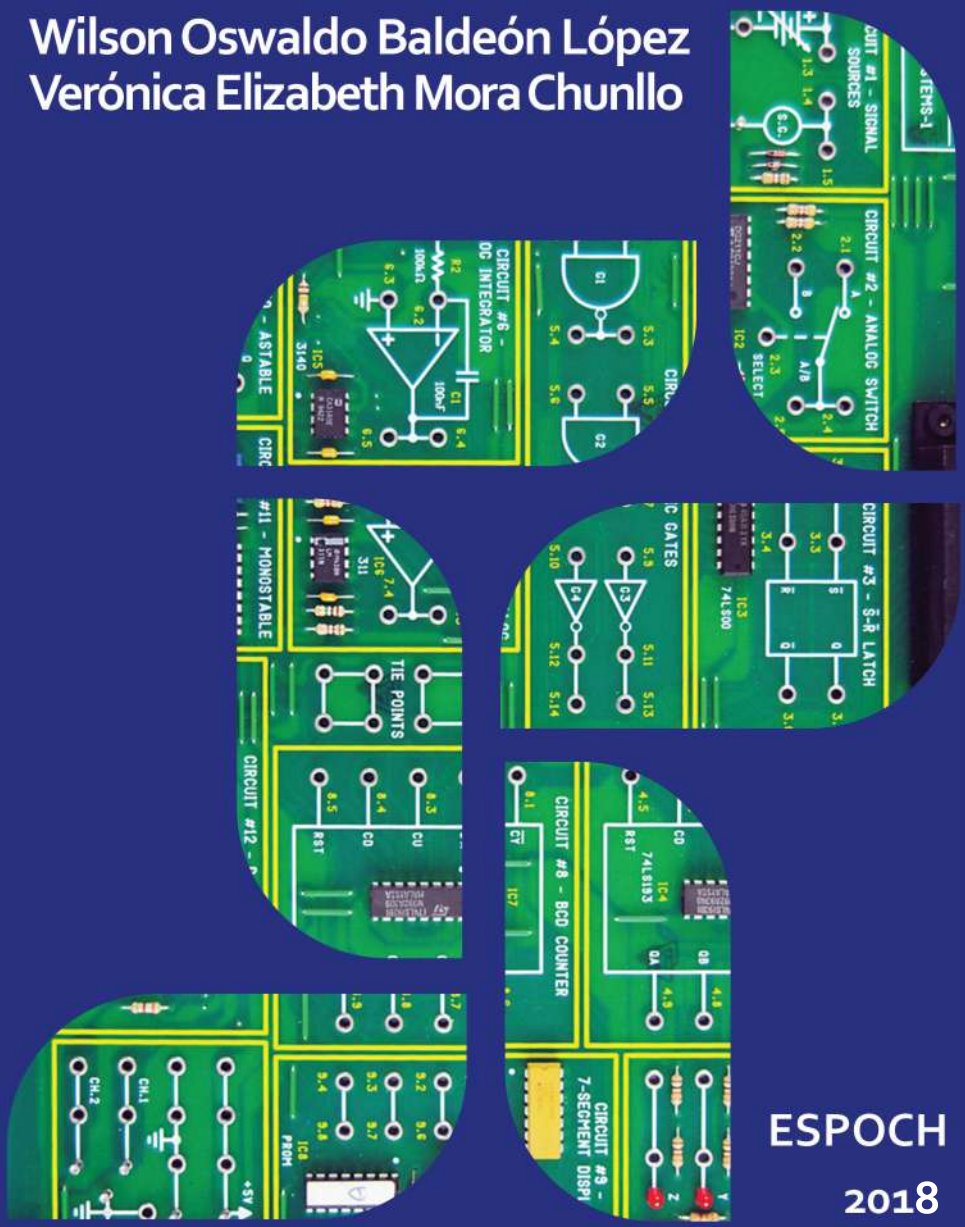
Sistemas digitales sincrónicos y VHDL

Latch y Flip-Flop

Wilson Oswaldo Baldeón López
Verónica Elizabeth Mora Chunllo



ESPOCH
2018

SISTEMAS DIGITALES SINCRÓNICOS Y VHDL

LATCH Y *FLIP-FLOPS*

SISTEMAS DIGITALES SINCRÓNICOS Y VHDL

LATCH Y *FLIP-FLOPS*

WILSON BALDEÓN
VERÓNICA MORA

DIRECCIÓN DE PUBLICACIONES

ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO
Fundada en 1972
Riobamba - Ecuador

SISTEMAS DIGITALES SINCRÓNICOS Y VHDL *LATCH* Y *FLIP-FLOPS*

Panamericana Sur, kilómetro 1 ½
Dirección de Publicaciones Científicas
Riobamba, Ecuador
Teléfono: 593 (03) 2 998-200
Código postal: EC0600155

Aval ESPOCH

Este libro se sometió a arbitraje bajo el sistema de doble ciego

(*peer review*)

Corrección y diseño

La Caracola Editores

Editorial Politécnica ESPOCH

Impreso en Ecuador

CDU: 004.3 + 004.4
SISTEMAS DIGITALES SINCRÓNICOS Y VHDL
LATCH Y *FLIP-FLOPS*
Riobamba: Escuela Superior Politécnica de Chimborazo
Dirección de Publicaciones, Año 2018
80 pp. vol: 17 x 24 cm
ISBN: 978-9942-35-650-5
1. Ciencia y tecnología de los ordenadores
2. *Hardware*. Componentes físicos del ordenador.
3. *Software*. Equipo lógico, componentes lógicos.

A Solange Baldeón

A Efraín Baldeón

A Sean Risley

CONTENIDO GENERAL

1. INTRODUCCIÓN A LOS CIRCUITOS SECUENCIALES SINCRÓNICOS.....8

1.1 Introducción 8
1.2 Circuitos combinacionales y secuenciales sincrónicos.....8
1.3 El elemento de memoria 15
1.4 *Latch* básico, celda binaria o báscula 16
1.5 Diagrama de tiempo para *set* y *reset* verdaderos 22
1.6 Aplicaciones de los latch básicos 24
1.6.1 El *latch* básico como eliminador de rebotes 24
1.7 El *flip–flop* activado por nivel o *latch* asincrónico.....26
1.7.1 Tipos de *latch* asincrónico.....28
1.8 El reloj O CLK 28
1.8.1 Tipos de reloj.....28
1.8.2 Reloj verdadero con nivel alto.....28
1.8.3 Reloj verdadero con nivel bajo.....30
1.8.4 Reloj verdadero con el flanco de subida.....31
1.8.5 Reloj verdadero con el flanco de bajada 31
1.9 Diseño de *latchs* asincrónicos.....32
1.9.1 El *latch* asincrónico tipo D 33
1.9.2 El *latch* asincrónico tipo T 39
1.9.3 El *latch* asincrónico tipo JK 43
1.9.4 El *latch* asincrónico tipo SR.....51
1.9.5 *latch* asincrónicos maestro-esclavo 56
1.9.6 *latch* asincrónicos maestro-esclavo JK.....58
1.10 Conversión entre *latchs* asincrónicos.....59
1.11 Los *flip–flop*s.....68
1.11.1 El *flip–flop* tipo D.....70
1.11.2 El *flip–flop* T.....72
1.11.3 El *flip–flop* JK 73

1.12 Retardos de propagación en *flip–flops* 74

2. EJERCICIOS PROPUESTOS 77

BIBLIOGRAFÍA 79

ACERCA DE LOS AUTORES

Wilson Oswaldo Baldeón López es ingeniero electrónico graduado en la Escuela Superior Politécnica del Litoral; es máster en Informática graduado en Chile, es máster en Diseño de Sistemas Electrónicos por la Universidad Tecnológicade La Habana, Cuba; es magíster en Gestión Académica Universitaria, tiene un diplomado superior en Pedagogía Universitaria y es experto en procesos *e-learning*.

Es miembro fundador del grupo de investigación GITCE, sus intereses de investigación son los sistemas digitales, el modelamiento y predicción del índice de radiación ultravioleta (IUV), ha publicado algunos textos básicos y varios artículos en revistas científicas; fue ingeniero de diseño digital en ELECSA; ha ganado varios premios.

Es docente titular en la Facultad de Informática y Electrónica de la Escuela Superior Politécnica de Chimborazo, fue autoridad académica y profesor titular en la Facultad de Ingeniería de la Universidad Nacional de Chimborazo, es Miembro de la Asociación Mundial de Tutores virtuales (ATM).

Verónica Elizabeth Mora Chunllo es ingeniera en Electrónica y Computación graduada en la Escuela Superior Politécnica de Chimborazo; es magíster en Ingeniería de *Software* graduada en la Escuela Superior Politécnica del Ejército; es máster en Diseño de Sistemas Electrónicos por la Universidad Tecnológica de La Habana, Cuba; es magíster en Educación a Distancia, tiene un diplomado superior en las Nuevas Tecnologías de Información y Comunicación Aplicado a la práctica Docente, es experta en procesos *e-learning*.

Es docente titular en la Facultad de Informática y Electrónica de la Escuela Superior Politécnica de Chimborazo; es directora y fundadora del grupo de investigación GITCE, sus intereses de investigación son los sistemas digitales. Es Miembro de la Asociación Mundial de Tutores virtuales (ATM), fue miembro de la Comisión Editorial de la revista científica *Perspectivas* FIE-ESPOCH; fue Subdirectora de Posgrado en la ESPOCH, ha publicado algunos textos básicos y varios artículos en revistas científicas nacionales.

PREFACIO

Este libro está basado en las experiencias académicas que los autores adquirieron en dos importantes universidades. Es el primero de cuatro libros que tratan sobre máquinas secuenciales sincrónicas.

Esta obra está escrita para un segundo curso de Sistemas Digitales de una carrera de Ingeniería Electrónica, Telecomunicaciones, Control o afín. En una sociedad como la de hoy, fuertemente digitalizada, un diseñador de sistemas debe dominar las técnicas de análisis y diseño de sistemas digitales combinacionales y secuenciales, las aplicaciones informáticas que permiten el diseño asistido por computadora (CAD) y un lenguaje de descripción de *hardware*.

La ciencia, en la que se fundamenta los sistemas digitales, en lo relativo a sus conceptos y fundamentos, no ha sufrido un cambio radical todavía, sin embargo, el diseño y aplicación práctica de esta ciencia (la implementación) ha cambiado sustancialmente en las últimas décadas con la aparición de los lenguajes de descripción de *hardware* (HDL) y herramientas CAD.

Este libro expone los conceptos fundamentales de los *latchs* y *flip-flops* desde una perspectiva clásica, pues su fundamento científico no ha cambiado. El diseño mediante HDL y herramientas CAD es tratado en el tercer libro de esta obra.

Existen infinidad de libros sobre sistemas digitales, sin embargo, este se adapta a las necesidades académicas y de laboratorios de las carreras de Ingeniería Electrónica de la ESPOCH. De ahí que uno de los objetivos de este libro es resolver estas necesidades.

El texto presenta su contenido de una forma en que el lector puede desarrollar habilidad intuitiva para entender y aplicar los conceptos fundamentales, la estructura, el análisis, el diseño y la conversión tanto de *latchs*, como de *flip-flops*. Se exponen ejemplos que permiten reforzar los conocimientos que el lector va adquiriendo.

1. INTRODUCCIÓN A LOS CIRCUITOS SECUENCIALES SINCRÓNICOS

1.1 Introducción

En la academia, para su estudio, los sistemas digitales suelen dividirse en dos partes. En la primera se estudian los circuitos combinacionales y, en la segunda, los circuitos secuenciales sincrónicos y asincrónicos. Este libro presenta exclusivamente el diseño de circuitos secuenciales sincrónicos.

1.2 Circuitos combinacionales y secuenciales sincrónicos

El estudio de los circuitos secuenciales sincrónicos, en este libro, se inicia mostrando las características que los circuitos combinacionales y secuenciales presentan.

Los circuitos digitales están construidos con compuertas lógicas interconectadas entre sí de alguna manera. Los valores lógicos de las salidas de estos circuitos están dados por ecuaciones booleanas.

En todo circuito digital, cada compuerta lógica tiene un retardo de propagación. Este retardo es el tiempo que tarda en cambiar el valor del nivel lógico que se encuentra en la salida de una compuerta cuando alguna de sus entradas ha cambiado.

Cuando no se considera el retardo de propagación, se asume que este es igual a cero y, en tal caso, la compuerta es ideal.

Desde el punto de vista de la velocidad de respuesta, en un circuito con compuertas idealizadas, da igual que el circuito tenga una sola compuerta o muchas compuertas interconectadas, porque se asume que los valores en las salidas aparecen instantáneamente cuando alguna de sus entradas ha cambiado.

Para analizar, una de las características de un circuito combinacional se toma como ejemplo una compuerta lógica AND (Y) de dos entradas y una salida. La característica fundamental de esta compuerta lógica, de acuerdo a su tabla de funcionamiento, es que el valor de su salida depende exclusivamente de los valores que tenga la puerta AND en sus entradas y de nada más. La figura 1.1. a) y b) muestra una compuerta AND y su tabla de verdad.

Por otro lado, si se toma en consideración el retardo de propagación (RP) de una compuerta AND (figura 1.1. c), esta compuerta, puede ser representada mediante una compuerta ideal, sin retardo, más una memoria virtual (conformada por el RP).

La memoria virtual aparece debido a que el valor lógico en la salida de esta compuerta se mantiene sin cambiar durante un tiempo igual al retardo de propagación, a pesar de que los valores en las entradas ya han cambiado.

En el diagrama de tiempo que muestra la figura 1.1. d), se puede ver el efecto del RP. AB es la señal que se aplica en las entradas, A y B, de la compuerta, la señal *i* es la señal de salida de la compuerta ideal y C es la señal de salida de la compuerta real.

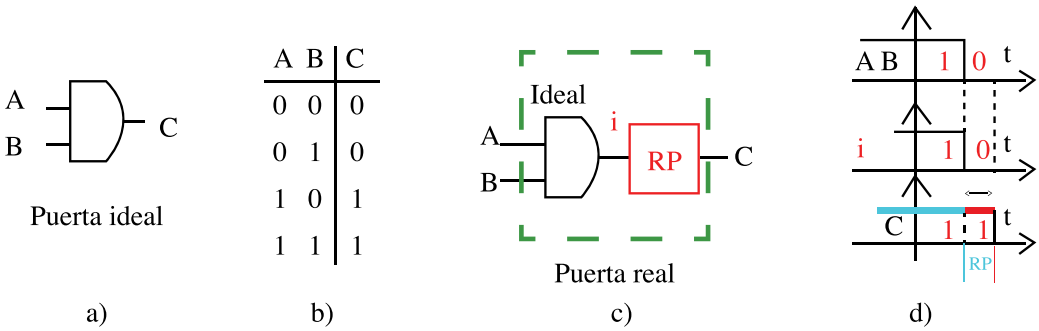


Figura 1.1. Circuito combinacional simple

Un circuito secuencial tiene realimentación. Para entender este concepto, la figura 1.2. muestra un ejemplo de un circuito digital realimentado.

Una señal de realimentación o *feedback*, es una señal que tiene su origen en una salida y se dirige al sistema de entradas del circuito. En la figura 1.2, S2 es la señal de salida de la compuerta P2 que realimenta la entrada de la compuerta P1.

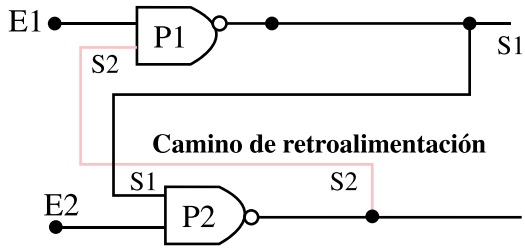


Figura 1.2. Circuito con retroalimentación o *feedback*

Si la identificación de una señal realimentada en un circuito es difusa, se dibuja el circuito de otra manera, por ejemplo, el circuito de la figura 1.2. se vuelve a dibujar como muestra la figura 1.3, allí se puede ver, con claridad, la señal de realimentación.

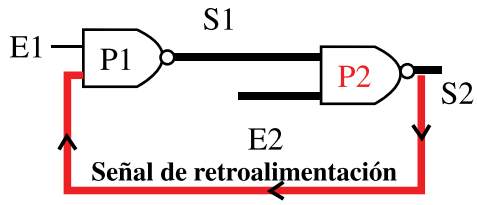


Figura 1.3. El circuito realimentado dibujado de otra manera.

En general, un sistema secuencial digital puede tener varios caminos de retroalimentación o realimentación. La figura 1.4. muestra un sistema con esta característica. En esta figura, el bloque superior, denominado de lógica de transformación y mezcla de las entradas y salidas recibe tanto las señales de entrada del mundo exterior, como, las salidas del bloque intermedio, denominado elemento de memoria; estas son las señales de salida que se retroalimentan y se combinan con las entradas al sistema. El tercer bloque, denominado de la lógica de las salidas condicionales se encarga, de generar las salidas del sistema.

El diagrama de la figura 1.4 representa un sistema secuencial digital general, en este diagrama, algunas características son evidentes.

1. La presencia de memoria
2. La presencia de realimentación

Por otro lado, los circuitos combinaciones tienen las características siguientes:

1. La ausencia de un dispositivo fisico de memoria.
2. La ausencia de caminos de retroalimentación.
3. Los valores en las salidas son función exclusiva de los valores en las entradas.

Respecto de la memoria virtual, esta debe ser entendida como la capacidad de memoria que tienen las compuertas lógicas debido a los retardos de propagación.

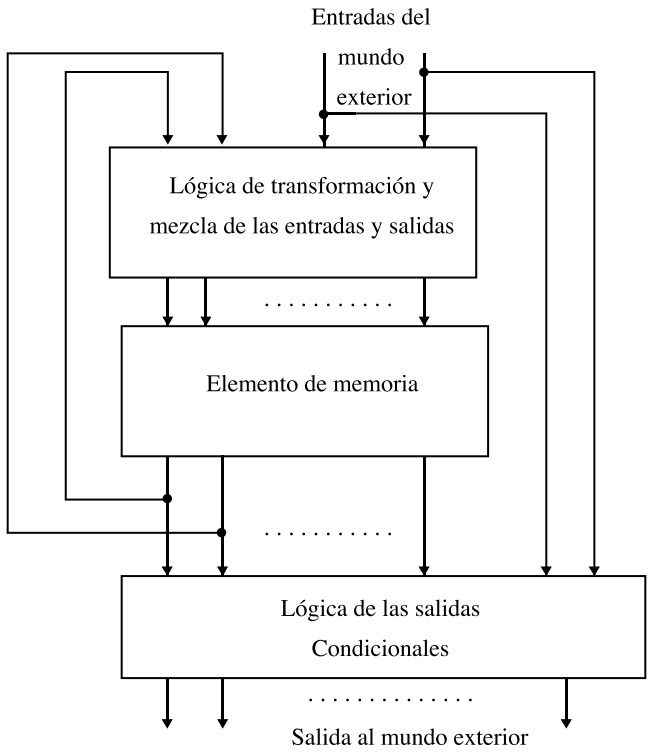


Figura 1.4 Diagrama de bloques de un sistema digital

A continuacion, se muestran algunos ejemplos de sistemas realimentados.

Ejemplo 1.1

Para el diagrama de bloques del sistema que se muestra en la figura 1.5, indique ¿cuántos caminos de realimentación existen?

Hay un solo camino de realimentación y va desde la salida del bloque dos a la entrada del bloque uno.

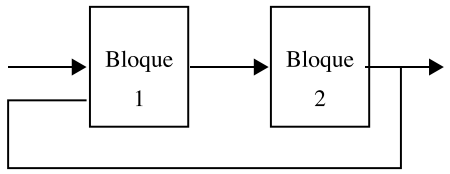


Figura 1.5 Diagrama de bloques de un sistema

Ejemplo 1.2

Para el diagrama de bloques del sistema que se indica en la figura 1.6, indique ¿cuántos caminos de retroalimentación existen?

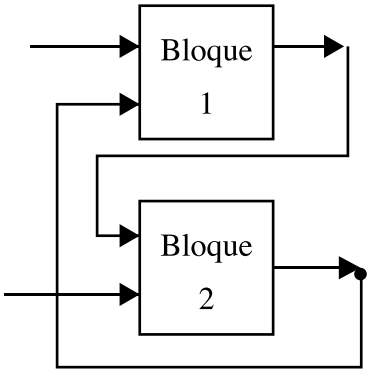


Figura 1.6 Diagrama de bloques para el ejemplo 1.2

Hay un camino de retroalimentación y va desde la salida del bloque dos a la entrada del bloque uno. Para distinguir de mejor manera los caminos realimentados se dibuja el sistema, de otra manera, como muestra la figura 1.7.

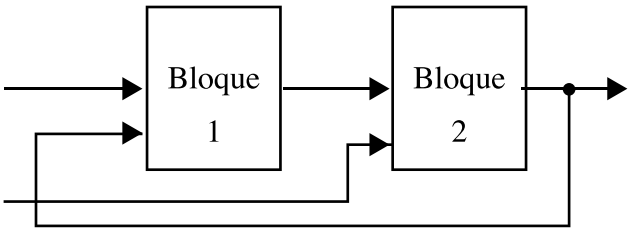


Figura 1.7 Diagrama de bloques modificado para el ejemplo 1.2.

Ejemplo 1.3

Identifique los caminos de realimentación que tiene el sistema que se muestra en la figura 1.8.

De la figura 1.8, se puede determinar que solo hay un camino de realimentación, puesto que el bloque tres es una interfaz que permite la realimentación de la salida del bloque 2 al sistema de entradas del bloque 1.

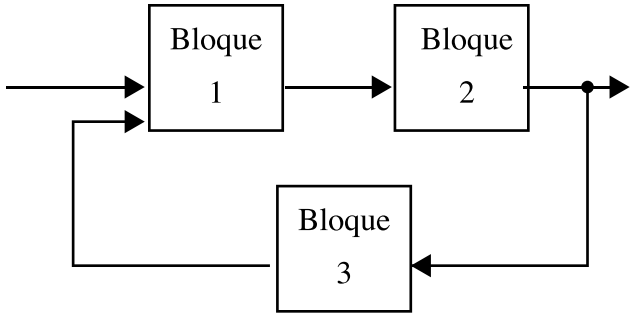


Figura 1.8. Diagrama de bloques de un sistema.

Ejemplo 1.4

Identifique los caminos de realimentación que tiene el sistema que se muestra en la figura 1.9.

El sistema de la figura 1.9 tiene dos caminos de realimentación, se debe observar que es el sistema del ejercicio anterior añadido el bloque cuatro que contiene un camino de realimentación.

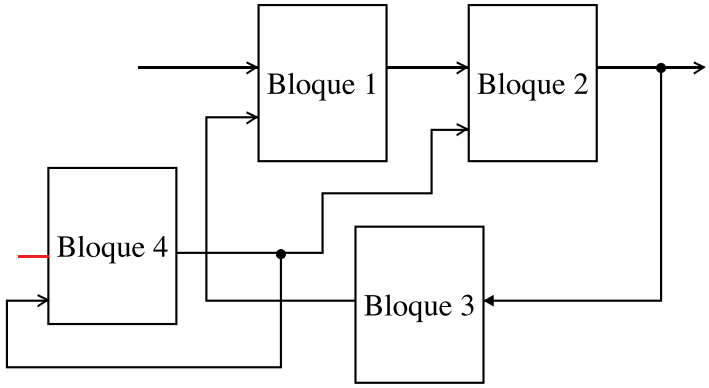


Figura 1.9. Diagrama de bloques para el ejercicio 1.4.

Retornando al estudio de las características de los circuitos combinaciones y secuenciales, la figura 1.10 muestra la relación que existe entre la salida S y las entradas E_1 y E_2.

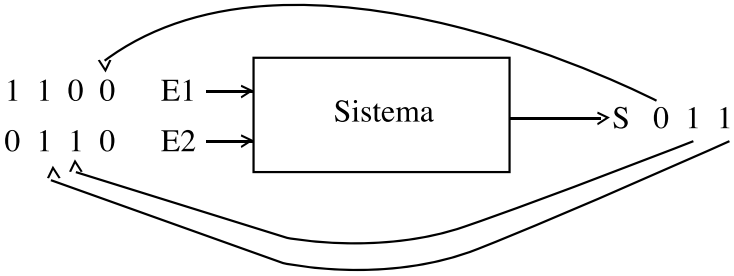


Figura 1.10. Representación gráfica de un sistema combinacional

Las flechas indican que la salida S solo depende del valor que tienen las entradas E_1 y E_2 en un instante de tiempo y de nada más

Tiempo	E1	E2	S
t0	0	0	0
t1	0	1	1
t2	1	1	1
t3	1	0	0

Tabla 1.1. Valores de las entradas E1, E2 y la salida S

La tabla 1.1 resume los valores hipotéticos que van tomando las entradas y la salida al transcurrir el tiempo.

Como muestra la tabla 1.1, las entradas E1 y E2 en el tiempo t0 reciben los valores 00 y el sistema genera la salida S igual a 0, por tanto, S solo depende del valor que las entradas tengan en el tiempo t0.

Luego, las entradas cambian a 01, en el tiempo t1, y la salida S que vale 1 solo depende de los valores presentes en las entradas en ese tiempo t1.

En seguida, las entradas cambian a 11 en el tiempo t2 y la salida S que vale 1 solo depende de los valores presentes en las entradas en ese instante t2.

A continuación, las entradas en el tiempo t3 cambian a 10 y la salida S cambia a 0, igual que en los casos anteriores, S, solo depende de los valores presentes en las entradas y el sistema responde de esta manera en forma indefinida, así es

como se comporta un circuito combinacional, la o las salidas dependen solo de los valores de las entradas.

En la figura 1.11, en cambio, se muestra un ejemplo en forma de un diagrama de bloque del posible funcionamiento de un sistema secuencial.

Las entradas E1 y E2 en el tiempo t0 reciben las entradas 00 respectivamente y el sistema genera la salida S igual a 0, este valor de esta salida solo depende de los valores que las entradas tengan al tiempo t0, ya que se supone que el sitema arranca al tiempo t0.

Luego las entradas cambian a 01, en el tiempo t1, y la salida S tiene ahora el valor 1, que depende de los valores de las entradas presentes (01) en ese tiempo t1, así como de los valores anteriores que en E1 y E2 respectivamente estuvieron en el tiempo t0 y que fueron 00.

A continuación, las entradas cambian a 11 en el tiempo t2 y la salida S tiene ahora el valor 1, que depende de los valores de las entradas presentes (11) en ese tiempo t2, además de los valores anteriores de E1 y E2 que en el tiempo t0 fueron 00, en el tiempo t1 fueron 01 respectivamente.

En el tiempo t3 las entradas E1 y E2 cambian a 10 y la salida toma el valor de 0 que depende de los valores de las entradas presentes (10) en ese tiempo t3, además de todos los valores anteriores de E1 y E2 que en el tiempo t0 fueron 00 en el tiempo t1 fueron 01 y en el tiempo t2 fueron 11 en E1 y E2 respectivamente y el sistema continúa respondiendo de esta manera.

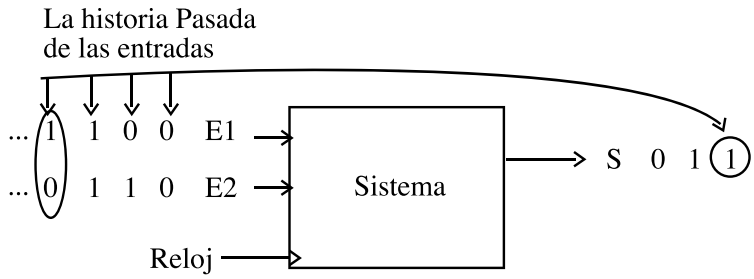


Figura 1.11. Representación gráfica de un sistema secuencial.

1.3 El elemento de memoria

El elemento de memoria más simple, construido con compuertas lógicas, almacena un solo bit de información en su salida y se llama *latch* básico, celda binaria o báscula.

1.4 *Latch* básico, celda binaria o báscula

Es un circuito construido con dos compuertas lógicas en acople cruzado, es realimentado y su función es almacenar un solo bit de información en su salida, ya sea un "uno lógico" o un "cero lógico".

La figura 1.12 muestra el diagrama de bloque de un *latch* básico. Tiene dos entradas, la una se denomina *SET* y la otra *RESET* y, dos salidas Q y /Q.

En este libro la barra oblicua, diagonal o *slash* cuyo símbolo es: "/", representa a una variable negada, por ejemplo, si A es una variable, entonces, /A es la negación de A.

Figura 1.12. Diagrama de un *latch* básico

En sistemas digitales un *latch* básico se construye mediante compuertas lógicas y hay dos formas de construirlo.

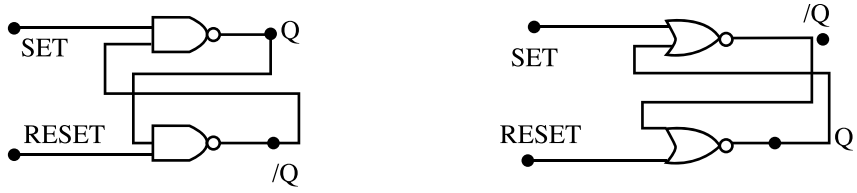
La figura 1.13 a) y b) muestra un *latch* básico construido con compuertas NAND en acople cruzado y con compuertas NOR, también en acople cruzado. Cada *lach* básico es realimentado.

Cuando la entrada *SET* es verdadera, la salida Q adquiere el valor verdadero, siempre que la entrada *RESET* sea falsa, y cuando la entrada *RESET* es verdadera, la salida Q se pone en su condición de falsa, siempre que la entrada *SET* sea falsa.

En resumen, para almacenar un uno lógico el *SET* (*set* implica poner a uno) tiene que ser verdadero y el *RESET* falso.

Para poner la salida Q a cero, la entrada RESET debe ser verdadera y falsa la entrada *SET*; la tabla 1.2 resume la operación del *latch* básico.

Los *latch* básicos, con compuertas NAND, responden sus entradas a lógica negativa, y los *latch* básicos con compuertas NOR responden sus entradas a lógica positiva.



a) NAND de acople cruzado. b) NOR de acople cruzado.

Figura 1.13. Dos formas de construir un *latch* básico

Set	*Reset*	Salida Q
F	F	Q, mantiene el valor de Q
F	V	0, OPERACIÓN DE *RESET*
V	F	1, OPERACIÓN DE *SET*
V	V	No permitido

Tabla 1.2 Indica el funcionamiento de un *latch* básico

La tabla 1.3 muestra la operación de un *latch* básico con compuertas NAND que trabajan con lógica negativa, donde un valor verdadero se representa con un cero lógico y un valor falso con un uno lógico.

Set	*Reset*	Salida Q
F=1	F=1	Q, mantiene el valor de Q
F=1	V=0	0, OPERACIÓN DE *RESET*
V=0	F=1	1, OPERACIÓN DE *SET*
V=0	V=0	No permitido

Tabla 1.3. Operación de un *latch* básico construido con compuertas NAND

La tabla 1.4 muestra la operación de un *latch* básico con compuertas NOR. Como ya se indicó, trabaja sus entradas con lógica positiva, esto implica que un valor verdadero se representa con un uno lógico y un valor falso con un cero lógico.

Set	*Reset*	Salida Q
F=0	F=0	Q, mantiene el valor de Q
F=0	V=1	0, OPERACIÓN DE *RESET*
V=1	F=0	1, OPERACIÓN DE *SET*
V=1	V=1	No permitido

Tabla 1.4. Operación de una *latch* básico construido con compuertas NOR.

Analizando el contenido de las tablas 1.3 y 1.4 se puede concluir que la salida Q tiene los mismos valores en las dos tablas; sin embargo, las entradas *SET* y *RESET* puestas en su respectiva lógica (positiva o negativa) cambian porque funciona en el primer caso con lógica negativa, y en el segundo, con lógica positiva; por lo tanto, las entradas a las tablas en función de ceros y unos lógicos son diferentes, pero realizan la misma operación o función.

El funcionamiento de un *latch* básico en función de valores verdaderos y falsos es siempre el mismo e independiente del tipo de lógica; es decir, cuando se realiza una operación de *SET*, este debe ser verdadero (*set*=v) y la salida Q se hace igual a 1 lógico, *set* significa poner, colocar, poner en la salida Q un uno lógico.

Cuando se realiza una operación de *RESET*, éste debe ser verdadero (*reset*=v) y la salida Q se hace igual a cero.

Cuando no se realiza una operación de *SET* y tampoco una de *RESET*, el *SET* debe ser falso (indica que no debe guardar un uno en Q) y el *RESET* también (indica que no debe guardar un cero en Q), en otras palabras, se está indicando al *latch* básico que no haga nada; es decir, el valor de Q no debe cambiar, se debe mantener.

En resumen, es importante mantener presente la tabla de operación general (tabla 1.2) de un *latch* básico y cómo a partir de esta tabla se puede construir las tablas de lógica positiva y negativa del *latch* básico. Las tablas 1.3 y 1.4 responden a lógica negativa y a lógica positiva respectivamente.

Otro punto importante es también notar que en un *latch* básico con compuertas NAND, la entrada *SET* está en la misma compuerta que contiene la salida **Q**; en cambio, en un *latch* básico con compuertas NOR, la entrada *SET* está en la misma compuerta que contiene la salida /Q.

Ejemplo 1.5

Demuestre la operación de una *latch* básico con compuertas NAND y explique claramente los resultados obtenidos.

Para demostrar la operación de un *latch* básico con compuertas NAND un camino es encontrar la tabla de verdad de funcionamiento de esta celda. Para lograr este propósito se debe primero encontrar las ecuaciones booleanas para las salidas Q y /Q y luego elaborar la tabla de verdad de estas ecuaciones.

En la figura 1.14 se muestra un *latch* básico con compuertas NAND. Para que queden identificadas con claridad todas las entradas y salidas que intervienen en este *latch* básico, se representa la celda como un bloque. La figura 1.15 muestra este bloque.

De la figura 1.15, se ve claramente que existen cuatro entradas al bloque y son: *SET*, *RESET*, Q y /Q, las dos últimas también son entradas debido a que son realimentadas hacia el sistema de entradas; sin embargo, Q y /Q, los dos, son redundantes entre sí debido a que la una es la negada de la otra y por lo tanto se puede considerar a una sola de ellas a Q por ejemplo.

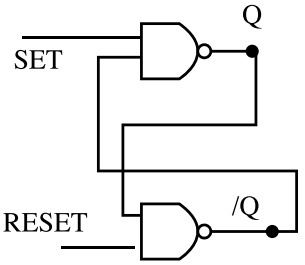


Figura 1.14. *Latch* básico con compuertas NAND.

Así, el sistema tiene tres entradas *SET*, *RESET* y Q, y dos salidas Q y /Q. Con este análisis en mente se escriben las ecuaciones booleanas y se construye la tabla de verdad. Como son tres entradas se tiene una tabla de verdad de filas.

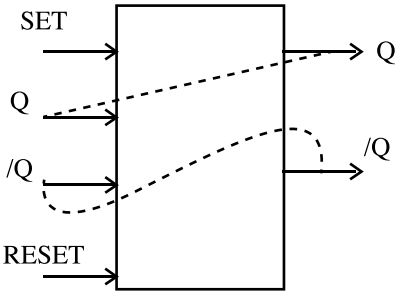


Figura 1.15. Diagrama de bloques de la *latch* básico con compuertas NAND

De la figura 1.14 se obtienen dos ecuaciones:

1. Q= /(*SET*./Q)
2. /Q=/(Q.*RESET*)

Remplazando /Q de (2) en (1), se tiene: Q= /(*SET*./(Q.*RESET*)).

SET	*RESET*	Q		*/SET*	Q.*RESET*	Q=/*SET* + (Q.*RESET*)
0	0	0		1	0	1
0	0	1		1	0	1
0	1	0		1	0	1 operación de
0	1	1		1	1	1 SET
1	0	0		0	0	0 operación de
1	0	1		0	0	0 RESET
1	1	0		0	0	0
1	1	1		0	1	1

Tabla 1.5. *Latch* básico con compuertas NAND.

Q= /*SET* + (Q.*RESET*). La tabla 1.5 representa a estas ecuaciones

La tabla 1.6 resume de mejor manera la tabla 1.5.

SET	*RESET*	Q	Q=/*SET* + (Q.*RES*
0	0	0	1 operación
0	0	1	1 deseadas
0	1	0	1 operación
0	1	1	1 de set
1	0	0	0 operación de *RESET*
1	0	1	0 *RESET*
1	1	0	0 Operación de
1	1	1	1 mantenimiento

Tabla 1.6. Resumen de la tabla 1.5

Como se puede observar en la tabla 1.6, la salida Q se encuentra en los dos lados de la tabla, como entrada y como salida. Este fenómeno se debe a que la salida Q, que está como entrada en la tabla de verdad, representa el valor actual de la salida Q, es decir antes de que se actualicen las entradas SET y RESET, por esta razón se le denomina Qn, que es la representación del estado actual o estado presente de Q.

La salida Q, que está como salida en la tabla de verdad representa el estado al que va la salida Q cuando se han actualizado los valores en las entradas *SET* y *RESET*, por esta razón se le llama el estado siguiente de Q y se la representa como Qn+1.

Entonces la salida Q de un *latch* básico representa tanto al valor actual, estado presente o Qn, como al valor siguiente o Qn+1 al cual va la salida Q cuando se han actualizado los valores en las entradas *SET* y *RESET*. En la figura 1.16, se representa esta situación.

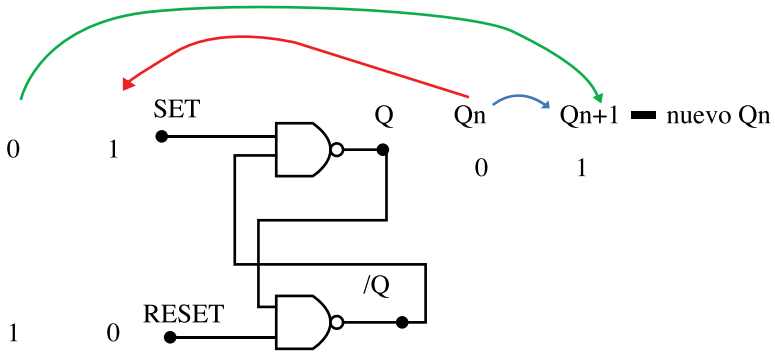


Figura 1.16. Representación de Qn y Qn+1.

En la figura 1.16 se supone que el estado presente Qn es 0, puesto que *SET* = 1 y *RESET* = 0. Ahora si se actualizan los valores de las entradas *SET* y *RESET* a 0 y 1 respectivamente, Q cambia a su nuevo valor (estado siguiente), es decir, Qn+1=1. Este estado siguiente es entonces, ahora, el nuevo estado presente, Qn.

Esta nomenclatura se utiliza para construir la tabla de verdad que se indica en la tabla 1.16.

Y la ecuación queda:

$$Qn+1 = /SET + (Qn.RESET) \quad (1)$$

1.5 Diagrama de tiempo para *SET* y *RESET* verdaderos

Para analizar la indeterminación de la salida Q de un *latch* básico, cuando *SET* y *RESET* son los dos verdaderos al mismo tiempo, se propone el diagrama de tiempo de la figura 1.18. Las compuertas reales tienen un retardo de propagación, que se define como el tiempo que tarda la salida de la compuerta en responder a cambios en sus entradas.

Para el primer caso, se supone un *latch* básico que funciona con lógica negativa y la compuerta a la que se encuentra conectada la señal *RESET*, tiene un retardo de propagación de ocho nanosegundos y la otra compuerta, 16 nanosegundos. La celda dibujada de otra manera muestra la figura 1.17.

Se supone que al tiempo t= 8, las señales *SET* y *RESET* son verdaderas al mismo tiempo, es decir las dos caen a cero voltios o nivel bajo.

Antes del tiempo t=0, se supone que: Q=0, /Q=1, *SET*=1 y *RESET*=0.

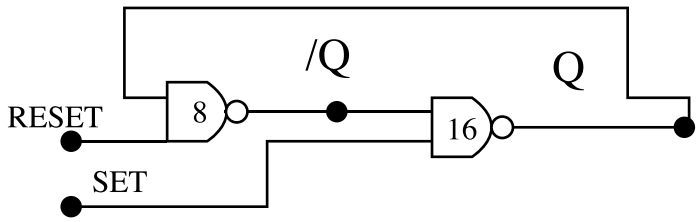


Figura 1.17. *Latch* básico y retardos de propagación.

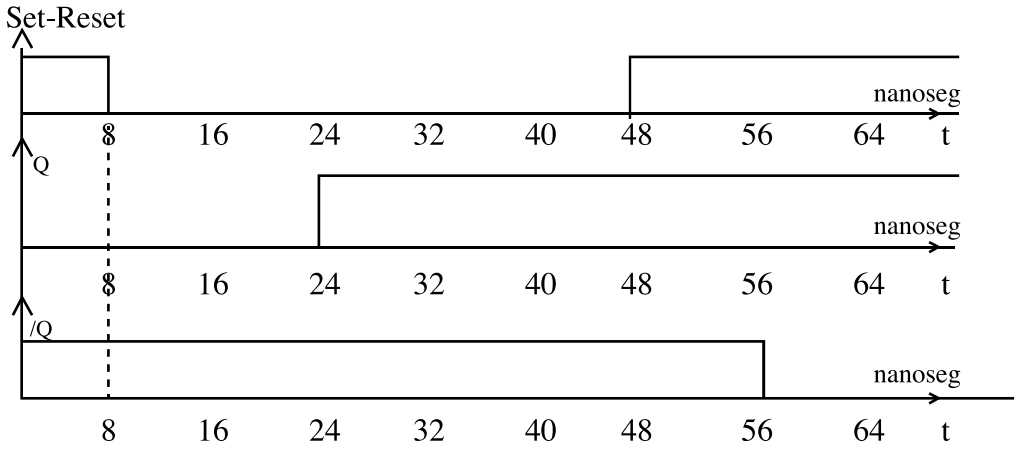


Figura 1.18. Diagrama de tiempo para *SET* y *RESET* verdaderos.

Para el segundo caso, la compuerta a la que se encuentra conectada la señal *RESET*, tiene un retardo de propagación de 16 nseg, y la otra compuerta, 8 nseg. La figura 1.19 muestra el *latch* básico dibujado de otra manera y el diagrama de tiempo se muestra en la figura 1.20.

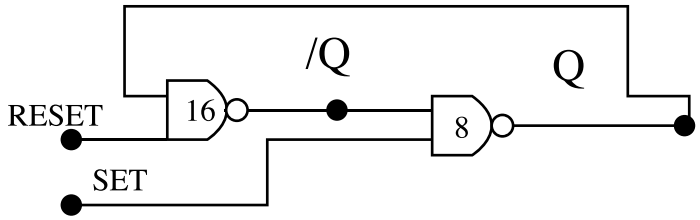


Figura 1.19. *Latch* básico y retardos de propagación.

Se supone que al tiempo t= 8, las señales *SET* y *RESET* son verdaderas al mismo tiempo, es decir las dos caen a cero voltios o nivel bajo. Antes del tiempo t=0 las condiciones son: Q=0, /Q=1, *SET*=1 y *RESET*=0.

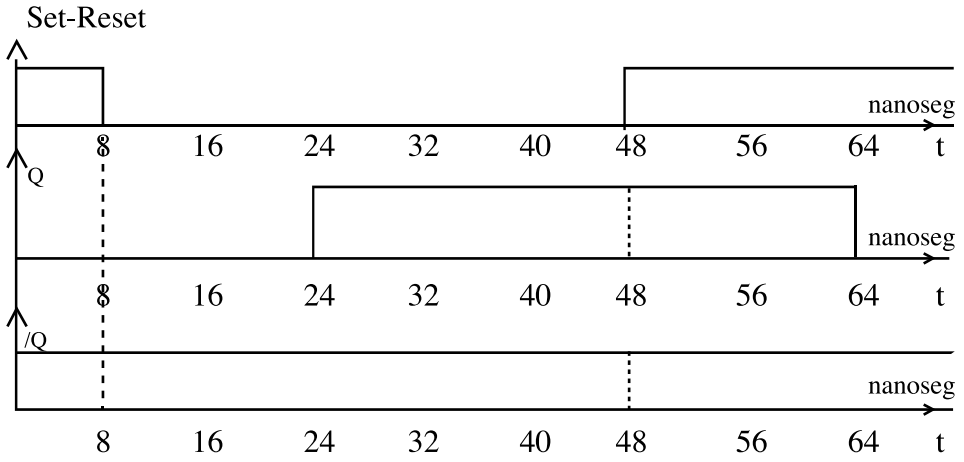


Figura 1.20. Diagrama de tiempo para *SET* y *RESET* verdaderos.

Se puede concluir, del análisis de los dos diagramas de tiempo, que cuando las dos entradas de un *latch* básico son verdaderas al mismo tiempo, el valor que toma Q no se puede predecir porque depende del retardo de propagación de las compuertas, y los retardos de las compuertas difícilmente son iguales. Incluso, es difícil determinar de antemano cuál de las compuertas tendrá el mayor o menor retardo, en definitiva el estado de Q bajo la condición estudiada es incierto y depende de la tecnología de construcción de las compuertas.

1.6 Aplicaciones de los *latch* básicos

Las celdas binarias o *latch* básicos tienen varias aplicaciones. Aquí se muestran tres.

1. Elemento de memoria
2. Eliminador de rebotes
3. *Latch* asincrónicos

La operación como elemento de memoria ya se describió. Como eliminador de rebotes, se discute en el siguiente apartado y el uso en la construcción de *latch* asincrónicos se describirá más adelante.

1.6.1 El latch básico como eliminador de rebotes

Un rebote se produce cuando se manipula un interruptor mecánico. Por ejemplo, en la figura 1.21 se muestra un interruptor que permite abrir o cerrar un circuito.

El voltaje de salida es cero cuando el interruptor está en la posición indicada en la figura 1.21.

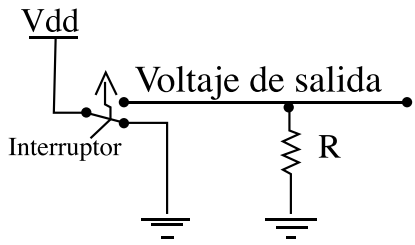


Figura 1.21. Interruptor abierto.

En la figura 1.22 a) se puede observar al interruptor cerrado y, en la figura 1.22 b), la forma de onda que se genera al cerrar el interruptor. Debe observarse que el interruptor en el instante en que se cierra genera un tren de niveles de voltajes altos y bajos que luego de unos pocos milisegundos se estabiliza en el nivel deseado, en este caso en el nivel alto, a esta etapa en donde se ha generado ese tren de pulsos se llama rebote de la señal.

Un interruptor ideal debe generar un solo nivel cuando se abre o cierra, pues el objetivo de un interruptor, en sistemas digitales, es poner un solo nivel en la entrada de un circuito digital y no un conjunto de niveles.

Todo interruptor mecánico presenta rebote y es necesario eliminarlo.

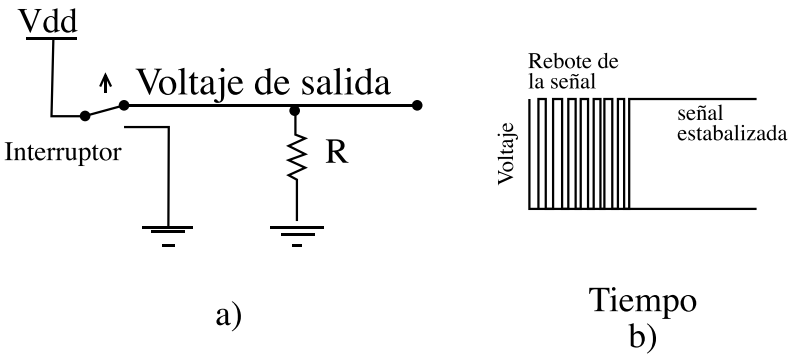


Figura 1.22. Rebote de la señal en un interruptor.

Un *latch* básico se puede utilizar como un dispositivo para eliminar los rebotes producidos por un interruptor mecánico.

El circuito de la figura 1.23 coloca el *SET* o *RESET* en nivel bajo, si el interruptor está en la posición como indica la figura 1.23 a), el *RESET* está en nivel alto y el *SET*, en bajo, por lo tanto, la salida Q va a nivel alto, se supone que es un *latch* básico con compuertas NAND.

Cuando el interruptor por efecto del rebote queda desconectado de la entrada *SET*, esta entrada va a nivel alto quedando las dos entradas *SET* y *RESET* momentáneamente en nivel alto y, por lo tanto, la salida Q mantiene el último nivel, el alto.

Para el caso de la figura 1.22 b), la entrada *SET* está en nivel alto y *RESET* está en nivel bajo y por lo tanto la salida Q va a nivel bajo.

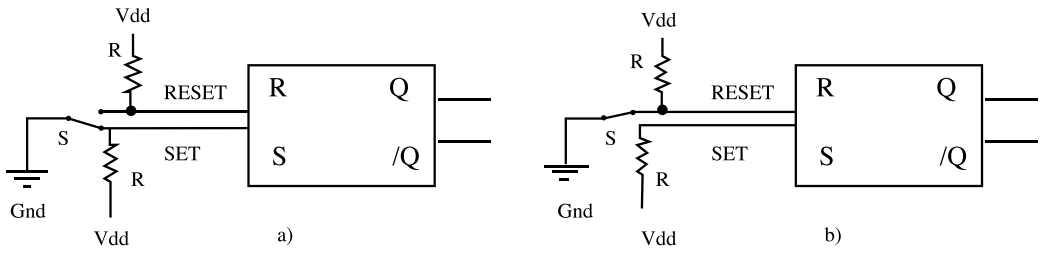


Figura 1.23. a) *SET* verdadero, b) *RESET* verdadero.

Otra forma de construir circuitos eliminadores de rebote es mediante inversores. La figura 1.24 muestra dos circuitos de este tipo.

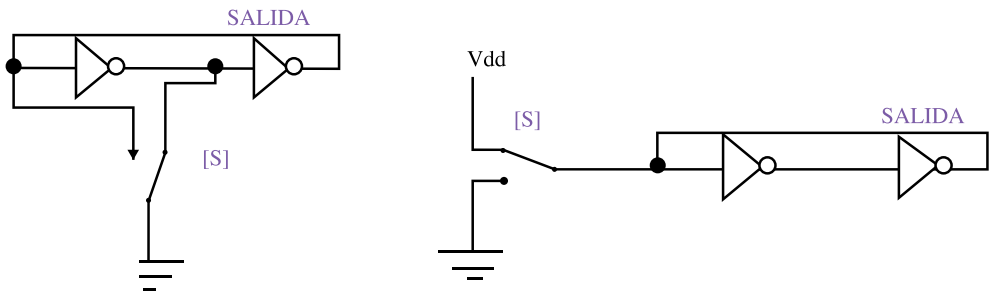


Figura 1.24. Inversores conectados como eliminadores de rebote

1.7 El *FLIP–FLOP* activado por nivel o *latch* asincrónico

Un *latch* básico almacena un bit en su salida Q. El valor de la salida Q depende de los valores que tengan sus entradas *SET* y *RESET*. Hay dispositivos que trabajan en una forma parecida a un *latch* básico, pero con una gran diferencia, y esta diferencia está en que estos dispositivos, actualizan y guardan un bit en su salida Q, si y solamente si, una señal especial de control les da la orden de actualizar y guardar. Vale recalcar que, si esta señal de control no da la orden, la salida Q no se actualiza pero mantiene guardado el bit antiguo en su salida Q.

La señal de control tiene la función de indicarle al dispositivo en que instante debe actualizar el valor de la salida Q. El valor al cual se actualiza la salida Q, un cero o un uno, va a depender de los valores que el dispositivo tenga en sus entradas *SET* y *RESET*. A estos dispositivos se les denomina *latchs* asincrónicos o *flip-flops* sensibles o activados por nivel.

Estos dispositivos guardan un bit en la salida Q de un *latch* básico, por lo tanto, todo *latch* asincrónico tiene en su interior un *latch* básico. Es más, las salidas Q y /Q del *latch* asincrónico son las mismas salidas, Q y /Q, del *latch* básico.

La señal de control en este tipo de dispositivos puede ser un interruptor, pero normalmente es una señal de voltaje periódica, con niveles altos y bajos, que se llama reloj o también conocida como *clock* (CLK), de sus siglas en inglés.

El diagrama de bloque general de un *flip-flop* activado por nivel o *latch* básico se muestra en la figura 1.25. Como se puede ver consta de n entradas (E1 a En), la señal de reloj (control) y dos salidas Q y /Q.

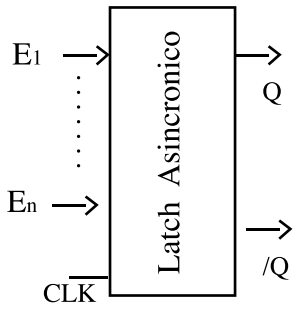


Figura 1. 25. Diagrama de bloque general de un *latch* asincrónico.

El diagrama de bloques, a diferencia del diagrama de bloque general del *latch* asincrónico muestra con más detalle el interior del mismo. En el consta el *latch* básico y el decodificador de *SET* y *RESET*, este diagrama se muestra en la figura 1.26.

Las n entradas del *latch* asincrónico definen el valor (uno o cero) que el dispositivo va a almacenar en su salida Q y el reloj el instante en que realiza esa acción. Como se indicó, el reloj actualiza la salida Q y por supuesto la salida /Q.

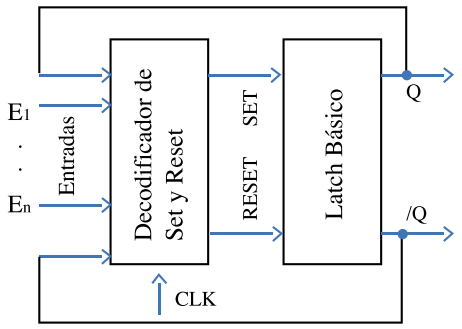


Figura 1.26. Diagrama de bloques de un *latch* asincrónico

1.7.1 Tipos de *latch* asincrónicos

Existe una gran variedad de *latch* asincrónicos. En realidad se podría decir que infinitos, ya que como se verá más adelante puede crearse uno personalizado, es decir, según las necesidades de los usuarios. Sin embargo, existen los siguientes *latchs* asincrónicos que son tipicos.

1. *Flip-flop* tipo D
2. *Flip-flop* tipo T
3. *Flip-flop* tipo J K
4. *Flip-flop* tipo S R

Todo *latch* asincrónico trabaja en sincronía con la señal de reloj; por lo tanto, es necesario conocer algunas características del reloj.

1.8 El reloj o CLK

Un reloj es una señal de voltaje formada por niveles altos y bajos alternados. Como se indica en la figura 1.27, tiene una frecuencia en hertzios (Hz) y un periodo en segundos.

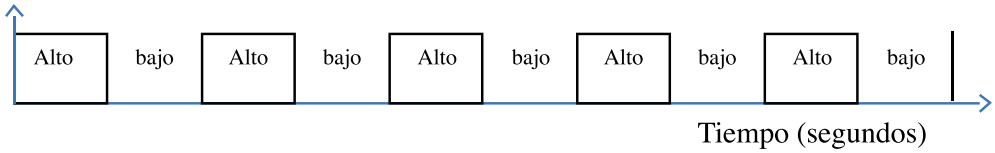


Figura 1.27. El reloj

1.8.1 Tipos de reloj

Los dispositivos que tienen una señal de reloj, se activan o son sensibles a uno de los niveles o a uno de los flancos del reloj, por ejemplo, pueden ser sensibles al nivel alto o al nivel bajo, o al flanco de subida o al flanco de bajada. De esta manera los relojes pueden ser:

1. Verdadero con nivel alto
2. Verdadero con nivel bajo
3. Verdadero con el flanco de subida
4. Verdadero con el flanco de bajada

1.8.2 Reloj verdadero con nivel alto

Los *flip-flops* activados por nivel o *latchs* asincrónicos trabajan con relojes sensibles a uno de los niveles de la señal de reloj, es decir, su reloj puede ser verdadero con el nivel alto o con el nivel bajo. En el caso de un reloj verdadero con el nivel alto, la acción de almacenar un bit de información se realiza cuando la señal de reloj está en nivel alto.

La figura 1.28 muestra un reloj con el nivel alto resaltado, a los relojes verdaderos con nivel alto se suele simbolizar como: CLK.H, H es de *high* o alto. La figura 1.29 muestra un dispositivo con la simbología para el reloj verdadero con nivel alto.

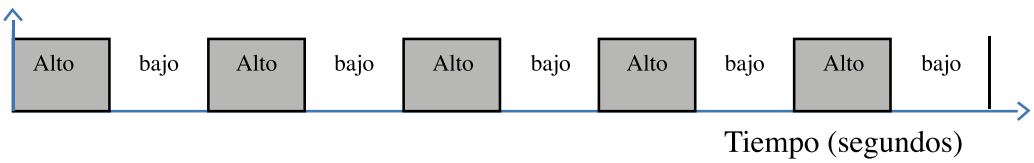


Figura 1.28 El reloj verdadero con nivel alto

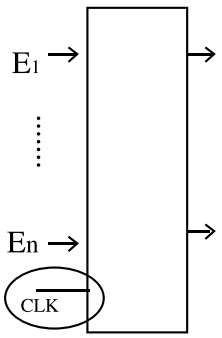


Figura 1.29. Símbolo del reloj verdadero con nivel alto

Nótese en la figura 1.29 que el símbolo para un reloj verdadero con nivel alto es una línea simple conectada al bloque y simbolizada como CLK. En general, si

un dispositivo digital trabaja con un reloj verdadero con nivel alto, la operación que tiene que ejecutar el dispositivo, se realiza cuando el reloj está en nivel alto; así por ejemplo, si se trata de un contador, la acción de contar, el contador la ejecutará cuando el reloj esté en nivel alto.

1.8.3 Reloj verdadero con nivel bajo

En los *latch* asincrónicos que trabajan con relojes verdaderos con nivel BAJO, la acción de actualizar y almacenar un bit de información se realiza cuando la señal de reloj está en el nivel bajo. La figura 1.30 muestra un reloj con el nivel bajo resaltado. Se suele simbolizar como CLK.L, la L es de *low* o bajo.

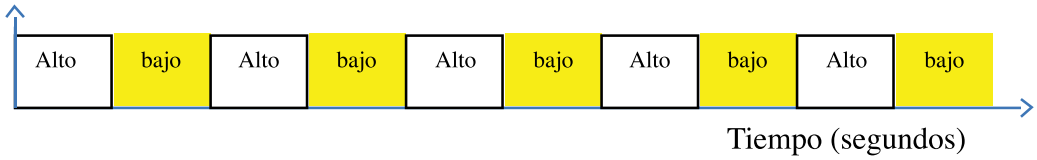


Figura 1.30 El reloj verdadero con nivel bajo

Por ejemplo, si se trata de un contador, la acción de contar, el contador la ejecutará cuando el reloj este en nivel bajo. La figura 1.31 muestra un dispositivo que tiene la simbología del reloj verdadero con nivel bajo. Nótese que el símbolo es una línea recta que termina en un cero, como haciendo referencia a un cero lógico y se conecta al dispositivo.

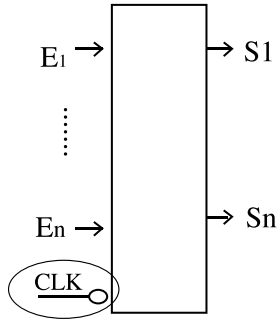


Figura 1.31. Simbología del reloj verdadero con nivel bajo

1.8.4 Reloj verdadero con el flanco de subida

En los *latch* asincrónicos que trabajan con relojes verdaderos con el flanco de subida, la acción de actualizar y almacenar un bit de información se realiza cuando la señal del reloj está haciendo su transición de nivel bajo a nivel alto.

La transición de nivel bajo a nivel alto se muestra en la figura 1.32. Se suele simbolizar como CLK.î, es decir CLK punto y una flecha hacia arriba.

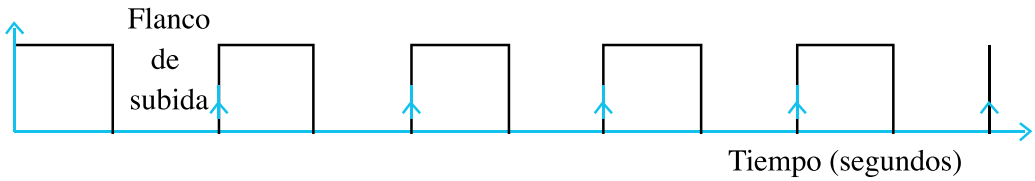


Figura 1.32 El reloj verdadero con el flanco de subida

Por ejemplo, si se trata de un contador, la acción de contar, el contador la ejecutará cuando el reloj esté haciendo su transición desde el nivel bajo al nivel alto. La simbología para un reloj verdadero con el flanco de subida se muestra en la figura 1.33, allí se puede ver que el reloj está representado por una recta que termina en una flecha que está dentro del dispositivo.

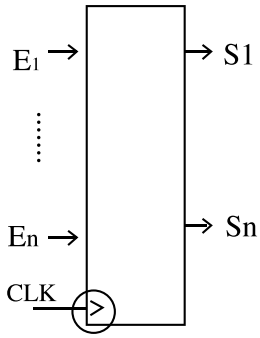


Figura 1.33. Símbolo del reloj verdadero con el flanco de subida

1.8.5 Reloj verdadero con el flanco de bajada

En los *flip-flops* que trabajan con relojes verdaderos con el flanco de bajada, la acción de almacenar un bit de información se realiza cuando la señal de reloj

está haciendo su transición de nivel alto a nivel bajo, esto se indica en la figura 1.34. Se suele simbolizar como CLK.!, es decir clk punto flecha hacia abajo.

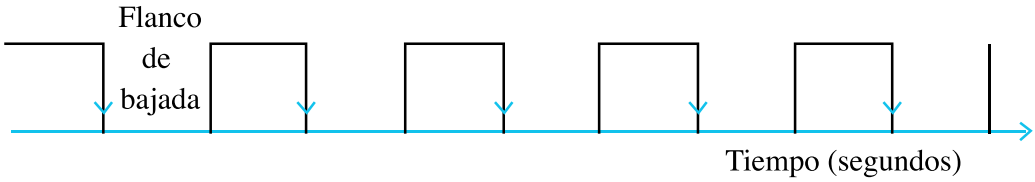


Figura 1.34 Reloj verdadero con el flanco de bajada

La simbología para un reloj verdadero con el flanco de bajada se muestra en la figura 1.35. Allí se puede ver que el reloj está representado por una recta que termina en un círculo seguido de una flecha que se encuentra dentro del dispositivo. Por ejemplo, si se trata de un contador, la acción de contar, el contador la ejecutará cuando el reloj este haciendo su transición desde el nivel alto al nivel bajo.

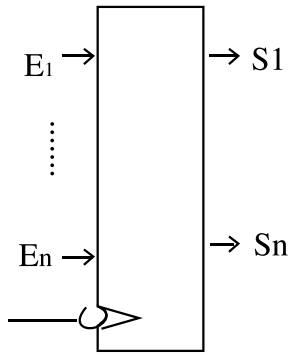


Figura 1.35. Símbolo del reloj verdadero con el flanco de bajada

1.9 Diseño de latchs asincrónicos

Como se explicó un *latch* asincrónico es sensible al nivel del reloj y se diseña sobre la base de un *latch* básico. Para diseñar un *latch* asincrónico se requiere la siguiente información:

1. Las especificaciones de cómo debe funcionar el *latch* asincrónico.

2. El tipo de *latch* básico (NAND o NOR) que va a utilizar.

Las especificaciones de cómo debe funcionar el *latch* asincrónico se pueden realizar mediante una descripción oral o escrita o mediante una tabla de verdad.

El *latch* básico que se va a utilizar como base para el diseño del *latch* asincrónico hay que especificar, porque un tipo trabaja con lógica positiva (NOR) y el otro, con lógica negativa (NAND).

El diseño del decodificador de *SET* y *RESET* se realiza utilizando las técnicas de diseño de circuitos combinacionales, es decir, mediante una tabla de verdad. Es importante notar que este decodificador es un circuito combinacional.

1.9.1 El *latch* asincrónico tipo D

El diagrama de bloque de un *latch* asincrónico tipo D muestra la figura 1.36. Su función es tomar el valor que tiene en la entrada D y transmitir a la salida Q ese valor, en otras palabras, la salida Q es igual a la entrada D cuando el reloj es verdadero. La descripción del funcionamiento de este *latch* asincrónico se resume en la tabla 1.7.

A esta tabla se le llama la tabla característica. Como se puede ver no es una tabla común ya que intervienen el estado presente y el estado siguiente de Q.

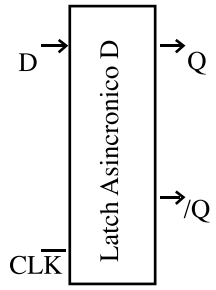


Figura 1.36. *Latch* asincrónico tipo D.

Como se puede concluir de la tabla 1.7, cuando el reloj está en su condición de falso, la salida Q no cambia, ya que el estado presente, Qn, es igual al estado siguiente Qn+1. Esto está en concordancia con la función que tiene el reloj.

CLK	D	Qn		Qn+1
F	0	0		0
F	0	1		1
F	1	0		0
F	1	1		1
V	0	0		0
V	0	1		0
V	1	0		1
V	1	1		1

Tabla 1.7. Tabla característica del *latch* tipo D

Cuando el reloj es verdadero el valor que se encuentra en la entrada D pasa a la salida Q, permitiendo que el estado siguiente de Q, Qn+1, sea el valor de D. El funcionamiento de este dispositivo es parecido al de un interruptor, que en un extremo tiene la entrada D y en el otro la salida Q. El CLK controla el interruptor. Si es verdadero el interruptor se cierra y la salida Q se conecta físicamente a la entrada D; si el reloj es falso, el interruptor está abierto y no hay conexión física entre Q y D; por lo tanto, la salida Q mantiene el valor, no cambia. La figura 1.37 a) y b) muestra la analogía del *latch* asincrónico D con un interruptor

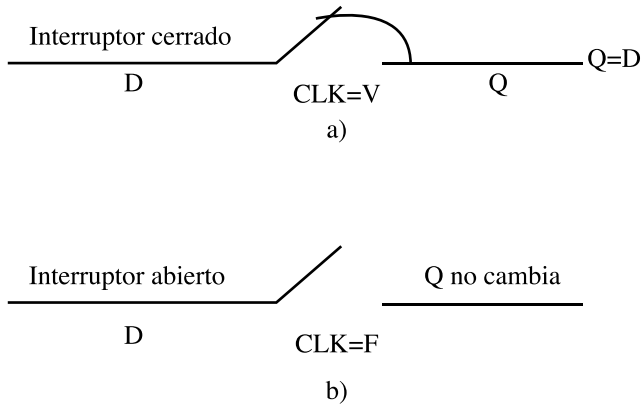


Figura 1. 37. *Latch* asincrónico D como un interruptor

Se debe observar que mientras el reloj es falso no hay cambio de estado para Q, por lo que Qn = Qn+1, cuando el reloj es verdadero el *latch* asincrónico D realiza la acción para la cual fue diseñado y Qn+1=D.

Un *latch* asincrónico tipo D que sea sensible al nivel alto del reloj se puede construir con un multiplexor de 2 a 1. Este multiplexor, funcionando como un *latch* asincrónico tipo D, se muestra en la figura 1.38. En la una entrada del multiplexor está la entrada D (línea 1), en la otra (línea 0) la salida Q y en la línea de selección del multiplexor está conectado el reloj. Cuando el multiplexor selecciona la línea 0, el reloj está en cero, la salida Q es igual a Q mismo, porque está realimentada. Cuando el reloj alcanza el nivel, la línea de selección del multiplexor selecciona la línea uno, es decir el valor de la entrada D.

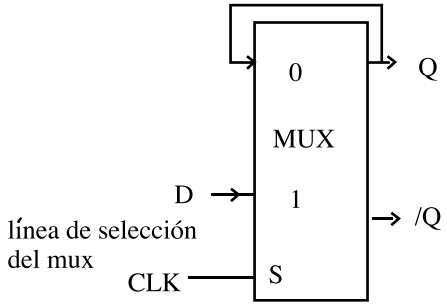


Figura 1.38. Un multiplexor funcionando como un *latch* asincrónico

Un *latch* asincrónico tipo D, sensible al nivel bajo del reloj construido con un multiplexor de 2 a 1, se muestra en la figura 1.39.

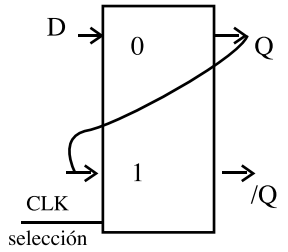


Figura 1.39. Un multiplexor funcionando como un *latch* asincrónico

La salida Q va conectada a la entrada identifica como 1 y la entrada D está conectada a la entrada cero. Cuando el reloj (línea de selección) es igual a cero, Q es igual a D; y cuando el reloj es igual a uno, Q es igual a Q mismo.

Ejemplo 1.6

Diseñe un *latch* asincrónico tipo D. El dispositivo debe tener un *latch* básico que trabaje con lógica positiva como elemento de memoria.

El *latch* básico con compuertas NOR trabaja con lógica positiva y es el que debe utilizarse. El diagrama de bloques general del *latch* asincrónico D y el diagrama de bloques general de un *latch* asincrónico que muestra el *latch* básico se grafican en la figura 1.40.

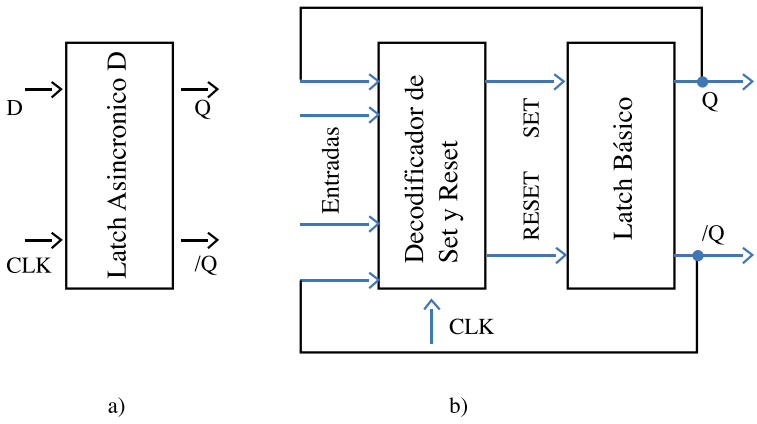


Figura 1.40. a) *Latch* asincrónico D, b) *Latch* asincrónico conteniendo un *latch* básico.

El procedimiento de diseño se explica a continuación:

1. Se comparan los diagramas de bloques.
2. Se dibuja, como una caja negra, el diagrama de bloques del latch asincrónico que se quiere diseñar.
3. Se conectan, mediante una interfaz, las entradas de la caja negra del paso anterior a las entradas *SET* y *RESET* del *latch* básico.
4. Se diseña el decodificador de *SET* y *RESET*.
5. Se implementa.

Paso 1. Se comparan los diagramas de bloques. La figura 1.40 muestra los dos diagramas de bloques.

Al comparar las figura 1.40 a) y b), la diferencia es evidente y está en las entradas. El gráfico a) tiene una sola entrada externa denominada D aparte del reloj y el bloque b) tiene varias entradas externas aparte del reloj.

Paso 2. Se dibuja el diagrama de bloques del *latch* asincrónico como una caja negra. En su interior debe estar el *latch* básico, puesto que todo *latch* asincrónico tiene en su interior un *latch* básico como muestra la figura 1.41 b.

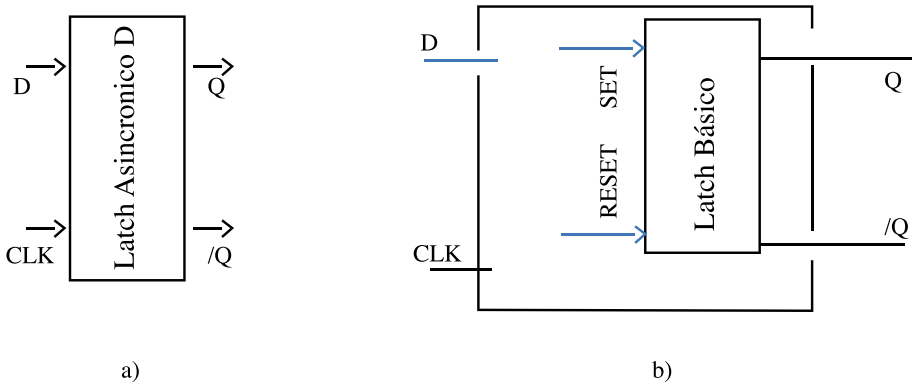


Figura 1.41. Bloques como cajas negras.

Los gráficos a) y b) de la figura 1.41 vistos como una caja negra son idénticos.

Paso 3. Hay que conectar la entra D y el CLK a las dos únicas entradas disponibles, es decir a *SET* y *RESET*. Como no puede ser una conexión directa entra líneas es necesario buscar algún mecanismo para realizar estas conexiones, una forma de hacerlo es mediante una interfaz, un bloque intermedio, que una las entradas D y CLK con set y reset.

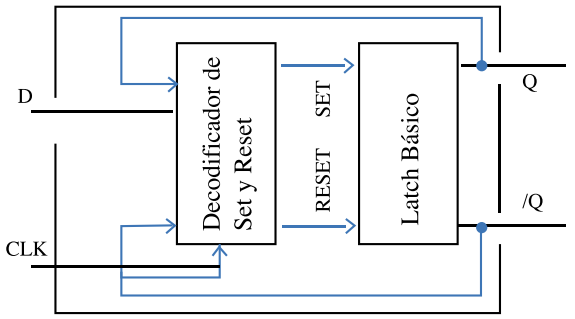


Figura 1.42. Conexión de D y CLK a *SET* y *RESET* del *latch* básico.

Esta conexión se muestra en la figura 1.42, se han conectado a esta interfaz también las salidas Q y /Q porque esta interfaz necesita saber cuál es el estado en el que están las salidas Q y /Q, ya que según la tabla de funcionamiento del *latch* básico, el valor al que irá la salida Q depende del valor de las entrada *SET*, *RESET*, Q y /Q.

Paso 4. En la figura 1.42, el único bloque desconocido es la interfaz. El *latch* básico se sabe ya cómo está construido, por lo que no hay que diseñar este bloque. Como no hay caminos de retroalimentación de las salidas del bloque a las entradas del mismo se concluye que este bloque es un circuito combinacional, por lo tanto, se utiliza la tabla 1.8 para diseñar las entradas que van al lado izquierdo y las salidas al lado derecho, no hay que olvidar que esta es una tabla característica; por lo tanto, contiene el estado presente Qn y el estado siguiente Qn+1, las salidas son *SET* y *RESET*.

CLK	D	Qn	Qn+1	SET	RESET
F	0	0	0	0	ϕ
F	0	1	1	ϕ	0
F	1	0	0	0	ϕ
F	1	1	1	ϕ	0
V	0	0	0	0	ϕ
V	0	1	0	0	1
V	1	0	1	1	0
V	1	1	1	ϕ	0

Tabla 1.8. Tabla para diseñar el decodificador *SET* - *RESET*.

De la tabla 1.8 se crean los mapas de Karnaught para simplicar las funciones. La tabla 1.9 muestra el mapa para simplificar la señal *SET*. *SET*= CLK.D.

Q \ CLK D	00	01	11	10
0	0	0	1	0
1	ϕ	ϕ	ϕ	0

Tabla 1.9. Mapa para la simplicación del *SET*.

La tabla 1.10 muestra el mapa para simplificar la señal *RESET*.

Q \ CLK D	00	01	11	10
0	0	0	0	ϕ
1	ϕ	ϕ	0	1

Tabla 1.10. Mapa para la simplicación del *RESET*.

$$RESET = CLK./D \; . \qquad (2)$$

Paso 5. El último paso es la implementación de las ecuaciones booleanas encontradas para *SET* y *RESET*. La figura 1.43 muestra la implementación.

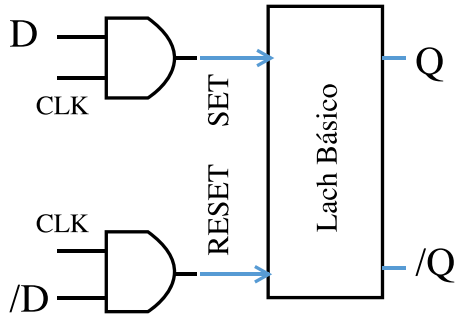


Figura 1.43. Implementación del *latch* asincrónico tipo D

La tabla de verdad queda como se indica en la tabla 1.7. Se observa que es la misma tabla del *latch* tipo D, añadida las señales de salidas *SET* y *RESET*.

1.9.2 El *lacht* asincrónico tipo T

El diagrama de bloque de un *latch* asincrónico tipo T muestra la figura 1.44 a). Este dispositivo tiene una entrada denominada T aparte del reloj y dos salidas Q y /Q. Su funcionamiento es como indica la tabla característica 1.11. Las columnas de esta tabla describen su funcionamiento y son: CLK, T, Qn y Qn+1, las columnas *SET* y *RESET* no deben ser tomadas en cuenta, son más bien útiles para el ejemplo 1.7.

Como se puede concluir del análisis de las cuatro columnas (CLK, T, Qn, Qn+1) de la tabla 1.11, este dispositivo mantiene su estado. La salida Q no cambia mientras la entrada T sea igual a cero, cuando esta entrada es igual a uno, el estado siguiente, Qn+1, es el opuesto al estado presente Qn.

Como en todo *latch* asincrónico, el reloj es la señal que controla el instante en que el dispositivo actualiza su salida Q. Si el reloj es falso, la salida Q no cambia independientemente del valor que tenga la entrada T; si el reloj es verdadero el dispositivo trabaja como indican las cuatro columnas de la tabla 1.11.

Ejemplo 1.7

Diseñe un *latch* asincrónico tipo T. Suponga que va a trabajar con lógica positiva el *latch* básico. Encuentre las ecuaciones booleanas, pero no implemente.

Paso 1. Se comparan los diagramas de bloques. La figura 1.44 muestra los dos diagramas de bloques. Al comparar las dos figura a) y b), la diferencia es evidente y está en las entradas. El gráfico a) tiene una sola entrada externa denominada T aparte del reloj; y el bloques b) tiene varias entradas externas aparte del reloj.

Paso 2. Se dibuja el diagrama de bloques del *latch* asincrónico como una caja negra, que, en su interior, debe tener el *latch* básico, puesto que todo *latch* asincrónico tiene en su interior un *latch* básico (ver la figura 1.45 b).

Los gráficos a) y b) de la figura 1.45 vistos como una caja negra son idénticos.

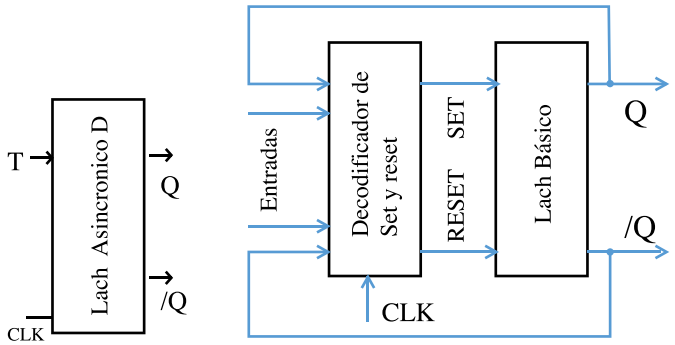


Figura 1.44. a) *Latch* asincrónico general T b) *Latch* asincrónico con un *latch* básico.

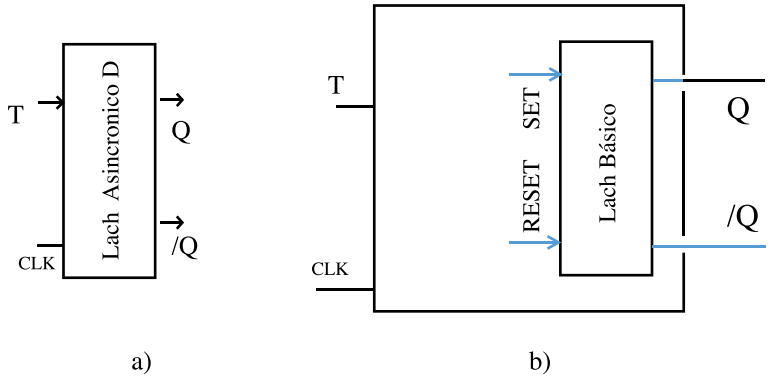


Figura 1.45. Bloques como cajas negras

Paso 3. Hay que conectar la entra T y CLK a las dos únicas entradas disponibles, es decir a *SET* y *RESET*. Lógicamente como no puede ser una conexión directa entra señales, es necesario buscar algún mecanismo para realizar estas conexiones. Se utiliza una interfaz que conecte las entradas T y CLK con *SET* y *RESET*.

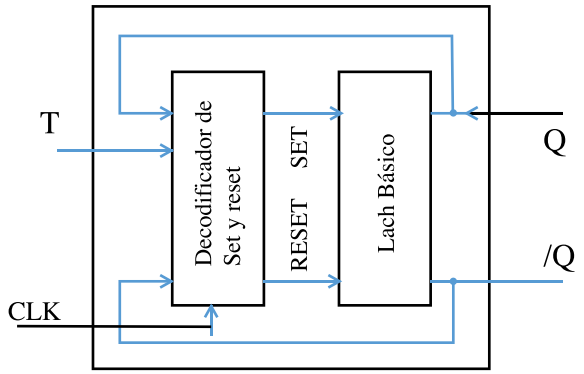


Figura 1.46. Conexión de T y CLK a *SET* y *RESET* del *latch* básico

Esta conexión se muestra en la figura 1.46. Se han conectado a esta interfaz también las salidas Q y /Q porque esta interfaz necesita saber cuál es el estado en el que están las salidas Q y /Q, ya que, según la tabla de funcionamiento del *latch* básico, la salida Q depende del valor de las entrada *SET*, *RESET*, Q y /Q.

Paso 4. En la figura 1.46, el único bloque desconocido es la interfaz. El *latch* básico se sabe ya cómo está construido, por lo que no hay que diseñar este bloque, como no hay caminos de retroalimentación de las salidas del bloque a las en-

tradas del mismo se concluye que este bloque es un circuito combinacional. Por lo tanto, se utiliza una tabla de verdad para diseñar, como se aprecia en la tabla 1.11, donde las entradas van del lado izquierdo y las salidas, del lado derecho. No hay que olvidar que esta no es una tabla común sino una tabla característica. Por lo tanto, contiene el estado presente Qn y el estado siguiente Qn+1, las salidas son *SET* y *RESET*.

De la tabla 1.11, se crean los mapas de Karnaught para simplicar las funciones. La tabla 1.12 muestra el mapa para simplificar la señal SET.

CLK	T	Qn	Qn+1	*SET*	*RESET*
F	0	0	0	0	ϕ
F	0	1	1	ϕ	0
F	1	0	0	0	ϕ
F	1	1	1	ϕ	0
V	0	0	0	0	ϕ
V	0	1	1	ϕ	0
V	1	0	1	1	0
V	1	1	0	0	1

Tabla 1.11. Tabla para diseñar el decodificador *SET* - *RESET*.

Q \ CLK T	00	01	11	10
0	0	0	1	0
1	ϕ	ϕ	0	ϕ

Tabla 1.12. Mapa para la simplicación del *SET*

SET= CLK.T/Q. (3)

La tabla 1.13 muestra el mapa para simplificar la señal *RESET*.

Q \ CLK T	00	01	11	10
0	ϕ	ϕ	0	ϕ
1	0	0	1	0

Tabla 1.13. Mapa para la simplicación del *RESET*

RESET=CLK.T.Q. (4)

Paso 5. El último paso es la implementación de las ecuaciones booleanas encontradas para *SET* y *RESET*, pero en este ejemplo no se pide implementar.

1.9.3 El *latch* asincrónico tipo JK

El diagrama de bloque de un *latch* asincrónico tipo JK se muestra en la figura 1.47 a). Este dispositivo tiene dos entradas denominadas J y K aparte del reloj y dos salidas Q y /Q, su funcionamiento es como indica la tabla característica 1.14. Las columnas de esta tabla que describen su funcionamiento son: CLK, J, K, Qn y Qn+1, las columnas *SET* y *RESET* no deben ser tomadas en cuenta en este momento, son más bien útiles para el ejemplo 1.8.

Como se puede concluir del análisis de las cuatro columnas (J, K, Qn, Qn+1) de la tabla 1.14, este dispositivo mantiene su estado, la salida Q no cambia, siempre que JK=00 y el reloj verdadero; cuando JK=10 y el reloj verdadero la salida Q es igual a uno; cuando JK=01 y el reloj verdadero la salida Q es igual a cero; y cuando JK=11, la salida Q cambia de estado, es decir Qn+1=/Qn,

La tabla 1.14 no incluye el reloj solo para evitar que esta tabla se vuelva muy extensa, ya que, si se considera el reloj, esta tabla tendría 16 combinaciones posibles y, por ende, 16 filas.

Como en todo *latch* asincrónico, el reloj es la señal que controla el instante en que el dispositivo actualiza su salida Q. Si el reloj es falso, la salida Q no cambia, independientemente del valor que tengan las entradas JK; si el reloj es verdadero, el dispositivo trabaja como indican las cuatro columnas de la tabla 1.14.

Ejemplo 1.8

Diseñe un *latch* asincrónico tipo JK. Base su diseño en un *latch* básico con compuertas NOR. No implemente.

Paso 1. Se comparan los diagramas de bloques. La figura 1.52 muestra los dos diagramas de bloques.

Al comparar las dos figuras a) y b), la diferencia es evidente y está en las entradas. El gráfico a) tiene dos entradas externas denominadas J y K aparte del reloj y el bloques b) tiene varias entradas externas aparte del reloj.

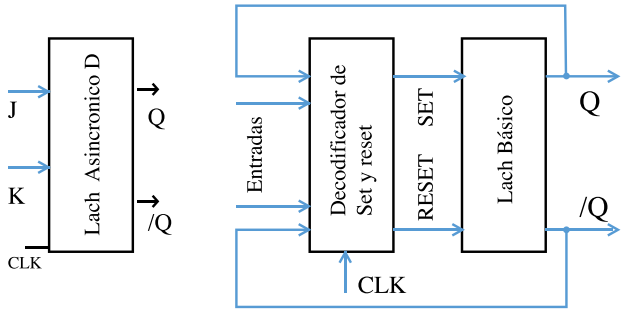


Figura 1.47. a) *Latch* asincrónico JK. b) *Latch* asincrónico mostrando al *latch* básico.

Paso 2. Se dibuja el diagrama de bloques del *latch* asincrónico como una caja negra, que en su interior debe estar el *latch* básico, puesto que todo *latch* asincrónico tiene en su interior un *latch* básico. Ver la figura 1.48 b). Los gráficos a) y b) de la figura 1.47 vistos como una caja negra son idénticos.

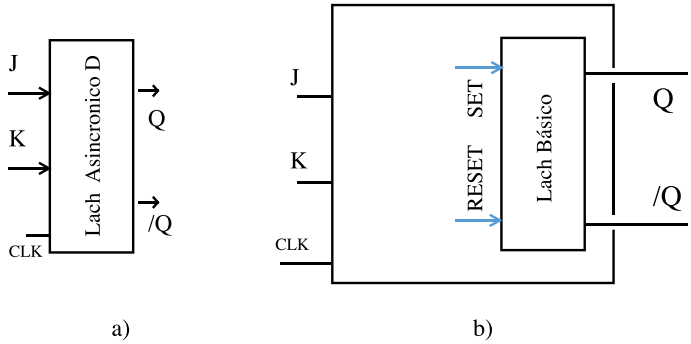


Figura 1.48. Bloques como cajas negras

Paso 3. Hay que conectar la entradas J, K y el CLK a las dos únicas entradas disponibles, es decir a *SET* y *RESET*. Lógicamente como no puede ser una conexión directa entra líneas, es necesario buscar algún mecanismo para realizar estas conexiones. Una forma de hacerlas es mediante una interfaz, un bloque intermedio, que una las entradas J, K y CLK con *SET* y *RESET*.

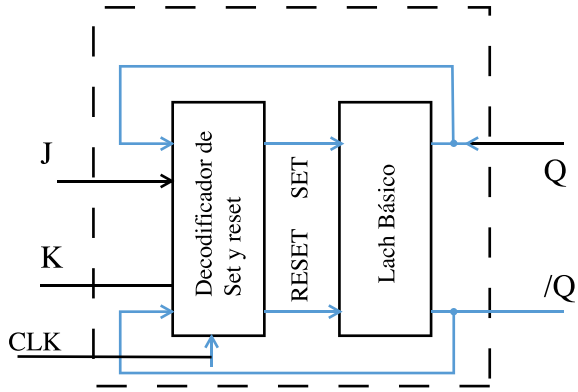


Figura 1.49. Conexión de JK y CLK a *SET* y *RESET* del *latch* básico

Esta conexión se muestra en la figura 1.49. Se han conectado a esta interfaz también las salidas Q y /Q porque esta interfaz necesita saber cuál es el estado en el que están las salidas Q y /Q, ya que según la tabla de funcionamiento del *latch* básico, la salida Q depende del valor de las entrada *SET*, *RESET*, Q y /Q.

Paso 4. En la figura 1.49, el único bloque desconocido es la interfaz, el *latch* básico se sabe ya cómo está construido, por lo que no hay que diseñar este bloque, como no hay caminos de retroalimentación de las salidas del bloque a las entradas del mismo se concluye que este bloque es un circuito combinacional, por lo tanto, se utiliza la tabla de verdad 1.14 para diseñar.

J	K	Qn	Qn+1	*SET*	*RESET*
0	0	0	0	0	ϕ
0	0	1	1	ϕ	0
0	1	0	0	0	ϕ
0	1	1	0	0	1
1	0	0	1	1	0
1	0	1	1	ϕ	0
1	1	0	1	1	0
1	1	1	0	0	1

Esta tabla trabaja como tal si y solo si el CLK es verdadero. Si CLK es falso, Qn=Qn+1

Tabla 1.14. Tabla para diseñar el decodificador *SET - RESET*

En la tabla 1.14, las entradas van del lado izquierdo. Con la finalidad de que esta tabla no sea muy grande el reloj se incluye al final. Las salidas van del lado derecho de la tabla, esta tabla contiene el estado presente Qn y el estado siguiente Qn+1; las salidas son *SET* y *RESET*. De la tabla 1.14, se crean los mapas de Karnaught. La tabla 1.15 muestra el mapa para simplificar la señal *SET*.

Q \ JK	00	01	11	10
0	0	0	1	1
1	ϕ	0	0	ϕ

Tabla 1.15. Mapa para la simplicación del *SET*. *SET*= J.K./Qn.CLK

,La tabla 1.16 muestra el mapa para simplificar la señal *RESET*.

Q \ JK	00	01	11	10
0	ϕ	ϕ	0	0
1	0	1	1	0

Tabla 1.16. Mapa para la simplicación del *RESET*.

RESET= CLK./D. (5)

Paso 5. El último paso es la implementación de las ecuaciones booleanas encontradas para *SET* y *RESET*.

Ejemplo 1.9

Diseñe un *latch* asincronico denominado SN, que trabaja según como se indica en la tabla 1.17, base su diseño en un *latch* básico con compuertas NOR.

S	N	Qn	Qn+1
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	0

Tabla 1.17. Funcionamiento del *latch* asincrónico SN

Paso 1. Se comparan los diagramas de bloques. La figura 1.50 muestra los dos diagramas de bloques.

Al comparar las dos figuras a) y b), la diferencia es evidente y está en las entradas, el gráfico a) tiene dos entradas externa denominada S y N aparte del reloj y el bloques b) tiene varias entradas externas aparte del reloj.

Paso 2. Se dibuja el diagrama de bloques del *latch* asincrónico como una caja negra, que en su interior contiene un *latch* básico como muestra la figura 1.51 b.

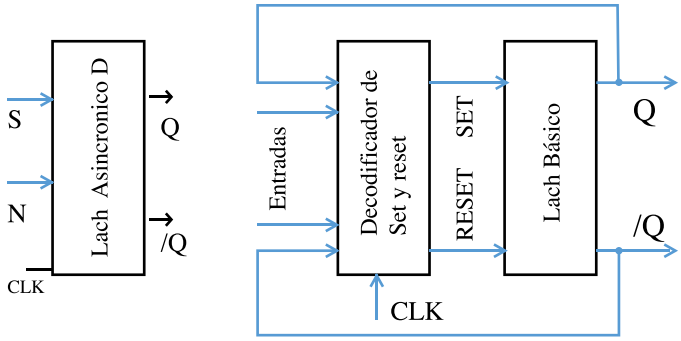


Figura 1.50. a) *Latch* asincrónico SN. b) *Latch* asincrónico con un *latch* básico

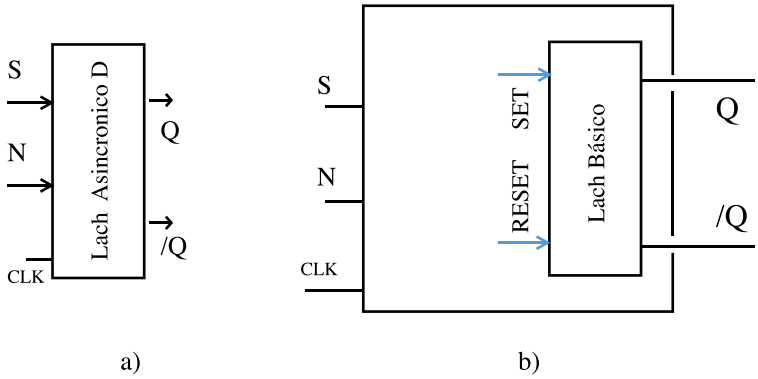


Figura 1.51. Bloques como cajas negras

Los gráficos a) y b) de la figura 1.51 vistos como cajas negras son idénticos.

Paso 3. Hay que conectar las entradas S, N y CLK a *SET* y *RESET* mediante una interfaz que una las entradas S, N y CLK con *SET* y *RESET*.

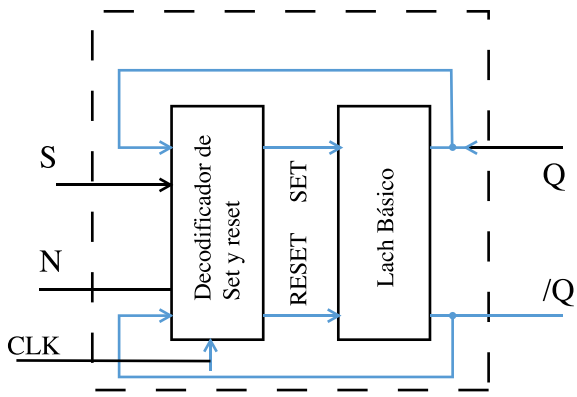


Figura 1.52. Conexión de SN y CLK a *SET* y *RESET* del *latch* básico

Esta conexión se muestra en la figura 1.52, se han conectado a esta interfaz también las salidas Q y /Q porque esta interfaz necesita saber cuál es el estado en el que están las salidas Q y /Q, ya que según la tabla de funcionamiento del *latch* básico, la salida Q depende del valor de las entrada *SET*, *RESET*, Q y /Q.

Paso 4. En la figura 1.52, el único bloque desconocido es la interfaz, el *latch* básico se sabe ya cómo está construido, por lo que no hay que diseñar este bloque. Como no hay caminos de retroalimentación de las salidas del bloque a las entradas del mismo se concluye que este bloque es un circuito combinacional, por lo tanto, se utiliza la tabla 1.16 para diseñar.

En la tabla de verdad 1.16, las entradas van del lado izquierdo. Con la finalidad de que esta tabla no sea muy grande, se deja al reloj para incluirlo al final. Las salidas van del lado derecho de la tabla. No hay que olvidar que esta no es una tabla común sino una tabla característica por lo tanto contiene el estado presente Qn y el estado siguiente Qn+1, las salidas son *SET* y *RESET*.

De la tabla 1.16 se crean los mapas de Karnaught para simplicar las funciones. La tabla 1.17 muestra el mapa para simplificar la señal *SET*.

S	N	Qn	Qn+1	*SET*	*RESET*
0	0	0	0	0	ϕ
0	0	1	0	0	1
0	1	0	1	1	0
0	1	1	1	ϕ	0
1	0	0	0	1	0
1	0	1	0	0	1
1	1	0	1	1	0
1	1	1	0	0	1

Esta tabla trabaja como tal si y solo si el CLK es verdadero. Si CLK es falso, Qn=Qn+1

Tabla 1.17. Mapa para la simplicación del *SET*

Q \ SN	00	01	11	10
0	0	1	1	1
1	0	ϕ	0	0

SET= ((N./Qn)+ (S./Qn)) Se incluye el reloj.

SET= ((N./Qn)+ (S./Qn)).CLK. (6)

La tabla 1.18 muestra el mapa para simplificar la señal *RESET*

Qn \ SN	00	01	11	10
0	ϕ	ϕ	0	0
1	0	1	1	0

Tabla 1.18. Mapa para la simplicación del *RESET*

RESET= (N./ Qn). (7)

Se incluye el reloj.

RESET= (N./Qn).CLK . (8)

Paso 5. El último paso es la implementación de las ecuaciones booleanas encontradas para *SET* y *RESET*, no se implementa.

1.9.4 El *latch* asincrónico tipo SR

El diagrama de bloque de un *latch* asincrónico tipo SR se muestra en la figura 1.53. Este dispositivo tiene dos entradas denominadas S y R aparte del reloj y dos salidas Q y /Q, su funcionamiento se indica en la tabla característica 1.19.

Como se puede concluir del análisis de las columnas de la tabla 1.19, este dispositivo mantiene su estado. La salida Q no cambia, siempre que SR=00 y el reloj verdadero, cuando SR=10 y el reloj verdadero la salida Q es igual a uno, cuando SR=01 y el reloj verdadero la salida Q es igual a cero, y cuando SR=11 la salida Q es indeterminada.

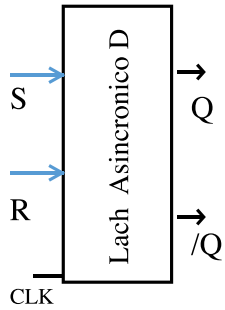


Figura 1.53. *Latch* asincrónico SR

La tabla 1.19 no incluye el reloj, solo para evitar que esta tabla no sea muy extensa, ya que si se considera el reloj, tendría 16 combinaciones posibles y por ende 16 filas.

S	R	Qn	Qn+1
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	Ø
1	1	1	Ø

Esta tabla trabaja como tal si y solo si el CLK es verdadero. Si CLK es falso, Qn=Qn+1

Tabla 1.19. Tabla característica del *latch* asincrónico tipo SR.

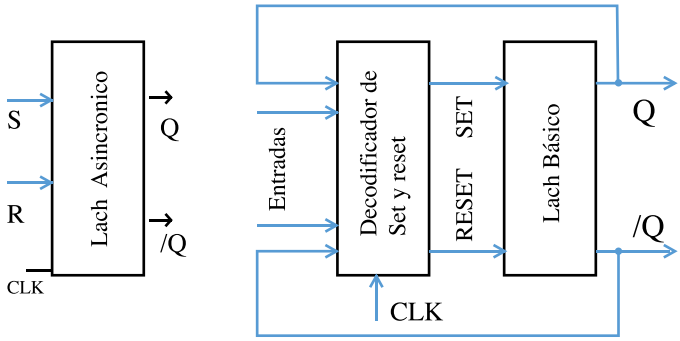
Como en todo latch asincrónico el reloj es la señal que controla el instante en que el dispositivo actualiza su salida Q. Si el reloj es falso la salida Q no cambia independientemente del valor que tengan las entradas SR. Si el reloj es verdadero el dispositivo trabaja como la tabla 1.19.

Este dispositivo trabaja en forma muy parecida al *latch* asincrónico JK. La diferencia está en el valor que toma la salida Q cuando las dos entradas son verdaderas simultáneamente.

Ejemplo 1.10

Diseñe un *latch asincrónico* tipo SR, base su diseño en un *latch* básico con compuertas NOR. No implemente.

Paso 1. Se comparan los diagramas de bloques. La figura 1.54 muestra los dos diagramas de bloques.

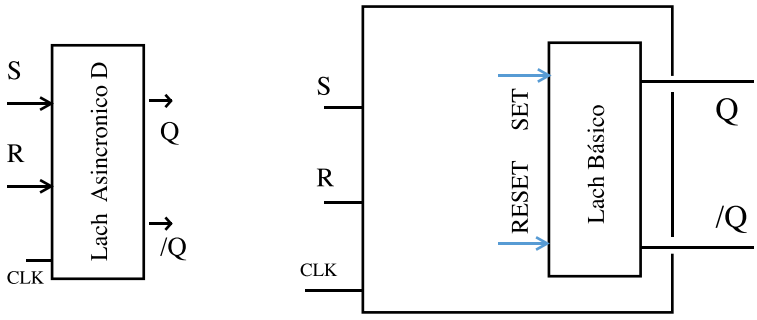


a) b)

Figura 1.54. a) *Latch* asincrónico SR. b) *Latch* asincrónico que contiene un *latch* básico

Al comparar las dos figuras a) y b), la diferencia es evidente y está en las entradas. El gráfico a) tiene dos entradas externas denominada S y R aparte del reloj; y el bloque b) tiene varias entradas externas aparte del reloj.

Paso 2. Se dibuja el diagrama de bloques del *latch* asincrónico como una caja negra, que en su interior debe contener un *latch* básico como muestra la figura 1.55 b). Los gráficos a) y b) de la figura 1.55 vistos como una caja negra son idénticos.



a) b)

Figura 1.55. Bloques como cajas negras

Paso 3. Hay que conectar la entras S, R y el CLK a las dos únicas entradas disponibles, es decir a *SET* y *RESET*. Lógicamente como no puede ser una conexión directa entra líneas, es necesario buscar algún mecanismo para realizar estas conexiones. Una forma de hacer es mediante una interfaz, un bloque intermedio, que una las entradas S, R y CLK con *SET* y *RESET*.

Esta conexión se muestra en la figura 1.56. Se han conectado a esta interfaz también las salidas Q y /Q porque esta interfaz necesita saber cuál es el estado en el que están las salidas Q y /Q, ya que según la tabla de funcionamiento del *latch* básico, la salida Q depende del valor de las entrada *SET*, *RESET*, Q y /Q

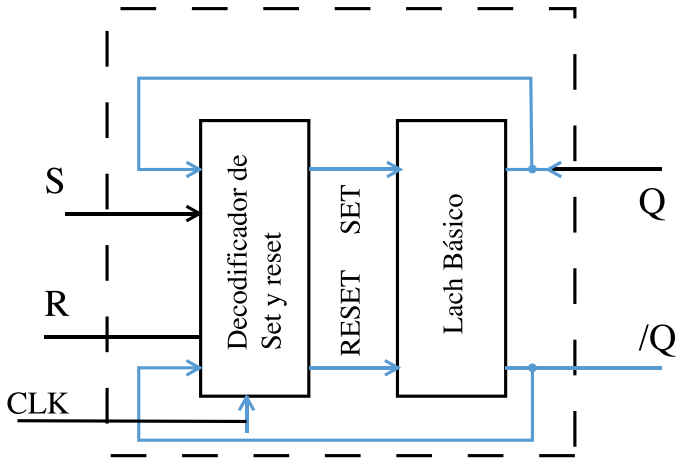


Figura 1.56. Conexión de JK y CLK a *SET* y *RESET* del *latch* básico

Paso 4. En la figura 1.56, el único bloque desconocido es la interfaz. El *latch* básico se sabe ya cómo está construido, por lo que no hay que diseñar este bloque. Como no hay caminos de retroalimentación de las salidas del bloque a las entradas del mismo se concluye que este bloque es un circuito combinacional, por lo tanto, se utiliza la tabla1.20 para diseñar.

S	R	Qn	Qn+1	*SET*	*RESET*
0	0	0	0	0	ϕ
0	0	1	1	ϕ	0
0	1	0	0	0	ϕ
0	1	1	0	0	1
1	0	0	1	1	0
1	0	1	1	ϕ	0
1	1	0	Ø	Ø	Ø
1	1	1	Ø	Ø	Ø

Esta tabla trabaja como tal si y solo si el CLK es verdadero. Si CLK es falso, Qn=Qn+1

Tabla 1.20. Tabla para diseñar el decodificador *SET - RESET*

En la tabla de verdad 1.20, las entradas van del lado izquierdo. Con la finalidad de que esta tabla no sea muy grande se deja al reloj para incluirlo al final.

Las salidas van del lado derecho de la tabla. Esta tabla característica contiene el estado presente Qn y el estado siguiente Qn+1; las salidas son *SET* y *RESET*.

De la tabla 1.20, se crean los mapas de Karnaught para simplicar las funciones.

La tabla 1.21 muestra el mapa para simplificar la señal *SET*.

Qn \ SN	00	01	11	10
0	0	0	ϕ	1
1	ϕ	0	ϕ	0

Tabla 1.21. Mapa para la simplicación del *SET*.

SET= S./Qn. (8)

Como se debe incluir el reloj en la ecuación anterior se tiene:

SET= S/Qn.CLK. (9)

La tabla 1.22 muestra el mapa para simplificar la señal *RESET*.

Qn \ SN	00	01	11	10
0	ϕ	ϕ	ϕ	0
1	0	1	ϕ	0

Tabla 1.22. Mapa para la simplicación del *RESET*.

RESET= N.

Como se debe incluir el reloj en la ecuación anterior se tiene:

RESET= N.CLK

Paso 5. El último paso es la implementación de las ecuaciones booleanas encontradas para *SET* y *RESET*.

1.9.5 *Latch* asincrónicos maestro-esclavo

Los *latch* asincrónicos maestro-esclavo son dispositivos que se construyen conectando en cascada dos *latch* asincrónicos del mismo tipo. El uno trabaja con el nivel alto del reloj y el otro con el mismo reloj, pero con el nivel bajo, mediante un inversor.

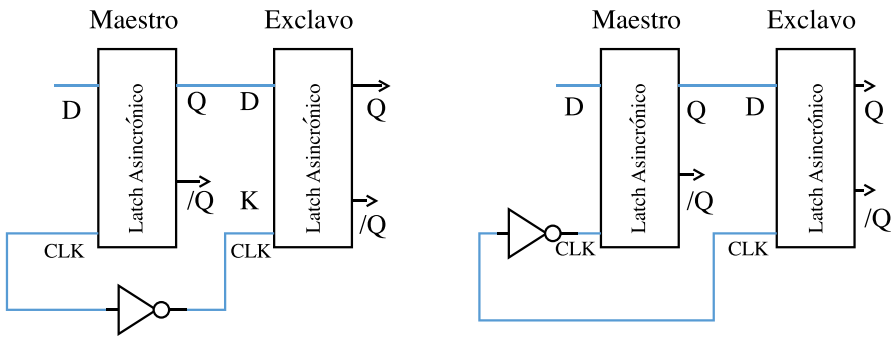


Figura 1.57. Diagramas de bloques de *latchs* asincrónicos maestro-esclavo D

La figura 1.57 muestra dos formas de construir *latch* asincrónicos maestro-esclavo tipo D. Nótese que el inversor está conectado al reloj de la etapa esclavo y, en el otro, al reloj de la etapa maestra.

Ejemplo 1.11

Analice el *latch* asincrónico que se muestra en la figura 1.57 a). Suponga que la señal de reloj y la señal D son las que se indican en la figura 1.58.

Una muy buena estrategia para analizar dispositivos conectados en serie es separar cada etapa y analizar cada una en forma independiente (como si las otras etapas no existieran), para este ejemplo se analiza primero la etapa maestra y luego la etapa esclava.

En la etapa maestra, cuando el reloj se encuentra en nivel alto, la señal D va a su salida Q, y cuando el reloj está en nivel bajo, la salida Q, se mantiene, no cambia de estado; en el gráfico 1.58 esa señal es la Q1.

La segunda etapa, la esclava, recibe en su entrada D la salida Q1 de la etapa maestra, como esta etapa se activa cuando el reloj está en nivel bajo, entonces la entrada Q1 va a la salida Q2 cuando el reloj está en nivel bajo.

En la figura 1.58 se muestra el diagrama de tiempo que resulta del análisis del *latch* asincrónico que se muestra en la figura 1.57 a).

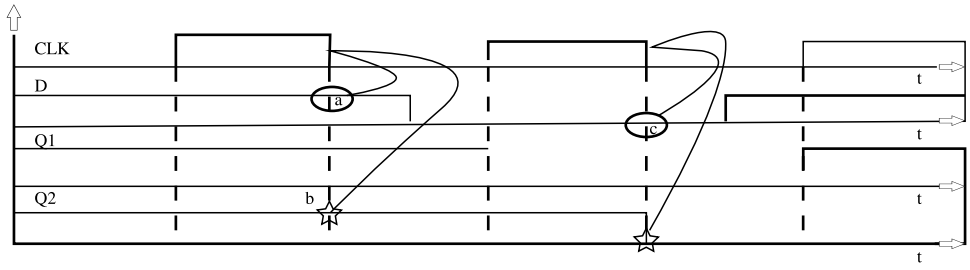


Figura1.58. Diagrama de tiempo para el ejemplo 1.11

Un análisis más detallado de la figura 1.58 muestra que la salida Q2 es igual al valor de la entrada D cuando el reloj está en su transición de nivel alto a nivel bajo, el flanco de bajada. Este dispositivo trabaja como si fuera activado por el flanco de bajada del reloj.

También se puede pensar que como la primera etapa trabaja con el reloj en nivel alto y la segunda etapa con el mismo reloj pero en nivel bajo, la coincidencia del reloj para las dos etapas es precisamente cuando el reloj está haciendo su transición de nivel alto a nivel bajo.

Las dos etapas, como un todo, deben responder a un solo reloj y el único lugar en donde los dos relojes coinciden es en la transición de nivel alto a bajo del reloj que se aplica a la etapa maestra.

De ahí que efectivamente este dispositivo maestro-esclavo trabaja con el flanco de bajada del reloj que ingresa a la etapa maestra.

Si se realiza el mismo análisis del dispositivo 1.57 b), se concluiría que el dispositivo trabaja con el flanco de subida del reloj; en este caso, las dos etapas, la maestra y esclava coinciden cuando el reloj está haciendo su transición de nivel bajo a nivel alto.

1.9.6 *Latch* asincrónicos maestro - esclavo JK

Un *latch* asincrónico maestro-esclavo tipo JK se construye uniendo en cascada dos *latch* asincrónicos tipo JK como muestra la figura 1.59 a) y b). En esa figura se ve que el primer *latch* asincrónico JK.

La etapa maestra, tiene conectado el reloj directamente y el otro *latch* asincrónico JK, la etapa esclava, tiene conectado el mismo reloj mediante un inver-

sor, esta conexión hace que la primera etapa responda al nivel alto del reloj y la segunda al nivel bajo del mismo reloj.

Las salidas Q y /Q de la primera etapa se conectan a la entrada J y K respectivamente de la segunda etapa, de tal manera que los datos ingresan por las entradas JK del primer *latch* asincrónico JK.

La etapa maestra, y la salida es la salida Q del segundo *latch* asincrónico JK, la etapa esclava.

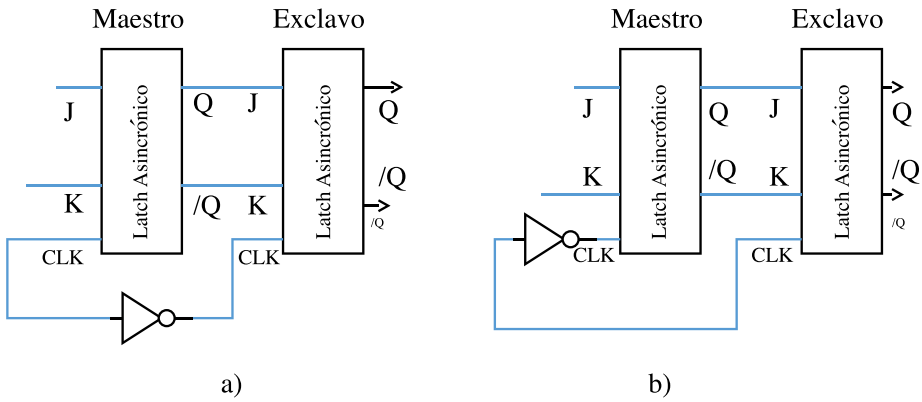


Figura 1.59. *Latchs* asincrónicos maestro-esclavo JK

Aplicando el mismo razonamiento que se aplicó al *latch* asincrónico maestro-esclavo tipo D, se puede concluir que si la etapa maestra del JK trabaja con el nivel alto del reloj y la etapa esclava con el nivel bajo del reloj, las dos etapas en conjunto deben trabajar con un único reloj y la única forma es cuando el nivel alto del reloj de la etapa maestra coincide con el nivel bajo del reloj de la etapa esclava y el lugar de coincidencia es la transición de nivel alto a nivel bajo. En conclusión, este dispositivo maestro-esclavo trabaja con el flanco de baja del reloj.

Respecto del dispositivo de la figura 1.59 b), aplicando el mismo razonamiento anterior, se puede concluir que trabaja con el flanco de subida del reloj que está aplicado en la etapa maestra.

1.10 Conversión entre *latch* asincrónicos

Los pasos que se pueden seguir para la conversión se indican a continuación:

1. Se comparan los diagramas de bloques.
2. Se dibuja, como una caja negra, el diagrama de bloques del *latch* asincrónico que se quiere obtener.
3. Se conectan, mediante una interfaz, las entradas de la caja negra a las entradas del latch asincrónico que se va a convertir.
4. Se diseña el decodificador.
5. Se implementa.

Este procedimiento se aplica a los ejemplos siguientes.

Ejemplo 1.12

Convierta o transforme un *latch* asincrónico tipo D, en un *latch* asincrónico tipo T.

Paso 1. Se comparan los diagramas de bloques. La figura 1.60 muestra los dos diagramas de bloques, del *latch* que se quiere convertir (D) y del *latch* al que se desea llegar (T).

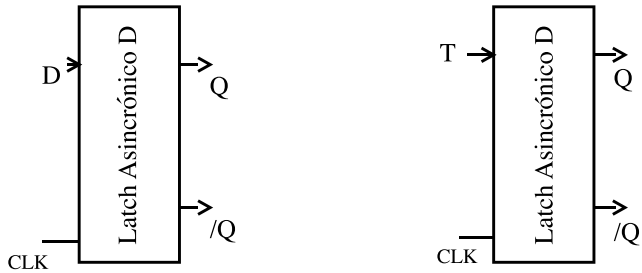


Figura 1.60. *Latch* asincrónico D y T

Al comparar las dos figuras a) y b), solo desde el punto de vista de diagramas de bloques y nada más, la diferencia es evidente y está en las entradas. El gráfico a) tiene una sola entrada externa denominada D aparte del reloj; y el bloques b) tiene una entrada denominada T aparte del reloj.

Paso 2. Se dibuja el diagrama de bloques del *latch* asincrónico, que se quiere obtener (T), como una caja negra, en su interior debe estar el *latch* asincrónico

que se quiere convertir (D). La figura 1.61 muestra este bloque.

Paso 3. Hay que conectar la entrada T a la única entrada disponible, es decir a D; lógicamente como no puede ser una conexión directa entra líneas, porque la línea D tiene un funcionamiento diferente de la línea T, es necesario buscar algún mecanismo para realizar esta conexión. Una forma de hacerlo es mediante una interfaz, un bloque intermedio, que una la entrada T y D como muestra la figura 1.62.

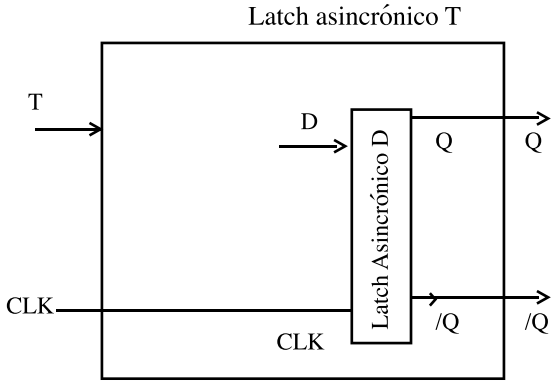


Figura 1.61. Bloque como caja negra

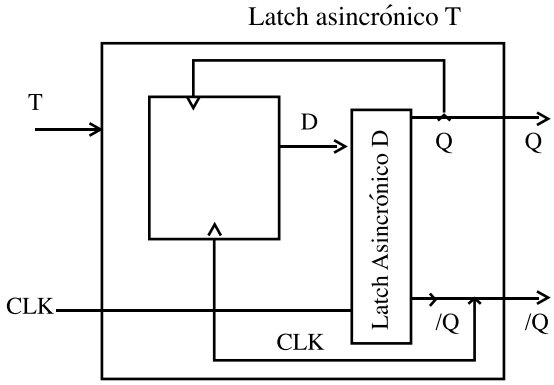


Figura 1.62. Conexión de T a D

En la figura 1.62, se han conectado a esta interfaz también las salidas Q y /Q porque esta interfaz necesita saber cuál es el estado en el que están las salidas Q y /Q, ya que según la tabla de funcionamiento del *latch* asincrónico T, su salida Q depende del valor de T.

Paso 4. En la figura 1.62, el único bloque desconocido es la interfaz. El *latch* asincrónico dado (D) se sabe ya cómo está construido, por lo que no hay que diseñar este bloque. Como no hay caminos de retroalimentación de las salidas del bloque de interfaz a las entradas del mismo se concluye que este bloque es un circuito combinacional, por lo tanto, se utiliza la tabla 1.23 para diseñar.

En la tabla de verdad 1.23, las entradas van del lado izquierdo y las salidas del lado derecho. No hay que olvidar que esta no es una tabla común, sino una tabla característica.

T	Qn	Qn+1	D
0	0	**0**	**0**
0	1	**1**	**1**
1	0	**1**	**1**
1	1	**0**	**0**

Tabla 1.23. Tabla para diseñar el decodificador T a D.

Por lo tanto, contiene el estado presente Qn y el estado siguiente Qn+1, la salida es D. De la tabla 1.23, se ve con claridad que la señal D es:

D= T exor Q. (10)

Paso 5. Implementación de las ecuaciones booleanas.

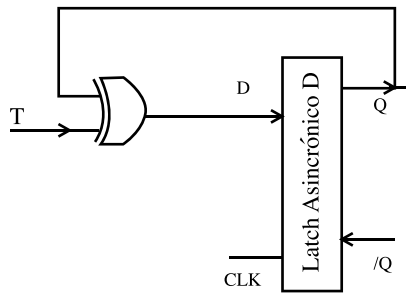


Figura 1.63. *Latch* asincrónico D convertido a T.

Ejemplo 1.13

Convierta o transforme un *latch* asincrónico tipo T, en un *latch* asincrónico tipo D.

Paso 1. Se comparan los diagramas de bloques. La figura 1.64 muestra los dos diagramas de bloques. El uno es el *latch* que se quiere convertir (D) y el otro es del *latch* al que se desea llegar (T).

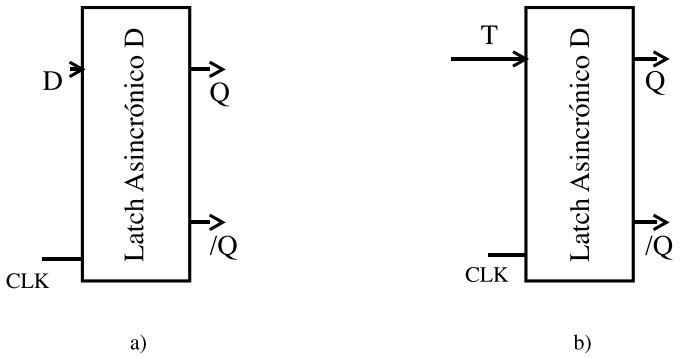


Figura 1.64. *Latch* asincrónico D y T

Al comparar las dos figuras a) y b), solo desde el punto de vista de diagramas de bloques y nada más, la diferencia es evidente y está en las entradas, el gráfico a tiene una sola entrada externa denominada D aparte del reloj y el bloque b tiene una entrada denominada T aparte del reloj.

Paso 2. Se dibuja el diagrama de bloques del *latch* asincrónico, que se quiere obtener (D), como una caja negra. En su interior debe estar el *latch* asincrónico que se quiere convertir (T), el *latch* dado. La figura 1.65 muestra este bloque.

Paso 3. Hay que conectar la entrada T a la única entrada disponible, es decir la D. Lógicamente como no puede ser una conexión directa entre líneas, porque la línea D (el latch D) tiene un funcionamiento diferente al de la línea T (el *latch* T), es necesario buscar algún mecanismo para realizar esta conexión, una forma de conectar es mediante una interfaz, un bloque intermedio que una las entradas T y D como muestra la figura 1.66.

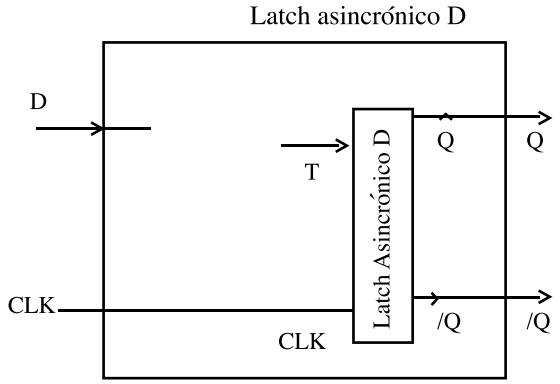


Figura 1.65. Bloque como caja negra

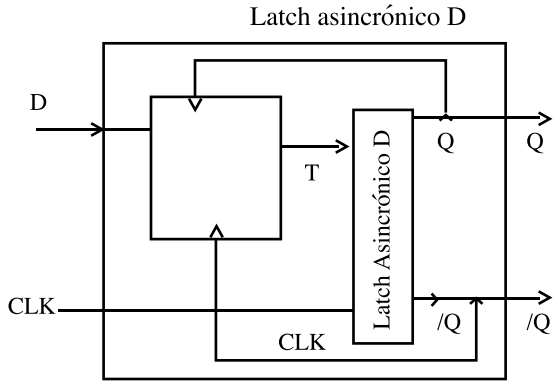


Figura 1.66. Conexión de D a T

Paso 4. En la figura 1.66, el bloque desconocido es la interfaz. El *latch* asincrónico dado (D) es conocido. No se diseña. El bloque de interfaz es un circuito combinacional y se utiliza la tabla 1.24 para diseñar.

En la tabla 1.24 las entradas van del lado izquierdo y las salidas del lado derecho, de la tabla característica; por lo tanto contiene el estado presente Qn y el estado siguiente Qn+1, la salida es D. De la tabla 1.24, la señal T es:

T	Qn	Qn+1	D
0	0	0	0
0	1	0	1
1	0	1	1
1	1	1	0

Tabla 1.24. Tabla para diseñar el decodificador T a D.

T= D exor Q.

Paso 5. La implementación de las ecuaciones booleanas encontradas.

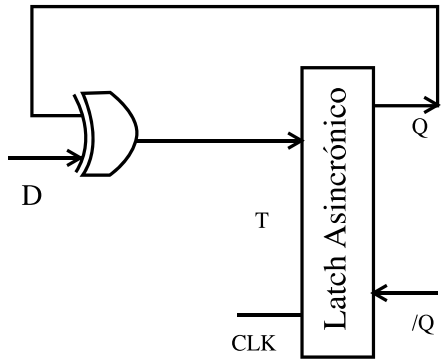


Figura 1.67. Implementación del circuito convertido

Ejemplo 1.14

Convierta o transforme un *latch* asincrónico tipo T, en un *latch* asincrónico tipo JK.

Paso 1. Se comparan los diagramas de bloques. La figura 1.68 muestra los dos diagramas de bloques, el uno es el *latch* que se quiere convertir (T) y el otro es del *latch* al que se desea llegar (JK).

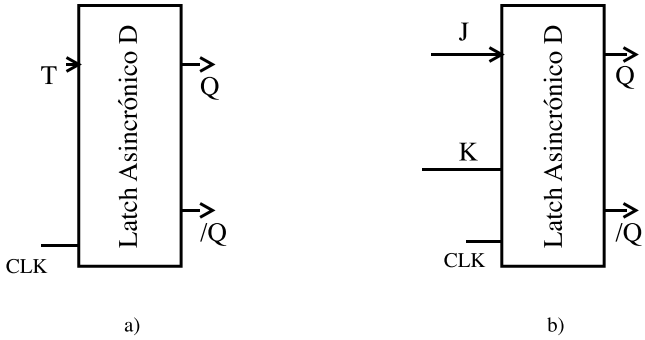


Figura 1.68. *Latch* asincrónico D y T

Al comparar las dos figura a) y b), solo desde el punto de vista de diagramas de bloques y nada más, la diferencia es evidente y está en las entradas. El gráfico a) tiene una sola entrada externa denominada T aparte del reloj; y el bloque b) tiene dos entradas denominadas JK aparte del reloj.

Paso 2. Se dibuja el diagrama de bloques del *latch* asincrónico, que se quiere obtener (JK), como una caja negra. En su interior, debe estar el *latch* asincrónico que se quiere convertir (T), el *latch* dado. La figura 1.69 muestra este bloque.

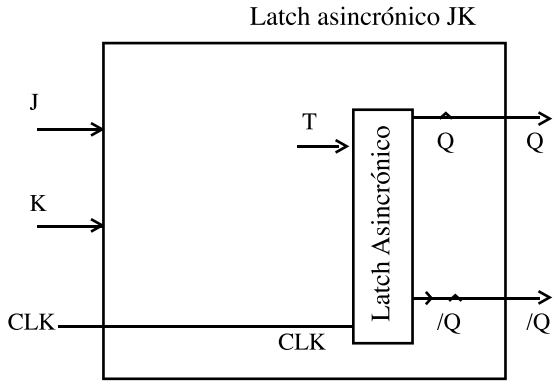


Figura 1.69. Bloque como caja negra

Paso 3. Hay que conectar las entras JK a las entradas disponibles, es decir a T. Lógicamente como no puede ser una conexión directa entre líneas, es necesario buscar algún mecanismo para realizar esta conexión. Una forma de conectar es mediante una interfaz, un bloque intermedio que una las entradas JK con la T como muestra la figura 1.70.

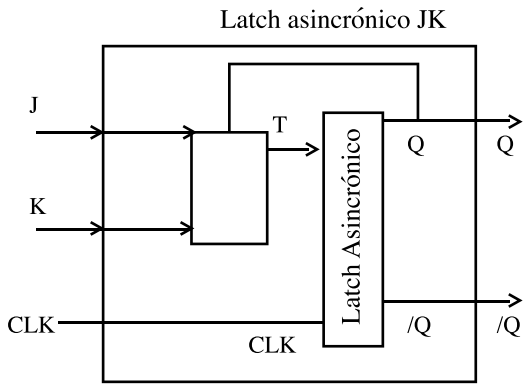


Figura 1.70. Conexión de JK a T

En la figura 1.70, se han conectado a esta interfaz también las salidas Q y /Q porque esta interfaz necesita saber cuál es el estado en el que están las salidas Q y /Q.

Paso 4. En la figura 1.70, el único bloque desconocido es la interfaz. El *latch* asincrónico dado (T) se sabe ya cómo está construido, por lo que no hay que diseñar este bloque. Como no hay caminos de retroalimentación directa de las salidas del bloque de interfaz a las entradas del mismo, este es un circuito combinacional; por lo tanto, se utiliza una tabla 1.25 para diseñar. En la tabla de verdad 1.25 las entradas van del lado izquierdo y las salidas del lado derecho. No hay que olvidar que esta no es una tabla común, sino una tabla característica por lo tanto, contiene el estado presente Qn y el estado siguiente Qn+1, la salida es T.

De la tabla 1.25, se ve con claridad que la señal T es:

$$T= J\ /Qn + K\ Qn \qquad (11)$$

T	K	Qn		Qn+1	T
0	0	0		0	0
0	0	1		1	0
0	1	0		0	0
0	1	1		0	1
1	0	0		1	1
1	0	1		1	0
1	1	0		1	1
1	1	1		0	1

Tabla 1.25. Tabla para diseñar el decodificador T a D

Paso 5. El último paso es la implementación de las ecuaciones booleanas. En este ejercicio, no se pide la implementación.

1.11 Los *flip-flops*

Los *latch* asincrónicos son sensibles a alguno de los niveles del reloj, si el *latch* es sensible al nivel alto, su salida Q se actualiza cada vez que el reloj está en nivel alto; por el contrario, si el *latch* es sensible al nivel bajo, su salida Q se actualiza cada vez que el reloj esté en nivel bajo.

En circuitos reales no es muy conveniente que el circuito actualice su salida Q cada vez y durante todo el tiemplo que el reloj esté en alguno de sus niveles, por ejemplo, el *latch* asincrónico tipo T, cuando la entrada T es igual a uno, de acuerdo a la tabla característica de funcionamiento de este dispositivo, la salida Q debe cambiar su valor a su opuesto, esto implica que Q debe actualizarse, cambiar a su valor opuesto, Qn+1= /Qn, continuamente, mientras el reloj este en nivel alto. La consecuencia de esto es que Q oscilará entre alto y bajo mientras T sea igual a uno y el reloj este en nivel alto.

Los circuitos secuenciales sincrónicos, requieren que su elemento de memoria actualice su estado (Q) solo una vez en cada período del reloj, la razón es que en las salidas Q del elemento de memoria, está el código del estado, y este debe mantenerse exactamente un período del reloj. Si se utiliza un *latch* asincrónico tipo T como el elemento que almacena el código del estado cuando T sea igual a uno y el reloj esté en nivel alto, la salida Q cambiara una gran cantidad de veces (el número depende del retardo de propagación del dispositivo) y por ende el código del estado cambiaría también una gran cantidad de veces.

Los *flip-flops* precisamente son dispositivos de memoria sensibles a uno de los flancos del reloj. ¿Qué ventaja tiene el hecho de que sean sensibles al flanco y no al nivel del reloj? La ventaja es que se pueden actualizar solo durante el flanco del reloj y, por ende, obligatoriamente se actualizan una sola vez en cada período del reloj. Por esta razón los *flip-flops* son utilizados como elementos de memoria en toda máquina secuencial sincrónica.

Cada *latch* asincrónico tiene su respectivo *flip-flop*, es decir, si hay un *latch* asincrónico tipo D, hay también un *flip-flop* tipo D, si hay un *latch* asincrónico tipo T hay también un *flip-flop* tipo T y así por el estilo.

Los *flip-flops* típicos son los siguientes:

1. El *flip-flop* tipo D.
2. El *flip-flop* tipo T.
3. El *flip-flop* tipo JK.

Por supuesto al igual que los *latch* asincrónicos se pueden diseñar *flip-flops*, que se ajusten a necesidades particulares.

Los *flip-flops* tienen su tabla característica idéntica a la tabla característica de su correspondiente *latch* asincrónico, es decir el *latch* asincrónico tipo D tiene la misma tabla característica que el *flip-flop* tipo D.

A pesar de que las tablas características de los *latch* son idénticas a las de sus correspondientes *flip-flops* su estructura interna es diferente.

Si un *flip-flop* es sensible al flanco de subida de su reloj, se dice que es verdadero con el flanco de subida, o simplemente verdadero.

Si un *flip-flop* es sensible al flanco de bajada del reloj, se dice que es verdadero con el flanco de bajada del reloj o simplemente verdadero.

1.11.1 El *flip-flop* tipo D

El *flip-flop* tipo D, al igual que el *latch* asincrónico tipo D, funciona como un interruptor controlado por su reloj.

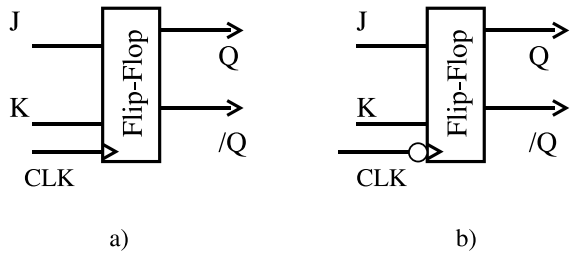


Figura 1.71. *flip-flop* D a) sensible al flanco de subida, b) sensible al flanco de bajada

La diferencia con el *latch* está en que, en este, el interruptor está controlado por uno de los niveles del reloj; en el *flip-flop* el interruptor está controlado por uno de sus flancos del reloj, es decir, cuando y solo cuando el reloj presenta su flanco de subida (o bajada) en caso de que sea sensible al flanco de subida (verdadero con el flanco de subida) la entrada D pasa a la salida Q (se actualiza); de lo contrario, su salida Q no se actualiza.

La figura 1.71 muestra el diagrama de bloque general de este *flip-flop*. La parte a) muestra el *flip-flop* con el símbolo del reloj verdadero con el flanco de subida y, en la parte b), se indica el *flip-flop* con la simbología del reloj disparado por el flanco de bajada.

Nótese que un reloj sensible al flanco de subida (verdadero con el flanco de subida) tiene una simbología especial y es una flecha con la punta dentro del dispositivo. En el *flip-flop* sensible al flanco de bajada, su simbología es una recta que termina en un círculo seguido de una flecha dentro del dispositivo.

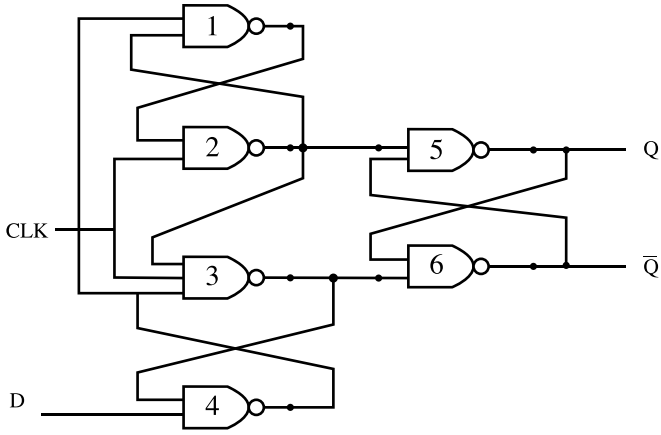


Figura 1.72. *flip-flop* tipo D sensible al flanco de subida

La figura 1.72 muestra la estructura interna del *flip-flop* tipo D. Como se puede ver, está construido con tres *latch* básicos, con compuertas NAND, dispuestos de una manera especial; esa disposición especial le hace sensible al flanco del reloj.

La tabla 1.26 a) es para un F*flip-flop* verdadero con el flanco de subida del reloj, en tanto que la figura 1.71 b) es para un *flip-flop* verdadero con el flanco de bajada del reloj.

CLK	D	Qn	Qn+1
↑	0	0	0
↑	0	1	0
↑	1	0	1
↑	1	1	1

CLK verdadero en el flanco de subida

a)

CLK	D	Qn	Qn+1
↓	0	0	0
↓	0	1	0
↓	1	0	1
↓	1	1	1

CLK verdadero en el flanco de bajada

b)

Tabla 1.26. Tabla característica del *flip-flop* D

1.11.2 El *flip-flop* T

El *flip-flop* tipo T, al igual que el *latch* asincrónico T, cambia su estado al opuesto (Qn+1= /Qn) cuando la entrada T es igual a uno y el reloj es verdadero.

Cuando T es igual a cero y el reloj, verdadero la salida Q no cambia (Qn+1 = Qn). Si el reloj es falso, tampoco se actualiza, Qn+1=Qn. La figura 1.73 muestra el diagrama de bloques de este *flip-flop*. La tabla 1.27 muestra la tabla característica de este *flip-flop*.

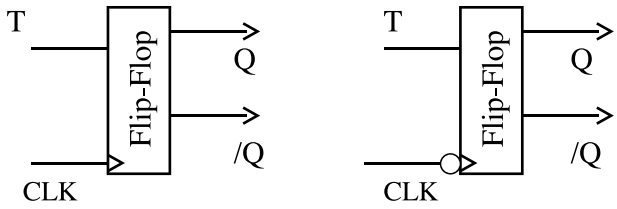


Figura 1.73. *Flip-flop* T sensible al flanco a) de subida b) de bajada

CLK	T	Qn	Qn+1
↑	0	0	0
↑	0	1	1
↑	1	0	1
↑	1	1	0

a)

CLK verdadero en el flanco de subida

CLK	D	Qn	Qn+1
↓	0	0	0
↓	0	1	1
↓	1	0	1
↓	1	1	0

b)

CLK verdadero en el flanco de bajada

Tabla 1.27. Tablas características del *flip-flop* T

Como todo *flip-flop*, este dispositivo es sensible a uno de sus flancos del reloj, al flanco de subida o al de bajada.

1.11.3 El *flip-flop* JK

El *flip-flop* tipo JK, al igual que el *latch* asincrónico JK, mantiene su estado, la salida Q no cambia, siempre que JK=00 y el reloj verdadero, cuando JK=10 y el reloj verdadero la salida Q es igual a uno, cuando JK=01 y el reloj verdadero la salida Q es igual a cero, si JK=11 el estado siguiente es el negado del estado presente, es decir la salida Q conmuta, y como en todo *flip-flop* si el reloj es falso no actualiza su salida Q. La figura 1.74 muestra el diagrama de bloques de este *flip-flop*.

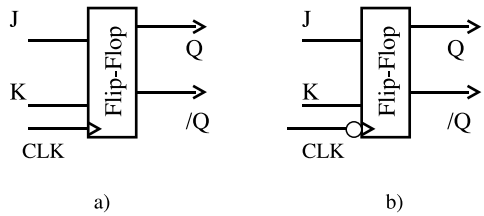


Figura 1.74. *Flip-flop* JK, a) sensible al flanco de subida, b) sensible al flanco de bajada

CLK	J	K	Qn	Qn+1	
↑	0	0	0	0	CLK verdadero en el flanco de subida
↑	0	0	1	1	
↑	0	1	0	0	
↑	0	1	1	0	
↑	1	0	0	1	
↑	1	0	1	1	
↑	1	1	0	1	
↑	1	1	1	0	

a)

CLK	J	K	Qn	Qn+1	
↓	0	0	0	0	CLK verdadero en el flanco de bajada
↓	0	0	1	1	
↓	0	1	0	0	
↓	0	1	1	0	
↓	1	0	0	1	
↓	1	0	1	1	
↓	1	1	0	1	
↓	1	1	1	0	

b)

Tabla 1.28. Tabla del *flip-flop* JK sensible al flanco a) de subida b) al de bajada

Ejemplo 1.15

Para el circuito que muestra la figura 1.75, dibuje las formas de onda de la salida Q1 y Q2. La figura 1.80 muestra la señal X y el reloj, y en la tabla 1.28, muestra la tabla del *flip-flop* JK sensible al flanco de subida y de bajada.

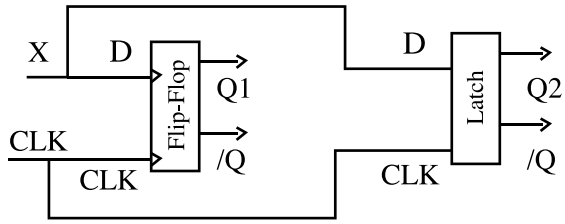


Figura 1.75. Diagrama de tiempo para el ejercicio

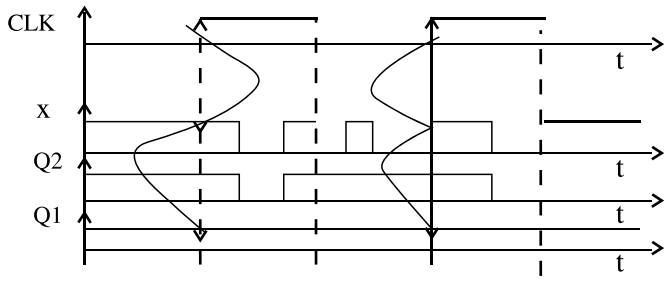


Figura 1.76. Salidas Q1 y Q2 del circuito de la figura 1.75

Como puede notarse en la figura 1.76, la salida Q2 del *latch* es idéntica a la entrada X cuando el reloj está en nivel alto, cuando el reloj está en nivel bajo la salida Q2 mantiene en ultimo valor.

En el caso del *flip-flop* la salida Q1 solo se actualiza cuando el reloj está en su flanco de subida y mantiene el valor de Q1 para el *RESET* o de los valores del reloj.

1.12 Retardos de propagación en *flip-flops*

Es necesario indicar que los análisis realizados anteriormente no han considerado los retardos de propagación de los *flip-flops*, esto es importante porque si

las señales que ingresan a las entradas de los *flip-flops* cambian al mismo tiempo que el reloj, los resultados en sus salidas pueden ser impredecibles.

En la figura 1.77, se muestra la salida Q de un *flip-flop* tipo D despreciando los retardos de propagación.

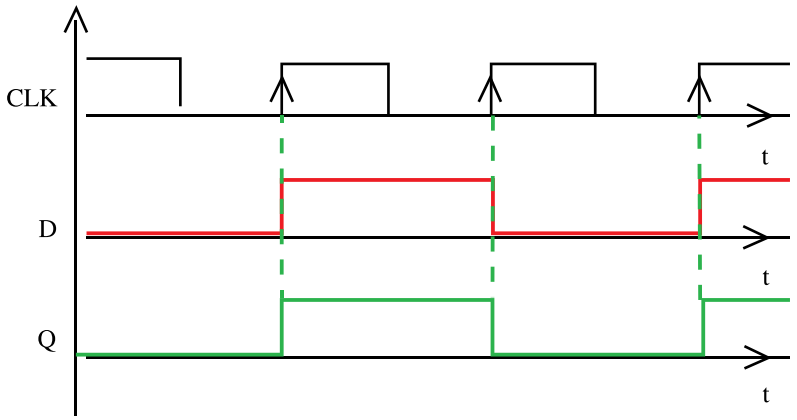


Figura 1.77. Diagrama de tiempo de un *flip-flop* ideal

Como se puede ver en la figura 1.77 tanto la entrada D como el reloj han cambiado al mismo tiempo y la respuesta del *flip-flop* como se indica en la señal Q. Esta repuesta es posible solo si se trata de un *flip-flop* ideal con un retardo de propagación igual a cero. Por supuesto en la realidad, la salida Q puede ser inesperada para esos cambios simultáneos en la entrada y el reloj.

En los *flip-flops* reales, es necesario que las entradas permanezcan estables durante un cierto tiempo, denominado tiempo de preparación (tsu), antes de que la señal del reloj cambie y deben además permanecer estables por un tiempo, denominado tiempo de espera (th), después de que el reloj ha cambiado.

En la figura 1.78, se muestra cómo debe ser la entrada D para que el dispositivo trabaje adecuadamente.

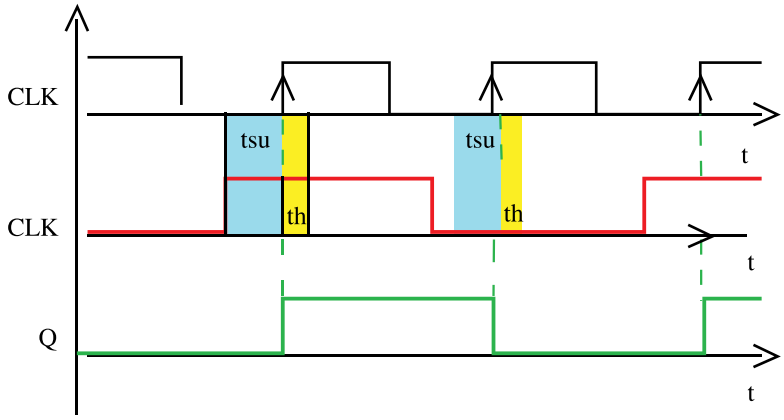


Figura 1.78. Diagrama de tiempo de un *flip-flop* real

2. EJERCICIOS PROPUESTOS

1. Indique las características de un circuito combinacional.
2. Indique las características de un circuito secuencial.
3. ¿Por qué son necesarios los circuitos secuenciales?
4. ¿Qué se entiende por camino de retroalimentación?
5. Dibuje un circuito que tenga cuatro caminos de realimentación.
6. ¿Qué es una memoria elemental? Dibuje su circuito.
7. Demuestre mediante ecuaciones booleanas la operación de un *latch* básico que trabaja con lógica positiva.
8. Demuestre mediante ecuaciones booleanas la operación de un *latch* básico que trabaja con lógica negativa.
9. Escriba la tabla característica de un *latch* básico en función de valores verdaderos y falsos.
10. Transforme la tabla de verdad del ejercicio anterior a lógica positiva.
11. Transforme la tabla de verdad del ejercicio anterior a lógica negativa.
12. ¿De qué depende el valor de Q de la salida de un *latch* básico cuando sus dos entradas *SET* y *RESET* son verdaderas simultáneamente?
13. Explique qué se entiende por rebote de un interruptor.
14. Dibuje un *latch* básico que elimine rebotes de un interruptor mecánico y explique cómo el latch elimina los rebotes.
15. ¿Qué es un *latch* asincrónico y cuál es la diferencia con el *latch* básico?
16. Diseñe un *latch* asincrónico que tenga dos entradas x e y. Si la entrada x es igual a uno y la entrada y es cero el estado siguiente debe ser el opuesto del estado presente, para el *RESET* o dé los casos en que el estado siguiente debe ser igual al estado presente.
17. Escriba la tabla característica del *latch* asincrónico tipo D. y explique su funcionamiento.
18. Escriba la tabla característica del *latch* asincrónico tipo T y explique su funcionamiento.

19. Escriba la tabla característica del *latch* asincrónico tipo JK y explique su funcionamiento.
20. Convierta el *latch* asincrónico tipo JK en el latch asincrónico SR.
21. ¿Cuál es la diferencia entre un dispositivo sensible al nivel alto de una señal de reloj y un dispositivo que sea sensible al flanco de subida del mismo reloj?
22. ¿Que es un *flip-flop* y cuál es la diferencia con un *latch*?
23. ¿Dibuje la estructura interna de un *flip-flop* tipo D y explique por qué este dispositivo es sensible al flanco de subida?
24. Se podría decir que a partir de un *flip-flop* D se pueden obtener cualquier otro tipo de *flip-flop*.
25. Escriba la tabla característica de un *flip-flop* tipo JK.
26. Escriba la tabla característica de un *flip-flop* tipo T.

BIBLIOGRAFÍA

Baldeón, W. (2010). *Problemas resueltos de sistemas digitales*. Riobamba: Ecopycenter.

Baldeon, W. (2010). *Analisis de señales*. Riobamba: Ecopycenter.

Brown, S., Vranesic, Z. (2006). *Fundamentos de lógica digital con diseño VHDL*. Mexico: Mc Graw Hill.

Maxinez, D. (2014). *Programación de sistemas digitales con VHDL*. México DF: Grupo Editorial Patria.

Morris, M. (2003). *Diseño digital*. Mexico DF: Mc Graw Hill.

Sánchez, L. (2018). *Problemas de electrónica digital*. Valencia: Universidad Politécnica de Valencia.

Wakerly, J. (2001). Diseño digital principios y prácticas. México DF: Prentice Hall.

La ciencia, en la que se fundamenta los sistemas digitales, en lo relativo a sus conceptos y fundamentos, no ha sufrido un cambio radical todavía, sin embargo, el diseño y aplicación práctica de esta ciencia (la implementación) ha cambiado sustancialmente en las últimas décadas con la aparición de los lenguajes de descripción de hardware (HDL) y herramientas CAD.

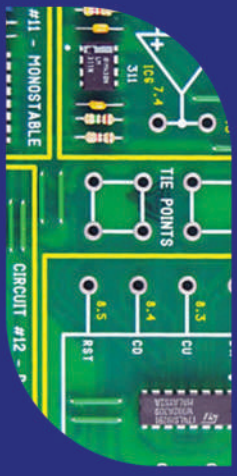
Este libro expone los conceptos fundamentales de los latchs y flip-flops desde una perspectiva clásica, pues su fundamento científico no ha cambiado.

Existen infinidad de libros sobre sistemas digitales, sin embargo, este se adapta a las necesidades académicas y de laboratorios de las carreras de Ingeniería Electrónica de la ESPOCH. De ahí que uno de los objetivos de este libro es resolver estas necesidades.

El texto presenta su contenido de una forma en que el lector puede desarrollar habilidad intuitiva para entender y aplicar los conceptos fundamentales, la estructura, el análisis, el diseño y la conversión tanto de latchs, como de flip-flops. Se exponen ejemplos que permiten reforzar los conocimientos que el lector va adquiriendo.

Wilson Oswaldo Baldeón López es ingeniero electrónico graduado en la Escuela Superior Politécnica del Litoral; es máster en Informática graduado en Chile; es máster en Diseño de Sistemas Electrónicos por la Universidad Tecnológica de La Habana, Cuba; es magíster en Gestión Académica Universitaria, tiene un diplomado superior en Pedagogía Universitaria y es experto en procesos *e-learning*.

Verónica Elizabeth Mora Chunllo es ingeniera en electrónica y computación graduada en la Escuela Superior Politécnica de Chimborazo; magíster en Ingeniería de Software graduada en la Escuela Superior Politécnica del Ejército; es máster en Diseño de Sistemas Electrónicos por la Universidad Tecnológica de La Habana, Cuba; es magíster en Educación a Distancia, tiene un diplomado superior en las Nuevas Tecnologías de Información y Comunicación Aplicadas a la Práctica Docente, es experta en procesos *e-learning*.



ISBN: 978-9942-35-650-5