Práctica 2: Lógica Digital - Secuenciales

Primer Cuatrimestre 2023

Organización del Computador I
DC - UBA

Introducción

Sobre la clase de hoy

Hoy vamos a ver los principios de diseño, práctica y ejemplos de circuitos secuenciales, la estructura de la clase va ser la siguiente:

- **Circuitos combinatorios - Timing**
- **Retroalimentación y cambio de modelo**
- **Circuitos secuenciales asincrónicos**
- **Circuitos secuenciales sincrónicos**
- **Flip-flops, registros y memorias**
- **Máquinas de estado**

Sobre la clase de hoy

Hoy vamos a ver los principios de diseño, práctica y ejemplos de circuitos secuenciales, la estructura de la clase va ser la siguiente:

- **Circuitos combinatorios - Timing**
- **Retroalimentación y cambio de modelo**
- **Circuitos secuenciales asincrónicos**
- **Circuitos secuenciales sincrónicos**
- **Flip-flops, registros y memorias**
- **Máquinas de estado**

Sobre la clase de hoy

Hoy vamos a ver los principios de diseño, práctica y ejemplos de circuitos secuenciales, la estructura de la clase va ser la siguiente:

- **Circuitos combinatorios - Timing**
- **Retroalimentación y cambio de modelo**
- **Circuitos secuenciales asincrónicos**
- **Circuitos secuenciales sincrónicos**
- **Flip-flops, registros y memorias**
- **Máquinas de estado**

Sobre la clase de hoy

Hoy vamos a ver los principios de diseño, práctica y ejemplos de circuitos secuenciales, la estructura de la clase va ser la siguiente:

- **Circuitos combinatorios - Timing**
- **Retroalimentación y cambio de modelo**
- **Circuitos secuenciales asincrónicos**
- **Circuitos secuenciales sincrónicos**
- **Flip-flops, registros y memorias**
- **Máquinas de estado**

Sobre la clase de hoy

Hoy vamos a ver los principios de diseño, práctica y ejemplos de circuitos secuenciales, la estructura de la clase va ser la siguiente:

- **Circuitos combinatorios - Timing**
- **Retroalimentación y cambio de modelo**
- **Circuitos secuenciales asincrónicos**
- **Circuitos secuenciales sincrónicos**
- **Flip-flops, registros y memorias**
- **Máquinas de estado**

Sobre la clase de hoy

Hoy vamos a ver los principios de diseño, práctica y ejemplos de circuitos secuenciales, la estructura de la clase va ser la siguiente:

- **Circuitos combinatorios - Timing**
- **Retroalimentación y cambio de modelo**
- **Circuitos secuenciales asincrónicos**
- **Circuitos secuenciales sincrónicos**
- **Flip-flops, registros y memorias**
- **Máquinas de estado**

Sobre la clase de hoy

Hoy vamos a ver los principios de diseño, práctica y ejemplos de circuitos secuenciales, la estructura de la clase va ser la siguiente:

- **Circuitos combinatorios - Timing**
- **Retroalimentación y cambio de modelo**
- **Circuitos secuenciales asincrónicos**
- **Circuitos secuenciales sincrónicos**
- **Flip-flops, registros y memorias**
- **Máquinas de estado**

Capas de abstracción

Un sistema complejo puede ser analizado desde diferentes niveles de abstraccion.

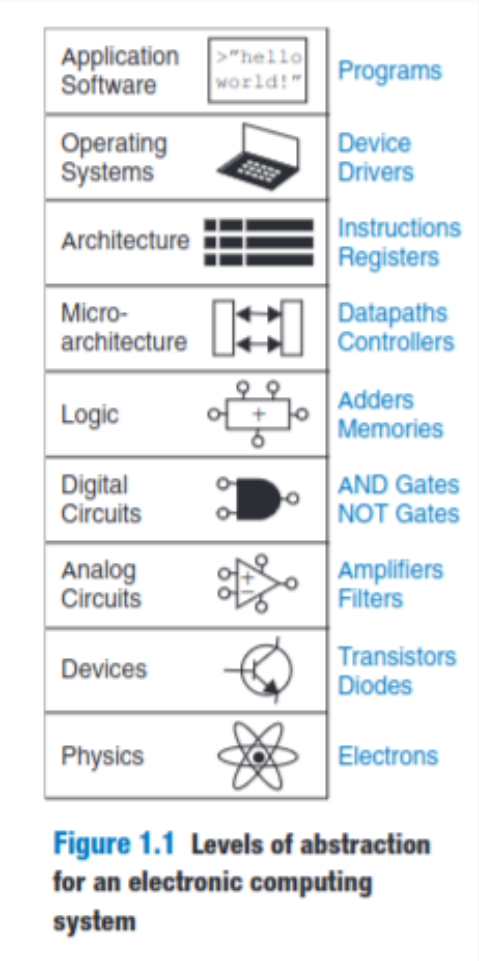


Figure 1.1 Levels of abstraction for an electronic computing system

Timing

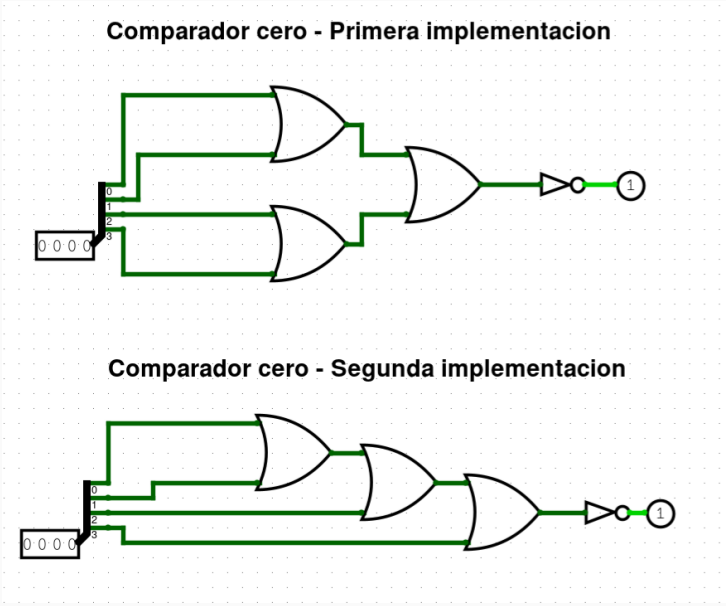
Timing

Las compuertas no son instantáneas.

Cada componente discreto va a tener un retardo característico que representa **la cantidad tiempo transcurrido entre que llegaron a estabilizarse sus salidas respecto del primer instante en que se estabilizaron las entradas**.

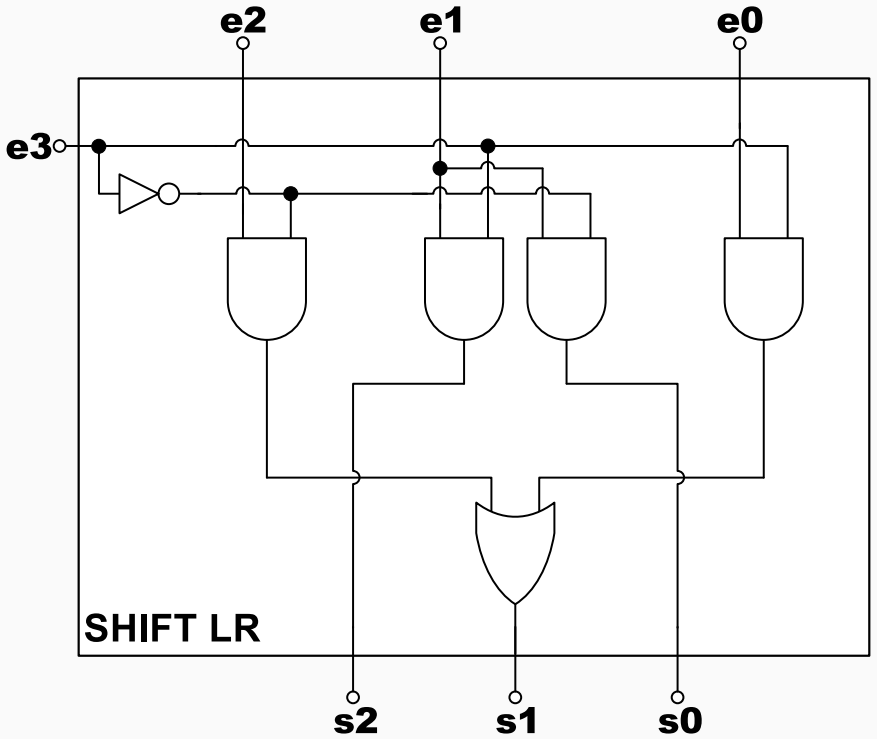
Timing - Ejercicio

Se tienen dos implementaciones distintas de un comparador cero de cuatro bits. Queremos calcular el tiempo que tarda el circuito en computar el resultado teniendo en cuenta que el retardo de cada compuerta es de 10ps.

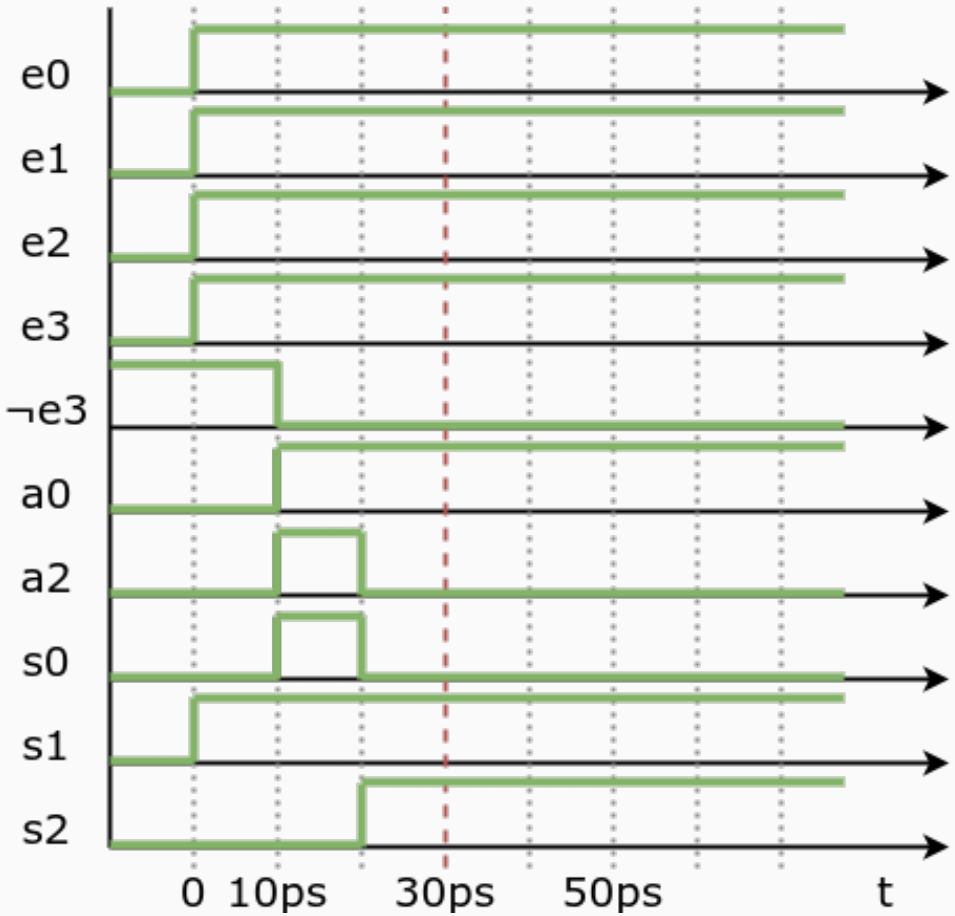


Timing

Otro ejemplo:



Timing



Circuitos combinatorios

Ahora volviendo un poco a los circuitos combinatorios, hay dos motivos fundamentales por los cuales podemos representar su comportamiento como una tabla que asigna valores a las salidas en base a los valores de las entradas, y tienen que ver con que:

1. **los tiempos de propagación** quedan fuera de nuestro modelo bajo el principio de abstracción.
2. no hay *retroalimentación de señales*.

Circuitos combinatorios

Ahora volviendo un poco a los circuitos combinatorios, hay dos motivos fundamentales por los cuales podemos representar su comportamiento como una tabla que asigna valores a las salidas en base a los valores de las entradas, y tienen que ver con que:

1. **los tiempos de propagación** quedan fuera de nuestro modelo bajo el principio de abstracción.
2. no hay *retroalimentación de señales*.

Circuitos combinatorios

Ahora volviendo un poco a los circuitos combinatorios, hay dos motivos fundamentales por los cuales podemos representar su comportamiento como una tabla que asigna valores a las salidas en base a los valores de las entradas, y tienen que ver con que:

1. **los tiempos de propagación** quedan fuera de nuestro modelo bajo el principio de abstracción.
2. no hay *retroalimentación de señales*.

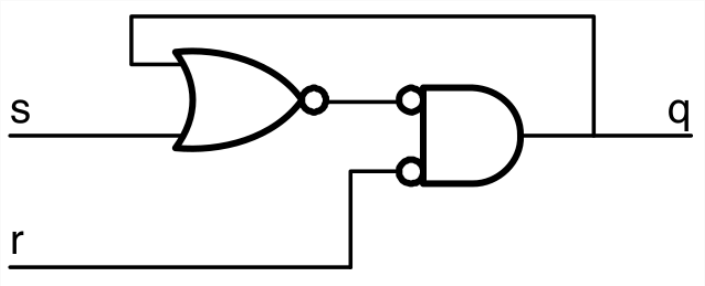
Retroalimentación y cambio de modelo

Retroalimentación

Ahora supongamos que queremos construir un circuito que permita conservar un bit de información. Básicamente queremos poder indicar cuando el bit vale 1 (activando la señal **s**et) y cuando vale 0 (activando la señal **r**eset), y proponemos la siguiente configuración:

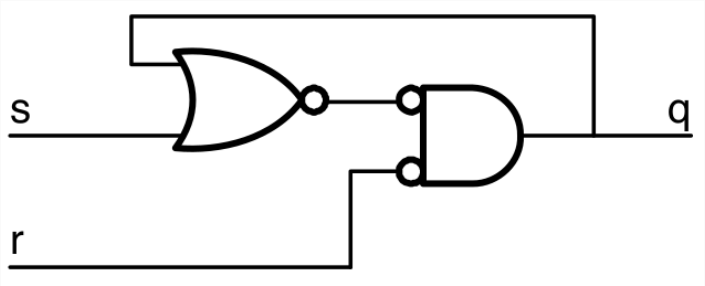
Retroalimentación

Ahora supongamos que queremos construir un circuito que permita conservar un bit de información. Básicamente queremos poder indicar cuando el bit vale 1 (activando la señal **s**et) y cuando vale 0 (activando la señal **r**eset), y proponemos la siguiente configuración:



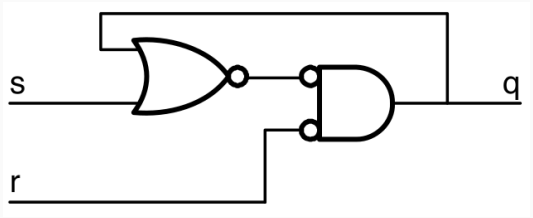
Retroalimentación

Ahora supongamos que queremos construir un circuito que permita conservar un bit de información. Básicamente queremos poder indicar cuando el bit vale 1 (activando la señal **s**et) y cuando vale 0 (activando la señal **r**eset), y proponemos la siguiente configuración:



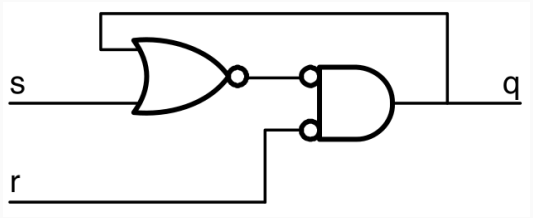
¿Qué valor tiene q para cada par (s, r), cuándo se estabiliza?

Retroalimentación



Intentemos armar su tabla de verdad.

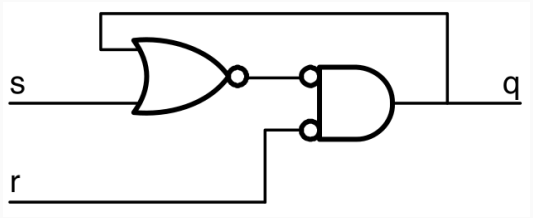
Retroalimentación



Intentemos armar su tabla de verdad.

s	r	q
0	0	?
0	1	?
1	0	?
1	1	?

Retroalimentación

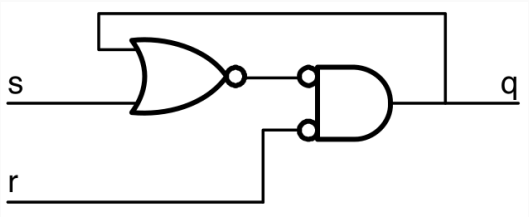


Intentemos armar su tabla de verdad.

s	r	q
0	0	?
0	1	?
1	0	?
1	1	?

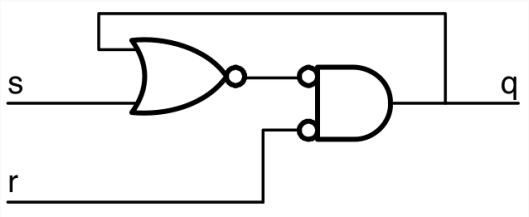
¿El valor de q es el actual o el estabilizado?

Retroalimentación



Definamos cómo se estabilizará (q') en base a su valor actual (q).

Retroalimentación



Definamos cómo se estabilizará (q') en base a su valor actual (q).

s	r	q	q'
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	0
1	1	1	0

Retroalimentación

Para poder razonar sobre el comportamiento del circuito debemos representar todas las señales, **de entrada (s,r), de salida (q') e internas o retroalimentadas (q)**.

s	r	q	q'
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	0
1	1	1	0

Retroalimentación

Vamos a llamar **valuación** a cada asignación de valores de verdad a las variables del circuito y representan todos los estados posibles que puede tener el circuito. En nuestro caso las variables del circuito son s, r y q, ya que q' está indicando los valores que puede tomar q en el próximo estado.

s	r	q	q'
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	0
1	1	1	0

Retroalimentación

Al introducir la retroalimentación y junto con ella la dependencia de señales internas, el circuito no puede pasar de una valuación a cualquiera otra, por ejemplo:

	Estado			
	s	r	q	q'
1:	0	0	0	0
2:	0	0	1	1

No podemos pasar del estado indicado en la primera fila $\langle 000 \rangle$ al estado indicado en la segunda $\langle 001 \rangle$ porque el valor actual de la salida (q) depende del valor que tenía en el estado anterior (q').

Retroalimentación

Intentemos modelar el comportamiento del circuito como los posibles cambios en su conjunto de señales.

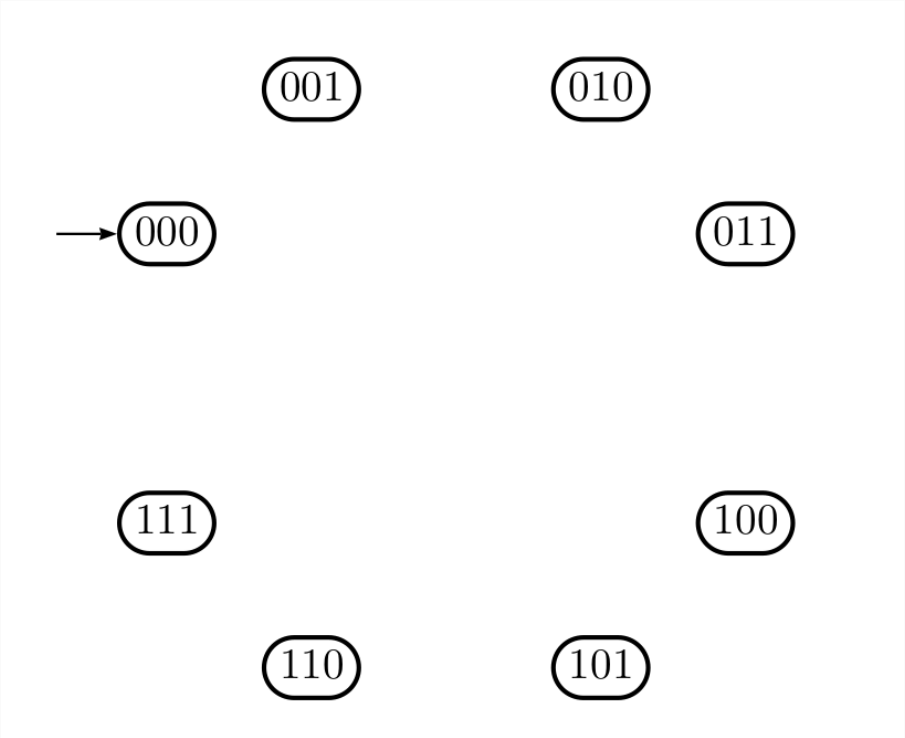
s	r	q	q'
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0

Retroalimentación

Representemos las distintas valuaciones.

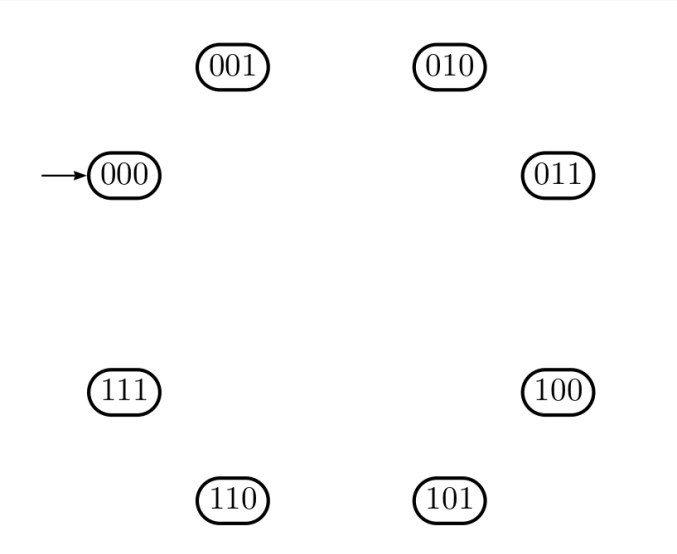
Retroalimentación

Representemos las distintas valuaciones.



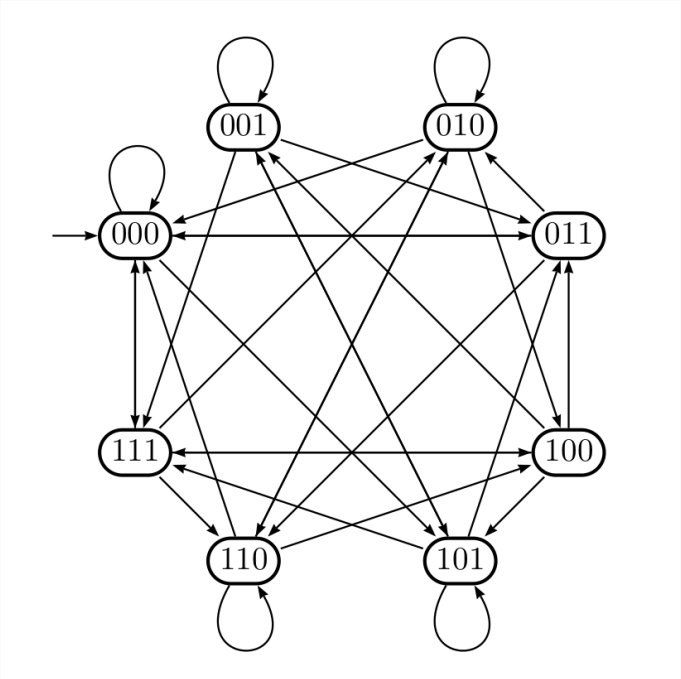
Retroalimentación

¿Cuáles son los cambios posibles de valores en las variables del circuito?



Retroalimentación

¿Cuáles son los cambios posibles de valores en las variables del circuito?



Retroalimentación

Se trata de un tipo de modelo llamado **máquina de estados** que permite representar, razonar y probar propiedades sobre el comportamiento de objetos de dominio heterogéneo, entre ellos:

- circuitos digitales
- protocolos de comunicación
- interfaces de programación de aplicaciones
- ejecuciones de un programa

Retroalimentación

Se trata de un tipo de modelo llamado **máquina de estados** que permite representar, razonar y probar propiedades sobre el comportamiento de objetos de dominio heterogéneo, entre ellos:

- circuitos digitales
- protocolos de comunicación
- interfaces de programación de aplicaciones
- ejecuciones de un programa

Retroalimentación

Se trata de un tipo de modelo llamado **máquina de estados** que permite representar, razonar y probar propiedades sobre el comportamiento de objetos de dominio heterogéneo, entre ellos:

- circuitos digitales
- protocolos de comunicación
- interfaces de programación de aplicaciones
- ejecuciones de un programa

Retroalimentación

Se trata de un tipo de modelo llamado **máquina de estados** que permite representar, razonar y probar propiedades sobre el comportamiento de objetos de dominio heterogéneo, entre ellos:

- circuitos digitales
- protocolos de comunicación
- interfaces de programación de aplicaciones
- ejecuciones de un programa

Retroalimentación

Se trata de un tipo de modelo llamado **máquina de estados** que permite representar, razonar y probar propiedades sobre el comportamiento de objetos de dominio heterogéneo, entre ellos:

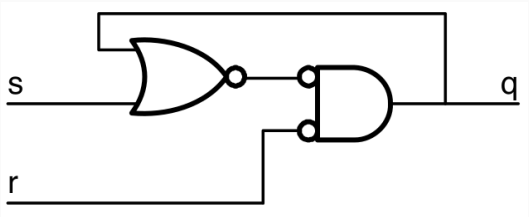
- circuitos digitales
- protocolos de comunicación
- interfaces de programación de aplicaciones
- ejecuciones de un programa

Retroalimentación

Repasando, queríamos implementar un circuito que pudiese almacenar un bit de información.

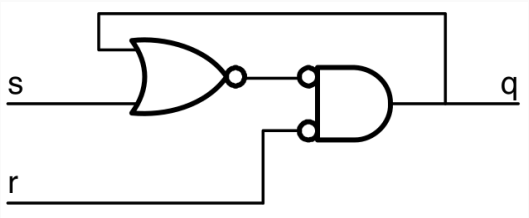
Retroalimentación

Repasando, queríamos implementar un circuito que pudiese almacenar un bit de información.



Retroalimentación

Repasando, queríamos implementar un circuito que pudiese almacenar un bit de información.



s	r	q	q'
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	0
1	1	1	0

Retroalimentación

Al retroalimentar una de las señales tuvimos que:

- abandonar el enfoque funcional para describir al circuito
- desdoblar los valores de la salida q' de su valor previo q
- introducir la noción de **valuación, estado** y **comportamiento**
- presentar informalmente la idea de **máquinas de estados**

Retroalimentación

Al retroalimentar una de las señales tuvimos que:

- abandonar el enfoque funcional para describir al circuito
- desdoblar los valores de la salida q' de su valor previo q
- introducir la noción de **valuación, estado** y **comportamiento**
- presentar informalmente la idea de **máquinas de estados**

Retroalimentación

Al retroalimentar una de las señales tuvimos que:

- abandonar el enfoque funcional para describir al circuito
- desdoblar los valores de la salida q' de su valor previo q
- introducir la noción de **valuación, estado** y **comportamiento**
- presentar informalmente la idea de **máquinas de estados**

Retroalimentación

Al retroalimentar una de las señales tuvimos que:

- abandonar el enfoque funcional para describir al circuito
- desdoblar los valores de la salida q' de su valor previo q
- introducir la noción de **valuación, estado** y **comportamiento**
- presentar informalmente la idea de **máquinas de estados**

Retroalimentación

Al retroalimentar una de las señales tuvimos que:

- abandonar el enfoque funcional para describir al circuito
- desdoblar los valores de la salida q' de su valor previo q
- introducir la noción de **valuación, estado** y **comportamiento**
- presentar informalmente la idea de **máquinas de estados**

Retroalimentación

Recordemos que había dos motivos por los cuales podíamos dar una visión funcional (que se describe completamente en base a sus valores de entrada y de salida) de los circuitos combinatorios:

1. **los tiempos de propagación** quedan fuera de nuestro modelo bajo el principio de abstracción
2. y no hay *retroalimentación de señales* en nuestros circuitos

Retroalimentación

Recordemos que había dos motivos por los cuales podíamos dar una visión funcional (que se describe completamente en base a sus valores de entrada y de salida) de los circuitos combinatorios:

1. **los tiempos de propagación** quedan fuera de nuestro modelo bajo el principio de abstracción
2. y no hay *retroalimentación de señales* en nuestros circuitos

Retroalimentación

Recordemos que había dos motivos por los cuales podíamos dar una visión funcional (que se describe completamente en base a sus valores de entrada y de salida) de los circuitos combinatorios:

1. **los tiempos de propagación** quedan fuera de nuestro modelo bajo el principio de abstracción
2. y no hay *retroalimentación de señales* en nuestros circuitos

Pero hasta ahora sólo observamos los problemas relacionados con la retrolimentación de señales.

Retroalimentación

Recordemos que había dos motivos por los cuales podíamos dar una visión funcional (que se describe completamente en base a sus valores de entrada y de salida) de los circuitos combinatorios:

1. **los tiempos de propagación** quedan fuera de nuestro modelo bajo el principio de abstracción
2. y no hay *retroalimentación de señales* en nuestros circuitos

Pero hasta ahora sólo observamos los problemas relacionados con la retrolimentación de señales.

¿Qué sucede cuando queremos razonar sobre circuitos secuenciales que se componen entre sí?

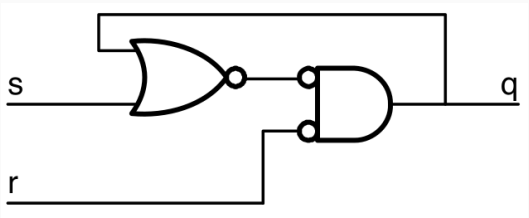
Circuitos secuenciales sincrónicos

Sincronización

El ejemplo presentado anteriormente es suficientemente simple como para poder evitar razonar sobre el orden de las señales.

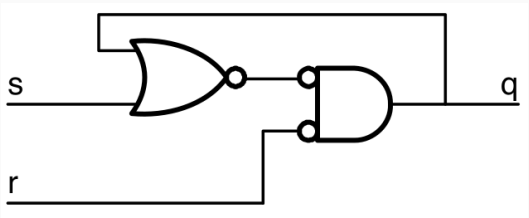
Sincronización

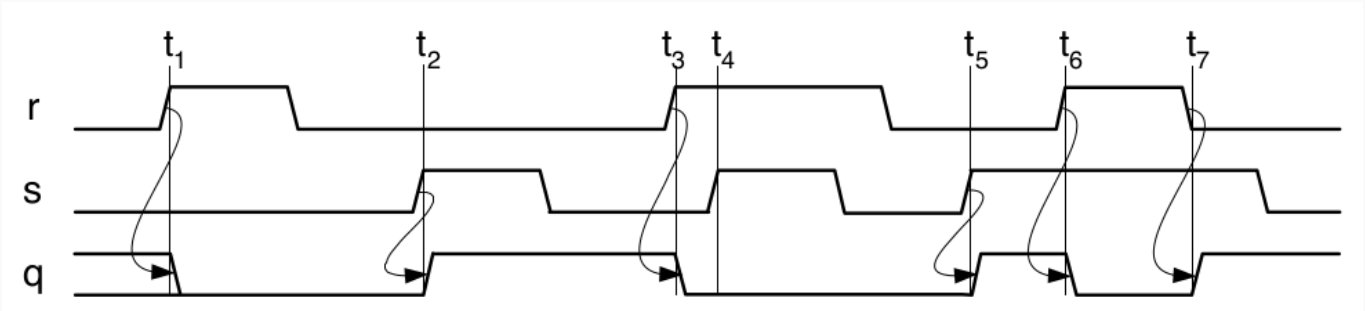
El ejemplo presentado anteriormente es suficientemente simple como para poder evitar razonar sobre el orden de las señales.



Sincronización

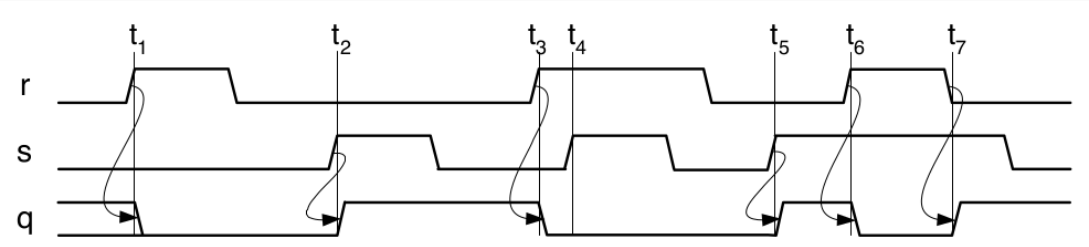
El ejemplo presentado anteriormente es suficientemente simple como para poder evitar razonar sobre el orden de las señales.





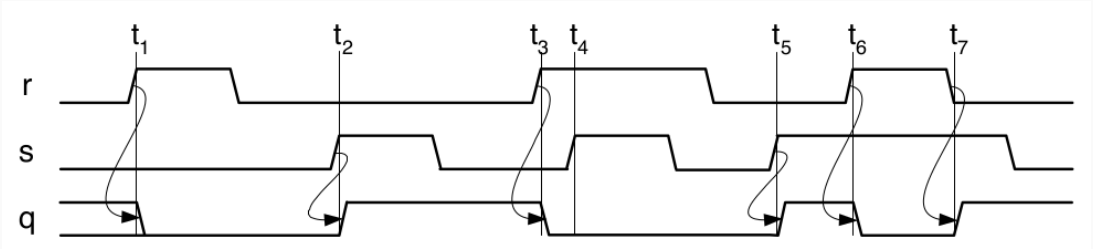
Sincronización

No hay garantías sobre el momento en el cuál las señales van a cambiar de valor.



Sincronización

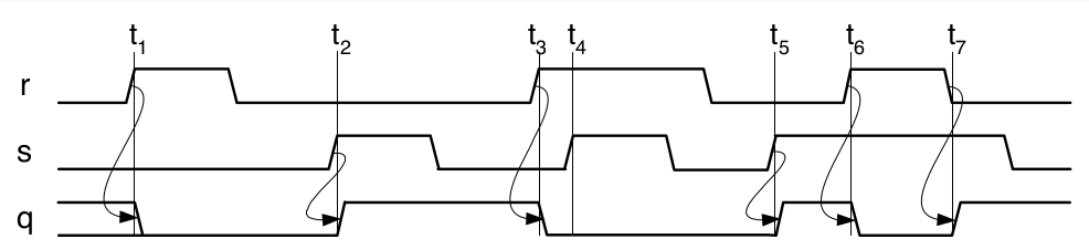
No hay garantías sobre el momento en el cuál las señales van a cambiar de valor.



Esto implica que no hay garantías sobre el momento en el cuál un componente cambia de estado (valuación).

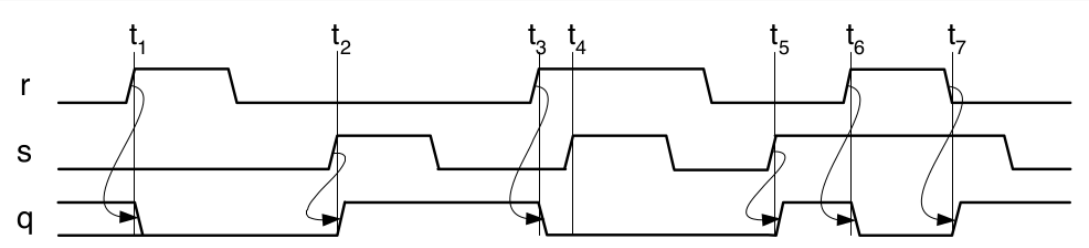
Sincronización

No hay garantías sobre el momento en el cuál las señales van a cambiar de valor.



Sincronización

No hay garantías sobre el momento en el cuál las señales van a cambiar de valor.



Necesitamos asegurar (idealmente) que el tiempo entre cambios de estado sea mayor que el peor tiempo de propagación de nuestros componentes.

Sincronización

Ya no podemos pensar que eventualmente las señales se estabilizan porque **deseamos que puedan cambiar a lo largo del tiempo**.

La retroalimentación así como la vimos introduce el siguiente problema:

- Tenemos que razonar sobre el orden y tiempo de propagación de las señales para poder saber cuándo una señal cambia de valor en la salida.
- Estos tiempos de propagación dependen de la cantidad de componentes y configuración interna de cada componente.

Sincronización

Ya no podemos pensar que eventualmente las señales se estabilizan porque **deseamos que puedan cambiar a lo largo del tiempo**.

La retroalimentación así como la vimos introduce el siguiente problema:

- Tenemos que razonar sobre el orden y tiempo de propagación de las señales para poder saber cuándo una señal cambia de valor en la salida.
- Estos tiempos de propagación dependen de la cantidad de componentes y configuración interna de cada componente.

Sincronización

Ya no podemos pensar que eventualmente las señales se estabilizan porque **deseamos que puedan cambiar a lo largo del tiempo**.

La retroalimentación así como la vimos introduce el siguiente problema:

- Tenemos que razonar sobre el orden y tiempo de propagación de las señales para poder saber cuándo una señal cambia de valor en la salida.
- Estos tiempos de propagación dependen de la cantidad de componentes y configuración interna de cada componente.

Sincronización

Ya no podemos pensar que eventualmente las señales se estabilizan porque **deseamos que puedan cambiar a lo largo del tiempo**.

La retroalimentación así como la vimos introduce el siguiente problema:

- Tenemos que razonar sobre el orden y tiempo de propagación de las señales para poder saber cuándo una señal cambia de valor en la salida.
- Estos tiempos de propagación dependen de la cantidad de componentes y configuración interna de cada componente.

Sincronización

Ya no podemos pensar que eventualmente las señales se estabilizan porque **deseamos que puedan cambiar a lo largo del tiempo**.

La retroalimentación así como la vimos introduce el siguiente problema:

- Se rompe el **principio de encapsulamiento** que nos permitía razonar sobre un componente sólo a través de su interfaz (entradas/salidas).
- ¿Por qué? Por que sin conocer su configuración **no podemos conocer su estado** (señales internas).

Sincronización

Ya no podemos pensar que eventualmente las señales se estabilizan porque **deseamos que puedan cambiar a lo largo del tiempo**.

La retroalimentación así como la vimos introduce el siguiente problema:

- Se rompe el **principio de encapsulamiento** que nos permitía razonar sobre un componente sólo a través de su interfaz (entradas/salidas).
- ¿Por qué? Por que sin conocer su configuración **no podemos conocer su estado** (señales internas).

Sincronización

Ya no podemos pensar que eventualmente las señales se estabilizan porque **deseamos que puedan cambiar a lo largo del tiempo**.

La retroalimentación así como la vimos introduce el siguiente problema:

- Se rompe el **principio de encapsulamiento** que nos permitía razonar sobre un componente sólo a través de su interfaz (entradas/salidas).
- ¿Por qué? Por que sin conocer su configuración **no podemos conocer su estado** (señales internas).

Sincronización

La necesidad que tenemos ahora es la de:

Sincronización

La necesidad que tenemos ahora es la de:

Diseñar un mecanismo que nos permita poder razonar con certeza sobre el estado de todos los componentes del sistema.

Sincronización

La propuesta es:

Sincronización

La propuesta es:

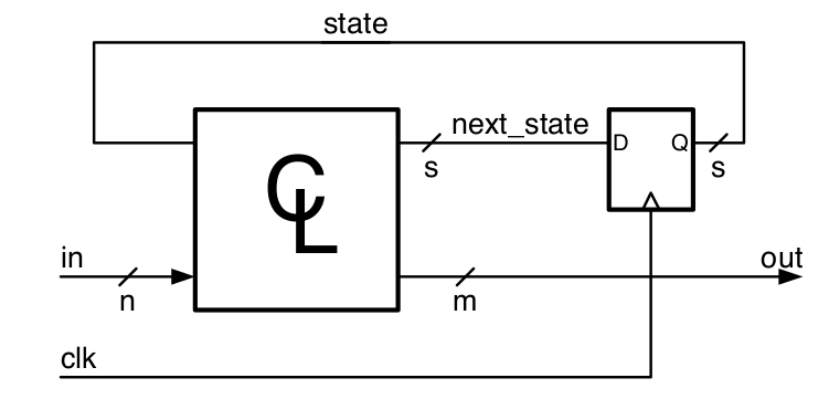
Introducir una señal especial (**clock**) que determina el intervalo de tiempo dentro del cuál los valores de salida (**siguiente estado**) se actualizan en base a los valores de entrada (**estado actual**).

Sincronización

Veamos un ejemplo:

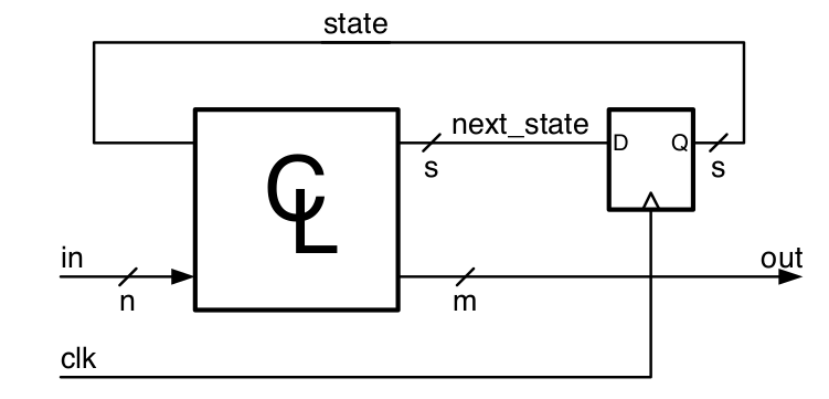
Sincronización

Veamos un ejemplo:



Sincronización

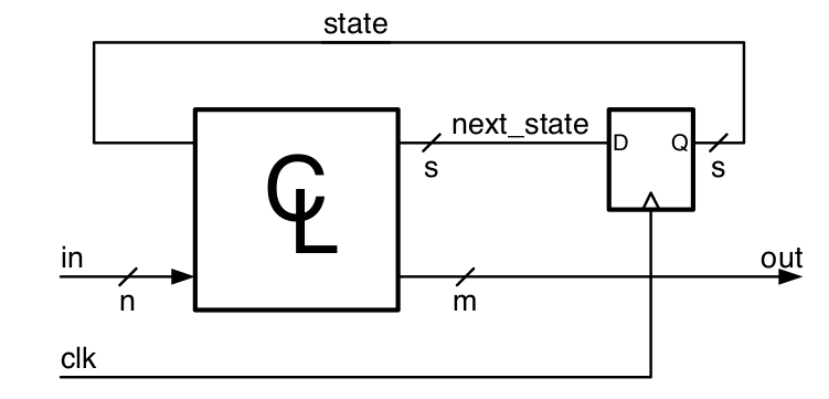
Veamos un ejemplo:



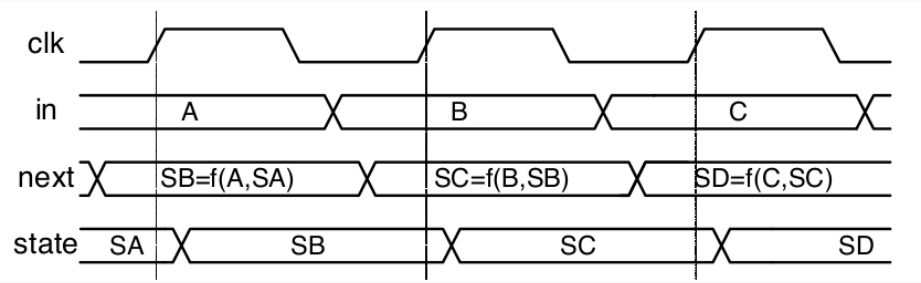
El clock determina en qué momento el estado siguiente pasa a ser el actual:

Sincronización

Veamos un ejemplo:



El clock determina en qué momento el estado siguiente pasa a ser el actual:



Sincronización

A partir de ahora las nociones de **estado** y de sincronización a través de una señal de **clock** van a ser elementos fundamentales de los diseños de nuestros circuitos secuenciales.

Sincronización

A partir de ahora las nociones de **estado** y de sincronización a través de una señal de **clock** van a ser elementos fundamentales de los diseños de nuestros circuitos secuenciales.

A continuación vamos a ver en detalle y en orden de complejidad creciente, **los elementos fundacionales que nos permiten construir y componer circuitos secuenciales sincrónicos**.

Sincronización

Estos circuitos son de la mayor relevancia para nuestor objetivo, que era diseñar un procesador, ya que precisamos que la información pueda transformarse a lo largo del tiempo.

Sincronización

Estos circuitos son de la mayor relevancia para nuestor objetivo, que era diseñar un procesador, ya que precisamos que la información pueda transformarse a lo largo del tiempo.

Para poder razonar y especificar el comportamiento de estos componentes **vamos a estudiar los formalismos de máquinas de estado más usados**, ya que nos permiten modelar el problema con un grado de abstracción suficientemente adecuado.

Cierre de la primera parte

Conclusión

Hasta ahora vimos:

- Circuitos combinatorios - Timing
- Retroalimentación y cambio de modelo
- Circuitos secuenciales asincrónicos (introducción)
- Circuitos secuenciales sincrónicos (introducción)

- Flip-flops, registros y memorias
- Máquinas de estado (en detalle)

Conclusión

Hasta ahora vimos:

- **Circuitos combinatorios - Timing**
- **Retroalimentación y cambio de modelo**
- **Circuitos secuenciales asincrónicos (introducción)**
- **Circuitos secuenciales sincrónicos (introducción)**

- **Flip-flops, registros y memorias**
- **Máquinas de estado (en detalle)**

Conclusión

Hasta ahora vimos:

- **Circuitos combinatorios - Timing**
- **Retroalimentación y cambio de modelo**
- **Circuitos secuenciales asincrónicos (introducción)**
- **Circuitos secuenciales sincrónicos (introducción)**

- **Flip-flops, registros y memorias**
- **Máquinas de estado (en detalle)**

Conclusión

Hasta ahora vimos:

- **Circuitos combinatorios - Timing**
- **Retroalimentación y cambio de modelo**
- **Circuitos secuenciales asincrónicos (introducción)**
- **Circuitos secuenciales sincrónicos (introducción)**
- **Flip-flops, registros y memorias**
- **Máquinas de estado (en detalle)**

Conclusión

Hasta ahora vimos:

- **Circuitos combinatorios - Timing**
- **Retroalimentación y cambio de modelo**
- **Circuitos secuenciales asincrónicos (introducción)**
- **Circuitos secuenciales sincrónicos (introducción)**

- **Flip-flops, registros y memorias**
- **Máquinas de estado (en detalle)**

Conclusión

Hasta ahora vimos:

- **Circuitos combinatorios - Timing**
- **Retroalimentación y cambio de modelo**
- **Circuitos secuenciales asincrónicos (introducción)**
- **Circuitos secuenciales sincrónicos (introducción)**

Falta ver:

- **Flip-flops, registros y memorias**
- **Máquinas de estado (en detalle)**

Conclusión

Hasta ahora vimos:

- **Circuitos combinatorios - Timing**
- **Retroalimentación y cambio de modelo**
- **Circuitos secuenciales asincrónicos (introducción)**
- **Circuitos secuenciales sincrónicos (introducción)**

Falta ver:

- **Flip-flops, registros y memorias**
- **Máquinas de estado (en detalle)**

Conclusión

Hasta ahora vimos:

- **Circuitos combinatorios - Timing**
- **Retroalimentación y cambio de modelo**
- **Circuitos secuenciales asincrónicos (introducción)**
- **Circuitos secuenciales sincrónicos (introducción)**

Falta ver:

- **Flip-flops, registros y memorias**
- **Máquinas de estado (en detalle)**

Conclusión

Hasta ahora vimos:

- **Circuitos combinatorios - Timing**
- **Retroalimentación y cambio de modelo**
- **Circuitos secuenciales asincrónicos (introducción)**
- **Circuitos secuenciales sincrónicos (introducción)**

Falta ver:

- **Flip-flops, registros y memorias**
- **Máquinas de estado (en detalle)**

¿Preguntas?