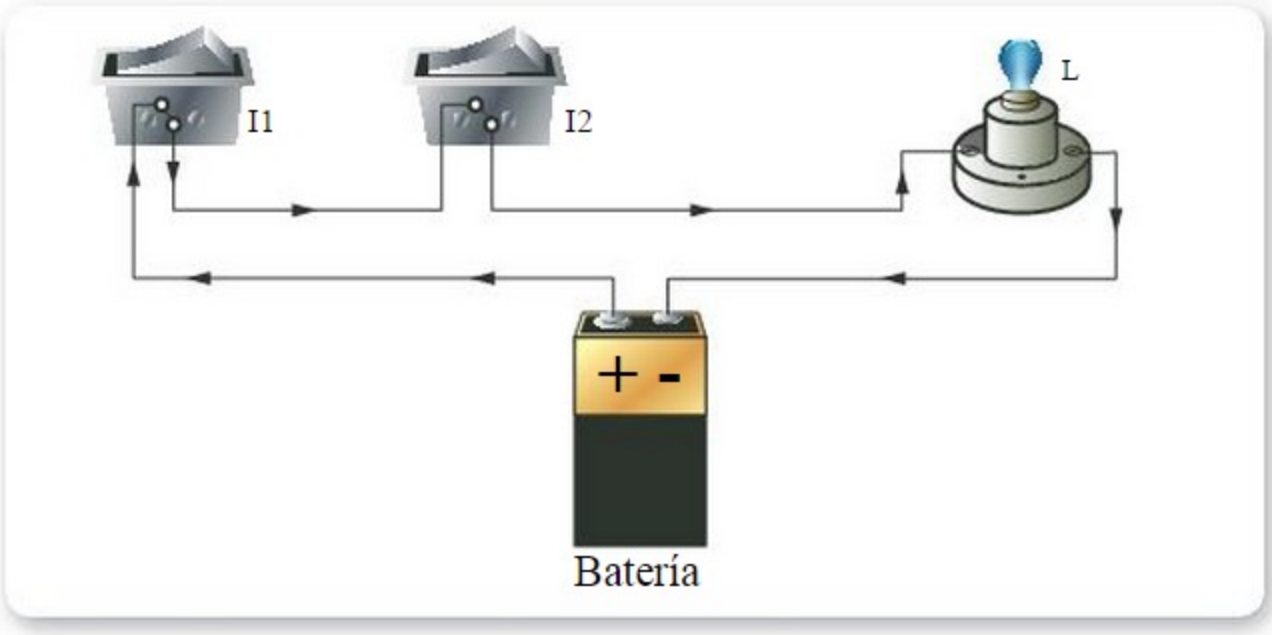


Figura 2. Diagrama de contactos eléctricos que representa la combinación I1 = 1 e I2 = 1 para una compuerta AND. Ambos interruptores cerrados es la única combinación posible que enciende la lámpara L.

En la figura siguiente se observa el símbolo de la compuerta AND y la simulación de su funcionamiento realizada por medio de Logicly.

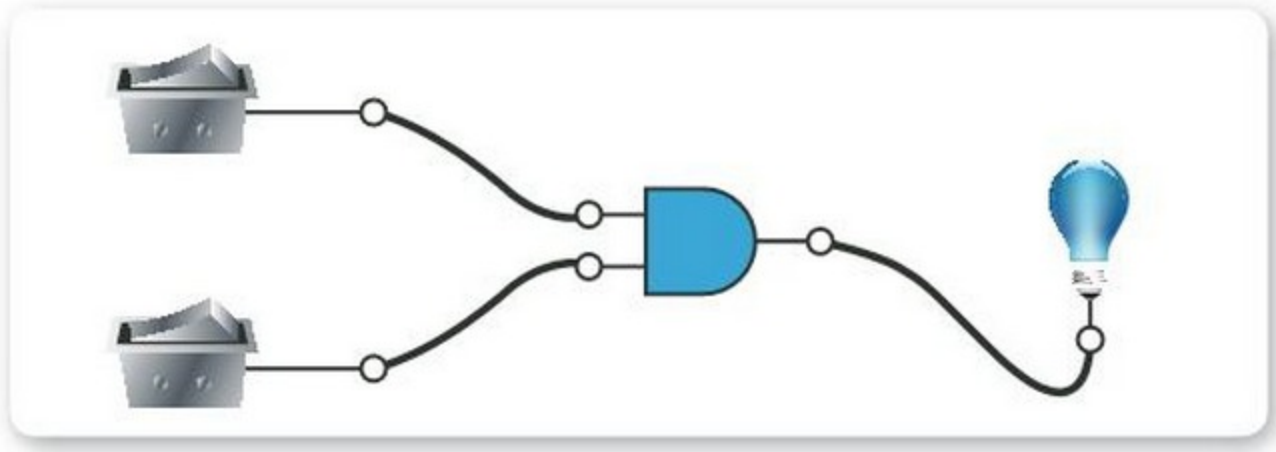


Figura 3. Cambiando la posición de los interruptores se recrean las distintas combinaciones de entrada en una compuerta AND para encender-apagar la lámpara L.

Compuerta OR (O)

Se la puede representar mediante un circuito eléctrico compuesto de una fuente de alimentación continua (pila, batería), dos interruptores en paralelo y un elemento que emita luz (led).

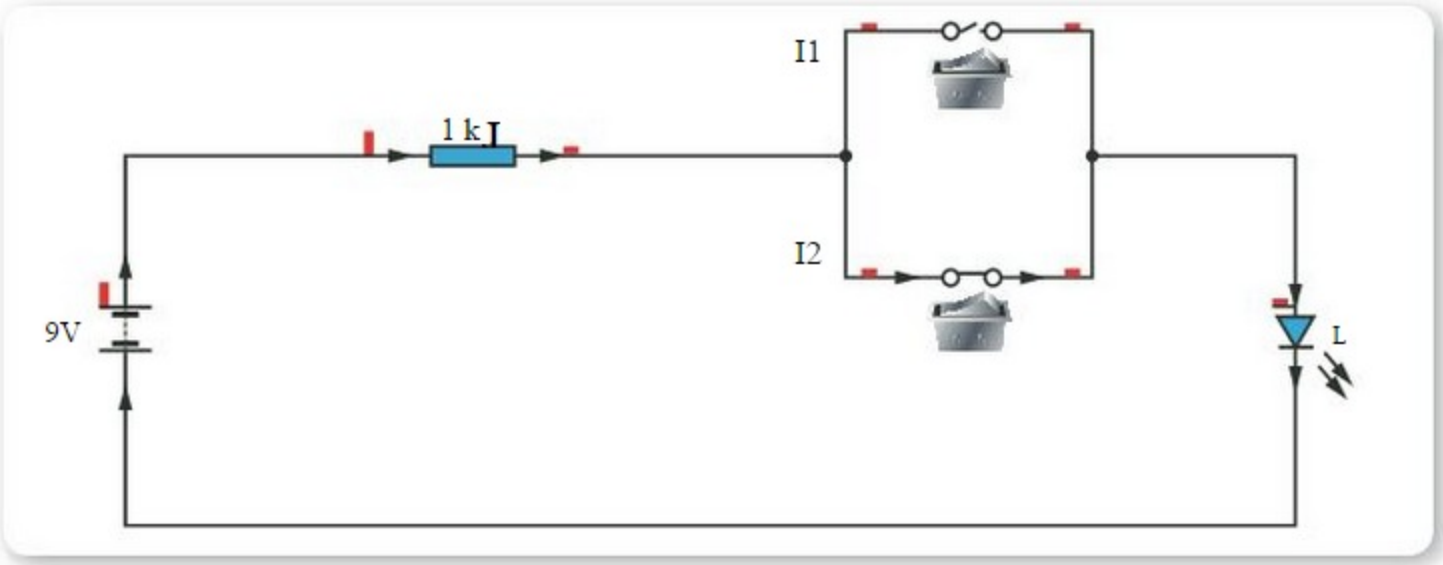


Figura 4. Representación de la combinación de interruptores I1=0 e I2=1 para una compuerta OR, es decir uno cerrado y otro abierto, una combinación posible para encender L.

Luego de analizar el circuito, se deduce que la lámpara L se encenderá cuando el interruptor I1 o el interruptor I2, o ambos, estén cerrados permitiendo el paso de la corriente eléctrica desde la batería hasta la lámpara. En la siguiente figura, se observa el símbolo de la compuerta OR y la simulación de su funcionamiento realizada mediante Logicly.

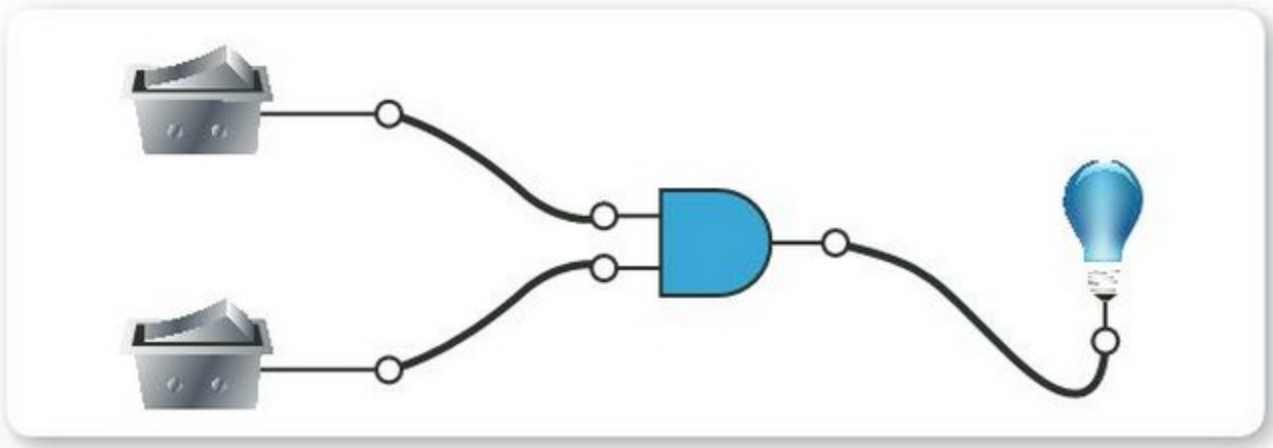


Figura 5. Cambiando la posición de los interruptores, se recrea la tabla de verdad de la compuerta OR para encender-apagar la lámpara L.

COMPUERTAS AND Y OR COMERCIALES

Las compuertas AND tienen versiones comerciales, por ejemplo, con los circuitos integrados **TTL 47LS08** (cuádruple compuerta AND de dos entradas positivas) equivalente al **CMOS 4081** , **TTL 74LS11** (triple compuerta AND de tres entradas positivas) y **74LS21** (cuádruple compuerta AND de cuatro entradas positivas). Las compuertas OR, con los **TTL 47LS32** (cuádruple compuerta OR de dos entradas positivas).

Inversor (NOT)

Consideremos el siguiente circuito compuesto por una batería, un interruptor y un diodo led como indicador luminoso.

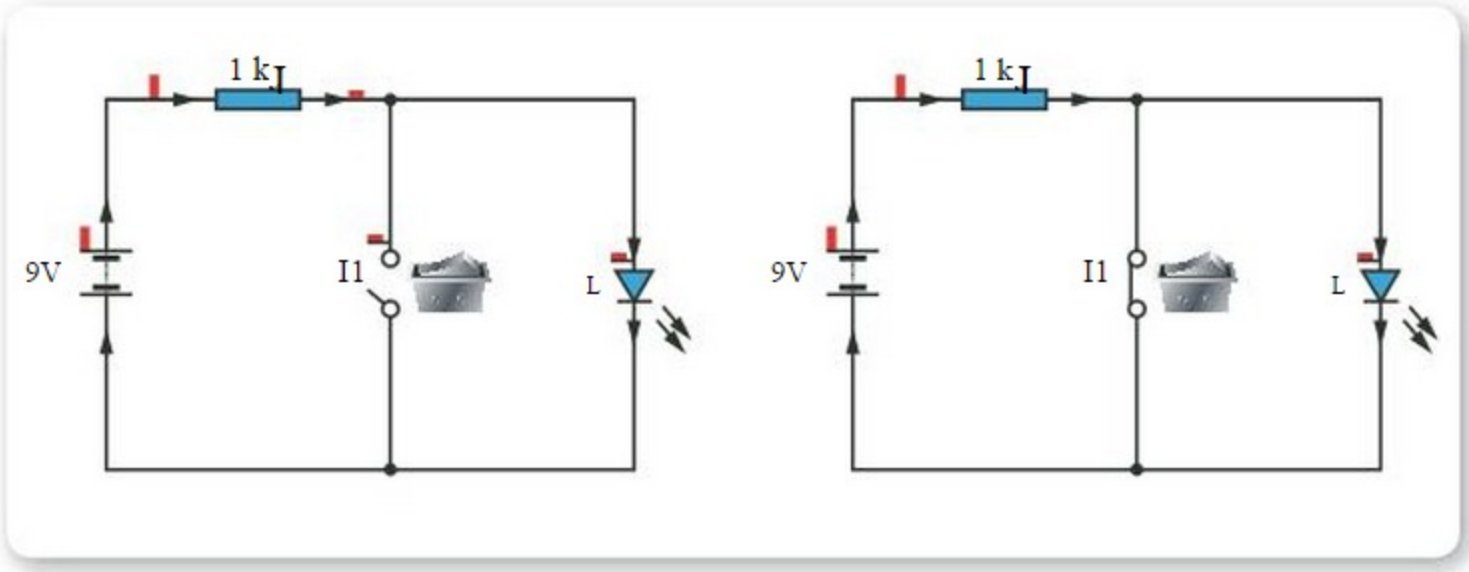


Figura 6. Cuando el led está encendido (1), el interruptor I1 está abierto (0), mientras que, cuando el led está apagado (0), el interruptor I1 está cerrado (1).

Se concluye que el led presenta un estado lógico inverso al del interruptor. En la siguiente figura, se observa la simulación de una compuerta NOT realizada con Logicly.

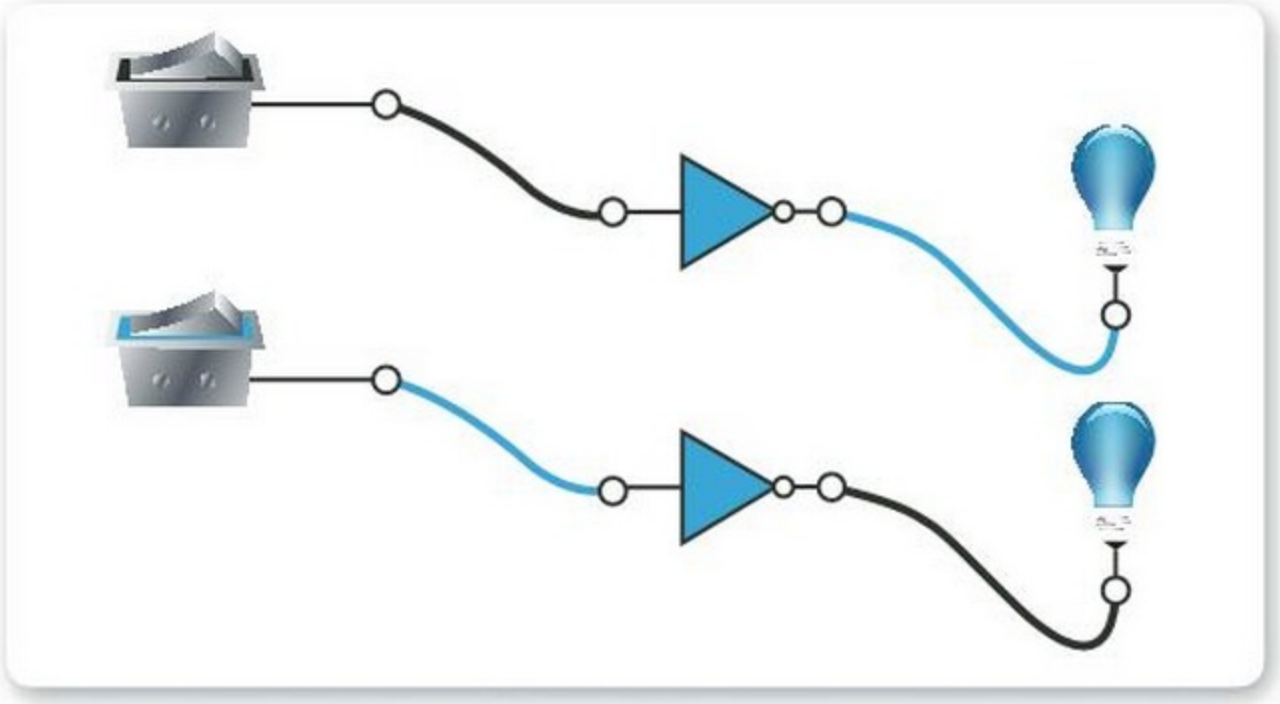


Figura 7. Cambiando la posición del interruptor, se recrea la tabla de verdad de la compuerta NOT para encender-apagar la lámpara L. Observemos que la compuerta NOT posee una sola entrada.

Si se conecta un inversor a la salida de una compuerta AND, se obtiene una compuerta **NAND**. En cambio, si la compuerta es OR, se obtiene una compuerta **NOR**. También se utilizan para producir

señales complementarias o **invertidas** en otros circuitos más complejos, como por ejemplo, circuitos lógicos decodificadores.

Si se conectan dos inversores en serie, se obtiene la misma señal original ya que se realiza una doble inversión sobre esta.

A continuación se presentan distintos tipos de compuertas y su simbología lógica.

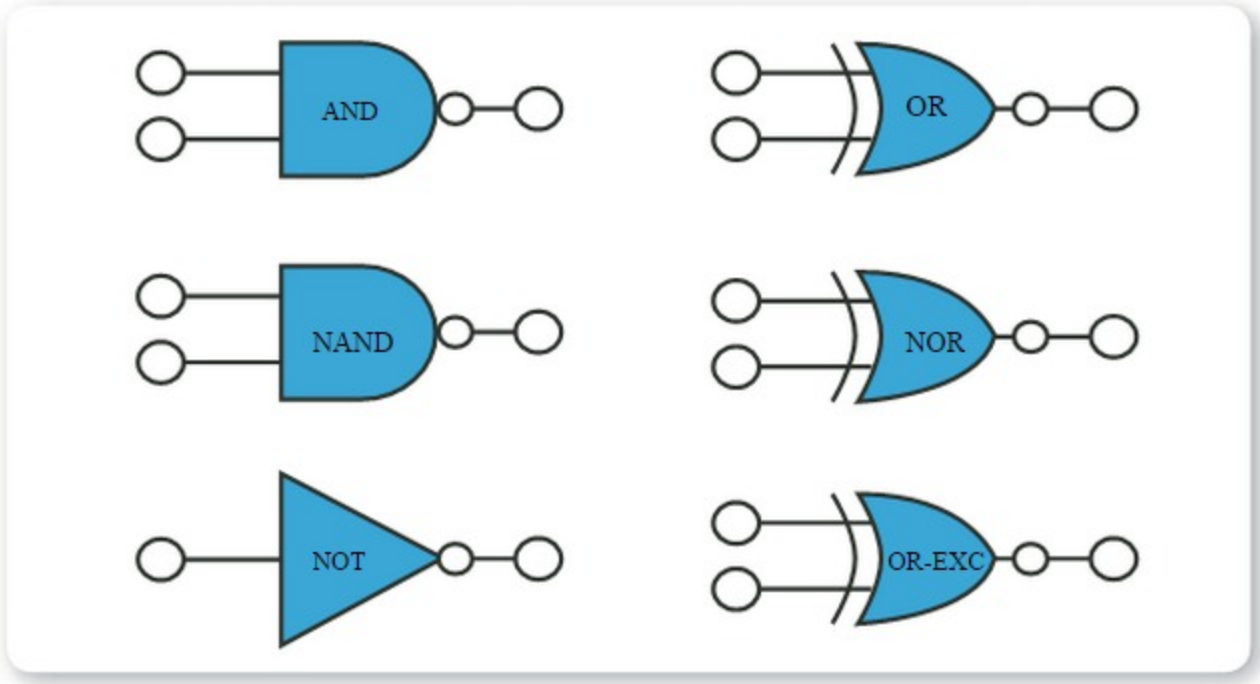


Figura 8. Las compuertas básicas AND, OR y NOT, y otras que se obtienen por combinación de las primeras.

Funciones lógicas y tabla de verdad

Si bien las funciones lógicas se pueden expresar de varias maneras, se utilizarán dos de ellas: las expresiones algebraicas y las tablas de verdad.

Las **expresiones algebraicas** están formadas por variables lógicas, los elementos 0 y 1, los operadores producto, suma y negación, y los símbolos , (coma) e = (igual). Mediante estos elementos se puede expresar cualquier función lógica. Por ejemplo, la función f (a, b) = a · b indica que f = 1 (valor cierto) solo cuando, simultáneamente, a y b son iguales a 1.

INVERSORES MEDIANTE COMPUERTAS

Si bien sabemos que existe una gran variedad de circuitos integrados digitales comerciales, tales como el TTL 74LS04, también sabemos que es posible optar por construir un inversor. Para hacerlo, simplemente debemos conectar las entradas entre sí en una compuerta NAND o en una compuerta NOR.

Una **tabla de verdad** expresa una función lógica especificando el valor que tiene la función para cada combinación de valores de las variables de entrada. El uso de esta herramienta se limita al álgebra de Boole, ya que las variables solo pueden adoptar dos valores, 0 o 1. Una tabla de verdad se organiza en una línea vertical que separa todas las posibles combinaciones de las variables de entrada, a la izquierda, del valor de la función para cada una de las combinaciones, a la derecha. Es frecuente indicar las entradas con las primeras letras del abecedario y la salida con las últimas, por ejemplo **a**, **b**, **c** y S, respectivamente. En la tabla de verdad siguiente se presenta la función AND.

TABLA DE VERDAD DE LA FUNCIÓN LÓGICA AND

▼ a	▼ B	▼ S	▼ DESCRIPCIÓN
0	0	0	a y b están abiertos, lámpara apagada.
0	1	0	a está abierto y b cerrado, lámpara apagada.
1	0	0	a está cerrado y b abierto, lámpara apagada.
1	1	1	a y b están cerrados, lámpara encendida.
Expresión booleana: a AND b			a · b

Tabla 1. Tabla de verdad de una función AND de dos entradas, a y b, y una salida, S.

Dadas **n** variables de entrada que solamente pueden tomar dos valores (0 y 1), la cantidad de combinaciones posibles diferentes será de 2^n, es decir, la tabla de verdad tendrá n columnas a la izquierda (una para cada variable de entrada) con 2^n filas (una para cada combinación posible), mientras que a la derecha habrá una columna con los valores que tome la función de acuerdo a las distintas combinaciones de entrada.

EXPRESIONES EQUIVALENTES

Una función lógica puede expresarse por una infinidad de expresiones algebraicas equivalentes, y los axiomas y las leyes del álgebra de Boole determinan si estas expresiones lo son. Así se puede construir una tabla de verdad a partir de cada una de ellas, de modo que, si las tablas resultantes son iguales, las dos expresiones analizadas corresponden a una misma función.

TABLA DE VERDAD DE LA FUNCIÓN LÓGICA OR			
▼ a	▼ B	▼ S	▼ DESCRIPCIÓN
0	0	0	a y b están abiertos, lámpara apagada.
0	1	1	a está abierto y b cerrado, lámpara encendida.
1	0	1	a está cerrado y b abierto, lámpara encendida.
1	1	1	a y b están cerrados, lámpara encendida.
Expresión booleana: a OR b			a + b

Tabla 2. Tabla de verdad de una función OR de dos entradas, a y b, y una salida, S.

Una forma de escribir las filas es comenzando por la combinación formada solo por ceros, luego la correspondiente al 1, y así de manera sucesiva en orden creciente hasta alcanzar la combinación formada solo por unos, 2^n-1, en el esquema conocido como **orden lexicográfico** .

En la siguiente tabla de verdad se presenta la función NOT.

TABLA DE VERDAD DE LA FUNCIÓN LÓGICA NOT		
▼ A	▼ S	▼ DESCRIPCIÓN
0	1	a está abierto, lámpara encendida.
0	0	a está cerrado, lámpara apagada.
Expresión booleana		NOT a o a'

Tabla 3. Tabla de verdad de una función NOT (una entrada, a, y una salida, S).

En las siguientes tablas se representan las tablas de verdad para funciones de una, dos y tres variables de entrada. Notaremos que los valores de la salida pueden ser cualesquiera y dependerán de las combinaciones de entrada indicadas como válidas en el problema por resolver.

MAPAS DE KARNAUGH PARA DISTINTAS VARIABLES					
▼ UNA VARIABLE	▼ DOS VARIABLES	▼ TRES VARIABLES			
a f	A B f	a	b	c	f
0	0 0	0	0	0	

UNA VARIABLE	DOS VARIABLES	TRES VARIABLES
1	0 1	0 0 1
	1 0	0 1 0
	1 1	0 1 1
		1 0 0
		1 0 1
		1 1 0
		1 1 1

Tabla 4. Tabla de verdad de la función lógica NOT.

Para fijar el orden de las columnas y las filas en una tabla de verdad, única para una función lógica dada, se coloca en la columna más a la izquierda la variable de mayor peso y, en la columna de la derecha, la de menos peso, por ejemplo: a, b, c respectivamente para tres variables de entrada en tres columnas. Se puede representar el comportamiento de varias funciones en una misma tabla de verdad en la que se comparten las mismas variables de entrada y, a la derecha de la línea vertical, se tendrán tantas columnas como funciones por representar. Estas funciones se pueden ubicar en cualquier orden.

EJEMPLO DE TABLA DE VERDAD CON TRES SALIDAS

a	B	C	F_1	F_2	F_3
0	0	0	1	0	1
0	0	1	0	0	0
0	1	0	0	1	0
0	1	1	0	0	1
1	0	0	1	0	1
1	0	1	0	0	0
1	1	0	0	1	0
1	1	1	0	0	1

Tabla 5. Esquema con tres entradas y tres salidas.

En la siguiente tabla se comparan las funciones lógicas de las compuertas lógicas de dos entradas.

TABLA COMPARATIVA ENTRE COMPUERTAS

ENTRADAS		TABLA DE VERDAD PARA LAS SALIDAS					
a	b	AND	NAND	OR	NOR	OR-EX	NOR-EX
0	0	0	1	0	1	0	1
0	1	0	1	1	0	1	0
1	0	0	1	1	0	1	0
1	1	1	0	1	0	0	1

Tabla 6. Salidas en cada compuerta en función de distintas combinaciones de entrada.

La próxima tabla muestra un resumen de las distintas funciones lógicas y la notación booleana equivalente.

TABLA RESUMEN

FUNCIÓN LÓGICA	NOTACIÓN BOOLEANA
AND	a · b
OR	a + b
NOT	a’
NAND	(a · b)’
NOR	(a + b)’
OR – EX	(a’ · b) + (a · b’) o a b ⊕
NOR – EX	(a’ · b’) + (a · b) o (a b)’ ⊕

Tabla 7. Resumen de las funciones lógicas y la notación booleana equivalente, que incluye las funciones básicas y las funciones exclusivas.

Se debe tener en cuenta que no solo existen compuertas con dos entradas puesto que, comercialmente, están disponibles compuertas lógicas con tres, cuatro y ocho entradas individuales. Así, en una compuerta AND de cuatro entradas, se necesita que todas las entradas estén presentes para producir la salida requerida y, por lo tanto, la tabla de verdad será más extensa que las analizadas hasta este punto.

Es importante comprender el paso de una expresión algebraica desde una función hasta su tabla de verdad, y viceversa. En el primer

caso, se presentan dos alternativas de solución: por un lado, evaluar el resultado de la función para cada una de las combinaciones de entrada escribiendo en el lugar correspondiente de la tabla el valor obtenido (0 o 1) o analizar la expresión algebraica deduciendo para qué valores de las variables de la función vale 0 o 1; por otro lado, completando la tabla de verdad.

En la práctica, para **sintetizar** circuitos lógicos a partir de la tabla de verdad, se puede representar la expresión lógica sobre la base de los términos de la función iguales a 1 (expresión en forma de suma de productos, minitérminos o minterms) o iguales a 0 (expresión en forma de productos de sumas, maxitérminos o maxterms).

Un criterio de decisión para utilizar una forma u otra es considerar la cantidad de 0 y 1 presentes en la función, y seleccionar aquella forma que necesite menor cantidad de términos para ser expresada.

Para obtener la función lógica a partir de la tabla de verdad, se parte de la lectura de cada combinación de entrada especificándola, por ejemplo, como suma de productos (minitérminos). Se denomina **término mínimo** o **minitérmino** al término producto (único) que expresa una función, ya que solo vale 1, para una sola combinación de las variables de entrada. En un minitérmino figuran siempre todas las variables de la función, ya sea negadas o sin negar.

SUMA DE MINITÉRMINOS

	▼ ENTRADAS			▼ SALIDA
b10	A	b	c	f
0	0	0	0	1
1	0	0	1	1
2	0	1	0	0
3	0	1	1	1
4	1	0	0	0
5	1	0	1	0
6	1	1	0	0
7	1	1	1	1

Tabla 8. Para construir la función expresada como suma de productos (minitérminos), se analizan las combinaciones de entrada que hacen a la función tomar el valor 1.

A partir de la tabla de verdad de una función, podemos obtener una expresión en suma de productos haciendo la suma lógica de los minitérminos que la forman. Por este motivo, la expresión obtenida de esta forma se llama **suma de minitérminos** .

$$f = (a' \cdot b' \cdot c') + (a' \cdot b' \cdot c) + (a' \cdot b \cdot c) + (a \cdot b \cdot c)$$

o: $f = \sum_3 (0,1,3,7)$

La **forma numérica** es la expresión de la función **f** de manera más simple, en la que se observa la sumatoria de tres entradas y, entre paréntesis, los números decimales para los cuales la función vale 1. Así, la función de tres entradas se obtiene sumando los productos que corresponden a las combinaciones en binario de los números decimales 0, 1, 3 y 7.

Una función lógica, también, se puede expresar como **producto de sumas**. Tanto la expresión como suma de minitérminos o como el producto de sumas se denomina **forma canónica** de la función: estas se caracterizan porque aparecen todas las variables ya sea en su forma directa (a) o en su forma complementada (a'). Para expresar la función del ejemplo anterior en forma de productos de sumas, se analizan las distintas combinaciones de entrada que hacen que la función sea igual a cero:

$$f = (a' + b + c') \cdot (a + b' + c') \cdot (a + b' + c) \cdot (a + b + c')$$

Cualquiera de las expresiones booleanas anteriores podrá simplificarse mediante alguno de los métodos que se analizarán a continuación.

SUMAS Y PRODUCTOS CANÓNICOS

Si una función canónica tiene **n** variables, cada uno de sus productos o sumas canónicas tendrán **n** variables. Como cada variable se puede representar en su forma directa o complementada (dos combinaciones por cada variable), el número de productos canónicos posibles será **2n**, al igual que el de sumas canónicas.

Minimización de funciones lógicas por el método de Karnaugh

Cualquier función lógica se puede implementar usando las compuertas básicas, y así es posible construir un circuito que se comporte de acuerdo a esa función.

Podemos implementar o sintetizar una función a partir de su expresión algebraica, sustituyendo cada operador de la función por la compuerta lógica adecuada. Por ejemplo, una compuerta AND de tres entradas permitirá implementar un término producto de tres variables. Las compuertas deben estar interconectadas entre sí y con las entradas.

En el momento de implementar un circuito lógico mediante compuertas lógicas comerciales, se debe conocer que no responden de manera instantánea a las variaciones en las señales de entrada, dado que experimentan un cierto retardo, distinto de cero, por el hecho de que los dispositivos electrónicos internos de una compuerta necesitan un tiempo determinado para modificar su estado. Cada compuerta tiene un retardo de tiempo en la modificación de la salida una vez que alguna entrada se ha modificado, y esto depende de la tecnología que se haya utilizado para construirla. Aunque los retardos son muy pequeños, en el orden de los nanosegundos, es necesario tenerlos en cuenta cuando construimos físicamente un circuito. Otra circunstancia por considerar es la transición entre diferentes niveles de tensión de una señal (0 a 1, y viceversa), ya que tampoco es instantánea. Una premisa fundamental en el momento del diseño es reducir este tiempo de respuesta que depende, entre otros aspectos, del número de entradas de una compuerta, y, como cada compuerta tiene un cierto retardo, un circuito será, en general, más rápido cuantos menos niveles de compuertas posea entre las entradas y las salidas. Entonces, el número de **niveles de compuertas** de un circuito es el número

ANÁLISIS DE UN CIRCUITO

En el proceso inverso a la síntesis, **análisis** de un circuito, es necesario escribir las expresiones que corresponden a la salida de cada compuerta. Esto se hace comenzando desde las entradas del circuito hasta obtener la expresión correspondiente en su línea de salida.

máximo de compuertas que una señal debe atravesar en forma consecutiva para generar la señal de salida.

La **síntesis a dos niveles** , una técnica para diseñar circuitos lógicos, es un proceso por el que se busca que un circuito no tenga más de dos niveles de compuertas (sin considerar las compuertas NOT) partiendo, por ejemplo, de la expresión como suma de minitérminos. Por otra parte, interesa que un circuito sea, además de rápido, pequeño y barato Tengamos en cuenta que cuantas menos compuertas tenga, más pequeño y más barato será.

CUALQUIER FUNCIÓN LÓGICA SE IMPLEMENTA USANDO COMPUERTAS BÁSICAS

Una herramienta para reducir dos o más minitérminos a un único término producto en el que aparecen menos variables es la **simplificación de funciones lógicas** de modo de implementar la función mediante un circuito de menores dimensiones, costo, consumo de potencia y retardos. Un objetivo adicional es realizar un circuito lógico empleando solo determinados tipos de compuertas lógicas. Existen diferentes métodos para simplificar la expresión de una función en suma de minitérminos. La **simplificación algebraica** se emplea en los casos en que dos o más minitérminos se pueden reducir a un único término producto en el que aparecerán menos variables. El **método gráfico** mediante **mapas de Karnaugh** proporciona una técnica sencilla para detectar visualmente los casos en los que se puede minimizar una expresión en suma de minitérminos ya que, entre todos los circuitos a dos niveles que implementan una función lógica, permite obtener fácilmente el menor de todos. En esta obra hacemos referencia solo a la simplificación algebraica y al **método de Karnaugh** .

OTROS MÉTODOS DE SÍNTESIS

Existen otros métodos de síntesis que generan circuitos con más de dos niveles más rápidos que los que se obtienen a partir de las expresiones de las funciones en suma de minitérminos, aunque no son objeto de esta obra. Ejemplos de estos son: el método de **Quine-McCluskey** , el método **tabular** y el método de **selección para tablas cíclicas.**

Simplificación algebraica

Dada una expresión algebraica, este método consiste en buscar dos términos canónicos adyacentes de **n** variables, que solo se diferencien en el estado de una de ellas (que aparece negada en uno de los términos y en el otro sin negar) y, que por lo tanto, se pueden simplificar. Supongamos la expresión booleana como suma de minitérminos:

$$f = (a' \cdot b' \cdot c') + (a' \cdot b' \cdot c) + (a' \cdot b \cdot c) + (a \cdot b \cdot c)$$

Mediante factor común entre el primero y segundo término $a' \cdot b' \cdot (c' + c)$. La expresión entre paréntesis vale 1 por complementación. Mediante factor común entre el tercero y cuarto término $b \cdot c \cdot (a' + a)$. La expresión entre paréntesis vale 1 por complementación. Entonces, la expresión simplificada resultante es:

$$f = (a' \cdot b') + (b \cdot c)$$

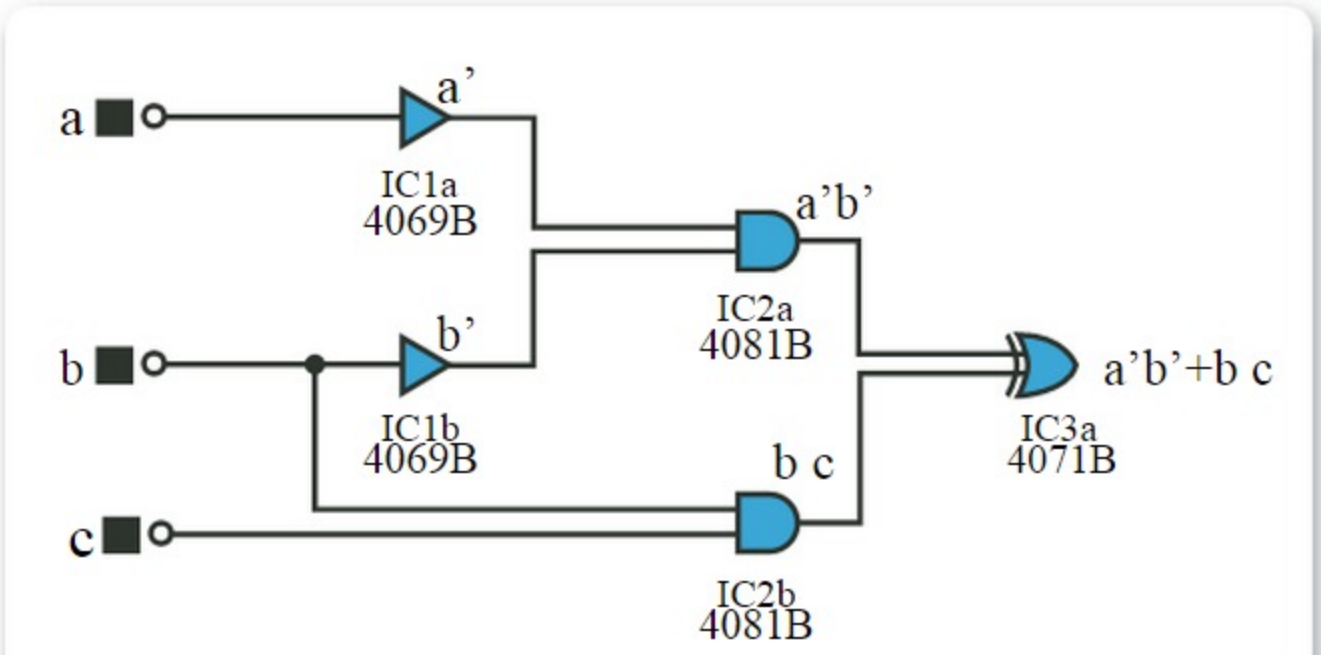


Figura 9. Circuito lógico a dos niveles que se obtiene a partir de la función simplificada.

Método de Karnaugh

Es otra forma de representar la tabla de verdad, con la ventaja de que ofrece una disposición espacial adecuada para la simplificación (o minimización: obtener la función mínima). El método se implementa a

partir del mapa de Karnaugh, un arreglo de filas y columnas (matriz) que forman casilleros, y cada uno de ellos representa un término canónico. Por una cuestión de adyacencia se emplea **código Gray** (una variante del código binario), de modo que los casilleros contiguos entre sí, horizontal y verticalmente (no en diagonal), representan términos canónicos adyacentes (que solo difieren en un bit), y es posible simplificar una variable.

> IMPLEMENTAMOS O SINTETIZAMOS UNA FUNCIÓN A PARTIR DE SU EXPRESIÓN ALGEBRAICA

En cada fila y columna de la matriz de casilleros se representan, con ceros y unos, los valores que toman las variables. En cada cuadrado se representa el valor 0 o 1 que toma la función para cada término canónico. La cantidad de casilleros del mapa es 2^n donde n es el número de variables de entrada. Es importante recordar que, en la función booleana, tienen que estar presentes todas las variables, ya sea en su forma normal o en su forma negada.

Para aplicar este método es necesario desarrollar cuatro pasos:

- Trasladar la tabla de verdad de la función lógica a un esquema conocido como mapa de Karnaugh.
- Determinar visualmente aquellos casos en que se puede sacar factor común.
- Deducir los términos producto más simples posibles.
- Obtener la expresión mínima de la función realizando la suma lógica de los términos producto.

A continuación, observaremos las estructuras de estos mapas para dos, tres y cuatro variables de entrada. Se dice que dos casilleros del mapa son adyacentes si corresponden a combinaciones en las que solo cambia el valor de una variable.

CÓDIGO GRAY

El **código Gray** es otro tipo de código basado en el sistema binario, aunque de construcción muy diferente a la de los demás códigos. Es un código no ponderado, y su principal característica es que dos números sucesivos cualesquiera solo varían en 1 bit. Por esta razón también se lo denomina **código Gray progresivo**.

MAPA DE KARNAUGH CON 4 CASILLEROS		
b \ a	0	1
0		
1		

Tabla 9. Esquema para dos variables de entrada: a y b.

MAPA DE KARNAUGH CON 8 CASILLEROS				
c \ a b	00	01	11	10
0				
1				

Tabla 10. Esquema para tres variables de entrada: a, b y c.

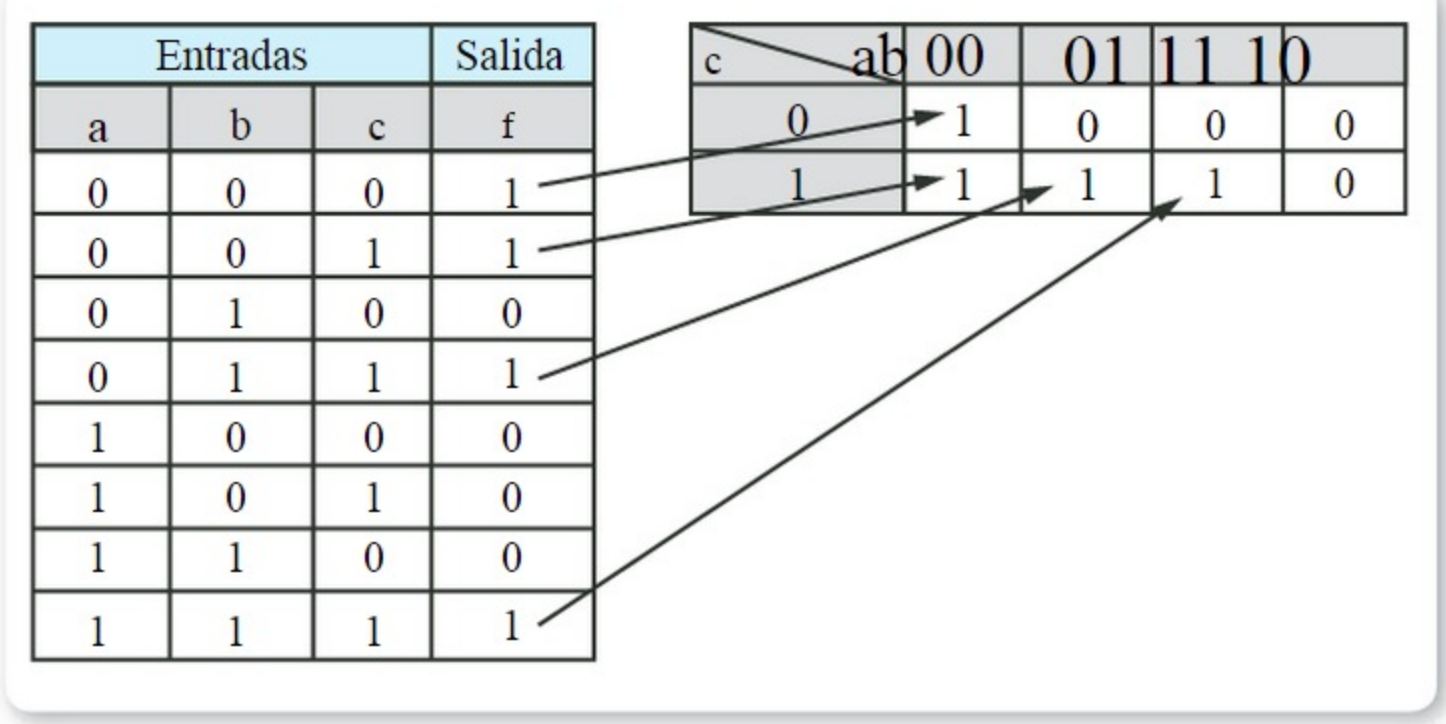
MAPA DE KARNAUGH CON 16 CASILLEROS				
c d \ a b	00	01	11	10
00				
01				
11				
10				

Tabla 11. Esquema para cuatro variables de entrada: a, b, c y d.

De acuerdo con los esquemas anteriores, se verifica que los casilleros adyacentes presentan tres características:

- Dos casilleros son **adyacentes** si únicamente cambia el valor de una variable.
- En los mapas de tres y cuatro variables, los casilleros de la columna de más a la derecha también son adyacentes con los de la columna más a la izquierda.
- En los mapas de cuatro variables, los casilleros de la fila superior también son adyacentes con los de la fila inferior.

Una vez determinado el mapa por utilizar de acuerdo a la cantidad de variables de entrada, se coloca dentro de cada casilla el valor de la función (0 o 1) para la combinación correspondiente de variables, a partir de la tabla de verdad.



Entradas			Salida
a	b	c	f
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

c \ ab	00	01	11	10
0	1	0	0	0
1	1	1	1	0

Figura 10. Un ejemplo que retoma la función booleana analizada antes y utiliza un mapa para tres variables.

Entonces, el valor 1 de la primera casilla del mapa se corresponde con la fila de la tabla de verdad donde a = b = c = 0, y así sucesivamente. De hecho, basta con rellenar los casilleros para los que la función vale 1.

El segundo paso en el método de Karnaugh consiste en agrupar mediante rectángulos los unos que están ubicados en casillas adyacentes formando grupos de 1, 2, 4, 8 o 16 unos y, luego, verificar que los lados de estos rectángulos sean un número de casilleros potencia de 2 y que en su interior solo haya unos.

ALGORITMO DE QUINE-MCCLUSKEY

Es otro método de simplificación de funciones booleanas funcionalmente idéntico a la utilización del **mapa de Karnaugh**, aunque su forma tabular lo hace más eficiente que dicho mapa cuando se trata de trabajar con más de cuatro variables. El tiempo de resolución del algoritmo crece de forma exponencial con el aumento del número de variables.

Si bien la forma para agrupar los unos no es única, es conveniente respetar las siguientes reglas:

- Todos los unos deben formar parte de algún grupo de unos.
- Estos grupos de unos deben ser lo más grande posible de modo que, en cada término, aparezca el menor número de variables. Así, es conveniente comenzar agrupando los unos en los grupos más grandes.
- Menor cantidad de unos es mejor para obtener el número menor de términos productos de sumas.
- El mismo uno puede formar parte de más de un grupo, si eso colabora con las dos reglas anteriores.
- Si luego de integrar los grupos más grandes de unos queda alguno libre, es conveniente verificar si este se puede agrupar como parte de algún grupo definido con anterioridad.

c \ ab	00	01	11	10
0	1	0	0	0
1	1	1	1	0

Figura 11. Aplicación a la función lógica definida anteriormente.

El tercer paso del método consiste en determinar cuáles son los términos producto de cada grupo. Para ello, debemos tener en cuenta las siguientes consideraciones:

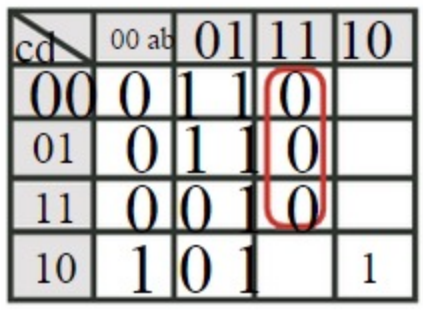
- Solo aparecen las variables cuyo valor es constante para todos los casilleros que forman un grupo.
- Si en todos los casilleros de un grupo una variable vale 1, esa variable aparece en el término producto sin negar.
- Si en todos los casilleros de un grupo una variable vale 0, esta aparece negada en el término producto.

Entonces, de acuerdo a ello, para el rectángulo a = 0, b = 0 y c = 0, se obtienen solamente las variables a = 0 y b = 0, ya que la variable c en un casillero es 0 y en el otro es 1, por lo que se simplifica (recordar el concepto de complementación). Lo mismo ocurre con la variable a en el casillero restante, del que se obtiene b = 1 y c = 1.

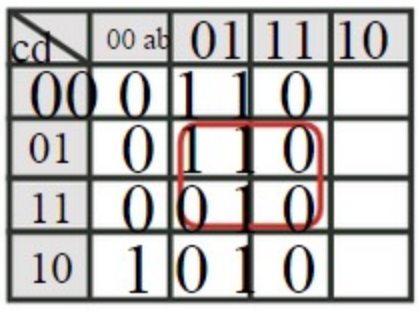
Como una función booleana se puede expresar mediante la suma lógica de todos sus minitérminos, los dos términos producto obtenidos antes constituyen los minitérminos de la función. Así se expresa la

función resultante mediante la suma lógica de estos términos producto. Tanto la función resultante como el circuito lógico son idénticos a los que se obtuvieron al simplificar algebraicamente la función original.

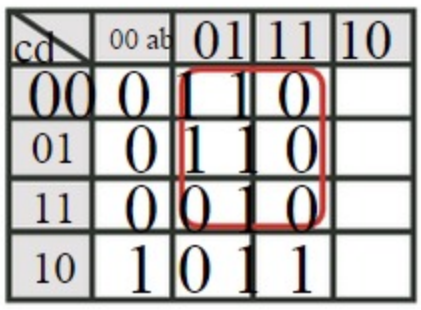
Por último, y a modo de referencia, se presenta una breve guía de agrupamiento de casilleros de manera correcta e incorrecta. Recordemos que la agrupación de 1 en diagonal no es válida y que un mismo 1 puede formar parte de más de una agrupación o bien conformar un término único. Si más de una forma de agrupar es correcta resulta conveniente elegir aquella que permita obtener la menor cantidad de términos producto. Al circuito correspondiente, se lo denomina **circuito mínimo de dos niveles**.



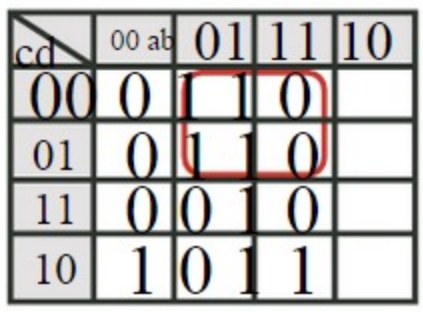
No es correcto ya que el rectángulo considera 3 x 1 casilleros



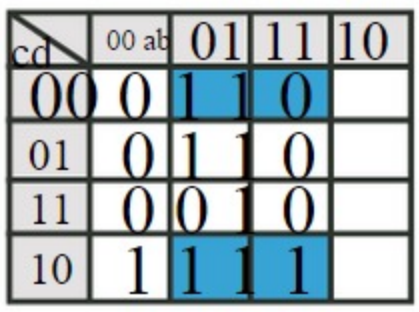
No es correcto ya que el rectángulo contiene "0"



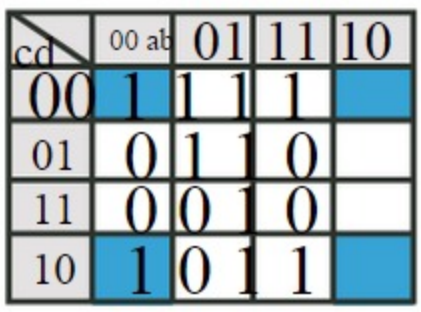
No es correcto ya que el rectángulo es de 3 x 2 y además contiene "0"



Correcto



Correcto: las filas coloreadas superior e inferior son adyacentes



Correcto: las columas coloreadas de los extremos son adyacentes

Figura 12. Agrupamientos correctos de valores lógicos 1 en el mapa de Karnaugh.

VALOR DE FUNCIÓN INDIFERENTE

Para el caso en que alguna de las combinaciones de entrada produzca un valor de función indiferente, indicado con x, es decir, que puede valer 0 o 1, esta característica se hace constar en el casillero correspondiente del mapa de Karnaugh. En el momento de agrupar unos para simplificar y obtener la mínima función en minitérminos, se seleccionará el valor 1 cuando convenga más.

Simulación de compuertas lógicas

Para simular el funcionamiento de circuitos que incorporen compuertas lógicas, es posible utilizar varias aplicaciones, algunas de ellas libres, y otras con versión de evaluación y funcionalidad reducida. Recordemos que **Logic Friday** es un software libre para optimizar, analizar y sintetizar funciones lógicas. También, **Logicly** enseña lógica de compuertas y circuitos digitales de una manera efectiva mediante una versión de evaluación que, luego de finalizada, requiere la licencia de uso. Es una herramienta muy interesante en la enseñanza de construcción de circuitos, para construir experiencia a través de la aplicación práctica del diseño, generación de prototipos, simulación y pruebas de circuitos eléctricos. **Proteus** es una compilación de programas de diseño y simulación electrónica desarrollados por Labcenter Electronics. Consta de dos programas principales: Ares e Isis, y los módulos VSM de simulación y Electra.

Karnaugh Map Minimizer , disponible en **http://k-map.sourceforge.net**, es un software GPL para minimizar funciones booleanas utilizando el método gráfico de los mapas de Karnaugh. Soporta mapas de hasta ocho variables y muestra tanto la tabla de verdad como el mapa de Karnaugh correspondiente. Otro programa freeware es **KarnaughMap 1.2** , disponible en **http://puz.com/sw/karnaugh/karnaugh_12.htm**. Es una aplicación freeware muy fácil de usar, que interactúa con el álgebra de Boole. Esta versión soporta tanto el formato suma de productos como producto de suma.

Combinación de circuitos lógicos

Los grandes temas que involucran a los sistemas digitales son dos: en relación con los fundamentos, y en relación con el análisis y el diseño de estos sistemas. En el **capítulo 1** presentamos la codificación, el álgebra de Boole y la representación de los sistemas digitales. En cuanto al análisis y diseño de estos sistemas, se deben considerar los circuitos secuenciales y los circuitos combinacionales.

- Los **circuitos combinacionales** se caracterizan porque el valor de las señales de salida en un momento determinado depende del valor de las señales de entrada, 0 o 1, en ese mismo instante. Entonces, la salida solo es determinada mediante una función lógica.
- Los **circuitos secuenciales** , son aquellos en los que el valor de las señales de salida en un momento determinado depende de los valores de las señales de entrada desde la puesta en funcionamiento del circuito. Es decir, tienen capacidad de memoria.

> LOS CIRCUITOS SECUENCIALES SE CARACTERIZAN POR TENER CAPACIDAD DE MEMORIA

Los sistemas combinacionales se conforman a partir de las compuertas lógicas **NAND**, **NOR** o **NOT**, que se combinan para producir circuitos lógicos de mayor complejidad. Estas compuertas son los **ladrillos** constructores de los circuitos lógicos combinacionales. Un ejemplo de un circuito combinacional es el decodificador, que permite convertir datos en código binario presentes en su entrada en una determinada cantidad de salidas diferentes que proporcionan el código equivalente decimal. Las tres formas principales para especificar la función de un sistema lógico combinacional son: el álgebra de Boole, la tabla de verdad y el diagrama o circuito lógico.

Desde el punto de vista gráfico, se puede representar un sistema combinacional de la siguiente manera:

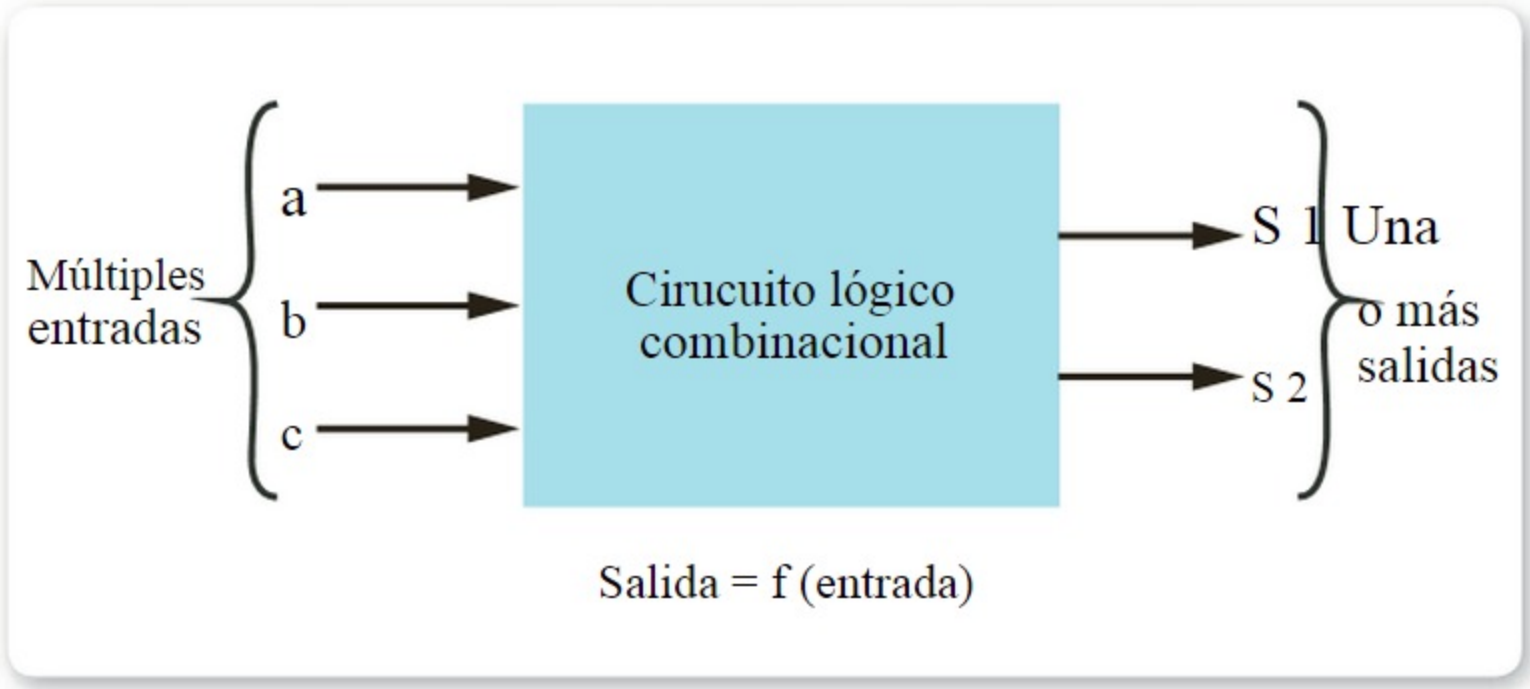


Figura 13. Representación gráfica de un sistema combinacional.

A continuación, se presenta una clasificación simple de la lógica combinacional.

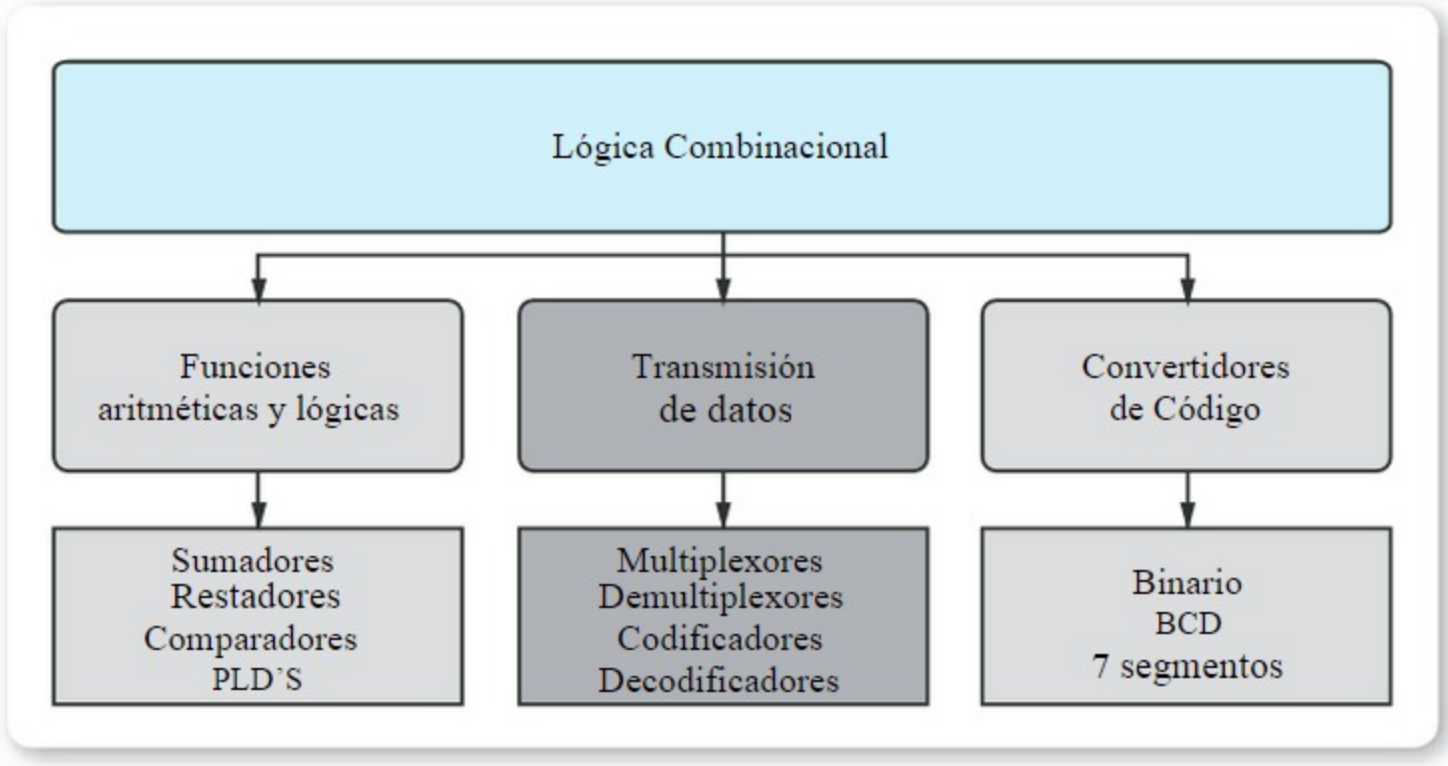
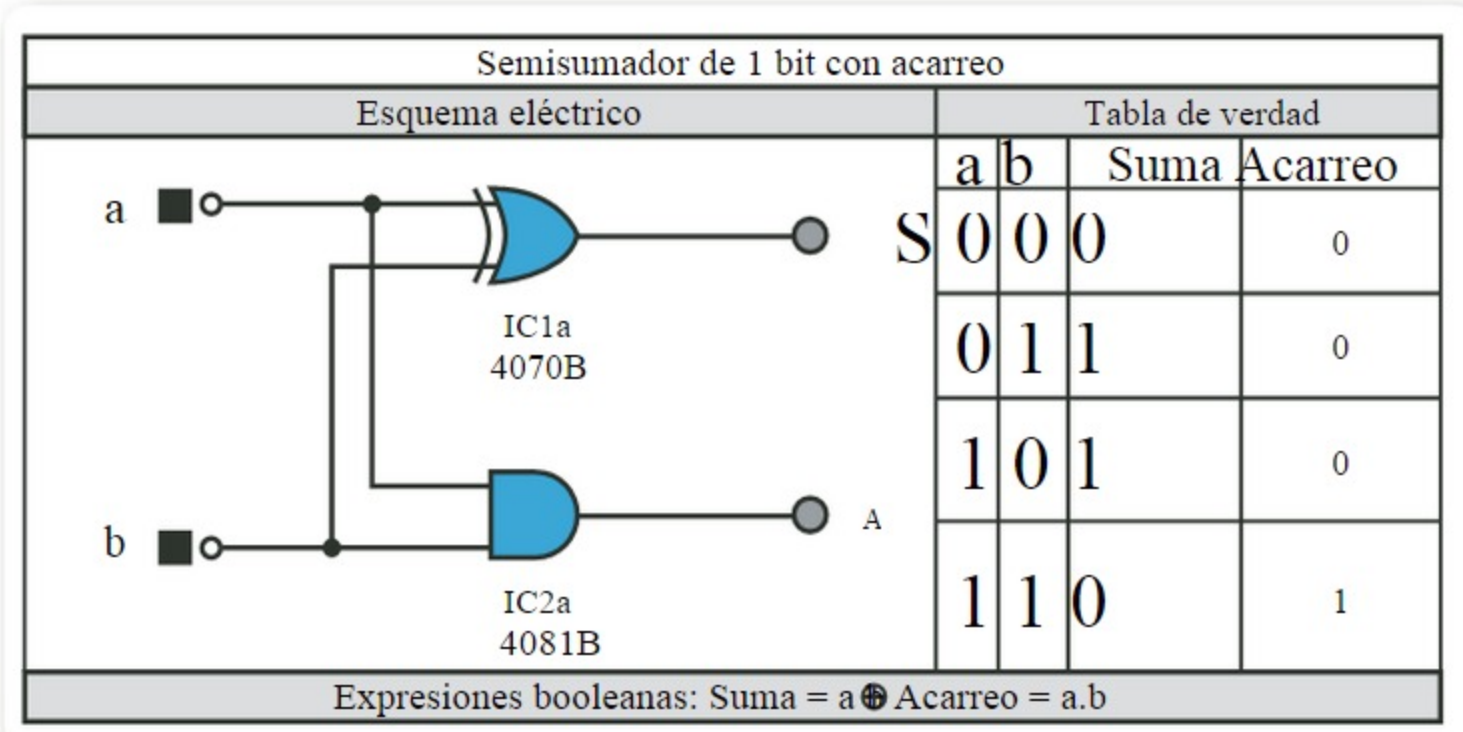


Figura 14. Distintos circuitos lógicos categorizados como combinacionales.

Circuitos sumadores

Uno de los circuitos lógicos combinacionales más utilizados es el **sumador binario**. Este se puede construir con algunas compuertas lógicas y es capaz de sumar números binarios puros o en complemento a 2. Algunos sumadores binarios son: semisumador binario, sumador total y sumador de 4 bits.



a	b	Suma	Acarreo
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Figura 15. Semisumador de 1 bit con acarreo.

Las combinaciones de a y b reflejan las reglas de la suma. Al sumar 1 + 1 se obtiene $\mathbf{2}_{10}$ o $\mathbf{10}_2$. En este caso, el 0 se toma como resultado de la suma y 1 como acarreo de ella. La columna **Suma** se implementa con una compuerta **OR-EXC** (OR-exclusiva, la salida es 1 solamente cuando un número impar de sus entradas lo es y es 0 cuando este número es par), mientras que la columna **Acarreo** se implementa con una compuerta AND normal. Como desventaja, en un semisumador binario no es posible considerar el acarreo (en inglés, *carry*) proveniente de circuitos previos cuando se suman múltiples bits. Una forma sencilla de solucionar esa desventaja es mediante el circuito sumador binario conocido como **sumador total**, un sistema combinacional con tres entradas y dos salidas.

Sumador Total de 1 bit con acarreo en la entrada (C-in)

Esquema eléctrico

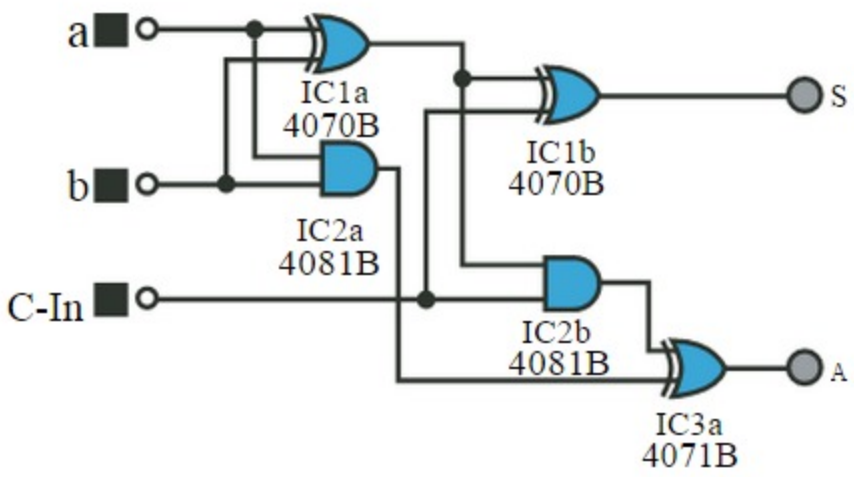


Tabla de verdad

C-in	a	b	Suma	C-0ut
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Expresiones booleanas: Suma = ⊕b⊕-in

Figura 16. Sumador Total de 1 bit con acarreo en la entrada.

Al analizar este sumador, se observa que consta de dos semisumadores conectados juntos y está conformado por tres compuertas OR-EXC, dos compuertas AND y una compuerta OR. En total, seis compuertas lógicas. La tabla de verdad incorpora una columna adicional, **C-in**, para considerar el acarreo proveniente de otros circuitos digitales.

Los circuitos integrados **TTL 74LS83** y **74LS283** poseen en su interior sumadores totales de 4 bits.

En la siguiente figura se presentan el símbolo del sumador total binario y el circuito lógico a partir de la interconexión de dos semisumadores binarios.

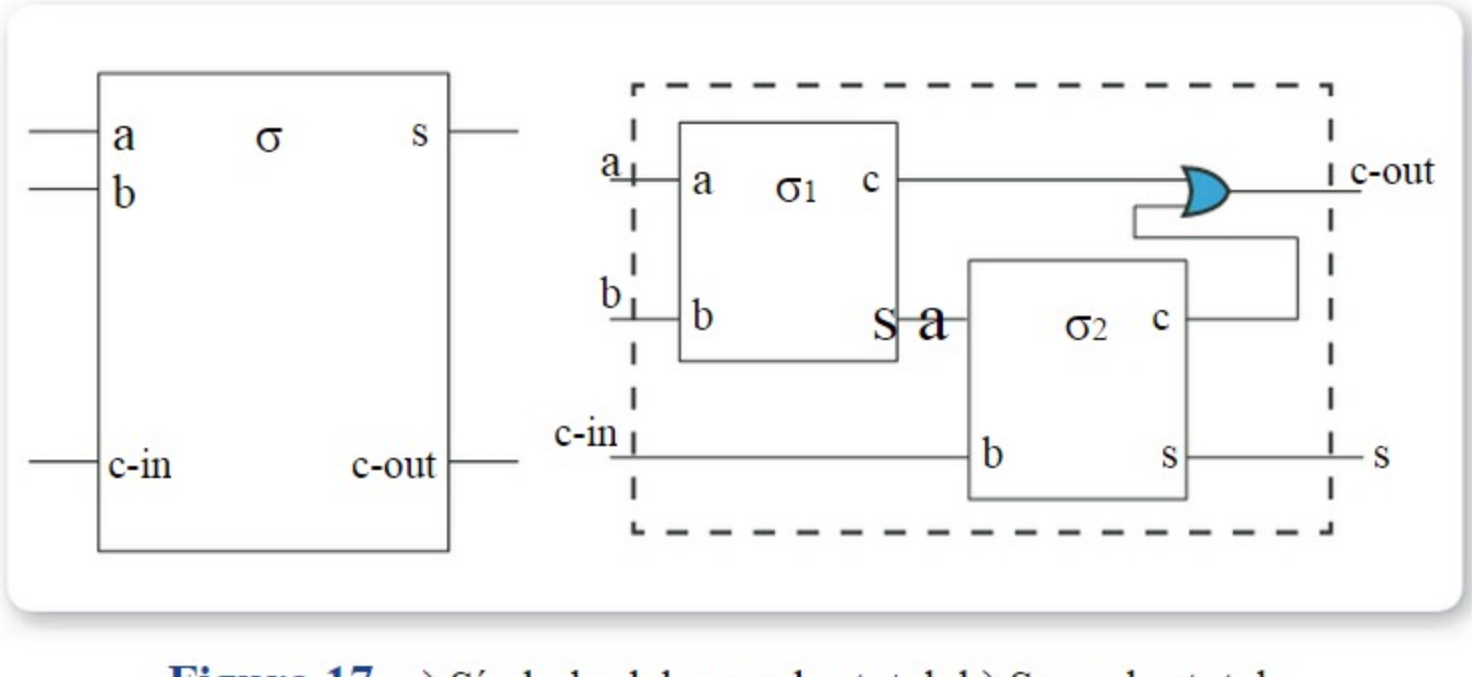


Figura 17. a) Símbolo del sumador total. b) Sumador total a partir de la interconexión de dos semisumadores binarios.

Un esquema más avanzado es el **cuádruple sumador total de 4 bits** que facilita la suma binaria de estos bits, también conocido como **sumador paralelo con acarreo serie** . Consta de cuatro sumadores totales en cascada. La suma comienza desde los bits menos significativos **LSB**, y el resultado es una salida. Si como consecuencia de la operación aparece un acarreo, se traslada al siguiente sumador como C-in, y así sucesivamente. El acarreo de entrada, **C-in = 0**, se conecta a 0 volt.

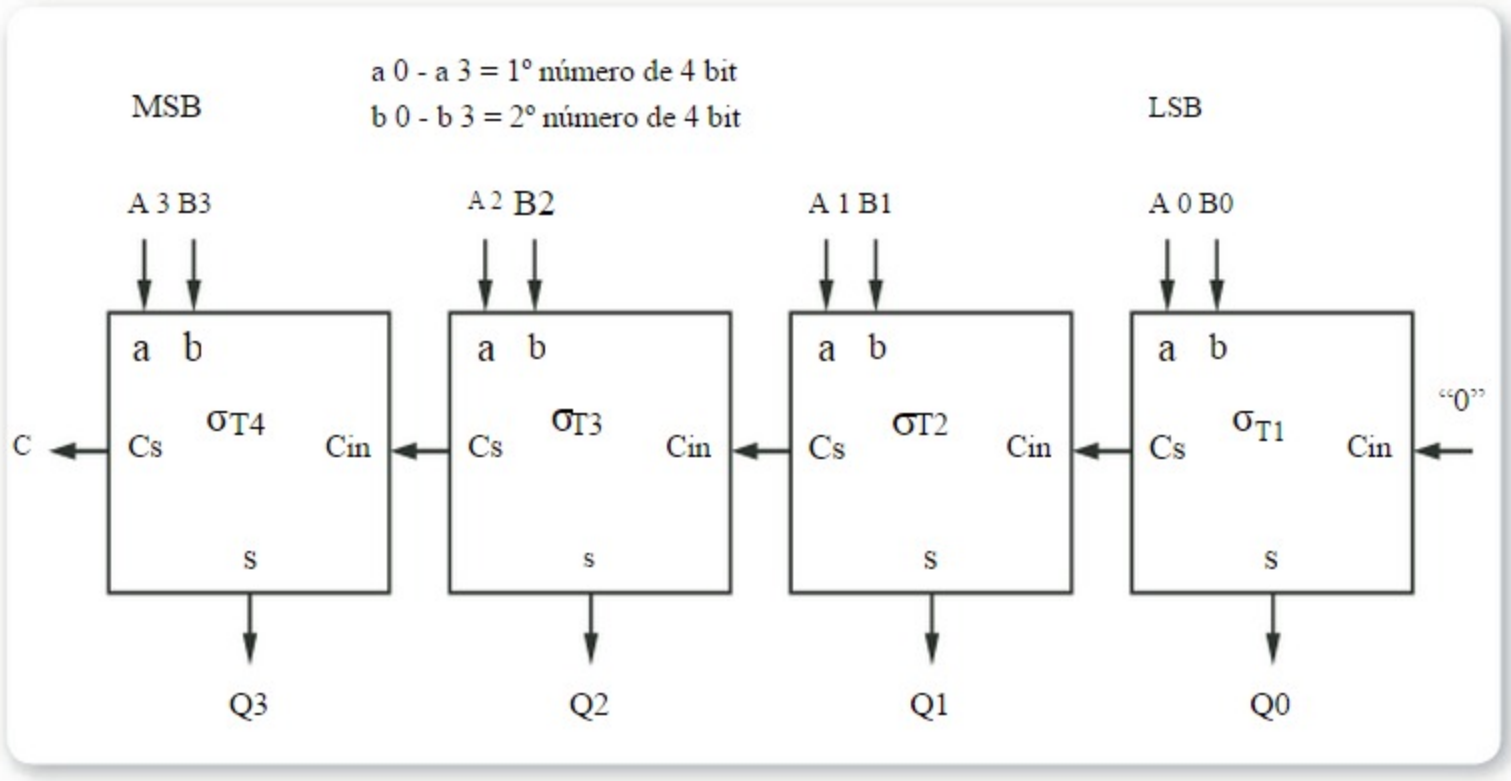


Figura 18. Esquema del cuádruple sumador total de 4 bits.

Si asociamos estos bloques, es posible construir sumadores de orden superior, por ejemplo, 16 bits (2 bytes).

El **sumador paralelo con acarreo paralelo** o **acarreo anticipado** elimina las limitaciones del sumador anterior y realiza la suma de dos grupos de n bits, cada uno aumentando la velocidad de procesamiento respecto del sumador paralelo con acarreo serie, al permitir la generación simultánea de todos los bits de acarreo en el mismo proceso de cálculo de las sumas parciales. A partir del sumador de cuatro bits, es posible construir un **restador binario** o sumador por complemento a 2 si al restar se complementa el segundo número a 2, invirtiendo 0 por 1, y viceversa, y sumando 1. La suma de 1 se realiza mediante la entrada de acarreo conectada directamente a + 5 volts o 1 a diferencia del sumador de cuatro bits donde se conecta a 0 volt.

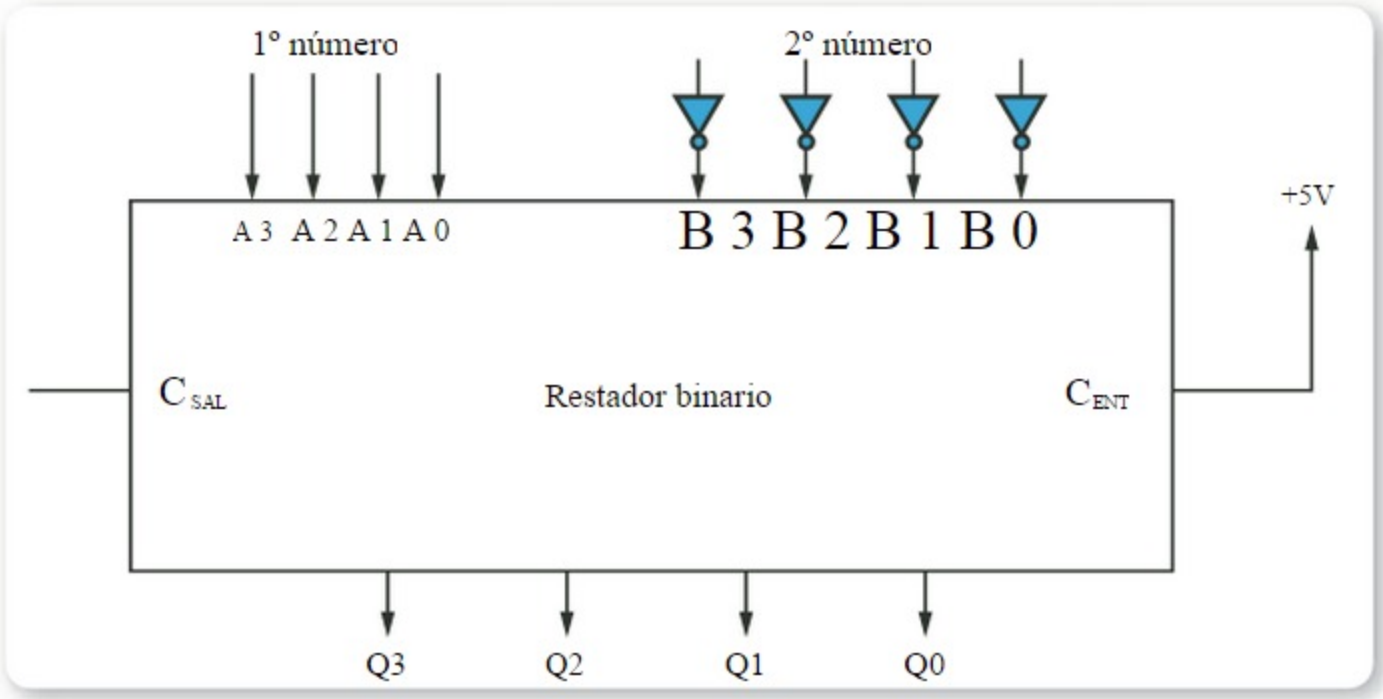


Figura 19. Resta binaria mediante complemento a 2.

Otros sistemas combinacionales

Otros sistemas combinacionales de interés son **comparadores**, **codificadores**, **decodificadores** y **multiplexores**. Un comparador

RETARDO DE PROPAGACIÓN EN UN SUMADOR

La principal desventaja del sumador paralelo con acarreo serie es el retardo de propagación: debemos esperar n tiempos de propagación de cada sumador, antes de obtener un resultado estable y fiable para la suma. Por esta razón, no se aconseja su uso en sistemas que precisen una cierta velocidad y utilicen palabras grandes de información.

es un circuito lógico que tiene por función determinar si dos bytes de entrada son iguales o no. Se construye alrededor de compuertas AND, NOR y NOT que comparan las señales digitales presentes en sus terminales de entrada, y produce una salida que depende de la condición de esas entradas.

En realidad, se tienen dos tipos principales de comparadores digitales: el **comparador de identidad** y el **comparador de magnitud**. En el primer caso, el comparador dispone solamente de una salida que es igual a 1 cuando A = B (es decir, A = B = 1) o 0 en caso contrario (A = B = 0). En el segundo caso, el comparador tiene tres terminales de salida, una para cada igualdad: A = B, A > B y A < B; siendo A: a1, a2, a3,…, an, etcétera y B: b1, b2, b3,… bn, etcétera. En la figura se observa un comparador digital de magnitud de 1 bit. No distingue entre dos ceros o dos unos cuando A = B, y cuando A = B se asemeja a una compuerta NOR-EXC donde **S = (A⊕B)'**.

Comparador digital de magnitud de 1 bit

Esquema eléctrico

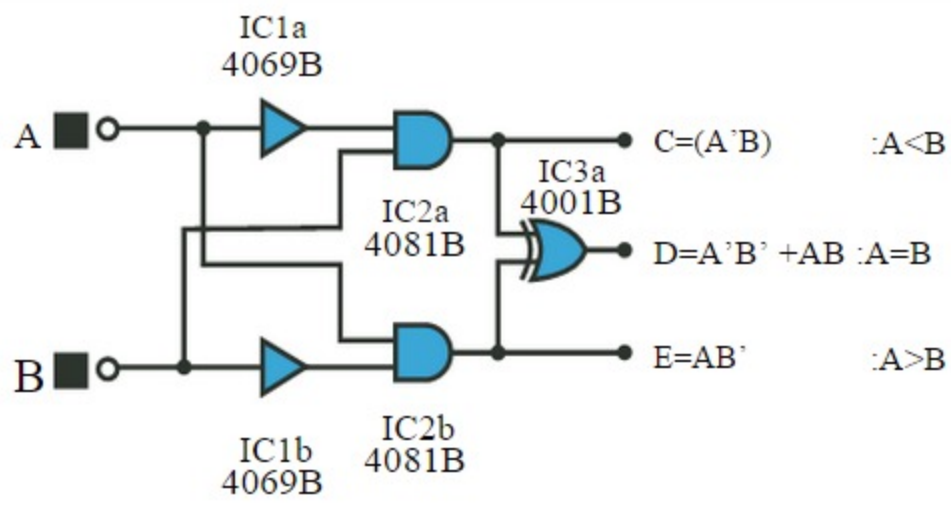


Tabla de verdad

0	0	0	1	0	
0	1	1	0	0	
1	0	0	0	1	
1	1	0	1	0	

Expresión booleana para A= B: S = (A⊕B)'

Figura 20. Comparador digital de magnitud de 1 bit.

APLICACIÓNDE LOS COMPARADORES DIGITALES

Los comparadores digitales poseen una amplia variedad de aplicaciones que podemos encontrar en **convertidores analógico-digitales** (**ADC**) y también en unidades **aritmético lógicas** (**ALU**). Esto hace posible implementar toda una variedad de operaciones aritméticas.

En la práctica, los comparadores se construyen mediante compuertas NOR-EXC. Si se comparan dos valores en binario, o en BCD, o variables una contra otra, se está comparando la magnitud de esos valores y se tiene un comparador de magnitud. Además de comparar bits individuales, como en el ejemplo anterior, es posible diseñar comparadores de mayor cantidad de bits colocando comparadores en cascada y así obtener un comparador de **n-bit**.

EL SUMADOR BINARIO ES UNO DE LOS CIRCUITOS LÓGICOS COMBINACIONALES MÁS UTILIZADOS

Un comparador **multi-bit** es útil para comparar tanto palabras en binario como en BCD y producir una salida, considerando si una de ellas es más larga, igual o menor que la otra. Un ejemplo de aplicación es el **comparador de magnitud de 4 bits** en el que dos palabras de 4 bits, o **nibbles**, se comparan una con la otra para generar una salida pertinente con una de las palabras conectadas a la entrada A, y la otra a la entrada B.

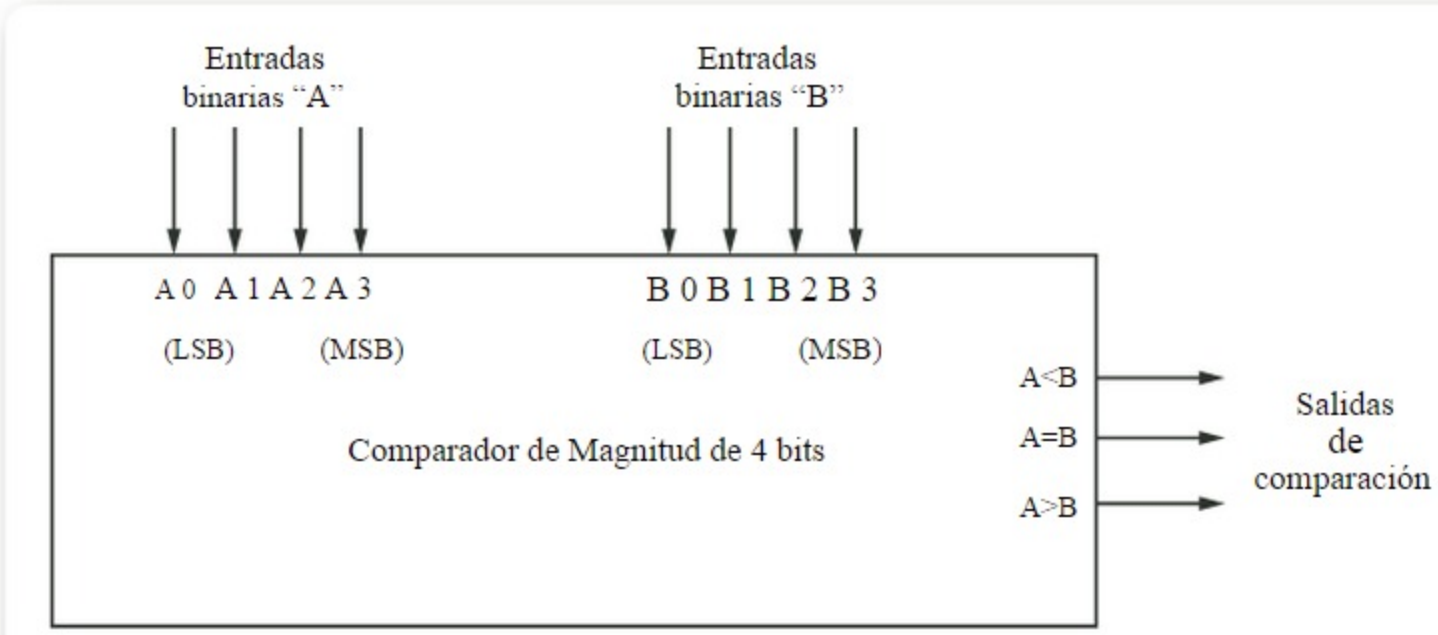


Figura 21. Comparador de magnitud de 4 bits.

Desde el punto de vista comercial, están disponibles comparadores de cuatro bits como el **74LS85** con tecnología TTL o el **4063** bajo tecnología CMOS. Ambos disponen de entradas para conectarlos en serie y así poder construir comparadores de magnitud de longitud de palabra mayor a 4 bits, es decir, n bit. Las salidas de un comparador se conectan a las entradas correspondientes en forma directa y así se pueden comparar palabras de 8, 16 o 32 bits.

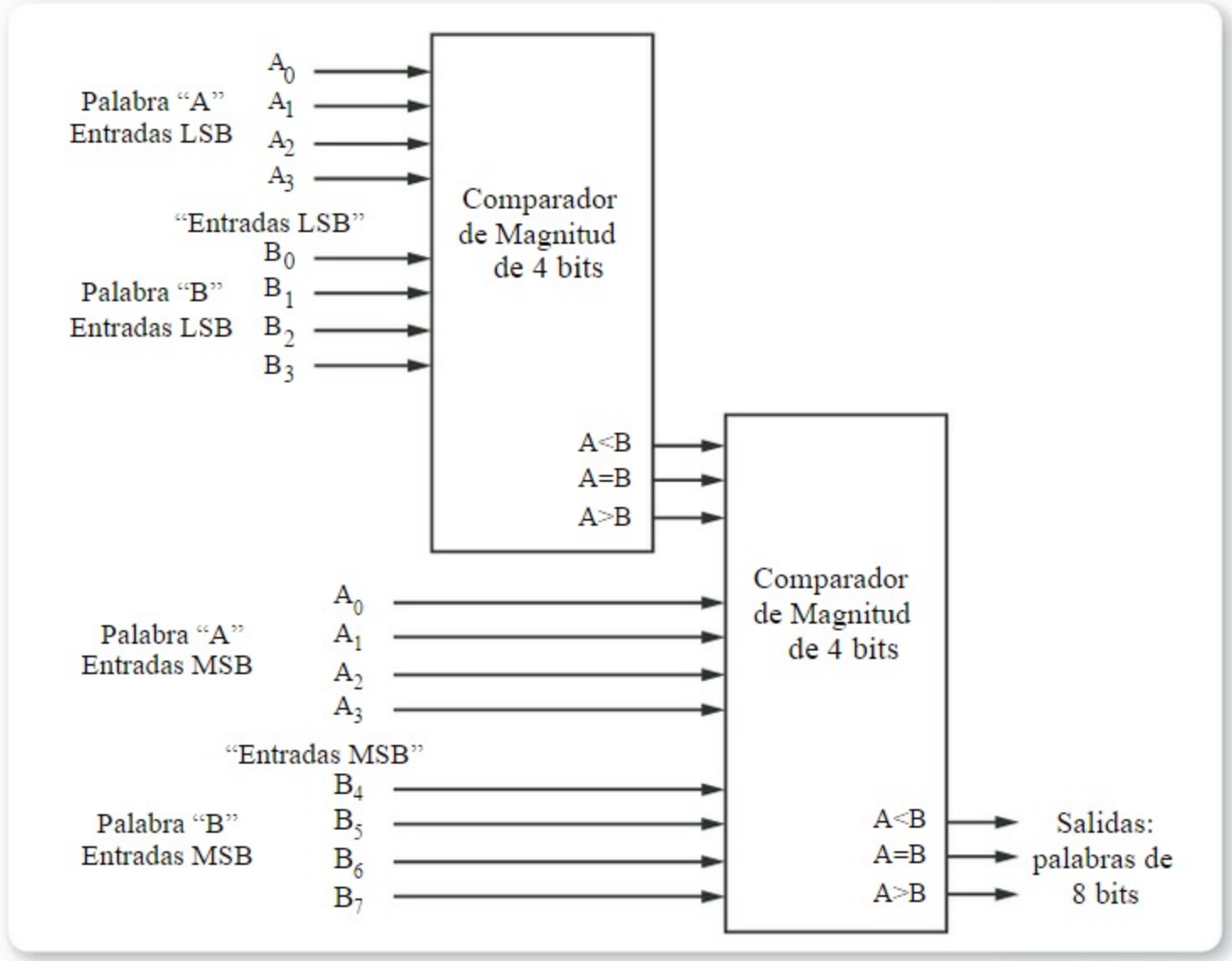


Figura 22. Comparador de magnitud de 8 bits.

El proceso de comparación comienza cuando el comparador compara primero los bits de mayor orden (**MSB**) para reducir el tiempo. Si esta comparación da igual, entonces se comparan los bits de menor orden (**LSB**). Si se mantiene la igualdad es porque ambos números comparados son iguales. En caso contrario, si MSB no da igual, sea por A > B o A < B, se detiene la comparación entre cualquier bit adicional de menor orden, ya que la relación entre ambos números ha sido determinada.

Un circuito digital **decodificador** decodifica información codificada desde un formato a otro. Por ejemplo, en un decodificador decimal, se transforma el conjunto de entradas digitales en código decimal equivalente como salida. En particular, un **decodificador binario** es un tipo de dispositivo digital con entradas para códigos de 2, 3 o 4 bits. Si el número de variables de entrada es **n**, entonces $\mathbf{2^n}$ es la cantidad de funciones de salida del circuito que se activarán de acuerdo a la combinación que adopten las variables de entrada.

En general, el código de salida del decodificador tiene mayor cantidad de bits que el código de entrada y, en la práctica, un

decodificador binario puede incluir distintas configuraciones: 2 a 4, 3 a 8 y 4 a 16. En las aplicaciones digitales, los decodificadores se utilizan para seleccionar posiciones de memoria. Por ejemplo, un **bus de direcciones** de **n** bit podrá seleccionar 2^n posiciones de memoria.

Un decodificador comercial para 2 bits de entrada (n = 2) y cuatro salidas es el TTL **74155**; para entradas de 3 bits (n = 3) y ocho salidas, el TTL **74138**; para 4 bits de entrada y 16 salidas, el TTL **74154**.

Se puede decir que un decodificador es un **demultiplexor** con una entrada adicional de datos que se utiliza para habilitar el funcionamiento del decodificador. Esta entrada se conoce como **Enable**, y el **CI TTL 74138** que dispone de esta entrada podría funcionar tanto como decodificador como demultiplexor. Enable se habilita con 1 y se deshabilita aplicándole un 0.

Decodificador 2 a 4 (de 2 variables de entrada a 4 variables de salida)

Esquema eléctrico

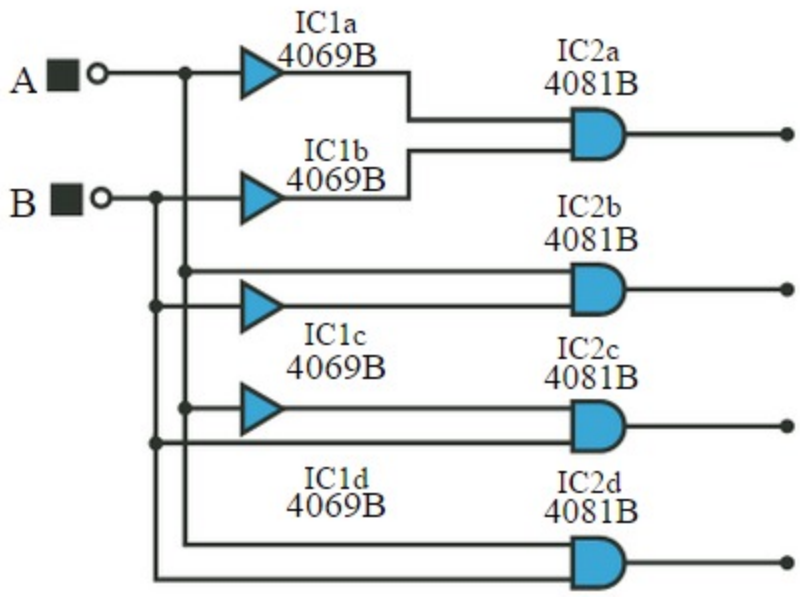
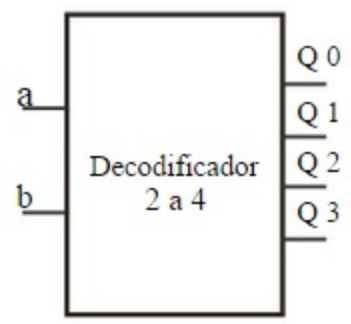


Tabla de verdad

a	b	Q_0	Q_1	Q_2	Q_3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1



Expresiones booleanas: $Q_0 = a' b'$; $Q_1 = a\, b'$; $Q_2 = a'\, b$; $Q_3 = a\, b$

Figura 23. Decodificador de dos entradas a cuatro salidas.

CIRCUITOS INTEGRADOS EN DECODIFICADORES

En decodificadores comerciales, disponemos de los circuitos integrados **7442** (TTL). También es posible encontrar la versión CMOS **4028** como decodificadores **BCD a decimal** y el circuito integrado TTL **7447** como decodificador **BCD a 7 segmentos** , **4511** en la versión CMOS.

UN COMPARADOR ES UN CIRCUITO LÓGICO QUE DETERMINA SI DOS BYTES DE ENTRADA SON IGUALES

Es conveniente utilizar compuertas AND como elemento básico de decodificación, ya que su salida es una solamente cuando todas las entradas están en 1. Si este esquema se implementara con compuertas NAND, para cada condición de entrada se obtendría un 0 en la salida correspondiente, el resto de ellas se mantendría en 1. Se puede implementar un decodificador binario 4 a 16 líneas mediante dos decodificadores 3 a 8. Las entradas **a**, **b** y **c** se utilizan para seleccionar cuál salida de este decodificador se habilitará (1). La entrada **d** permite seleccionar el decodificador habilitado: un 1 habilita el decodificador inferior, mientras que un 0 lo hace con el superior.

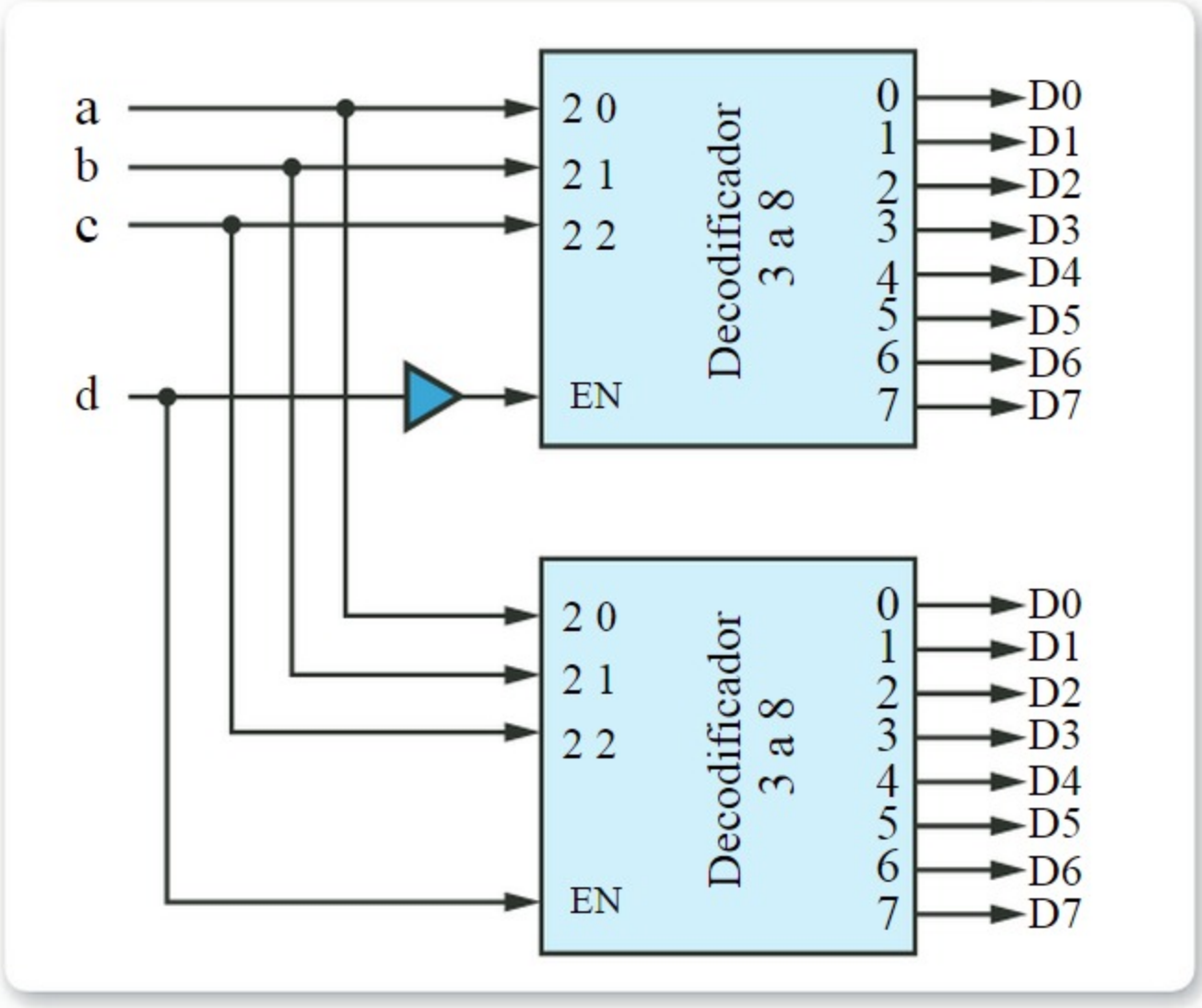


Figura 24. Decodificador binario de 4 a 16 líneas mediante dos decodificadores de 3 a 8.

Existe un límite práctico para el número de entradas por utilizar en un decodificador particular dado que, al incrementar n, la cantidad de compuertas AND requeridas para generar una salida en el decodificador también crece. Así, resulta necesaria una gran corriente de salida para

que las compuertas AND, que se encuentran hacia la salida, conmuten adecuadamente. Este parámetro se conoce como **cargabilidad de salida** (en idioma inglés, *fan-out*).

Un circuito digital **codificador** genera **n** variables de salida a partir de $\mathbf{2^n}$ variables de entrada realizando el proceso inverso al decodificador y, de acuerdo a la entrada activada con 1, será la combinación de 0 y 1 que se obtenga en la salida. Por ejemplo, si en la entrada $\mathbf{A_3}$ se coloca un 1 lógico, en la salida del codificador aparecerá la combinación $\mathbf{3}_{10}$ o $\mathbf{011}_2$.

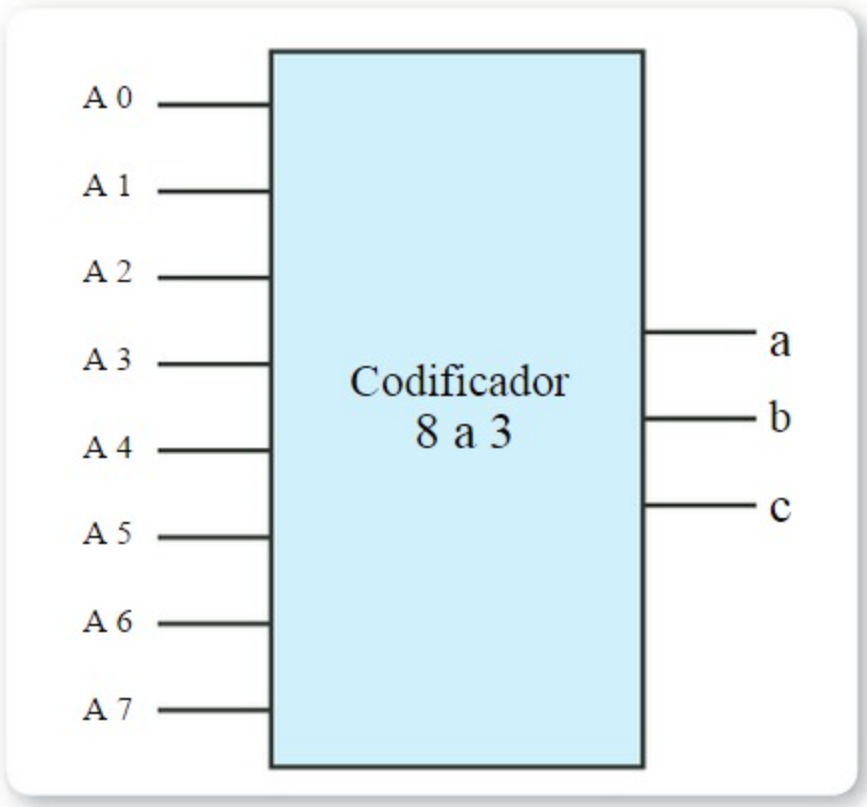


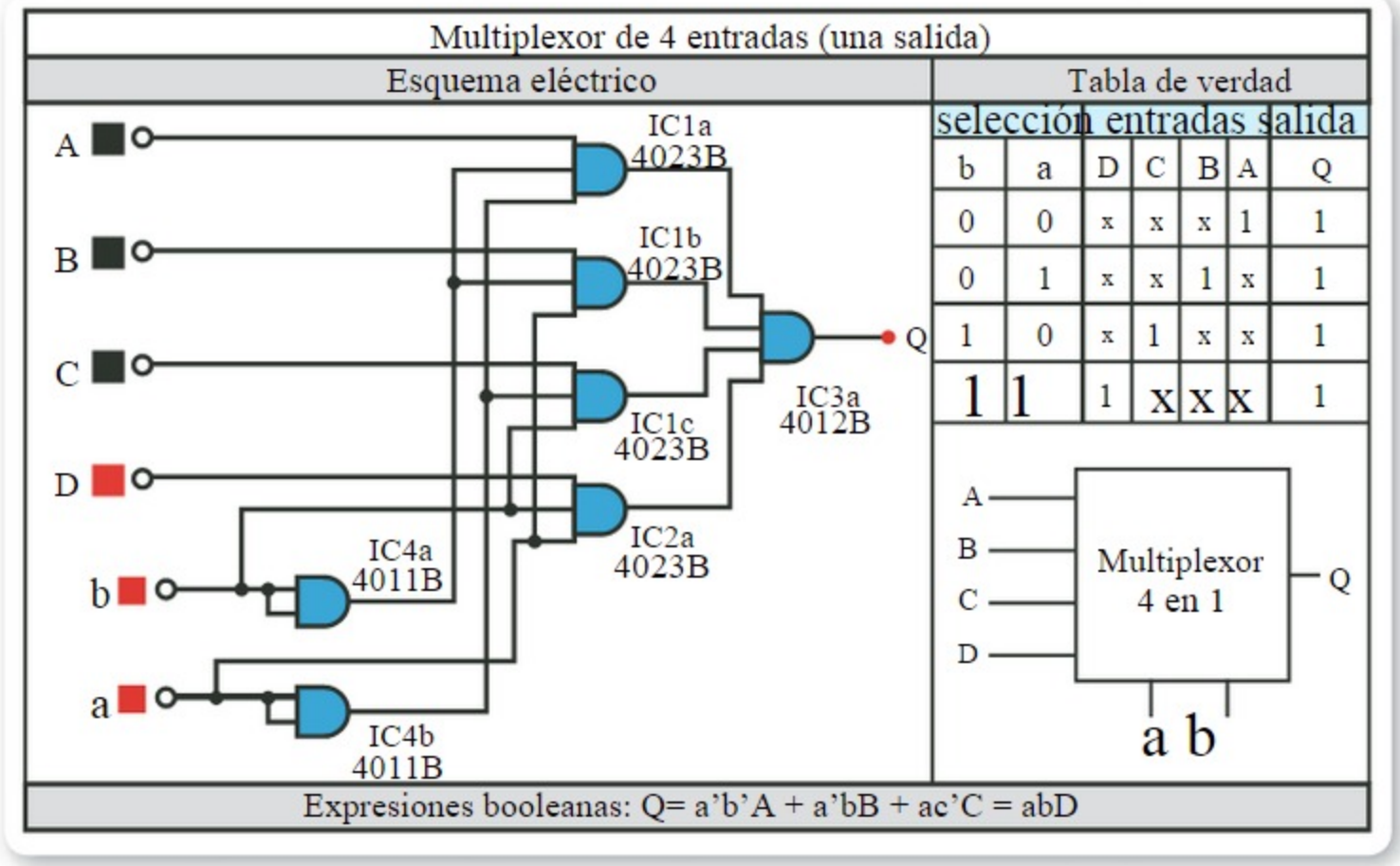
Figura 25. Codificador ocho líneas de entrada a tres de salida.

El **multiplexor** es un circuito lógico de **x** variables de entrada, **n** variables de selección (tal que $\mathbf{X = 2^n}$) y una salida única **Q,** de modo de seleccionar una de las entradas y enviarla a la salida del dispositivo.

En el multiplexor se utiliza un número de entradas adicionales llamadas **líneas de control** para que, de acuerdo con la condición binaria de esta entrada de control 1 o 0, la entrada de datos deseada se conecte directamente con la salida.

MULTIPLEXADO DIGITAL

El **multiplexado digital** es la operación que envía una o más señales digitales sobre una salida común en tiempos diferentes. Permite, entonces, reducir el número de compuertas lógicas requerido en un circuito o cuando se necesite una única línea o bus de datos para transportar una o más señales digitales diferentes.



selección		entradas				salida
b	a	D	C	B	A	Q
0	0	x	x	x	1	1
0	1	x	x	1	x	1
1	0	x	1	x	x	1
1	1	1	x	x	x	1

Figura 26. Esquema eléctrico y tabla de verdad del multiplexor de cuatro entradas.

Se observa que, en cualquier instante de tiempo, solamente una de las cuatro entradas está presente en la salida Q. Cuál de ellas será, dependerá del código de direccionamiento que se haya considerado en las entradas de control a y b. Por ejemplo, si se desea seleccionar solamente la entrada B para que aparezca en la salida Q, la combinación binaria de las variables de control será a = 1 b = 0, sin importar el valor del resto de las entradas A, C y D (indistinto: **x**). En los comercios se encuentran multiplicadores de ocho entradas o canales, como el TTL **74151**.

Sistemas secuenciales

En los **sistemas combinacionales** , el valor de la salida en un instante de tiempo depende del valor de las señales de entrada en ese mismo momento. Cuando las señales de entrada varían, como consecuencia, también varía la señal de salida. Por el contrario, los **sistemas secuenciales** son sistemas lógicos donde las señales de salida no dependen solo del valor de las entradas en el mismo momento, sino también del valor anterior de esas entradas. Así, presentan la capacidad de memoria para las secuencias de valores que toman las señales, y son esas secuencias las que les proporcionan su nombre.

Estos circuitos se conocen también como **circuitos de dos estados** o **biestables**, ya que las salidas pueden estar en estado lógico 1 o en estado lógico 0 y permanecen enclavados indefinidamente en el estado actual hasta que se les aplique un pulso de disparo, para provocar que el biestable cambie nuevamente el estado de la salida. En la siguiente figura se presenta el modelo clásico de un sistema secuencial, y los diferentes bloques y señales que lo conforman, donde se pueden distinguir las señales de entradas, de salida y de estado.

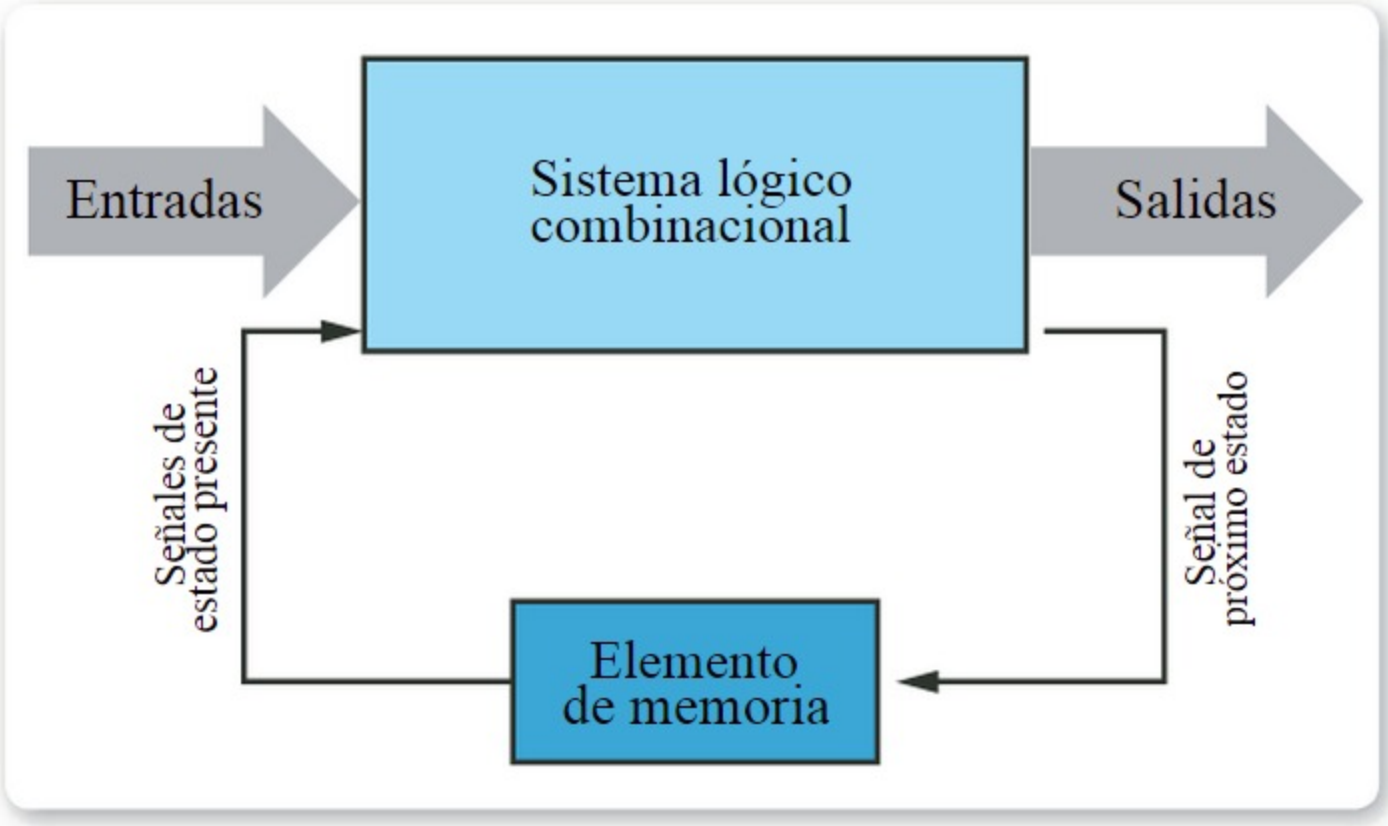


Figura 27. Diagrama en bloques de un sistema combinacional.

Tanto las señales de entrada como de salida tienen el mismo significado que en los sistemas combinacionales, mientras que las señales de estado, específicas de los sistemas secuenciales, mantienen la información de la historia pasada del sistema. Además, las señales de estado se clasifican según se consideren a la salida o a la entrada del **elemento de memoria** .

Si consideramos los elementos de memoria a la salida (entrada del bloque combinacional), se denominan **señales de estado presente** e indican el estado en el que se encuentra el sistema para realizar una operación, mientras que, si se consideran las señales de estado a la entrada de los elementos de memoria (la salida del bloque combinacional), se denominan **señales del próximo estado**, puesto que indican el estado al que llegará el sistema secuencial luego que el bloque combinacional haya realizado la operación.

LA SEÑAL DE SALIDA DEPENDE DEL VALOR DE ENTRADA DE ESE MOMENTO Y DEL ANTERIOR

Una forma simple de clasificar los sistemas secuenciales de acuerdo a cómo gestionan el tiempo es en: **asíncronos** (que cambian de estado inmediatamente) y **síncronos** (el cambio de estado está sincronizado con una señal de reloj específica).

La señal de reloj **clk** (en inglés, clock) es el mecanismo de sincronización de circuitos utilizado para determinar en qué instantes de tiempo el circuito secuencial debe cambiar de estado al tomar el valor 0 o 1 de forma cíclica y continua, desde que el circuito se pone en marcha hasta que se detiene. Una señal de reloj se caracteriza por la duración del ciclo que forma la secuencia de valores 0 y 1, antes que se repita, y que se denomina **período T**. Este posee una duración determinada en nanosegundos y constante. La **frecuencia f** del reloj es la inversa del período, **1/T**, y determina el número de ciclos de reloj que ocurren durante un segundo. Si bien se mide en **hertz** (**Hz**, ciclos por segundo), es habitual utilizar el múltiplo **gigahertz**, **GHz**, que representa mil millones de ciclos en un segundo.

Considerando los intervalos en que la señal de reloj cambia de 0 a 1, entre flancos ascendentes, el intervalo entre un flanco ascendente y el siguiente se denomina **ciclo** o **ciclo de reloj** y tiene una duración de **T [seg]**. La señal de reloj permite sincronizar circuitos de varias formas, una de ellas es mediante la sincronización por **flanco ascendente** (la más utilizada), aunque otras son por **flanco descendente**, por **nivel 0** y por **nivel 1**.

Dada la característica de memorizar el estado de la salida, mediante flip-flop se pueden construir **memorias de un bit** . Se trata de un dispositivo electrónico digital capaz de almacenar un bit de información manteniendo las salidas estables, aunque se modifique

BLOQUES CONSTRUCTORES

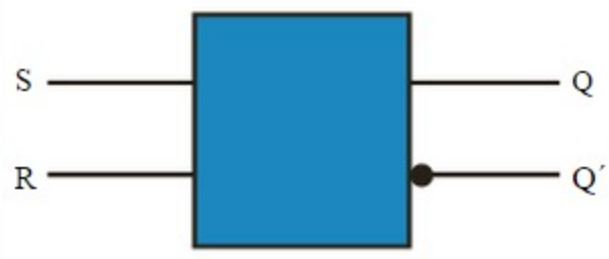
Es posible construir un biestable mediante flip-flop, y estos se podrían montar combinando **compuertas NAND**. A partir de los flip-flop, es posible construir sistemas secuenciales más complejos, tales como los registros de almacenamiento, los dispositivos de memoria y los contadores.

el estado de sus entradas. Estos dispositivos también se conocen como **multivibradores** . De acuerdo a la cantidad de estados estables del multivibrador, se encuentran distintos dispositivos: si tiene dos estados estables, se denomina **multivibrador biestable** o flip-flop. A un multivibrador con un estado estable único, se lo conoce como **multivibrador monoestable** . Por otro lado, un multivibrador que no tiene ningún estado estable y cambia de estado permanentemente es un astable u oscilador de ondas cuadradas.

Un elemento básico de memoria, multivibrador biestable, biestable o flip-flop (en inglés, latch), tiene dos estados estables y, a partir de estos elementos básicos de memoria, se construyen otros más complicados, como registros de desplazamiento y contadores.

De acuerdo a la lógica utilizada y al tipo de disparo, existen cuatro tipos principales de flip-flop, conocidos como **RS**, **JK**, **D** y **T**, respectivamente. En un biestable, se puede realizar un cambio de estado por nivel de tensión o por cambio entre niveles lógicos (disparo por flanco); también de manera sincrónica mediante pulsos de reloj, o asincrónica, si las salidas cambian solo cuando cambian las entradas.

Se explica el funcionamiento de un flip-flop RS con una tabla de verdad considerando la realimentación salidas a entradas.



S	R	Q	Q´
0	0	memoria	
0	1	0	1
1	0	1	0
1	1	prohibido	

Figura 28. Símbolo lógico y tabla de verdad de un flip-flop RS con sus entradas SET y RESET, y dos salidas Q y Q' (inversa de Q).

En el estado **SET**, S = 1 y R = 0, Q' = 0 y con R = 0, Q = 1.

En el estado **RESET**, S = 0 y R = 1, ocurre lo contrario: Q = 0 y Q' = 0.

Si ahora S = 0 y R = 0, el biestable RS se comporta como una memoria elemental de un bit que conserva la última configuración de salida. Finalmente, si S = 1 y R = 1, las salidas Q y Q' tomarían el valor 0, valor prohibido puesto que el estado de Q y Q' debe ser inverso.

Un flip-flop **RS asíncrono** contiene dos puertas **NOR** que realimentan las salidas entre sí y que se modificarán ante cambios en las entradas. Así, se representa cualquier tipo de flip-flop mediante símbolos específicos. Por ejemplo, para un flip-flop RS se emplea un rectángulo especial.

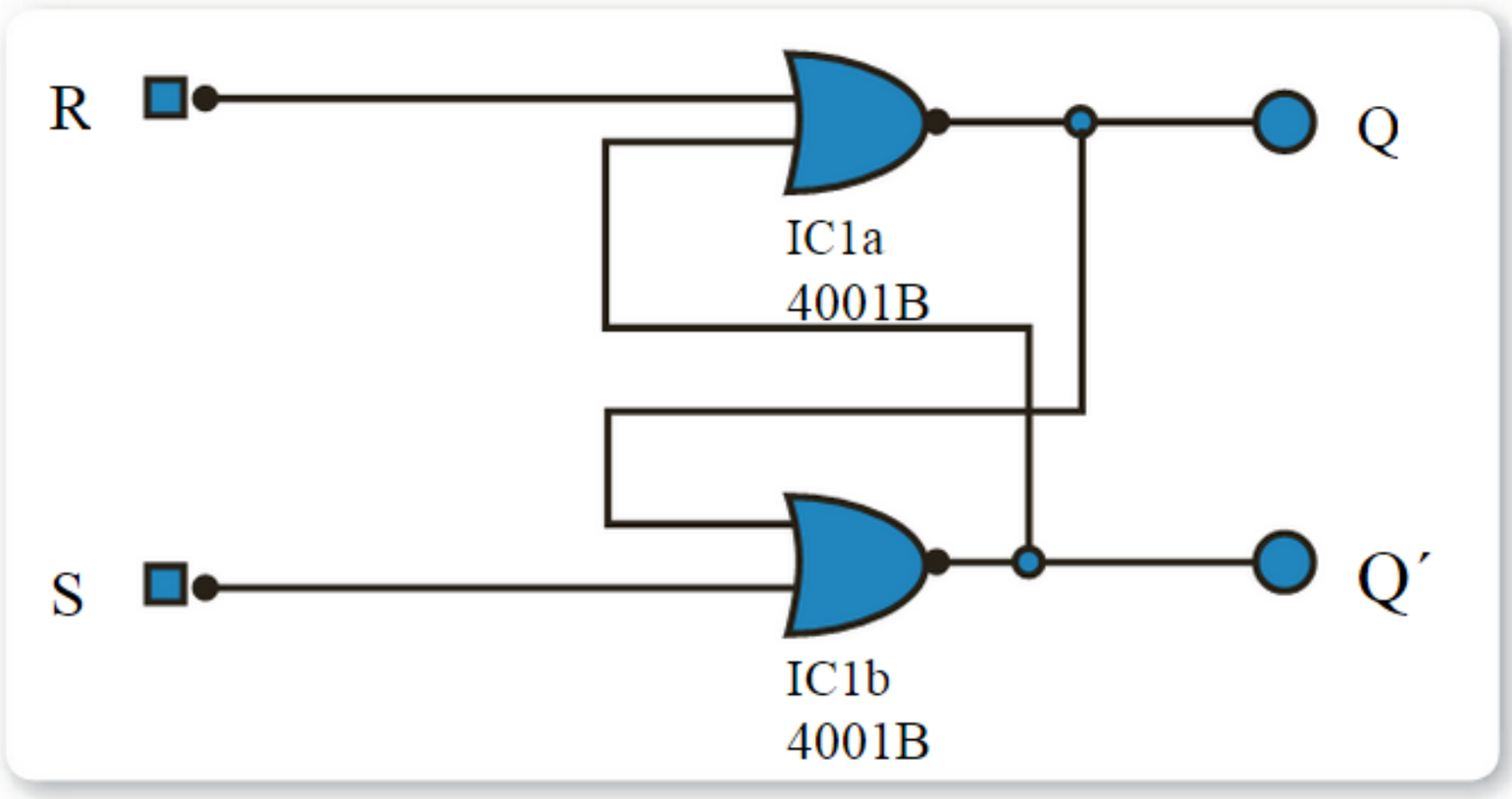


Figura 29. Flip flop RS asíncrono construido mediante compuertas NOR.

Un **flip-flop asíncrono** tiene algunas desventajas, entre ellas, la posibilidad de generar estados de salida ambiguos, baja inmunidad al ruido y sensibilidad a las carreras lógicas; un problema frecuente en circuitos con relés en los que, cuando varias señales lógicas indican cerrar simultáneamente varios relés, los más rápidos cierran primero sus contactos, y los más lentos lo hacen un tiempo después. Durante ese lapso, el circuito lógico se descontrola, y se torna impredecible el resultado final.

Por eso es necesario utilizar **flip-flop síncronos** en diferentes aplicaciones. Podemos construir un biestable síncrono a partir de uno asíncrono. Para un flip-flop RS, incorporamos una entrada de disparo, C, y así obtenemos un flip-flop **RS síncrono** .

ESTADO METAESTABLE

Es un estado prohibido que ocurre cuando las entradas toman valores inválidos, y las salidas Q y Q' se comportan de manera inestable. Afecta a biestables asíncronos y síncronos. Se evita manteniendo constantes las entradas del biestable durante un tiempo especificado antes y después del pulso de reloj, **setup time** (tsu) y **hold time** (th), respectivamente.

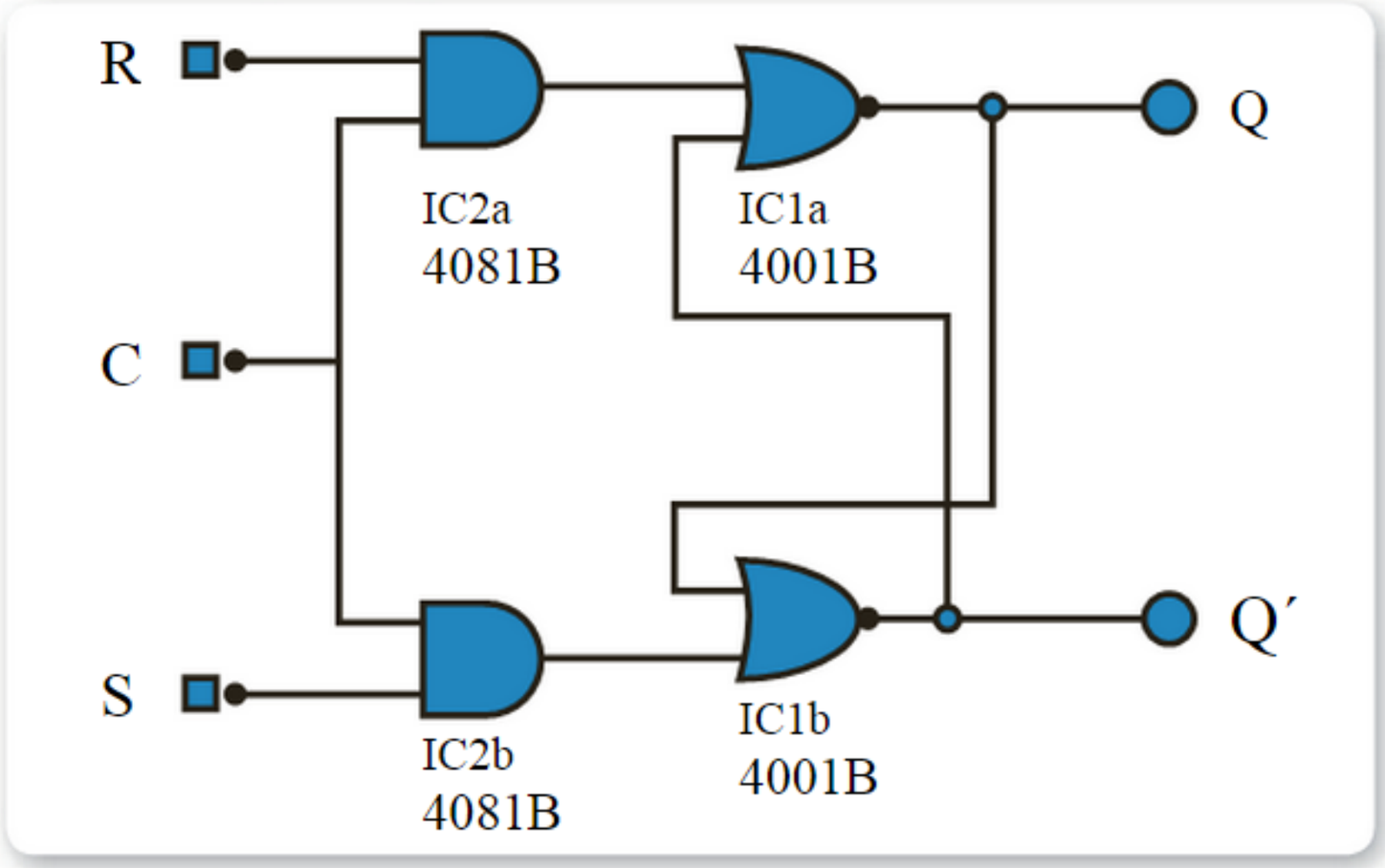


Figura 30. Se construye un flip flop RS síncrono sumando al esquema anterior dos compuertas AND de modo de incorporar una entrada de disparo o entrada de control, C.

La **entrada de disparo** , C, permite el cambio de las salidas si está presente y, en general, proviene de un reloj. En los biestables síncronos también se produce el **estado metaestable** si una de las entradas del flip flop está cambiando cuando llega el pulso de reloj a la entrada del mismo nombre.

El circuito electrónico encargado de generar los pulsos de reloj es el **multivibrador astable** . Como su nombre lo indica, no tiene estados estables, por lo que es capaz de generar una señal de reloj u onda cuadrada, es decir, una sucesión continua de niveles altos (1) y bajos (0) de tensión que se repiten en el tiempo.

Si se define la frecuencia de oscilación del astable mediante un circuito RC externo, el circuito electrónico se conoce como **oscilador RC** (**Resistencia-Condensador**). La onda cuadrada resultante del oscilador tiene un período T que es la suma de dos períodos T1 y T2. Si T1 = T2, la onda cuadrada es simétrica y, si son diferentes, la onda cuadrada resultante es asimétrica.

La ventaja de un oscilador astable RC es la simplicidad, mientras que su mayor desventaja es la estabilidad, del orden de 0.1 %. Esto implica que, si se genera una onda cuadrada de 1 KHz (1000 Hz), la frecuencia real de la onda estará comprendida entre 1000 Hz – 0.1 % de 1000 Hz, y 1000 Hz + 0.1 % de 1000 Hz, es decir, entre 999 Hz y 1001 Hz.

LOS SISTEMAS SECUENCIALES SE PUEDEN CLASIFICAR EN SÍNCRONOS O ASÍNCRONOS

Un rango de frecuencias demasiado grande para circuitos digitales que requieran de una señal de reloj más estable, lo que se logra recurriendo a un oscilador controlado mediante cristal de cuarzo.

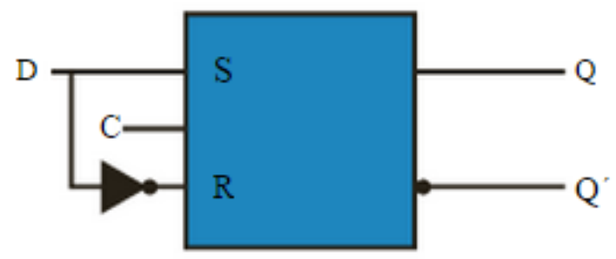
Del mismo modo, se podría construir un oscilador astable mediante el conocido **555**, que utiliza tecnología de transistores bipolares. Es compatible con lógica TTL y CMOS, y funciona con amplias condiciones de tensión de alimentación (desde 4.5 V hasta 18 V). El ciclo de trabajo, es decir, la relación entre los períodos de la onda cuadrada, cuando está en alto y cuando está en bajo (D %), depende de los valores de las resistencias utilizadas.

Para un ciclo de trabajo (en inglés, duty cicle) del 50 %, se deben seleccionar cuidosamente los valores de estas resistencias para que el período en el cual la señal está activa sea el deseado.

Por último, además de las alternativas mencionadas, podemos utilizar otros circuitos integrados, por ejemplo, generadores de pulsos de reloj, como el conocido **CD 4047B** (**CMOS**), configurable tanto astable como monoestable, o el circuito integrado especializado **XR-2240**, capaz de generar distintas formas de onda, por ejemplo, ondas cuadradas.

Otro tipo de biestable ampliamente utilizado es el **flip-flop D**, basado en el flip-flop RS síncrono, pero que prescinde de la entrada Reset (R) que se conecta a la salida de un inversor y cuya entrada se vincula a la entrada Set (S) del flip-flop, ahora la entrada **D**. De este modo, se elimina el estado no válido presente en la tabla de verdad en un flip-flop RS síncrono.

La entrada D controla los cambios en las salidas Q y Q', mientras que la entrada C controla el funcionamiento del flip-flop como memoria.



C	D	Q	Q´
0	0	memoria	
0	1	memoria	
1	0	0	1
1	1	1	0

Figura 31. Representación de un flip-flop D y tabla de verdad. La entrada de reloj es una onda cuadrada que cambia constantemente de nivel alto a nivel bajo.

Un flip-flop **JK** permite modificar la condición de indeterminación de sus salidas cuando la entrada **R** y la entrada **S** están en 1. Para ello, se modifica el flip-flop RS asíncrono agregando dos compuertas AND y dando origen a dos nuevas entradas llamadas **J** y **K**.

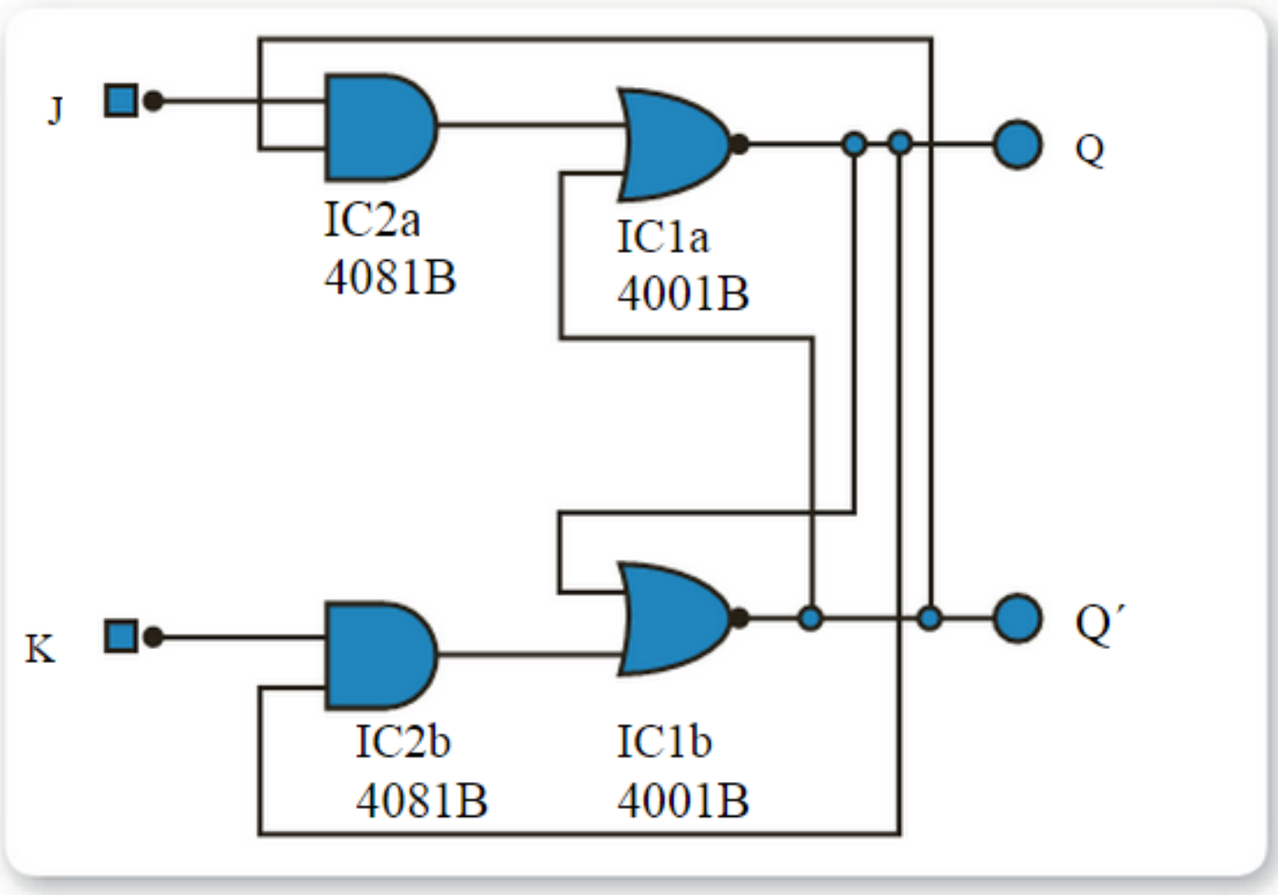
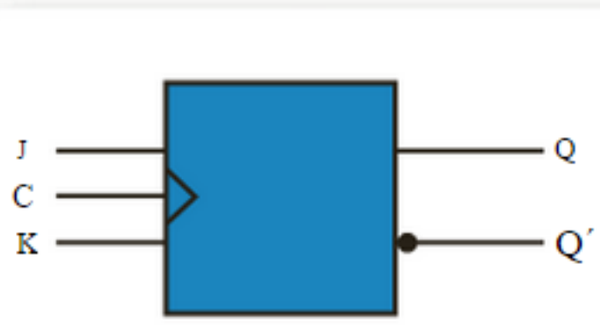


Figura 32. Un flip-flop JK asíncrono corrige la indeterminación en los flip flop RS con las entradas R y S en 1, evitando estados prohibidos.

La incorporación de un detector de flanco al flip flop JK asíncrono, lo transforma en un flip-flop **JK síncrono** disparado por flancos. Además de salvar la indeterminación existente en un flip flop RS (cuando S = R = 1), se tiene una entrada de reloj, C, que recibe la señal de reloj para sincronización.

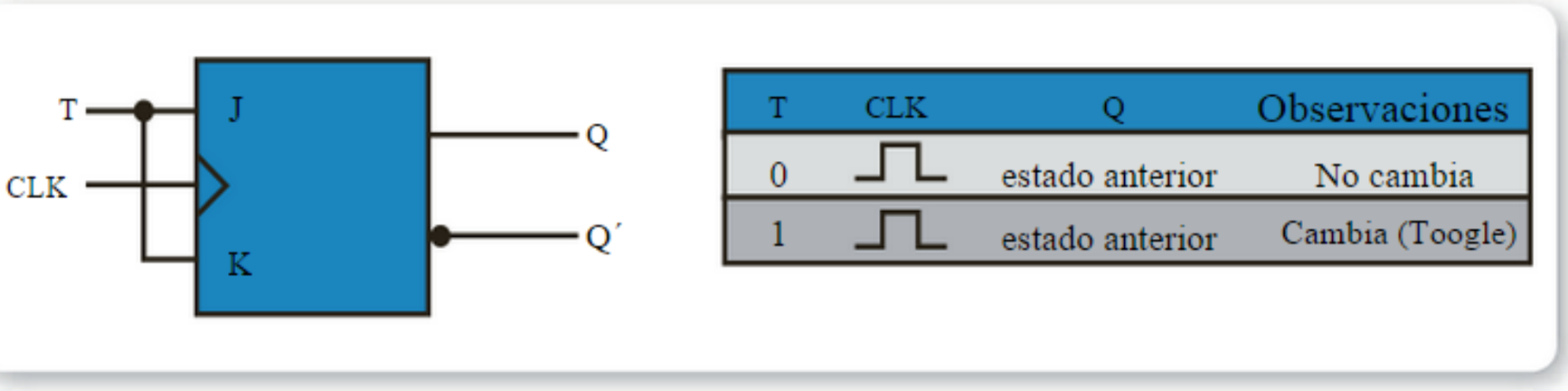


C	J	K	Q	Q´
⎍	0	0	memoria	
⎍	0	1	0	1
⎍	1	0	1	0
⎍	1	1	cambia	cambia

Figura 33. Un flip flop JK síncrono disparado por flancos es un flip flop JK asíncrono que incorpora una entrada de señal de reloj, C, para la sincronización.

Se obtiene un **multivibrador T** síncrono (en inglés, *toggle*) a partir de otros flip flop, por ejemplo, utilizando un flip flop JK con sus entradas J y K unidas para conseguir el nuevo flip flop. A diferencia de

otros flip flop, un multivibrador T permite modificar el estado de las salidas frente a variaciones en sus entradas. El flip-flop T construido a partir de un flip-flop RS se utiliza en contadores y divisores, entre otros dispositivos lógicos.



T	CLK	Q	Observaciones
0	⎍	estado anterior	No cambia
1	⎍	estado anterior	Cambia (Toogle)

Figura 34. Se obtiene un multivibrador T síncrono a partir de un flip-flop JK y uniendo las entradas J y K. Si T = 1, un cambio en la señal de reloj producirá una modificación en el estado de las salidas.

Una aplicación de los flip-flop JK es como divisor de frecuencia por dos mediante un circuito integrado **TTL 7473,** que contiene en su interior dos flip flop JK. Uniendo las entradas J y K, se obtiene un flip-flop T y, conectándola a + V (5 Volt), es decir, 1 lógico, obtenemos un divisor por dos. Así, por cada dos pulsos de reloj que ingresan a la entrada de reloj, obtenemos un pulso en cada una de las salidas del **CI 7473**.

Otra aplicación es en registros de desplazamiento (en inglés, *shift register*), que tienen **n** células básicas de memoria, **RS** o **JK**, que almacenan información. Un **registro de desplazamiento** es una memoria de acceso secuencial serie tanto para la lectura como para la escritura.

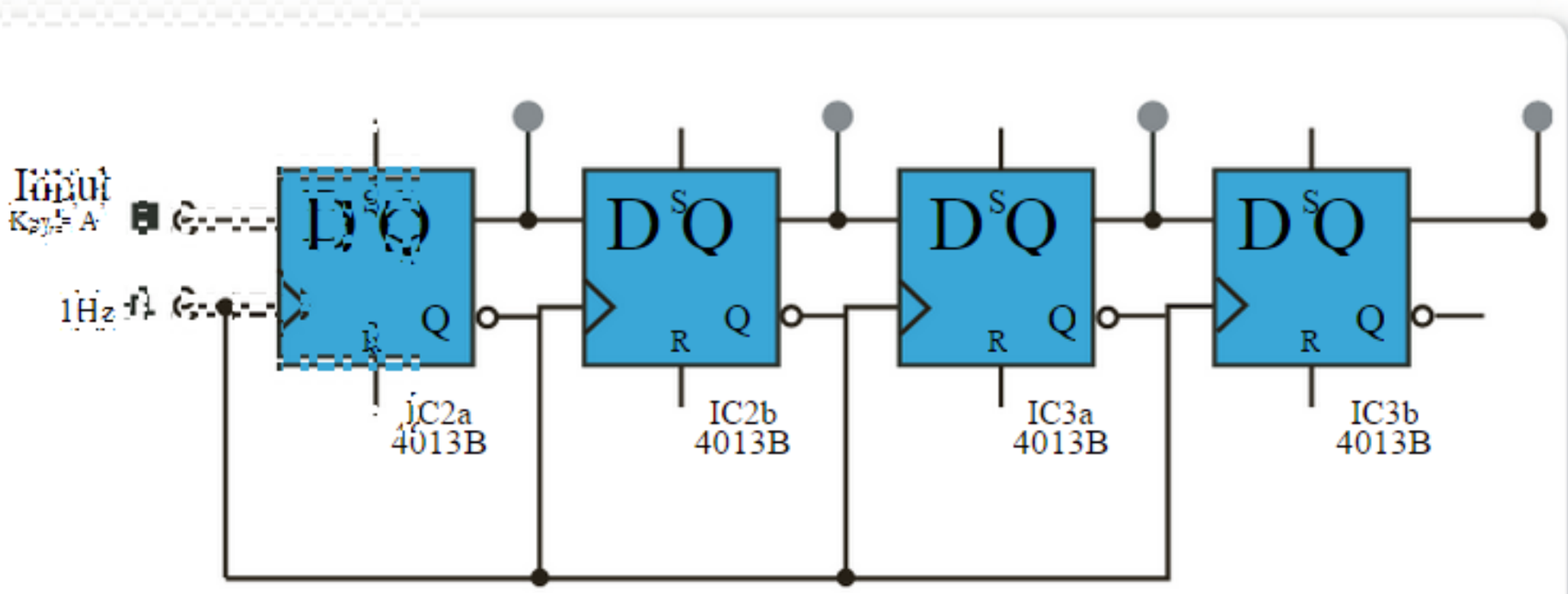


Figura 35. Diagrama de un registro de desplazamiento de cuatro bits mediante flip-flop SR.

Un **contador binario** es una aplicación que utiliza flip-flop JK, RS, D o T y puede contar los pulsos recibidos en una entrada específica. El conteo se muestra en un código binario determinado; agregando un decodificador **BCD** a siete segmentos, se puede leer en un display. Existen dos tipos de contadores: **serie** o asíncronos y **paralelo** o síncronos. En un **contador asíncrono** cada flip-flop activa al siguiente, mientras que en un **contador síncrono** los flip-flop cambian en forma simultánea. Un contador puede ser ascendente si se incrementa el valor contado con la llegada de pulsos, o descendente en el caso contrario. El circuito integrado **7490** contiene cuatro flip flop **JK síncronos** y a la salida entrega una combinación de cuatro bits en código BCD.

> EN LOS CONTADORES SÍNCRONOS LOS FLIP-FLOP CAMBIAN EN FORMA SIMULTÁNEA

Simulación de circuitos sumadores, combinacionales y secuenciales

En un **sumador-restador de dos números de cuatro bits** cada uno, mediante los interruptores, se determinan los valores de los cuatro bits de cada número, y así se llega a que los bits A1-A4 corresponden al primer operando, y los bits B1 a B4, al segundo operando, eventualmente el sustraendo.

El circuito integrado **IC3** (**TTL 7483**) corresponde a un sumador total que recibe al operando 1, mientras que el operando 2, luego de complementarlo a 1 mediante las compuertas **OR-EXC** y sumar 1 (en el caso de restar dos números para obtener su complemento a 2), se envía al segundo sumador total TTL 7483, **IC2**.

PARA RECORDAR

Un **multivibrador** es un secuencial en el que el estado de las salidas depende del estado de las entradas y la información almacenada. Puede construir una memoria de un bit simple con un flip-flop RS. En la tabla de verdad se observa el estado de la salidas del flip-flop JK síncrono en función de la combinación en las entradas. Dos 7473 proporcionan cuatro flip-flop T con un factor de división por 16.

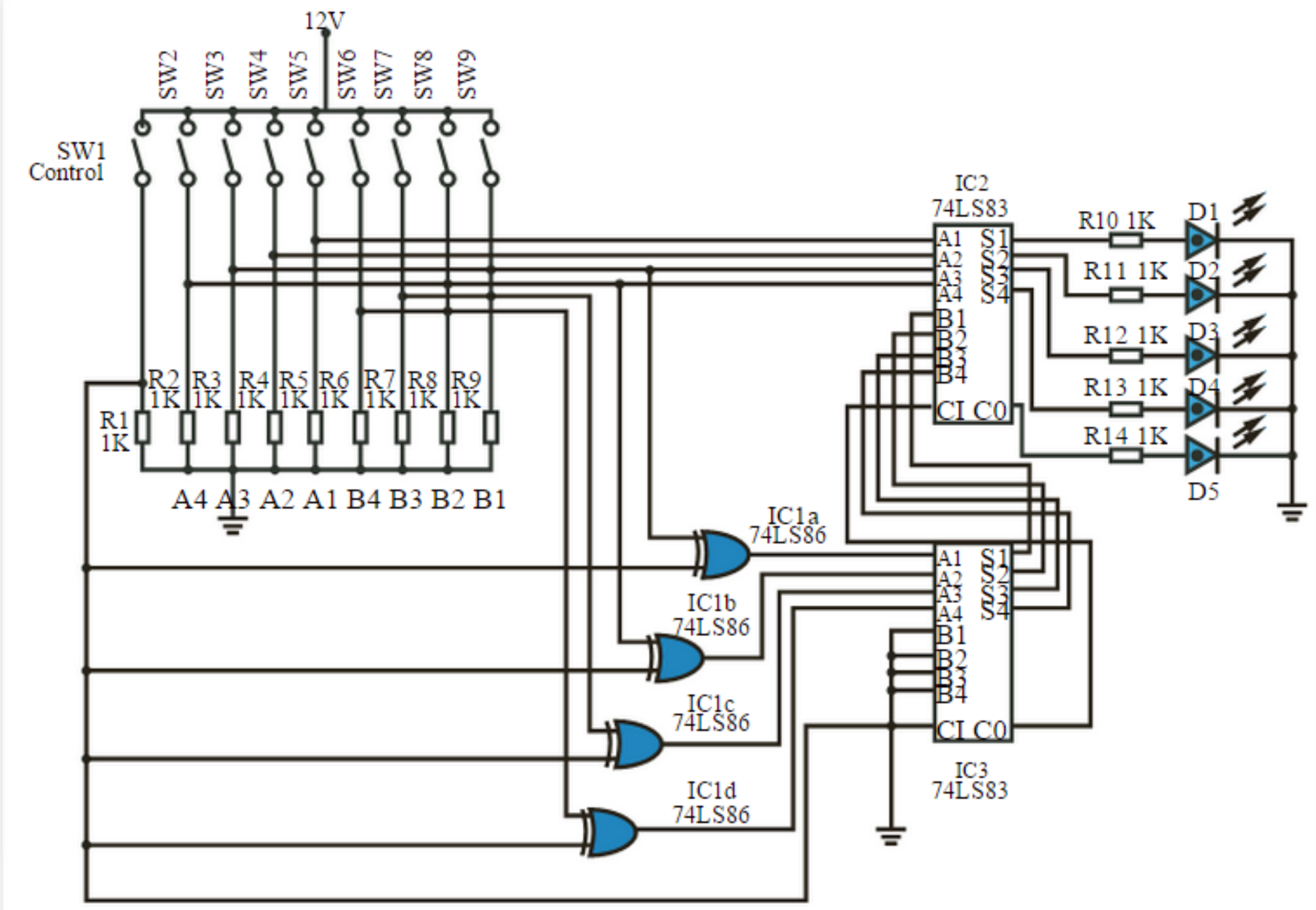


Figura 36. Esquema eléctrico de un sumador-restador de cuatro bits.

En la operación suma, los cuatro bits del operando 2 no experimentan ninguna alteración ya que se les suma 0.

Para seleccionar la operación por realizar, suma o resta binaria (en realidad, se trata de suma por complemento a 2), se utiliza el valor del **bit de control**: 0, para sumar ambos operandos y 1 para restarlos. El resultado de la operación realizada se visualiza mediante 4 leds y un led de acarreo (D5).

Es posible construir cada uno de los siguientes circuitos combinacionales mediante compuertas lógicas combinadas, para obtener circuitos más complejos y demostrar el funcionamiento de los distintos circuitos, presionando cada interruptor de entrada.

La simulación de un sistema secuencial, en este caso un flip-flop (o también, biestable), debería expresar cómo se pueden utilizar dos compuertas **NAND** realimentadas para crear una memoria sencilla. Presionando los pulsadores SW1 y SW2, se obtienen distintas funciones: SW1 setea el biestable, mientras que SW2 lo resetea. Mientras no se resetee el biestable, la salida memoriza el último estado seleccionado.

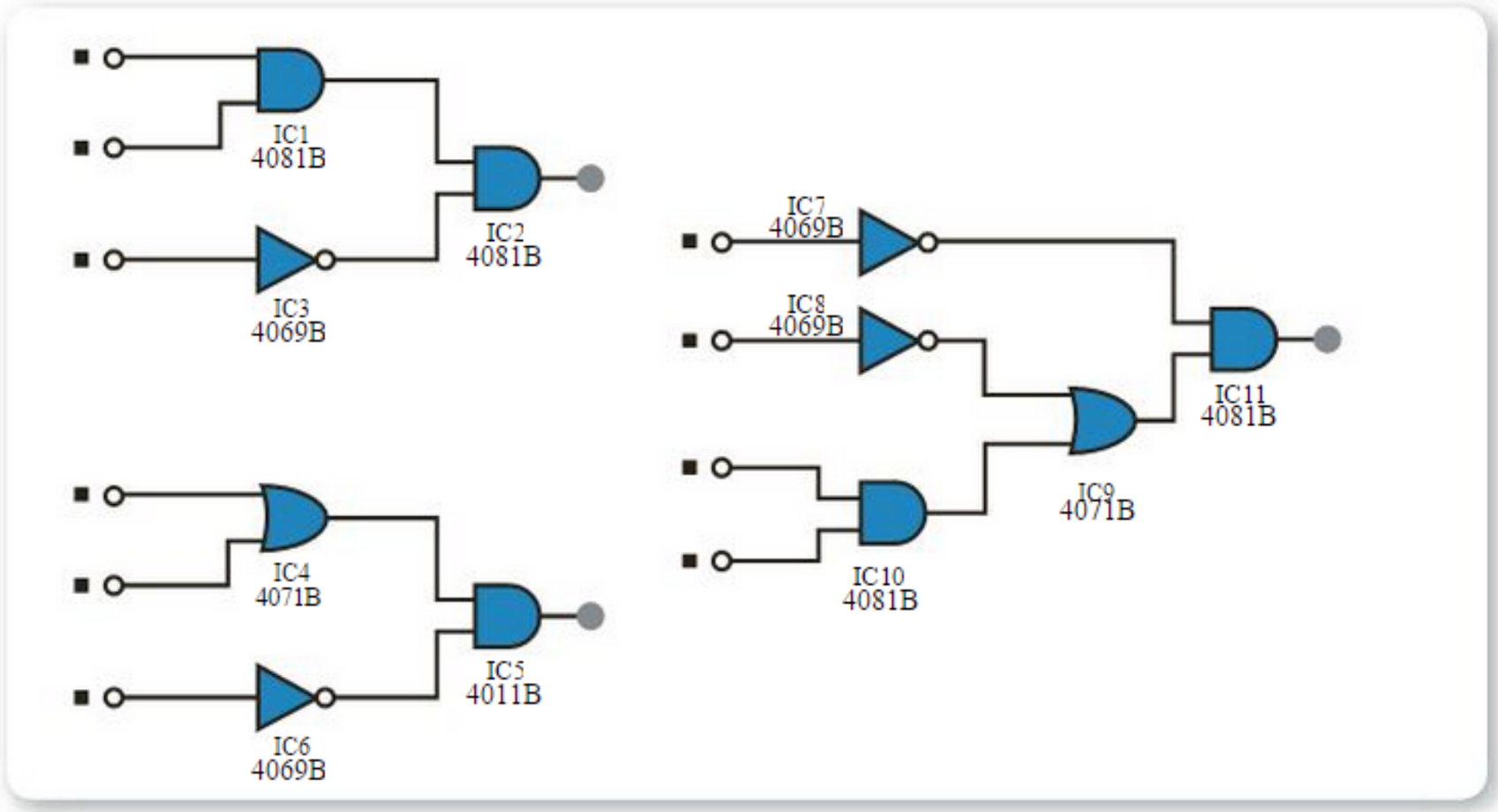


Figura 37. Simulación de distintas compuertas lógicas.

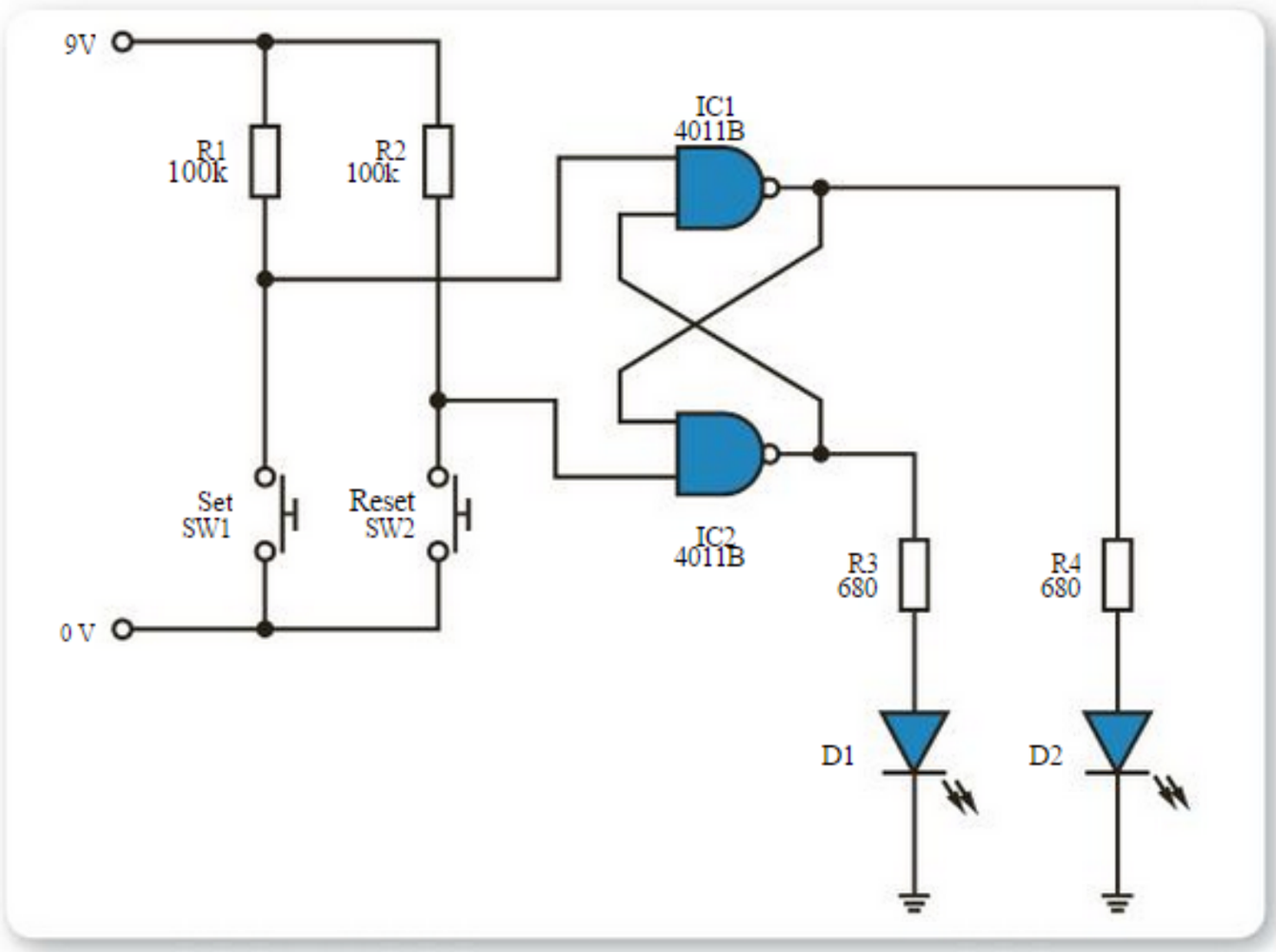


Figura 38. Simulación de una memoria de un bit mediante un flip-flop construido con compuertas NAND.

RESUMEN

En este segundo capítulo abordamos la información general que existe sobre los dispositivos digitales, que permiten implementar las expresiones lógicas básicas, también conocidas como expresiones booleanas. La compuerta lógica es el bloque constructivo más básico en la lógica digital. Además de las tres compuertas lógicas OR, AND y NOT y sus derivadas, se presentaron algunas nociones referidas a la lógica tanto combinacional como secuencial.

Actividades

TEST DE AUTOEVALUACIÓN

1 Considere un flip-flop R-S realizado mediante compuertas NAND. Si S = R = 1:
- Las salidas están en 0. - Las salidas están en 1.
- Las salidas no cambian de estado.

2 En un flip-flop JK se actualiza el valor de las salidas cuando:
- Se produce el flanco ascendente del reloj. - Las salidas están en 1.
- Se produce el flanco descendente del reloj.

3 Considere cuatro variables binarias. A partir de ellas puede obtener:
- Seis combinaciones diferentes. - Ocho combinaciones diferentes.
- Dieciséis combinaciones diferentes.

EJERCICIOS PRÁCTICOS

1 Realice el diagrama de contactos eléctricos para una compuerta OR-EXC.

2 Minimice la función f = Σ3 (0, 3, 5, 7) mediante el método de Karnaugh.

3 Diseñe un circuito semisumador binario mediante compuertas NAND (sin emplear circuitos inversores).

4 Construya un multiplexor de ocho canales utilizando un circuito integrado TTL 74151 y compruebe su funcionamiento.

5 Suponga que una vivienda tiene un garaje al que una persona puede acceder desde dos habitaciones. Realizar el diagrama de contactos eléctricos y la función booleana correspondiente para que esta persona pueda encender o apagar la luz del garaje desde el mismo garaje o desde una habitación, independientemente de la posición de los otros dos interruptores.

6 Implemente la solución al ejercicio anterior mediante compuertas lógicas.

PROFESOR EN LÍNEA

Si tiene alguna consulta técnica relacionada con el contenido, puede contactarse con nuestros expertos: profesor@redusers.com.

*03 –

Dispositivos basados en microprocesadores

Identificaremos los conceptos básicos para comprender cómo funciona un microprocesador, cuáles son las partes que lo constituyen y qué lo diferencia de otros dispositivos digitales de uso masivo, los microcontroladores. Para ello, recurriremos a conceptos comprendidos en los capítulos 1 y 2.

Servicio de atención al lector: usershop@redusers.com

Organización de una computadora basada en microprocesador

Una **computadora** puede definirse como un dispositivo electrónico capaz de efectuar las siguientes funciones: recibir información, almacenarla, procesarla y generar los resultados. Para ello, dispone de cuatro componentes principales: el **procesador**, que gestiona y controla las operaciones realizadas en la computadora; la **memoria**, donde se almacena la información en forma de programas (y los datos necesarios para ejecutarlos); el **sistema de E/S**, que permite transferir los datos entre la computadora y los dispositivos externos, posibilitando a los usuarios introducir información y observar los resultados además de facilitar la comunicación con otras computadoras, y, finalmente, el **sistema de interconexión**, que facilita la interconexión de todos los componentes.

Es importante distinguir entre los conceptos de arquitectura y organización o estructura de una computadora. La **arquitectura** de la computadora se refiere a los elementos visibles desde la perspectiva de un programador en lenguaje ensamblador: juego de instrucciones y modos de direccionamiento, tipos y formatos de los operandos, mapa de memoria y de E/S y modelos de ejecución. La **organización** o **estructura** de la computadora se refiere a las unidades funcionales y a su interconexión: los sistemas de interconexión y de control, la interfaz entre la computadora y los periféricos, y las tecnologías utilizadas; todos elementos transparentes al programador. Así, se tienen computadoras que comparten la misma arquitectura, aunque están organizadas de manera diferente.

ARQUITECTURA X86-64

Es un ejemplo de arquitectura compartida y organización diferente. La arquitectura se comparte tanto en las computadoras basadas en microprocesadores **Intel** (Intel64) como en las que utilizan microprocesadores **AMD** (AMD64), aunque estos procesadores tengan una organización diferente.

En cuanto a los tipos de organización de computadoras, si bien se trata de su estructura, tradicionalmente se ha utilizado el término arquitectura para distinguir entre los dos tipos de organización más populares: arquitectura **Von Neumann** y arquitectura **Harvard**. La mayoría de las computadoras actuales se valen de la arquitectura Von Neumann modificada con características provenientes de la arquitectura Harvard, muy utilizada en **microcontroladores** y **procesadores de señal digital**, **DSP**.

> LA ORGANIZACIÓN O ESTRUCTURA SUPONE LAS UNIDADES FUNCIONALES Y SU INTERCONEXIÓN

La principal diferencia entre ambas arquitecturas radica en el **mapa de memoria**. Mientras que la arquitectura Von Neumann tiene un espacio de memoria único para datos y para instrucciones, la arquitectura Harvard destina dos espacios de memoria separados, uno para los datos y otro para las instrucciones. La arquitectura Von Neumann se utiliza en los sistemas basados en microprocesadores y en las computadoras personales, por ejemplo, ya que permite ahorrar líneas de E/S en los sistemas donde el microprocesador se instala en un zócalo sobre una placa madre, y porque además simplifica el trabajo de diseño y reduce el costo final de estos sistemas.

La ventaja de la arquitectura Harvard radica en que permite adecuar el tamaño de los buses a las características de cada tipo de memoria, y la CPU puede acceder a cada una de ellas de forma simultánea, lo que implica un aumento significativo de la velocidad de procesamiento. La desventaja está en que consume numerosas líneas de E/S de la CPU.

Diagrama general de una computadora basada en microprocesador

En función de los cuatro componentes de una computadora explicitados antes y con la descripción realizada, se puede desarrollar un diagrama general para las computadoras basadas en microprocesador.

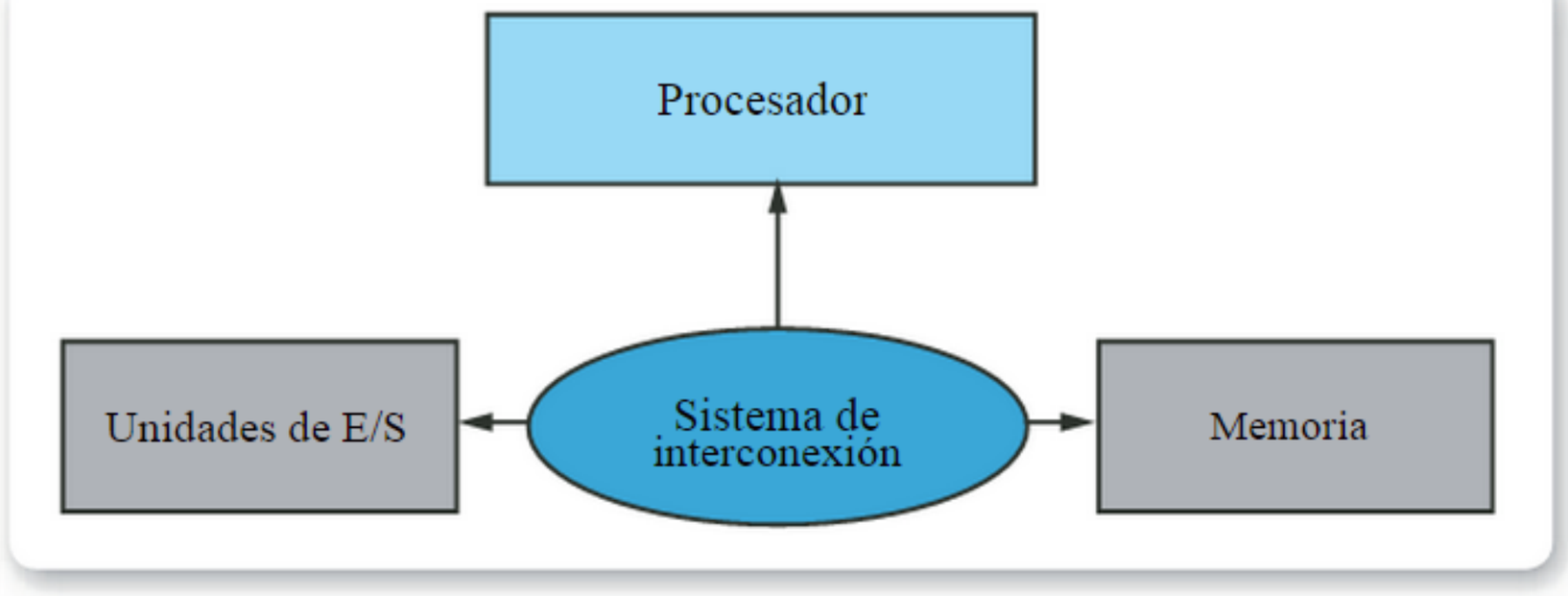


Figura 1. Diagrama general de una computadora basada en microprocesador.

El concepto que subyace en la arquitectura Von Neumann es la construcción de un sistema con la suficiente flexibilidad como para resolver diferentes tipos de problemas. Para solucionar algún problema, se diseña un sistema de propósito general programable mediante un programa específico. En la arquitectura Von Neumann, están presentes todos los elementos que integran una computadora: microprocesador o CPU, memoria, unidades de E/S y un sistema de interconexión.

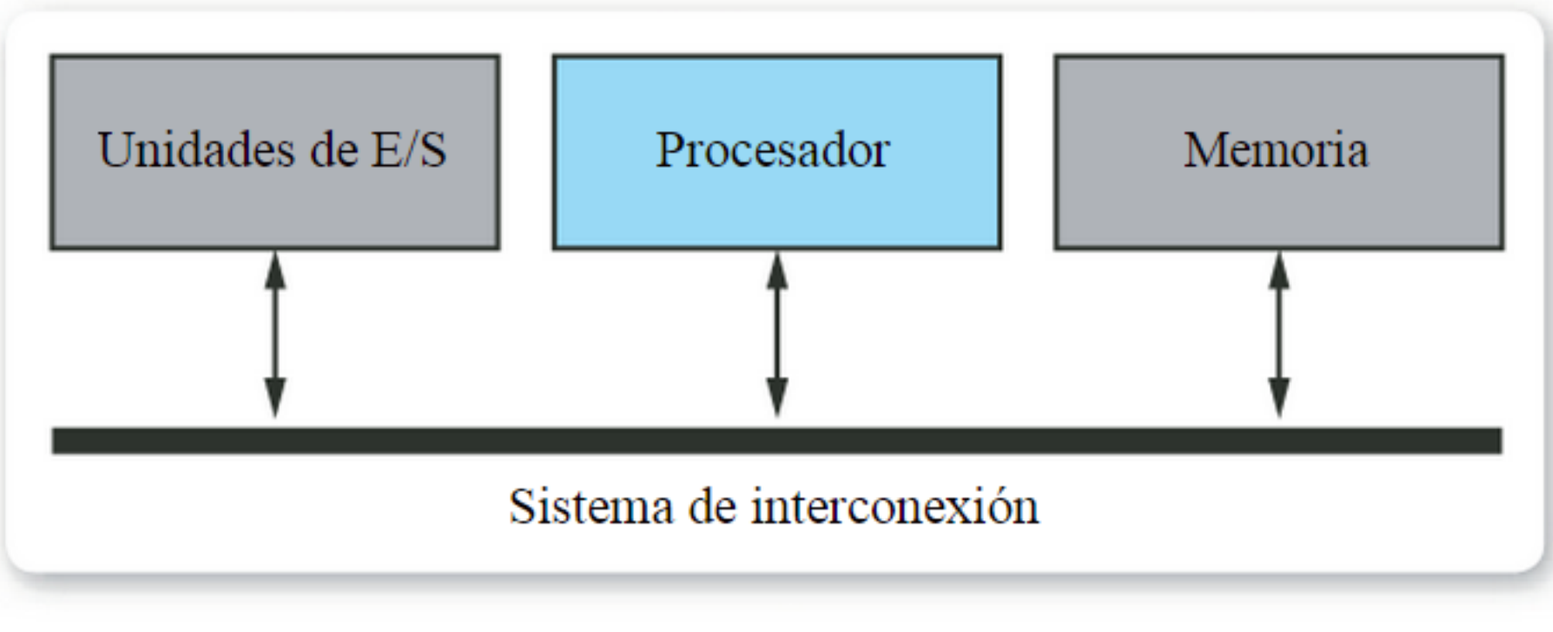


Figura 2. Esquema general de la arquitectura Von Neumann.

En una computadora organizada de acuerdo a la estructura de Von Neumann, se procesa la información de una manera específica por medio de un programa y de un conjunto de datos, ambos almacenados en la memoria principal.

Dado que la arquitectura Von Neumann se utiliza para implementar sistemas basados en microprocesadores, el objeto de este libro, es importante conocer sus propiedades específicas.

En primer lugar, se tiene un espacio de memoria de lectura y escritura único que incluye las instrucciones y los datos necesarios. Además, el contenido de la memoria es accesible por posición, independientemente de que se acceda a datos o a instrucciones, y la ejecución de las instrucciones se produce de manera secuencial: luego de ejecutar una instrucción, se ejecuta la instrucción siguiente almacenada en la memoria principal, aunque se puede romper la secuencia de ejecución mediante instrucciones de **ruptura de secuencia** . De acuerdo a esto último, se propone un sistema general basado en microprocesador, buses y memorias.

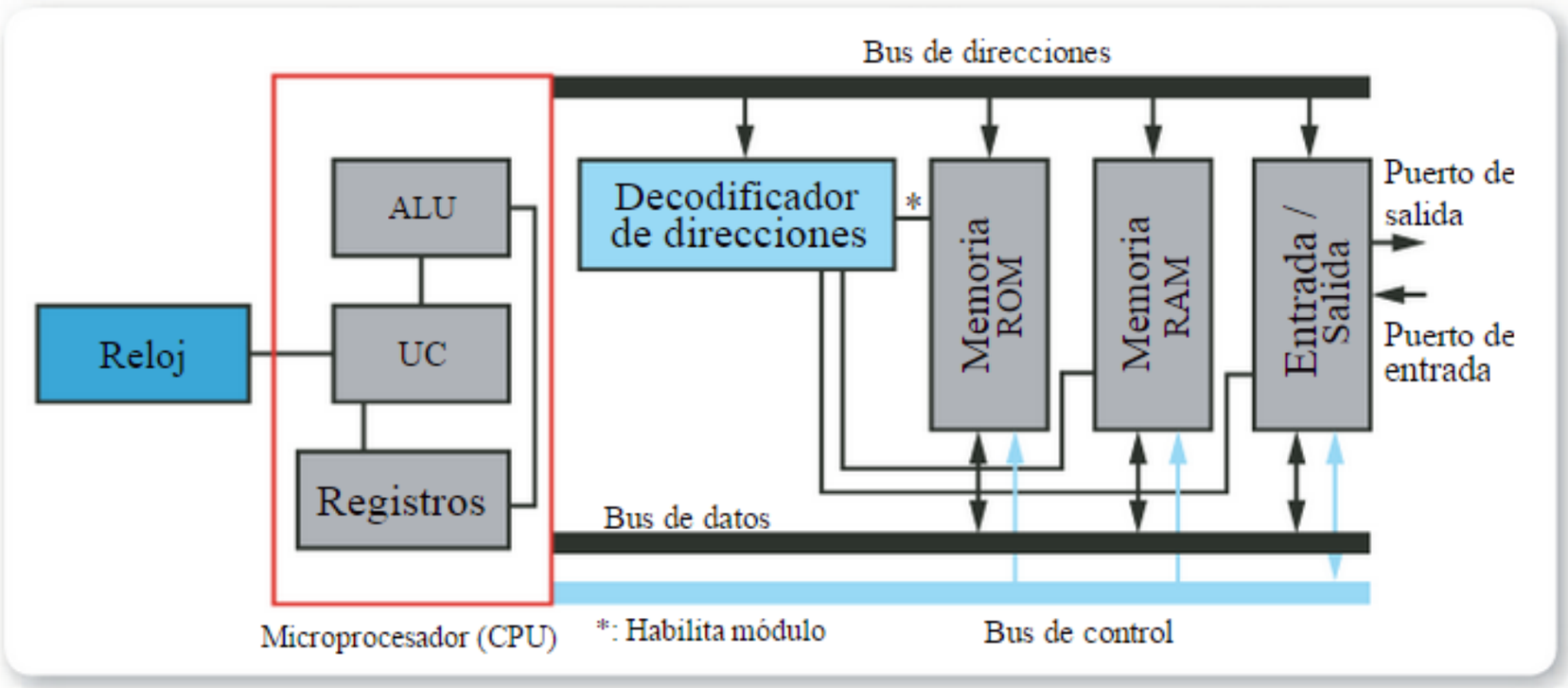


Figura 3. Esquema básico de un sistema basado en microprocesador, y los buses y las memorias que lo integran.

Con cierta frecuencia se hace referencia al microprocesador como la unidad central de procesamiento (CPU), aunque estrictamente la CPU es la parte del microprocesador cuya función consiste en reconocer y ejecutar las instrucciones de un programa. Las interfaces de entrada/salida se emplean para comunicarse con el exterior y la memoria donde se almacenan instrucciones de programas y datos.

PROGRAMAS E INSTRUCCIONES

Un programa consta de instrucciones simples o instrucciones máquina, que, básicamente, son las siguientes: **aritméticas** (suma, resta, multiplicación, división), **lógicas** (AND, OR, XOR, NOT), **transferencia de datos** (mover un dato de una localización a otra) y **ruptura de secuencia** (salto incondicional, salto condicional, etcétera).

Los microprocesadores han evolucionado y pasaron de ser un circuito integrado con un solo núcleo a un circuito integrado conocido como **microprocesador multinúcleos** . Estos últimos incluyen varios núcleos; cada uno de ellos contiene todas las funcionalidades de un microprocesador, es decir, ALU, unidad de control y registros.

En 1971, Intel lanzó al mercado el primer microprocesador del mundo, destinado a utilizarse en calculadoras electrónicas. El 4004 tenía 4 bits, podía direccionar 4096 posiciones de memoria de 4 bits de ancho, disponía de un conjunto de 45 instrucciones y fue ampliamente utilizado en los primeros videojuegos y sistemas de control.

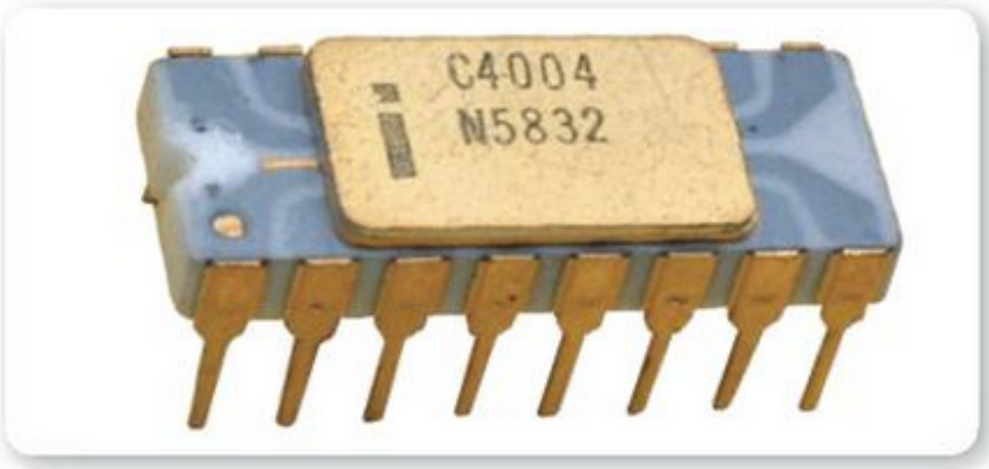


Figura 4. Microprocesador **Intel 4004** de 4 bits del año 1971.

Luego del gran éxito comercial y como respuesta a mayores requerimientos para aplicaciones más complejas, Intel lanzó al mercado el primer microprocesador de 8 bits, el 8008. Además de su mayor velocidad, mayor cantidad de instrucciones, mayor capacidad de direccionamiento y mayor velocidad de procesamiento, se caracterizaba por direccionar 16 Kb, una memoria basada en 8 bits y un set de 48 instrucciones. El 8085, aparecido en 1977, fue el último microprocesador de 8 bits.

En 1978, Intel lanzó al mercado el primer microprocesador de 16 bits, el 8086 y, luego, el 8088. Ambos contaban con registros internos de 16 bits, un bus de datos externos de 16 y 8 bits, respectivamente, y eran capaces de direccionar 1 Mb de memoria con un bus de direcciones de

SISTEMAS EMBEBIDOS

Un **microcontrolador** es un sistema que contiene dentro de un mismo chip el microprocesador, la memoria, y los dispositivos de entrada y salida. Un **microprocesador embebido** es un sistema microprocesador que se utiliza en procedimientos de control de una función específica. Se inicia sin intervención humana, y los programas que requiere están autocontenidos.

20 líneas, además de poder realizar la multiplicación y la división por hardware. El 80286, capaz de direccionar 16 Mb de memoria, fue el último microprocesador Intel de 16 bits.

La generación Intel de 32 bits se inició con el 80386, que direccionaba hasta 4 Gb de memoria, velocidades de operación más altas, conjuntos de instrucciones más grandes y además contaba con memoria interna (caché) de 8 Kb en las versiones más básicas, y del cual surgieron diferentes opciones.

En la década de 1980, comenzó la ruptura entre la evolución tecnológica de los microprocesadores y la de los microcontroladores, dado que los primeros tienen más y mejores capacidades para las aplicaciones en las que se requiere el manejo de grandes volúmenes de información, mientras que los segundos han incorporado más capacidades que les permiten la interacción con el mundo físico en tiempo real, además de mejores desempeños en ambientes de tipo industrial.

Unidad central de procesamiento (CPU)

Ya mencionamos que la CPU es la parte del microprocesador en donde se procesan los datos, y a la que llegan instrucciones y datos. Si bien la estructura interna de un microprocesador, o **arquitectura**, depende del microprocesador considerado, es posible analizarlo mediante una estructura interna simplificada, que se denomina **arquitectura general** de un microprocesador. Allí se observa que el microprocesador o CPU en realidad consta de la unidad lógica y aritmética (**ALU**), la unidad de control (**UC**) y los registros.

Al igual que en un microprocesador comercial, en este microprocesador genérico se ejecutan funciones de unidad lógica y aritmética (ALU), registros y unidad de control. En la ALU se realizan operaciones

OPERACIONES EN UNA ALU

En una ALU se realizan **operaciones aritméticas** y **operaciones lógicas**. Entre las primeras se incluyen suma, resta, multiplicación y división; además de operaciones específicas de incremento positivo (+1) o negativo (–1). Las operaciones lógicas incluyen operaciones AND, OR, NOT, XOR, operaciones de desplazamiento de bits a la izquierda y a la derecha, y operaciones de rotación de bits.

aritméticas y lógicas con números enteros y números reales tanto en punto fijo como en punto flotante. La UC es determinante en cuanto a la temporización y secuencia de las operaciones que se realizan, ya que generan las señales de temporización necesarias para leer una instrucción de programa en la memoria. Cada operación se caracteriza por la cantidad de ciclos de reloj que necesita para ser ejecutada.

Como la ALU es responsable de la gestión de los datos, los **registros internos** al microprocesador se emplean como almacenamiento temporal de los datos internos que la CPU utiliza mientras ejecuta las instrucciones, dado que solo es posible leer una dirección de memoria a la vez. Entonces, un registro es un elemento de memoria de acceso rápido ubicado dentro del microprocesador. Cada microprocesador comercial dispone de un conjunto de registros que lo caracterizan y que cumplen una determinada función. Así, se dispone de diversos registros en cantidad, dimensión y tipo.

REGISTROS Y FUNCIONES EN UNMICROPROCESADOR

▼ REGISTRO	▼ ABREVIATURA	▼ DESCRIPCIÓN
Acumulador	**A, Acc**	Almacenar temporalmente los resultados de la ALU y participar en todas las transferencias de datos asociadas con la ejecución de las operaciones aritméticas y lógicas.
Estado (o de código de condición o registro de banderas)		Almacenar información relacionada con el resultado de la última operación ejecutada en la ALU mediante bits individuales usados como banderas y que tienen un significado especial.
Contador de programa (o apuntador de instrucciones IP, eninglés, **instruction pointer**)	**PC** (en ingl{es, **program counter**)	Utilizado por la CPU para controlar la posición en que se encuentra dentro de un programa.
Direccionamiento de memoria	**MAR** (en inglés, **memory address register**)	Contiene la dirección de memoria donde se almacenan los datos.
Instrucciones	**IR** (en inglés, **instruction register**)	Almacena instrucciones.

▼ REGISTRO	▼ ABREVIATURA	▼ DESCRIPCIÓN
Propósito general		Permiten almacenar temporalmente datos o direcciones. Se utilizan en operaciones que involucren transferencias entre varios registros.
Apuntador de la pila	**SP** (en inglés, **stack pointer register**)	Su contenido almacena una dirección especial que define el inicio de un área especial de la memoria RAM llamada pila, donde se almacenan los valores del contador de programa cuando se ejecuta una subrutina.

Tabla 1. Registros más comunes en un microprocesador y su función.

Los registros se clasifican en: registros de propósito general, de instrucción, de acceso a memoria, y de estado y control. El contenido de los **registros de propósito general** se emplea como operandos en las instrucciones en lenguaje ensamblador. Pueden almacenar datos o direcciones de memoria.

Los **registros de instrucción** se relacionan con el acceso a las instrucciones; los dos principales son: el **contador de programa** (**PC**) y el **registro de instrucción** (**IR**). El **contador de programa** contiene la dirección de la siguiente instrucción por ejecutar en el programa. Cada vez que una instrucción se ejecuta, el valor del PC se incrementa, de modo que siempre contiene la dirección de memoria en la que se almacena la siguiente instrucción por ejecutar. Esta ejecución secuencial se puede interrumpir mediante instrucciones específicas como **salto** (en inglés, *kump*) o **ramificación** (en inglés, *branch*).

> LA CPU ES LA PARTE DEL MICROPROCESADOR DONDE SE PROCESAN LOS DATOS

El **registro de instrucciones** contiene la instrucción por ejecutar.

Los **registros de acceso a memoria** se relacionan con las operaciones de lectura o escritura de la memoria. Los dos más importantes son: el **registro de direcciones de memoria** (en inglés, *Memory Address Register* , **MAR**), en el que se almacena la dirección de la memoria a la que se desea acceder, y el **registro de datos de memoria** (en inglés, *Memory Buffer Register* , **MBR**).

Otro registro, el **acumulador**, almacena temporalmente los resultados aritméticos y lógicos intermedios que después serán tratados por la ALU.

Esta proporciona la dirección de memoria donde se encuentra el dato requerido. Luego, la ALU utiliza los datos/instrucciones requeridos por el bus de datos (por ejemplo, si se suma un número A y otro número B, el número A se trae desde su dirección en memoria y se almacena temporalmente en el acumulador. Mientras tanto, la CPU trae el número B desde otra dirección de memoria. Ahora, la ALU puede operar con ambos números, y el resultado de la suma se almacena en el acumulador).

BANDERAS

BANDERA	AJUSTE (1)	RESTABLECIMIENTO (0)
Z	Si el resultado = 0.	Si el resultado no es 0.
N	Si el resultado es (-).	Si el resultado no es (-).
C	Si existe acarreo.	Si no existe acarreo.
V	Si existe desbordamiento.	Si no existe desbordamiento.
I	Se ignora la interrupción.	Interrupción procesada normalmente.

Tabla 2. Banderas más comunes y circunstancias en las que toman el valor 0 o el valor 1.

El **registro de estado** es un registro de memoria donde se almacena el estado del procesador, y se indica mediante bits o **banderas**, que son modificados por el microprocesador como resultado de la ejecución de instrucciones aritméticas o lógicas, o por acciones como pedidos de interrupción. Las banderas más comunes son: Z, N, C, V, I. Cada bandera se puede ajustar (1) o restablecer (0). Por ejemplo, al sumar los números hexadecimales 1 y 5, el resultado es 8. Entonces, el valor que toman las banderas es Z = 0, N = 0, C = 0 y V = 0.

En los **registros de control** se almacena la información generada por la unidad de control además de información específica del sistema operativo.

En el **registro de direccionamiento de memoria**, en la suma de un número A y otro número B, la dirección del número A se encuentra en el registro de direccionamiento de memoria. El número A almacenado en esa dirección se transfiere al acumulador. Luego, se almacena el número B en el registro de direccionamiento de memoria. El número B almacenado en esa dirección se transfiere al acumulador para sumarlo al número A. El resultado de la suma se almacena en una dirección de memoria que llama el registro de direccionamiento de memoria.

En el **registro de direcciones** se desarrolla el ciclo conocido como **traer y ejecutar**, en el que, desde la memoria, la CPU trae, por el bus de datos, una instrucción y la almacena en el registro de instrucciones. A continuación, el microprocesador incrementa en uno al contador de programa, que ahora apunta a la dirección de memoria donde se almacena la próxima instrucción por traer. De esta manera, la instrucción traída en el inicio del ciclo se puede decodificar para ejecutar la operación que corresponda.

Circuito de reloj

Los microprocesadores, como dispositivos secuenciales, operan utilizando una señal de sincronización como **señal de reloj**, que es una señal en forma de onda cuadrada periódica con una determinada frecuencia. Sin esta señal, el microprocesador no podría operar.

Todas las operaciones que realiza el microprocesador son gobernadas por la señal de reloj, y un **ciclo de reloj** determina el parámetro conocido como **unidad básica de tiempo**, esto es, la duración mínima de una operación del procesador.

Así, para ejecutar una instrucción, se necesitan uno o más ciclos de reloj, de acuerdo al tipo de instrucción y a los operandos que tenga. Existen microprocesadores que requieren un circuito externo para generar la señal de reloj, como el Zilog Z80 de 8 bits, y otros que solo requieren un cristal de cuarzo externo dado que, en su interior, poseen la lógica para generar esas señales, como los utilizados en las computadoras personales.

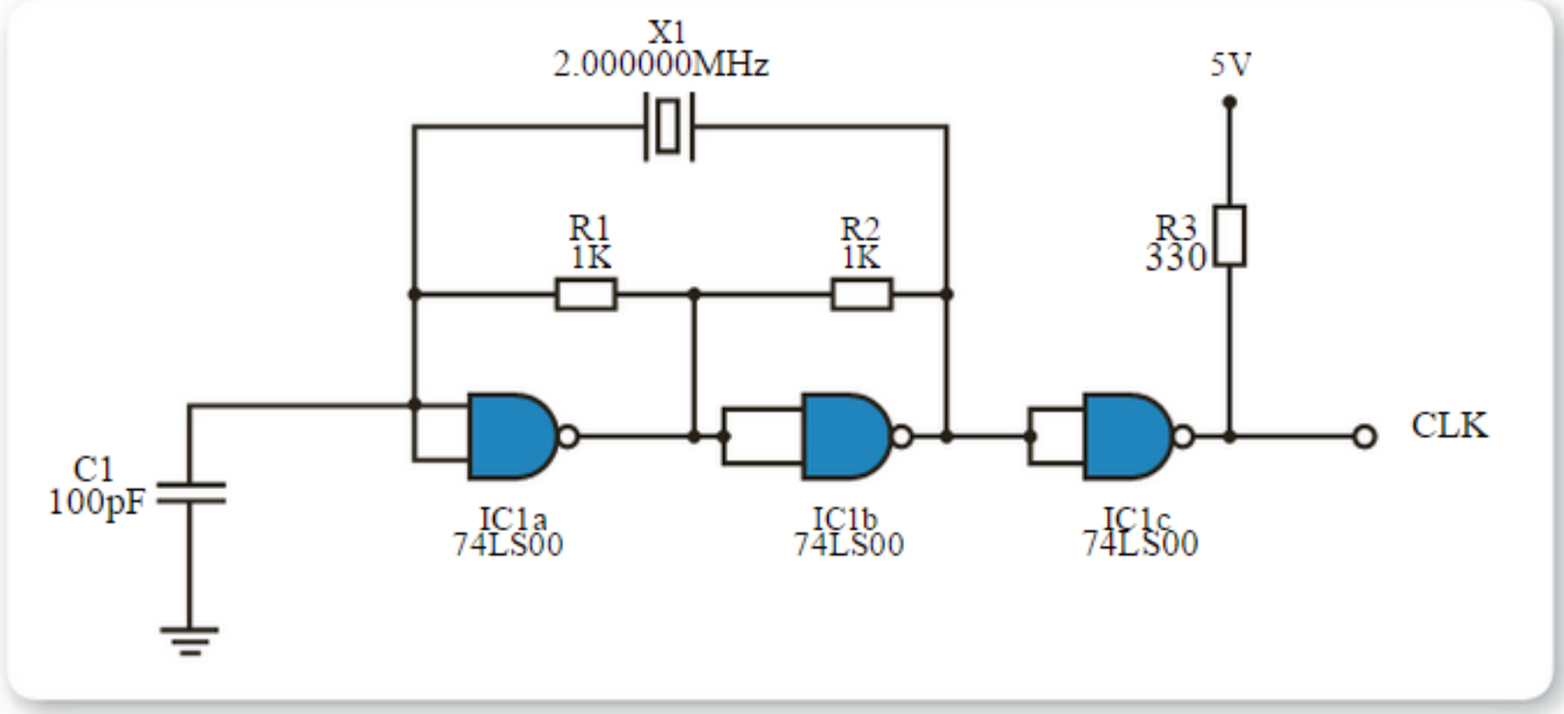


Figura 6. Circuito de reloj externo con cristal de cuarzo para un microprocesador **Zilog Z80** de 8 bits.

El Z80 requiere un circuito generador de reloj externo que se implementa por una de tres formas diferentes atendiendo a la precisión de la señal de reloj y el costo de la solución: mediante un oscilador discreto con **cristal de cuarzo**, mediante un circuito **oscilador RC** (resistencia-condensador) o por medio de un **oscilador integrado con cristal compatible con TTL**. A frecuencias cercanas al valor máximo provisto por el fabricante o cuando se desea mayor precisión, se recomienda un oscilador controlado por cristal, en el que la frecuencia de oscilación del cristal determina la frecuencia de la señal del reloj. La estabilidad del cristal mantiene constante el tiempo de ejecución.

Una opción de menor precisión y bajo costo la constituye el **circuito RC**. Finalmente, para simplificar el circuito y reducir el consumo de energía eléctrica, se puede usar un oscilador integrado con cristal compatible con TTL.

Al energizar el microprocesador por primera vez, y debido a que tiene registros y flip-flop internos que pueden adoptar valores o estados aleatorios, la operación del microprocesador puede tornarse impredecible.

Además, el Z80 requiere una señal de **reset** que obligue al microprocesador a alcanzar un estado inicial conocido.

Memoria

La función principal de la memoria en un microprocesador es almacenar datos binarios con la forma de instrucciones de programa o de números necesarios para realizar distintas operaciones.

Una memoria está conformada por grandes cantidades de **celdas de 1 bit** con capacidad para almacenar un 0 o un 1. Estas celdas de 1 bit se agrupan de modo que puedan almacenar una palabra y, para acceder a estas, se identifica cada posición mediante una dirección única.

FRECUENCIA DE LA SEÑAL DE RELOJ

Se la define como el número de impulsos por unidad de tiempo medida en ciclos por segundo o hertz (Hz) y determina la velocidad de operación del procesador. En los microprocesadores actuales, se utilizan prefijos como giga (GHz), para miles de millones de hertz, y mega (MHz), para millones de hertz. Por ejemplo, en una frecuencia de reloj de 1 GHz, se tiene un período de reloj de un nanosegundo.

Así, un bus de direcciones de 4 bits posibilita identificar hasta 16 direcciones diferentes ($2^4 = 16$), cada una capaz de almacenar una palabra, que comienzan en 0000 y finalizan en 1111. La cantidad de **posiciones de memoria** disponibles se mide en K, donde 1 K es $\mathbf{2^{10}} = \mathbf{1024}$ posiciones. Entonces, una memoria de 4 K dispone de 4096 posiciones (4 * 1024) capaces de almacenar, por ejemplo, 1 byte (grupo de 8 bits) en cada una de ellas.

> LA MEMORIA ALMACENA DATOS BINARIOS EN FORMA DE INSTRUCCIONES DE PROGRAMA

Físicamente, una memoria se construye en uno o varios circuitos integrados especializados.

La clasificación de las memorias se basa en si solamente permiten la lectura o si se puede leer/escribir en ellas. Las **memorias solo de lectura** (en inglés, *Read Only Memory* o **ROM**) almacenan los datos en forma permanente. Estos datos se les incorporan durante el proceso de fabricación del circuito integrado y, luego de esto, solo es posible leer los datos almacenados en ella. Se las utiliza principalmente en situaciones en las que se requieran programas fijos, como en el sistema de arranque de una computadora y en programas para aplicaciones. Una característica fundamental es que mantienen su contenido, aunque se desconecte la alimentación eléctrica.

Las memorias **ROM programables** o **PROM** son memorias ROM que pueden ser programadas por el mismo usuario. Para ello, se hace pasar una corriente por un fusible que funciona como eslabón en cada celda de memoria. De esta forma, interrumpe de manera permanente la conexión eléctrica y pasa el contenido de la celda desde 0 (fusible cerrado) a 1 (fusible abierto). Así se almacenan definitivamente los datos en la memoria, y ya no se la puede modificar.

Una memoria **ROM borrable y programable** o **EPROM** se puede programar y modificar. Para ello se incorporan en un circuito integrado pequeñas celdas electrónicas que almacenan una carga. Aplicando, o no, una tensión eléctrica a las celdas, se configuran celdas cargadas (1) y celdas no cargadas (0), respectivamente, una situación que permanece hasta que se hace incidir una luz ultravioleta por una ventana de cuarzo en la parte superior del circuito integrado y que descarga todas las celdas (0). Así es posible programar nuevamente el circuito integrado.

Por último, la memoria **ROM borrable eléctricamente** o **EEPROM** es similar a la memoria EPROM, aunque el borrado se realiza aplicando una tensión eléctrica bastante alta en lugar de utilizar radiación ultravioleta.

Las **memorias de acceso aleatorio** (en inglés, *Random Access Memory* o **RAM**), por el contrario, son memorias temporales para guardar datos que se pueden leer o escribir. Se denominan de **acceso aleatorio**, porque se puede leer o escribir en una posición de memoria con un **tiempo de espera** igual para cualquier posición, sin que sea necesario seguir un orden para acceder a la información de la manera más rápida posible.

Las memorias **RAM dinámicas** (en inglés, *Dinamic RAM* o **DRAM**) y las memorias **RAM estáticas** (en inglés, *Static RAM* o **SDRAM**) son dos tipos de memoria RAM, que se diferencian por la tecnología que utilizan para almacenar los datos. La memoria DRAM tiene que actualizarse miles de veces por segundo, **ciclo de refresco**, no así la memoria SRAM, que es más rápida, tiene menor consumo y también resulta más costosa. Por lo tanto, cuando sea necesario disponer de menor tiempo de acceso, de un consumo reducido, o de ambos, se utiliza una SDRAM.

Por su compleja estructura interna, una memoria SRAM es menos densa que DRAM, y por ello no se emplea si se necesita alta capacidad de datos, como por ejemplo en la memoria principal de las computadoras personales.

Respecto de las memorias DRAM, la principal ventaja es su utilidad para construir memorias con gran densidad de posiciones que funcionan con altas velocidades, por lo que se fabrican circuitos integrados con millones de posiciones y velocidades de acceso medidos en millones de bits por segundo.

Ambos tipos de memoria RAM son **volátiles** y pierden su contenido cuando se desconecta la energía del sistema.

FIRMWARE Y SOFTWARE

Una vez que el programa se almacenó en una memoria ROM, se encuentra disponible para ser utilizado en cuanto se activa el sistema. Estos programas se conocen como **firmware** o microprogramas. Los programas almacenados en una memoria RAM se denominan **software** y, cuando el sistema se activa, el software se carga en la RAM desde un dispositivo periférico.

La **jerarquía de memorias** es una mejora en las computadoras que han incorporado **memoria caché**, más pequeña y rápida que la memoria principal, aunque más costosa. Para que cumpla con su función, se coloca como memoria intermedia entre la memoria principal y el microprocesador, de modo que, cuando el microprocesador necesite un dato/instrucción, se verifica primero si se encuentra en la memoria caché. Solo en caso contrario, se trae de la memoria principal para acceder a él.

> LA MEMORIA CACHÉ PUEDE DIVIDIRSE EN CACHÉ DE INSTRUCCIONES Y CACHÉ DE DATOS

Actualmente se utilizan, al menos, tres niveles de memoria caché denominados **L1**, **L2** y **L3**, aunque se pueden integrar en el microprocesador algunos niveles o todos juntos.

La memoria caché puede dividirse en **caché de instrucciones** y **caché de datos**. Los diferentes tipos de caché se organizan por niveles conformando una jerarquía. En general, a mayor cercanía a la CPU, corresponde mayor velocidad de acceso y menor capacidad de almacenamiento.

El nivel 1 (L1) o **caché interno** es el nivel más cercano a la CPU, ya que está en el mismo núcleo con lo que el acceso se produce a la velocidad de trabajo del microprocesador, la máxima velocidad. Presenta un tamaño muy reducido: en Intel, 4 a 32 KB; en VIA/Cyrix, 1 a 64 KB; en AMD, 8 a 128 KB. El nivel 2 (L2) o **caché externo**, inicialmente se instalaba en la placa base en el exterior de la CPU, aunque a partir de los procesadores Pentium 4 se han incorporado en el procesador, pero no precisamente en el núcleo.

Este nivel apareció con el procesador Pentium Pro y es una memoria más lenta que L1, aunque con mayor capacidad. Los tamaños típicos de la memoria caché L2 oscilan en la actualidad entre 256 KB y 4 MB.

REPRESENTACIÓN DE INFORMACIÓN ENTERA

Un sistema basado en microprocesador trabaja tanto con números enteros como con números reales. Los primeros se representan mediante notaciones **signo magnitud**, **complemento a 1** y **complemento a 2**; esta última es la más habitual en las computadoras. Todas las notaciones representan los números enteros en binario mediante bits. Las computadoras actuales son de 32 y 64 bits.

El nivel 3 (L3) se encuentra en algunas placas base, procesadores y tarjetas de interfaz. El microprocesador de Intel Itanium trae incorporado en su cartucho el nivel L3, que soporta un tamaño de hasta 4 MB, y el Itanium 2, que tolera hasta 6 MB de caché L3. Finalmente, el nivel 4 (L4) se encuentra ubicado en los periféricos y en algunos microprocesadores como el Itanium.

Buses

Los **buses** son los elementos que transportan señales eléctricas digitales, **0** y **1**, que se comparten entre el resto de los circuitos integrados que conforman el sistema. La comunicación es en paralelo, es decir, cada una de las **4**, **8**, **16**, **32** o **64** líneas o conexiones en paralelo que tiene un bus puede llevar 1 bit de una palabra de datos simultáneamente.

Tres son los tipos de buses en un sistema basado en microprocesador: de datos, de direcciones y de control. El **bus de datos** tiene por misión soportar los datos asociados a las funciones de procesamiento de la CPU y es **bidireccional**, es decir, transporta palabras de datos desde y hacia la CPU y la memoria o las interfaces de entrada/salida. Cada línea de datos transporta 0 o 1.

Por ejemplo, dado un bus de 8 bits y la palabra 11110000, cada línea del bus transporta 1 bit de la palabra comenzando con el bit menos significativo en la primera conexión del bus, 0 en este caso, hasta llegar al primer 1 desde la izquierda para ser transportado por la octava conexión del bus. Con un bus de 8 bits se pueden transportar hasta 2^8 combinaciones diferentes. Si bien los microprocesadores de 8 bits, como Motorola 6800, Intel 8085A y Zilog Z80, aún se emplean en sistemas de control, en las computadoras se utilizan microprocesadores con longitudes de palabras de 32 y 64 bits.

REPRESENTACIÓN DE INFORMACIÓN REAL

Los números reales se representan en **punto fijo** o en **punto flotante**. En punto fijo, la posición de la coma binaria es fija y utiliza un número concreto de bits para la parte entera y para la parte decimal, mientras que punto flotante se representa utilizando tres campos: signo, mantisa y exponente. El valor del número es: ± **mantisa** · **2** $^{\text{exponente}}$.

El **bus de direcciones** transporta señales que permiten identificar en qué posición de memoria se encuentran los datos/instrucciones, o selecciona alguno de los puertos de entrada o salida. Una ubicación de memoria se identifica mediante su dirección de memoria. Una vez seleccionada la dirección y colocada en el bus de direcciones, esta dirección será la única por utilizar para interactuar con la comunicación enviada desde la CPU. Un microprocesador con un bus de datos de 8 bits tiene un bus de direcciones de 16 bits que permite direccionar hasta 2^{16} direcciones o 65.536, 64 K donde 1 K = 1024.

En síntesis, mientras mayor tamaño tenga la memoria, mayor será la cantidad de direcciones y el tamaño del bus de direcciones así como la complejidad del software que se utilizará.

Finalmente, el **bus de control** transporta señales relacionadas con acciones de control, por ejemplo, para que el microprocesador informe a las memorias si se encuentra leyendo (**READ**) datos provenientes de un dispositivo de entrada o si está escribiendo (**WRITE**) datos a un dispositivo de salida. Otra función importante de este bus es transportar señales periódicas provenientes del oscilador controlado por cristal de cuarzo o **reloj** por todo el sistema, de modo de sincronizar las acciones realizadas.

Interfaces de entrada/salida

La acción de transferencia de datos entre el microprocesador y los dispositivos periféricos en el mundo exterior se conoce como **entrada/salida** (**E/S**). Como las características y velocidades de los periféricos pueden ser muy distintas a la de un microprocesador, se interconectan por medio de dispositivos específicos o **interfaces**. Entre las funciones más importantes de una interfaz, podemos encontrar la de sincronizar la

REPRESENTACIÓN DE NÚMEROS BINARIOS

Los formatos para representar números binarios en punto flotante son **precisión simple** (32 bits: 1 bit de signo, 8 bits para el exponente y 23 para la mantisa), **doble precisión** (64 bits: 1 bit de signo, 11 bits para el exponente y 52 para la mantisa) o **cuádruple precisión** (128 bits, 1 bit de signo, 15 bits para el exponente y 112 para la mantisa).

> EL ANCHO DE BUS DETERMINA LA CANTIDAD DE BITS QUE PUEDEN TRANSMITIRSE SIMULTÁNEAMENTE

transferencia de datos entre el microprocesador y el dispositivo periférico en uso. Si la operación es de entrada, entonces el dispositivo de entrada coloca los datos en el registro de datos de la interfaz donde permanecen hasta que los lea el microprocesador.

El microprocesador verifica la integridad de los datos que le aporta la interfaz mediante un **muestreo** por el cual la interfaz utiliza un 1 como bit de estado para indicarle al microprocesador que los datos son válidos.

Esta es precisamente la dificultad del método: el microprocesador debe continuar verificando hasta encontrar el bit de estado en 1.

Otra forma es mediante **interrupción**, ya que la interfaz le envía los datos al microprocesador cuando estos son válidos, por lo que se suspende la ejecución del programa principal para ejecutar la rutina asociada con la interrupción y así leer los datos.

Las E/S están asociadas con sistemas de interconexión tanto internos como externos. Los sistemas de interconexión internos utilizan interrupciones como técnica de acceso a los periféricos, de modo que el microprocesador no tenga que esperar que un dispositivo esté libre para realizar la transferencia. También usan acceso directo al medio (en inglés, *direct medium access* o **DMA**) para transferir bloques de datos entre el periférico y la memoria sin la intervención del microprocesador, mediante controladores de DMA que se comportan como procesadores específicos de E/S, denominados **canales de E/S**.

Los sistemas de interconexión externos entre una computadora y los dispositivos periféricos incluyen buses serie de alta velocidad (FireWire y USB) y sistemas de interconexión punto a punto inalámbricos (Bluetooth, TM WiFi).

MEJORA DEL RENDIMIENTO

Para mejorar el rendimiento de una computadora, se debe contemplar el **sistema de interconexión**. Para ello, se recurre a una jerarquía de buses separados, para aislar los dispositivos más rápidos de los más lentos. Se emplean buses serie de alta velocidad y conexiones punto a punto para eliminar problemas que se producen al compartir un bus entre elementos diferentes.

Ancho de bus, velocidad, direccionamiento y set de instrucciones

Uno de los aspectos más importantes en un sistema microprocesador es el **ancho de un canal** o **bus**, ya que determina la cantidad de bits que pueden transferirse simultáneamente. La **velocidad del canal** es la velocidad medida en MHz. El **rendimiento** o **capacidad** de un canal **C** medido en bits/seg o Mbits/segundo se obtiene al multiplicar la velocidad **V** en MHz y el ancho **A** en bits. Por ejemplo, si V = 400 MHz y A = 64 bits, C = 25.600.000.000 b/s = 25.600 Mb/s.

Respecto del bus de direcciones, su tamaño determina la cantidad de direcciones diferentes, **direccionamiento** , de posiciones de memoria y dispositivos de E/S que el microprocesador puede seleccionar. Para un bus de direcciones 32 bits ese número se obtiene por 2^{32}.
Para el bus de datos, 2^{64}, si 64 fuera el ancho de este bus o canal.

La **velocidad del microprocesador** se puede medir en **hertz** (MHz o GHz) de acuerdo a la frecuencia de la señal de reloj, en **MIPS** (millones de instrucciones por segundo) o en **FLOPS** (operaciones de punto flotante por segundo).

Se denomina **set** o **juego de instrucciones** de un microprocesador al conjunto de instrucciones que un determinado microprocesador puede ejecutar y que pueden ser de transferencia de datos, aritméticas y lógicas, de salto, de E/S y de control. En la definición del set de instrucciones para un microprocesador, intervienen diferentes factores que impactan sobre la arquitectura del sistema microprocesador y que es necesario conocer, por ejemplo: cómo se ejecutan las instrucciones, como están formadas y de qué manera es posible acceder tanto a ellas como a los datos.

El set de instrucciones del microprocesador Z80 contiene 158 instrucciones, incluyendo las 78 instrucciones del microprocesador 8080 y manteniendo la compatibilidad de software con el 8080.

MOTHERBOARD

Es una placa de circuito impreso que sirve como medio de conexión física entre el microprocesador, los circuitos electrónicos, los slots o ranuras para conectar la memoria RAM del sistema, la memoria ROM BIOS y otros slots que permiten la conexión de las placas de expansión adicionales.

Manejo de interrupciones

Una **interrupción** es un evento que altera la secuencia en la que el microprocesador ejecuta las instrucciones.

Resulta especialmente importante comprender el concepto de interrupción como una técnica que mejora el rendimiento del sistema, al evitar que el microprocesador permanezca detenido o ejecutando tareas improductivas mientras espera que un determinado periférico se encuentre preparado para realizar una nueva operación de E/S y así utilizar este tiempo para ejecutar otros programas. Por lo tanto, se delega en el dispositivo periférico la responsabilidad de comunicarse con el microprocesador cuando lo necesite, mientras que el microprocesador no interroga a ningún dispositivo de E/S y queda en espera de un requerimiento desde esos dispositivos cuando necesiten realizarle algún tipo de transferencia.

La gestión de interrupciones involucra tanto aspectos de hardware como de software. En el caso del hardware, porque dentro del bus de control de una computadora tiene que existir una línea especial o **línea de petición de interrupción**, **INT**, que sirve como vínculo para que el módulo de E/S comunique al microprocesador que se encuentra preparado para realizar una transferencia. En ese caso, la señal INT activa el módulo E/S para que el microprocesador reciba el dato.

Dentro del **ciclo de ejecución de una instrucción** se tiene una etapa de **comprobación de interrupciones** utilizada por el microprocesador para comprobar periódicamente si el módulo E/S requiere su atención. Recibida la **petición de interrupción**, **por hardware o por software**, por el microprocesador se ejecuta el **ciclo de reconocimiento de la interrupción** durante el que se detiene la ejecución del programa actual para acceder al módulo E/S y transferir

FUNCIONES DEL MOTHERBOARD

El motherboard determina el tipo de procesador, clase y cantidad de memoria, conectores, puertos instalados, velocidad de los buses. Realiza tareas para el funcionamiento de la computadora, como interconexión física de los dispositivos, administración/control/distribución de la energía eléctrica, comunicación de datos, temporización y sincronismo, y control y monitoreo de la temperatura, entre otros.

los datos al microprocesador. Finalizado este, se ejecuta un **retorno de interrupción** para continuar con la ejecución del programa principal.

Otra forma de clasificar las interrupciones es en **sincrónicas**, generadas por la CPU al ejecutar una instrucción, o **asincrónicas**, que son las generadas por otros dispositivos y no están sincronizadas con la señal de reloj, CLK, del sistema. Por ejemplo, Intel designa como excepciones las interrupciones sincrónicas y como interrupciones propiamente dichas las asincrónicas.

Manejo simultáneo de datos e instrucciones: SISD, SIMD, MIMD

Hasta ahora se analizó un modelo computacional estándar conocido como **arquitectura de Von Neumann**, que se caracteriza por emplear una CPU única, ejecutar un programa único, acceder a una memoria única, permitir operaciones read/write y la interconexión con otros dispositivos. Se trata de un modelo robusto que independiza al programador de la arquitectura subyacente y que facilita el desarrollo de distintas técnicas de programación estándares. En síntesis, Von Neumann propone una **máquina secuencial** y requiere que el microprocesador ejecute los programas procesando instrucciones en código máquina, una por una.

Figura 7. Supercomputadora paralelo **Blue Gen** de la empresa **IBM**, la más veloz del mundo, en el año 2005.

Para aumentar el rendimiento, se distribuye la carga computacional entre varias CPU trabajando simultáneamente en una técnica conocida como **paralelismo**. La **computación paralela** es una forma de cómputo en la que numerosas instrucciones se ejecutan en simultáneo de acuerdo con el principio que establece que un problema grande es posible dividirlo en otros más pequeños, para resolverlos al mismo tiempo (en paralelo).

Como necesidad de determinar arquitecturas alternativas, se pueden mencionar el aumento en las dimensiones de los problemas por resolver computacionalmente en cuanto a volumen de datos y precisión, además de las limitaciones físicas que impactan de manera directa sobre el poder de cómputo de los microprocesadores actuales.

En este contexto, se sugieren arquitecturas paralelas. En 1966, **Michael Flynn** categorizó las posibles arquitecturas paralelas considerando la forma en que se aplican las instrucciones y se manejan los datos.

TAXONOMÍA DE FLYNN			
		▼ INSTRUCCIONES	
		SI	SI
Datos	SD	SISD	MISD
	MD	SIMD	MIMD
S: Single	M: Multi	I: Instrucción	D: Datos

Tabla 3. Categorías de Flynn para soluciones de hardware (**Taxonomía de Flynn**).

Así, **SISD** significa 'instrucción y dato único' (en inglés, *Single Instruction Single Data*) y está asociado a la arquitectura convencional

INTERRUPCIONES DE HARDWARE Y DE SOFTWARE

Las **interrupciones de hardware** son provocadas por elementos externos al microprocesador y se gestionan con una línea dedicada en el bus de datos y ejecutando la comprobación de la interrupción. Las **interrupciones de software** o **excepciones** son acontecimientos internos (errores en la programación y condiciones anómalas). Se gestionan en el mismo momento, sin una línea de control ni comprobaciones.

de Von Neumann, por ejemplo las PCs. **MISD** significa 'instrucción múltiple, dato único' (en inglés, *Multiple Instruction Single Data*) y está asociado con computadoras que intentan resolver problemas de sistemas de propósito específico, en los que se debe balancear el ancho de banda entre una cantidad de cálculos intensiva y requerimientos de E/S sumamente grandes.

SIMD significa 'instrucción única, dato múltiple' (en inglés, *Single Instruction Multiple Data*) y está asociado con las computadoras útiles para el procesamiento de imágenes. Finalmente, **MIMD** significa 'instrucción y dato múltiples' (en inglés, *Multiple Instruction Multiple Data*) y está asociado a un modelo general que permite varias implementaciones (multiprocesadores y multicomputadoras), por ejemplo, mediante microprocesadores de propósito general o **clusters** de computadoras. Por supuesto que las soluciones propuestas a nivel hardware deben tener su correlato en un software específico.

Arquitecturas RISC y CISC

CISC, la principal tendencia tecnológica en el diseño de microprocesadores de los años 70, permitió crear estos dispositivos con capacidad para responder a un gran número de instrucciones que requerían más de un ciclo de reloj para ser ejecutadas y que, también, podían acceder a la memoria y tenían formatos diferentes. Esta corriente se conoció con el nombre de **CISC** o **computadora de set de instrucciones complejo** (en inglés, *Complex Instruction Set Computer*).

En realidad, los estudios de David Patterson y Carlo Sequin, entre otros, determinaron que las instrucciones más complejas, que requerían de varios ciclos de reloj para ejecutarse, se utilizaban con baja frecuencia y que, en general, eran las responsables de la caída en el rendimiento del sistema, ya que el esfuerzo de procesamiento se concentraba en su mayoría alrededor de las instrucciones más simples.

La tendencia actual y opuesta, **RISC** o **computadora de set de instrucciones reducido** (en inglés, *Reduced Instruction Set Computer*), pretendía construir una computadora que empleara muy pocas instrucciones para simplificar el diseño de la unidad de control, solamente las más requeridas, y otras para construir las instrucciones ausentes.

Algunas diferencias importantes con la corriente CISC radican en que solo una instrucción debe ejecutarse en cada ciclo, todas las

instrucciones tienen igual tamaño (pequeño, para ejecutarse en menos tiempo), el acceso a la memoria está restringido a las instrucciones del tipo cargar/almacenar (en inglés, *load/store*), posee gran cantidad de registros para almacenar las variables dentro del microprocesador, un compilador más complejo como consecuencia, y memorias caché en el interior del microprocesador para aumentar el rendimiento del sistema al mantener datos e instrucciones próximos a ser utilizados y, además, para soportar lenguajes de alto nivel.

Tecnología MMX

Una de las dificultades de los microprocesadores con tecnología anterior a la tecnología **extensiones multimedia** o **MMX** (en inglés, *MultiMedia eXtension*) es la baja eficiencia en el momento de ejecutar tareas relacionadas con los archivos de audio en cuanto a su reproducción, grabación y reconocimiento de la voz. Con respecto al video, optimizar la imagen en el monitor y en los gráficos para mejorar lo relacionado con el trazado de líneas, la manipulación de figuras y el trabajo con objetos en tres dimensiones, entre otras características. En cuanto a Internet, las sucesivas mejoras en las versiones del software utilizado en su diseño agregaron interactividad respecto de los gráficos, audio y video en el monitor del usuario cuando se accede a la Web a través de los navegadores.

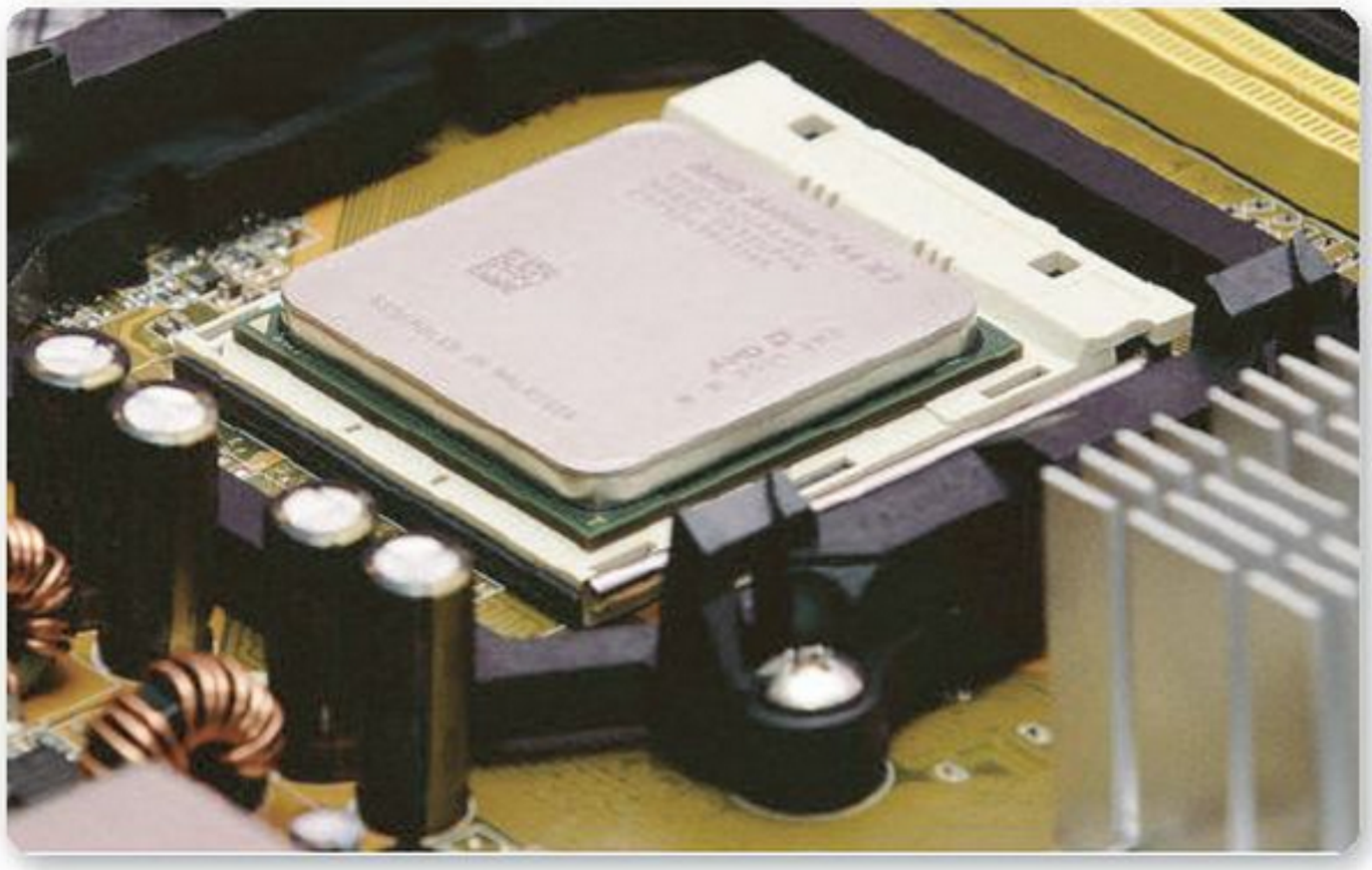


Figura 8. Microprocesador de la empresa **AMD Athlon** de 64 bits, 3600 MHz y doble núcleo.

En 1997, MMX fue una de las primeras tecnologías destinada a mejorar y acelerar las funciones relacionadas con multimedia e Internet, y uno de los microprocesadores de Intel, el Pentium MMX (Intel i860), en su momento, incorporó instrucciones SIMD adicionales a las que podía ejecutar un Pentium normal, lo que le permitió trabajar con 64 bits simultáneamente mediante la técnica SIMD, analizada antes en este capítulo. Es decir, que podía ejecutar una cuádruple palabra de 64 bits, 2 dobles palabras de 32 bits, 4 palabras de 16 bits. La palabra simple era de 8 bits.

Uno de los aspectos centrales para Intel fue mantener la compatibilidad del software existente con estos nuevos dispositivos, circunstancia que también debió enfrentar su competencia, AMD. Esto ocasionó que se limitara la incorporación de nuevos registros en los modelos de procesadores de ambas empresas.

RESUMEN

En este capítulo comenzamos por el análisis de los sistemas que incluyen microprocesadores y que constan de una unidad de procesamiento central (CPU), interfaces de entrada/salida y memoria. Explicamos también la arquitectura Von Neumann, que sustenta la estructura de un sistema basado en un microprocesador, describimos su organización interna, los elementos que la conforman, el funcionamiento general y cómo se interconectan cada una de sus partes.

Actividades

TEST DE AUTOEVALUACIÓN

1 Considere un microprocesador como el analizado en este capítulo y explique los roles que este dispositivo cumple en relación con un registro acumulador, un registro de estatus, una dirección de memoria y un registro contador de programa.

2 Explique las diferencias entre un microprocesador y un microcontrolador.

3 ¿Cuáles son las diferencias entre la arquitectura Von Neumann y la Harvard?

4 ¿Cómo se conocen las instrucciones que se ejecutan en un microprocesador?

5 ¿Cómo se denomina el registro interno de la CPU que contiene la dirección de la siguiente instrucción por ejecutar?

6 ¿Cómo se denominan los bits individuales que conforman el registro de estatus de un microprocesador?

EJERCICIOS PRÁCTICOS

1 Seleccione la hoja de datos de un microprocesador, por ejemplo 6800 de Motorola, y analice la información sobre su encapsulado, su diagrama de pines y la función de cada uno de ellos.

2 Identifique y mencione al menos cinco registros de uso general localizados en la CPU de un microprocesador Intel 8080.

3 Identifique la frecuencia de reloj en un microprocesador Motorola 6800, el período de reloj y la cantidad de ciclos de reloj que se requiere para ejecutar una instrucción.

4 Considere la suma de los números hexadecimales F9 y 08. Sobre la base del resultado de la suma de ambos, determine el valor de las banderas Z, N, C y V.

PROFESOR EN LÍNEA

Si tiene alguna consulta técnica relacionada con el contenido, puede contactarse con nuestros expertos: profesor@redusers.com.

*04

Interfaces del microprocesador

En los capítulos 1, 2 y 3, se hizo referencia a los sistemas microprocesadores, por lo tanto, debido a que estos operan como parte de ese sistema, requieren de interconexiones con los otros integrantes. Las conexiones se realizan mediante un dispositivo conocido como interfaz. El objetivo de este capítulo es comprender la arquitectura y las características de las interfaces microprocesador-memoria ROM, microprocesador-memoria RAM y microprocesador-dispositivos de E/S.

Servicio de atención al lector: usershop@redusers.com

Características eléctricas del microprocesador

La organización interna de un microprocesador, su arquitectura, varía de un modelo a otro, por lo que es complicado definir un modelo de microprocesador que represente todas las alternativas posibles, ya que cada uno posee su propia lógica de funcionamiento. El microprocesador Intel 8085 ha sido considerado un estándar de la industria de los microprocesadores durante mucho tiempo y, por esa razón, es el dispositivo de 8 bits sobre el que se centra nuestro análisis, aunque realizaremos también referencia a microprocesadores de otras empresas, como el Z80 de la empresa Zilog, uno de los mejores y más completos microprocesadores de 8 bits disponibles en el mercado. Por las razones indicadas antes, describiremos el microprocesador 8085 de Intel desde el punto de vista eléctrico, lo que permitirá, a partir de la comprensión de la teoría general e interpretando adecuadamente la hoja de datos del fabricante, entender dispositivos microprocesadores más complejos (16, 32 y 64 bits) con los que comparte la misma arquitectura.

Diagrama de terminales del microprocesador

El microprocesador 8085 combina el 8080, el circuito de reloj y el controlador del sistema en un circuito integrado único. Está fabricado con tecnología N-MOS y requiere un voltaje único de 5 volts. El 8085 se optimizó para que pudiera formar un sistema completo utilizando

FACTOR DE CARGA

Otro aspecto por considerar en la selección de un microprocesador es el **factor de carga** , que se relaciona con la cantidad de entradas a circuitos lógicos posibles de conectar a una salida del microprocesador. Este parámetro dependerá de la intensidad de corriente que necesitan dichas entradas y de la que es capaz de proporcionar a la salida considerada sin que disminuya la tensión que entrega.

dos circuitos periféricos especiales, uno de ellos con memoria RAM, puertos de entrada/salida y un timer (8155 u 8156), y el otro con memoria ROM o EPROM y puertos (8355 u 8755). Desde el punto de vista de la programación, su aporte ha sido prácticamente nulo, ya que se esperaba un **set de instrucciones** más poderoso y, en cambio, solo aumentó en dos las instrucciones del 8080.

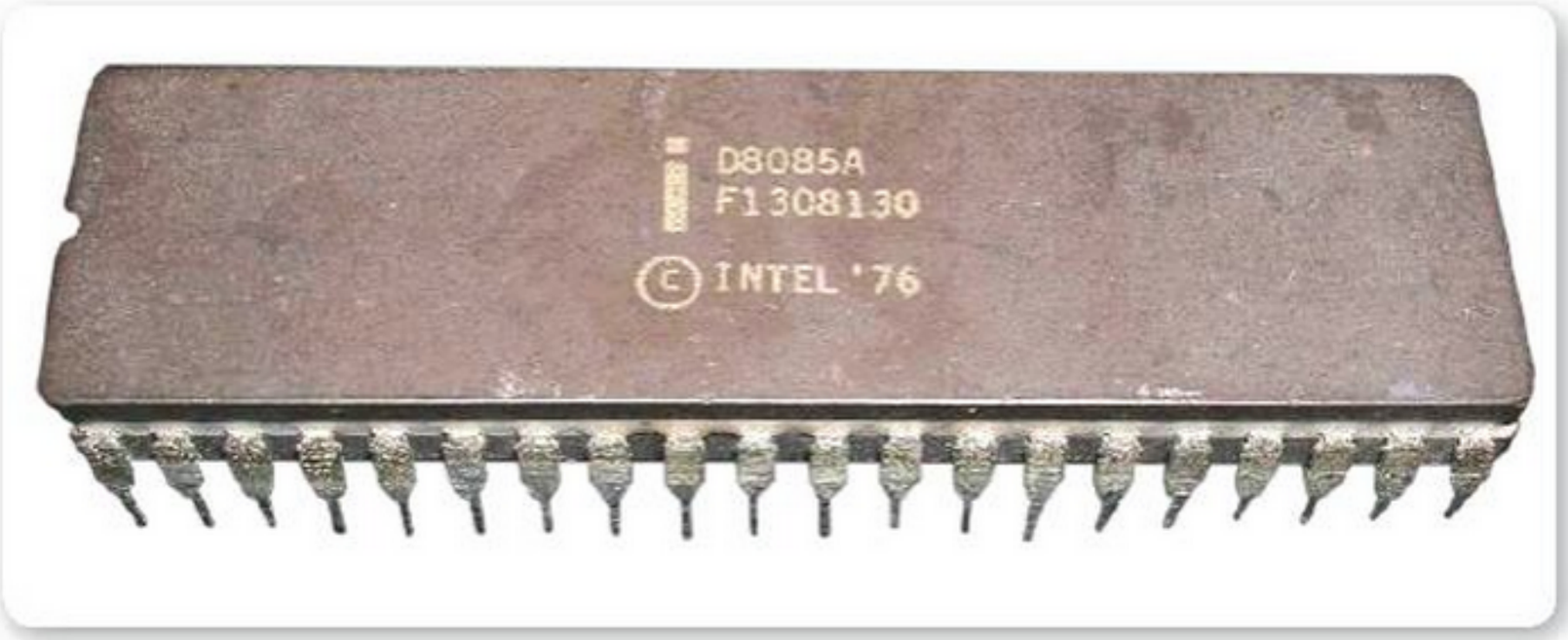


Figura 1. Microprocesador **Intel 8085A** introducido en el mercado en 1976. Utiliza tecnología **N-MOS** (MOS de canal N).

Niveles de voltaje y factores de carga

Es importante conocer las características eléctricas de cualquier dispositivo electrónico y, en particular, las de un microprocesador, de modo de garantizar su desempeño dentro de los límites seguros. Exponer un microprocesador a condiciones máximas durante períodos prolongados podría afectar seriamente la confiabilidad del dispositivo o causar daños irreversibles en su estructura interna. Los parámetros más importantes en la selección de un microprocesador son: la **tensión de alimentación** ; los **requerimientos en las tensiones de entrada** para los dos estados lógicos (0 y 1), expresados como un porcentaje de la tensión de alimentación, y la **disipación de potencia**, definida por la capacidad del microprocesador para transformar parte de la energía eléctrica que emplea para funcionar en calor sin deteriorarse.

En el caso del microprocesador 8085, su arquitectura está contenida en **cápsulas de doble fila** (**DIP**) de 40 pines, plásticas o cerámicas. Este chip se encuentra disponible en dos versiones: una **versión estándar** llamada **8085A** que opera a 3 MHz de reloj y otra **versión de alta velocidad** llamada **8085A-2** que opera a 5 MHz.

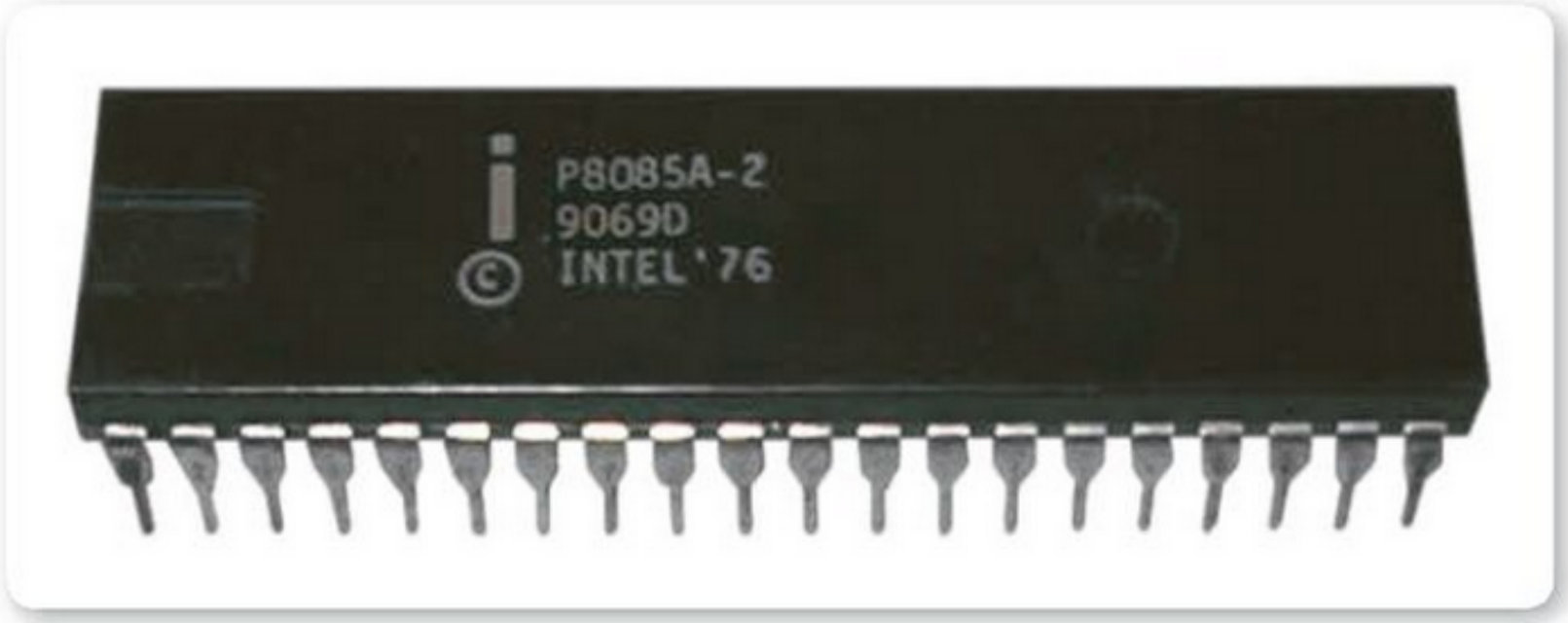


Figura 2. Microprocesador **Intel 8085A-2** con reloj a 5 MHz, que mejora el rendimiento del sistema donde está implementado.

Antes de utilizar un microprocesador es conveniente consultar la distribución de pines y la organización interna o **arquitectura del dispositivo** . Algunos de los parámetros sustanciales son: el número y tipo de registros utilizados, el ancho del bus de datos, el número de bits del bus de direcciones, las fuentes de alimentación requeridas, las entradas y salidas de control disponibles, y las entradas de interrupción existentes.

Estudiar el set de instrucciones del dispositivo facilitará descubrir el tipo de instrucciones por ejecutar, la forma de comunicación con los dispositivos de entrada/salida, con qué condiciones se efectúan saltos o bifurcaciones, qué operaciones aritméticas puede realizar, cómo opera con las interrupciones, y otros aspectos específicos para maximizar el uso de estos dispositivos. También es importante comprender la descripción de las **señales de control** , diagramas de temporización y características técnicas generales del microprocesador así como la función específica de cada uno de los pines.

Una vez comprendidos estos aspectos, podemos avanzar con el conocimiento del diagrama esquemático correspondiente al **sistema**

DIAGRAMA DE TEMPORIZACIÓN

En particular, los **diagramas de temporización** son muy importantes ya que muestran las relaciones entre la señal o las señales de reloj, y las demás señales internas y externas que están involucradas en la operación de un sistema microprocesador.

mínimo utilizando un microprocesador en particular y que se describe en la hoja de datos del fabricante. Allí se encuentra información respecto a cómo podemos conectar el microprocesador con otros circuitos de soporte.

Voltaje de alimentación

Si bien es necesario conocer el voltaje de alimentación del microprocesador (en el caso del 8085 de Intel está comprendido entre 4.75 y 5.25 volts), también es importante tener en cuenta las características eléctricas más significativas. Estas se extraen a partir del análisis de la **hoja de datos** del fabricante y se resumen en una tabla como la siguiente.

VOLTAJES					
DEFINICIÓN DEL PARÁMETRO	SÍMBOLO	VALOR MÍNIMO	VALOR MÁXIMO	UNIDAD DE MEDIDA	CONDICIONES EN QUE SE REALIZÓ LA PRUEBA
Temperatura	TA	0	70	°C	
Alimentación	Vcc	4.75	5.25	volt	
Tensión de entrada bajo VIL		-0.5	+0.8	volt	
Tensión de entrada alto VIH		2.0	Vcc + 0.5 volt		
Tensión de salida bajo	VOL		0.45	volt	IOL=2 mA
Tensión de salida alto	VOH	2.4		volt	IOL=400 uA
Corriente de alimentación ICC			170	mA	
Máxima corriente en bajo IOL			2	mA	
Máxima corriente en alto IOH			400	uA	

Tabla 1. Características eléctricas del microprocesador **Intel 8085**.

Circuitos de reloj y reinicio

Si bien cada microprocesador tiene sus propios circuitos de reloj y de reinicio, para el Intel 8085 se presentan y describen dos **circuitos de reloj** y un **circuito de reinicio** o **reset**. Este último facilita que podamos desconectar mediante hardware un microprocesador en función del estado de un pin en particular, de modo de reiniciar el estado de sus diferentes circuitos electrónicos internos.

Las configuraciones típicas de reloj son las siguientes: si se desea alta estabilidad de la frecuencia de salida, se utiliza un **cristal de cuarzo** ; mientras que, si la estabilidad de la frecuencia de salida no es un factor determinante, se utiliza una red **RC** (resistencia-condensador).

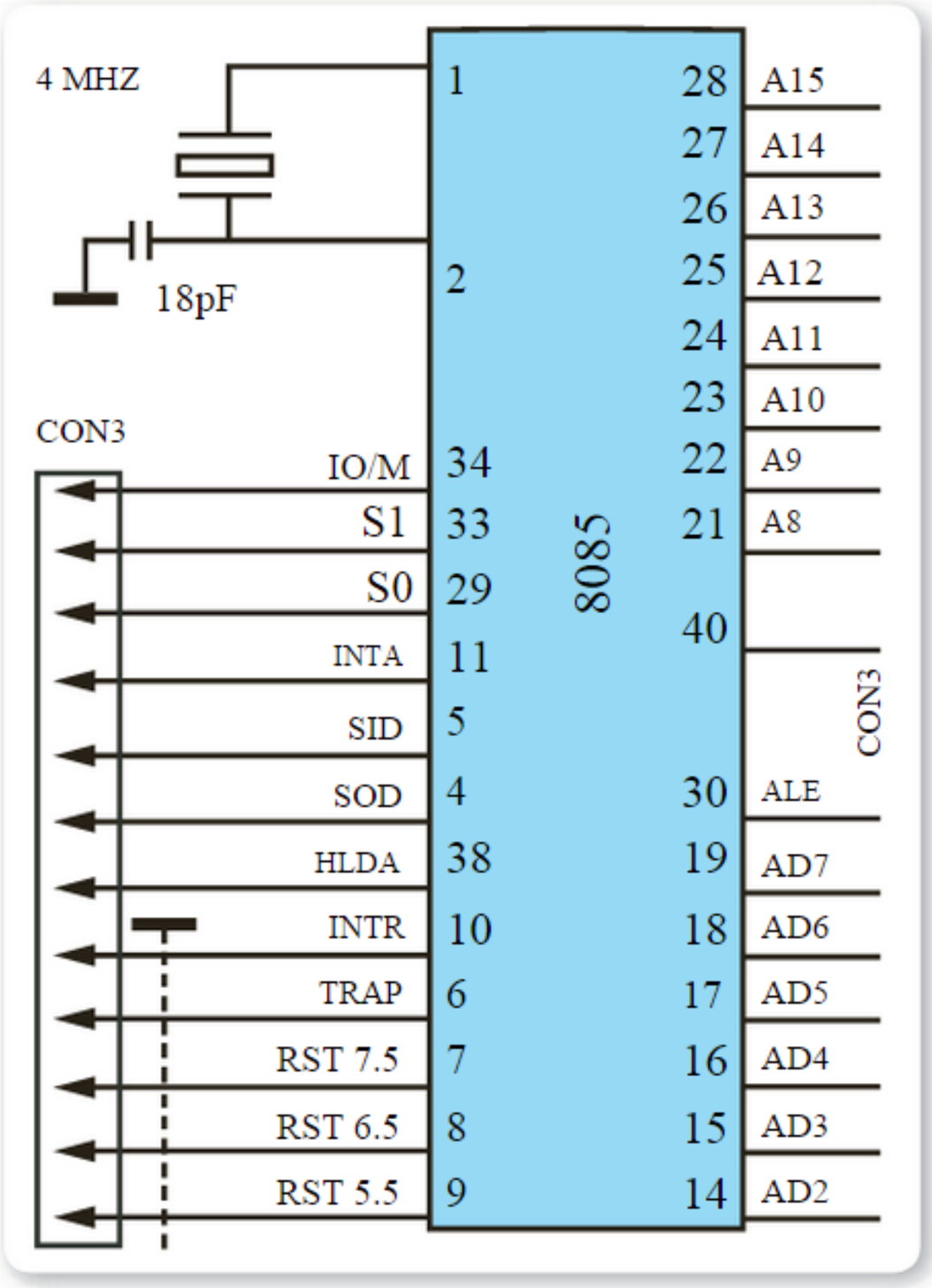


Figura 3. Configuración típica de un circuito de reloj basado en un cristal de cuarzo conectado en los pines 1 (X1) y 2 (X2) del 8085.

Es interesante tener en cuenta los parámetros más importantes que debe tener un cristal de cuarzo para ser utilizado en circuitos de reloj. La **frecuencia de resonancia paralelo** (**fr**) es el doble de la

IMPORTANCIA DE LOS CIRCUITOS DE RELOJ

Todo microprocesador necesita de un generador de pulsos de reloj (CLK, que controla el contador de programa y sincroniza sus funciones. En el microprocesador 8085, el circuito de reloj viene incorporado en la misma CPU, y su frecuencia se puede controlar mediante un cristal o con una red RC externa conectada entre las líneas X1 (pin 1) y X2 (pin2).

frecuencia de reloj deseada; la **capacidad de carga** (**CI**) debe ser menor o igual a 30 pF; la **capacidad en paralelo** (**Cs**), menor o igual a 7 pF; la **resistencia equivalente en paralelo** (**Rs**), menor o igual a 75 Ohm; la **potencia disipada** (**Pd**), menor a 10 mW, y la **estabilidad en frecuencia** (**fs**) de ± 0.005 %.

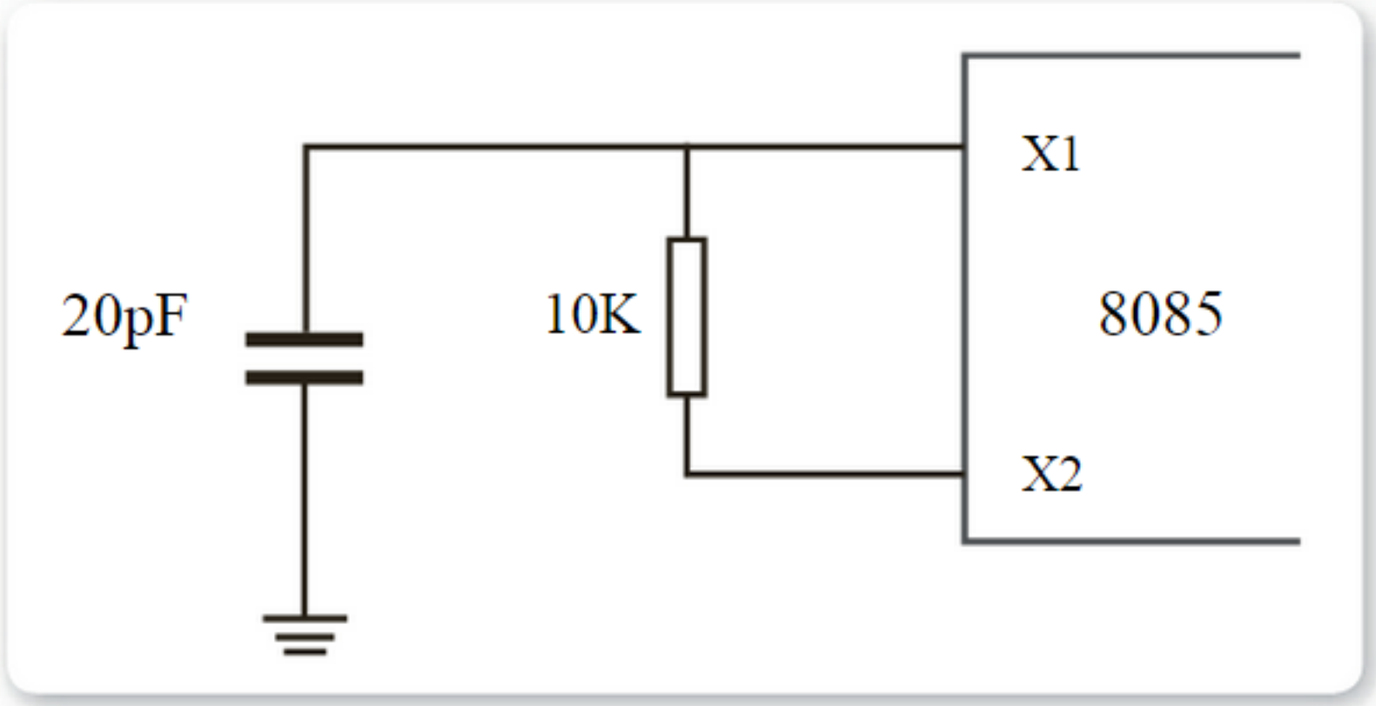


Figura 4. Configuración de un circuito de reloj basado en una red RC conectada entre los pines 1 (X1) y 2 (X2) del 8085.

En ambos casos, la señal de reloj se obtiene en el pin 37 del 8085 (CLKOUT) y tiene una frecuencia igual a la mitad de la definida por el cristal de cuarzo. En esta configuración, los límites de funcionamiento están claramente definidos: la frecuencia de resonancia del cristal debe ser superior a 1 MHz e igual al doble de la frecuencia interna del reloj deseada. La máxima frecuencia de reloj interna generada está determinada por la versión del microprocesador.

Para la versión 8085A (estándar), la frecuencia básica de reloj es de 3 MHz y, en la versión de alta velocidad, 8085A-2, de 5 MHz. Esto significa que, para operar ambas versiones en su máxima velocidad, se deberá utilizar un cristal de 6 MHz en la versión estándar y de 10 MHz en la versión de alta velocidad. También es posible controlar el reloj interno del 8085 mediante una señal externa. En la hoja de datos del fabricante se encuentran algunas alternativas de circuitos para realizarlo.

> CADA MICROPROCESADOR TIENE SUS PROPIOS CIRCUITOS DE RELOJ Y DE REINICIO

En cuanto al reset, el 8085 se puede conectar o desconectar sin mayor dificultad mediante hardware, de acuerdo al estado de su

entrada de reset (pin 36). El microprocesador ingresa en estado bajo cuando la entrada de reset pasa a un valor bajo (0). Entonces, el contador de programa se carga con ceros y se borran los flip-flops tanto de habilitación de interrupciones como de reconocimiento de hold.

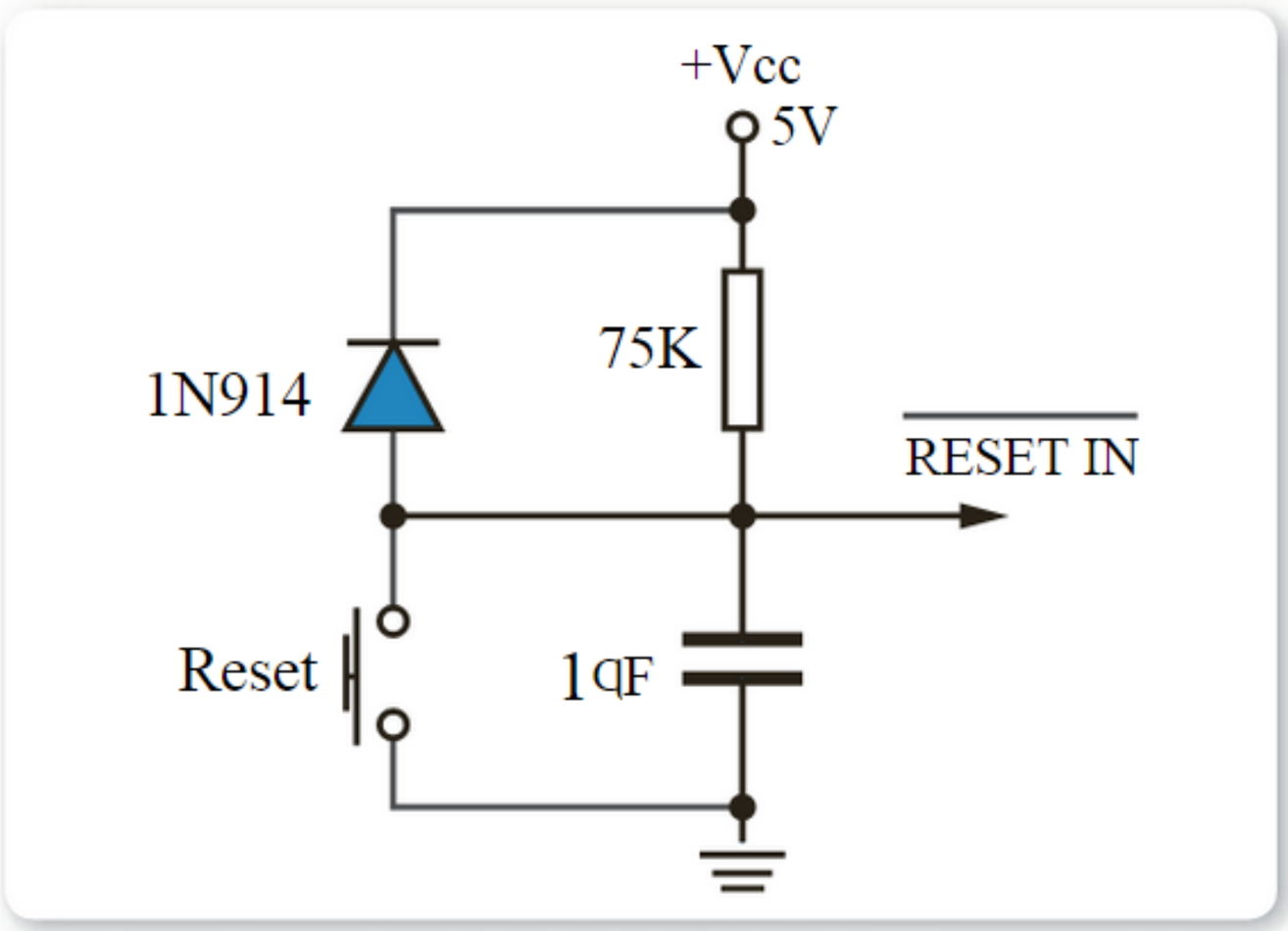


Figura 5. Configuración típica de un circuito de reset en el microprocesador Intel 8085.

Por otra parte, en el momento en que la señal en el pin 36 pasa a un valor alto (1), el microprocesador 8085 ejecuta las instrucciones del programa almacenado en la memoria. En un sistema práctico se podría utilizar la señal de reset para inicializar el microprocesador en el momento en que se conecta la fuente de alimentación, para acceder tanto al bus de datos como al bus de direcciones durante el proceso de programación del sistema basado en el microprocesador 8085.

CRISTAL DE CUARZO

Su elevado **factor Q** proporciona señales muy estables con bajos niveles de ruido de fase en osciladores y a un costo razonable. La **frecuencia de oscilación** es fija. Una desventaja es el tamaño físico del cristal, que se relaciona con su frecuencia natural de resonancia (**efecto piezoeléctrico**). Se deben extremar las precauciones respecto de la máxima temperatura y del tiempo de exposición al calor durante la soldadura.

Interfaz con los dispositivos de memoria

La estructura del sistema de E/S está formada principalmente por los periféricos, los módulos de E/S y los sistemas de interconexión externos. En una operación básica de E/S se realizan, en primer lugar, la programación de la operación de E/S, luego la transferencia de datos y, posteriormente, la finalización de la operación de E/S. Se tiene una transferencia de datos de **entrada** cuando el periférico es el que genera la información recibida por el microprocesador o por la memoria, y es de **salida** en el caso inverso.

Existen circunstancias en las que el microprocesador libera los buses de datos y de direcciones durante un tiempo en el que un dispositivo periférico puede acceder directamente a la memoria principal del sistema sin hacerlo por medio del microprocesador, en un tipo de acceso denominado **acceso directo a memoria** (en inglés, *Direct Memory Access* o **DMA**). En general, las transferencias de datos de entrada y de salida desde el microprocesador a los buses correspondientes se relacionan con operaciones de lectura de memoria, escritura de memoria, lectura de E/S, escritura de E/S o manipulación de las interrupciones.

Interfaz con la memoria ROM

Supongamos un sistema microprocesador y focalicemos en la conexión microprocesador 8085, 16 líneas en el bus de direcciones y 8 líneas en el bus de datos, con una memoria ROM de, por ejemplo, 4 K x 8 bits (memoria de 4 K con celdas direccionables de 8 bits cada una o, lo que es lo mismo, 4 K).

MEMORIAS ROM

Una vez programadas las **memorias de solo lectura** (en inglés, **Read Only Memory** , **ROM**), o memorias **pasivas**, solo realizarán operaciones de lectura. No son volátiles y se pueden utilizar para almacenar códigos, generadoras de caracteres, funciones aritméticas complejas, unidades de control microprogramadas o almacenamiento de partes del sistema operativo (BIOS), entre otras funciones.

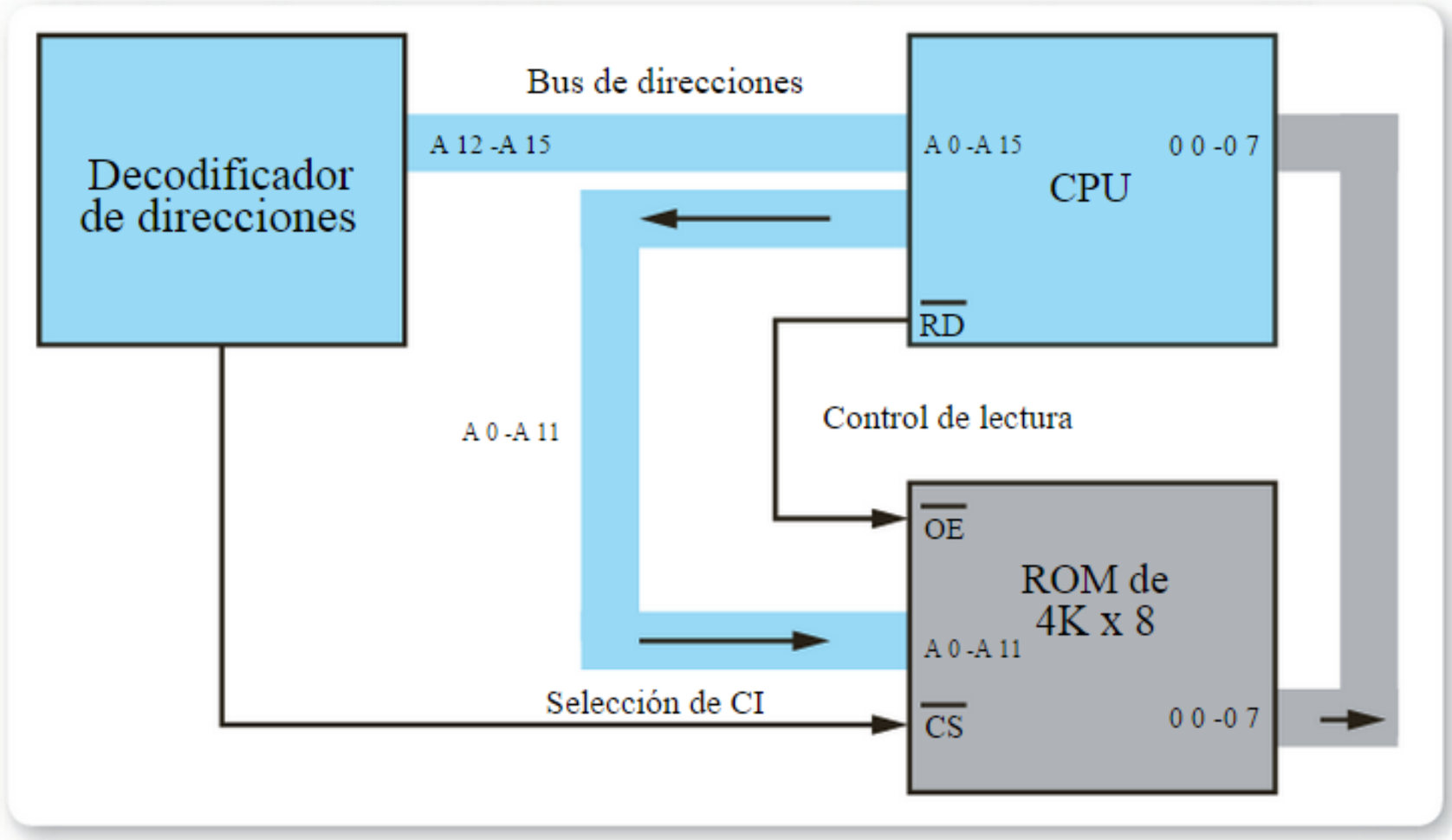


Figura 6. Interfaz de la memoria ROM con un microprocesador de 8 bits de datos y 16 bits de direcciones.

Con un bus de datos de 8 bits (ancho de palabra "m" de 8 bits), las 8 líneas del bus se conectan a los pines de salida de la memoria ROM, mientras que la **salida de lectura** del microprocesador va a la **entrada de habilitación de salida** de la ROM.

En cuanto al bus de direcciones (A_0-A_{15}), los 12 bits menos significativos (A_0-A_{11}) se conectan a la memoria ROM de 4 K x 8, por lo que el decodificador de direcciones tiene capacidad para acceder a 4096 posiciones, o palabras de solo lectura de memoria, de 8 bits cada una ($2^{12} = 4096$). Por otra parte, (A_{12}-A_{15}), las 4 líneas del bus de direcciones más significativas, se utilizan en el decodificador de direcciones para poner en estado bajo la **entrada de habilitación** de la memoria ROM. El microprocesador debe cumplir con las siguientes etapas para acceder y leer los datos desde la memoria ROM.

MEMORIAS RAM

Las **memorias de acceso aleatorio** (en inglés, **Random Access Memory**, **RAM**), también llamadas **activas**, se caracterizan por tener tiempos de lectura y escritura similares para cualquier posición de la memoria, por ser volátiles y perder su contenido cuando dejan de ser alimentadas eléctricamente.

En primer lugar, inicializar las direcciones A_0-A_{11}. Luego, poner la entrada de habilitación de salida de la ROM en estado bajo mediante la salida de lectura del microprocesador y, finalmente, poner en bajo la entrada de habilitación de la memoria ROM.

A modo de ejemplo, supongamos que el microprocesador accede a la posición de memoria 0000_H ($0000\ 0000\ 0000\ 0000_2$, 0_{10}). De los 16 bits correspondientes al bus de direcciones, los 12 bits menos significativos se aplican a las 12 líneas de direcciones que unen al microprocesador con la memoria ROM, mientras que los 4 bits más significativos se decodifican en el decodificador de direcciones.

Si los bits A_{12} a A_0 son 0000_2, la salida del decodificador de direcciones producirá un valor bajo (0), que habilitará la entrada de selección de la memoria ROM.

> EL TIEMPO DE ACCESO DE LECTURA ES UN PARÁMETRO CARACTERÍSTICO DE LA MEMORIA ROM

Ciclos de búsqueda y ejecución

En el diseño y utilización de una interfaz entre una memoria ROM y el microprocesador, debemos tener en cuenta el **direccionamiento**, analizado antes, y la **temporización**. Físicamente, el cambio de estado de un dispositivo electrónico en una entrada o salida no se produce en forma instantánea. En una memoria ROM, un parámetro característico de la memoria que se esté utilizando es el **tiempo de acceso de lectura** y se define como el tiempo que se necesita para que los decodificadores internos de la memoria ROM localicen el byte correcto en la memoria. A partir de ese momento, se puede realizar la lectura de la posición de memoria seleccionada.

MAPA DE MEMORIA Y BITS DE DIRECCIONES

Es un esquema para conocer cómo se organiza la memoria en un dispositivo. En microprocesadores de 8 bits con bus de direcciones de 16 bits, es de 64 K posiciones de memoria. De los 16 bits de direcciones, los 4 bits más significativos (A_{12}-A_{15}) seleccionan un segmento de memoria, y los 12 bits menos significativos (A_0-A_{11}) determinan la posición específica de memoria dentro de ese segmento.

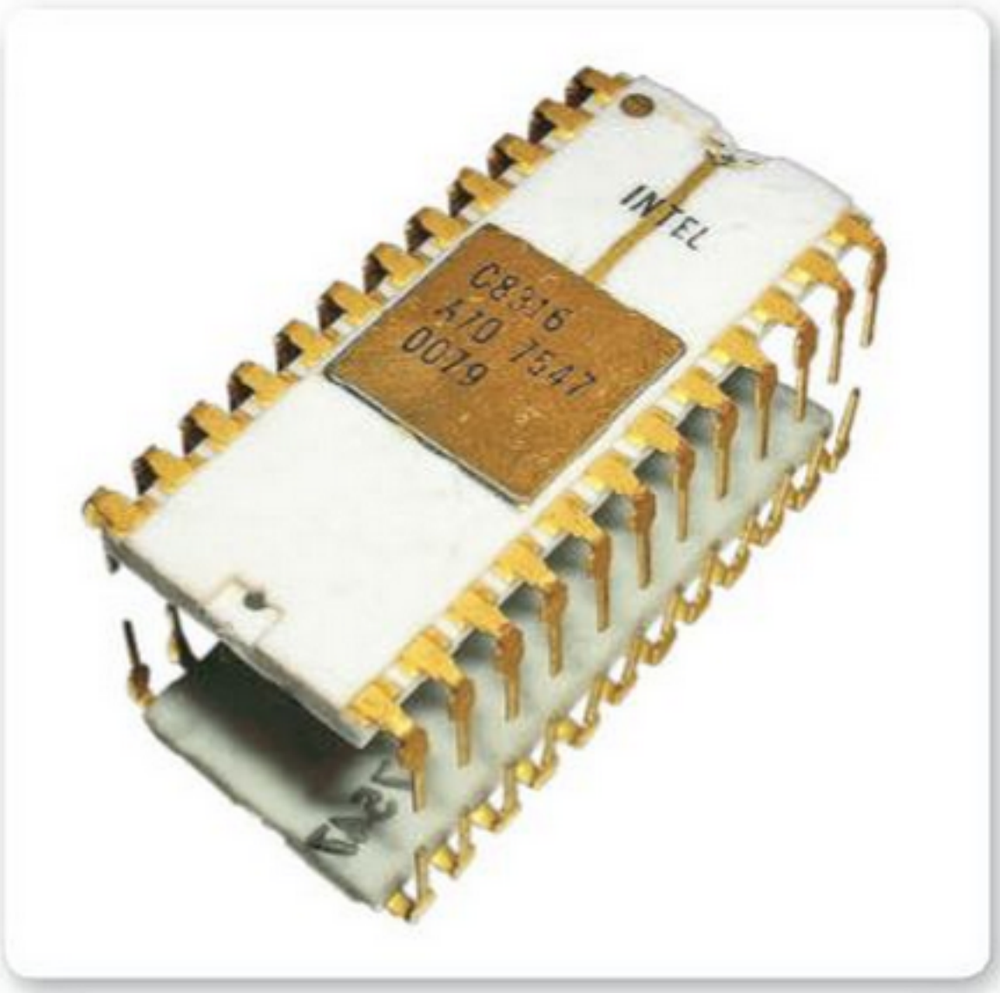


Figura 7. Circuito integrado **Intel 8316** . Es una memoria ROM de 2 K x 8 con tecnología **N-MOS**.

Un circuito integrado perteneciente a la familia de microprocesadores 8085 es el 8316, con una memoria ROM de 2 K x 8 construida bajo tecnología N-MOS, cuyo tiempo de acceso es de 550 nanosegundos (ns).

Para configurar una memoria ROM de 4K x 8, es necesario utilizar dos circuitos integrados 8316.

Otro chip de memoria RAM utilizado por el microprocesador 8085 es el chip 8102 (también 2102), construido con tecnología N-MOS, una memoria estática (SRAM) de 1 Kb y 350 ns de tiempo de acceso.

Esta memoria está organizada como una matriz de 1024 x 1 bit (1 kbit) y, por ello, requiere diez líneas de direccionamiento ($2^{10} = 1024$). Posee, además, líneas de datos de entrada y salida separadas, así como las líneas usuales de habilitación de lectura y escritura (R/W), y de selección de chip CE (habilitación). CE = 1 inhabilita la memoria, mientras que CE = 0 la habilita para su lectura o escritura (con R/W = 1 y R/W = 0, respectivamente). Por lo tanto, se pueden asociar módulos para conformar memorias RAM mayores, tanto en número de palabras como en el tamaño de la palabra.

Interfaz con la memoria RAM

Se denomina **memorias RAM** a los dispositivos de almacenamiento de lectura/escritura; estas memorias pueden ser dinámicas o también estáticas. Las estáticas se conectan con mayor facilidad que las dinámicas, mediante interfaces.

En un sistema microprocesador típico, la memoria RAM está organizada como una unidad de 4096 palabras (4 K) de 8 bits cada una, aunque físicamente este módulo de memoria podría contener varios circuitos integrados RAM.

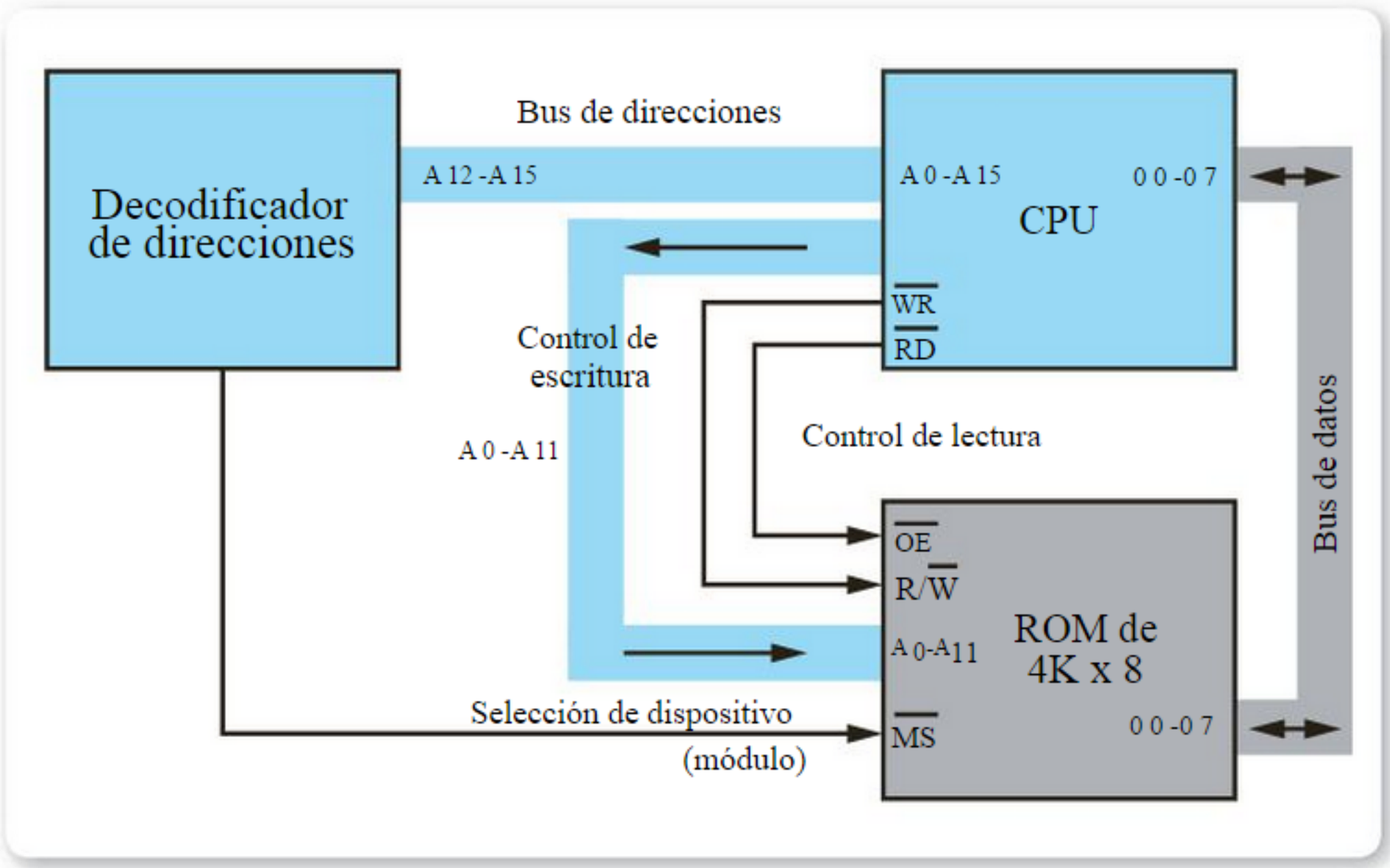


Figura 8. Interfaz de la memoria RAM con un microprocesador de 8 bits de datos y 16 bits de direcciones.

Al igual que lo que ocurre con la memoria ROM, el decodificador de direcciones proporciona una señal para habilitar la línea de selección de la memoria, enviando un pulso bajo a la entrada de selección del

módulo RAM para habilitar la memoria RAM. Esto es así solamente cuando las cuatro líneas de dirección más significativas (A_{12}-A_{15}) son las correspondientes a la posición de la memoria seleccionada, mientras que los 12 bits de direcciones menos significativos (A_0-A_{11}) son decodificados mediante el circuito de decodificación de la misma memoria RAM. Además, la memoria RAM posee una línea de control de escritura entre la entrada de habilitación de escritura del microprocesador y la entrada de lectura/escritura de la RAM: un nivel alto en esta entrada habilitará la operación de lectura en la RAM, mientras que un nivel bajo habilitará las salidas del módulo RAM.

Ciclos de lectura y escritura en RAM

El **ciclo de lectura de datos** se analiza mediante su **diagrama de tiempos** correspondiente. Las líneas de dirección de memoria pueden estar en estado alto o en estado bajo.

Un parámetro importante es $\mathbf{t_{RC}}$, que debe ser de, al menos, una cantidad de nanosegundos. $\mathbf{t_{RA}}$ es el **tiempo de acceso de lectura** e indica el retraso en la operación de lectura desde el instante en que se aplica la dirección a los pines A_0-A_{15} (bus de direcciones) hasta que el dato se encuentra listo en los pines D_0-D_7 del bus de datos.

Un circuito integrado perteneciente a la familia de microprocesadores 8085 y que funciona como memoria RAM es el 8111: una RAM estática de 256 palabras de 4 bits (256 x 44 bits). Para la memoria RAM 8111, este tiempo es de un máximo de 850 ns.

$\mathbf{t_{CEA}}$ es el **tiempo de acceso para la habilitación del chip** y mide el tiempo transcurrido desde la aplicación de la señal habilitadora hasta que el dato está listo. Todos estos parámetros se encuentran en la hoja de datos del componente.

TIEMPO DE CICLO Y VELOCIDAD

El **tiempo de un ciclo** es el tiempo mínimo entre dos accesos sucesivos a la memoria; este es mayor que el tiempo de acceso. La **velocidad de transferencia** es la velocidad a la que se puede leer o escribir un dato de memoria y, en las memorias RAM, es inverso al tiempo de ciclo. Se mide en bytes por segundo o en MiB/segundo o GiB/segundo. También se lo conoce como **ancho de banda** de la memoria.

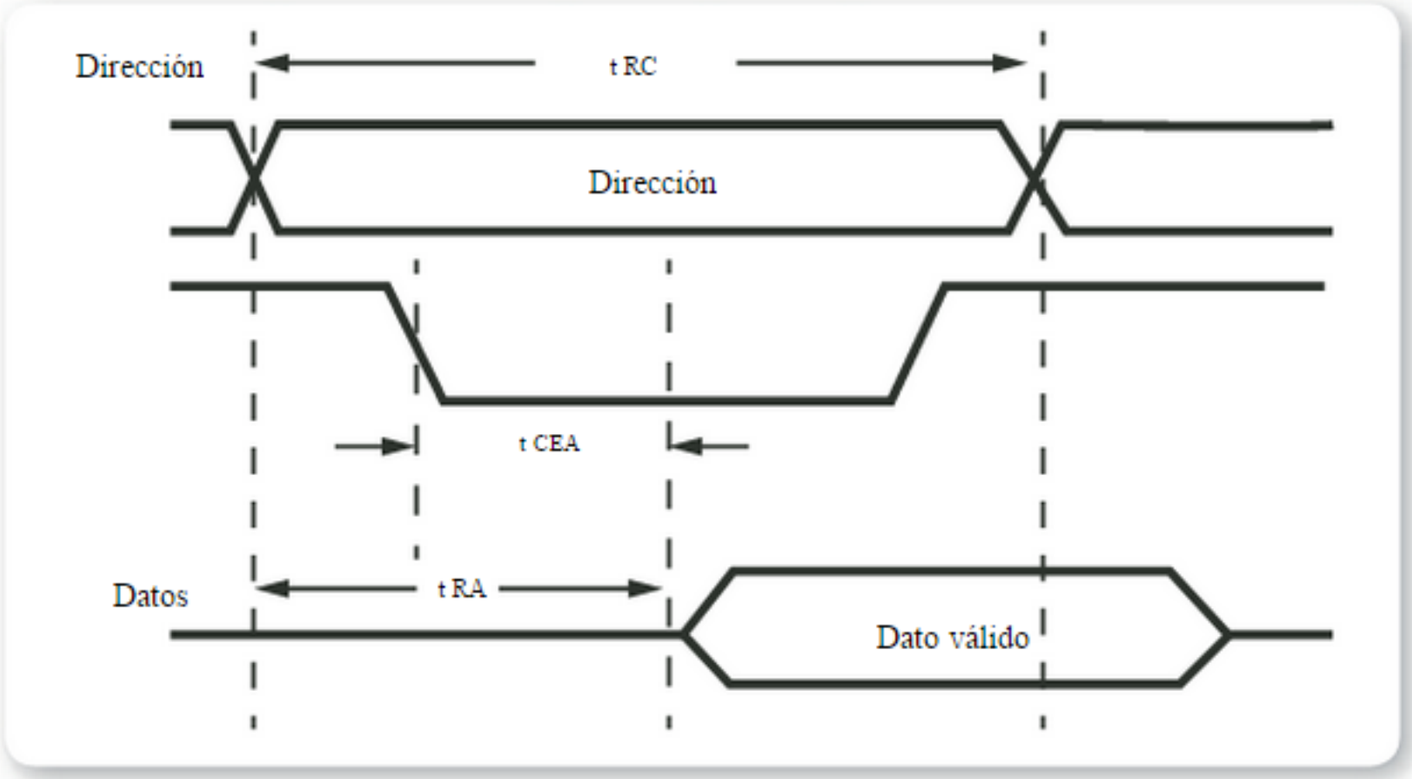


Figura 9. Diagrama de tiempos correspondiente al ciclo de lectura de datos en una memoria RAM genérica.

El **ciclo de escritura de datos** también se analiza mediante su diagrama de tiempos correspondiente. Es importante reconocer el **tiempo del ciclo de escritura** $\mathbf{t_{WC}}$, que para la RAM 8111 memoria es de un mínimo de 850 ns. El **tiempo de establecimiento** $\mathbf{t_S}$ (**setup time**) se define como el tiempo en que se alcanza un nivel lógico en una entrada antes de que ocurra una transición en otra entrada. Es necesario una cierta cantidad mínima de tiempo entre estas transiciones para evitar errores lógicos.

Se identifican tres tiempos de establecimiento en las formas de onda.

- El **tiempo de establecimiento de la dirección**, $\mathbf{t_{AS}}$, es el tiempo en que la dirección tiene que estar establecida antes que se aplique el pulso WRITE y, así, evitar la posibilidad de la escritura de datos erróneos en la memoria.
- El **tiempo de establecimiento de la habilitación del chip**, $\mathbf{t_{CES}}$, tiene un mínimo especificado de 0 ns.
- El tercer tiempo de establecimiento es el de los datos, $\mathbf{t_{DS}}$, que

TIEMPO DE ACCESO (LATENCIA)

En una memoria RAM, se denomina **tiempo de acceso** al tiempo que transcurre desde que una dirección de memoria es visible para los circuitos de la memoria hasta que el dato se encuentra almacenado (escritura) o está disponible para ser utilizado (lectura).

mide el intervalo entre la aplicación de los datos de entrada y la deshabilitación de la línea WRITE.

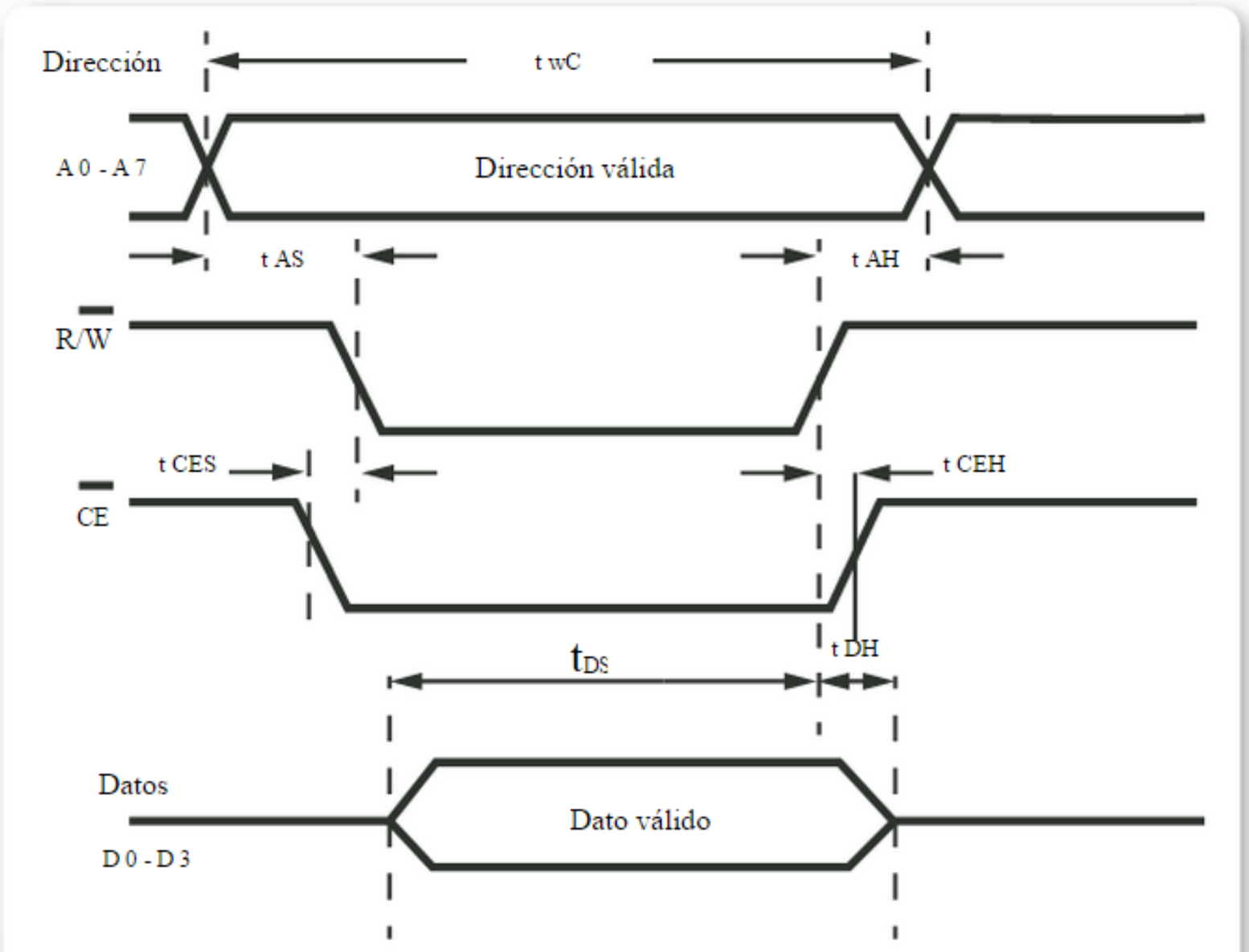


Figura 10. Diagrama de tiempos correspondiente al ciclo de escritura de datos en una memoria RAM genérica.

Finalmente, el **tiempo de sostenimiento** , t_H (**hold time**), es el tiempo durante el cual un nivel lógico se sostiene en una entrada después que ocurre una transición en otra entrada. En el ciclo de escritura también se identifican el **tiempo de sostenimiento de la dirección**, t_{AH}, un intervalo luego que se deshabilita la línea WRITE donde la dirección no debe cambiar para evitar que los datos se escriban en otra celda. t_{ECH}, **tiempo de sostenimiento de la habilitación del chip**, se especifica con un mínimo de 0 ns, y el **tiempo de sostenimiento del dato** , t_{DH}, debe ser, al menos, de 100 ns en la RAM 8111.

Todos los intervalos temporales citados antes corresponden al chip Intel 8111, y sus valores específicos se pueden consultar en la hoja de datos. Debemos tener en cuenta que otras memorias pueden tener valores diversos para dichos tiempos e, incluso, utilizar otra nomenclatura.

Direccionamiento de memoria

La memoria es una de las unidades más importantes (luego de la unidad central de control o CPU) en los sistemas microprocesadores y computadoras actuales, dado que almacena las instrucciones codificadas en binario y todos los datos de un programa. En el mundo de la electrónica digital, la memoria es una de las unidades que más se ha desarrollado por el hecho de que el microprocesador tiene que acceder y modificar los datos y programas almacenados en ella lo más rápido posible.

Se entiende por **direccionamiento** a la forma en que se interpretan los bits de un campo de dirección de una instrucción para localizar el operando o la dirección destino del resultado de la instrucción.

A su vez, los **modos de direccionamiento** son los procedimientos empleados por el microprocesador para acceder a operandos, instrucciones, posiciones de memoria, registros de entrada/salida, entre otros.

Analizaremos los modos de direccionamiento que soporta el microprocesador 8085 de Intel. Así, los campos de bits dedicados a direcciones serán interpretados según el tipo de direccionamiento empleado en cada caso para obtener la **dirección efectiva**, un número que identifica la posición de memoria por utilizar (dónde se encuentra la información o dónde se quiere depositar dicha información).

Arquitectura Von Neumann

Dentro del modelo de Von Neumann, la memoria fue el elemento que permitió la construcción de las computadoras como se las conocen actualmente.

DECODIFICADOR DE DIRECCIONES

Desde la óptica del hardware, este decodificador es un dispositivo que permite incrementar el número de direcciones por utilizar. Cada configuración del bit de dirección seleccionado habilita una única posición de memoria. Las funciones básicas de un decodificador son dos: adaptar los buses de direcciones y seleccionar el rango de direcciones en el bus de direcciones del microprocesador.

Antes que Von Neumann presentara su modelo de arquitectura que incluía una memoria para almacenar instrucciones, las computadoras requerían introducir datos y programas cada vez que era necesario un proceso distinto. Tradicionalmente los sistemas con microprocesadores se basan en esta arquitectura, donde la unidad central de proceso (CPU) está conectada a una memoria principal única en la que se guardan las instrucciones de programa y los datos. A dicha memoria se accede a través de un sistema de buses único (control, direcciones y datos).

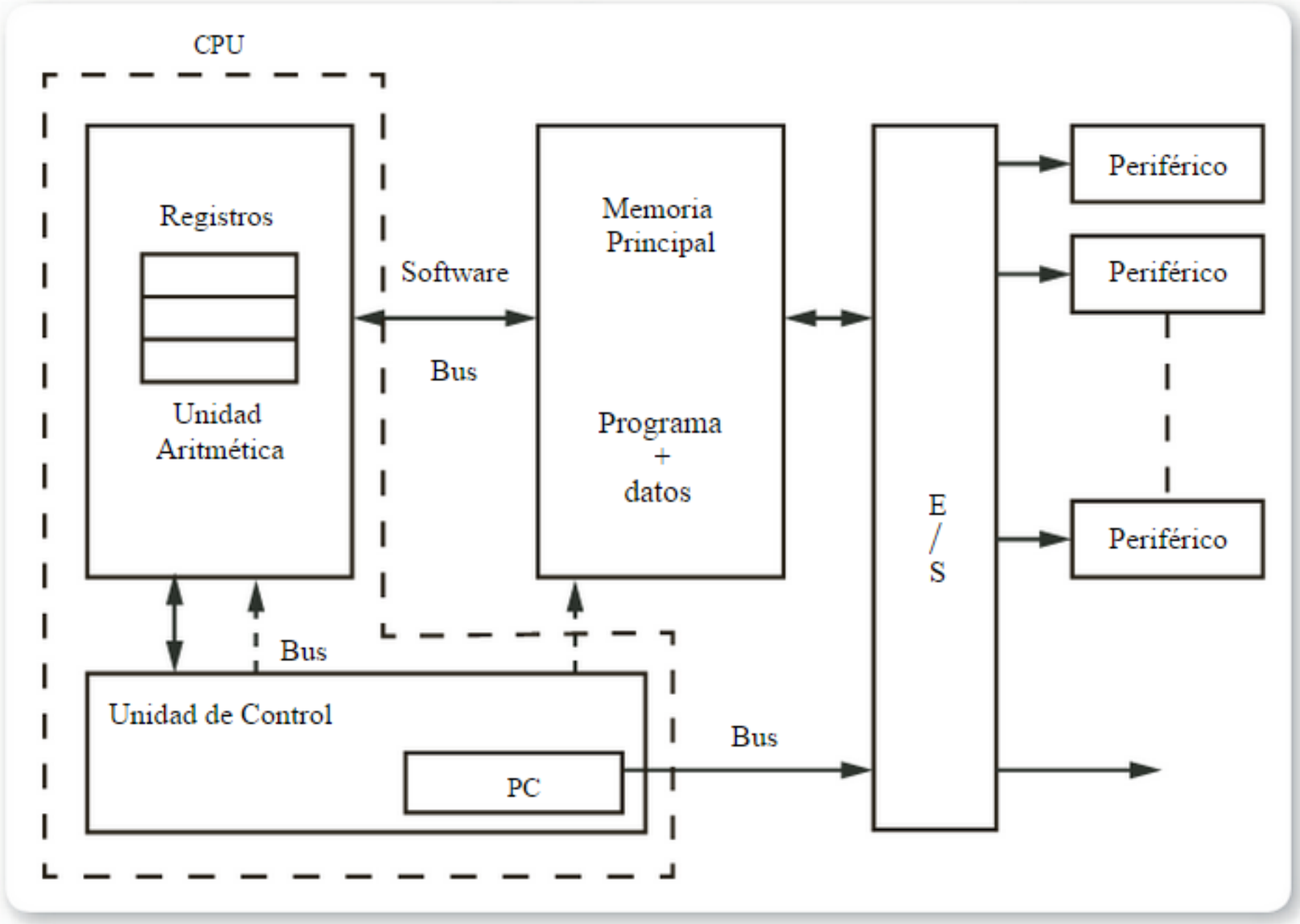


Figura 11. Diagrama de la arquitectura Von Neumann. Se accede a la memoria principal que almacena datos y programas mediante un bus único.

En un sistema con arquitectura Von Neumann, el tamaño de la unidad de datos o instrucciones está fijado por el ancho del bus que comunica la memoria con la CPU. Por ejemplo, un microprocesador de 8 bits, como el 8085 con un bus de 8 bits, deberá operar con datos e instrucciones de una o más unidades de 8 bits (bytes) de longitud. Así, para acceder a una instrucción o dato de más de un byte de longitud, tendrá que realizar más de un acceso a la memoria. De cualquier forma, al tener un bus único, el microprocesador será más lento en su respuesta dado que no accederá a otra posición de memoria hasta que no finalicen las transferencias de datos de la instrucción anterior.

En cuanto al direccionamiento de la memoria, el microprocesador 8085 utiliza cinco modos de direccionamiento: **implicado**, **de registro**, **inmediato**, **directo** e **indirecto**, además de modos **combinados**. En el 8085, cada modo de direccionamiento tiene instrucciones asociadas que pertenecen al set de instrucciones del microprocesador.

DIRECCIONAMIENTOS EN EL 8085

DIRECCIONAMIENTO	CARACTERÍSTICAS
Implicado	Inherente a la instrucción correspondiente.
Por registro	La instrucción lleva el registro en el que está el dato que va a ser tratado.
Inmediato	La instrucción contiene el dato con el que se desea operar; el dato puede ser de uno o dos bytes.
Directo absoluto	La instrucción contiene la dirección exacta y completa donde se encuentra el dato.
Indirecto de registro	En la instrucción se especifica un registro cuyo contenido apunta a una dirección de memoria en la que se encuentra el dato.
Combinados	Algunas instrucciones utilizan combinaciones de direccionamiento, como por ejemplo indirecto y de registro.

Tabla 2. Modos de direccionamiento en el microprocesador Intel 8085.

Los diferentes modos de direccionamiento del microprocesador 8085 le aportan una cierta potencialidad y le simplifican tanto realizar operaciones con datos como llevarlas a cabo con mayor facilidad.

Arquitectura Harvard

Las debilidades de la arquitectura Von Neumann se relacionan con la restricción en la longitud de las instrucciones en el bus de datos, lo que provoca que el microprocesador tenga que realizar varios accesos a la memoria para buscar instrucciones complejas, y la limitación de la velocidad de operación a causa del bus único que no permite el acceso simultáneo a datos e instrucciones, impidiendo la superposición de ambos tiempos de acceso.

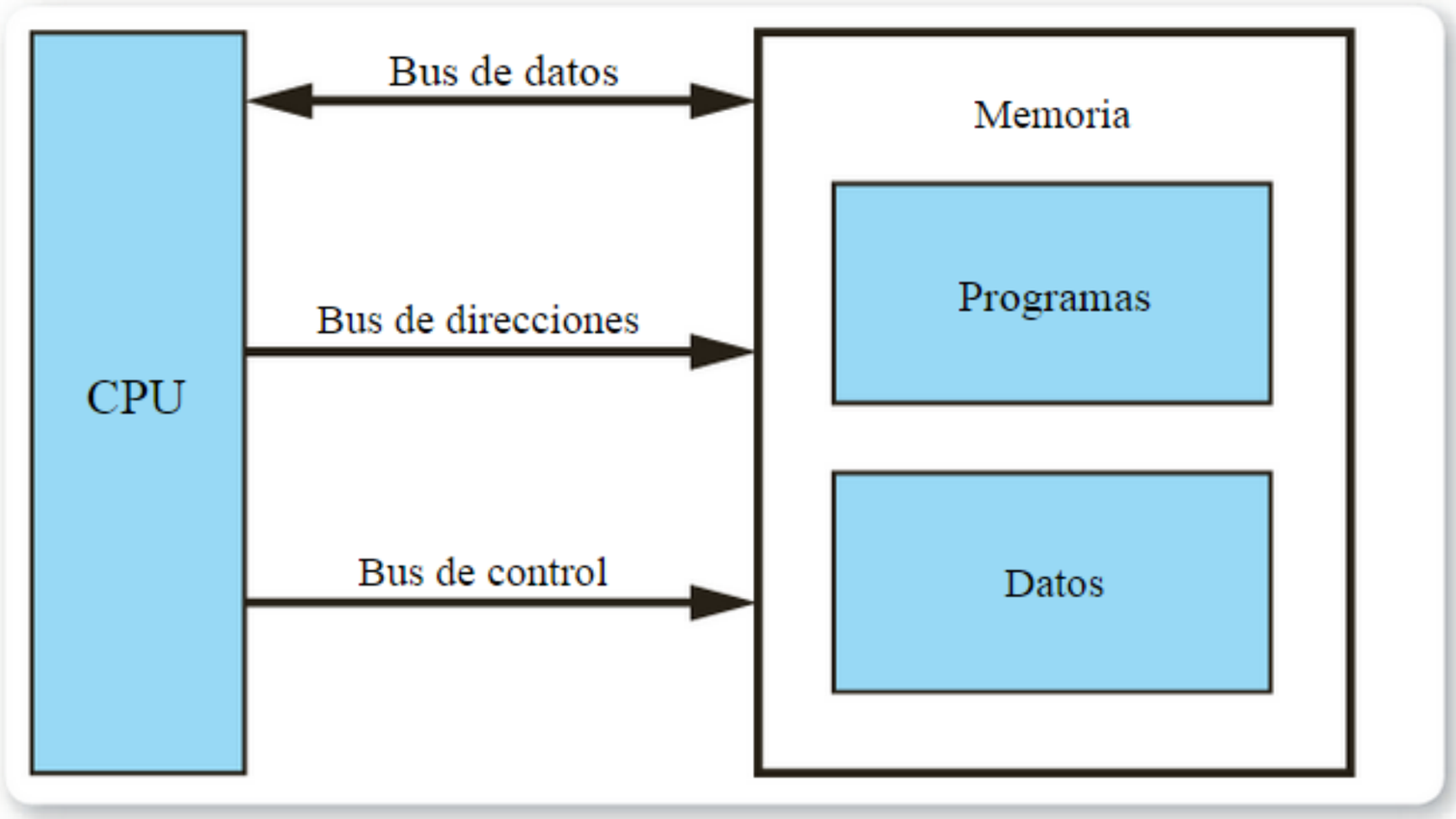


Figura 12. Diagrama de la arquitectura Harvard. Las memorias de instrucciones y de datos están separadas, y los buses también.

La arquitectura Harvard, desarrollada en 1970, pretende solucionar los problemas de velocidad de procesamiento que presentaba la arquitectura Von Neumann y, para ello, conecta la CPU con su memoria de programa por medio de un bus de direcciones, y con la memoria de datos mediante el bus de datos. Así, el ancho del bus de instrucciones no está limitado por el de datos, y el procesador puede recibir instrucciones por diferentes buses aprovechando el tiempo del ciclo de máquina.

Las ventajas de esta arquitectura tienen que ver, por un lado, con que el tamaño de las instrucciones no está relacionado con el de los datos, por lo que se optimizan para que cualquier instrucción ocupe una sola posición de memoria de programa. De esta forma se logra mayor velocidad y menor longitud de programa. Por otro lado, con que el tiempo de acceso a las instrucciones puede superponerse con el de los datos, para lograr una mayor velocidad en cada operación.

MICROPROCESADOR VS. MICROCONTROLADOR

Un **microprocesador** es un circuito integrado que procesa datos/instrucciones almacenados en la memoria; requiere de periféricos adicionales para su operación; tiene altas prestaciones, elevado costo y tamaño considerable. En cambio, un **microcontrolador** es un circuito integrado que contiene el microprocesador y los periféricos; tiene capacidades limitadas, bajo costo y reducido tamaño.

Decodificación de memoria

A medida que aumenta la complejidad de las aplicaciones, los sistemas microprocesadores deben acompañar este crecimiento y, en particular, ampliar su capacidad de memoria. Su incremento tiene dos objetivos: aumentar en la memoria tanto el tamaño de las palabras como su número. Por otra parte, un sistema basado en microprocesador tiene una capacidad de direccionamiento de acuerdo al bus de direcciones, y un ancho de palabra en función del bus de datos. Todos los buses (datos, direccionamiento y control) son específicos de cada modelo de microprocesador.

Intel fabrica microprocesadores de su familia y compatibles con variadas capacidades, desde dispositivos de bajo costo, a los que se les tiene que agregar memoria de programa (ROM) externa, hasta dispositivos que contienen la mayoría de los circuitos necesarios incluidos en el mismo circuito integrado. En general, todos los microprocesadores poseen la capacidad de direccionar una memoria externa para conectar memoria de programa o de datos, ampliar la memoria interna, adicionar dispositivos de I/O, entre otras necesidades, habilidades compartidas por los microcontroladores de esta empresa. Por ejemplo, la familia de microcontroladores Intel 8051 posee dos puertos para utilizar en operaciones de acceso a memoria externa: el **puerto 0**, como un bus multiplexado de datos/direcciones bajas ($A_0 - A_7$), y el **puerto 2**, como un bus de direcciones altas ($A_8 - A_{15}$). El puerto 0 aporta, en la primera parte del ciclo de una instrucción, la información correspondiente a la parte baja del bus de direcciones, mientras que, en la segunda parte, mantiene sus pines en estado de alta impedancia y, en la tercera, permite el flujo bidireccional de datos para operaciones de lectura/escritura entre el microprocesador y dispositivos externos conectados a él.

CAPACIDAD DE UN SISTEMA MICROPROCESADOR

La capacidad del sistema microprocesador depende de su habilidad de direccionamiento (2^n, "n" corresponde al número de bits del bus de direcciones); el ancho de palabra "m" (con "m" igual al número de bits del bus de datos y del bus de control) es específico del modelo de microprocesador.

La decodificación del bus de datos se realiza mediante una memoria temporal (LATCH) de 8 bits, activada mediante la señal **ALE** (*Address Latch Enable*). El puerto 2 se utiliza como una salida que mantiene la información correspondiente a la parte alta del bus de direcciones todo el tiempo que dura el ciclo de instrucción. Una vez decodificado el bus de direcciones (de 16 bits), el microprocesador puede acceder hasta 64 KB de información ($2^{16} = 65.536$). Sin embargo, la capacidad total del circuito es de 128 KB con 64 KB destinados a la memoria de programa y 64 KB a la memoria de datos.

Interfaz con bancos de memoria

Describiremos brevemente el procedimiento de diseño del mapa de memoria de un sistema basado en microprocesador. El **mapa de memoria** es un esquema que representa el lugar donde se almacena la información en un sistema microprocesador. Allí se observan las posiciones de memoria ocupadas (memoria ROM, memoria RAM, dispositivos de E/S) dentro de la zona direccionable por el microprocesador, así como las ubicaciones no utilizadas y que eventualmente podrían emplearse para algún propósito.

En síntesis, el procedimiento para el diseño del mapa de memoria de un sistema basado en microprocesador consiste en ejecutar los siguientes pasos:

1. Detallar las necesidades del sistema en cuanto a direccionamiento, anchura de palabra y tipo de memoria por utilizar (ROM/RAM).

2. Determinar los circuitos integrados de que se dispone (tanto en longitud como en anchura de palabras) y definir los que se necesitan.

3. Construir el mapa de memoria.

4. Determinar el tamaño de página (o banco) de memoria, y diseñar la tabla de direcciones y ocupación de cada circuito integrado.

5. Determinar la circuitería auxiliar necesaria para el control del circuito.

6. Dibujar el circuito completo de la memoria.

A modo de ejemplo, para comprender este procedimiento se realizará el diseño del mapa de memoria de un sistema basado en microprocesador Intel 8085, suponiendo que se necesitan 8 K × 8 de

memoria ROM y 4 K × 8 de memoria RAM, y que el laboratorio dispone de circuitos integrados **ROM de 2 K × 8** y circuitos integrados **RAM de 2 K × 8**. El mapa de memoria comienza a partir de la dirección $0, empezando por la ROM y colocando a continuación la RAM.

Entonces, el requerimiento original de memora es 8 K x 8 de ROM y 4 K x 8 de RAM. Luego, el N.° de CI de memoria ROM es el N.° de bits necesarios/N.° de bits por cada CI disponible, es decir, 8 K x 8/2 K x 8 o 4 CI de memoria ROM. Por otra parte, el N.° de CI de memoria RAM es el N.° de bits necesarios/N.° de bits por cada CI disponible, que será 4 K x 8/2 K x 8,2 CI de memoria RAM.

La configuración del mapa de memoria incluye tanto las memorias RAM como las ROM.

MAPA DE MEMORIA	
IC0 (2Kx8)	ROM
IC1 (2Kx8)	ROM
IC2 (2Kx8)	ROM
IC3 (2Kx8)	ROM
IC4 (2Kx8)	RAM
IC5 (2Kx8)	RAM
Libre	……
Libre	……

Tabla 3. Mapa de memoria para el sistema microprocesador basado en Intel 8085 tomado como ejemplo.

Luego, se debe determinar la cantidad de bits del bus de direcciones en función del tamaño total de la memoria.

8 K ROM + 4 K RAM = 12 K

Para direccionar 12 K de memoria, se necesitan 2^n = 12 K donde "n" es el número de bits de direcciones. Entonces:

2^{12} = 8.192 bytes y 2^{14} = 16.384 bytes (16 K), por lo que se necesitan n = 14 bits (A_0-A_{13}).

Bus de datos: 8 bits (D_0-D_7).

Tamaño de la página: 2 K, ya que todos los CI ROM y RAM disponibles son de 2 K (2048 bytes). $2^{11} = 2.048$ bytes, A_0-A_{10} en el bus de direcciones.

Bits de selección de página: se tienen 6 CI (4 CI ROM y 2 CI RAM). Entonces, $2^3 = 8$ páginas de las cuales se utilizan las seis primeras, y las dos restantes quedan libres. Los bits del bus de direcciones involucrados son A_{11}, A_{12} y A_{13}.

Además del microprocesador y las memorias ROM y RAM, es necesario emplear circuitos auxiliares. Como el número de páginas es 6 y el mínimo por controlar es 8 ($2^3 = 8$), deberemos utilizar un decodificador de 3 a 8, de forma que las líneas del bus de direcciones del sistema A_{11}, A_{12} y A_{13} se conectarán, respectivamente, a las entradas I_0, I_1 e I_2 del decodificador, y cada una de las salidas de O_0 a O_5 de este se conectarán a los chip select (CS) de cada uno de los circuitos integrados de IC0 a IC5 (memorias ROM y RAM). Como decodificador/demultiplexor de una entrada/ocho salidas, podría utilizar el CI 74F138, un chip que posee tres entradas (A_0-A_2) y ocho salidas negadas (Q_0-Q_7).

> E/S SUPONE CUALQUIER INTERCAMBIO DE INFORMACIÓN ENTRE UN MICROPROCESADOR Y UN PERIFÉRICO

En el momento de dibujar el esquema eléctrico completo, podemos seleccionar entre una gran cantidad de software CAD especializado para electrónica. **Eagle**, **Isis-Ares** y **Altium Designer** son algunas de las herramientas profesionales que facilitan el diseño del circuito eléctrico y su PCB. El esquema eléctrico debería

TABLA DE DIRECCIONES I

Para el CI IC0 (ROM), hay que direccionar 2048 posiciones de memoria (2 K x 8), por lo que necesitará 11 bits ($2^{11} = 2048_{10}$). A_0-A_{11} en el bus de direcciones. La primera posición de memoria es $\$0000_{16}$, y la última es 2047_{10} o $\$07FF_{16}$.

TABLA DE DIRECCIONES II

Para construir el resto de la tabla de direcciones, recordemos que la última posición de memoria para el CI IC0 (ROM) es $2047_{10} = 11111111111_2 = \$07FF_{16}$, además de las reglas de conversión entre sistemas de numeración analizadas antes en esta obra, y así sucesivamente con cada CI ROM/RAM considerado.

TABLA DE DIRECCIONES

A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0	DIRECCIÓN HEXADECIMAL	C.I.
0	0	0	0	0	0	0	0	0	0	0	0	0	0	$0000	IC0
0	0	0	1	1	1	1	1	1	1	1	1	1	1	$07FF	
0	0	1	0	0	0	0	0	0	0	0	0	0	0	$0800	IC1
0	0	1	1	1	1	1	1	1	1	1	1	1	1	$0FFF	
0	1	0	0	0	0	0	0	0	0	0	0	0	0	$1000	IC2
0	1	0	1	1	1	1	1	1	1	1	1	1	1	$17FF	
0	1	1	0	0	0	0	0	0	0	0	0	0	0	$1800	IC3
0	1	1	1	1	1	1	1	1	1	1	1	1	1	$1FFF	
1	0	0	0	0	0	0	0	0	0	0	0	0	0	$2000	IC4
1	0	0	1	1	1	1	1	1	1	1	1	1	1	$27FF	
1	0	1	0	0	0	0	0	0	0	0	0	0	0	$2800	IC5
1	0	1	1	1	1	1	1	1	1	1	1	1	1	$2FFF	

Tabla 4. Tabla de direcciones para el sistema microprocesador basado en Intel 8085 considerado como ejemplo.

incluir el microprocesador 8085, las memorias RAM y ROM así como los circuitos auxiliares.

En caso de diseñar un sistema microprocesador que, además de utilizar memorias RAM y ROM, incorpore unidades de entrada/salida, el mapa de memoria se inicia a partir de la dirección $0 ubicando la memoria ROM en primer lugar, la memoria RAM a continuación y, por último, las unidades de entrada/salida. El procedimiento de diseño es similar al descripto anteriormente.

Puertos paralelos de entrada/salida

El concepto de E/S involucra cualquier intercambio de información entre el microprocesador y un dispositivo externo o **periférico** específico. Si la información se dirige hacia el microprocesador, la operación se conoce como **operación de entrada** y, si lo hace desde el microprocesador, tenemos una **operación de salida**. Los **puertos** son los dispositivos que facilitan las operaciones de E/S entre el microprocesador y el mundo exterior, y resultan indispensables en cualquier sistema microprocesador, mientras que la **interfaz** es responsable del proceso de interconexión entre un dispositivo de E/S y el sistema microprocesador. Los puertos de E/S (puertos I/O o adaptadores periféricos de interfaz), la memoria y el microprocesador se comunican entre sí mediante los buses de datos, de direcciones y de control. Un puerto de entrada facilita la introducción de información en el sistema microprocesador procedente desde teclados, interruptores, conversores A/D, entre otros periféricos

ACCESO DIRECTO A MEMORIA

Si bien en un sistema microprocesador se realizan todas las operaciones en el microprocesador, existen instrucciones de programa que permiten a un dispositivo periférico acceder a la memoria principal sin tener que pasar por el microprocesador, liberando el uso de los buses de datos y direcciones. Esta forma de acceso se conoce como **acceso directo a memoria** (**DMA**, **Direct Memory Access**).

de entrada, mientras que un puerto de salida le permite, al sistema microprocesador, enviar datos hacia dispositivos de salida, como display, motores paso a paso, impresoras, etcétera.

Interfaz básica de salida

Para construir un puerto o interfaz de E/S, podemos utilizar compuertas, flip-flop, registros, codificadores, decodificadores; mediante circuitos integrados especializados como la **interfaz periférica programable** (**PPI**) o el **controlador programable de interrupciones** (**PIC**), y por medio de puertos E/S programables que utilizan alta escala de integración diseñados específicamente para operar con E/S. Estos circuitos presentan una serie de ventajas, como la facilidad de conexión con los buses del sistema microprocesador, debido a que poseen líneas de adaptación directa con las salidas del microprocesador y E/S programables, lo que permite adaptarlos con facilidad a los circuitos externos mediante programación.

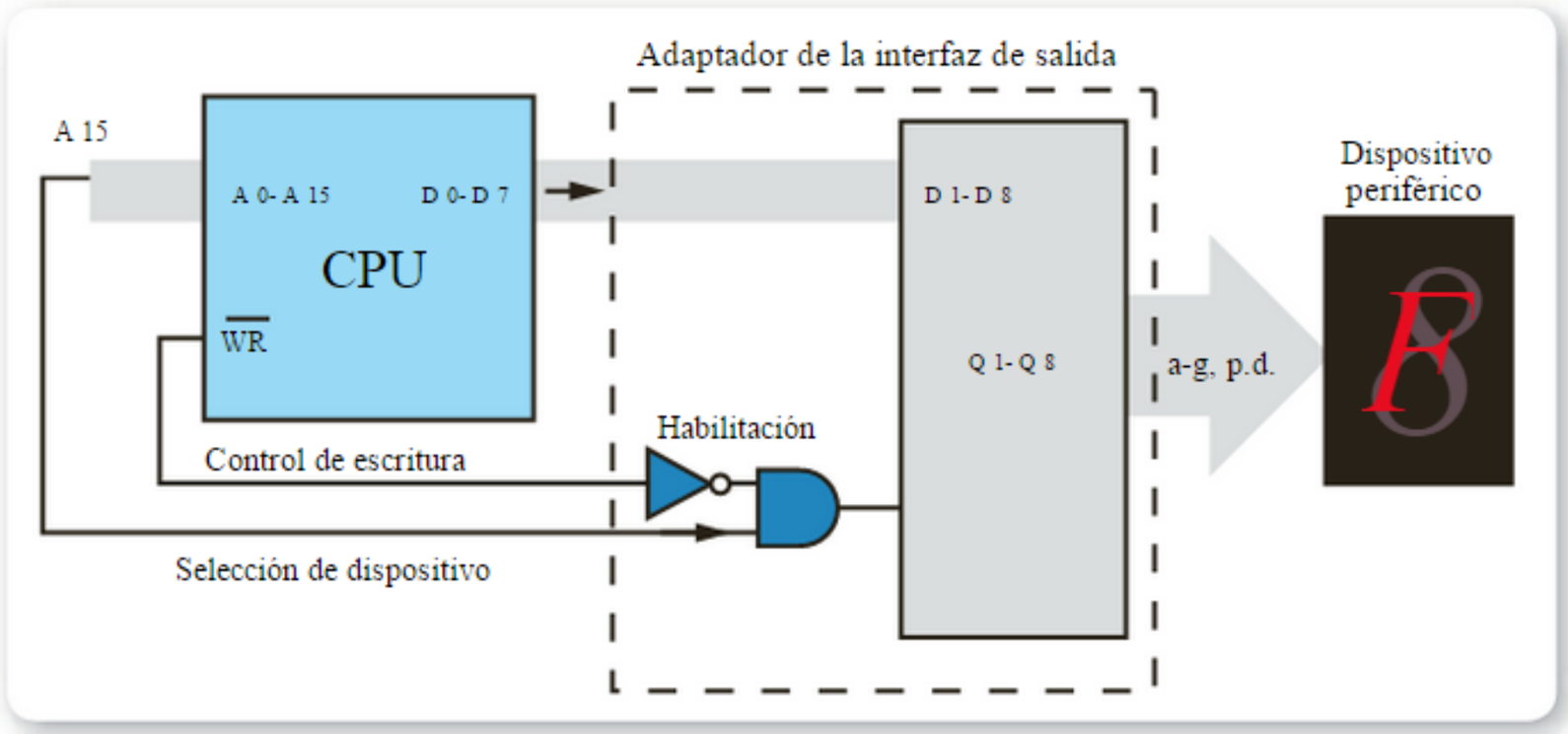


Figura 13. Diagrama básico de un puerto de salida con registro de datos para interconectar un microprocesador con un display.

TRANSMISIÓN DE DATOS EN SERIE

Se utiliza una línea que transmite los bits en forma de palabras o caracteres de manera secuencial, uno a continuación de otro, y se emplea con dispositivos alejados entre sí. Además de la línea de datos, requiere enviar una señal de reloj para sincronizar la transferencia para indicar el inicio y el fin de cada bit.

Las interfaces de E/S programables se clasifican en dedicadas, de propósito general y universales. Las **interfaces dedicadas** se especializan en realizar una función concreta, por ejemplo el control de interrupciones, la temporización de eventos y el control de periféricos, entre otros. Las **interfaces de propósito general** se pueden adaptar a una gran variedad de aplicaciones, como por ejemplo, la transmisión de datos en serie y la transmisión de datos en paralelo. Finalmente, una **interfaz universal** es un circuito integrado que, mediante un microprocesador y otros circuitos electrónicos ubicados en su interior, controla la realización de tareas sumamente complejas.

> LAS INTERFACES DEDICADAS SE ESPECIALIZAN EN REALIZAR UNA FUNCIÓN CONCRETA

En los sistemas microprocesadores prácticos, la salida del microprocesador no va conectada directamente a un dispositivo periférico, sino que se conecta a un dispositivo de memoria que almacena el dato para el periférico, y se conoce como **bloque adaptador de entrada** o **bloque adaptador de salida** . Un adaptador de E/S, por lo general, tiene características distintas de una memoria.

Interfaz básica de entrada

Los microprocesadores utilizan instrucciones para transferir los datos desde los puertos de E/S o hacia ellos. En el caso del microprocesador INTEL 8085, las instrucciones se denominan **IN** y **OUT**, y se identifican mediante los códigos de operación **DB** y **D3** en lenguaje de máquina. IN es una instrucción de dos bytes que mueve un

TRANSMISIÓN DE DATOS EN PARALELO

Los bits que configuran la palabra de datos se envían/reciben simultáneamente por varias líneas de un bit cada una. La velocidad de comunicación es en bytes/seg. Entre cada byte existe un intervalo de tiempo y es intrínsecamente más rápida que la serie. El microprocesador la utiliza para comunicaciones internas y requiere una señal de reloj para indicar al receptor cuándo la palabra está disponible.

byte de datos desde el puerto de entrada hacia el registro acumulador. El segundo byte de la instrucción contiene la dirección del puerto de entrada. Existen hasta 256 puertos de entrada posibles para el 8085. Durante la ejecución de la instrucción IN, se coloca la dirección del dispositivo de 8 bits en el bus de direcciones y, con la ayuda de un decodificador externo, se decodifica la dirección del dispositivo a efectos de seleccionar solo el dispositivo de entrada deseado.

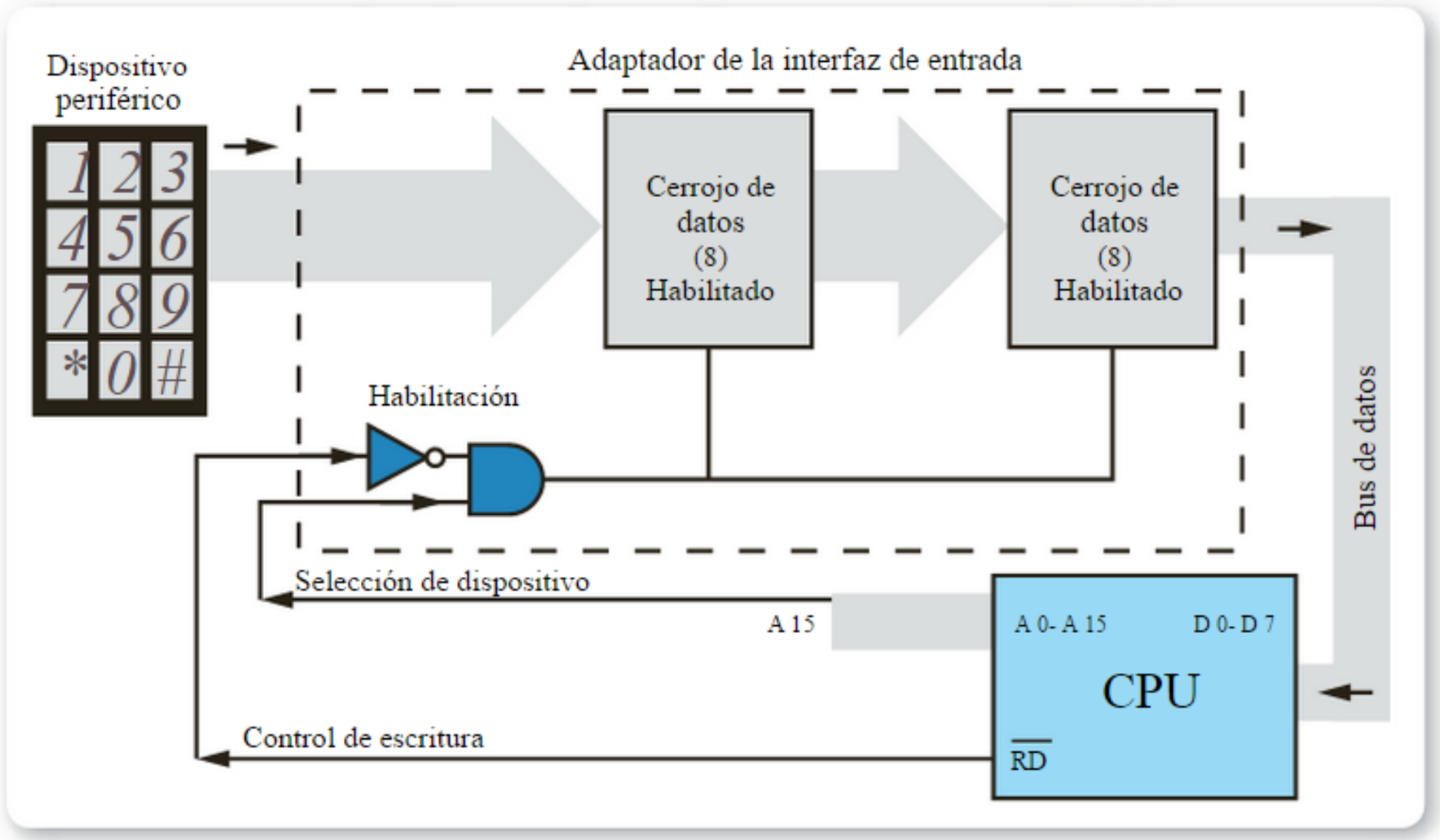


Figura 14. Diagrama básico de un puerto de entrada para interconectar un microprocesador con un teclado.

La instrucción OUT tiene dos bytes y mueve un byte de datos desde el acumulador hasta el puerto de salida. El segundo byte de datos contiene la dirección de 8 bits del puerto de salida. Así, el microprocesador INTEL 8085 puede direccionar hasta 256 puertos de salida ($2^8 = 256$).

FUNCIONES DE UN PUERTO

Un puerto o dispositivo de interfaz de E/S tiene que **identificar direcciones** para establecer una conexión con los buses de datos y de control del sistema microprocesador, **interpretar instrucciones**, **adaptar físicamente** el sistema microprocesador a los requerimientos del periférico y **temporizar** la transferencia de información entre el microprocesador y el periférico.

La técnica de E/S de mapa de memoria utiliza las direcciones normales de memoria. Esta es la técnica más común y puede utilizarse con cualquier microprocesador, mientras que la técnica de E/S aisladas puede utilizarse solo con microprocesadores que tengan instrucciones IN y OUT separadas, entradas/salidas especiales, y salidas de control de lectura y escritura.

Direccionamiento y mapa de puertos

Las distintas unidades que componen un sistema microprocesador (memorias RAM y ROM, dispositivos de entrada/salida) se direccionan de acuerdo al mapa de memoria diseñado. Si en el bus de direcciones del microprocesador aparece una dirección específica, solamente una palabra de memoria debe ser accedida. Esto implica que, en ese momento, solo una unidad de memoria tiene que estar conectada al bus. Por este motivo, cada unidad que se conecta a un bus del microprocesador posee una entrada de selección de chip (en inglés, *chip select*, CS) que solo cuando se activa permite que esa unidad se conecte al bus. Si CS no está habilitada, la salida de la unidad permanece en estado de alta impedancia. Los circuitos decodificadores que reciben las líneas correspondientes del bus de direcciones activan las entradas CS.

Una salida puede tener uno de tres estados: 0 lógico, 1 lógico y de alta impedancia. Una salida con tres estados posibles se conoce como **salida de tres estados** o **salida triestado**. Los dispositivos de tres estados tienen una entrada extra, conocida como **entrada de deshabilitación OE** (en inglés, *enable*) para establecer las salidas del dispositivo en el estado de alta impedancia. Las salidas de tres estados se implementan en circuitos, como microprocesadores, memorias y periféricos.

ESTADO DE ALTA IMPEDANCIA

Las señales lógicas se componen de dos estados normales, alto (1) y bajo (0). Sin embargo, algunas salidas tienen un tercer estado eléctrico que no es un estado lógico y que se denomina **estado de alta impedancia**. En este estado, la salida se comporta como si aún no estuviera conectada al circuito, permitiendo que múltiples circuitos compartan la misma línea de salida.

Decodificación de puertos

INTEL ha diseñado una serie de dispositivos periféricos especializados para realizar transmisiones de datos y sincronizar transmisor-receptor, que pueden ser acoplados directamente al bus del microprocesador 8085. Cada uno de ellos posee determinadas características que los hacen aptos para algunas aplicaciones. Por ejemplo, el periférico programable **PPI 8255** para uso general es capaz de trabajar directamente con el microprocesador 8085 y posee tres puertos de E/S de 8 bits cada uno.

El PPI 8255 permite tres modos de direccionamiento distintos. El **Modo 0** proporciona un método de E/S básico por cada uno de los tres puertos. Se puede escribir un dato en el puerto o se puede leer el dato en un puerto. Técnicamente es posible configurar dos puertos de 8 bits y dos de 4 bits, cualquier puerto puede ser E/S; las salidas contienen flip-flop, mientras que las entradas son directas, y existen 16 posibles configuraciones de E/S.

En el **Modo 1** se realizan transferencias E/S hacia un puerto o desde él. Tanto el puerto A como el puerto B pueden definirse individualmente como E/S y soportan una multiplicidad de aplicaciones. Existen dos grupos de trabajo: A y B, donde cada grupo posee 8 bits de datos y un puerto adicional de 4 bits de control de datos. Los puertos A y B (de 8 bits) pueden ser ambos de E/S mediante flip-flop en ambos casos. El puerto de 4 bits se emplea para control y estado de los puertos de 8 bits.

Finalmente, el **Modo 2** proporciona un medio de comunicación con periféricos mediante un bus de 8 bits para transmitir y recibir datos (**bus bidireccional**). También contiene la generación de interrupciones y funciones de actuación/inhibición. En cuanto a las características de este modo de funcionamiento, solo se emplea el grupo A; el puerto A es el bus bidireccional de 8 bits, el puerto C emplea 5 bits de control, y las E/S utilizan flip-flop.

MODOS DE OPERACIÓN 8285

MODO	CARACTERÍSTICAS
0	Los puertos A, B, C pueden funcionar como E/S. No se dispone de líneas de control auxiliares para interconectarse con los periféricos de E/S.

MODO	CARACTERÍSTICAS
1	**Puertos A y B: E/S. Puerto C:** control de transferencia de datos entre los periféricos y el 8255.
2	**Puerto A:** bidireccional (E/S). Puerto C: PC3-PC7: controlan el intercambio con los periféricos, y el resto puede actuar como E/S o como líneas auxiliares del puerto B.

Tabla 5. El **PPI 8255** admite tres modos diferentes de funcionamiento, programables con una palabra de control enviada desde el microprocesador al chip mediante el bus de datos.

Interfaz periférica programable (PPI)

Entre los circuitos de apoyo para el microprocesador 8085 se incluyen memorias ROM y RAM, además de circuitos de interfaz de entrada/salida. A continuación, describiremos uno de los circuitos de soporte más útiles y flexibles, el integrado 8255 (**interfaz programable de periféricos** o **PPI**), que contiene tres puertos programables de entrada/salida de 8 bits, denominados A, B y C, programables de manera independiente, para que actúen como entradas, salidas o en forma bidireccional.

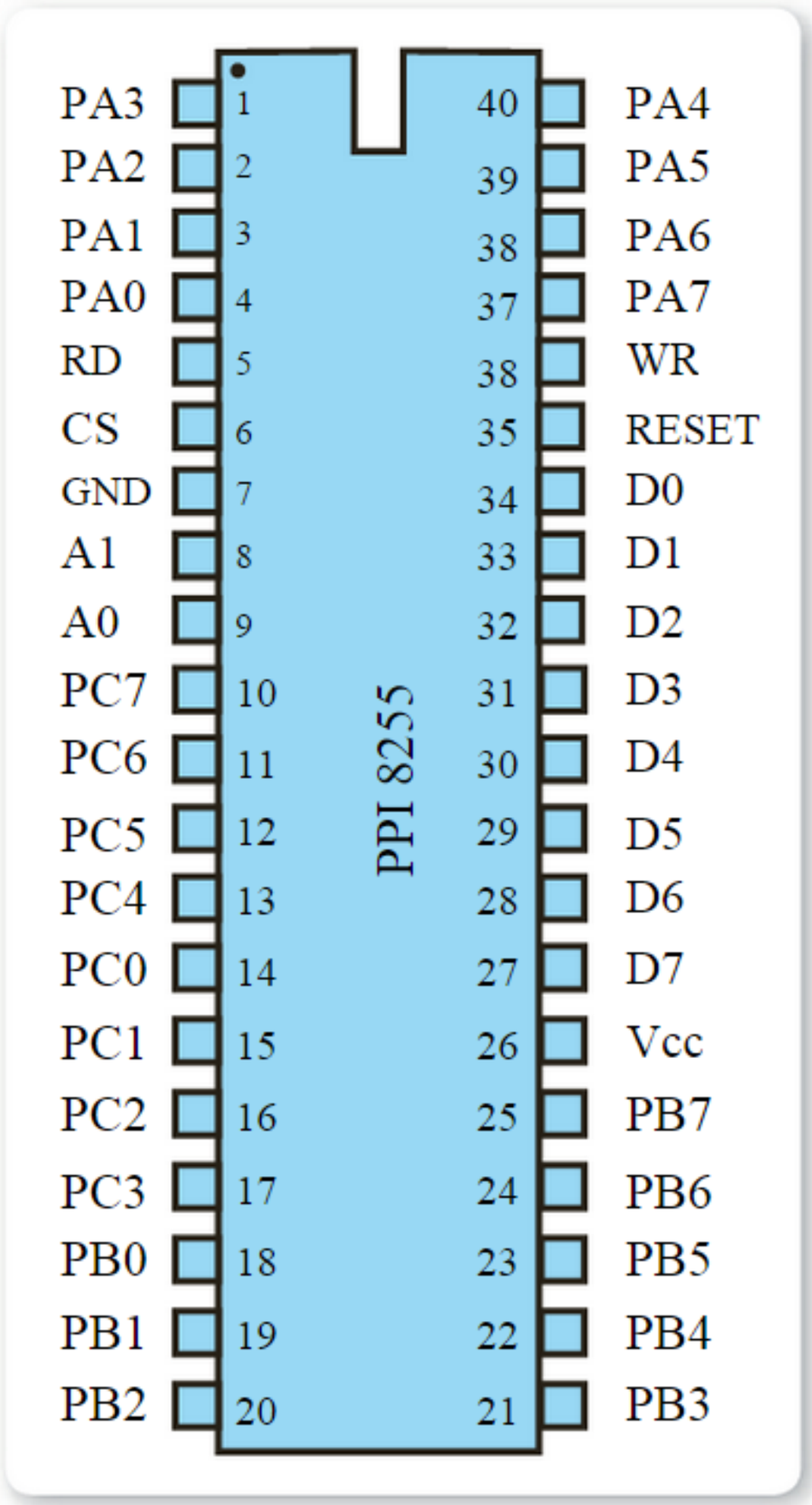


Figura 15. Descripción de pines del integrado **8255** (PPI).

La selección del modo de operación se realiza con una palabra de control de 8 bits que es enviada mediante el bus de datos, desde el microprocesador hacia el 8255. De los 40 pines del 8255, 24 son líneas de entrada/salida: PA0-PA7 (puerto A), PB0-PB7 (puerto B) y los grupos de líneas PC7-PC4 y PC3-PC0 (puerto C), que pueden funcionar como dos puertos individuales de cuatro bits o en combinación con los puertos A y B. Además, el 8255 dispone de cuatro registros: tres de ellos almacenan la información que ingresa/sale por los puertos A, B y C, mientras que el cuarto, **registro de control**, realiza las funciones de control. Programando este último puerto, se configuran los puertos y se define el funcionamiento general del integrado 8255.

Otros pines importantes del 8255 son:

- El **pin 36** (**write**), que se activa en bajo para que el microprocesador realice una operación de escritura sobre el 8255.
- El **pin 5** (**read**) también se activa en bajo para que el microprocesador realice una operación de lectura sobre el 8255.
- El **pin 6** (**chip select**) se activa en bajo y permite la selección del circuito integrado: si está en 0, permite que el 8255 se comunique con el microprocesador, mientras que en 1 no lo permite.
- El **pin 35** (**RESET**) se activa en alto e inicializa el 8255, y borra (pone en 0) todos los registros internos. Las líneas de datos que comunican al 8255 con el bus bidireccional de datos del microprocesador se encuentran en los pines 27 a 40.
- Los **pines 8 y 9** en combinación con los pines read y write permiten seleccionar el puerto o registro del 8255 sobre el cual se va a realizar la operación de escritura o lectura.

RESUMEN

En este capítulo, iniciamos el estudio de las características eléctricas de un microprocesador y analizamos la interfaz entre el microprocesador y las memorias ROM y RAM. Luego, aprendimos sobre el direcciona-miento de memoria y realizamos un examen de los puertos paralelos de entrada/salida. En síntesis, nos introducimos en la comprensión de la arquitectura y las características de las distintas interfaces entre el microprocesador-memoria ROM, microprocesador-memoria RAM y microprocesador-dispositivos de E/S.

Actividades

TEST DE AUTOEVALUACIÓN

1 Desarrolle el esquema en bloques general de un microprocesador y analice cómo interactúan.

2 Exprese cómo funcionan las memorias RAM y compare una RAM estática con una RAM dinámica.

3 ¿Cómo se denomina la característica de una memoria ROM relacionada con el tiempo que se tarda en decodificar una dirección y acceder a una posición específica?

4 ¿Qué tipo de memoria RAM (estática o dinámica) es más fácil de conectar a un microprocesador mediante una interfaz?

5 ¿Cómo se denomina la interconexión entre las partes en un sistema basado en microprocesadores: memoria, direccionamiento, interfaz, o bus de datos?

EJERCICIOS PRÁCTICOS

1 ¿Cuántas entradas de dirección necesita una memoria ROM de 4 K para decodificar las 4096 direcciones de memoria?

2 Identifique las características eléctricas más importantes de los distintos circuitos integrados indicados a lo largo del capítulo. Utilice las hojas de datos de los fabricantes. Por medio de internet, acceda a los sitios web de las empresas.

3 Indique las líneas de control que posee el microprocesador Intel 8085.

4 Describa los modos de funcionamiento del PPI 8255.

5 Si dispone de memorias RAM y ROM de 4 K x 8 en lugar de 2 K x 8, ¿cómo reconfiguraría el mapa de memoria de la tabla 3?

PROFESOR EN LÍNEA

Si tiene alguna consulta técnica relacionada con el contenido, puede contactarse con nuestros expertos: profesor@redusers.com.

*05

Programación en lenguaje ensamblador

Hemos referenciado los sistemas microprocesadores desde un enfoque hardware. Ahora, se presentarán los conceptos necesarios para comprender los fundamentos de la programación en lenguaje ensamblador y desarrollar algunas aplicaciones básicas para microprocesadores de 8 y 16 bits, que facilitarán que se pueda comprender la programación de un microprocesador.

Servicio de atención al lector: usershop@redusers.com

Programación en lenguaje ensamblador

Es importante recordar que los microprocesadores más poderosos, de 32 y 64 bits, comparten la misma arquitectura (Von Neumann) que los microprocesadores de 8 y 16 bits, objeto de este libro.

El software se refiere al conjunto de instrucciones por medio de las cuales se le indica a un microprocesador qué acciones debe realizar. El grupo de instrucciones que un microprocesador reconoce se denomina **conjunto o set de instrucciones** , mientras que se llama **programa** al conjunto de instrucciones necesarias para ejecutar una determinada acción. Un microprocesador funciona sobre la base del sistema de numeración binario, y el **código máquina** es el conjunto de instrucciones escritas en ese sistema. Esto es así dado que escribir programas directamente en sistema hexadecimal o en sistema binario resulta problemático, ya que la probabilidad de cometer errores durante la escritura es muy alta, con lo que se dificulta el desarrollo de un programa.

El **código mnemotécnico** conocido como **lenguaje ensamblador** , que desarrollaremos en este capítulo, es una alternativa de escritura de programas que permite mejorar la efectividad del código escrito al representar, de manera abreviada mediante un nombre simbólico, la acción que realiza una instrucción. Permite reducir la probabilidad de cometer errores durante su escritura y facilita la comprensión. Para realizar la conversión del programa escrito en lenguaje ensamblador a código de máquina, el único que interpreta el microprocesador, se recurre a programas específicos denominados **compiladores para lenguaje ensamblador** . El programa escrito en ensamblador

OPERANDOS EN ENSAMBLADOR

Para expresar operaciones, el lenguaje ensamblador facilita diferentes operadores: + suma, - resta, * multiplicación, / división, ** potenciación, **.NOT** complementación, **.SHR** desplazamiento a la derecha, **.SHL** desplazamiento a la izquierda, **.AND** operación lógica AND (Y), **.OR** operación lógica OR (O), **.XOR** operación OR Exclusiva, **.EQ** igual, **.GT** mayor que y **.LT** menor que.

se denomina **programa fuente**, con extensión **.ASM**, y mediante el compilador se obtiene el código máquina (valores binarios), con extensión **.OBJ**, que será ejecutado por el microprocesador una vez almacenado en la memoria del sistema.

El microprocesador Intel 8085 utiliza un formato que divide a cada línea del lenguaje ensamblador en cuatro campos: de etiqueta, de código de operación, de operandos y de comentarios.

El campo de **etiqueta**, opcional, es solamente identificativo y representa el nombre de la línea.

El campo de **código de operación**, obligatorio, contiene el nemotécnico del código de operación de la instrucción del 8085 por ejecutar.

El campo de **operandos**, a veces denominado **argumento**, da el dato con el que va a operar el código de operación especificado, y su sintaxis varía según la instrucción utilizada. Puede llevar uno o más operandos separados por una coma (,). En primer lugar, se escribe el operando destino y, en segundo lugar, el operando origen. Los operandos pueden ser expresados mediante nombres simbólicos, nombres de registros, valores numéricos, expresiones aritméticas, etcétera.

> MICROPROCESADORES DE 32 Y 64 BITS TIENEN LA MISMA ARQUITECTURA QUE LOS DE 8 Y 16 BITS

El campo de **comentario**, opcional, contiene información útil para explicar la función de la instrucción, por lo que su uso es de gran importancia para documentar un programa.

En el lenguaje ensamblador también se emplean las líneas completas para comentarios: comienzan necesariamente por el símbolo (;) y por lo general son situadas al principio del programa o al principio de cada módulo o subrutina. Estas líneas facilitan la documentación de subrutinas y programas, incluyendo en ellas el nombre, la función, las precondiciones, etcétera. Siempre el final del programa debe estar indicado con la instrucción especial del lenguaje ensamblador (**pseudoinstrucción** o **directiva de ensamblador**) **END**.

Modelo de programación

El **modelo de programación** resume los registros disponibles en un microprocesador para ser utilizados por el programador.

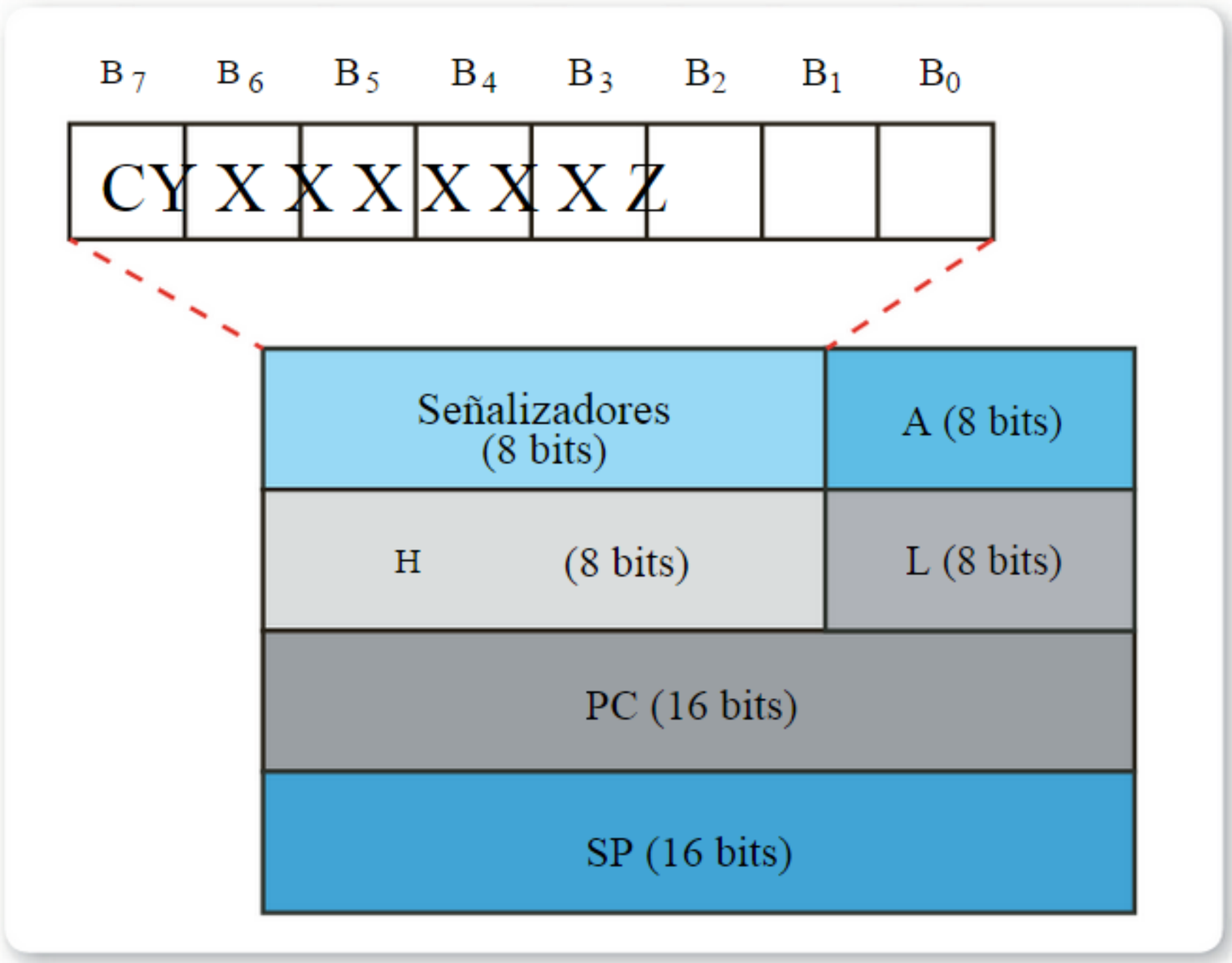


Figura 1. Modelo de programación para un microprocesador genérico aplicable al **Intel 8085**.

El registro de estatus de 8 bits contiene distintos señalizadores, aunque un microprocesador real incluye algunos registros adicionales. El registro acumulador, A, de propósito general y de 8 bits se encuentra en la misma fila del modelo. Los registros de datos/direcciones multipropósito de 8 bits cada uno, H y L, se pueden utilizar en forma separada como registros de datos de propósito general o como registro par, o par HL, utilizado para apuntar hacia direcciones. Finalmente, se tienen dos registros dedicados de 16 bits cada uno, el contador de programa, PC, que señala la siguiente dirección por ejecutar en la CPU y el puntero de pila, SP, que contiene la dirección superior de la pila

PROCESO DE PROGRAMACIÓN

Implica desarrollar una serie de pasos para solucionar un problema: 1) definir y comprender el problema; 2) encontrar un algoritmo que resuelva el problema y establecer estructuras de datos, entradas necesarias, flujo de control y salidas deseadas; 3) codificar el algoritmo en lenguaje ensamblador y verificar; 4) realizar pruebas de testeo; 5) documentar el proceso desarrollado.

(la pila está localizada en la memoria RAM). El programador también debe conocer la arquitectura general del microprocesador por utilizar y comprender el mapa de memoria de este.

Registros de funciones especiales (SFR)

En los microprocesadores, los registros de propósito específico, o especiales, (**SFR**) almacenan información específica sobre el estado del sistema. Por ejemplo, el puntero de pila o el registro de estado en contraposición con los registros de propósito general (en inglés, *General Purpose Registers* o **GPR**), que permiten almacenar tanto datos como direcciones y se agrupan en bancos de registros.

En los microcontroladores, los periféricos se encuentran incluidos en el mismo circuito integrado con la CPU y otros dispositivos. Por ejemplo, en el microcontrolador de 8 bits Intel 8051, el primero de esta empresa, los SFR son registros destinados en su mayoría a controlar los periféricos integrados en el 8051. Por medio de distintos SFR se puede acceder a los puertos de entrada/salida del 8051, leer o escribir en el puerto serie del microcontrolador, controlar los temporizadores y contadores, configurar el sistema de interrupciones, entre otras facilidades.

> LOS REGISTROS ESPECIALES GUARDAN INFORMACIÓN SOBRE EL ESTADO DEL SISTEMA

Estado inicial de los registros de la CPU

Al energizar un microprocesador, tanto sus registros como sus flip flop internos adoptan valores o estados aleatorios. Esto ocasiona la configuración de un estado inicial desconocido que puede comprometer el funcionamiento del microprocesador y ocasionar que ejecute un programa con errores (por ejemplo, por direccionar una posición de memoria inexistente), que traerá como consecuencia el congelamiento del microprocesador.

Para evitar esto, se obliga al microprocesador a tomar un estado inicial conocido mediante la **señal de reset**. La señal de reset se implementa mediante un circuito eléctrico específico que se encuentra explicado en el **capítulo 4**.

Bancos de registros

Se entiende por **registro** un tipo de memoria de alta velocidad y baja capacidad, integrada al microprocesador, que permite almacenar temporariamente valores muy utilizados, como por ejemplo en las operaciones matemáticas.

Por lo general, los registros se implementan mediante un **banco de registros**, un término que también se utiliza para referirse al grupo de registros que se pueden indexar directamente como operandos de una instrucción, de acuerdo a lo definido en su propio conjunto de instrucciones.

Registros con bits direccionables

Los registros del microprocesador se emplean para controlar instrucciones en ejecución, manejar el direccionamiento de la memoria y proporcionar capacidad aritmética. Los registros son direccionables por medio de un nombre y de acuerdo a diferentes modos de direccionamiento.

Modos de direccionamiento

El **modo de direccionamiento** es la técnica empleada para buscar el operando deseado durante la ejecución de una instrucción. En el microprocesador Intel 8085, tenemos cinco modos de direccionamiento: inmediato, directo, indirecto de registro, implícito y de registro.

- Direccionamiento **inmediato**: las instrucciones tienen los datos inmediatamente a continuación del código de operación en la memoria del programa. Por ejemplo, para la instrucción **ADDI** (suma inmediata), el microprocesador busca el código de operación en la memoria del programa. Luego de decodificar la instrucción, localiza el dato inmediato en la siguiente posición consecutiva de la memoria del programa, después del código de operación. Así, el dato inmediato se suma al contenido del acumulador, y el resultado se almacena en el acumulador.
- Direccionamiento **implícito**: este modo de direccionamiento también es conocido como implícito, implicado o inherente, y

está determinado por la función de la instrucción. La instrucción **STC**, que pone a 1 el señalizador de arrastre, se relaciona con el señalizador de arrastre solamente y no con otros registros o posiciones de memoria.

- Direccionamiento **por registro** : se especifican la fuente del operando y la operación. Para la instrucción **ADD C** , sumar el registro C con el acumulador, una vez ejecutada la instrucción el resultado se deposita en el acumulador. Ambos operandos están localizados en registros internos del microprocesador. Este modo de direccionamiento es muy eficiente, ya que las instrucciones utilizan un espacio de la memoria de programa de 1 byte y se ejecutan rápido, dado que no tienen que buscar operandos en la memoria.
- Direccionamiento **directo**: las instrucciones se especifican mediante el formato de instrucción de 3 bytes. El primero contiene el código de operación para la instrucción de direccionamiento directo, el segundo contiene el byte de orden inferior de la dirección del operando, y el tercero contiene el byte de orden superior de la dirección del operando. Para la instrucción **LDA**, cargar A directo, el código de operación es 3AH. Los dos siguientes bytes de la memoria son acoplados por el microprocesador en una dirección de 16 bits. La CPU accede a esta dirección de la memoria de datos, y su contenido se carga en el acumulador. Este tipo de instrucciones requiere de gran espacio en la memoria de programa, y de tiempo de ejecución alto, por los numerosos accesos a la memoria que son necesarios.

> EL MICROPROCESADOR INTEL 8085 TIENE CINCO MODOS DE DIRECCIONAMIENTO

- Direccionamiento **indirecto de registro**: las instrucciones referencian la memoria mediante el contenido de un registro par, que señala la dirección del operando. La instrucción **ADDM**, sumar memoria, suma el contenido del acumulador con el contenido de la posición de memoria indicado por la dirección del registro HL de la CPU.

Los modos de direccionamiento **relativo**, **absoluto**, **indexado** y **de la pila** son otros modos de carácter general que no se utilizan en el microprocesador Intel 8085.

Conjunto de instrucciones

Los sistemas basados en microprocesadores necesitan software para su funcionamiento. Este software consiste en programas desarrollados y luego almacenados en las memorias permanentes del sistema. Allí son buscados por el microprocesador para desarrollar todas las funciones y tareas deseadas.

Instrucciones de transferencia de datos

Permiten la transferencia de datos entre registros o entre las posiciones de memoria y los registros. Estas instrucciones incluyen transferencias, cargas, almacenamientos e intercambios, y también, las instrucciones de movimiento **MOV** de byte (8 bits) o de Word (palabra, 16 bits). El nemónico de la instrucción es MOV, el significado **movimiento de datos** , y el formato **MOV D (destino) , F (fuente)** sin afectar los flags o bits indicadores de estado. De acuerdo a la descripción de Intel para la instrucción **MOV A,M** el código op es 7EH, transfiere datos de una posición de memoria al registro A; la posición de memoria se identifica por el contenido del registro par HL, utiliza dos ciclos de máquina y siete estados T para su ejecución, mientras que el direccionamiento es indirecto de registro.

Instrucciones aritméticas

Son las instrucciones que realizan operaciones de tipo aritmético, como sumas, restas, incrementos o decrementos, en datos de registros o memoria. De acuerdo a la descripción de Intel para **ADD C**, suma de

GRUPOS DE INSTRUCCIONES-INTEL 8085

En un microprocesador Intel 8085, las instrucciones de programa se almacenan en grupos de 8 bits (bytes) en la memoria de programa mediante instrucciones de 1, 2 y 3 bytes, donde el primer byte de la instrucción es el código de operación que especifica cuál de las 200 instrucciones se va a ejecutar. Las categorías funcionales que utiliza Intel son las siguientes: transferencia de datos, aritméticas, lógicas, de bifurcación y de pila, E/S y control de máquina.

registro, el código de op es 81H, suma el contenido del registro C al contenido del acumulador y utiliza direccionamiento de registro.

> LOS SISTEMAS BASADOS EN UN MICROPROCESADOR NECESITAN SOFTWARE PARA FUNCIONAR

Instrucciones lógicas

Permiten realizar operaciones AND, OR, XOR, comparaciones, desplazamientos circulares y complementación de datos en registros o entre la memoria y un registro.

De acuerdo a la descripción de Intel para la instrucción de 1 byte **AND M**, AND memoria, el código op es A6H, realiza la operación AND del contenido del acumulador con el contenido de una posición de la memoria de datos señalada por el registro par HL.

Instrucciones de manejo de bits

Intel 8086 es un microprocesador de 16 bits en cuanto a su estructura y sus conexiones externas, mientras que el 8088 es un procesador de 8 bits que internamente es muy similar al 8086; se diferencian casi exclusivamente por el tamaño del bus de datos externo.

Ambos fueron desarrollados a partir del microprocesador Intel 8085, son compatibles con este y suponen varias mejoras, ya que los microprocesadores 8086/8088 poseen una unidad de procesamiento central, CPU o MPU, de 16 bits frente a los 8 bits de la misma unidad en un microprocesador 8085.

Las instrucciones de manejo de bits se utilizan en el microprocesador Intel 8088 y permiten el movimiento de datos entre registros, y entre registros y memoria externa.

PILA, E/S Y CONTROLDEMÁQUINA

El microprocesador Intel 8085 tiene su propio juego de instrucciones para mantener la pila, leer los puertos de entrada, escribir en los puertos de salida, inicializar y leer máscaras de interrupción, e inicializar y borrar señalizadores. Ejemplos de este grupo de instrucciones en ensamblador son **PUSH rp** (introducir) y **PUSH PSW** (introducir la palabra de estatus del microprocesador).

EL DIAGRAMA DE FLUJO SE DEBE DESARROLLAR EN DOS NIVELES: UNO FUNCIONAL Y OTRO DETALLADO

Incluyen las instrucciones IN y OUT de Entrada/Salida para operar sobre un mapa de E/S independiente del mapa de memoria, y se tendrán instrucciones particulares para transferir información del mundo exterior o enviarla a este.

Otras instrucciones de manejo de bits son las de intercambio (**XCHG** o Exchange) entre fuente y destino, sin afectar la fuente como ocurre con la instrucción MOV, de traducción de byte **XLAT**. Estas se emplean tanto en la búsqueda de tablas como en las instrucciones específicas que se utilizan para cargar directamente los registros de segmento y otro registro de propósito general, en una sola operación y desde la memoria.

Instrucciones de flujo de programa

El lenguaje ensamblador incluye instrucciones de llamadas y retornos de subrutinas, llamadas y retornos de interrupciones, saltos condicionales de acuerdo al resultado de las comparaciones y saltos incondicionales que, en el microprocesador Intel 8085, se utilizan específicamente para saltos condicionales o incondicionales, llamadas, regresos y reinicializaciones. **JMP addr** (salto) es una de las principales instrucciones de este tipo, de 3 bytes y direccionamiento inmediato. La instrucción **CALL** es una operación especial de bifurcación capaz de realizar el salto desde el programa principal hacia una subrutina, y combina las operaciones de salto e introducción de datos en la pila.

Instrucciones de control del procesador

Estas instrucciones son muy necesarias cuando queremos actuar internamente sobre el microprocesador para provocar, por ejemplo, detenciones en la ejecución del programa.

Un ejemplo de estas son las **instrucciones de rearranque**, instrucciones de propósito especial que se utilizan en interrupciones; introducen el contenido del contador de programa en la pila y saltan a una de ocho direcciones predeterminadas. La instrucción **RST n** (rearrancar) ejecuta la tarea anterior y, mediante un código de tres bits en su código de op, especifica la dirección del salto.

Técnicas y herramientas para el desarrollo de programas

Al desarrollar un programa, es importante ejecutar un proceso que permita hacer más eficiente el esfuerzo. Luego de definir y analizar el problema, es conveniente realizar un diagrama de flujo de su solución antes de escribir el programa en lenguaje ensamblador, generar la versión del programa en lenguaje de máquina y depurarlo para luego documentar el programa.

Diagrama de flujo

Un diagrama de flujo se desarrolla en dos niveles. En el **diagrama de flujo funcional** del problema, se representa gráficamente el procedimiento general para resolverlo. La ventaja reside en que se puede utilizar para cualquier microprocesador, y el inconveniente del esquema está en que no contiene todos los detalles necesarios para escribir directamente el programa en lenguaje ensamblador. Por el contrario, en el **diagrama de flujo detallado** cada bloque corresponde a una operación en el microprocesador, por lo que depende del microprocesador que específicamente se esté programando.

Por ejemplo, para sumar dos números, el diagrama de flujo funcional se iniciaría con el bloque correspondiente al inicio, luego indicaría cargar el número en el acumulador, sumar el segundo número, almacenar la suma en la memoria y, finalmente, detener la ejecución del programa.

En cambio, el diagrama de flujo detallado se corresponderá con el microprocesador específico que utilice para sumar los dos números, y de aquel surgirá el código en lenguaje ensamblador.

En las próximas páginas, desarrollaremos sencillos ejemplos que involucran tanto el código en lenguaje ensamblador como el diagrama de flujo detallado específico para el problema por resolver.

Subrutinas

Las subrutinas son herramientas que facilitan la programación y consisten en bloques de programación que se utilizan varias veces

en el mismo programa. De esta manera, el programa se desarrolla secuencialmente hasta que encuentra una subrutina, allí se desvía hacia la subrutina por ejecutar ubicada en otra parte de la memoria y, a continuación, regresa al programa principal para continuar con su ejecución.

En el momento de ejecutar la subrutina, se almacena el contenido del contador de programa (posición de la última instrucción ejecutada en el programa principal) para recuperarlo, una vez que la subrutina ha finalizado y se debe retomar la ejecución del programa principal.

Desde el punto de vista del hardware, intervienen el **registro de pila** (de tipo **LIFO**, *last in first out* o lo último que ingresa es lo primero en salir), donde se almacena el contenido del contador de programa, y el **registro apuntador de pila**, que almacena la siguiente dirección libre en el área de la memoria RAM utilizada para el registro de pila.

Desde el punto de vista del software, el microprocesador dispone de dos grupos de instrucciones para implementar una subrutina: **JSR** (salto a subrutina) o **CALL** (invocar) para llamar a una subrutina, y **RTS** (regreso de la subrutina) o **RET** (regresar), que se emplean como última instrucción de la subrutina para retornar al sitio preciso del programa que llamó a la subrutina.

Ejemplos de programación con base en distintos microprocesadores

Un microprocesador únicamente comprende el código binario (0 y 1) o lenguaje máquina. Aquí las instrucciones son cadenas o series de caracteres binarios mediante los cuales se especifica una operación. Las direcciones de memoria implicadas en la operación se denominan **instrucciones de máquina** o **código máquina**. Si bien es sumamente veloz en cuanto a la ejecución del código, ya que no necesita de un intérprete que traduzca cada línea de instrucciones, es más complicado y difícil de utilizar, por lo que la probabilidad de error en la programación es muy alta.

Por ello, ha sido sustituido por lenguajes de programación de bajo nivel, más sencillos de aprender y utilizar, y que además reducen la posibilidad de cometer errores. El más conocido es el ensamblador, una

representación simbólica del lenguaje máquina asociado, que facilita una programación menos tediosa, aunque también sea necesario conocer la arquitectura del microprocesador en particular.

Ensamblador, simulador, emulador y terminal

Para programar en lenguaje ensamblador, es sumamente importante diferenciar entre los conceptos de ensamblador, simulador, emulador y terminal.

En lenguaje ensamblador, el **ensamblador** propiamente dicho convierte el programa fuente, escrito en lenguaje ensamblador, en su equivalente en hexadecimal, denominado **programa objeto**.

Un **simulador** permite crear, ejecutar y depurar programas para un microprocesador determinado a partir de un formato de texto.

El **emulador** representa el modo de operación de un microprocesador sobre una computadora personal. Funciona con programas sobre una máquina virtual y emula al hardware verdadero: dispositivos de la pantalla, de la memoria y de entrada-salida.

Finalmente, una **terminal** o consola es un dispositivo electrónico o electromecánico de hardware usado para introducir o mostrar datos de una computadora.

Ejemplos de programación

En la programación de microprocesadores, es sustancial el uso de simuladores porque facilitan verificar el funcionamiento de un programa escrito en lenguaje ensamblador antes de la instalación física en el microprocesador.

GNUSim8085 es un simulador gráfico, ensamblador y depurador que funciona tanto en sistema operativo Linux como bajo Windows. En el sitio **http://gnusim8085.org**, sedescarga el software; allí se ofrece una guía sobre el lenguaje ensamblador y manuales de uso del programa.

En **www.charlesescobar.com/wp-content/uploads/2013/05/GNUSIM8085.pdf**, se encuentra un manual en español. Para

dispositivos móviles basados en Android, **Intel 8085 Simulator** ofrece una aplicación para programar el 8085 en código máquina y se descarga desde **https://play.google.com/store/apps/details?id=mp.project.intel8085simulator** . Finalmente, en el sitio **www.glitchwrks.com/8085projects.html** , sepresentan proyectos completos utilizando este famoso microprocesador de la firma Intel.

```
; Calcula la suma de los números del
; 1 al 10. A almacena el resultado
        .org    h'2000
        mvi     b,10        ; inicialmente B : =10
        xra     a           ; hace A : =0
loop:   add     b           ; A ; = A+ B
        dcr     b
        jnz     loop        ; salto si B no es cero
        hlt
        .end
```

Figura 2. Ejemplo de un programa en ensamblador para microprocesador Intel 8085, que suma los números del 1 al 10.

EL USO DE UN SIMULADOR FACILITA LA VERIFICACIÓN DEL FUNCIONAMIENTO DE UN PROGRAMA

La instrucción **mvi b, 10** de la **figura 2** inicializa a B con el valor 10 y **xra a** hace A = 0. A continuación, se ejecuta un lazo (loop) donde se suman ambos números, y el resultado se almacena en A, mientras B no sea igual a cero, en cuyo caso finaliza la ejecución del programa. Este programa, que se puede escribir con un procesador de texto, se denomina **código fuente**, y su extensión será **.ASM**. Por ejemplo, **SUMA8085.ASM**.

Luego del proceso de ensamblado, se generan dos nuevos archivos. **SUMA8085.LST** es un archivo de texto que contiene un listado (reporte) de cada instrucción del código fuente junto a la dirección y el correspondiente código de máquina, además de los valores ya resueltos de las etiquetas usadas y una indicación de los errores que pudieran existir. **SUMA8085.OBJ** contiene una serie de números expresados

en hexadecimal que se podría importar y grabar en la memoria del microprocesador Intel 8085 mediante una aplicación específica o cargarlo en otra aplicación, por ejemplo **GNUSim8085**, para simular la ejecución que se desarrollaría en este microprocesador.

El programa de la siguiente figura, escrito en ensamblador para el microprocesador Intel 8086, permite comparar los números almacenados en las posiciones de memoria 300 y 301, seleccionar el menor de ellos y almacenarlo en la posición de memoria 302.

```
Comparar:   MOV AL, [300]
            CMP AL, [301]
            JNB SII
            MOV AL, [301]
SII         MOV [302], AL
            MOV AH, 4C
            INT 21
```

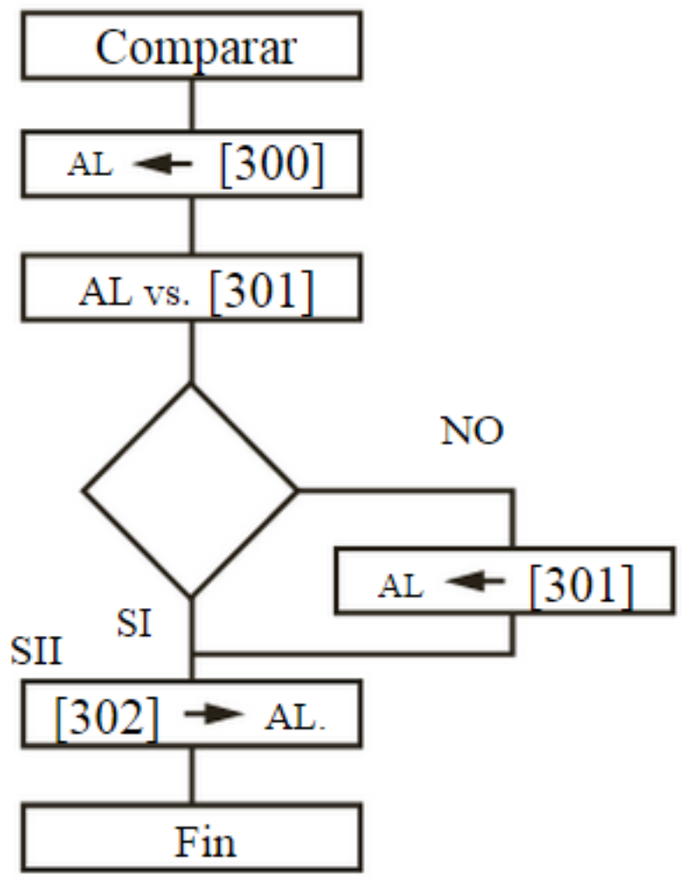


Figura 3. Ejemplo de diagrama de flujo y programa en ensamblador para microprocesador **Intel 8086** que compara y selecciona el menor de dos números.

Podemos observar que la instrucción **MOV AL, [300]** transfiere el contenido de la posición 300 de memoria al registro AL (el byte bajo del registro acumulador AX; byte alto, registro AH). AX se utiliza para almacenar resultados, lectura o escritura desde o hacia los puertos. **CMP AL, [301]** resta al contenido de Al el contenido de [301] sin afectar a ninguno de ellos y se utiliza con instrucciones de salto. **JNB SII** compara dos enteros sin signo y no salta a SII si el contenido del registro Al es mayor o igual que el contenido en la posición de memoria [300]. Finalmente, **INT 21** salta al código de la interrupción indicada por el operando inmediato, 21 (llamada al sistema operativo DOS), y finaliza el programa.

Programación en lenguaje ensamblador para microprocesadores de 8 bits

El microprocesador de 8 bits Motorola **MC6800** tiene un bus de direcciones de 16 bits para acceder a 64 KB de memoria, además de un bus de datos bidireccional de 8 bits. Provee 72 instrucciones con siete modos de direccionamiento y 197 códigos de operación. Originalmente tenía una frecuencia máxima de reloj de 1 MHz, aunque luego se incrementó a 2 MHz.

El **MC6802** es similar, e incluye 128 bytes de memoria RAM y un reloj integrado. Al igual que los microprocesadores Intel y Zilog Z80, los dispositivos MC6800/6802 aún se utilizan en automatización. También constituyen una forma excelente de aprendizaje para luego comprender los fundamentos de los microcontroladores y diseñar sistemas más extendidos actualmente, basados en estos.

Es interesante analizar el programa que permite desplazar el contenido de la dirección 0050 de la memoria un bit a la izquierda, mostrar el resultado en la dirección de memoria 0051 y limpiar la posición del bit desplazado para un microprocesador M6800.

```
LAB     $50     toma el dato en la dirección 0050
ASLB            desplaza un bit a la izquierda
STAB    $51     almacena el resultado en la dirección 0051
SW1             Interrupción por software
```

Figura 4. Ejemplo de un programa en ensamblador desarrollado para microprocesador **Motorola MC6800** , que desplaza un bit a la izquierda.

EMULADOR Y DEPURADOR PARAMC6800

En el sitio **www.hvrsoftware.com/6800emu.htm** se tiene un emulador/depurador (debugger) y ensamblador para el microprocesador MC6800 diseñado para propósitos educativos donde experimentar con distintos programas, mientras que en el sitio **http://koti.mbnet.fi/~atjs/mc6809** se encuentra una herramienta para el microprocesador Motorola MC6809.

Los acumuladores A y B son virtualmente intercambiables, y muchas instrucciones pueden utilizar un registro u otro. Sin embargo, las instrucciones **ABA** (en inglés, *Add Accumulators* o sumar el acumulador) y **SBA** (en inglés, *Subtract Accumulators* o restar el acumulador) almacenan el resultado en el acumulador A. la instrucción **ASLB** desplaza a la izquierda un bit y limpia el bit menos significativo. El bit más significativo es para el acarreo (en inglés, *carry*).

> GNUSIM8085 ES UN SIMULADOR GRÁFICO Y ENSAMBLADOR QUE FUNCIONA EN LINUX Y WINDOWS

El microprocesador **Zilog Z80** tiene un bus de datos de 8 bits, aunque opera instrucciones y direcciones de 16 bits (podría direccionar hasta 64 KB) compatible con el juego de instrucciones del microprocesador Intel 8080. Emplea 22 registros, y la velocidad de reloj depende de la versión: comenzó a 2.5 MHz y alcanzó los 20 MHz, aunque la versión más popular lo hace a 4 MHz. Tiene seis modos de direccionamiento diferentes, alimentación a + 5 V, y necesita menos circuitos auxiliares para generar la señal de reloj y para el enlace con la memoria y con la E/S. Se utilizan en dispositivos recreativos de bajo costo, y en automatización y procesamiento de señales. La versión con arquitectura de 16 bits es el **Zilog Z8000**.

```
LD      A,(ADR1)        carga OP1 en A
LD      HL,ADR2         carga la dirección de OP2 en HL
ADD     A,(HL)          sumar OP2 a OP1
LD      (ADR3),A        almacenar el resultado ETS en ADR3
```

Figura 5. Ejemplo de un programa en ensamblador para microprocesador **Zilog Z80**, que suma dos números de 8 bits y los almacena en la memoria.

SIMULADOR PARA Z80 EN LA WEB

Z80 SIMULATOR IDE 0.30 es una aplicación para el microprocesador de 8 bits Zilog Z80 y contiene un emulador, un compilador básico, ensamblador, desensamblador y debugger que cuenta con un editor de memoria de 64 KB. Está disponible en el sitio **http://z80-simulator.sourceforge.net/**.

Para sumar dos operandos de 8 bits, OP1 y OP2, almacenados respectivamente en las direcciones de memoria ADR1 y ADR2, se utiliza la instrucción **RES**; el resultado obtenido se almacena en la dirección ADR3.

Programación en lenguaje ensamblador para microprocesadores de 16 bits

El microprocesador Motorola **MC68000** es el hermano de mayor potencia que complementa la línea de productos de 8 bits 6800. Tiene un bus de datos de 16 bits, aunque el tamaño de sus registros generales es de 32 bits, por lo que realiza dos ciclos de escritura/lectura de 16 bits para simular la gestión de datos de 32 bits. Posee 24 bits de direcciones para direccionar hasta 16 MB. Es un microprocesador con **arquitectura CISC** (en inglés, *Complex Instruction Set Computer* o computador con conjunto de instrucciones complejas), y el conjunto de instrucciones se caracteriza por ser muy amplio y permitir operaciones complejas entre operandos situados en la memoria o en los registros internos en contraste con la **arquitectura RISC**. Se alimenta con +5V, la frecuencia de funcionamiento es de 4/25 MHz, soporta 5 tipos de datos, proporciona 56 diferentes tipos de instrucciones y 14 modos de direccionamiento, entre otras características.

> MOTOROLA MC68000 ES EL HERMANO DE MAYOR POTENCIA QUE COMPLEMENTA LOS PRODUCTOS DE 8 BITS

En la figura anterior, se observa un programa para sumar dos datos almacenados en dos registros e indicar si ocurre overflow.

OPEN SOURCE EN LAWEB

EASy68K es un programa open source distribuido bajo licencia de uso público general GNU y se encuentra en el sitio **www.easy68k.com** . Esun editor/ensamblador y simulador para el microprocesador Motorola MC68000, que funciona bajo Windows y no requiere hardware adicional.

```
                ORG         0
                MOVE.L      D0,D3
                ADD.L       D1,D3
                BVS         ERROR
                BRA         FIN

ERROR           LEA         0,A0
                MOVE        # 'ERROR',D5
                MOVEP       D5,$3100(AO)
                MOVEP.W     D5,$3108(A0)

FIN             TRAP        #1

                END
```

Figura 6. Ejemplo de diagrama de flujo y programa en ensamblador para microprocesador **MC68000**, que suma dos datos e indica si existe condición de overflow.

RESUMEN

En este capítulo presentamos los fundamentos de la programación en lenguaje ensamblador y desarrollamos algunas aplicaciones básicas. Comenzamos describiendo las características de la programación en lenguaje ensamblador, los modos de direccionamiento y el conjunto de instrucciones del microprocesador Intel 8085, uno de los dispositivos lógicos de referencia en el mundo de las CPU de 8 bits. A continuación, explicamos varias técnicas y herramientas básicas para el desarrollo de programas y describimos algunos ejemplos de programación en lenguaje ensamblador para microprocesadores de 8 y 16 bits.

Actividades

TEST DE AUTOEVALUACIÓN

1 El microprocesador Intel 8085 utiliza una fuente de alimentación de ____ volts.

2 Además del acumulador, indique seis registros de propósito general de 8 bits en el microprocesador 8085.

3 Indique los distintos tipos de direccionamiento que utiliza el microprocesador Intel 8085.

4 La instrucción MOV B, A del microprocesador Intel 8085 transfiere el contenido del registro A al registro B. Por lo tanto, esta instrucción utiliza el modo de direccionamiento _____.

5 La instrucción JMP addr es una instrucción de ____.

6 La instrucción CALL es una instrucción de ____.

EJERCICIOS PRÁCTICOS

1 El código de op (hexadecimal) para la instrucción ADD C es _____.

2 Esta operación suma el contenido del registro ____ al contenido del acumulador.

3 La instrucción ADD C utiliza direccionamiento (indicar respuesta correcta): de registro ___ o indirecto de registro ___.

4 El código de op (hexadecimal) para la instrucción MOV A,M es ___. Esta operación transfiere datos de/desde _____ a/al _____.

5 En la instrucción MOV A,M la posición de memoria se identifica por el contenido del _____.

6 El código de op (hexadecimal) para la instrucción ANA M es _____.

7 La instrucción ANA M es una instrucción de ___ byte/s.

PROFESOR EN LÍNEA

Si tiene alguna consulta técnica relacionada con el contenido, puede contactarse con nuestros expertos: profesor@redusers.com.

Microprocesadores de 8 bits

Presentaremos las tecnologías básicas en microprocesadores de 8 bits más utilizadas actualmente. Veremos sus características fundamentales y la representación de sistemas mínimos para realizar el desarrollo de aplicaciones basadas en microprocesador, con algunos ejemplos de programación y aplicaciones típicas. De esta manera, aplicaremos los conocimientos adquiridos a lo largo de cada uno de los capítulos.

Servicio de atención al lector: usershop@redusers.com

Características fundamentales

Los microprocesadores, dispositivos de gran escala de integración que contiene la CPU completa, necesitan de circuitos auxiliares exteriores para funcionar. Según han ido evolucionando los microprocesadores, también lo han hecho sus prestaciones y capacidades de servicio, así como la facilidad de adquisición. Además de formar parte de las computadoras, se emplean en aplicaciones que requieren de funciones de control y supervisión, como por ejemplo, control industrial y de máquinas, controles de motores y telemetría.

Un microprocesador es capaz de ejecutar millones de instrucciones por segundo a partir del programa almacenado en su memoria.

Definición

Un **microprocesador** es un dispositivo electrónico que contiene varias unidades diseñadas para realizar una tarea específica. La **arquitectura del microprocesador** se refiere a las unidades, su diseño y organización. Las principales unidades básicas de todos los microprocesadores son la unidad lógica y aritmética (**ALU**), registros y unidad de control que, en conjunto con otras estructuras auxiliares, conforman un microprocesador específico. Por medio de las unidades básicas, un microprocesador es capaz de procesar información, controlar la transferencia de datos entre él mismo y la memoria o el sistema de entrada/salida, llevar a cabo operaciones lógicas y aritméticas, y ejecutar programas por medio de los cuales se desarrollan una serie de instrucciones.

Buses

Un **bus** es un conjunto de conductores eléctricos comunes que interconectan los componentes de un sistema basado en microprocesadores. Los buses de datos, direcciones y control forman parte de las conexiones internas y externas que utilizan los microprocesadores para enviar datos, direcciones y señales de control. El **bus de datos** es bidireccional y por él se transfieren datos o

instrucciones al microprocesador y los resultados de una operación desde el microprocesador. Dependiendo de cada microprocesador el tamaño del bus de datos es de 8, 16, 32, o 64 bits.

El **bus de direcciones** es unidireccional, y por medio de él un microprocesador envía un código de dirección a una memoria o a un dispositivo externo. El ancho del bus de direcciones se especifica mediante el número de líneas y, a mayor número, se puede acceder a más posiciones de memoria.

> UNMICROPROCESADOR PUEDE EJECUTAR MILLONES DE INSTRUCCIONES POR SEGUNDO

El **bus de control** se emplea para coordinar las operaciones y comunicarse con dispositivos externos. Posee señales que permiten leer y escribir datos en la memoria o en un puerto de E/S en el instante apropiado. Además, se utilizan para insertar estados de espera para los dispositivos más lentos y evitar congestión en el bus, condición que podría producirse cuando dos o más dispositivos intentan transmitir al mismo tiempo. Algunas líneas del bus de control comunes a distintos microprocesadores son **control de lectura de memoria** , **control de escritura de memoria** , **control de lectura de E/S** y **control de escritura de E/S** . También se tienen las señales de reloj, reset y manejo de interrupciones.

Arquitecturas clásica y paralela

La arquitectura clásica de un microprocesador responde al modelo de Von Neumann que propone tres unidades básicas. El núcleo del procesamiento matemático del sistema microprocesador, la ALU, es la parte de la computadora que realiza las operaciones aritméticas (suma y sustracción) y las operaciones lógicas con los datos (NOT, AND, OR). El resto de las unidades (control, registros, memoria y E/S) suministran los datos a la ALU y recuperan los resultados. Los **registros** almacenan los datos para la ALU y luego almacenan los resultados de las operaciones producidas por la ALU. Estos registros son posiciones de memoria temporales, internas al microprocesador, conectadas a la ALU. Como resultado de una operación, se activan indicadores de estado o flags, cuyos valores se almacenan dentro del procesador en el registro de estado.

La **unidad de control** proporciona las señales que gobiernan el funcionamiento de la ALU, y la transferencia de datos dentro y fuera de ella. Las funciones básicas son dos: el **secuenciamiento**, que hace que el microprocesador avance a través de una serie de microoperaciones basadas en el programa ejecutado, y la **ejecución**, gracias a la que se ejecuta cada microoperación.

Para superar los límites en la construcción de microprocesadores por disipación de potencia eléctrica, que restringe el tamaño de los transistores, y la velocidad de reloj, limitada por la velocidad de la luz, se recurre a otras arquitecturas diponibles. De esta forma, se acelera el funcionamiento del sistema microprocesador frente a los diseños tradicionales.

Figura 1. Computadora de placa reducida **Raspberry Pi** modelo B basada en un microprocesador **ARM1176JZF-S** de 32 bits a 700 MHz.

MICROOPERACIONES

Sabemos que la ejecución de un programa consiste en la ejecución secuencial de instrucciones, de acuerdo a lo que fue pensado por el programador. Recordemos también que cada instrucción se ejecuta durante el ciclo de instrucción conformado por subciclos más cortos. De esta manera, la ejecución de cada subciclo involucra una o más operaciones sencillas llamadas **microoperaciones**.

Ya sea tanto en aplicaciones de simulación y modelado de sistemas complejos como en problemas que requieren la manipulación/cómputo de grandes cantidades de datos y en grandes problemas físicos, económicos o biológicos, una posible solución es la **arquitectura paralela**. Esta es una organización de sistemas microprocesadores que incorporan varias CPU en una disposición que dependerá del número y complejidad de las CPU utilizadas, la disponibilidad de memoria compartida, la topología de interconexión y los dispositivos de E/S.

En la actualidad, uno de los sistemas de procesamiento paralelo más poderoso es el **Cray XK7** de 560640 núcleos basados en procesadores AMD Opteron. El Computational Engineering and Design Research Group de la Universidad de Southampton, cuyo sitio web es **www.southampton.ac.uk/~sjc/raspberrypi/pi_supercomputer_southampton.htm**, ha desarrollado una supercomputadora de bajo presupuesto utilizando la computadora de placa única **Raspberry Pi**.

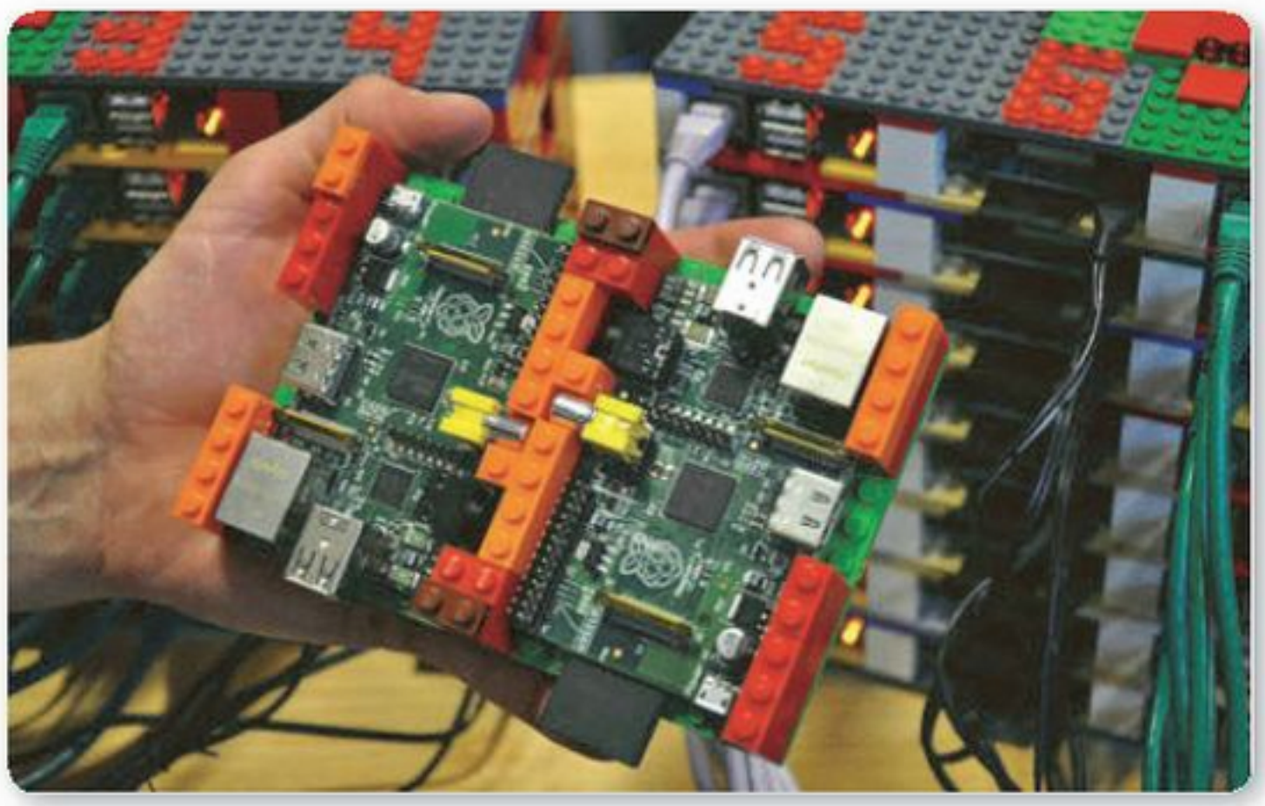


Figura 2. Supercomputadora paralelo desarrollada en la Universidad de Southampton basada en la placa de bajo costo Raspberry Pi.

PARALELIZACIÓN DE LOS PROGRAMAS

Es la estrategia para construir un programa paralelo y consiste en descomponer el problema en tareas de menor complejidad, que puedan ejecutarse simultáneamente. Los pasos son: descomponer el problema en tareas menores, asignar las tareas a los microprocesadores que puedan operar en paralelo y coordinar las tareas simultáneas.

Tecnologías de fabricación

Los microprocesadores se fabrican empleando técnicas similares a las utilizadas para otros circuitos integrados, como chips de memoria, aunque los microprocesadores tienen una estructura más compleja y su fabricación exige técnicas extremadamente precisas.

Entre las etapas del proceso, podemos identificar: la creación de sustrato, la oxidación, la litografía, el grabado, la implantación iónica y la deposición de capas. La creación de un sustrato de silicio en forma de oblea de enorme pureza es la primera etapa.

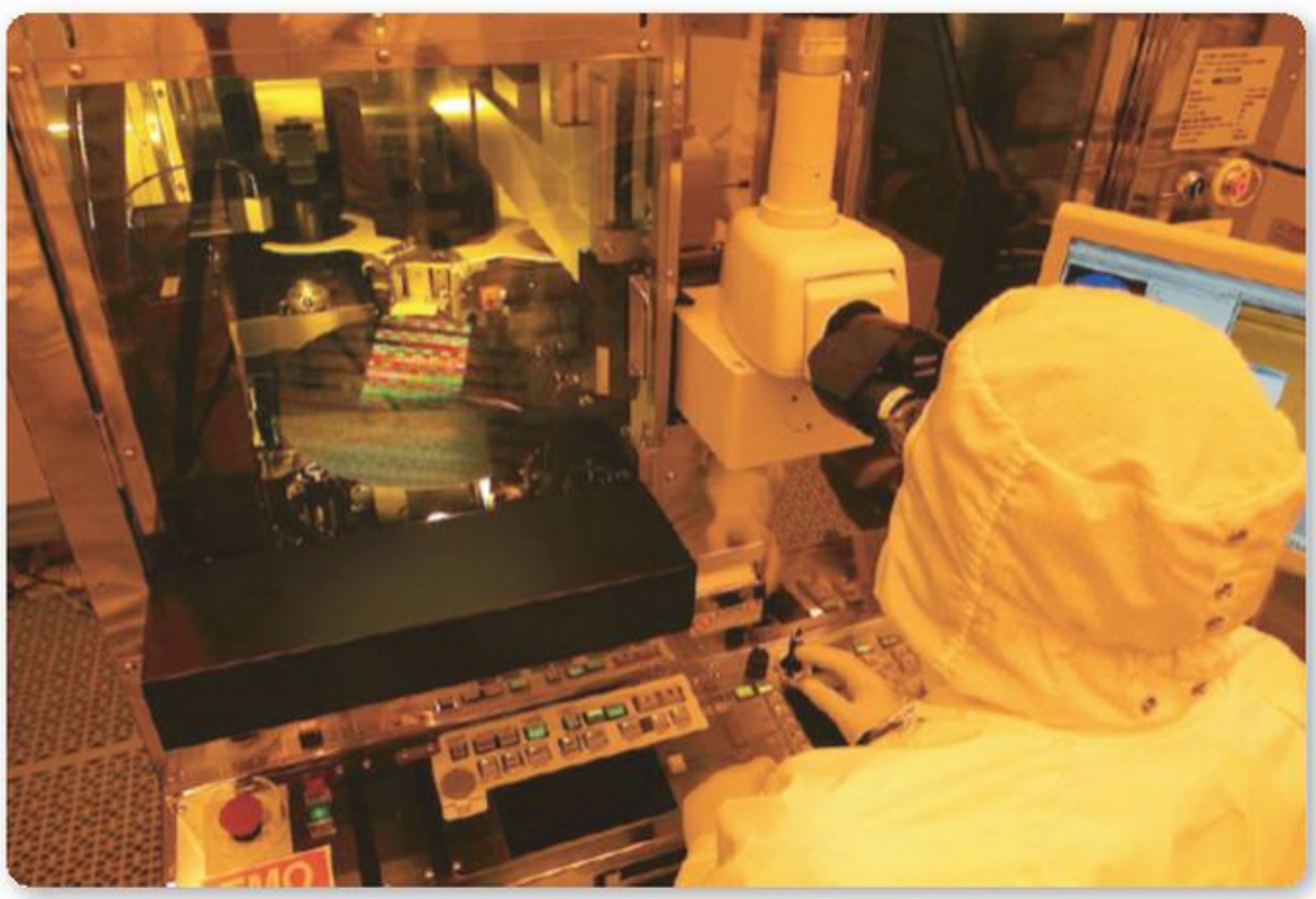


Figura 3. El proceso de fabricación de un microprocesador es sumamente complejo y requiere transformar el silicio en obleas a partir de las que se obtienen los microprocesadores.

RASPBERRY PI

Es una computadora de placa reducida o placa única (**SBC**) de bajo costo, desarrollada en Reino Unido por la Fundación Raspberry Pi, con el objetivo de estimular la enseñanza de las ciencias de la computación en las escuelas, y realizar proyectos de electrónica y aplicaciones normalmente ejecutadas en computadoras de mayor tamaño. **Raspberry Pi B** utiliza un microprocesador (CPU) ARM1176JZF-S de 32 bits a 700 MHz. Podemos verla en: **www.raspberrypi.org** .

> LAS TÉCNICAS DE FABRICACIÓN SON SIMILARES A LAS EMPLEADAS CON LOS CHIPS DE MEMORIA

En la etapa de **oxidación**, se coloca una capa eléctricamente no conductora, llamada **dieléctrico de dióxido de silicio** , por exposición de la oblea de silicio a una atmósfera de oxígeno en un horno a unos 1.000 ºC. El oxígeno se combina con el silicio para formar una delgada capa de óxido de unos 75 angstroms de espesor. En la **implantación iónica** se introducen en el silicio impurezas de boro o fósforo para alterar su conductividad; estas se incrustan sobre la superficie de la oblea mediante un implantador iónico. Finalmente, las capas de material empleadas para fabricar el microprocesador se depositan mediante el bombardeo atómico en un plasma; así se obtiene una película de gran pureza cuyo espesor debe controlarse con una precisión de una fracción de micra.

Fabricantes

En microprocesadores de 8 bits, los fabricantes más detacados han sido Intel, Motorola, Zilog, entre otras empresas.

Intel Corporation es una compañía pionera en el desarrollo y comercialización de microprocesadores, que actualmente mantiene una enorme cuota del mercado mundial de los microprocesadores, dado que estos se utilizan en la mayoría de las computadoras compatibles con PC. En Motorola, uno de sus principales exponentes fue el **MC6800** lanzado al mercado en 1975 poco tiempo después del Intel **8080**. Está formado por un conjunto de 78 instrucciones y, posiblemente, sea el primer microprocesador que contó con un registro índice. Por lo general, se fabricaba en un encapsulado DIP de 40 pines. Zilog, es un fabricante de microprocesadores de 8 bits, y su producto más reconocido es el Zilog Z80. La empresa Zilog fue fundada en California en 1974 por Federico Faggin, quien trabajó perfeccionando el primer microprocesador de Intel, el Intel 4004.

Evolución y perspectivas a futuro

Tanto los microprocesadores como los microcontroladores representan un gran desarrollo de la tecnología electrónica, y los

artefactos que los incorporan han modificado los hábitos de la humanidad. En nuestros días, distintos instrumentos de medición, electrodomésticos y en general cualquier dispositivo electrónico utiliza alguno de estos dos componentes para optimizar su funcionamiento.

Las aplicaciones de control, medición, instrumentación, entretenimiento y consumo impulsan el creciente mercado tanto para los microprocesadores como para los microcontroladores.

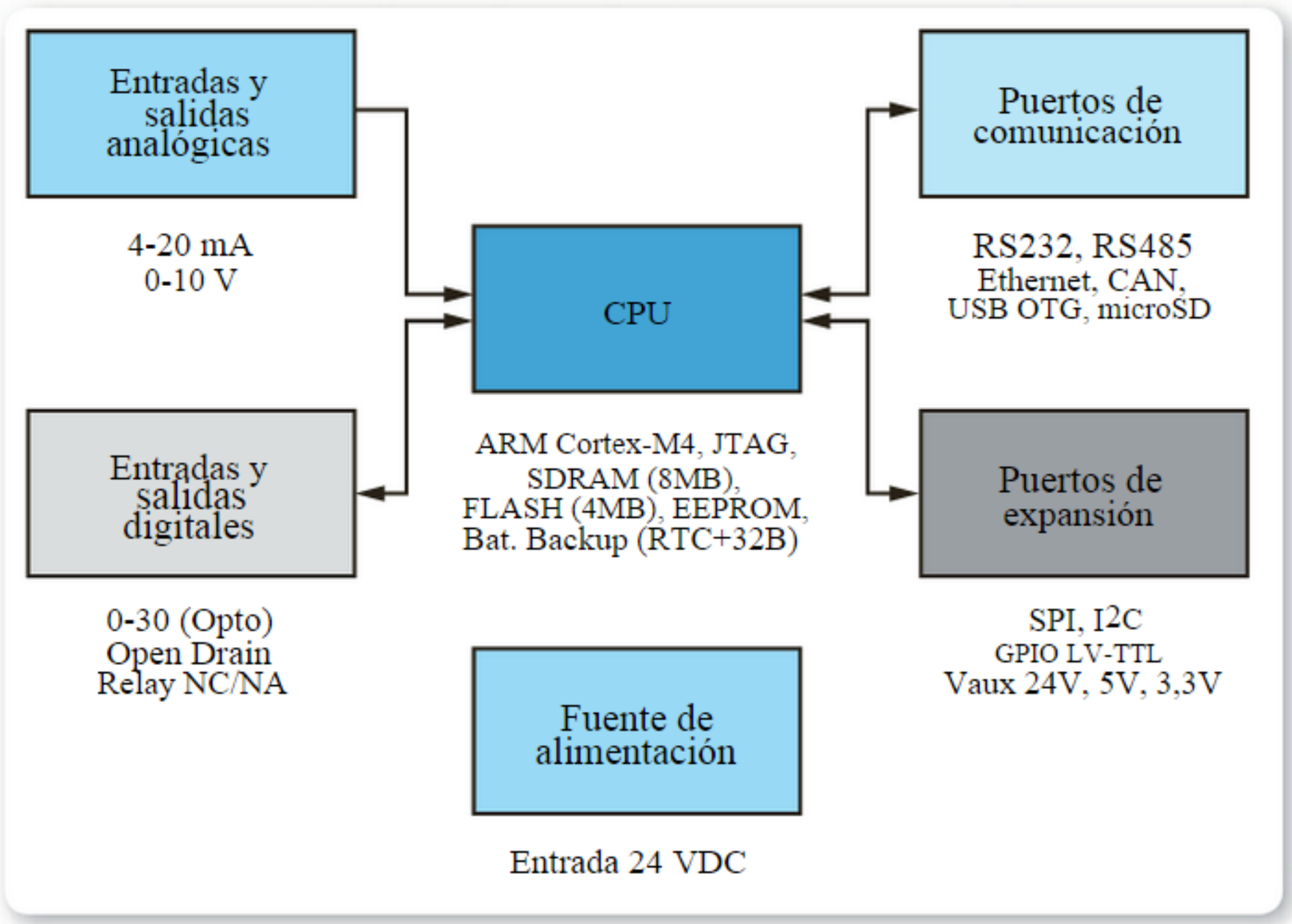


Figura 4. Diagrama en bloques de la **CIAA** (**Computadora Industrial Argentina Abierta**) organizada alrededor de una CPU ARM.

En la Argentina, se desarrolla el proyecto **Computadora Industrial Abierta Argentina (CIAA)** , cuyo sitio web es **www.proyecto-ciaa.**

APLICACIONES DE LOS MICROPROCESADORES

Los microprocesadores de baja gama (4, 8 y 16 bits) se emplean para el control de electrodomésticos, juguetes, smart-cards y periféricos. Los de gama media (16 y 32 bits), para procesamiento y control, teléfonos móviles, automóviles, PDA. Los de gama alta (64 y 128 bits) se usan para el procesamiento de señales, computadoras personales y videoconsolas, entre otras aplicaciones.

com.ar/devwiki/doku.php. Se trata de una plataforma electrónica preparada especialmente para aplicaciones industriales, y su diseño está disponible para ser usado en forma libre y gratuita en el desarrollo de productos y servicios. La CIAA sirve para aplicaciones en agroindustria, industria automotriz, fábricas de alimentos, metal-mecánica, control de procesos químicos, máquinas textiles, etcétera, donde se usan sistemas electrónicos para automatizar procesos, y también es ideal para la enseñanza en universidades, institutos terciarios y escuelas secundarias.

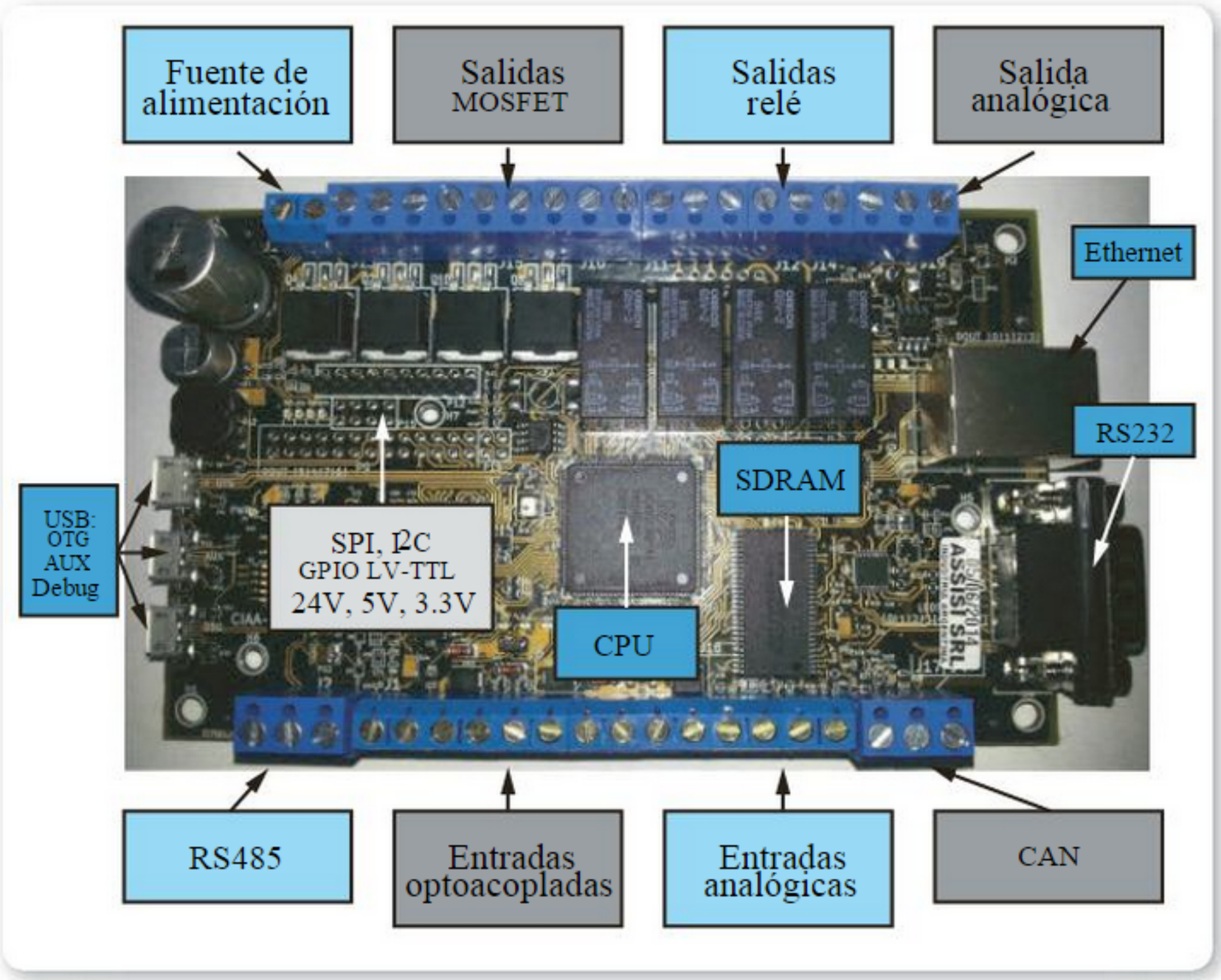


Figura 5. CIAA especialmente diseñada para aplicaciones de automatización.

COMPUTADORA INDUSTRIAL ABIERTA ARGENTINA

El proyecto **Computadora Industrial Abierta Argentina** (**CIAA**) es una plataforma electrónica preparada especialmente para aplicaciones industriales, basada en un poderoso microprocesador **ARM Cortex-M4F**. Para más información, se puede consultar el sitio web **www.arm.com/products/processors/cortex-m/cortex-m4-processor.php**.

Modo microprocesador (modo expandido)

Tanto los microprocesadores como los microcontroladores necesitan comunicarse con el mundo exterior. Para ello, utilizan puertos de E/S a los que se pueden conectar distintos periféricos. En el caso de los microprocesadores, se emplean dispositivos adicionales, como latch o buffer, que facilitan la compatibilidad de esos periféricos con el microprocesador.

Los microprocesadores más populares

Realizaremos una breve descripción de las tecnologías de los microprocesadores de 8 bits más utilizadas y la representación de sistemas mínimos para realizar el desarrollo de aplicaciones basadas en microprocesador.

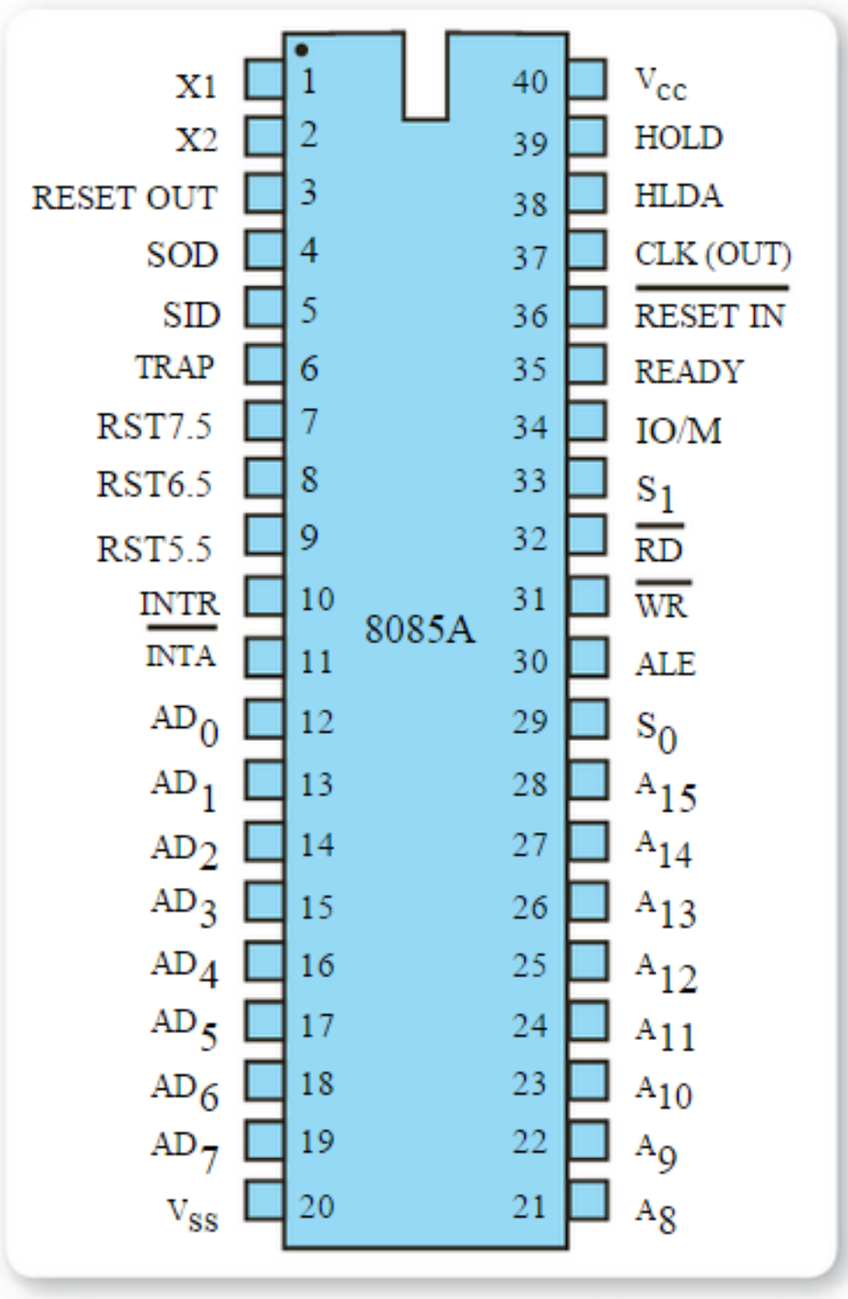


Figura 6. Diagrama de pines del popular microprocesador de 8 bits Intel **8085A**.

En todos los casos, haremos referencia a los fabricantes más conocidos, como son los casos de Intel, Motorola y Zilog.

Intel 8085, Motorola 6800, Zilog Z80

La línea de microprocesadores de 8 bits es considerada como el núcleo para otras familias y fabricantes. **Intel 8080** e **Intel 8085** fueron utilizados en computadoras a finales de los 70 y comienzos de los 80. Específicamente, el diseño del microprocesador 8080 fue la base para la creación del **Z80**, microprocesador que superó, con amplitud, las prestaciones y el mercado del 8080.

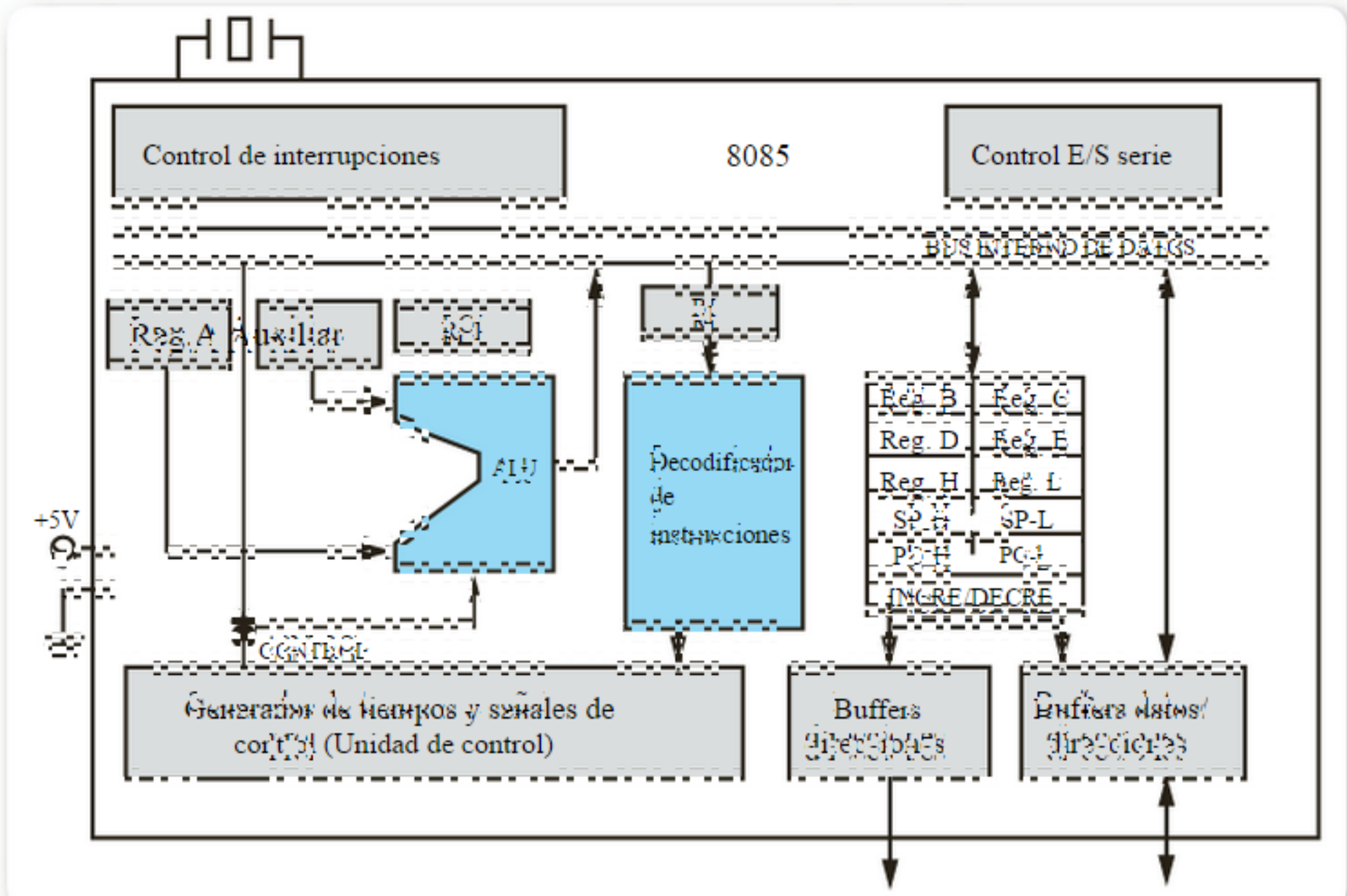


Figura 7. Arquitectura del microprocesador de 8 bits Intel **8085** con los distintos bloques que lo conforman.

En la actualidad, hay muchas versiones clónicas de esos microprocesadores e, incluso, algunos núcleos de microcontroladores siguen, como diseño base en la CPU, el diseño de los Z80.

Respecto de Intel, su primer microprocesador fue el **4004** (de 4 bits). A continuación, desarrolló el primer microcontrolador de 8 bits de la historia, el **8080**, que sería la base del algo más avanzado **8085**, y este, a su vez, el fundamento para los microprocesadores de 16 bits **8086/8088**.

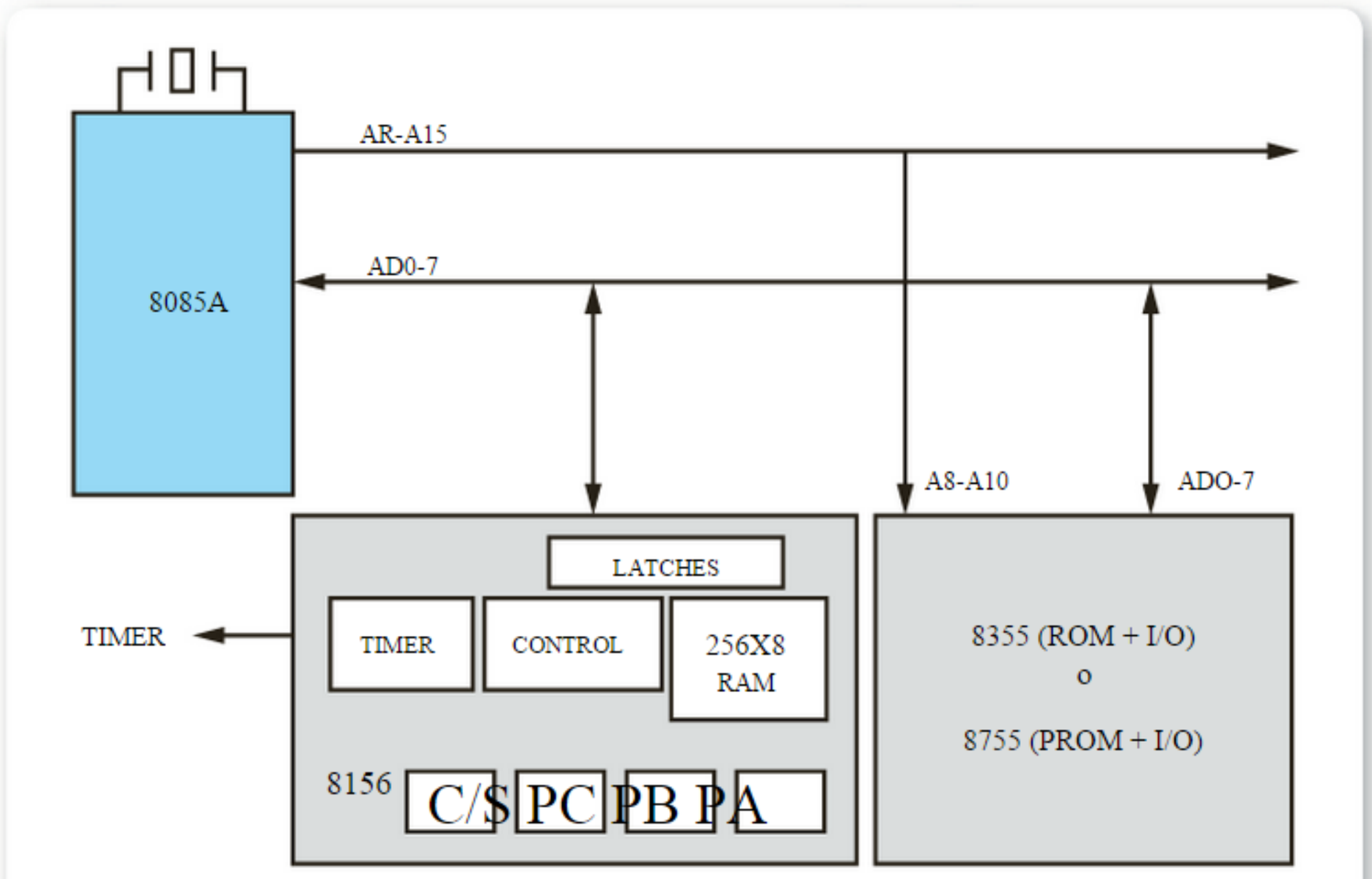


Figura 8. Sistema básico para el microprocesador **8085A** que incluye bloques asociados para desempeñar funciones determinadas y muy concretas.

Desarrollos posteriores involucraron a los conocidos **80286/80386/80486/Pentium** , hasta llegar a las tecnologías actuales **Intel Core i7** de cuarta generación y 22 nm, que ofrecen avances significativos en cuanto a desempeño, ya que incluyen mejoras importantes en gráficos, duración de la batería y seguridad, e incorporan la tecnología Hyper-Threading. Esta le permite, a cada núcleo del microprocesador funcionar en dos tareas al mismo tiempo para optimizar la multitarea, la tecnología de gráficos Iris™ para funciones visuales 3D, y edición de fotografías y videos con mayor rapidez.

Si bien Motorola ingresó en el mercado de los microprocesadores luego de Intel partiendo de un microprocesador de 8 bits en 1974, los procesadores Motorola han evolucionado en algunos de los diseños más influyentes de los sistemas microprocesadores, como el **68000** y la arquitectura PowerPC. Luego del éxito del microprocesador **8008** de 8 bits de Intel, Motorola presentó su primer microprocesador de 8 bits, el **6800**, en el año 1974. El **MC6800** predominó en el mercado en parte por el hardware de soporte orientado al sistema, que la empresa introdujo con el 6800. Cerca de 1977, Motorola introdujo el **6809**, un

procesador de 8 bits con ciertas características de 16 bits. El 6809 tenía dos acumuladores de 8 bits y dos registros de índices, y punteros de pila de 16 bits, lo que permitió modos de direccionamiento de memoria avanzada. Después llegarían el microprocesador **MC6800** de 16 bits y la tecnología PowerPC desarrollada en alianza con Apple e IBM, que utilizaba una arquitectura reducida de instrucciones. Actualmente, es desarrollado por Power(**www.power.org**). Por otro lado, la división Motorola de microprocesadores es, desde 2004, una compañía **Freescale** (**www.freescale.com**).

La empresa Zilog Inc. desarrolló, durante 1976, el microprocesador de 8 bits **Z80**, construido con tecnología NMOS y basado en el Intel 8080, aunque ampliado, es decir, admitía todas las instrucciones. Dado su éxito en el mercado, a lo largo del tiempo se han producido numerosas versiones clónicas y actualmente se utiliza en una multitud de **sistemas embebidos**. Federico Faggin, fundador de Zilog, fue el diseñador en jefe del microprocesador Intel 4004 y del Intel 8080. Hoy, Zilog es una compañía **IXXs**.

RESUMEN

El objetivo de este capítulo ha sido presentar al lector las tecnologías básicas en microprocesadores de 8 bits. Para ello, analizamos las características fundamentales de estos microprocesadores, luego examinamos las familias de microprocesadores más utilizadas, tanto en la industria como en educación, y focalizamos en los microprocesadores más destacados de cada fabricante. También se realizaron referencias a los sistemas microprocesadores que utilizan arquitectura paralela y se presentaron ejemplos desarrollados sobre la base de poderosos microprocesadores.

Actividades

TEST DE AUTOEVALUACIÓN

1 Explique las diferencias básicas entre la arquitectura Von Neumann y la arquitectura paralela.

2 ¿Qué significa una arquitectura de microprocesador de 8 bits?

3 Explique la compatibilidad entre los microprocesadores Zilog Z80 e Intel 8080.

4 ¿Qué significa modo expandido en un microprocesador?

5 ¿Qué microprocesadores de 16 bits derivan de Intel, Motorola y Zilog de 8 bits?

EJERCICIOS PRÁCTICOS

1 Ingrese al sitio de la empresa Zilog e investigue acerca de su gama de microprocesadores.

2 Imagine un microprocesador con bus de datos de 8 bits y direcciones de 16 bits, ¿qué cantidad de memoria direccionaría?

3 Compare los microprocesadores MC6800 y Z80 en: tamaño de los datos, frecuencia de trabajo, capacidad de direccionamiento, densidad de integración, número de registros internos, juego de instrucciones CISC o RISC y arquitectura interna/externa.

4 Investigue microprocesadores de 8 bits de fabricantes no indicados en el texto.

5 Investigue acerca de la estabilidad en la frecuencia de reloj con cristales piezoeléctricos y a partir de circuitos RC.

6 ¿Cómo opera el microprocesador 8085 al direccionar 16 bits con un bus de direcciones de 8 bits?

PROFESOR EN LÍNEA

Si tiene alguna consulta técnica relacionada con el contenido, puede contactarse con nuestros expertos: profesor@redusers.com.

*07

Microprocesadores de 16 bits

En el capítulo 6 presentamos los microprocesadores de 8 bits, sus características, fabricantes, proceso de fabricación y modelos más populares. En este capítulo, desarrollaremos un análisis similar, aunque focalizado en los de 16 bits, haciendo referencia a los microprocesadores Intel y Motorola.

Servicio de atención al lector: usershop@redusers.com

Características fundamentales

En arquitectura de microprocesadores, 16 bits se refiere a CPU y ALU basadas en registros, bus de direcciones o bus de datos de ese ancho. Los procesadores de 16 bits más conocidos son: **Intel 8086**, **Motorola 68000** e **Intel 80286**. El Intel 8088 es compatible en código con el Intel 8086 y puede considerarse de 16 bits respecto de registros e instrucciones aritméticas, mientras que su bus de datos es de 8 bits. Los microprocesadores de 16 bits permanecen en uso en una amplia variedad de aplicaciones embebidas.

Figura 1. Microprocesador **Zilog Z8000** de 16 bits. El **Z80000** fue una continuación del diseño en 32 bits.

Definición

La definición de microprocesador también es válida para arquitecturas de microprocesadores de 16 bits. Por medio de las unidades básicas que lo conforman, es capaz de procesar

EN SÍNTESIS

Nunca está de más fijar los datos. Los microprocesadores 8080/8085, 6800, 6502 y Z80 son de 8 bits. Los 8086, 8088, 68000, 65816 y Z8000 son microprocesadores típicos de 16 bits, mientras que los 80386, 68020 y Z80000 son ejemplos de microprocesadores avanzados de 32 bits.

información, controlar la transferencia de datos entre él mismo y la memoria o el sistema de entrada/salida, realizar operaciones lógicas y aritméticas, y ejecutar programas por medio de los cuales se desarrollan una serie de instrucciones.

Buses

Los microprocesadores de 8 y 16 bits comparten los mismos buses: datos, direcciones y control, aunque con diferentes características.

En el bus de datos de los microprocesadores de 8 bits, solo se puede accesar un byte de memoria en un solo ciclo de reloj. En un microprocesador de 16 bits, se comunican simultáneamente 16 bits (2 bytes) entre el microprocesador y la memoria, por lo que el microprocesador puede gestionar bloques de memoria en la mitad del tiempo y con la mitad de instrucciones. Es decir, se hace un uso más eficiente del acceso a la memoria por parte del microprocesador. Al tamaño del bus de datos también suele llamársele **tamaño de palabra** .

> LOS DE 16 BITS SON 8086, 8088, 65816 Y Z8000, MIENTRAS QUE LOS DE 32, 80386, 68020 Y Z80000

Arquitecturas clásica y paralela

La arquitectura clásica de un microprocesador de 16 bits también responde al modelo de Von Neumann, que propone tres unidades básicas: ALU, registros y Unidad de Control.

Ya explicamos en capítulos anteriores que las limitaciones en esta arquitectura se relacionan con que el microprocesador ejecuta una instrucción por vez, y la comunicación con la memoria y los dispositivos de E/S disminuyen el rendimiento.

Algunas mejoras a la arquitectura Von Neumann original se relacionan con buses especializados, gestión de interrupciones, implementación de unidades de punto flotante, uso de memoria caché y pipelines.

Un enfoque alternativo es alejarse de la arquitectura clásica de Von Neumann agregando microprocesadores (**arquitectura paralela**). Algunos ejemplos de utilización de microprocesadores más poderosos que los basados en 8 y 16 bits se presentaron en el **capítulo 6** .

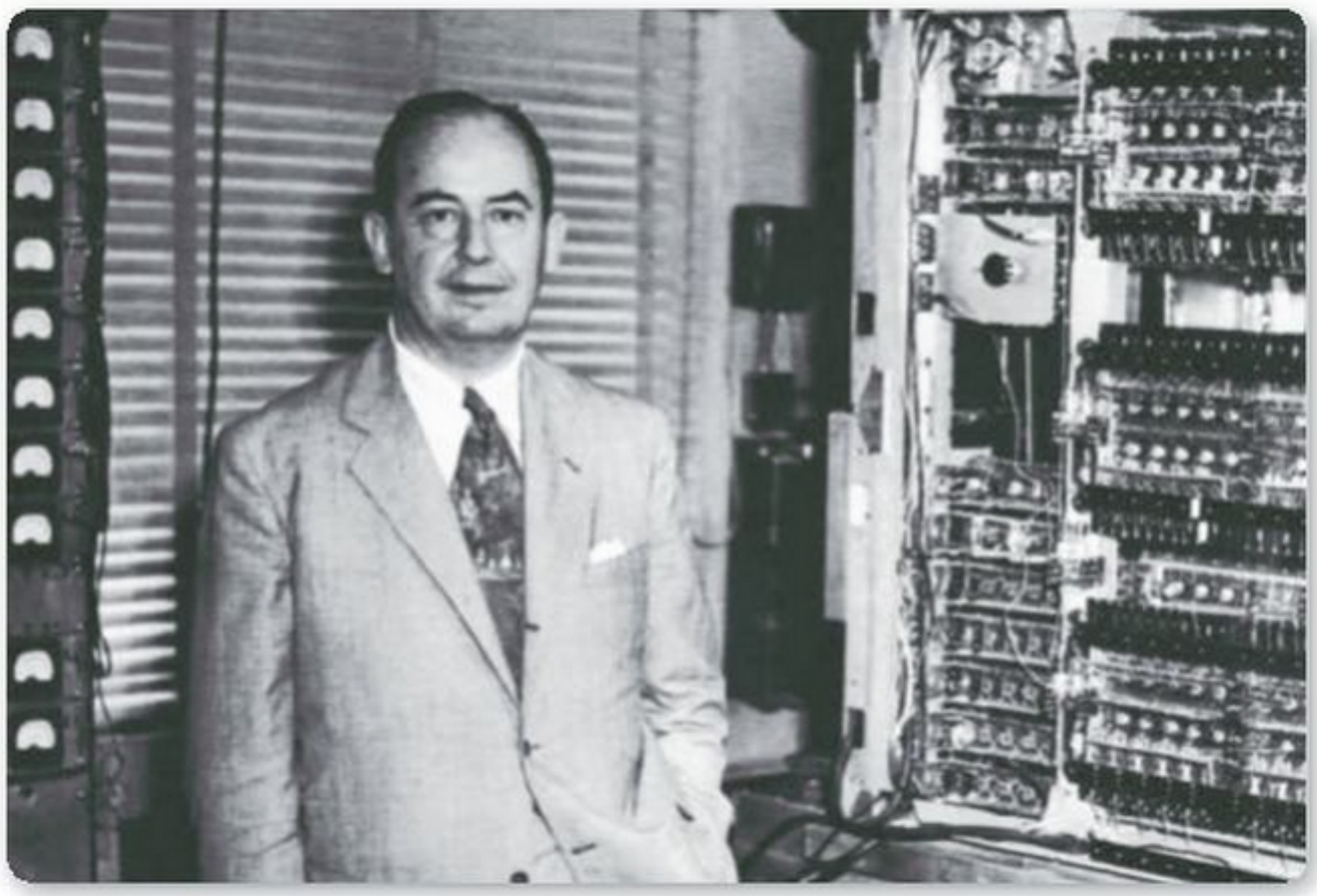


Figura 2. John Von Neumann, matemático y diseñador de la arquitectura de computadoras que utiliza la misma memoria para almacenar instrucciones y datos.

Tecnologías de fabricación

Los microprocesadores de 16 bits, al igual que los de 8 bits, se fabrican empleando técnicas similares a las utilizadas para otros circuitos integrados, como chips de memoria, aunque los microprocesadores tienen una estructura más compleja y su fabricación exige técnicas extremadamente precisas. Entre las etapas del proceso analizadas antes, se tienen: la creación de sustrato, la oxidación, la litografía, el grabado, la implantación iónica y la deposición de capas.

Fabricantes

Algunos de los fabricantes más conocidos de microprocesadores de 16 bits han sido Intel y Motorola, entre otras empresas. En cuanto a Zilog, luego del Z80, introduce varios microprocesadores de 16 bits, por ejemplo el **Z8000**, y de 32 bits, aunque sin gran éxito. Por este motivo, se orienta al mercado de microcontroladores fabricando CPU básicas y circuitos integrados para aplicaciones específicas (**ASIC/ ASSP**), construidos alrededor del núcleo de sus procesadores.

Evolución y perspectivas a futuro

Las arquitecturas de 16 bits, que aún se utilizan en determinadas aplicaciones, han evolucionado hacia arquitecturas de 32 bits facilitando, por ejemplo, la implementación de sistemas operativos que emplean memoria virtual.

ALGUNOS DE LOS FABRICANTES MÁS CONOCIDOS SON INTEL Y MOTOROLA

Por su parte, las arquitecturas de 32 bits evolucionaron hacia las de 64 bits. El bus de datos de 64 bits permite acceder a una memoria de 64 bits, y las versiones posteriores de estos microprocesadores proporcionaron al usuario una gestión más eficiente de las aplicaciones multimedia a una velocidad de reloj mayor.

Figura 3. Microprocesador **80386 CISC** con arquitectura x86 de 32 bits, fabricado y producido por la empresa Intel.

MICROPROCESADOR 80386 DE 32 BITS

La relevancia de los microprocesadores 80386, además de sus 32 bits, reside en que es el prototipo que marcó el inicio de la tercera generación de microprocesadores. En estos, se destaca la incorporación de una unidad de traslación de páginas, que hace posible la implementación de Sistemas Operativos que emplean memoria virtual.

Modo microprocesador (modo expandido)

Al igual que los microprocesadores de 8 bits, los de 16 bits también necesitan comunicarse con el mundo exterior por medio de puertos de E/S a los que se les pueden conectar distintos periféricos. En el caso de los microprocesadores, se emplean dispositivos adicionales, como latch o buffer, que facilitan la compatibilidad de esos periféricos con el microprocesador.

Los microprocesadores más populares

En este apartado, realizaremos una breve descripción de las tecnologías de microprocesadores de 16 bits más utilizadas y la representación de sistemas mínimos para realizar el desarrollo de aplicaciones basadas en microprocesador. Además, haremos referencia a los fabricantes más conocidos, como el caso de Intel y Motorola.

Motorola 68000, Intel 8086/88/286

El exitoso microprocesador **Motorola 68000** (**MC68000**) es de 16 bits en su bus de datos, aunque utiliza 32 bits en sus registros generales y en la mayoría de las operaciones matemáticas, mientras que su software es de 32 bits

MICROPROCESADORES DE 64 BITS

Constituyen una arquitectura en la cual el tamaño de los datos es de 64 bits, y las direcciones de memoria se componen de 8 bytes. Contienen una CPU de 64 bits internos, buses e instrucciones de 64 bits y aumentan la capacidad de memoria (tanto RAM como virtual) superando los 4 GB de los microprocesadores de 32 bits. Las ventajas de la arquitectura de 64 bits se relacionan con una mayor capacidad de procesamiento, y la mejora en la estabilidad y seguridad de la información.

y compatible hacia adelante con otros procesadores de 32 bits de la misma familia. El 68000 está basado en dos bancos de 8 registros de 32 bits: un banco es de datos, y el otro, de punteros. Además, posee un contador de programa de 32 bits, un registro de estado de 16 bits, y dos ALU diferentes para operar con datos y direcciones en forma independiente y simultánea.

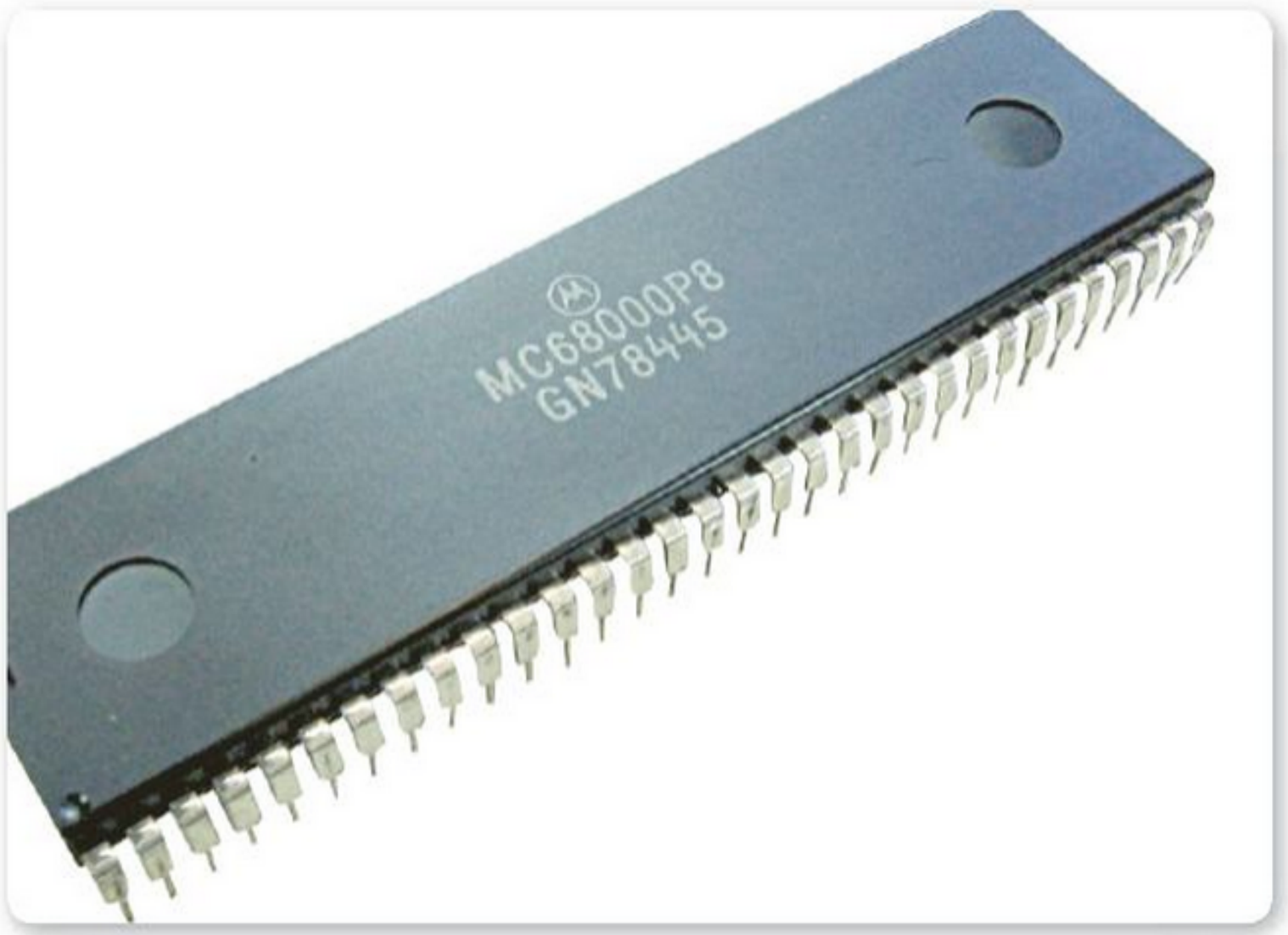


Figura 4. Microprocesador **Motorola MC68000** de 16 bits, 64 pines y tecnología HMOS fabricado actualmente por Freescale.

Los microprocesadores **8086** y **8088** son dos microprocesadores de 16 bits diseñados por Intel en 1978 e iniciadores de la arquitectura x86. La diferencia entre ambos es que el 8088 utiliza un bus externo de 8 bits para el empleo de circuitos de soporte al microprocesador más económicos en contraposición al bus de 16 bits del 8086.

Al igual que el 8086, el 80186 tiene un bus externo de 16 bits, mientras que el 80188 lo tiene de 8 bits, como el 8088, para hacerlo más económico. La velocidad de reloj es de 6 MHz para ambos microprocesadores. El uso está centrado básicamente en sistemas embebidos.

> EN UNA ARQUITECTURA DE 64 BITS, LAS DIRECCIONES DE MEMORIA SE COMPONENDE 8 BYTES

Figura 5. Los microprocesadores **80186** y **80188** fueron desarrollados por Intel alrededor de 1982 y son una mejora del Intel 8086/8088, respectivamente.

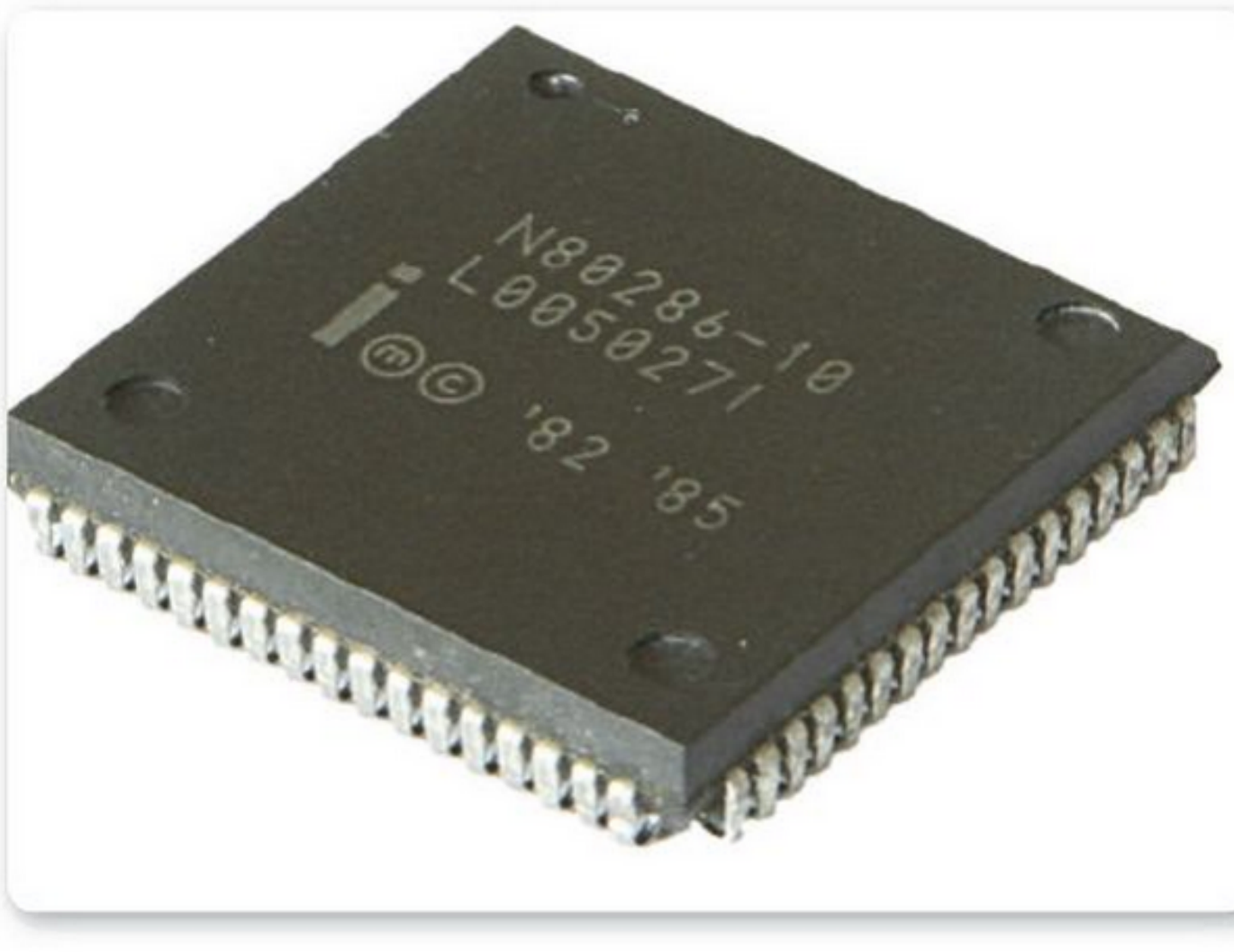


Figura 6. Microprocesador **Intel 80286** de 16 bits. Pertenece a la familia x86 y cuenta con 134.000 transistores.

El **Intel 80286** (llamado oficialmente **iAPX 286** y también conocido como **i286** o **286**) es un microprocesador de 16 bits de la familia

x86. Fue lanzado al mercado por Intel el 1 de febrero de 1982. Las versiones iniciales del i286 funcionaban a 6 MHz y a 8 MHz, aunque alcanzó una velocidad de hasta 20 MHz. Funciona al doble de velocidad que su predecesor, el Intel 8086, y puede direccionar hasta 16 MB de memoria RAM frente al MB del i8086. En máquinas DOS, esta memoria adicional solo podía ser accedida a través de emulación de memoria expandida, previamente habilitada la memoria extendida, mediante software. El i286 fue diseñado para ejecutar aplicaciones multitarea, que incluyen comunicaciones, control de procesos en tiempo real y sistemas multiusuario.

RESUMEN

El objetivo de este capítulo ha sido presentar al lector las tecnologías básicas en microprocesadores de 16 bits. Para ello, se analizaron las características fundamentales de estos microprocesadores, luego se examinaron las familias de microprocesadores más utilizadas tanto en la industria como en educación, y se focalizó en los microprocesadores más importantes de cada fabricante. También se realizaron referencias a los sistemas microprocesadores que utilizan arquitecturas de 32 y 64 bits.

Actividades

EJERCICIOS PRÁCTICOS

1 Convierta el número binario 1010111.0101 a su equivalente en hexadecimal y octal. Marque la respuesta correcta: 57.5_{16} __, $75.A_{16}$, 127.24_{8} __, 125.9_{8} __.

2 Profundice e investigue respecto a las extensiones de archivos de ensamblador más usuales.

3 Genere su propia explicación para los términos más usados en microprocesadores.

4 Realice la decodificación de BCD a 7 segmentos por software.

5 Seleccione varias instrucciones dentro del set de instrucciones de un microprocesador Motorola MC6800 y determine el modo de direccionamiento utilizado.

6 Seleccione varias instrucciones dentro del set de instrucciones de un microprocesador Zilog Z80 y determine el modo de direccionamiento utilizado.

7 Seleccione varias instrucciones dentro del set de instrucciones de un microprocesador Intel de 16 bits y determine el modo de direccionamiento utilizado.

8 Seleccione el microprocesador Intel 8085, examine el datasheet o la documentación del dispositivo, determine sus funciones, elementos, arquitectura y puertos disponibles.

9 Seleccione el microprocesador Intel 80286, examine el datasheet o la documentación del dispositivo, determine sus funciones, elementos, arquitectura y puertos disponibles.

10 Diseñe e implemente un proyecto sencillo que involucre la mayoría de los conceptos sobre microprocesadores desarrollados en el libro.

PROFESOR EN LÍNEA

Si tiene alguna consulta técnica relacionada con el contenido, puede contactarse con nuestros expertos: profesor@redusers.com.

CONÉCTESE CON LOS MEJORES
LIBROS DE COMPUTACIÓN

Consejos y secretos indispensables para ser un técnico profesional e implementar la solución más adecuada a cada problema.

>> HARDWARE / REDES
>> 320 PÁGINAS
>> ISBN 978-987-1949-19-9

LLEGAMOS A TODO EL MUNDO VÍA OCA* Y DHL**

MÁS INFORMACIÓN / CONTÁCTENOS

usershop.redusers.com +54 (011) 4110-8700 usershop@redusers.com

* SÓLO VÁLIDO EN LA REPÚBLICA ARGENTINA // ** VÁLIDO EN TODO EL MUNDO EXCEPTO ARGENTINA

Fundamentos de microprocesadores

El microprocesador es un dispositivo programable de propósito general que integra diversos productos: desde juguetes para niños hasta computadoras, dispositivos móviles inteligentes y equipos de ingeniería biomédica. Comprender las funcionalidades de un microprocesador constituye una base de conocimientos para estudiar microcontroladores y perfeccionarse en el diseño de aplicaciones que faciliten el trabajo y mejoren la calidad de vida de las personas.

Este libro propone un acercamiento al hardware y software de sistemas basados en microprocesadores, a partir del análisis de modelos de 8 y 16 bits, como paso previo a la comprensión del funcionamiento de microprocesadores más avanzados, de 32 y 64 bits.

En el comienzo de la obra se presenta la arquitectura de los sistemas de numeración, el álgebra de Boole y los dispositivos básicos a partir de los cuales se construyen los microprocesadores. Luego se identifican los conceptos fundamentales para la comprensión del funcionamiento de un microprocesador, junto con las interfaces microprocesador-memorias y E/S. Finalmente, se explica la programación en lenguaje ensamblador y se aplican los conocimientos adquiridos a dispositivos de 8 y 16 bits.

Con el objetivo de que los lectores alcancen un aprendizaje completo, el texto se complementa con ejemplos de aplicación, actividades de autoevaluación, ejercicios prácticos y contenido gráfico.

CONTENIDO

NIVEL DE USUARIO

- PRINCIPIANTE
- INTERMEDIO
- AVANZADO
- EXPERTO

RedUSERS

En nuestro sitio podrá encontrar noticias relacionadas y también participar de la comunidad de tecnología más importante de América Latina.

Ante cualquier consulta técnica relacionada con el libro, puede contactarse con nuestros expertos:

profesor@redusers.com

OTROS LIBROS DE ESTA COLECCIÓN

ISBN: 978-987-734-003-7