Tema 6:

Implementación de sistemas secuenciales síncronos

Fundamentos de computadores I

José Manuel Mendías Cuadros
Dpto. Arquitectura de Computadores y Automática
Universidad Complutense de Madrid

Contenidos

- ✓ Biestable SR.
- ✓ Biestable D.
- ✓ Síntesis con biestables D.
- ✓ Inicialización de sistemas secuenciales.

- ✓ Apéndice tecnológico

Transparencias basadas en los libros:

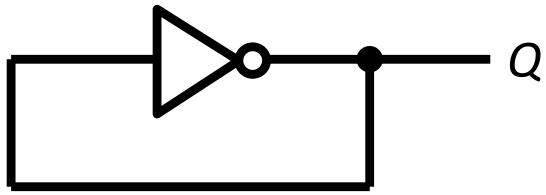
- R. Hermida, F. Sánchez y E. del Corral. *Fundamentos de computadores.*
- D. Gajsky. *Principios de diseño digital.*

Biestable

- Dispositivo capaz de almacenar físicamente un bit de información (tener 2 estados estables).

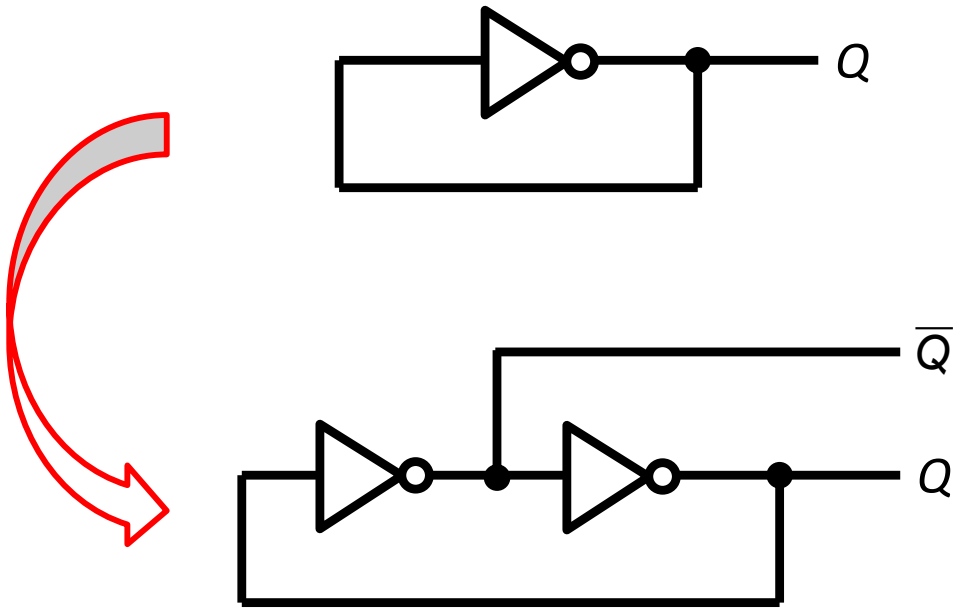
Biestable

- Dispositivo capaz de almacenar físicamente un bit de información (tener 2 estados estables).
  - mediante un circuito combinacional realimentado



Biestable

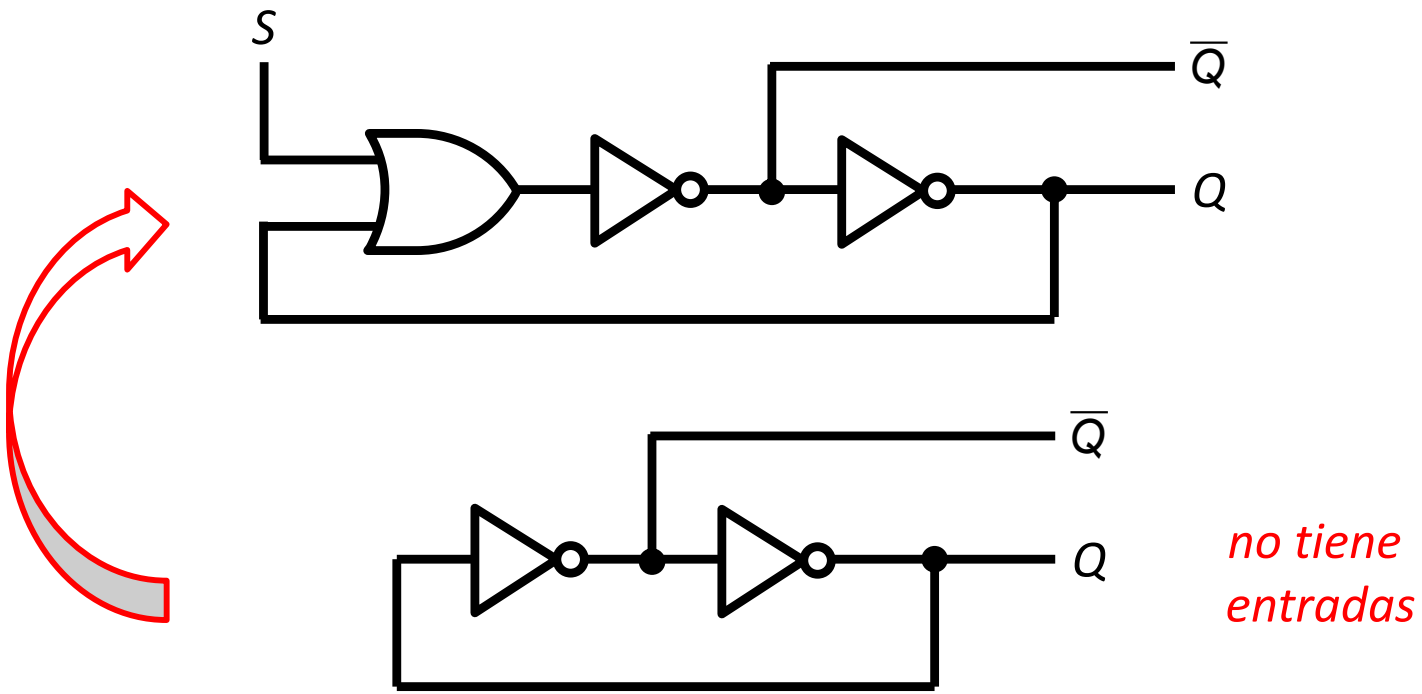
- Dispositivo capaz de almacenar físicamente un bit de información (tener 2 estados estables).
  - mediante un circuito combinacional realimentado



oscila

Biestable

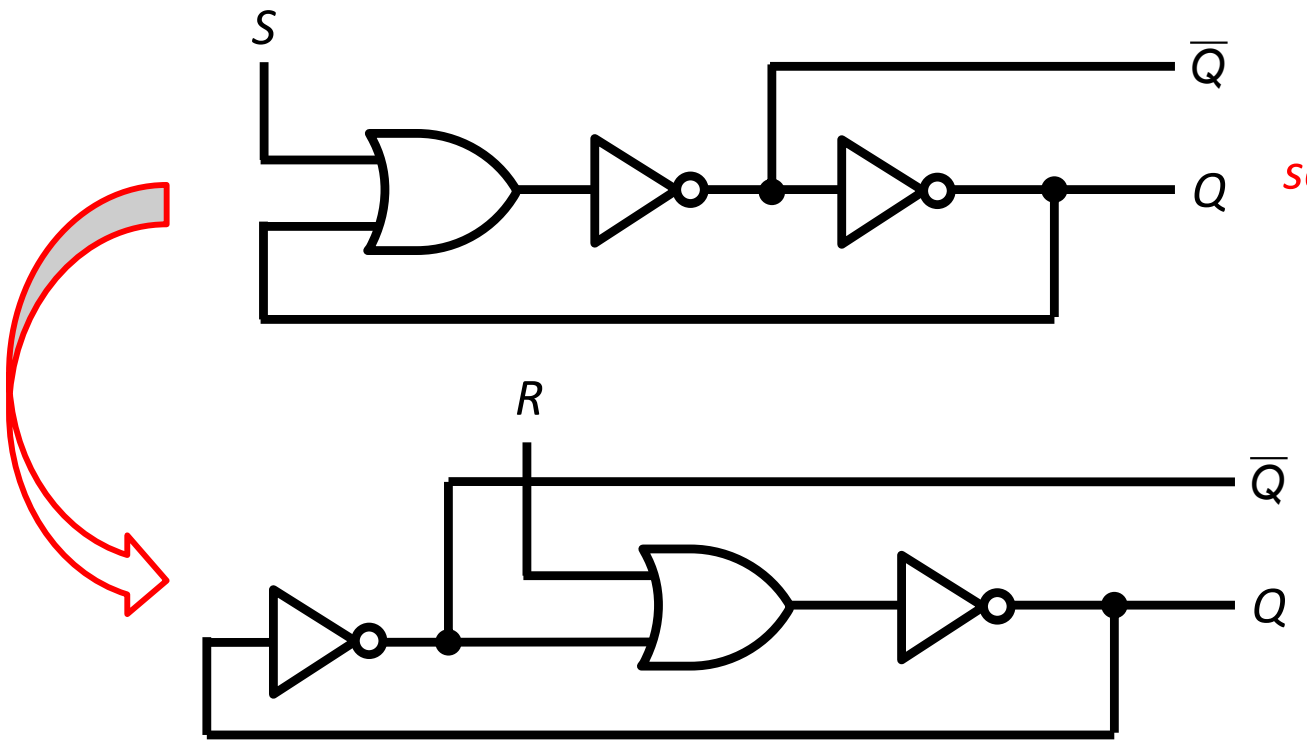
- Dispositivo capaz de almacenar físicamente un bit de información (tener 2 estados estables).
  - mediante un circuito combinacional realimentado



no tiene entradas

Biestable

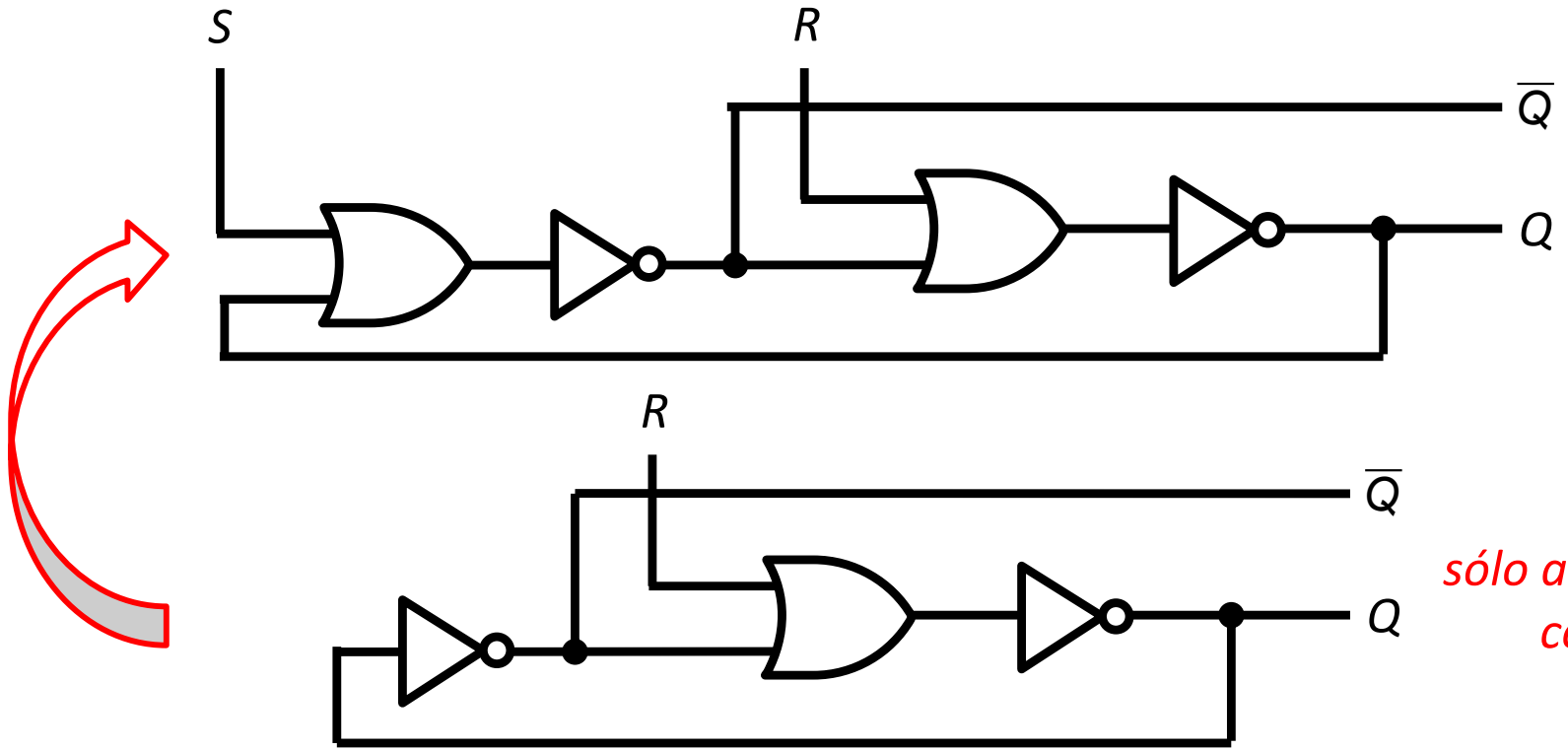
- Dispositivo capaz de almacenar físicamente un bit de información (tener 2 estados estables).
  - mediante un circuito combinacional realimentado



sólo almacena unos

Biestable

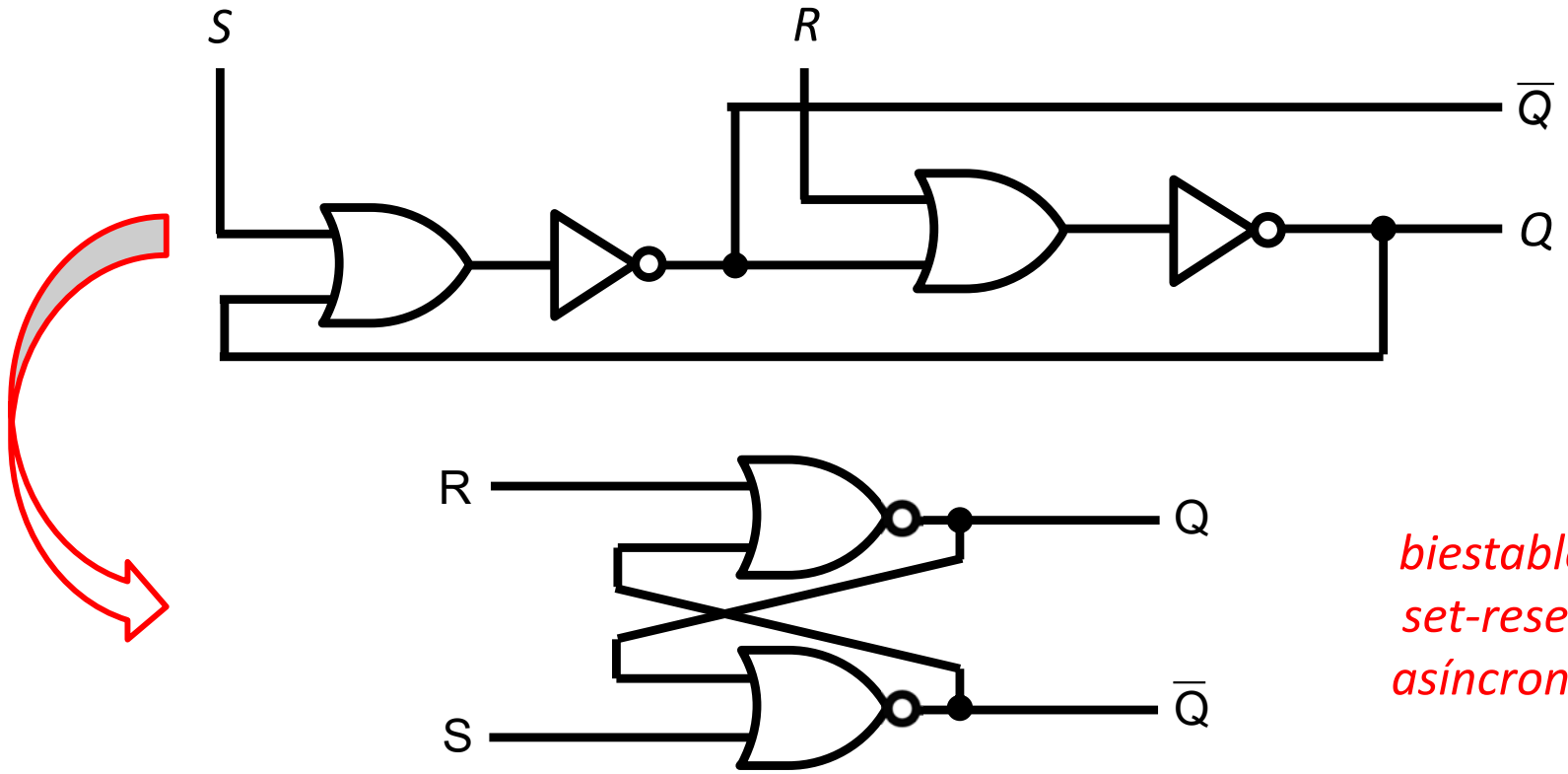
- Dispositivo capaz de almacenar físicamente un bit de información (tener 2 estados estables).
  - mediante un circuito combinacional realimentado



sólo almacena ceros

Biestable

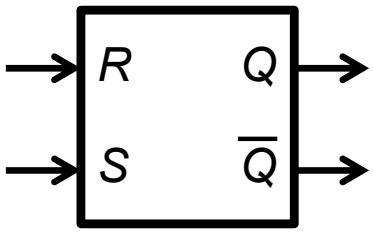
- Dispositivo capaz de almacenar físicamente un bit de información (tener 2 estados estables).
  - mediante un circuito combinacional realimentado

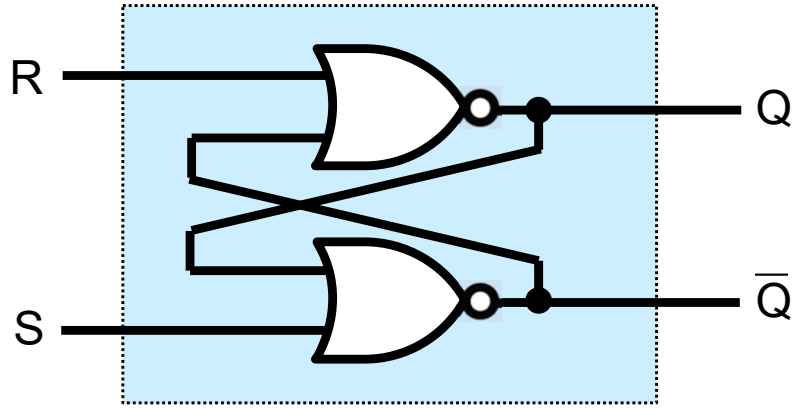


biestable set-reset asíncrono

Biestable

- Dispositivo capaz de almacenar físicamente un bit de información (tener 2 estados estables).
  - mediante un circuito combinacional realimentado

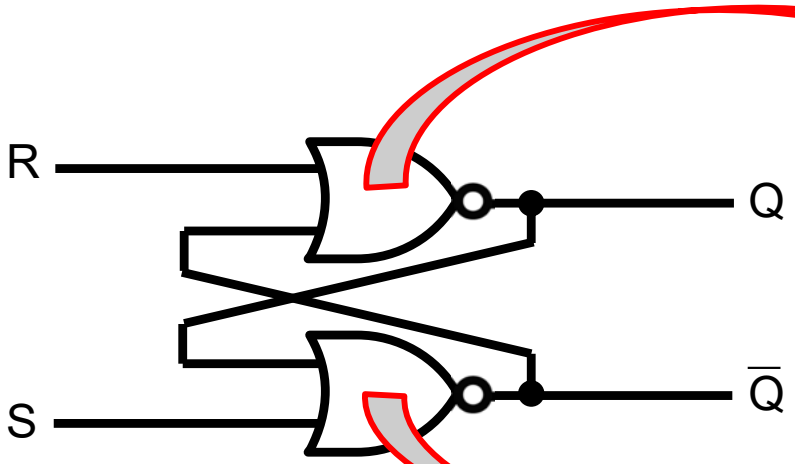




Biestable SR asíncrono

R(t)	S(t)	Q(t+Δt)	
0	0	Q(t)	*conserva valor*
0	1	1	*almacena un 1*
1	0	0	*almacena un 0*
1	1	prohibido	*entradas contradictorias*

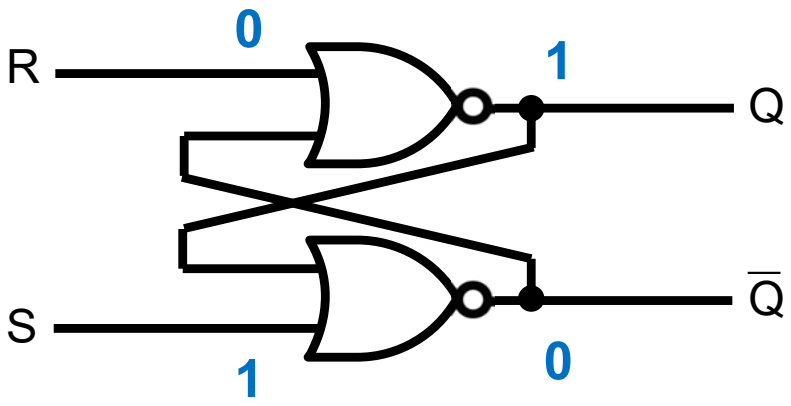
Biestable SR asíncrono



R	$\overline{Q}$	Q
0	0	1
0	1	0
1	0	0
1	1	0

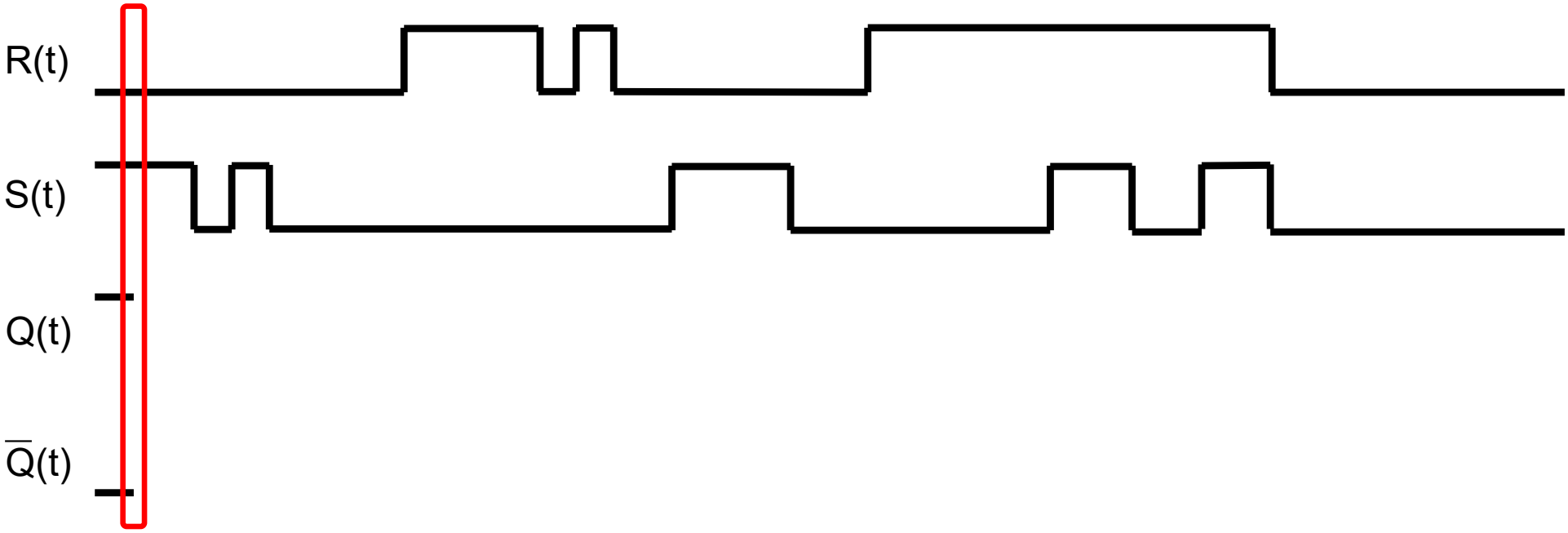
S	Q	$\overline{Q}$
0	0	1
0	1	0
1	0	0
1	1	0

Biestable SR asíncrono

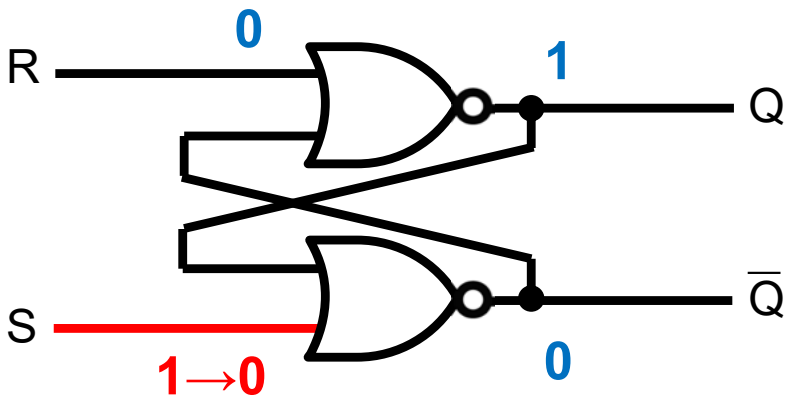


R	$\overline{Q}$	Q
0	0	1
0	1	0
1	0	0
1	1	0

S	Q	$\overline{Q}$
0	0	1
0	1	0
1	0	0
1	1	0

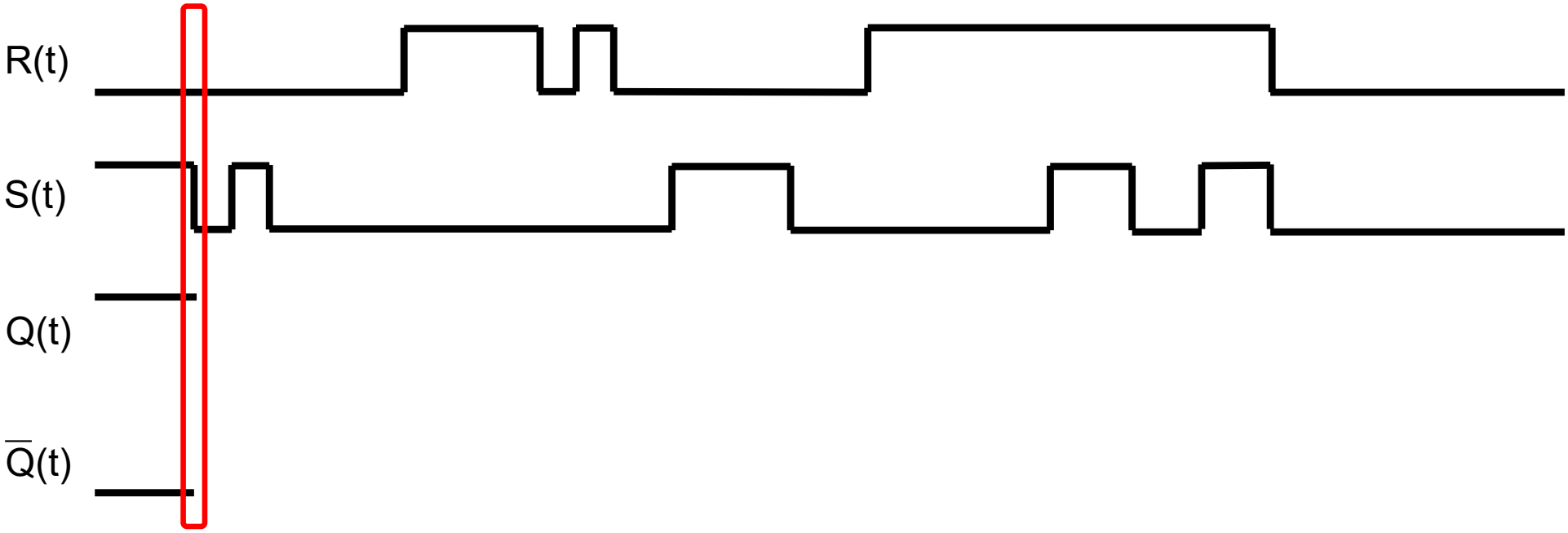


Biestable SR asíncrono

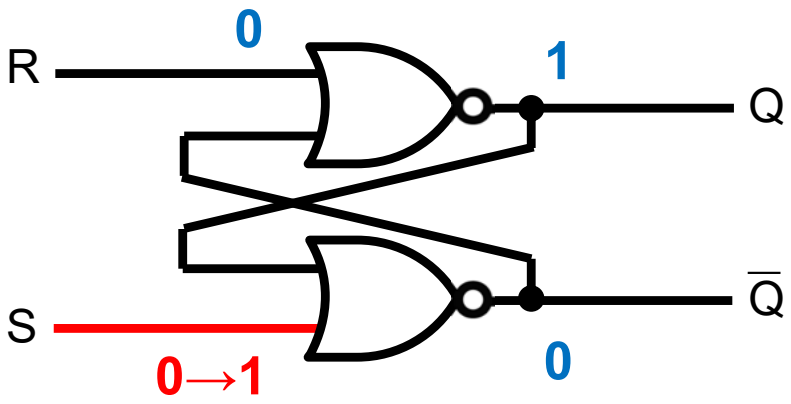


R	$\overline{Q}$	Q
0	0	1
0	1	0
1	0	0
1	1	0

S	Q	$\overline{Q}$
0	0	1
0	1	0
1	0	0
1	1	0

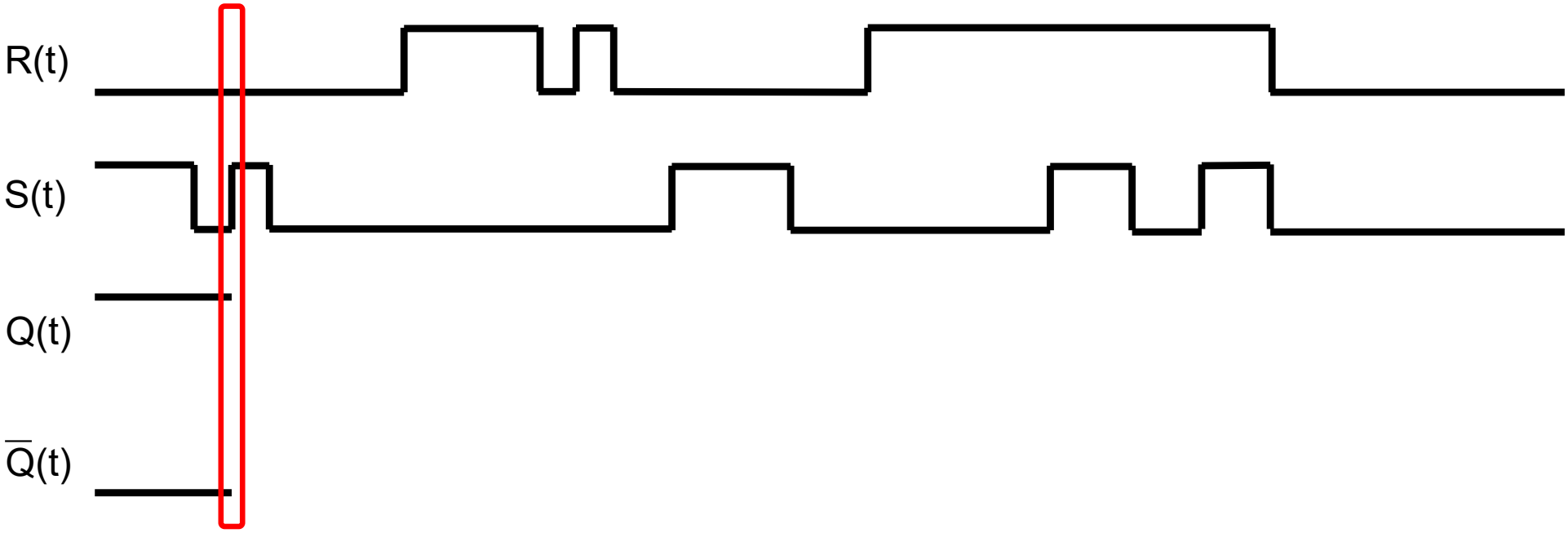


Biestable SR asíncrono

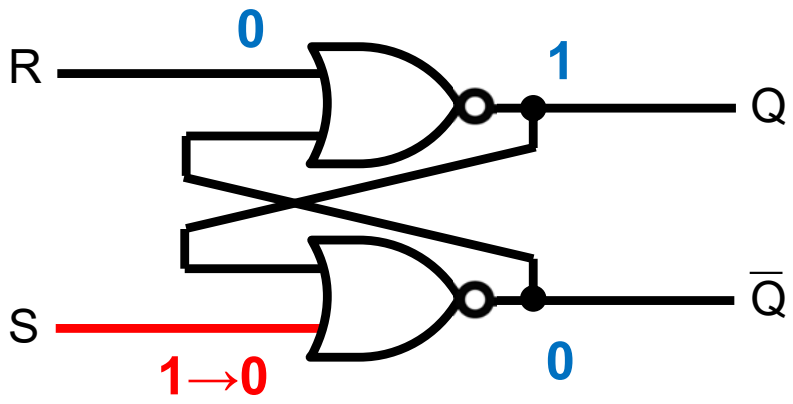


R	$\overline{Q}$	Q
0	0	1
0	1	0
1	0	0
1	1	0

S	Q	$\overline{Q}$
0	0	1
0	1	0
1	0	0
1	1	0

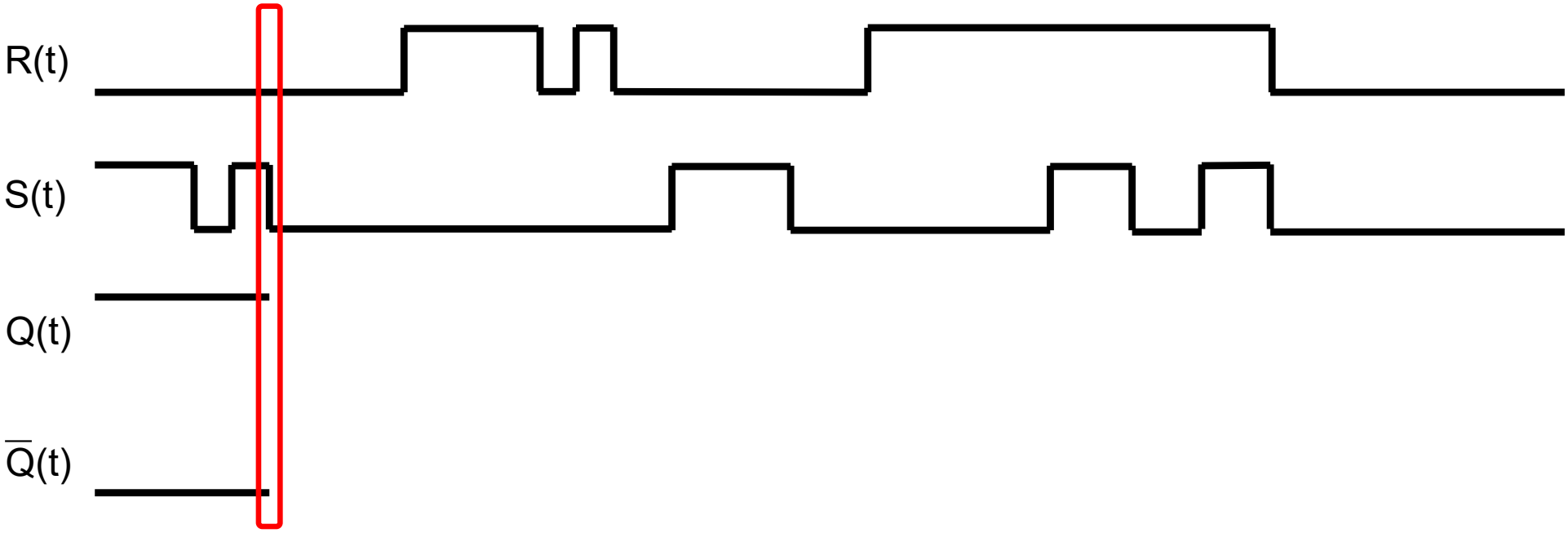


Biestable SR asíncrono

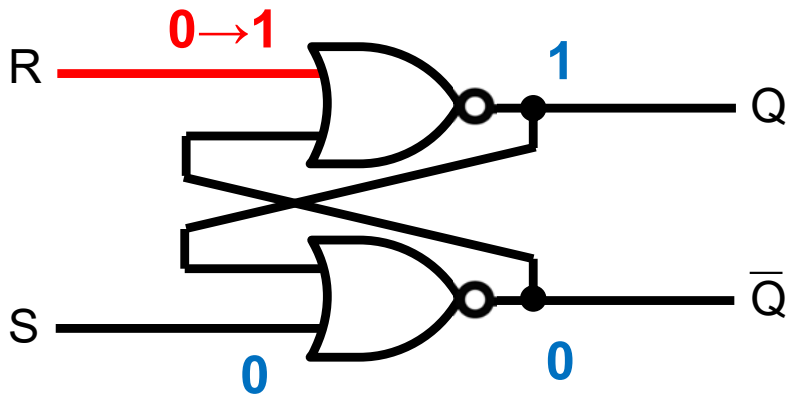


R	$\overline{Q}$	Q
0	0	1
0	1	0
1	0	0
1	1	0

S	Q	$\overline{Q}$
0	0	1
0	1	0
1	0	0
1	1	0

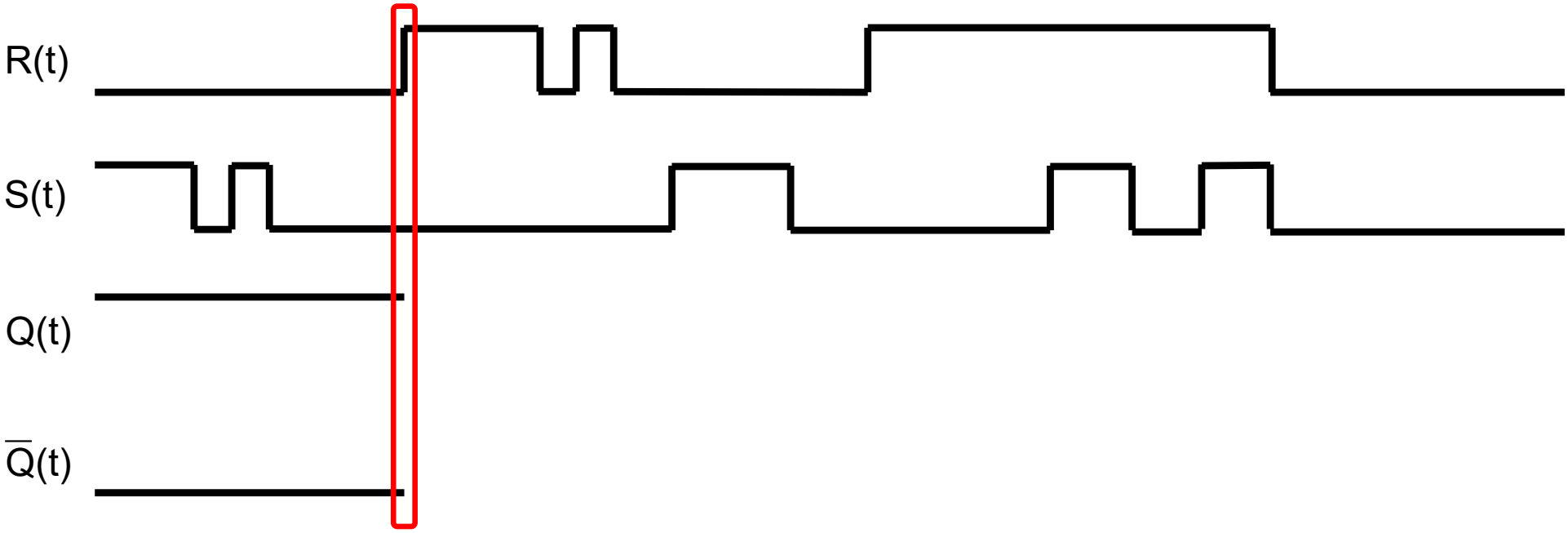


Biestable SR asíncrono

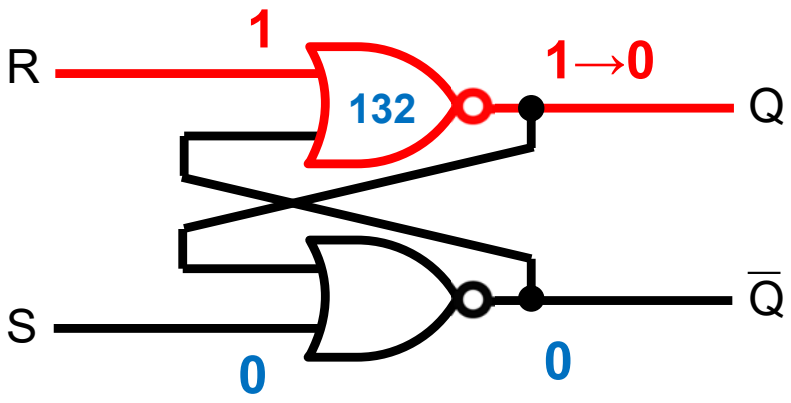


R	$\overline{Q}$	Q
0	0	1
0	1	0
1	0	0
1	1	0

S	Q	$\overline{Q}$
0	0	1
0	1	0
1	0	0
1	1	0

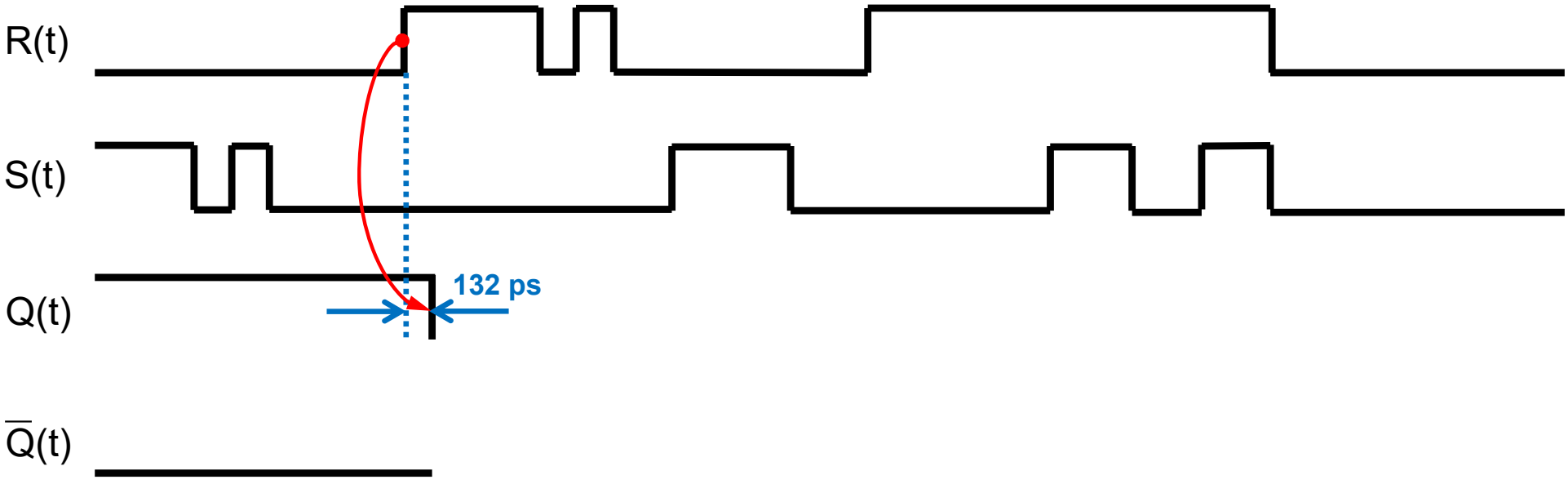


Biestable SR asíncrono

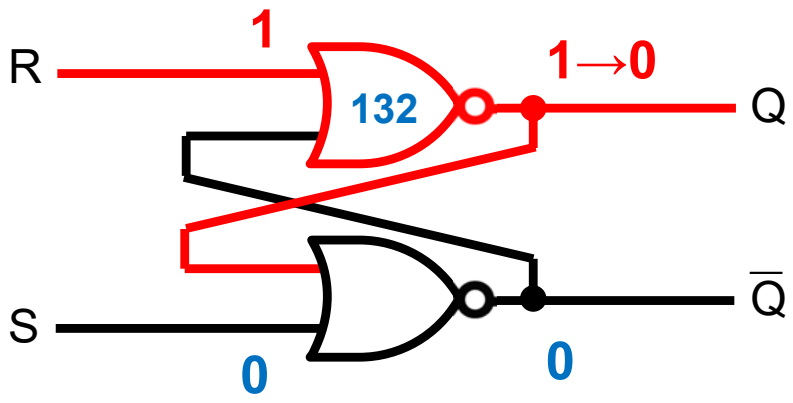


R	$\overline{Q}$	Q
0	0	1
0	1	0
1	0	0
1	1	0

S	Q	$\overline{Q}$
0	0	1
0	1	0
1	0	0
1	1	0

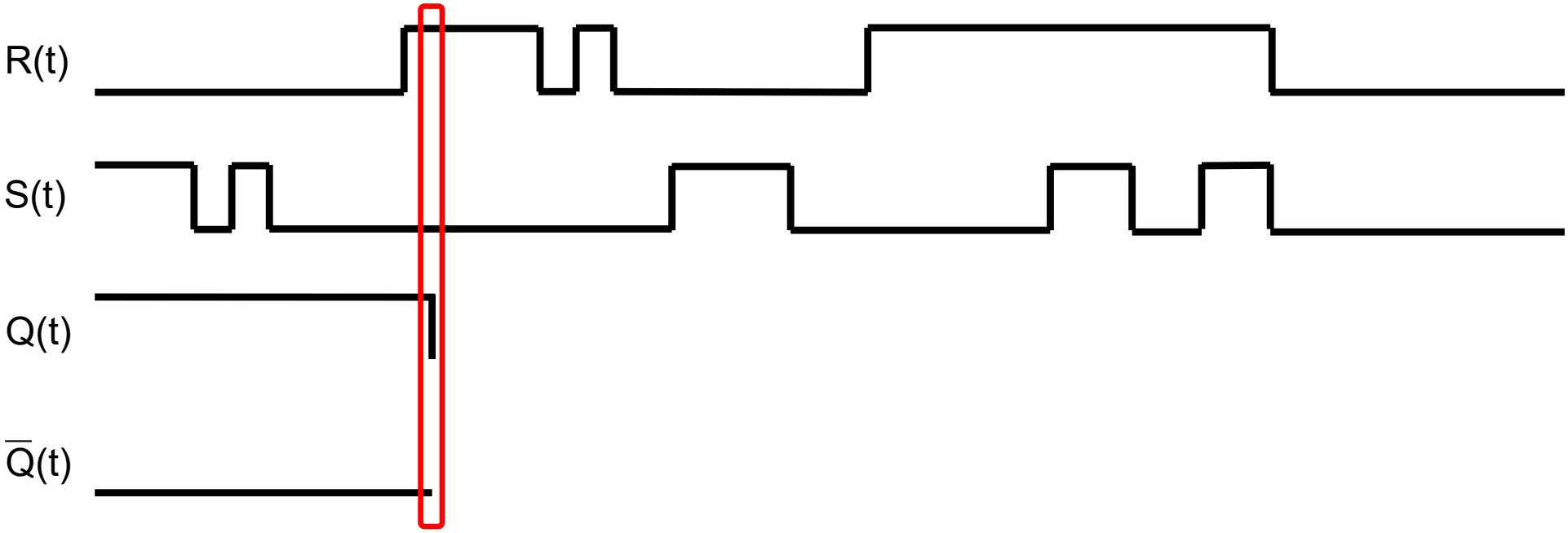


Biestable SR asíncrono

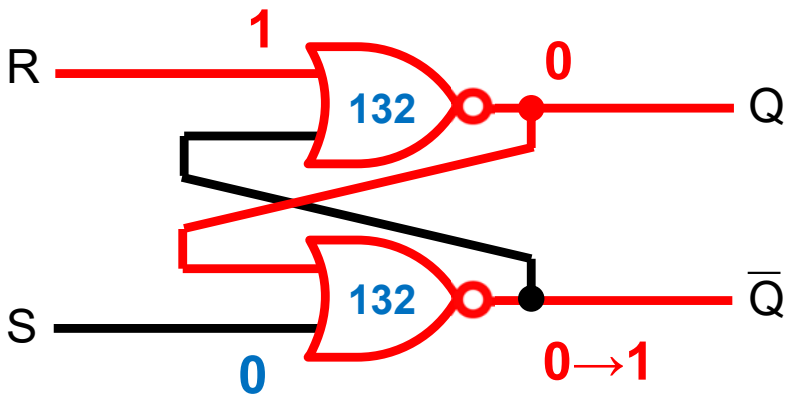


R	$\overline{Q}$	Q
0	0	1
0	1	0
1	0	0
1	1	0

S	Q	$\overline{Q}$
0	0	1
0	1	0
1	0	0
1	1	0

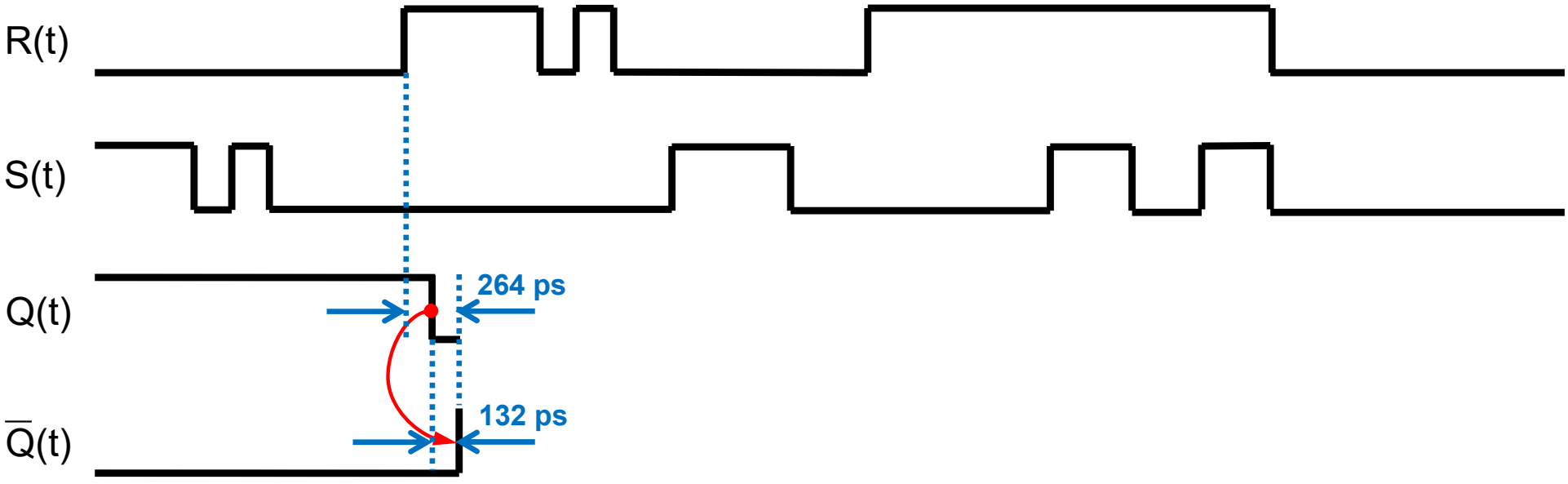


Biestable SR asíncrono

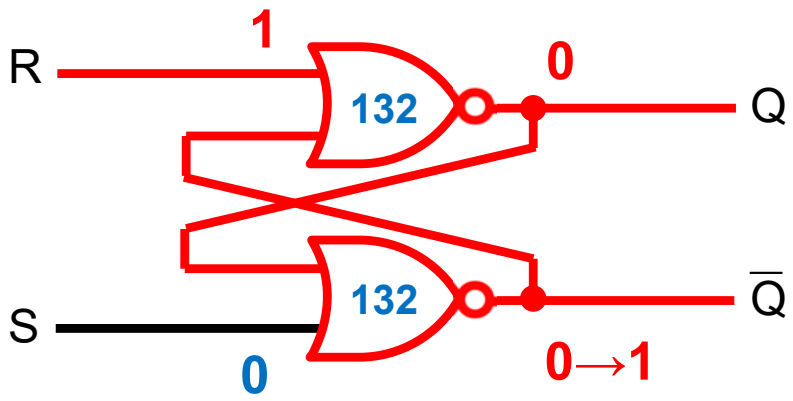


R	$\overline{Q}$	Q
0	0	1
0	1	0
1	0	0
1	1	0

S	Q	$\overline{Q}$
0	0	1
0	1	0
1	0	0
1	1	0

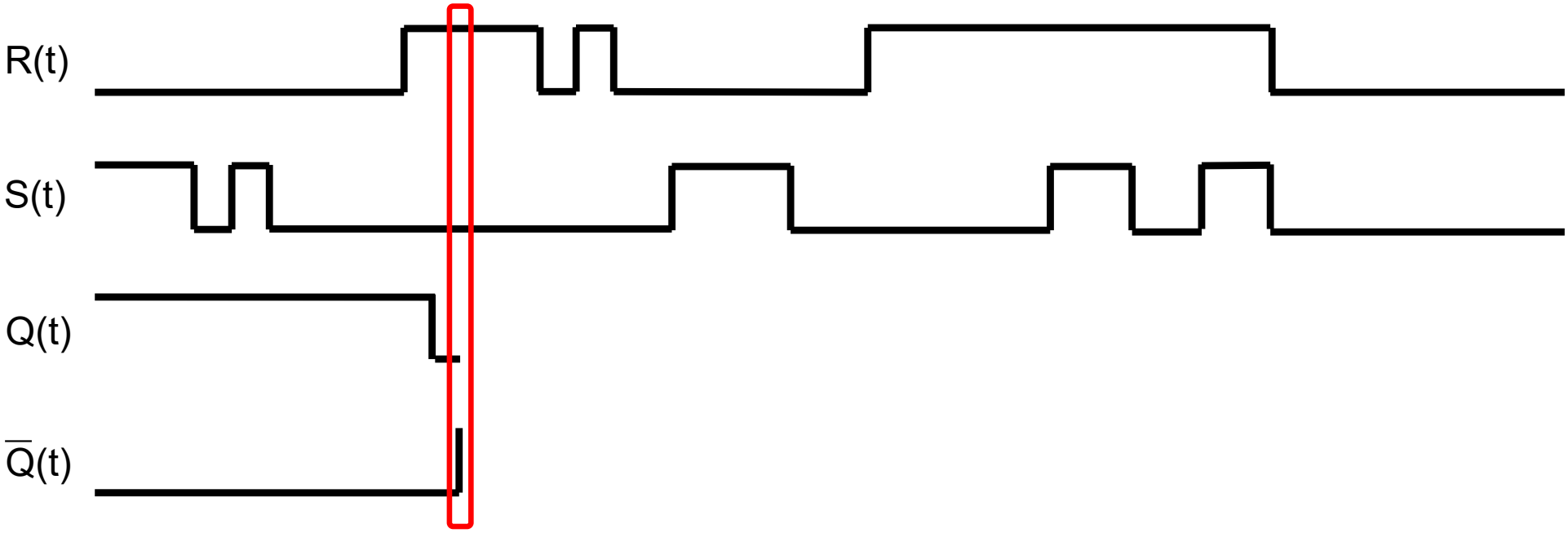


Biestable SR asíncrono

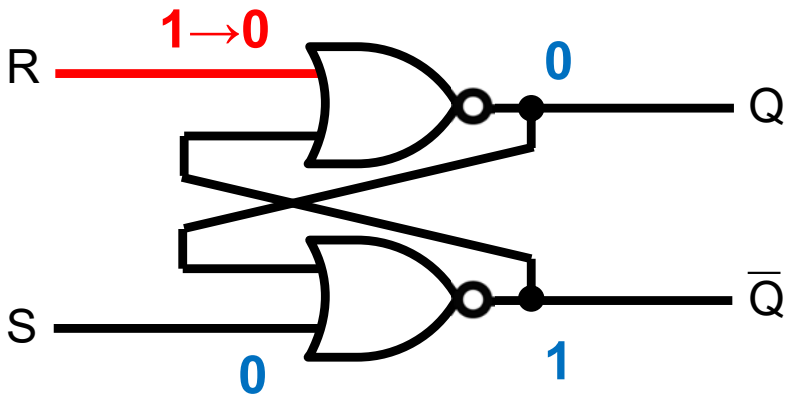


R	$\overline{Q}$	Q
0	0	1
0	1	0
1	0	0
1	1	0

S	Q	$\overline{Q}$
0	0	1
0	1	0
1	0	0
1	1	0

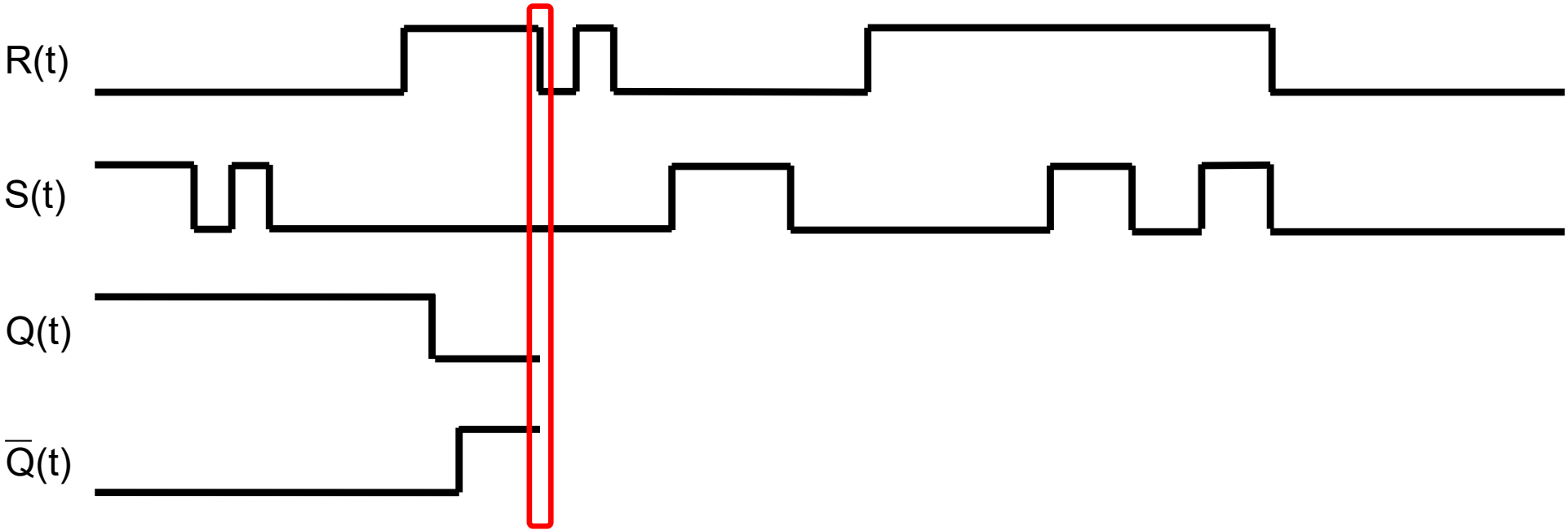


Biestable SR asíncrono

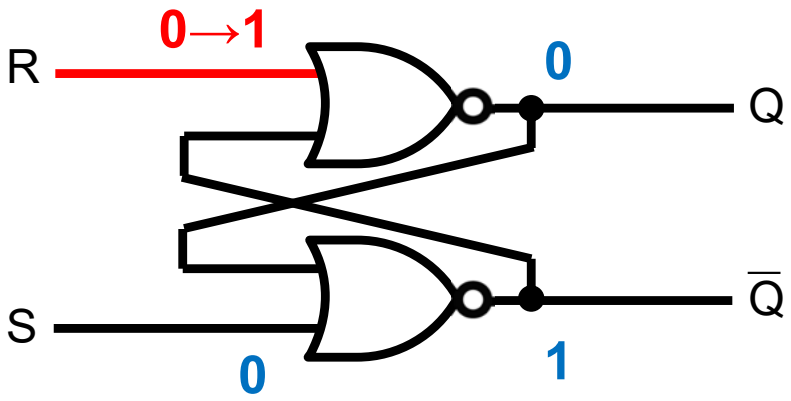


R	$\overline{Q}$	Q
0	0	1
0	1	0
1	0	0
1	1	0

S	Q	$\overline{Q}$
0	0	1
0	1	0
1	0	0
1	1	0

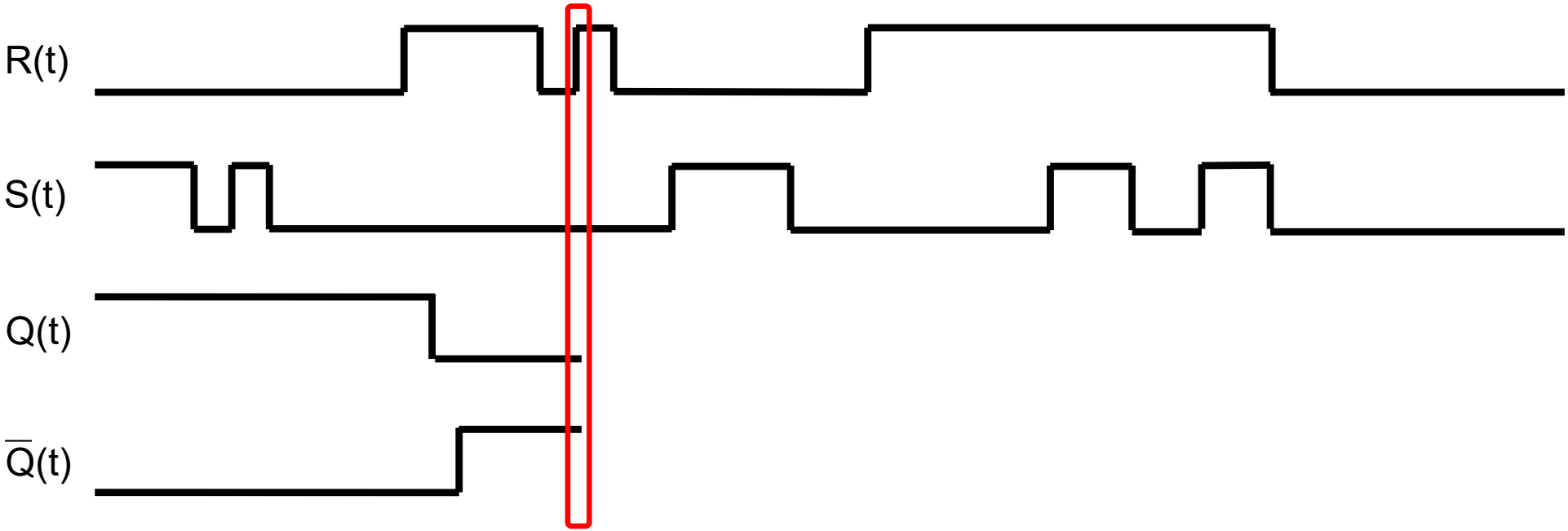


Biestable SR asíncrono

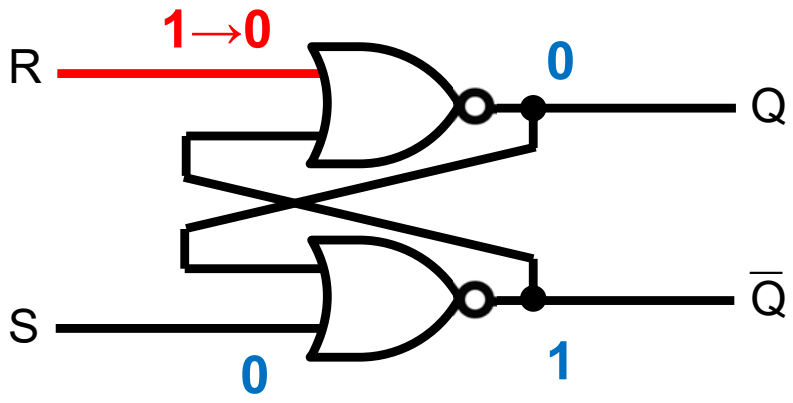


R	$\overline{Q}$	Q
0	0	1
0	1	0
1	0	0
1	1	0

S	Q	$\overline{Q}$
0	0	1
0	1	0
1	0	0
1	1	0

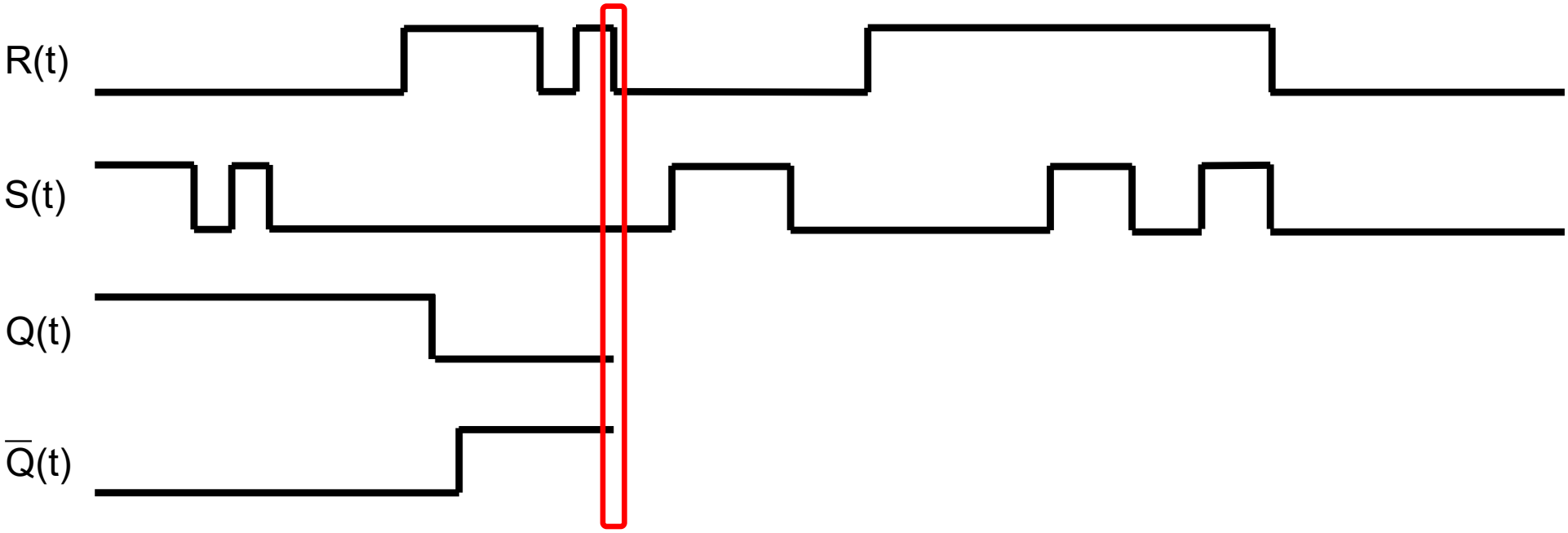


Biestable SR asíncrono

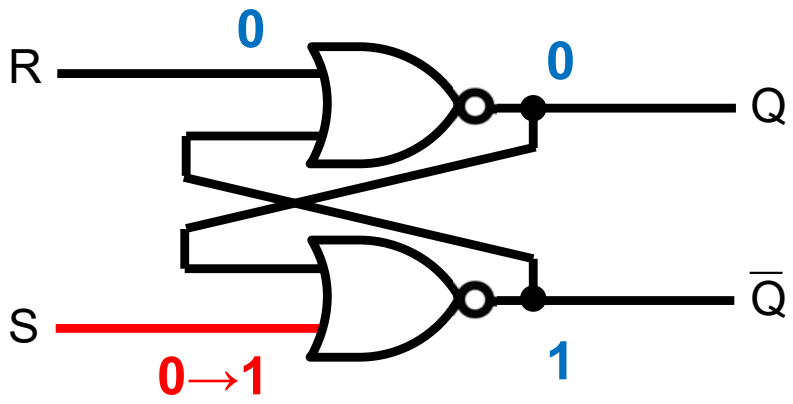


R	$\overline{Q}$	Q
0	0	1
0	1	0
1	0	0
1	1	0

S	Q	$\overline{Q}$
0	0	1
0	1	0
1	0	0
1	1	0

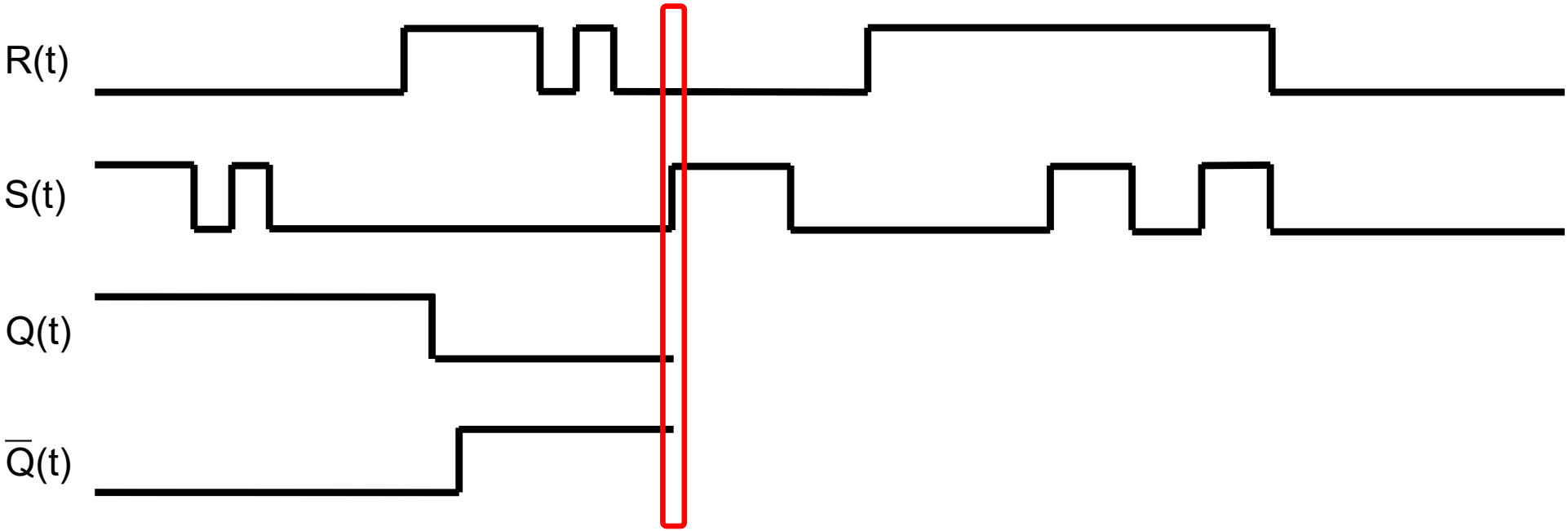


Biestable SR asíncrono

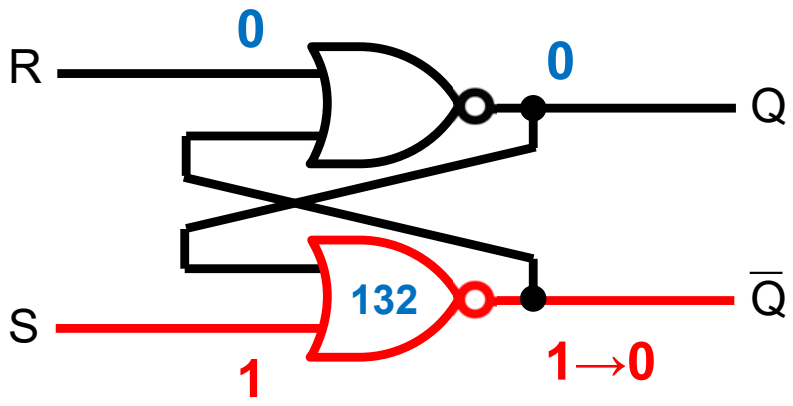


R	$\overline{Q}$	Q
0	0	1
0	1	0
1	0	0
1	1	0

S	Q	$\overline{Q}$
0	0	1
0	1	0
1	0	0
1	1	0

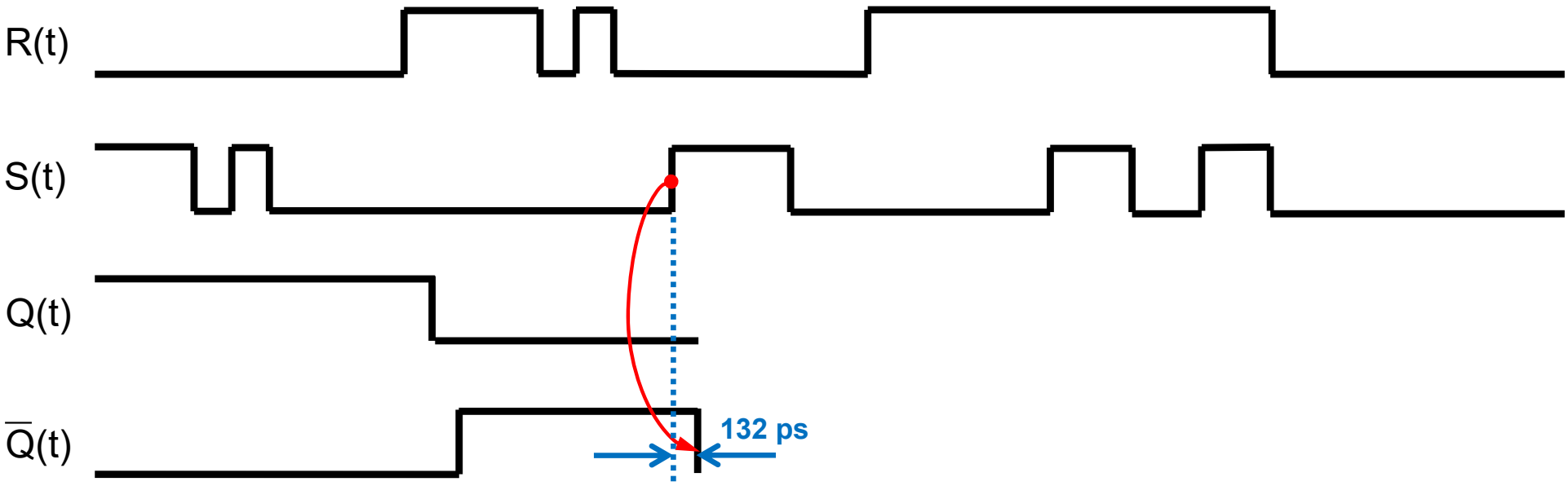


Biestable SR asíncrono

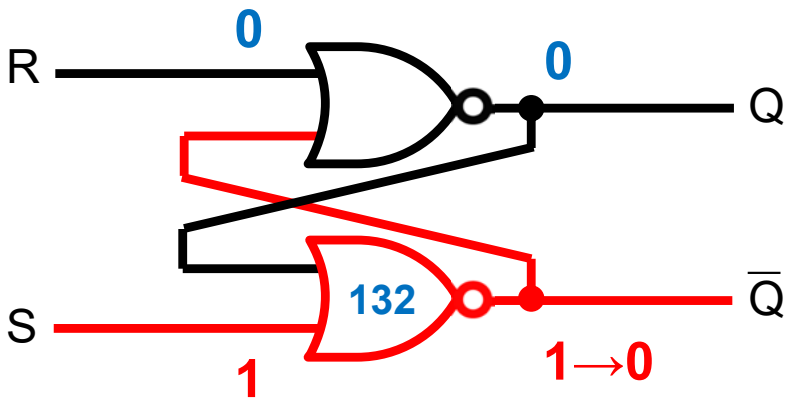


R	$\overline{Q}$	Q
0	0	1
0	1	0
1	0	0
1	1	0

S	Q	$\overline{Q}$
0	0	1
0	1	0
1	0	0
1	1	0

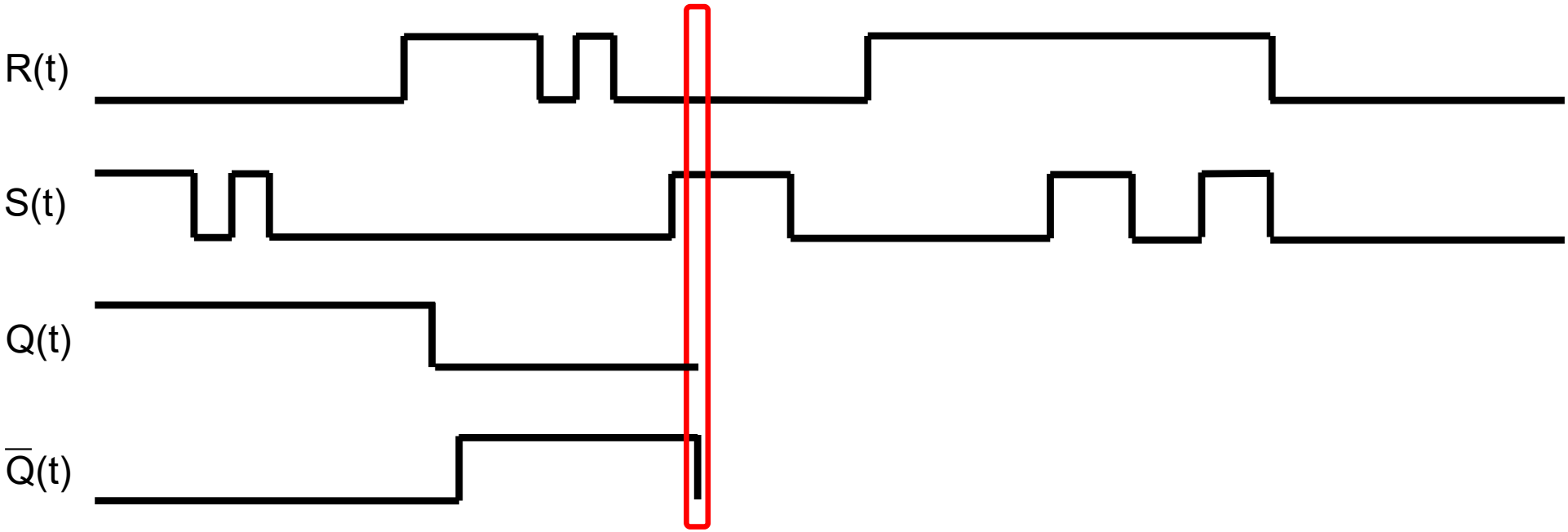


Biestable SR asíncrono

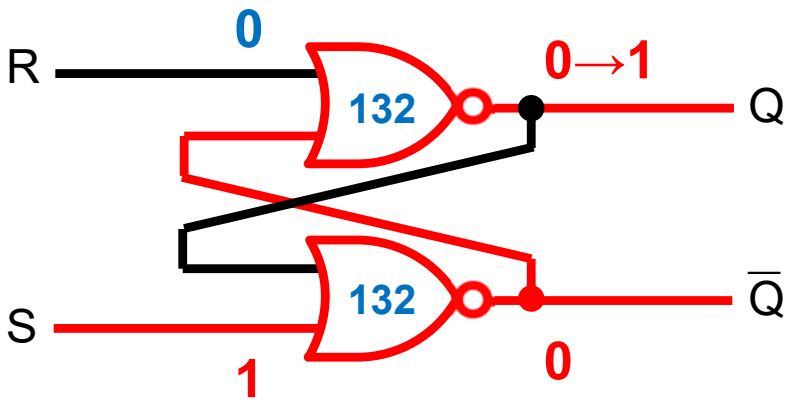


R	$\overline{Q}$	Q
0	0	1
0	1	0
1	0	0
1	1	0

S	Q	$\overline{Q}$
0	0	1
0	1	0
1	0	0
1	1	0

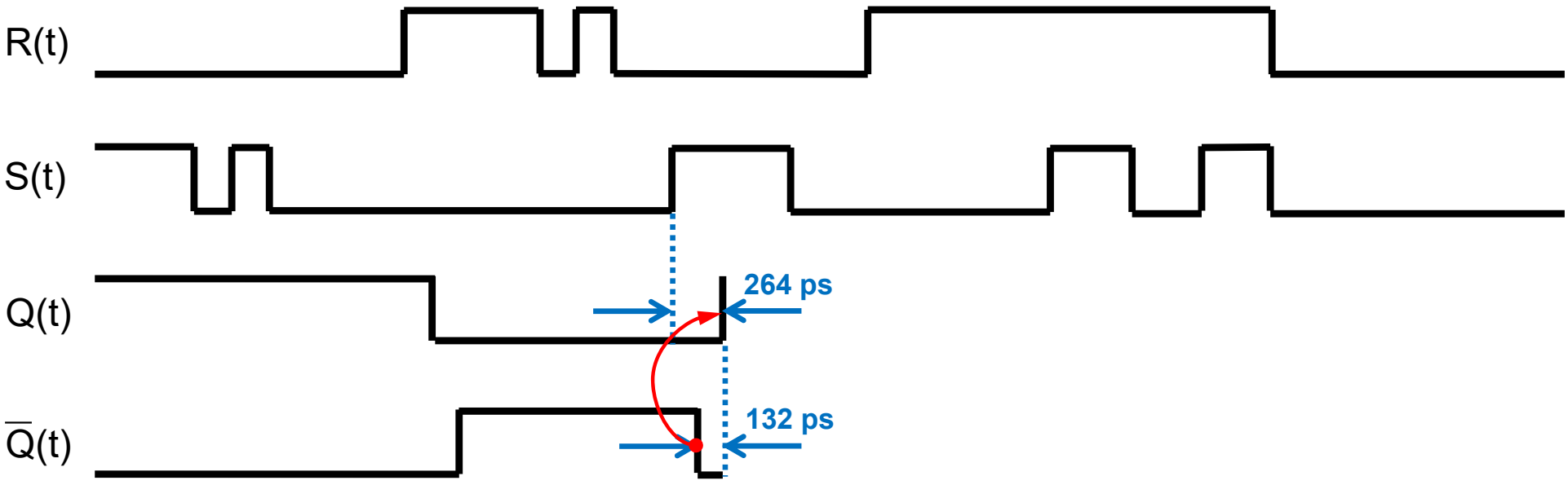


Biestable SR asíncrono

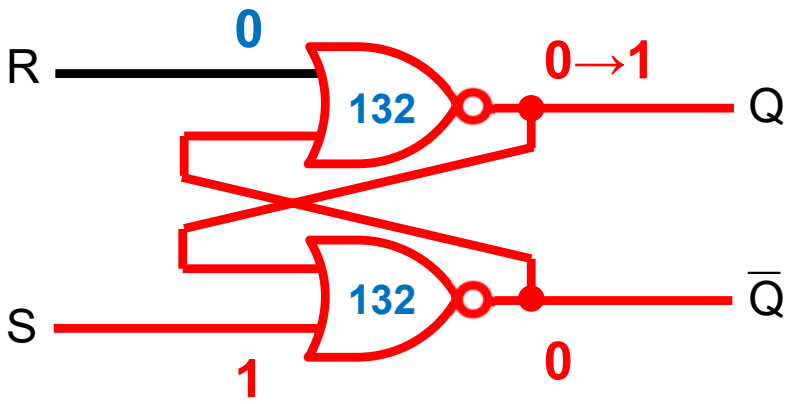


R	$\overline{Q}$	Q
0	0	1
0	1	0
1	0	0
1	1	0

S	Q	$\overline{Q}$
0	0	1
0	1	0
1	0	0
1	1	0

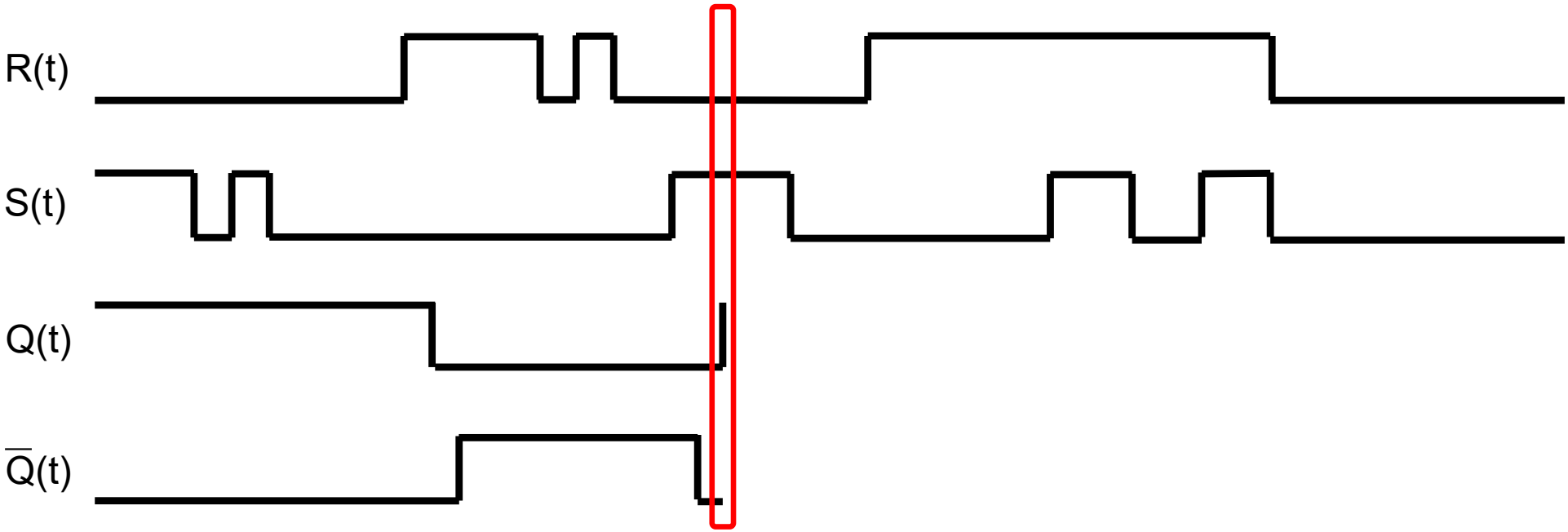


Biestable SR asíncrono

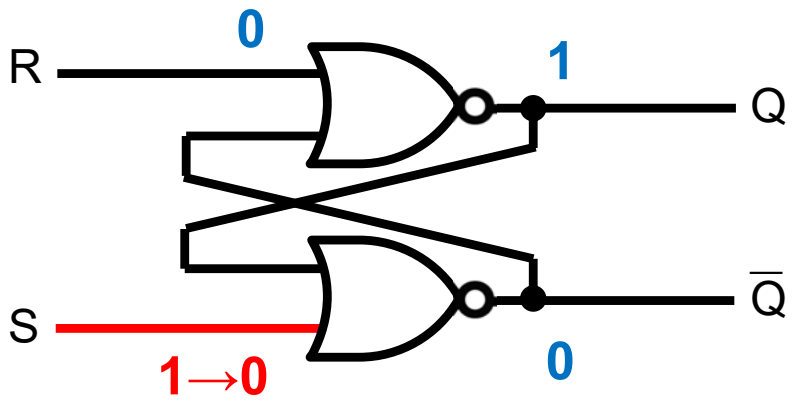


R	$\overline{Q}$	Q
0	0	1
0	1	0
1	0	0
1	1	0

S	Q	$\overline{Q}$
0	0	1
0	1	0
1	0	0
1	1	0

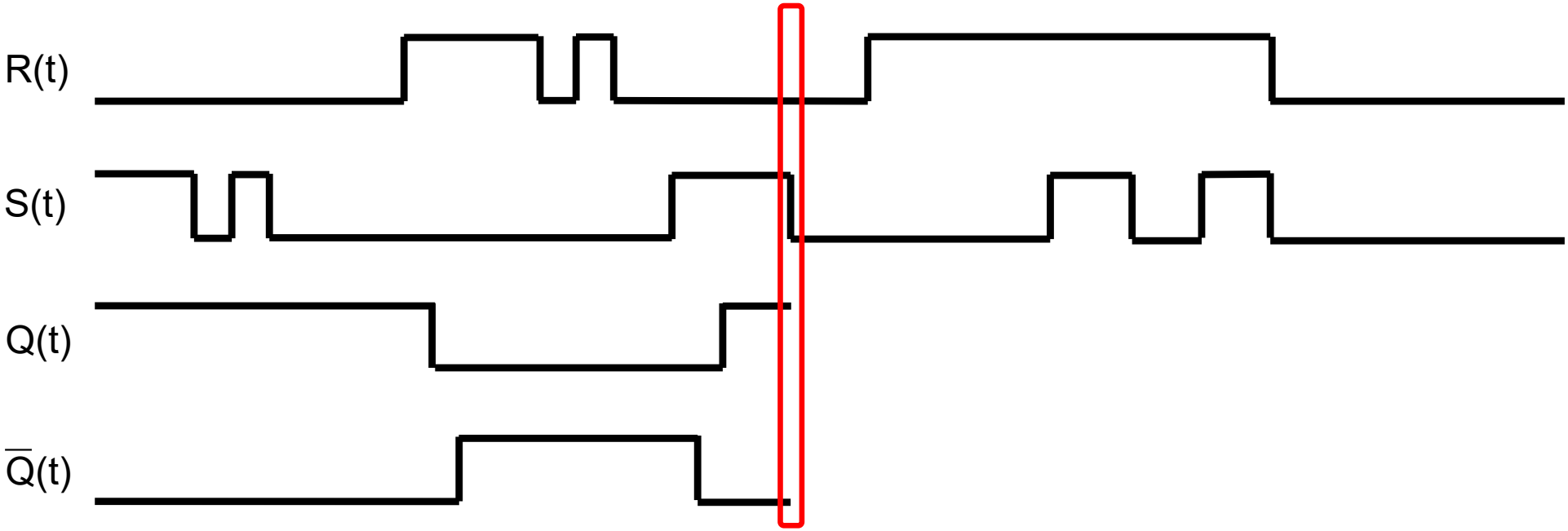


Biestable SR asíncrono

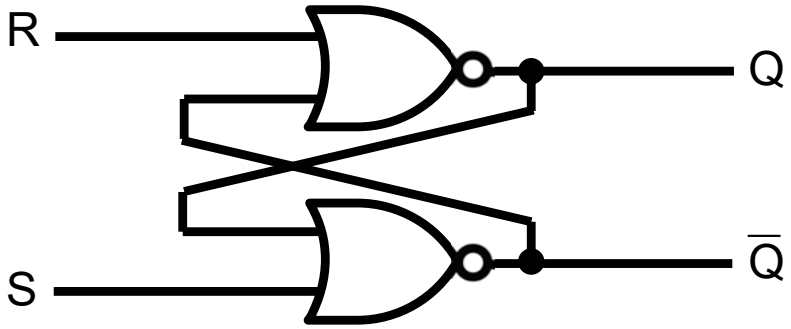


R	$\overline{Q}$	Q
0	0	1
0	1	0
1	0	0
1	1	0

S	Q	$\overline{Q}$
0	0	1
0	1	0
1	0	0
1	1	0

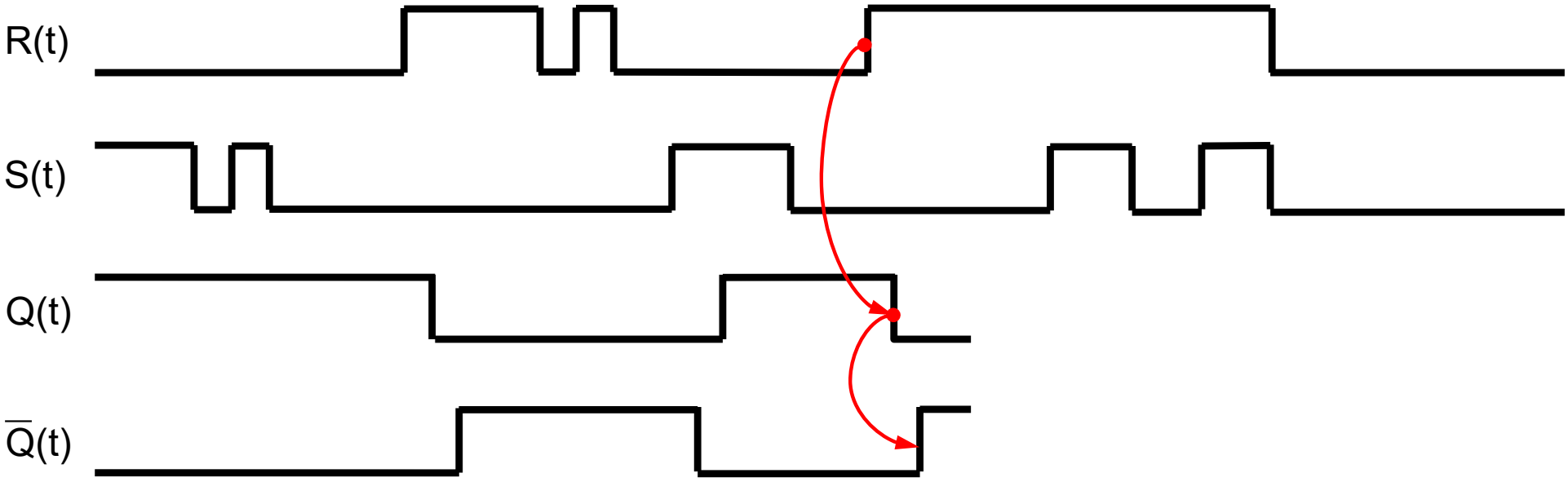


Biestable SR asíncrono

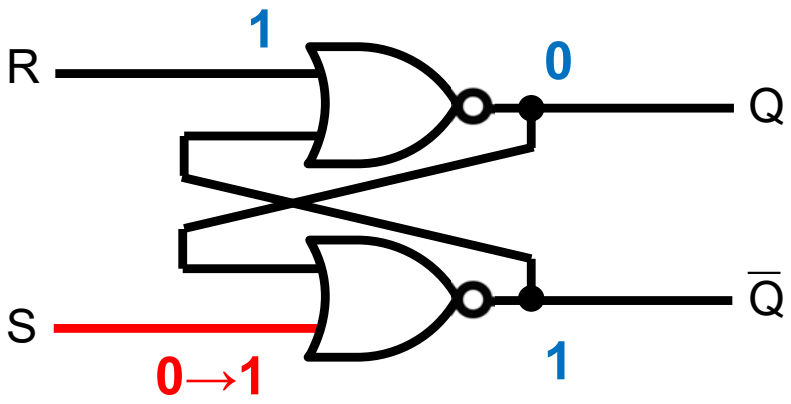


R	$\overline{Q}$	Q
0	0	1
0	1	0
1	0	0
1	1	0

S	Q	$\overline{Q}$
0	0	1
0	1	0
1	0	0
1	1	0

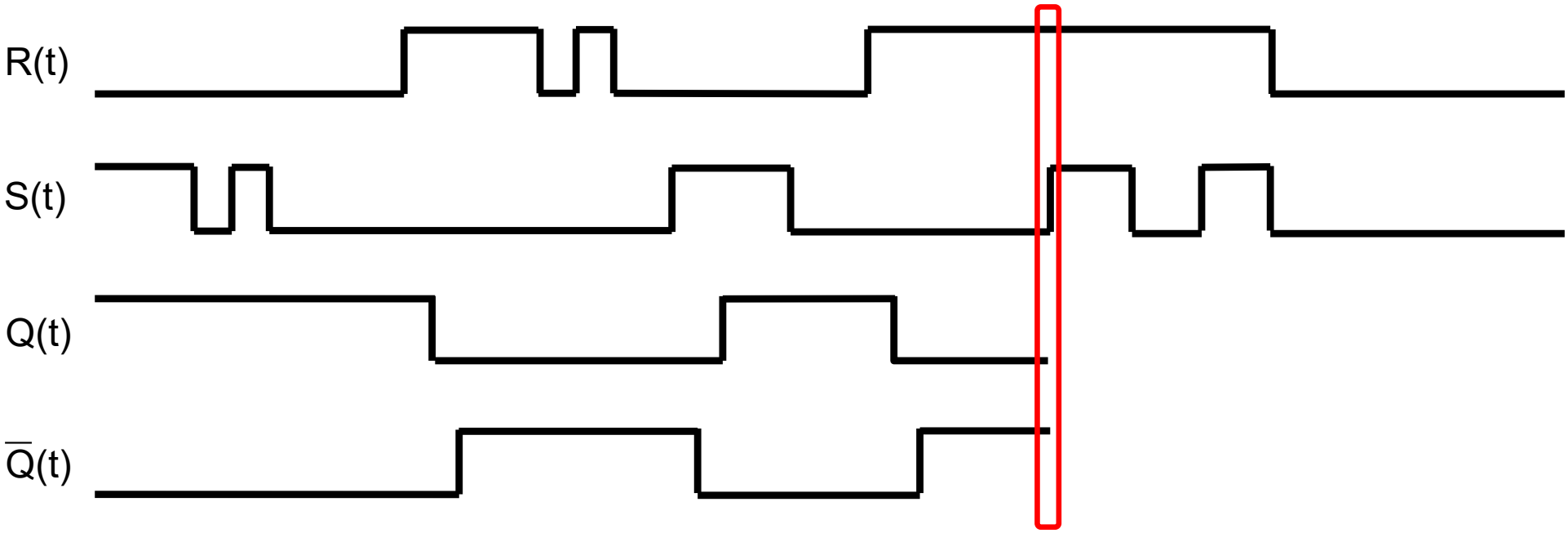


Biestable SR asíncrono

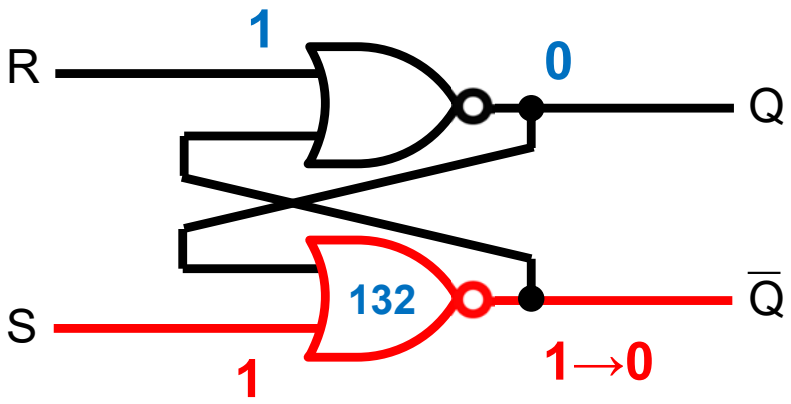


R	$\overline{Q}$	Q
0	0	1
0	1	0
1	0	0
1	1	0

S	Q	$\overline{Q}$
0	0	1
0	1	0
1	0	0
1	1	0

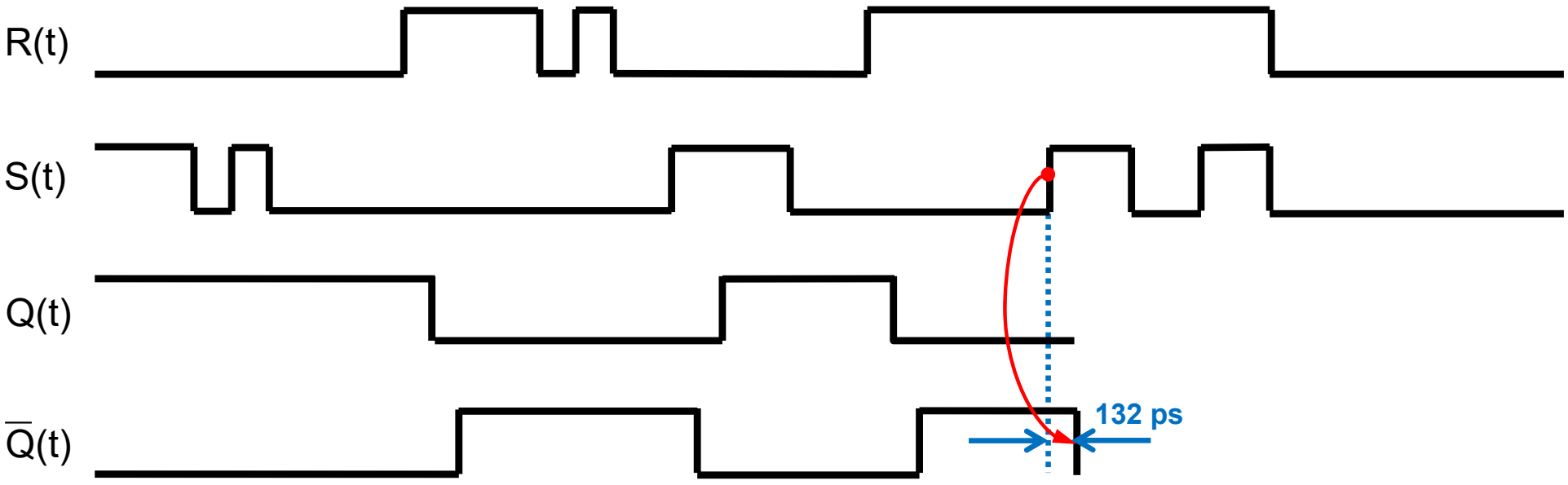


Biestable SR asíncrono

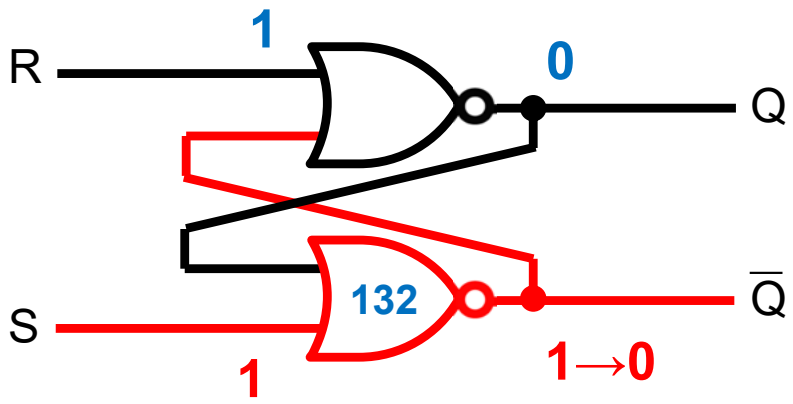


R	$\overline{Q}$	Q
0	0	1
0	1	0
1	0	0
1	1	0

S	Q	$\overline{Q}$
0	0	1
0	1	0
1	0	0
1	1	0

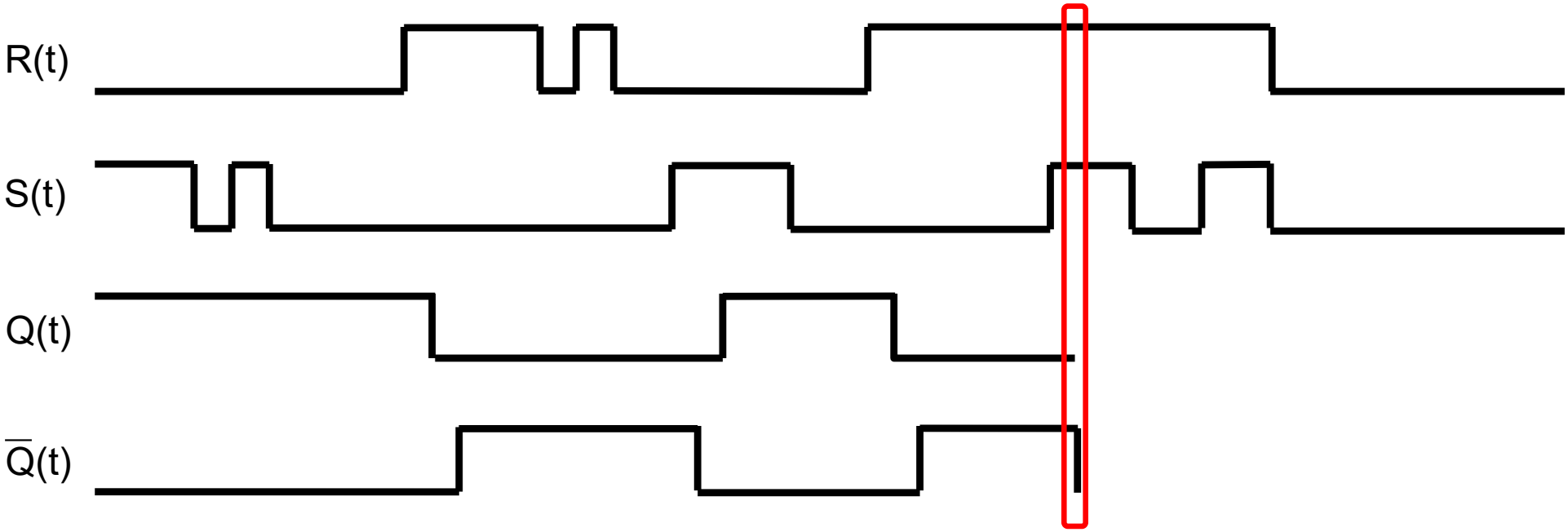


Biestable SR asíncrono

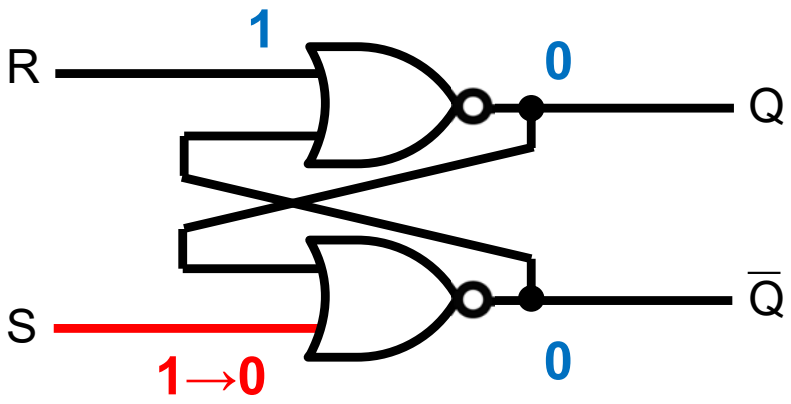


R	$\overline{Q}$	Q
0	0	1
0	1	0
1	0	0
1	1	0

S	Q	$\overline{Q}$
0	0	1
0	1	0
1	0	0
1	1	0

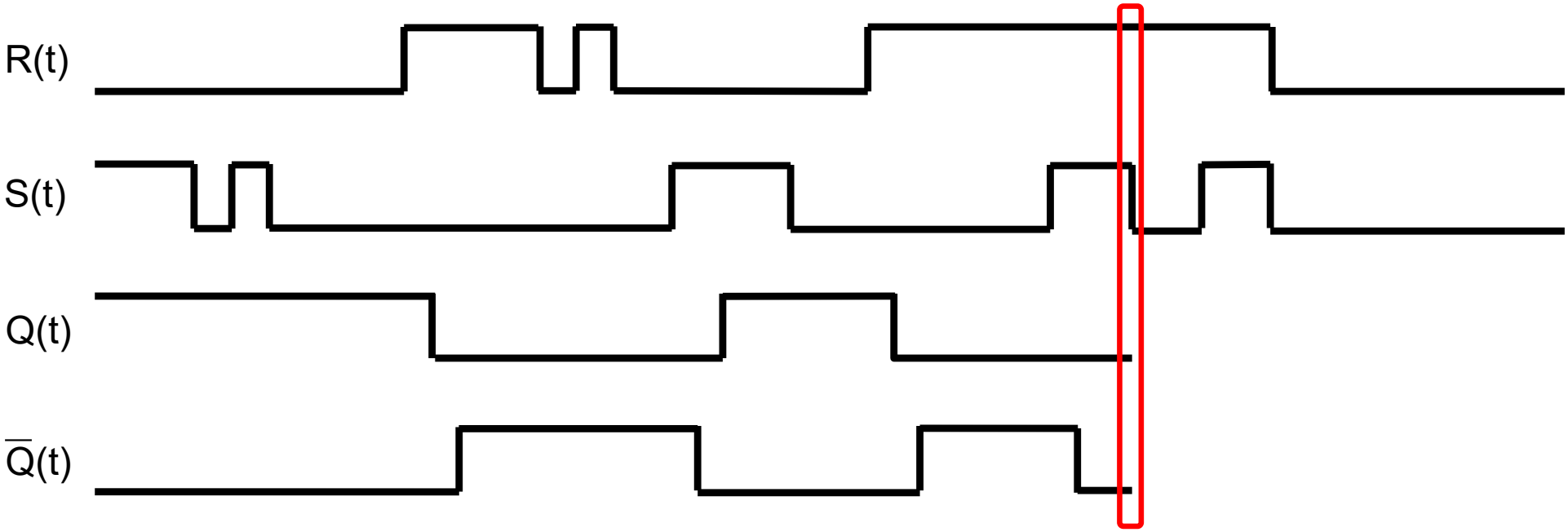


Biestable SR asíncrono

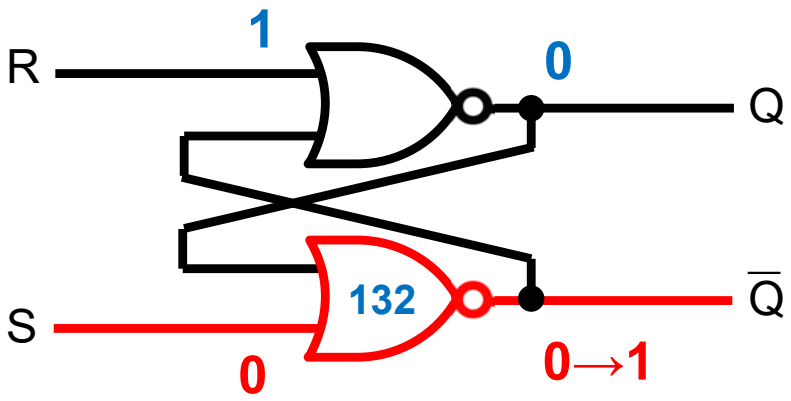


R	$\overline{Q}$	Q
0	0	1
0	1	0
1	0	0
1	1	0

S	Q	$\overline{Q}$
0	0	1
0	1	0
1	0	0
1	1	0

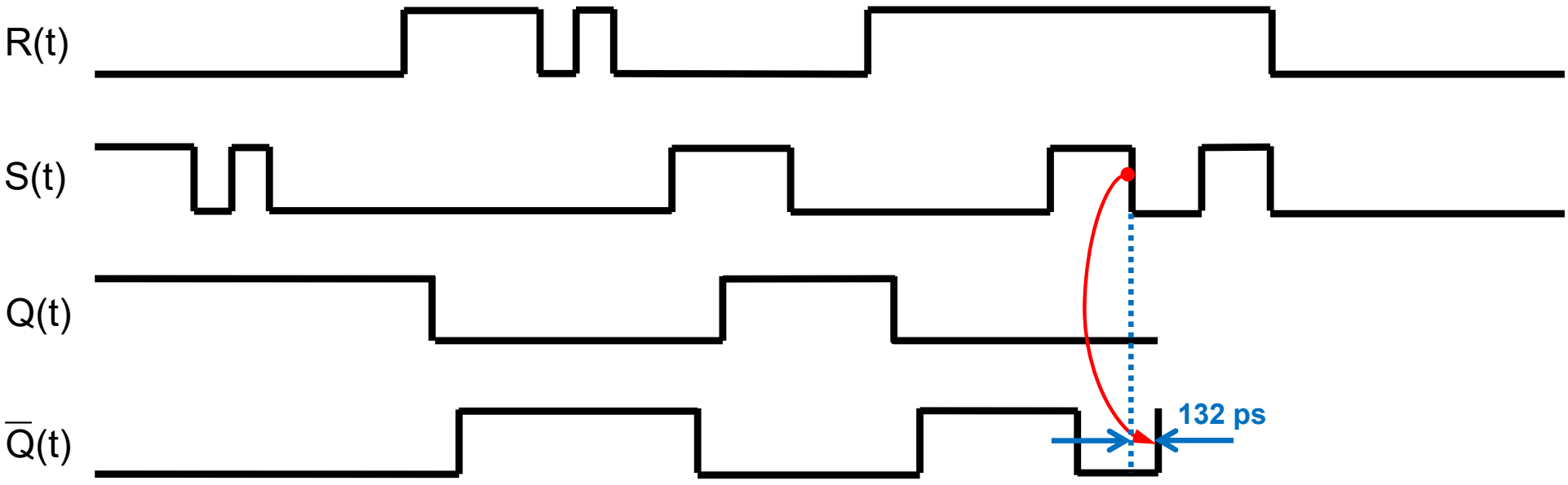


Biestable SR asíncrono

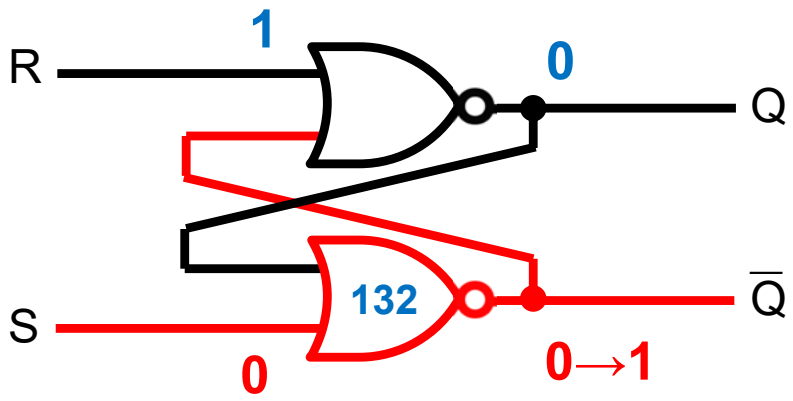


R	$\overline{Q}$	Q
0	0	1
0	1	0
1	0	0
1	1	0

S	Q	$\overline{Q}$
0	0	1
0	1	0
1	0	0
1	1	0

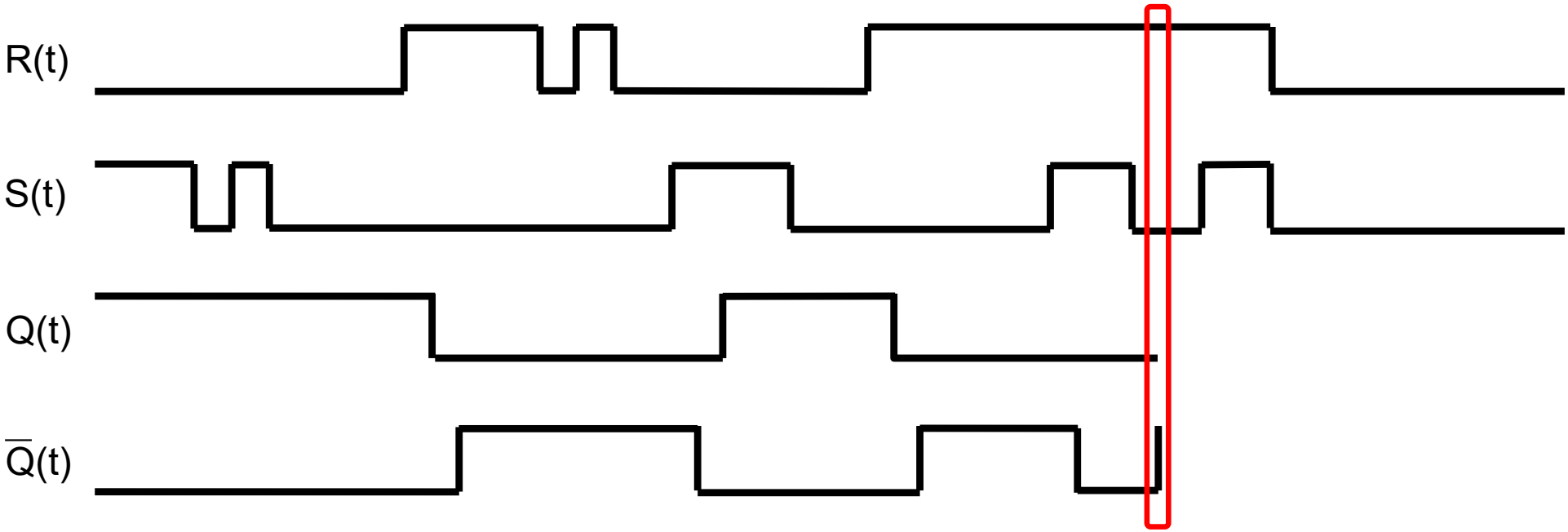


Biestable SR asíncrono

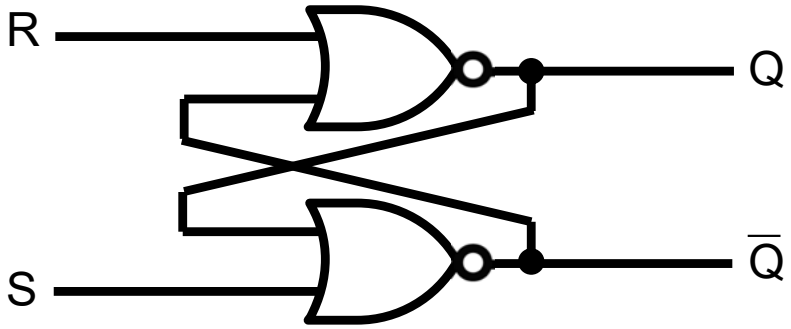


R	$\overline{Q}$	Q
0	0	1
0	1	0
1	0	0
1	1	0

S	Q	$\overline{Q}$
0	0	1
0	1	0
1	0	0
1	1	0

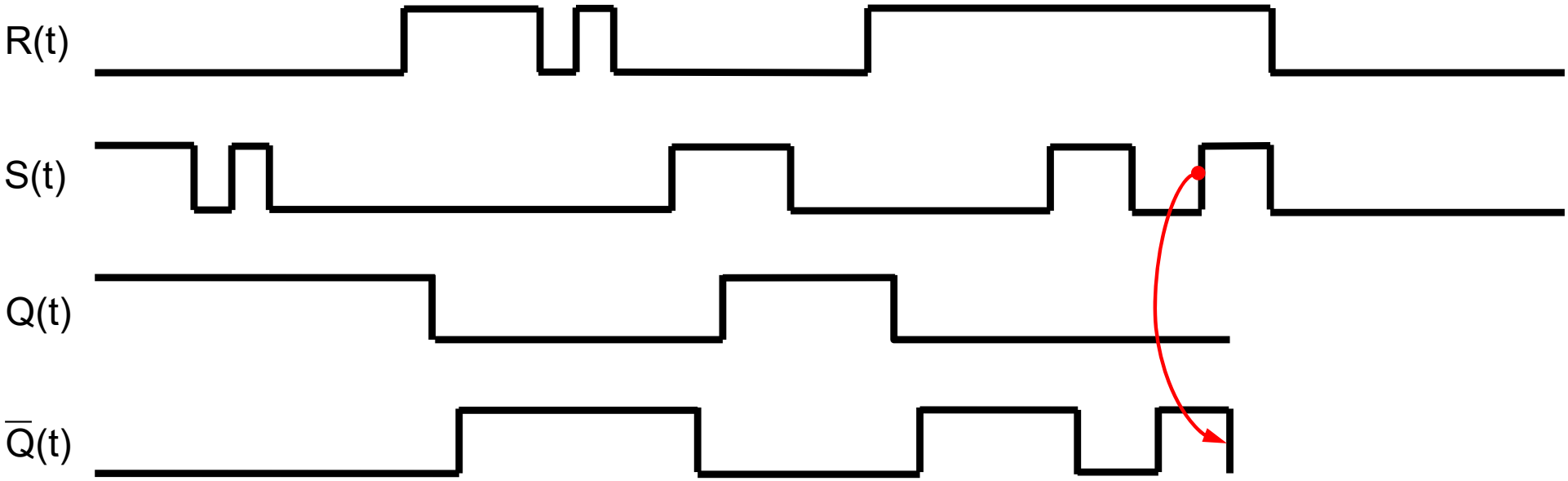


Biestable SR asíncrono

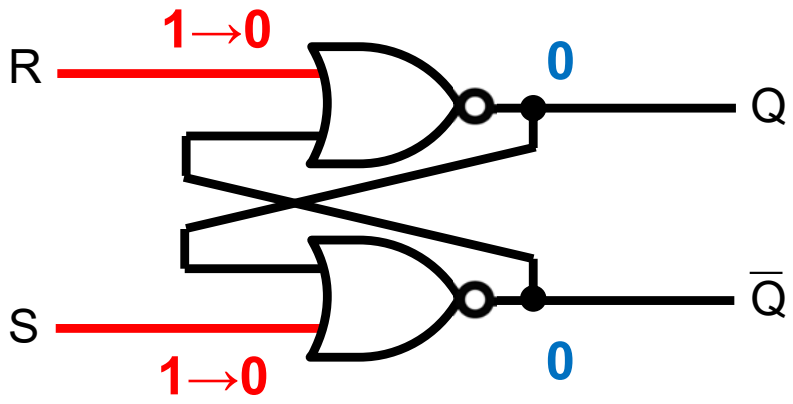


R	$\overline{Q}$	Q
0	0	1
0	1	0
1	0	0
1	1	0

S	Q	$\overline{Q}$
0	0	1
0	1	0
1	0	0
1	1	0

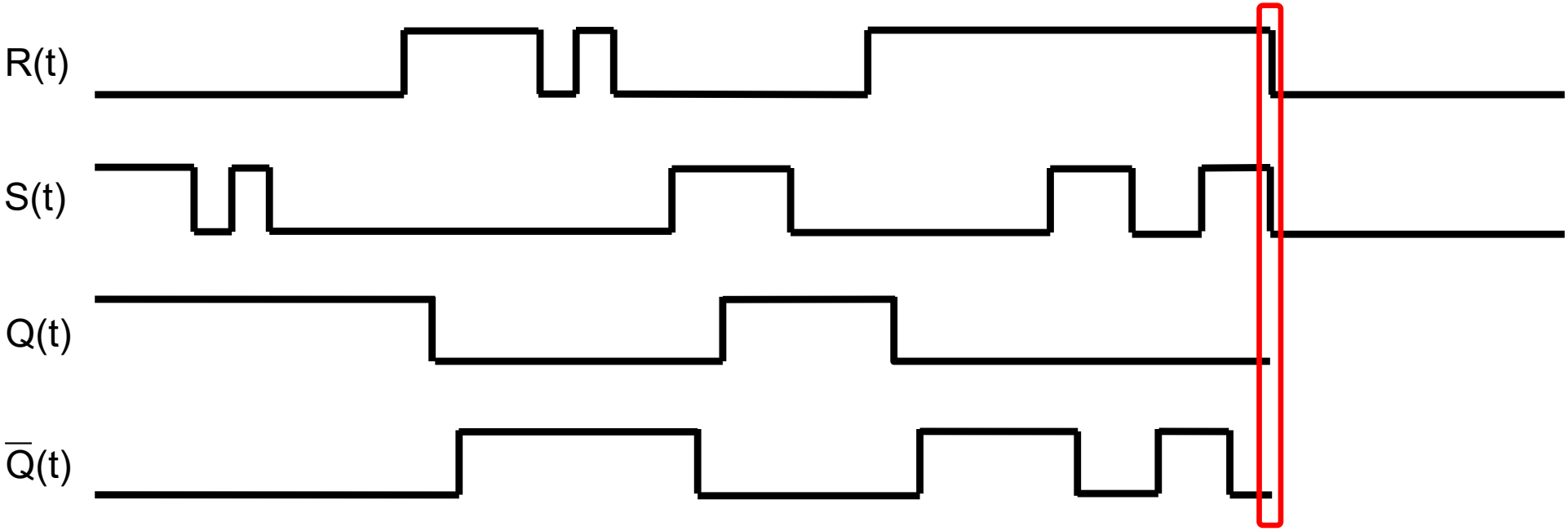


Biestable SR asíncrono

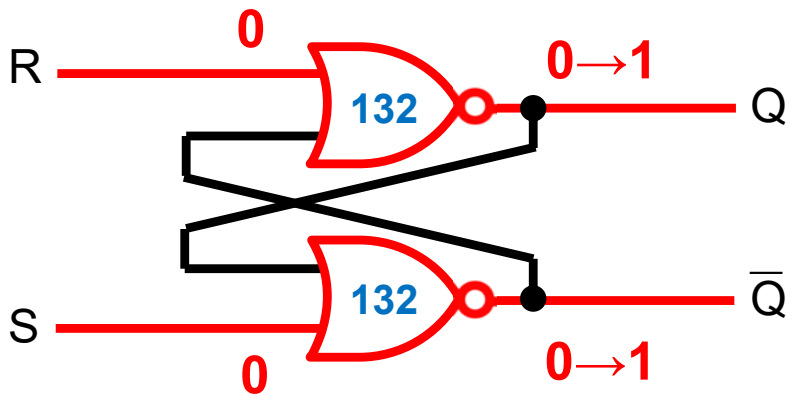


R	$\overline{Q}$	Q
0	0	1
0	1	0
1	0	0
1	1	0

S	Q	$\overline{Q}$
0	0	1
0	1	0
1	0	0
1	1	0

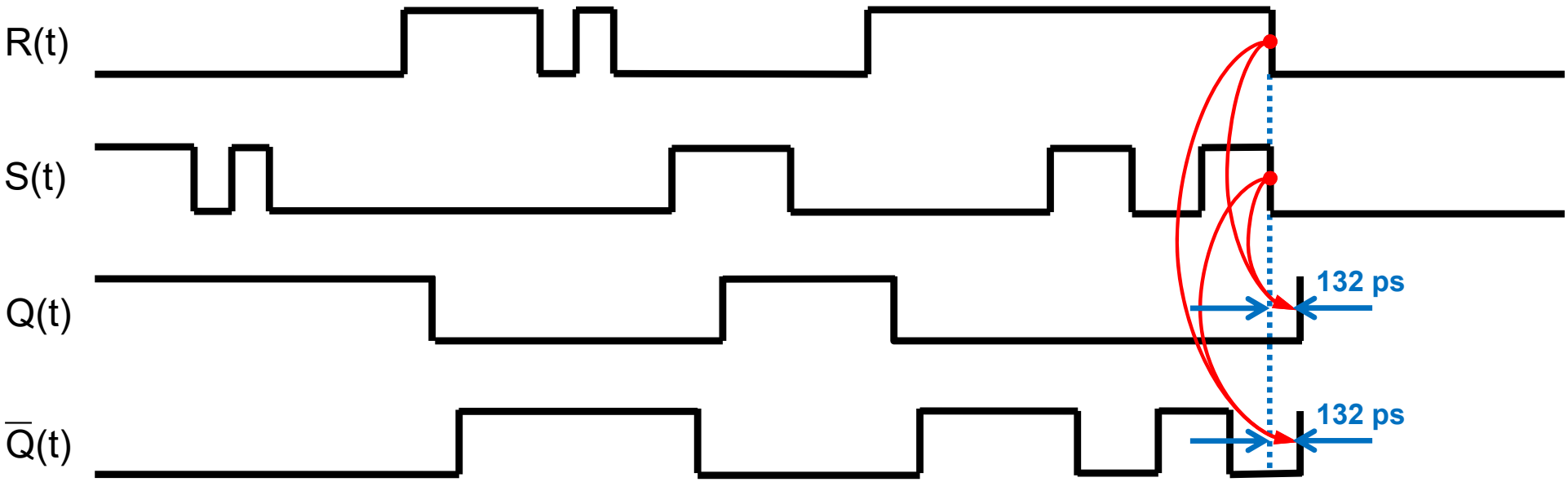


Biestable SR asíncrono

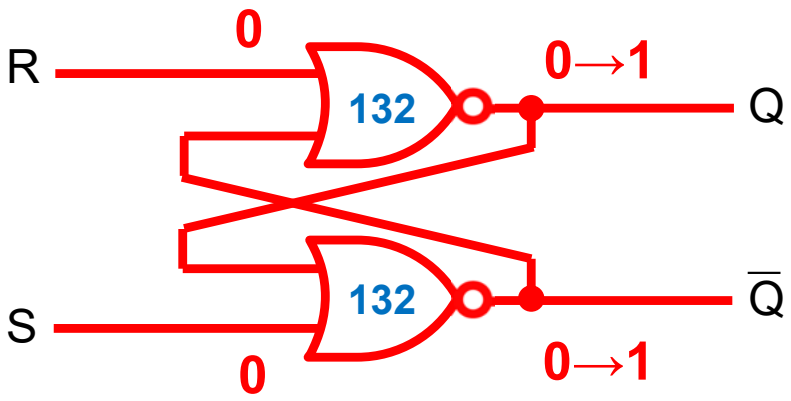


R	$\overline{Q}$	Q
0	0	1
0	1	0
1	0	0
1	1	0

S	Q	$\overline{Q}$
0	0	1
0	1	0
1	0	0
1	1	0

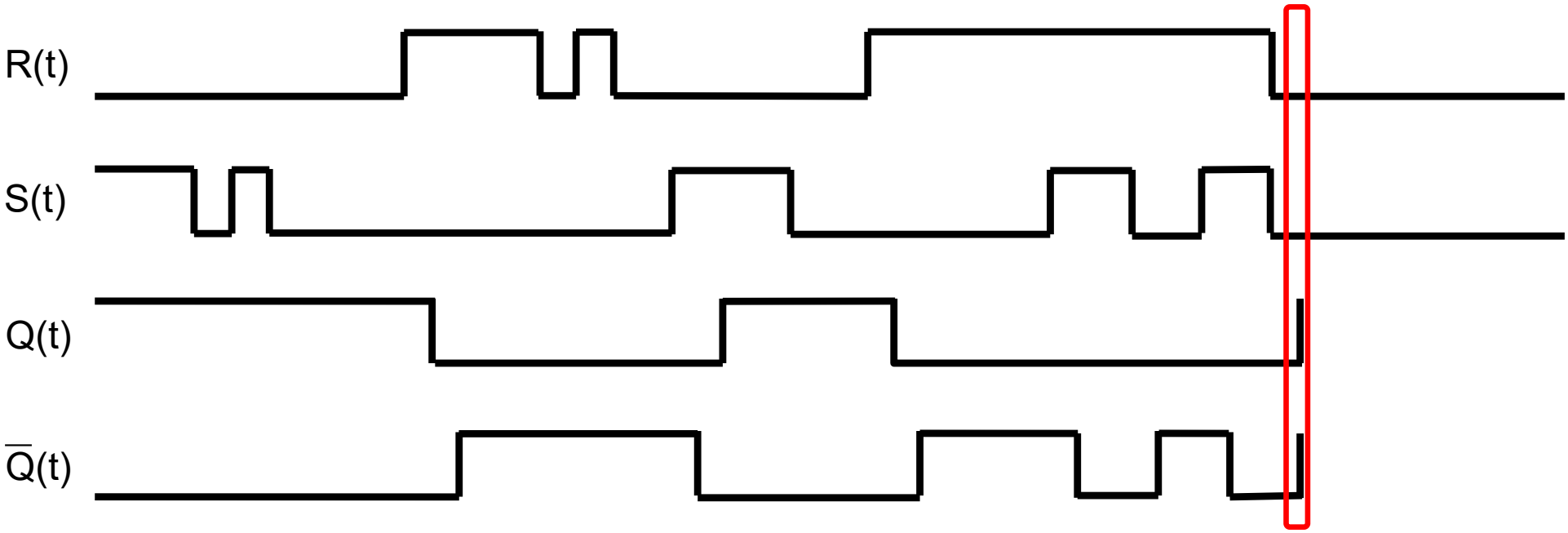


Biestable SR asíncrono

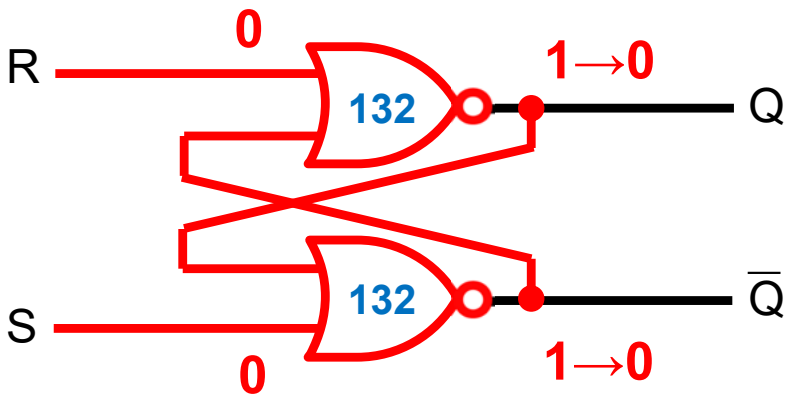


R	$\overline{Q}$	Q
0	0	1
0	1	0
1	0	0
1	1	0

S	Q	$\overline{Q}$
0	0	1
0	1	0
1	0	0
1	1	0

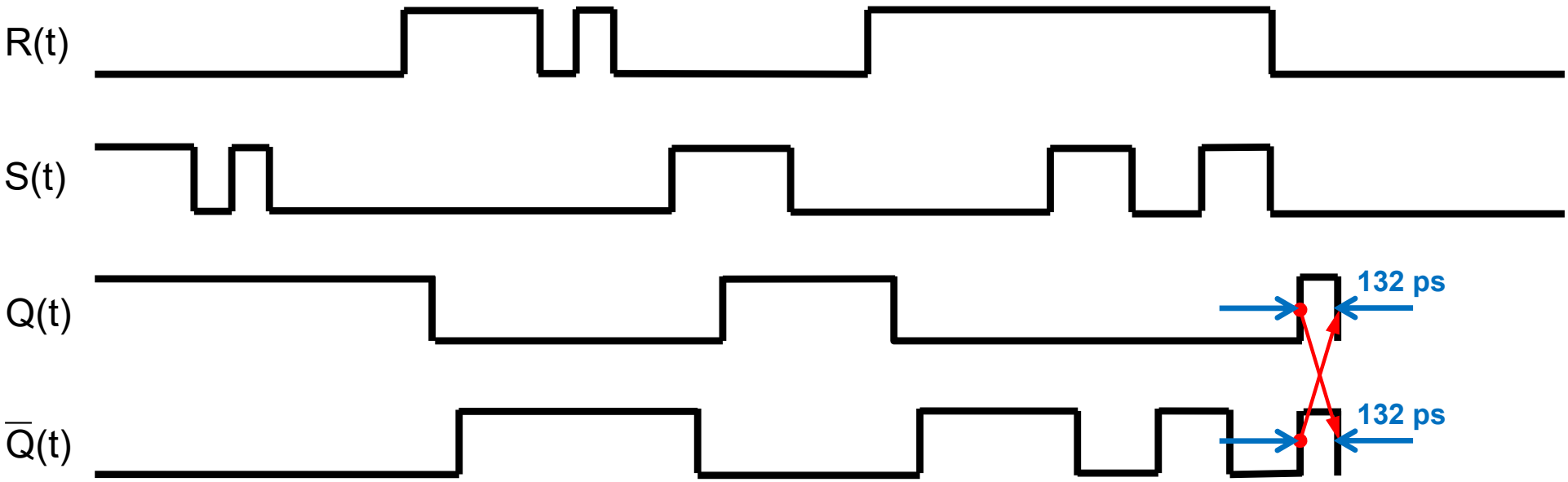


Biestable SR asíncrono

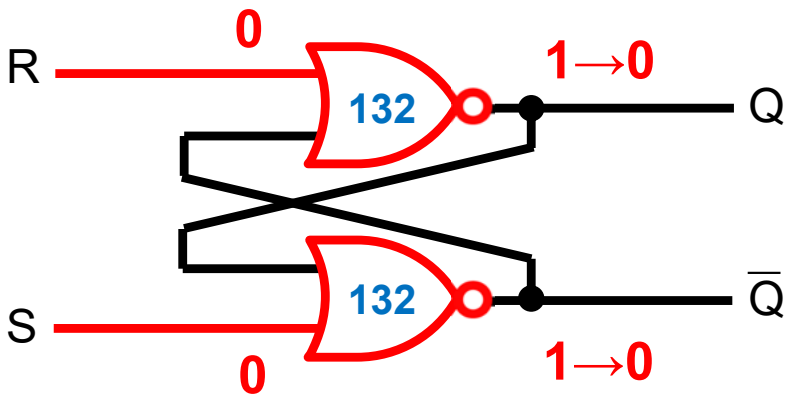


R	$\overline{Q}$	Q
0	0	1
0	1	0
1	0	0
1	1	0

S	Q	$\overline{Q}$
0	0	1
0	1	0
1	0	0
1	1	0

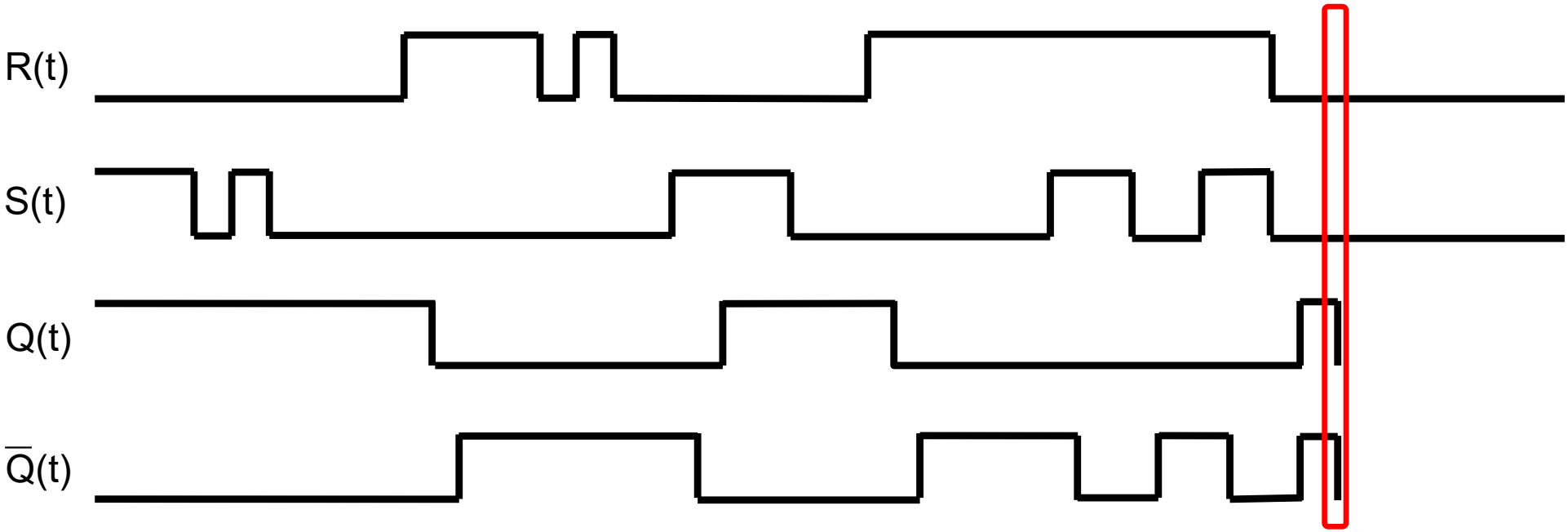


Biestable SR asíncrono

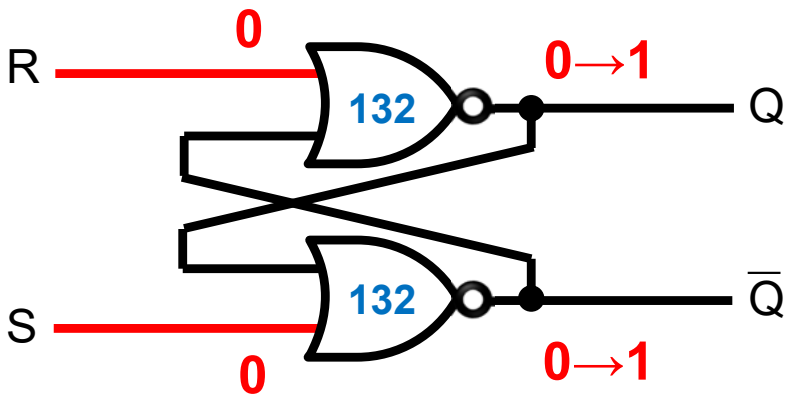


R	$\overline{Q}$	Q
0	0	1
0	1	0
1	0	0
1	1	0

S	Q	$\overline{Q}$
0	0	1
0	1	0
1	0	0
1	1	0

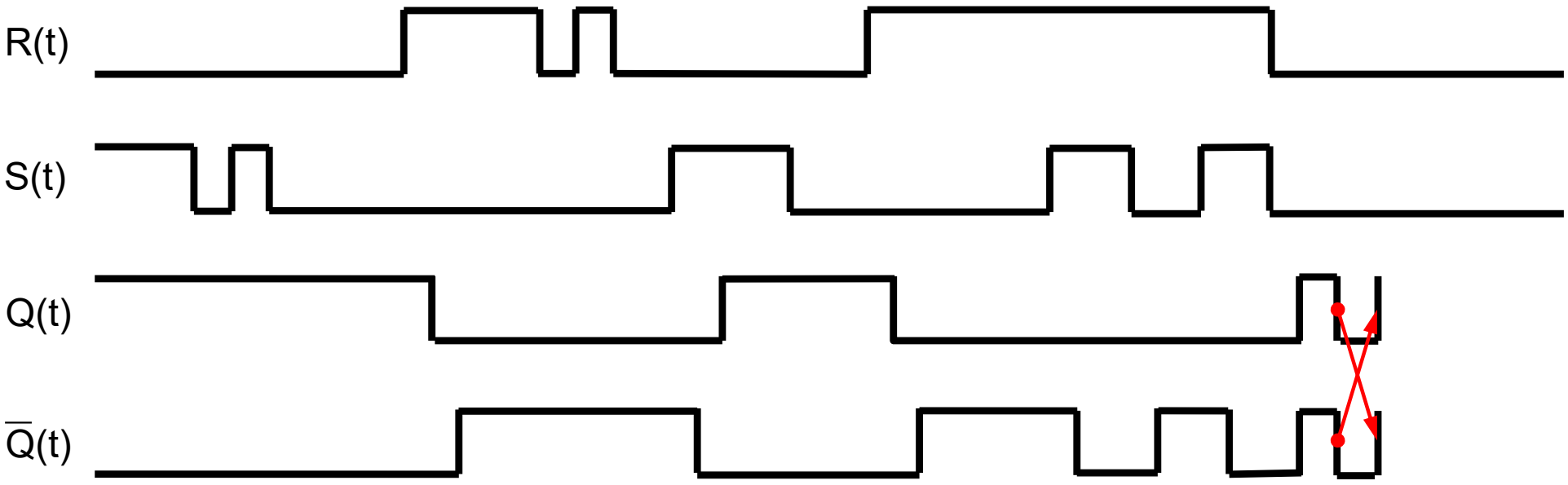


Biestable SR asíncrono

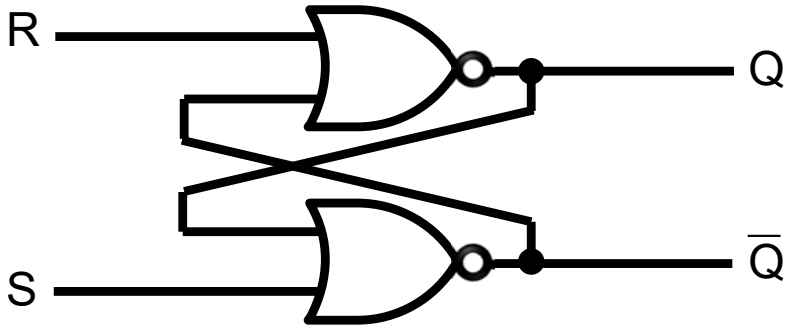


R	$\overline{Q}$	Q
0	0	1
0	1	0
1	0	0
1	1	0

S	Q	$\overline{Q}$
0	0	1
0	1	0
1	0	0
1	1	0

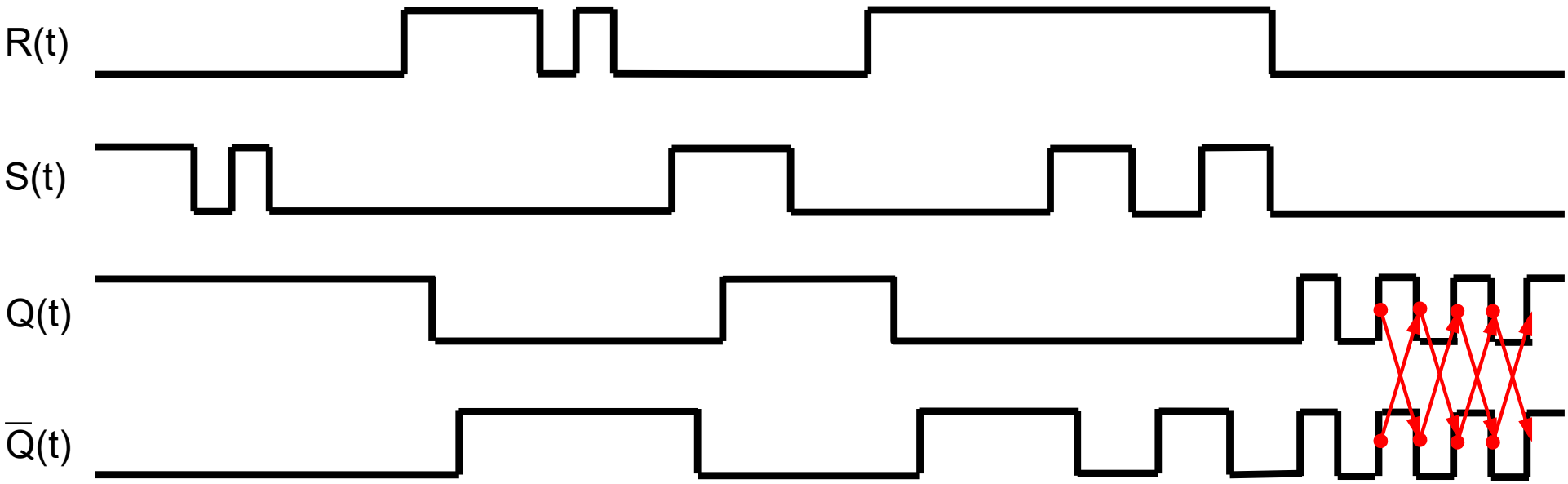


Biestable SR asíncrono

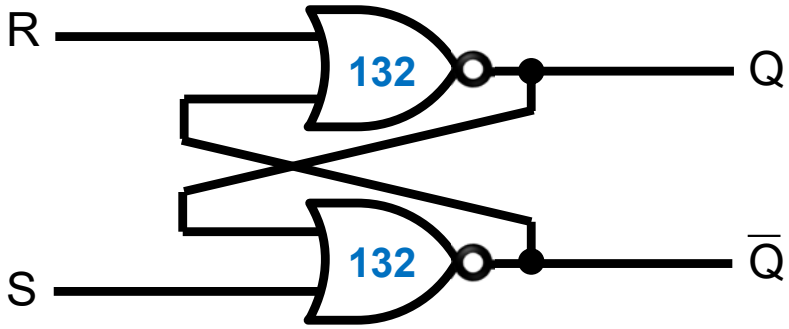


R	$\overline{Q}$	Q
0	0	1
0	1	0
1	0	0
1	1	0

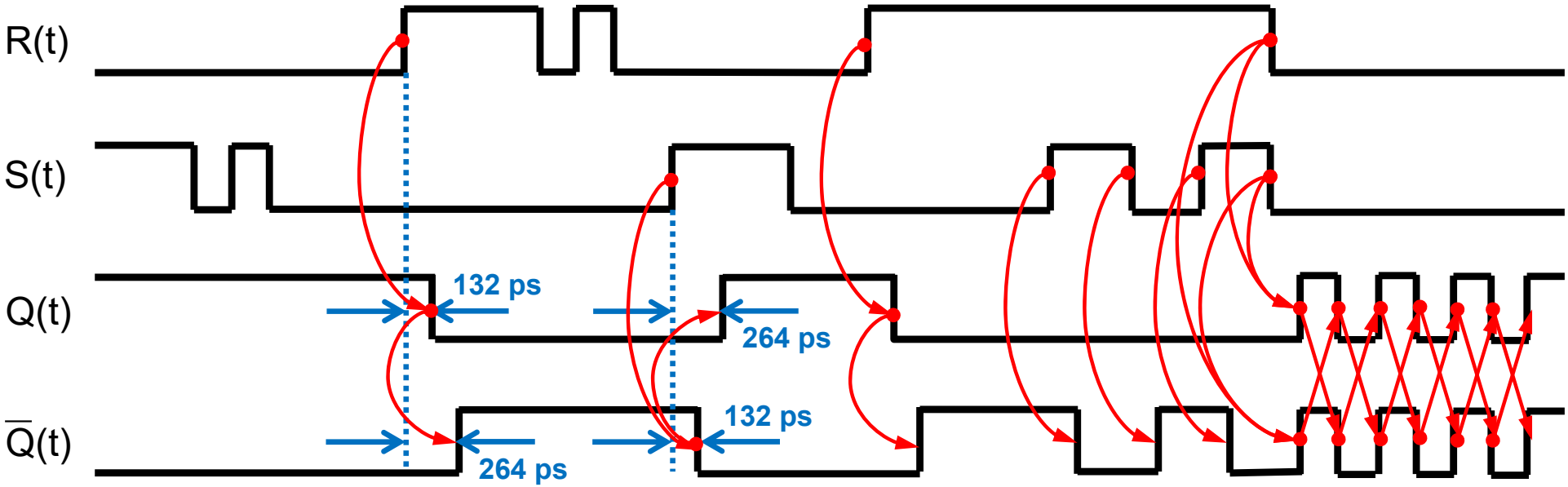
S	Q	$\overline{Q}$
0	0	1
0	1	0
1	0	0
1	1	0



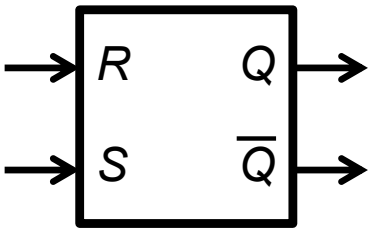
Biestable SR asíncrono



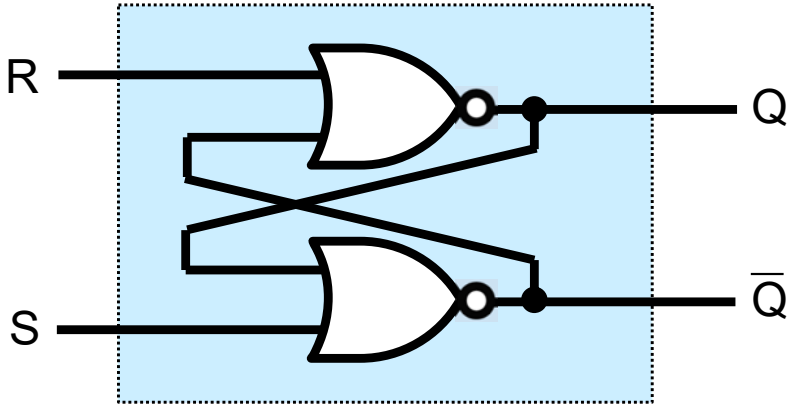
R(t)	S(t)	Q(t+Δt)
0	0	Q(t)
0	1	1
1	0	0
1	1	prohibido



Biestable SR asíncrono

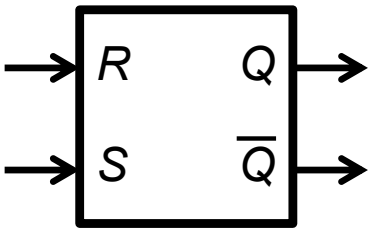


R(t)	S(t)	Q(t+Δt)
0	0	Q(t)
0	1	1
1	0	0
1	1	prohibido

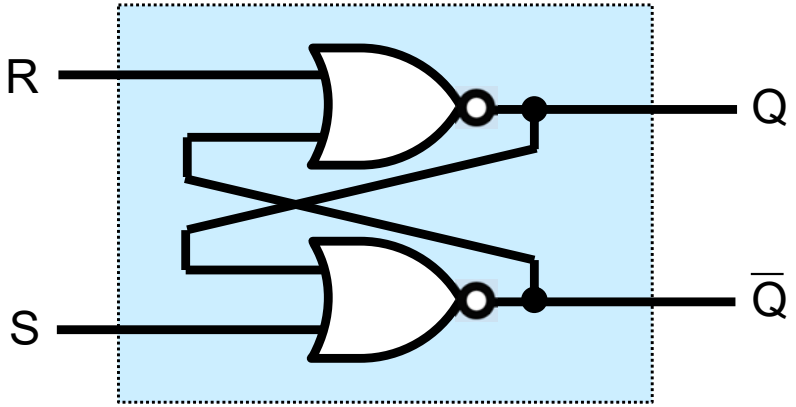


Biestable SR asíncrono
(implementación con NOR)

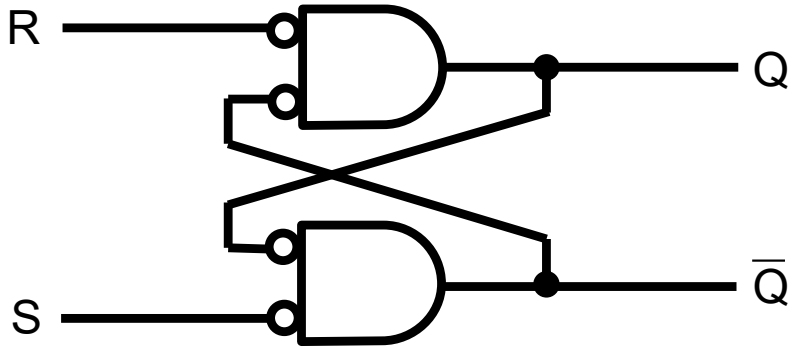
Biestable SR asíncrono



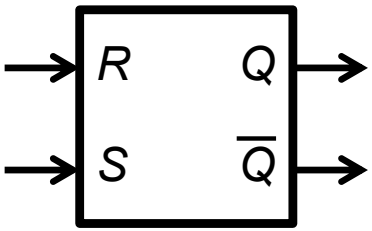
R(t)	S(t)	Q(t+Δt)
0	0	Q(t)
0	1	1
1	0	0
1	1	prohibido



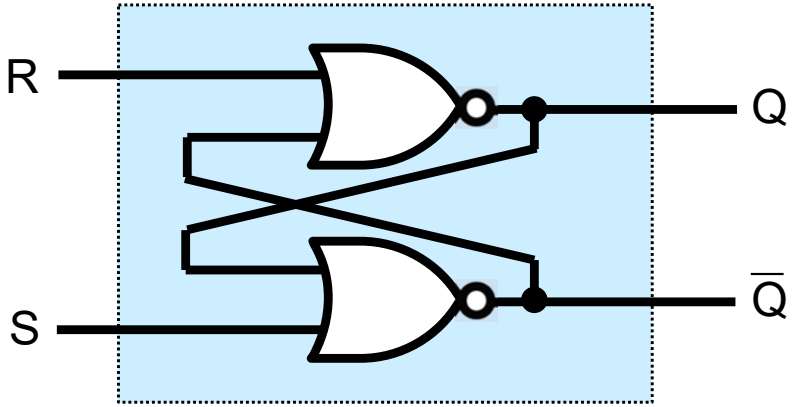
Biestable SR asíncrono
(implementación con NOR)



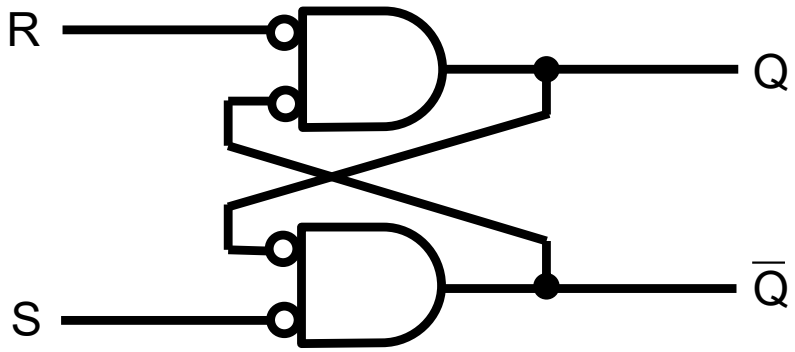
Biestable SR asíncrono

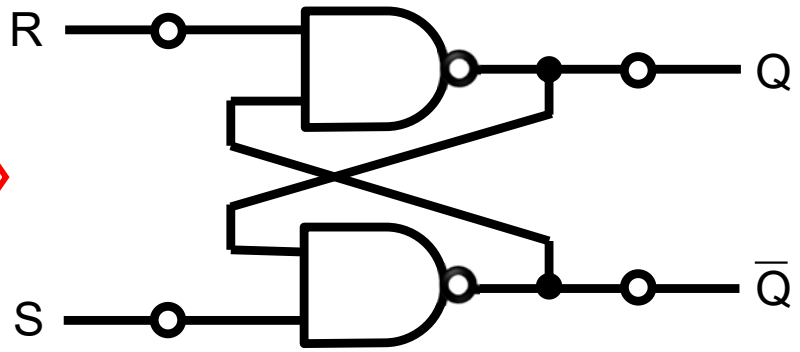


R(t)	S(t)	Q(t+Δt)
0	0	Q(t)
0	1	1
1	0	0
1	1	prohibido

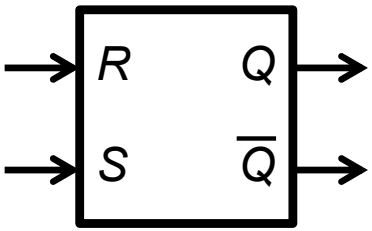


Biestable SR asíncrono
(implementación con NOR)

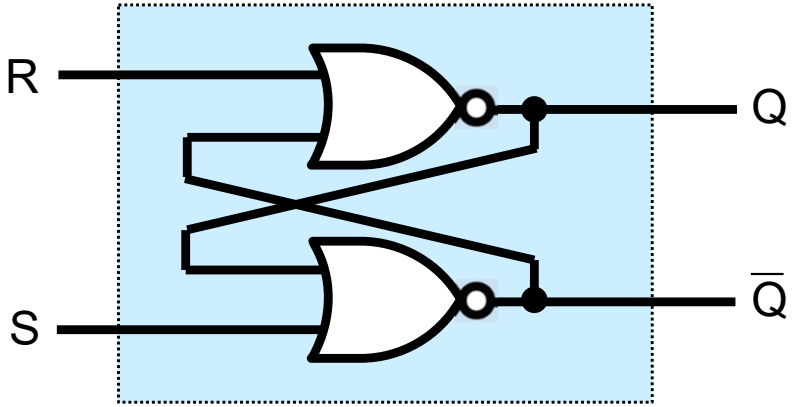




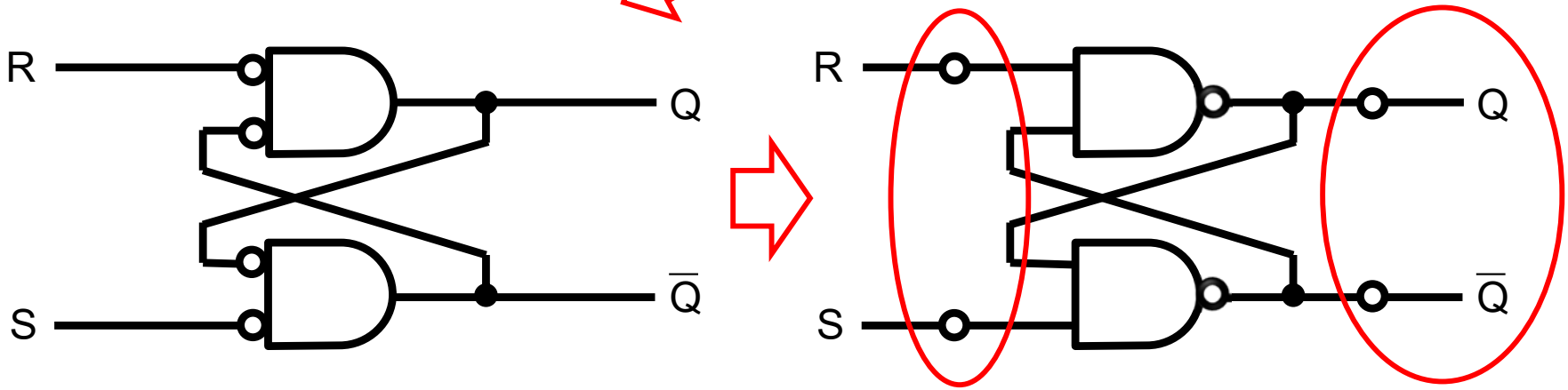
Biestable SR asíncrono



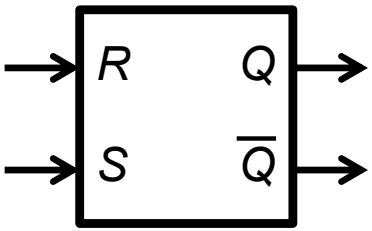
R(t)	S(t)	Q(t+Δt)
0	0	Q(t)
0	1	1
1	0	0
1	1	prohibido



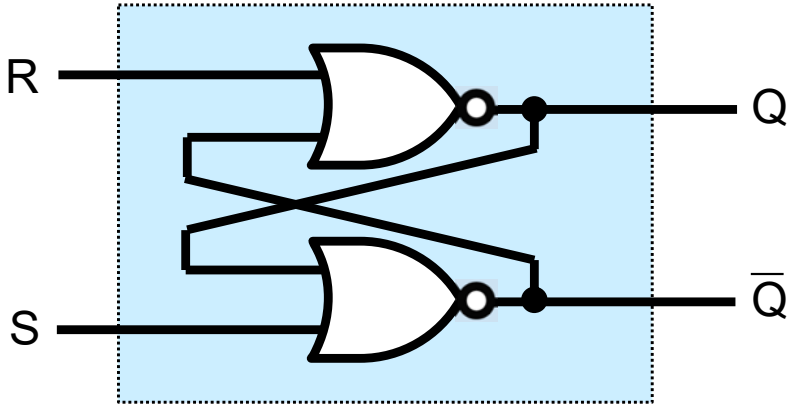
Biestable SR asíncrono
(implementación con NOR)



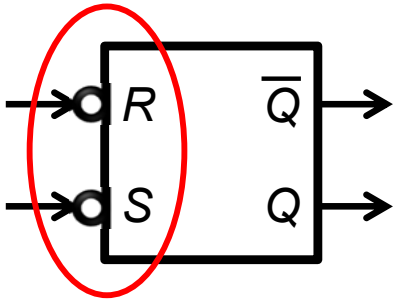
Biestable SR asíncrono

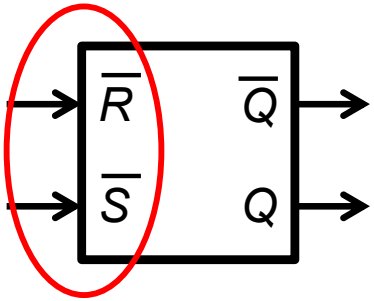


R(t)	S(t)	Q(t+Δt)
0	0	Q(t)
0	1	1
1	0	0
1	1	prohibido

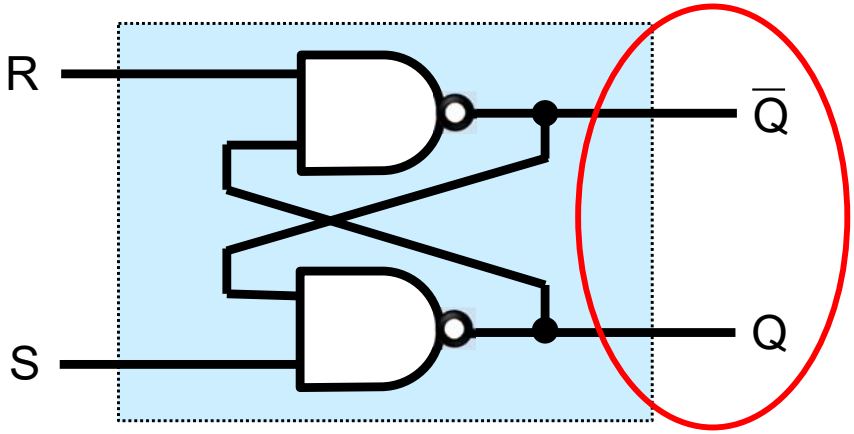


Biestable SR asíncrono
(implementación con NOR)



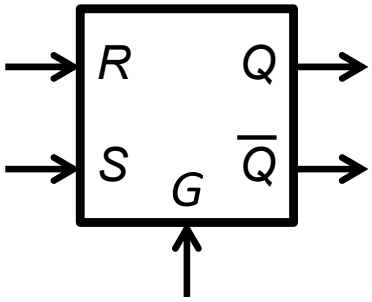


R(t)	S(t)	Q(t+Δt)
0	0	prohibido
0	1	0
1	0	1
1	1	Q(t)



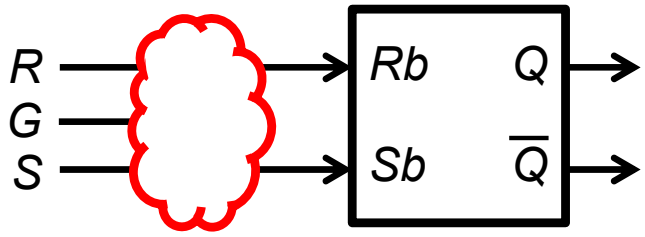
Biestable SR asíncrono
(implementación con NAND)

Biestable SR síncrono (por nivel)



G(t)	R(t)	S(t)	Q(t+Δt)
0	X	X	Q(t)
1	0	0	Q(t)
1	0	1	1
1	1	0	0
1	1	1	prohibido

Biestable SR síncrono (por nivel)



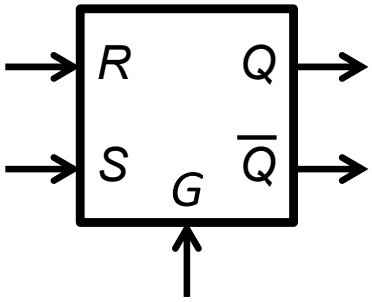
$$Rb/Sb = \begin{cases} 0 & \text{si } G=0 \\ R/S & \text{si } G=1 \end{cases}$$

G	R	Rb
0	0	0
0	1	0
1	0	0
1	1	1

$Rb = G \cdot R$

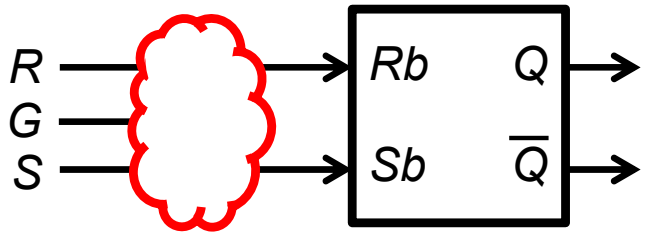
G	S	Sb
0	0	0
0	1	0
1	0	0
1	1	1

$Sb = G \cdot S$



G(t)	R(t)	S(t)	Q(t+Δt)
0	X	X	Q(t)
1	0	0	Q(t)
1	0	1	1
1	1	0	0
1	1	1	prohibido

Biestable SR síncrono (por nivel)



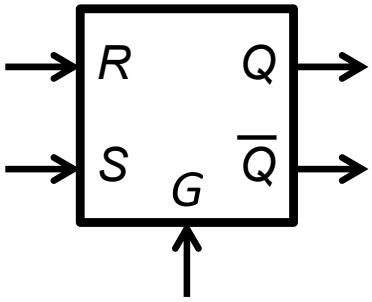
$$Rb/Sb = \begin{cases} 0 & \text{si } G=0 \\ R/S & \text{si } G=1 \end{cases}$$

G	R	Rb
0	0	0
0	1	0
1	0	0
1	1	1

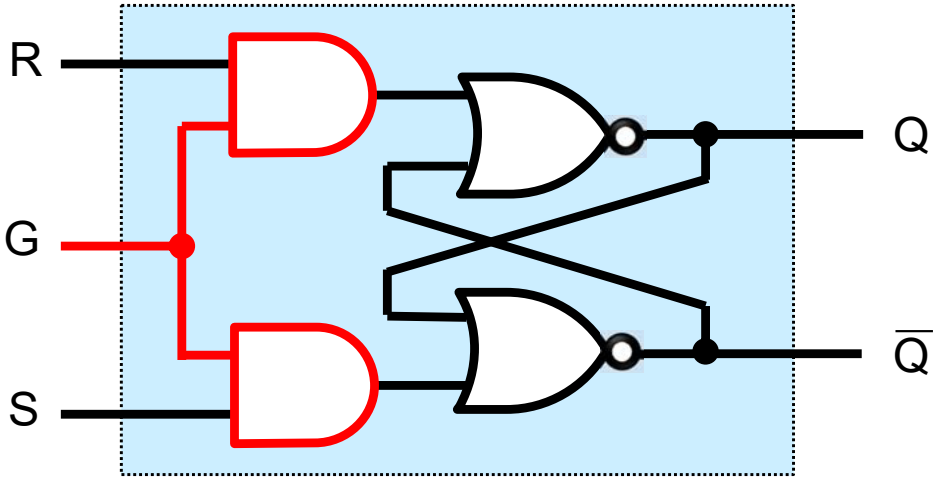
$Rb = G \cdot R$

G	S	Sb
0	0	0
0	1	0
1	0	0
1	1	1

$Sb = G \cdot S$

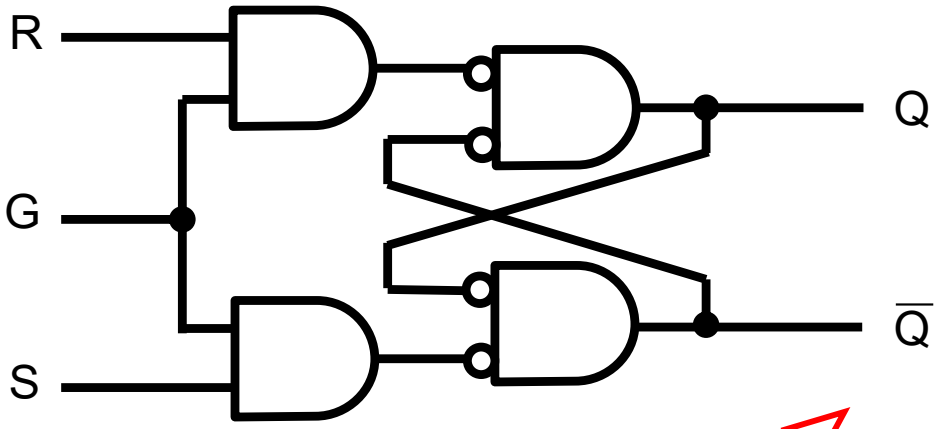


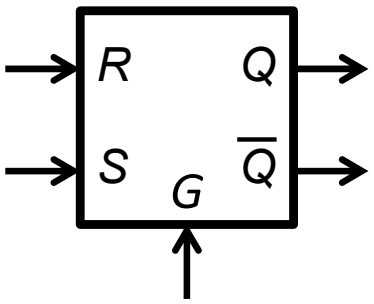
G(t)	R(t)	S(t)	Q(t+Δt)
0	X	X	Q(t)
1	0	0	Q(t)
1	0	1	1
1	1	0	0
1	1	1	prohibido



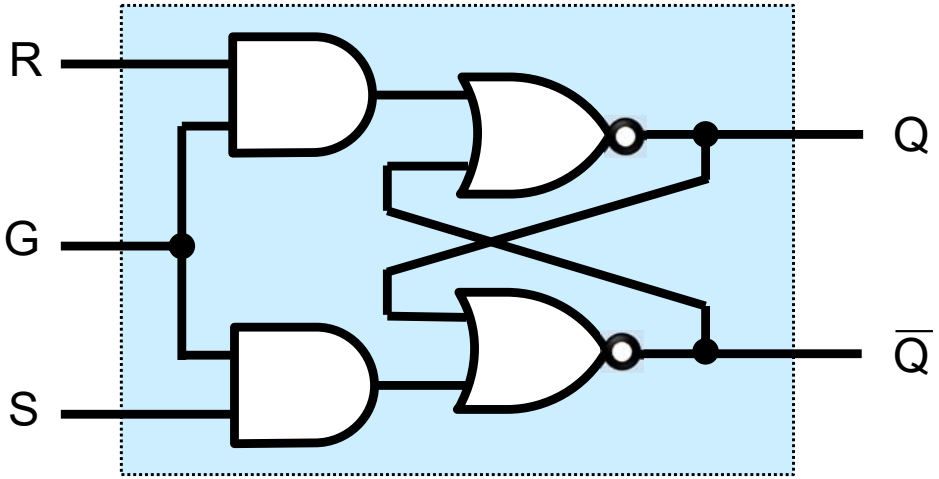
Biestable SR síncrono disparado por nivel
(Latch SR)

Biestable SR síncrono (por nivel)



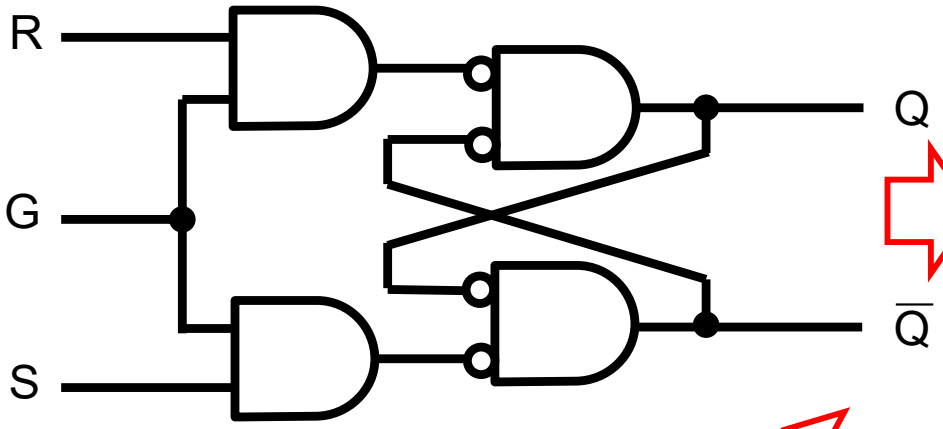


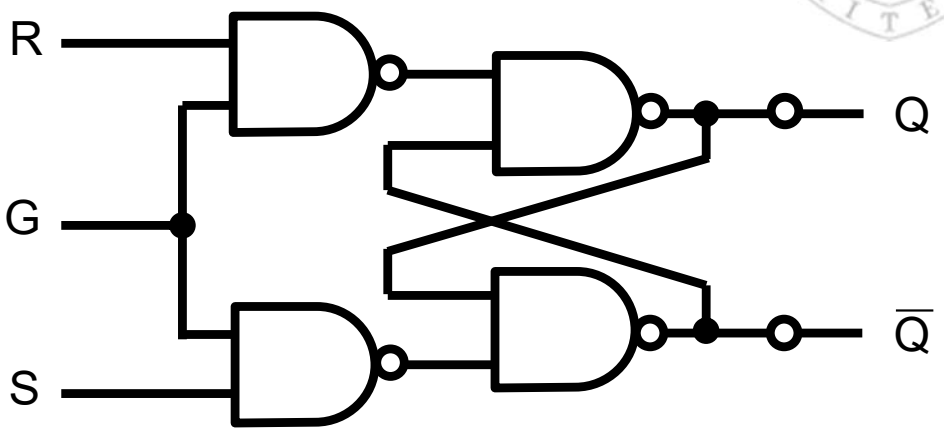
G(t)	R(t)	S(t)	Q(t+Δt)
0	X	X	Q(t)
1	0	0	Q(t)
1	0	1	1
1	1	0	0
1	1	1	prohibido

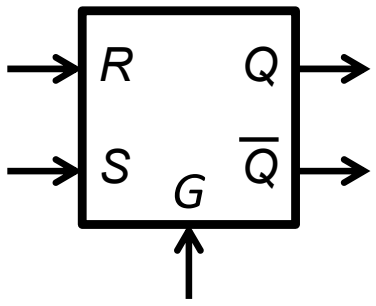


Biestable SR síncrono disparado por nivel
(Latch SR)

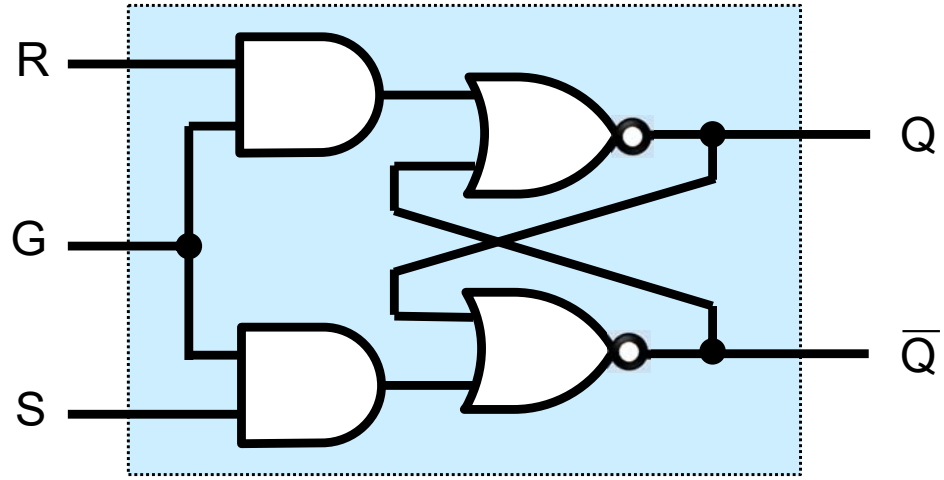
Biestable SR síncrono (por nivel)





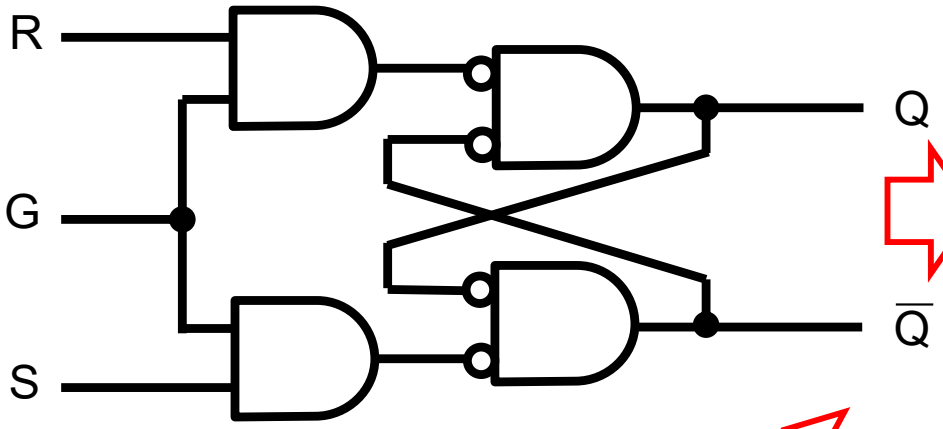


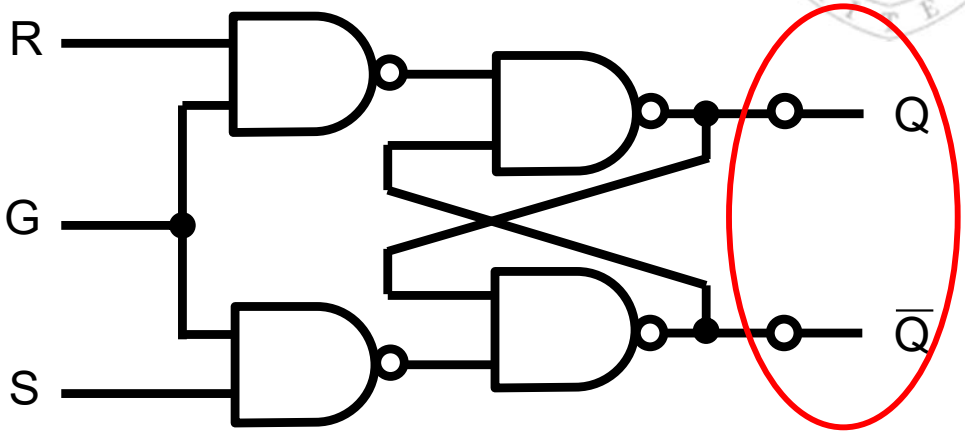
G(t)	R(t)	S(t)	Q(t+Δt)
0	X	X	Q(t)
1	0	0	Q(t)
1	0	1	1
1	1	0	0
1	1	1	prohibido

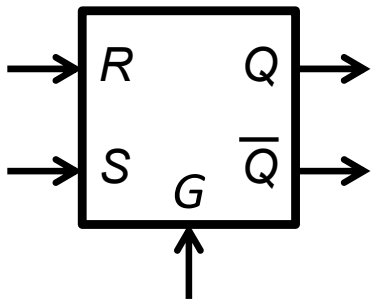


Biestable SR síncrono disparado por nivel
(Latch SR)

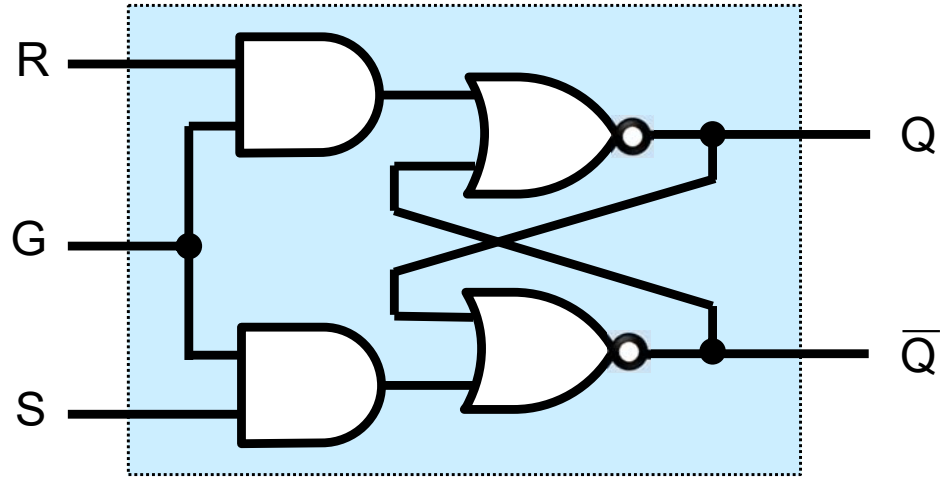
Biestable SR síncrono (por nivel)





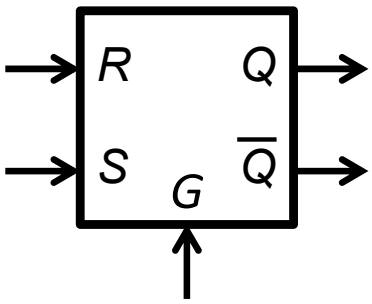


G(t)	R(t)	S(t)	Q(t+Δt)
0	X	X	Q(t)
1	0	0	Q(t)
1	0	1	1
1	1	0	0
1	1	1	prohibido

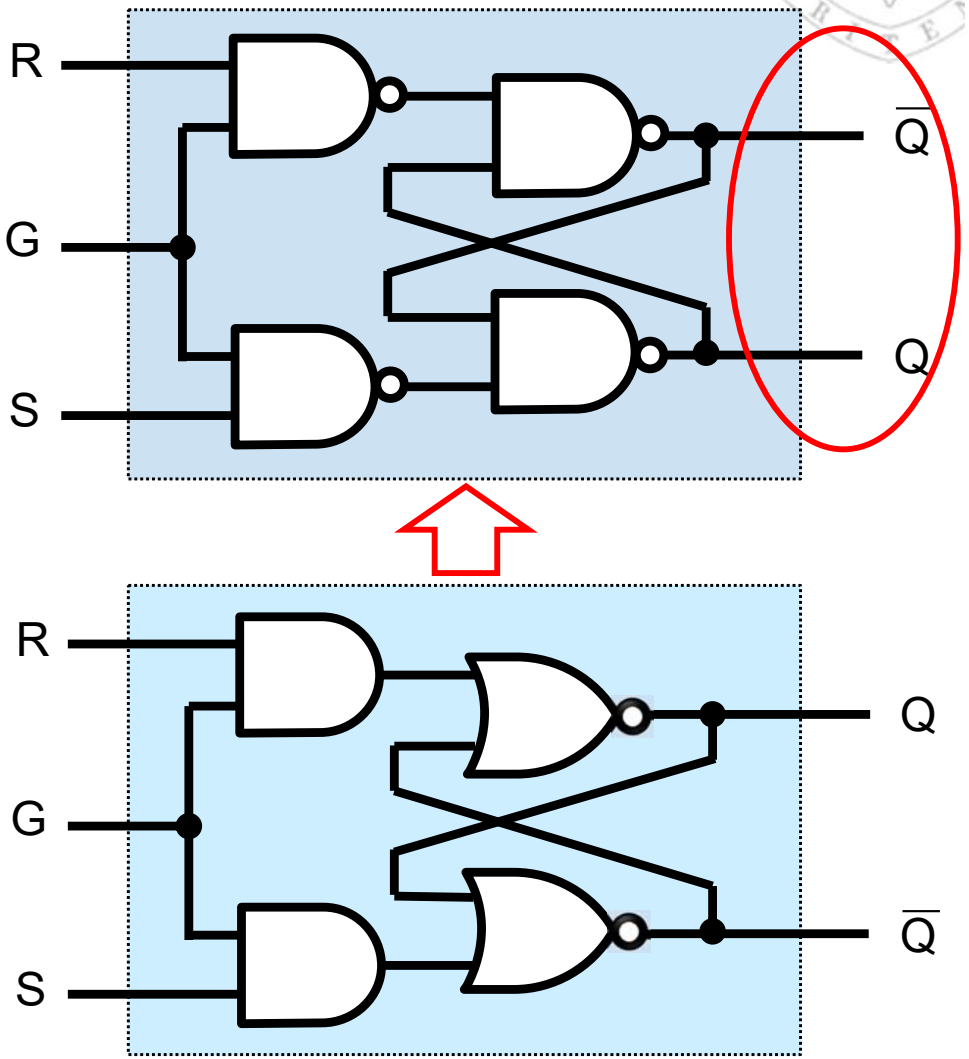


Biestable SR síncrono disparado por nivel
(Latch SR)

Biestable SR síncrono (por nivel)

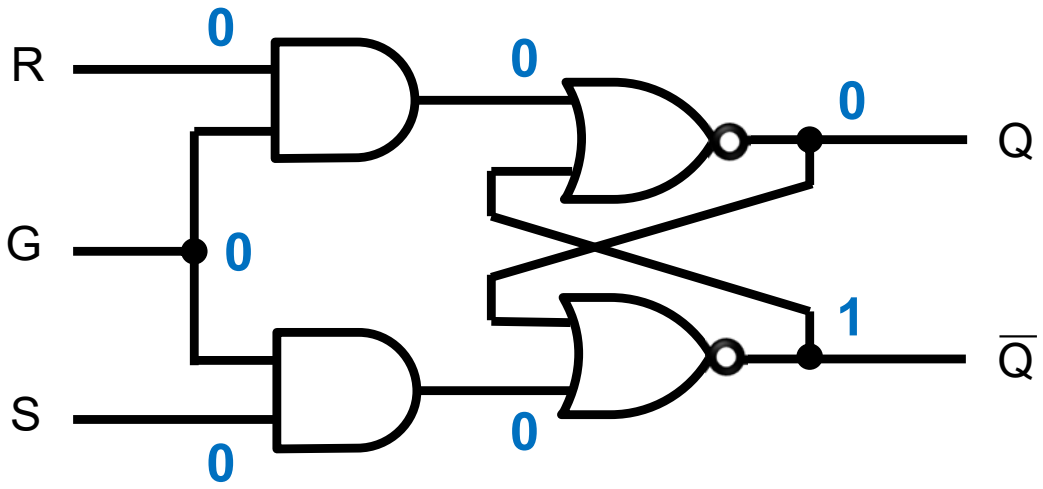


G(t)	R(t)	S(t)	Q(t+Δt)
0	X	X	Q(t)
1	0	0	Q(t)
1	0	1	1
1	1	0	0
1	1	1	prohibido

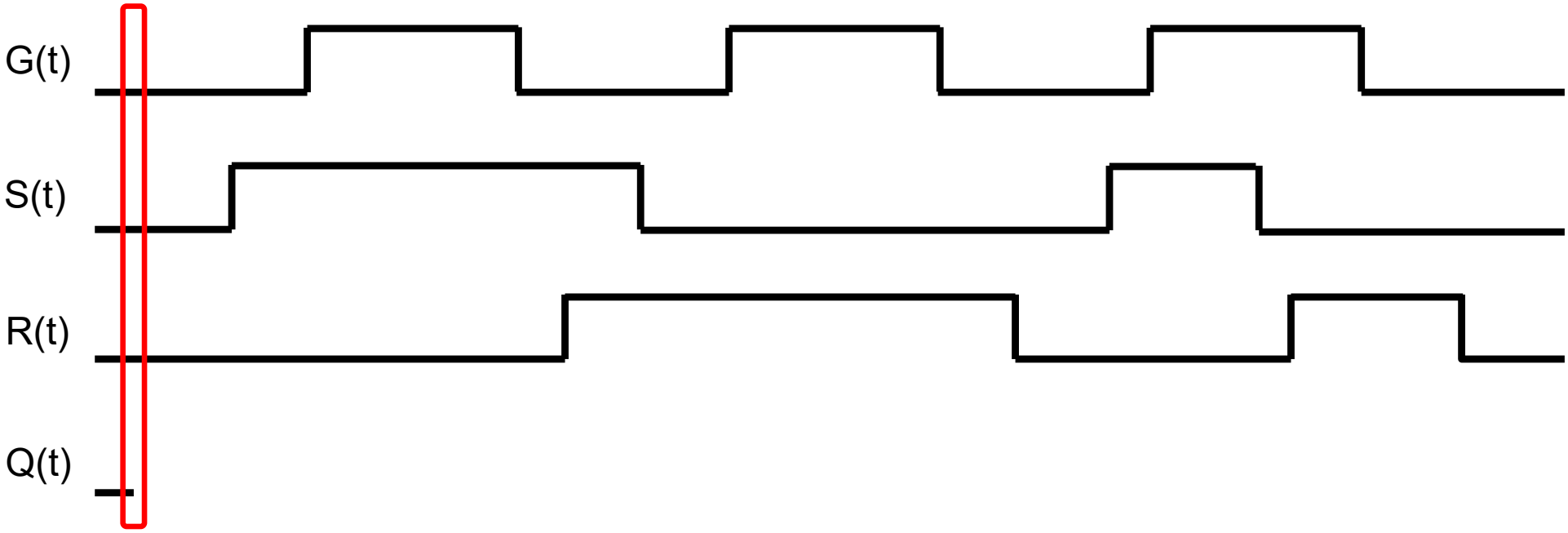


Biestable SR síncrono disparado por nivel
(Latch SR)

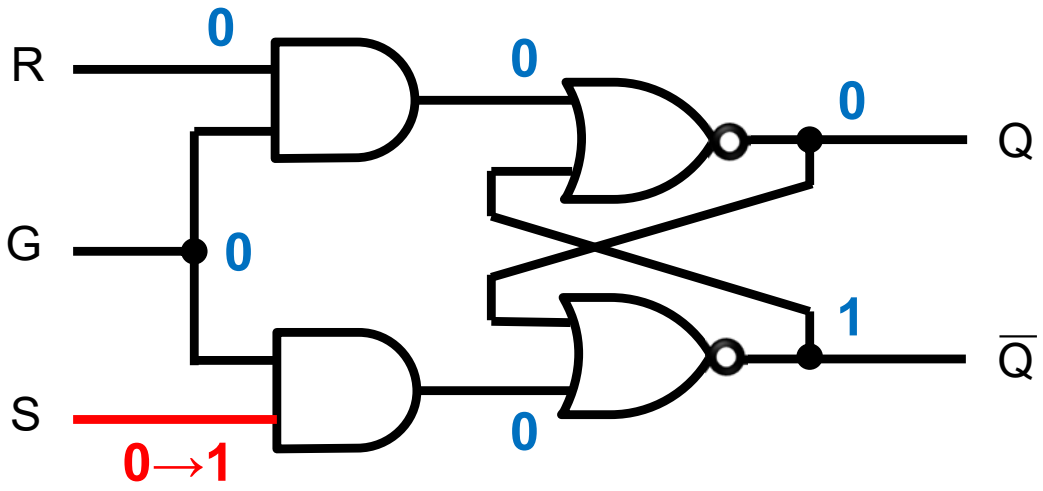
Biestable SR síncrono (por nivel)



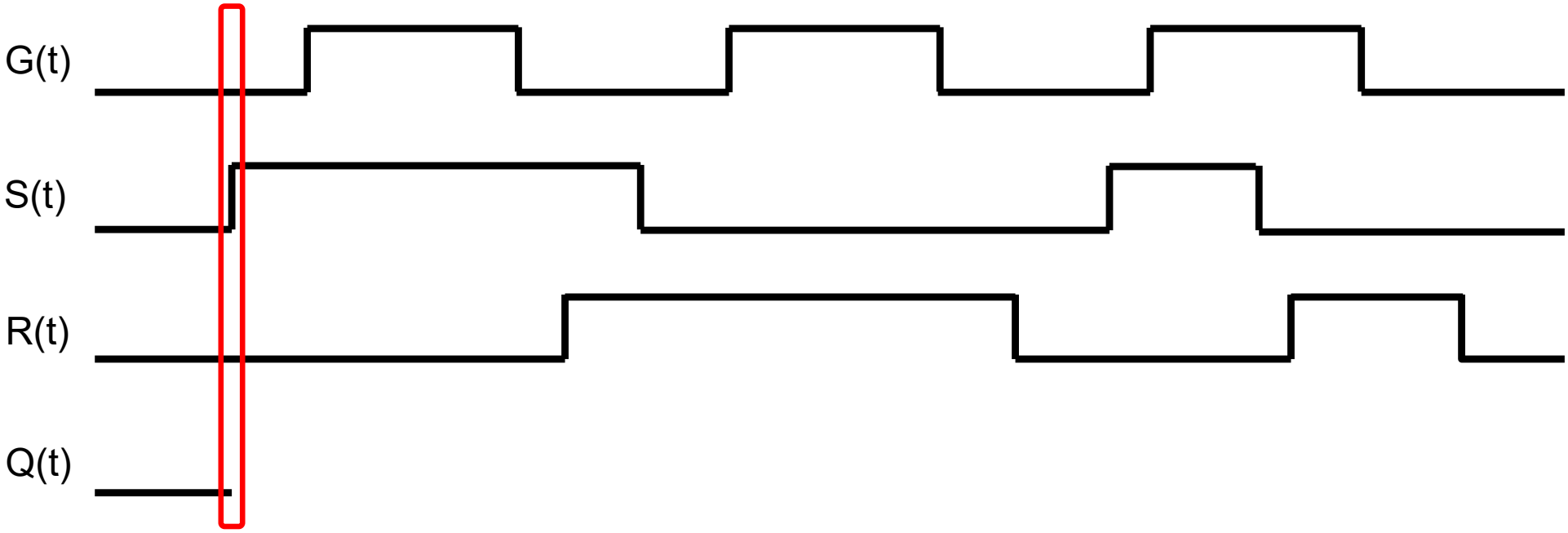
R(t)	S(t)	G(t)	Q(t+Δt)
X	X	0	Q(t)
0	0	1	Q(t)
0	1	1	1
1	0	1	0
1	1	1	prohibido



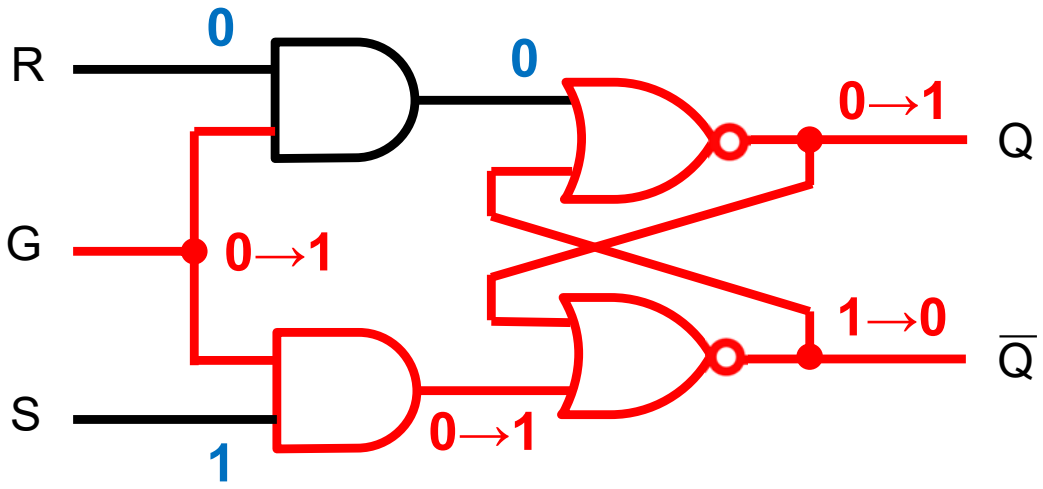
Biestable SR síncrono (por nivel)



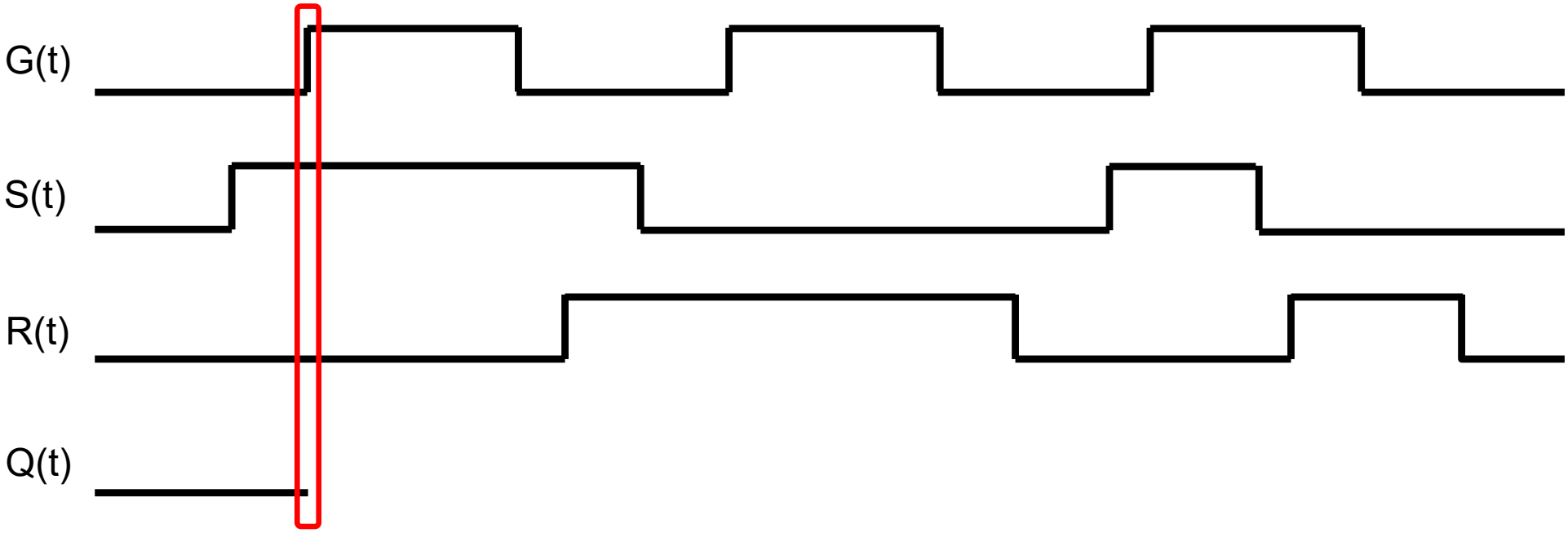
R(t)	S(t)	G(t)	Q(t+Δt)
X	X	0	Q(t)
0	0	1	Q(t)
0	1	1	1
1	0	1	0
1	1	1	prohibido



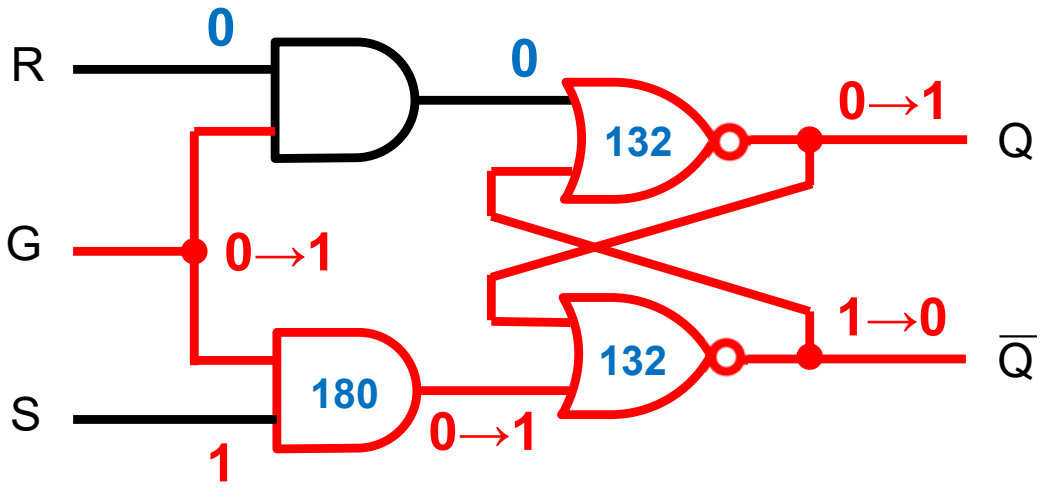
Biestable SR síncrono (por nivel)



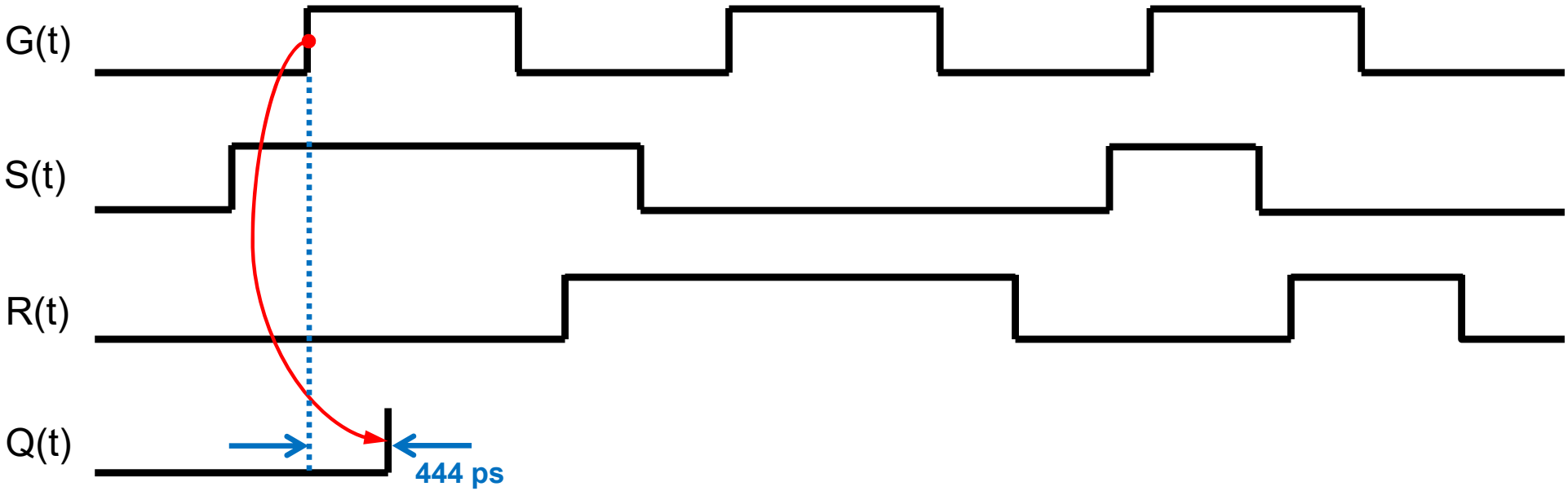
R(t)	S(t)	G(t)	Q(t+Δt)
X	X	0	Q(t)
0	0	1	Q(t)
0	1	1	1
1	0	1	0
1	1	1	prohibido



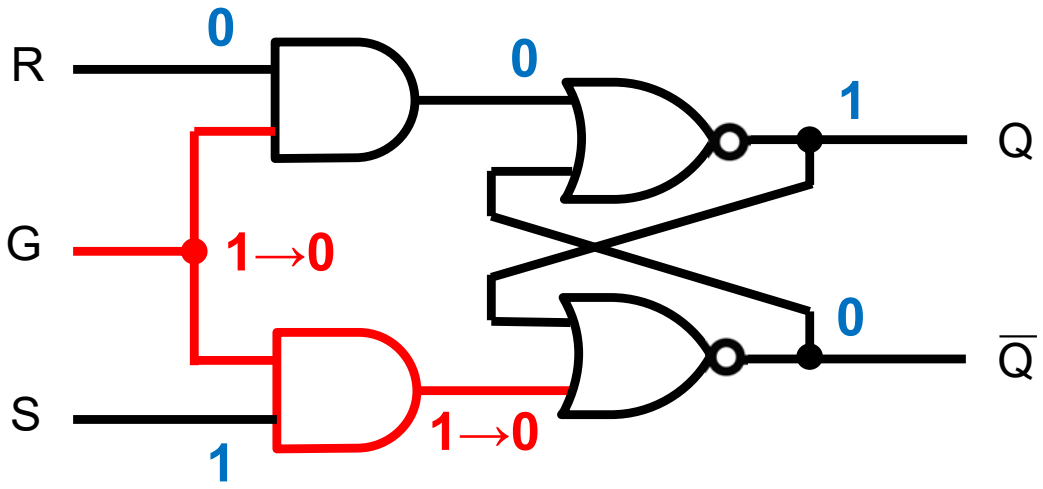
Biestable SR síncrono (por nivel)



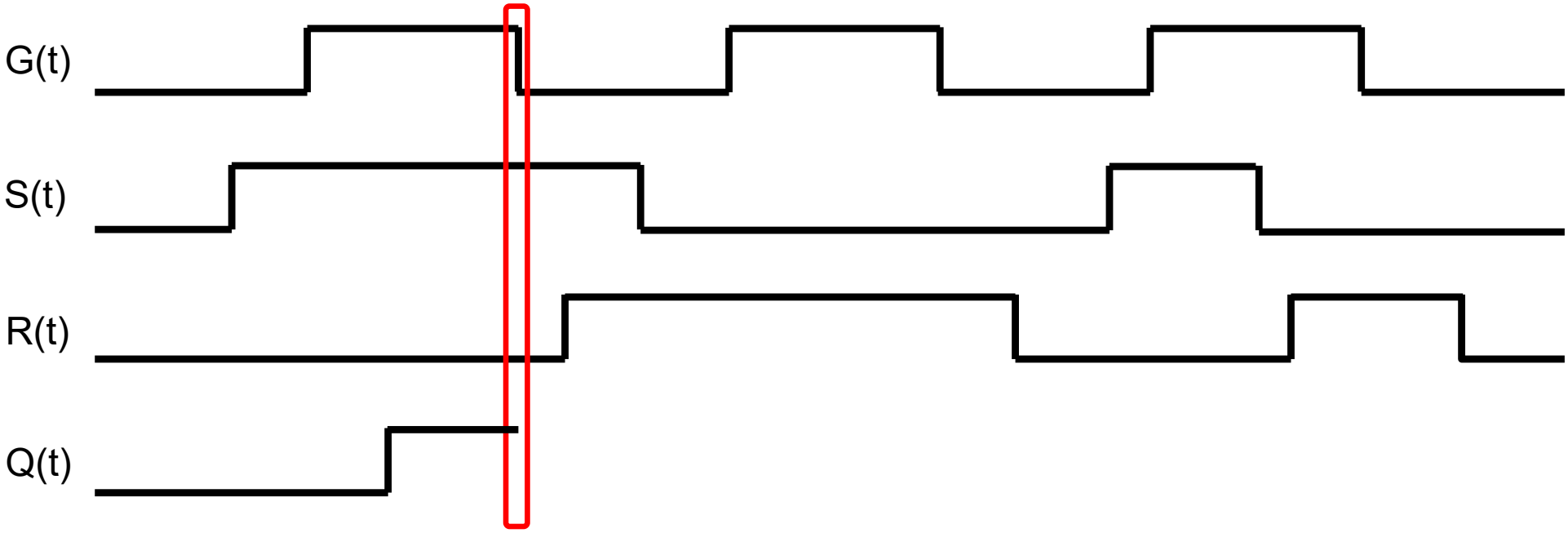
R(t)	S(t)	G(t)	Q(t+Δt)
X	X	0	Q(t)
0	0	1	Q(t)
0	1	1	1
1	0	1	0
1	1	1	prohibido



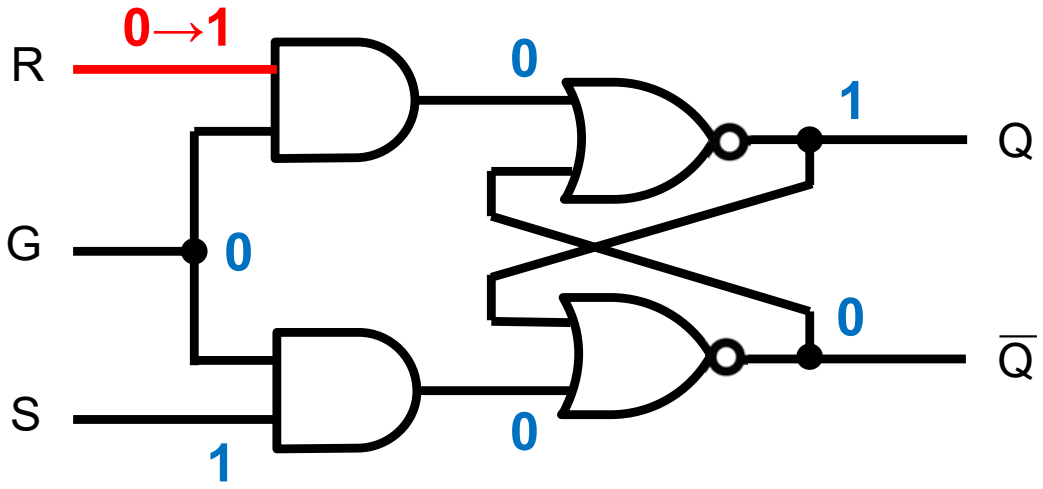
Biestable SR síncrono (por nivel)



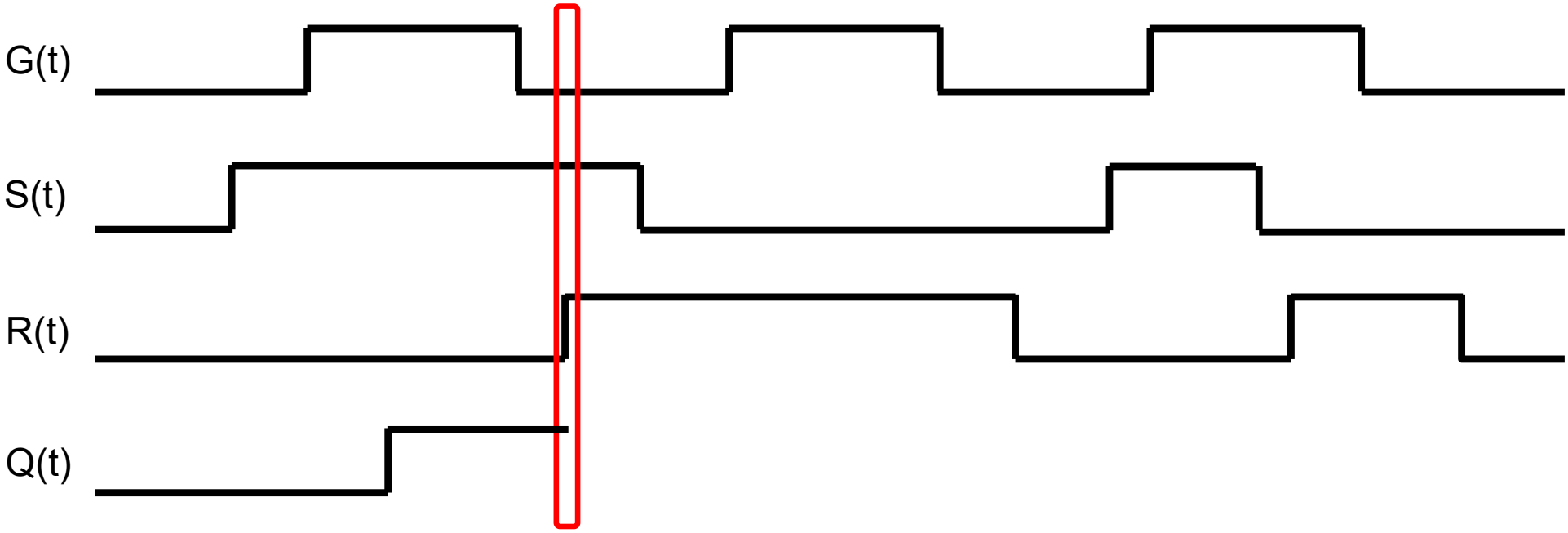
R(t)	S(t)	G(t)	Q(t+Δt)
X	X	0	Q(t)
0	0	1	Q(t)
0	1	1	1
1	0	1	0
1	1	1	prohibido



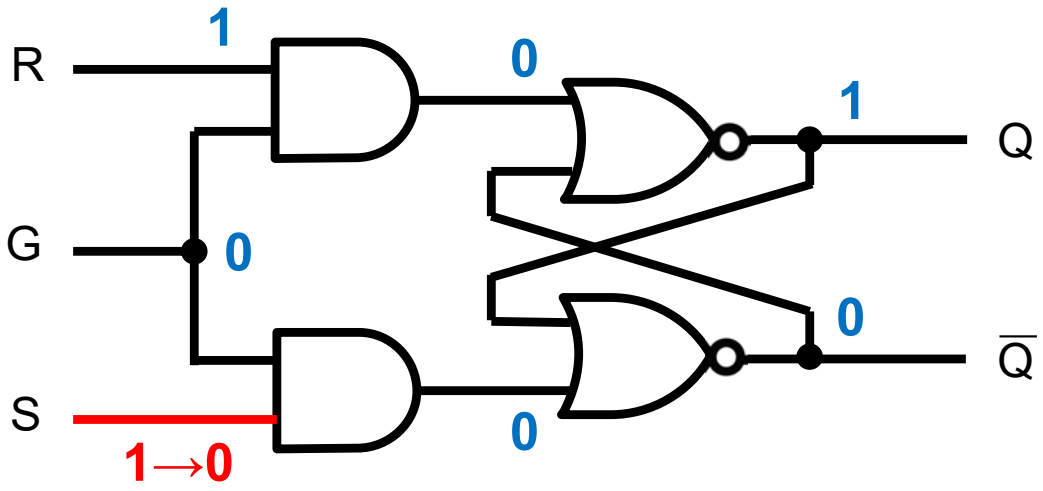
Biestable SR síncrono (por nivel)



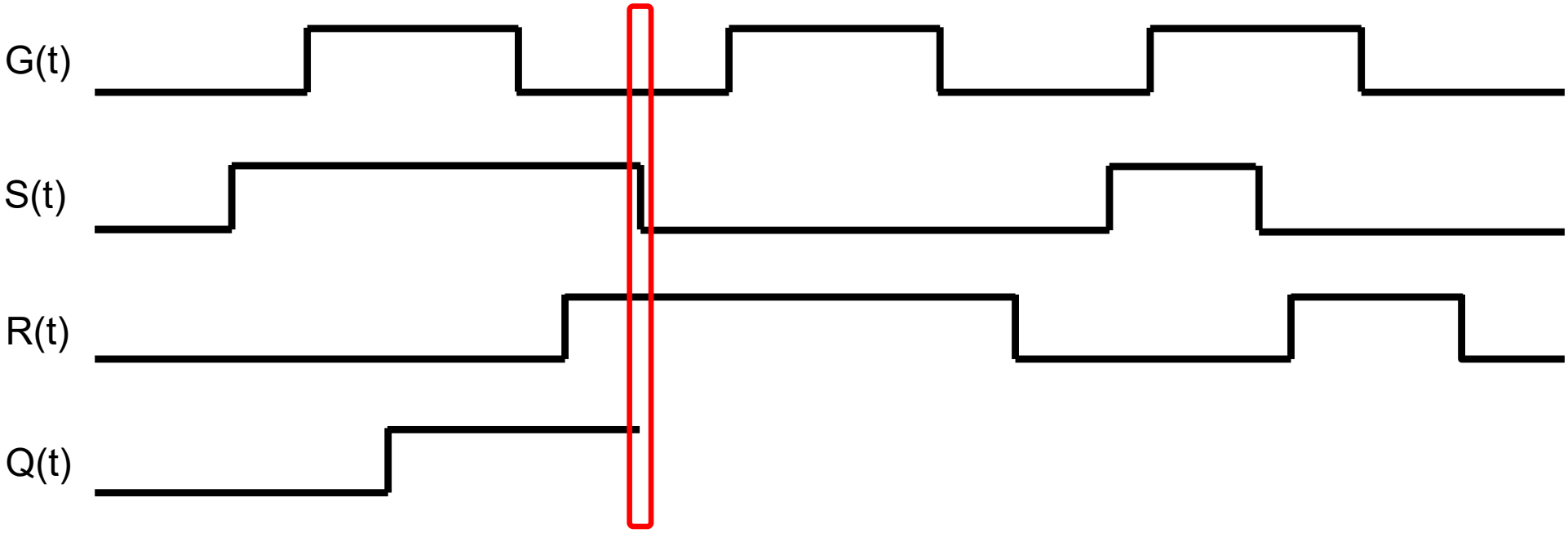
R(t)	S(t)	G(t)	Q(t+Δt)
X	X	0	Q(t)
0	0	1	Q(t)
0	1	1	1
1	0	1	0
1	1	1	prohibido



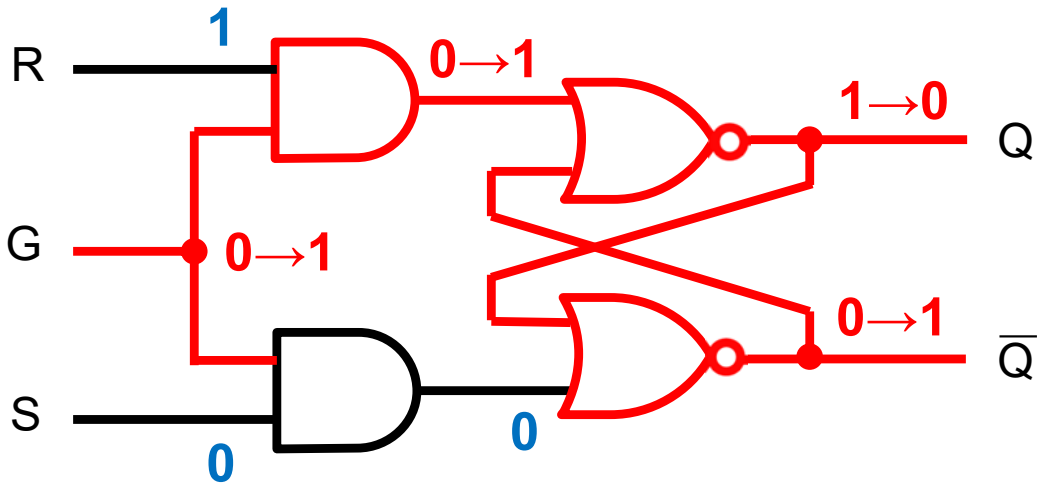
Biestable SR síncrono (por nivel)



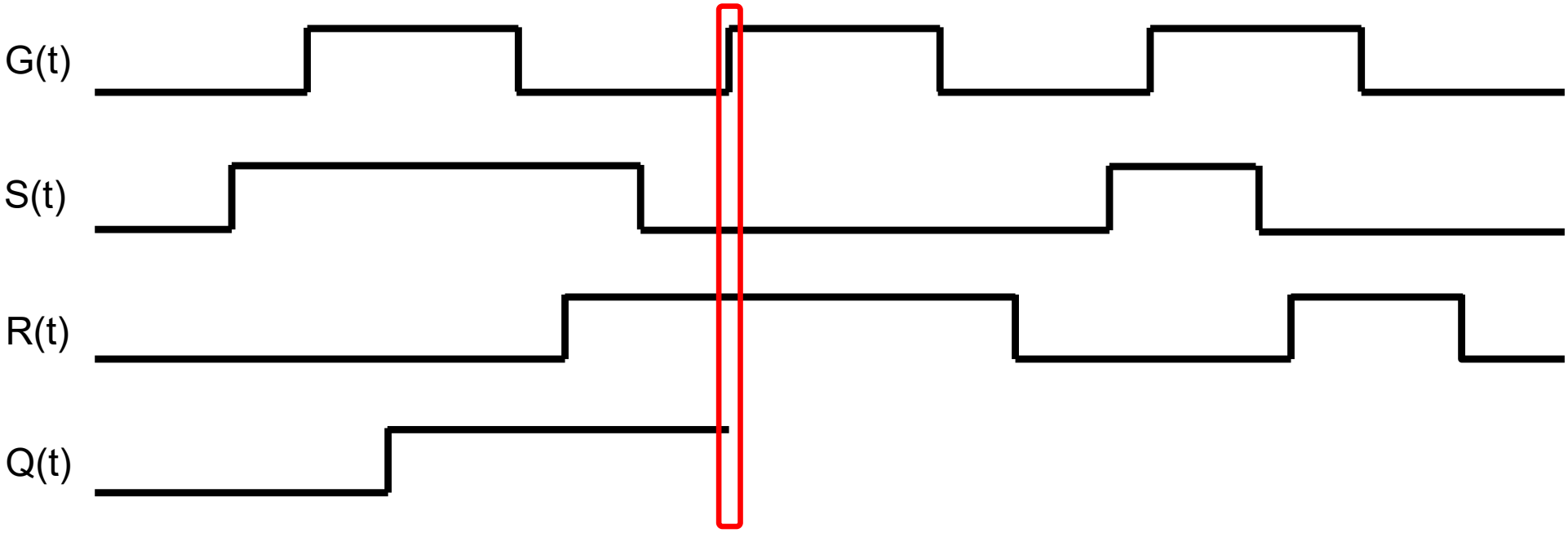
R(t)	S(t)	G(t)	Q(t+Δt)
X	X	0	Q(t)
0	0	1	Q(t)
0	1	1	1
1	0	1	0
1	1	1	prohibido



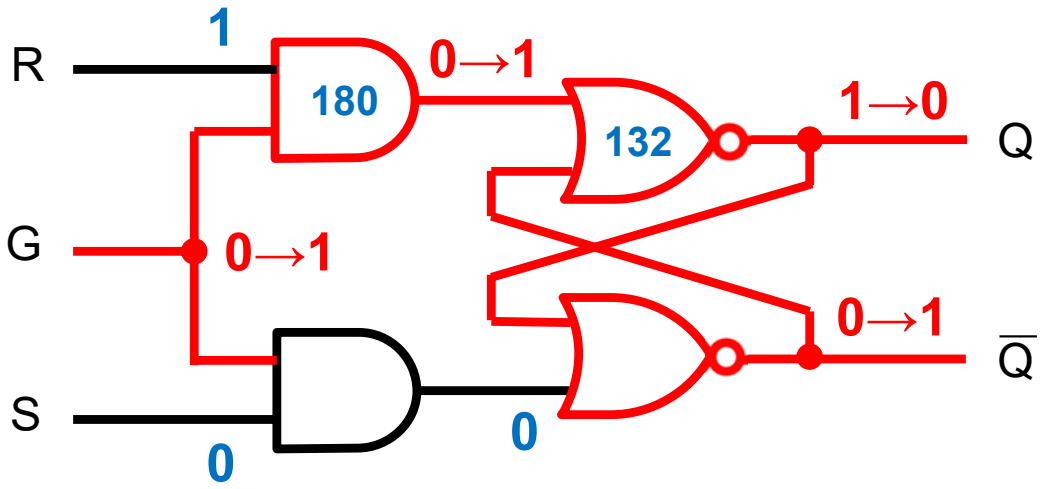
Biestable SR síncrono (por nivel)



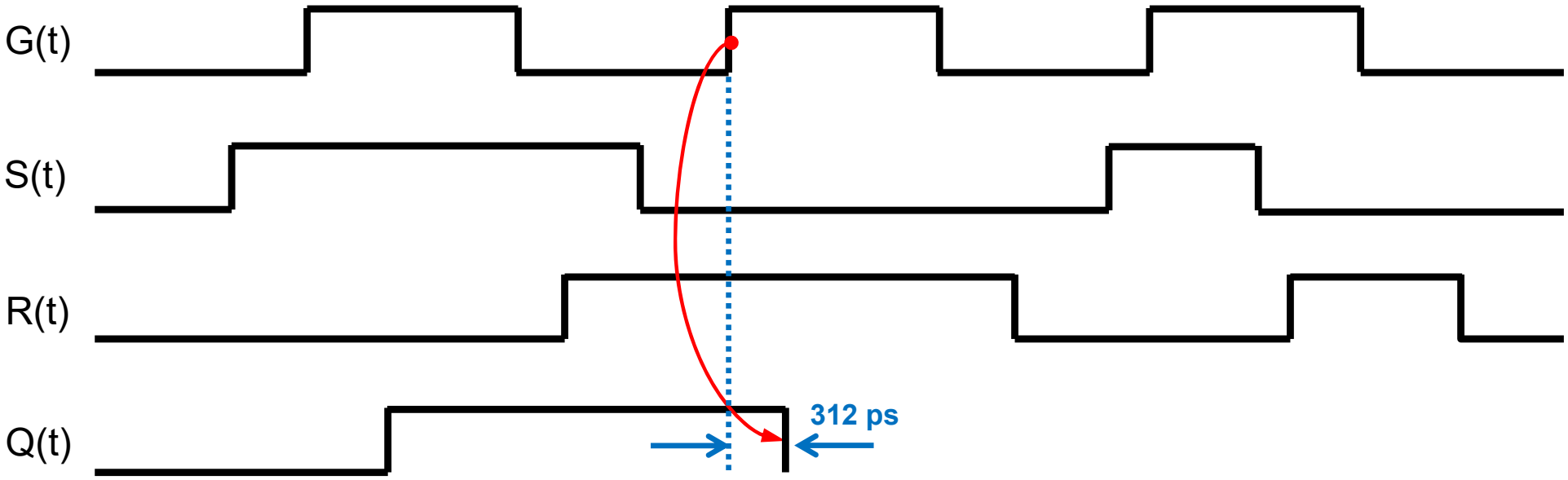
R(t)	S(t)	G(t)	Q(t+Δt)
X	X	0	Q(t)
0	0	1	Q(t)
0	1	1	1
1	0	1	0
1	1	1	prohibido



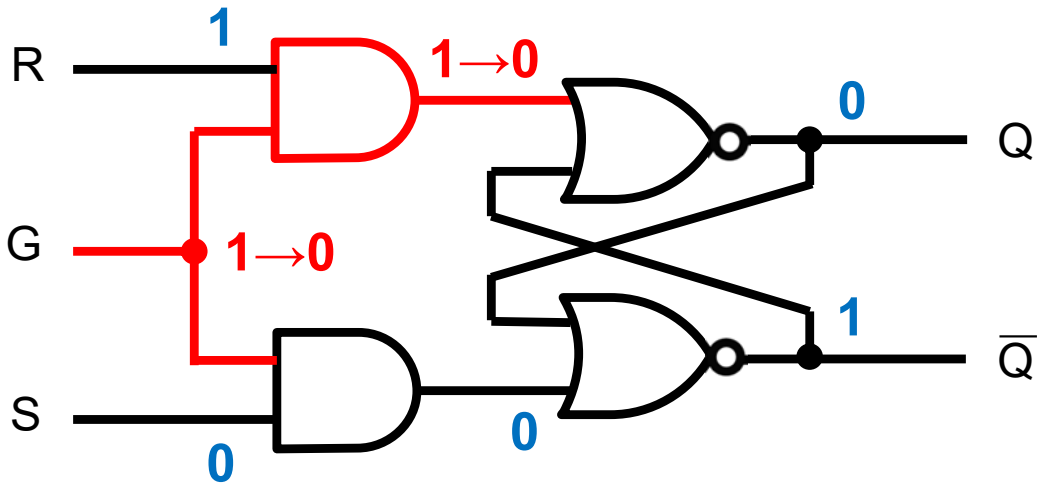
Biestable SR síncrono (por nivel)



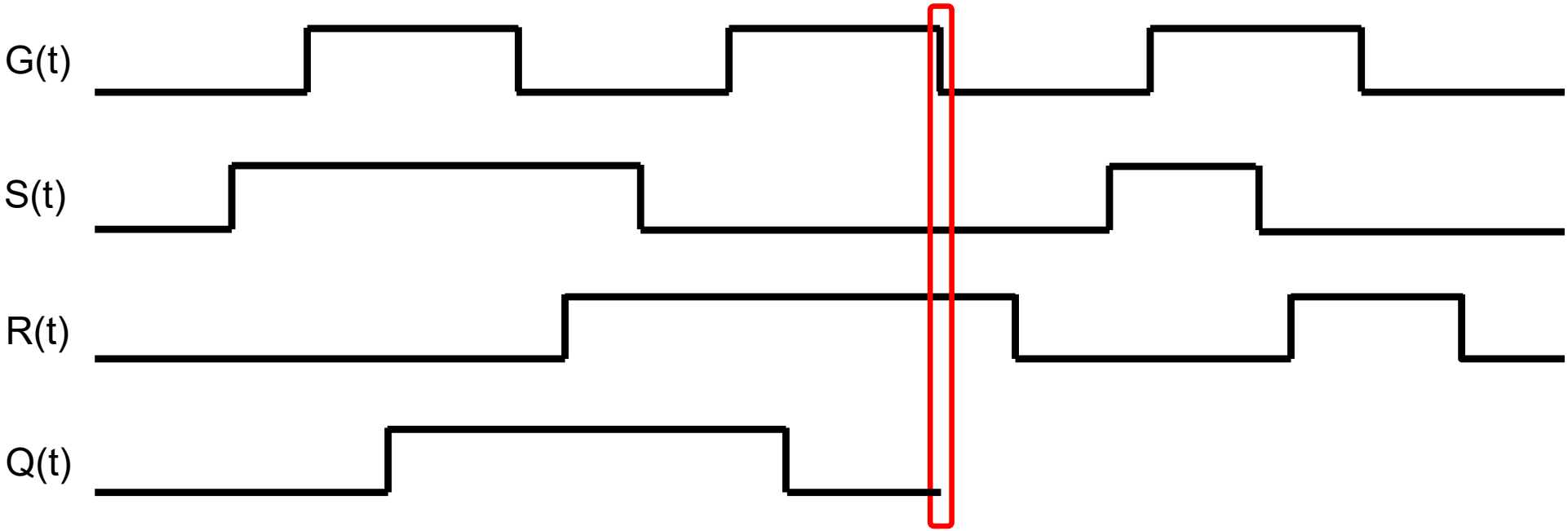
R(t)	S(t)	G(t)	Q(t+Δt)
X	X	0	Q(t)
0	0	1	Q(t)
0	1	1	1
1	0	1	0
1	1	1	prohibido



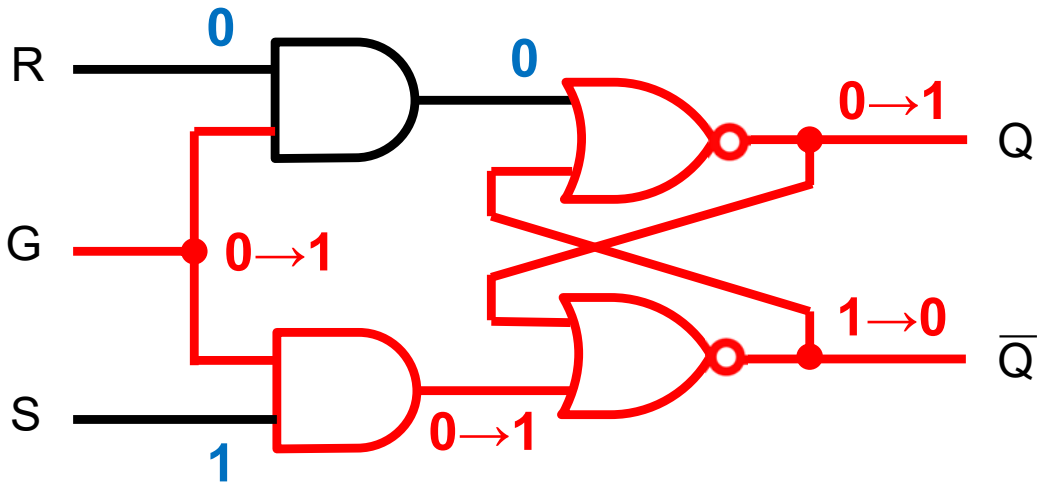
Biestable SR síncrono (por nivel)



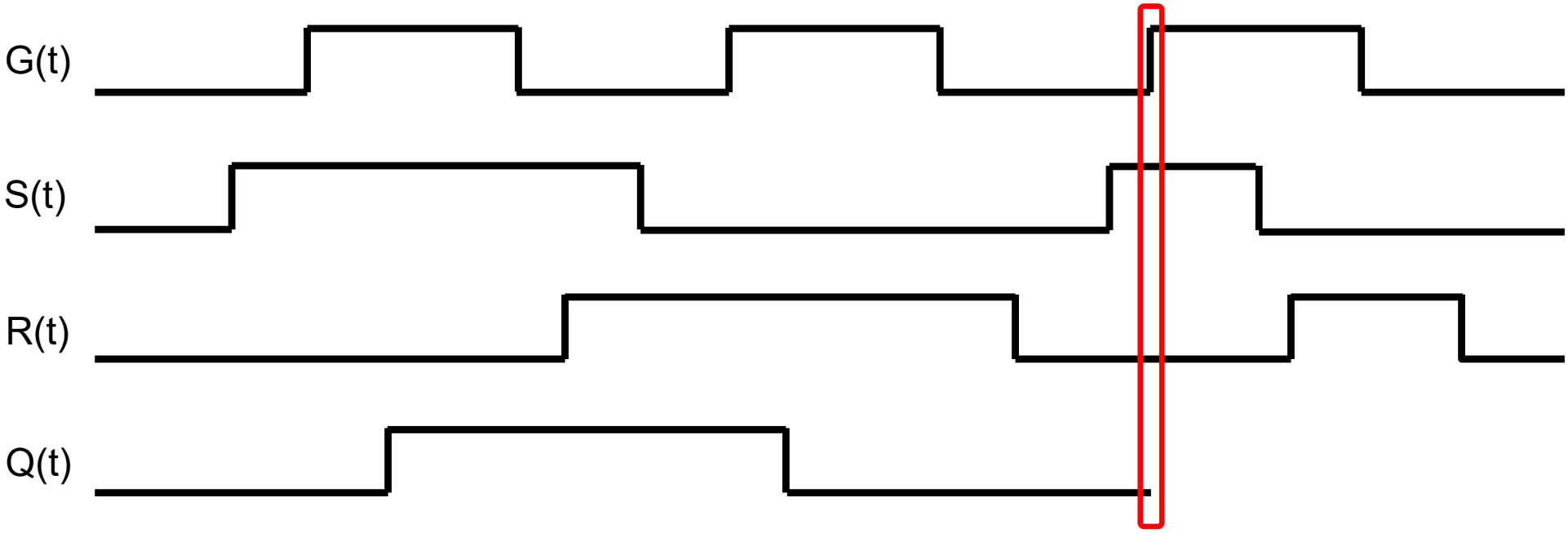
R(t)	S(t)	G(t)	Q(t+Δt)
X	X	0	Q(t)
0	0	1	Q(t)
0	1	1	1
1	0	1	0
1	1	1	prohibido



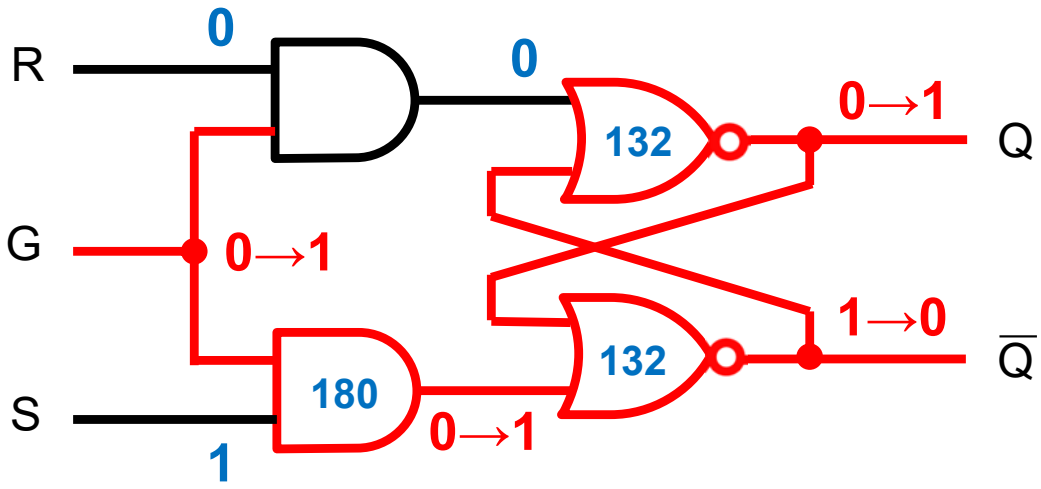
Biestable SR síncrono (por nivel)



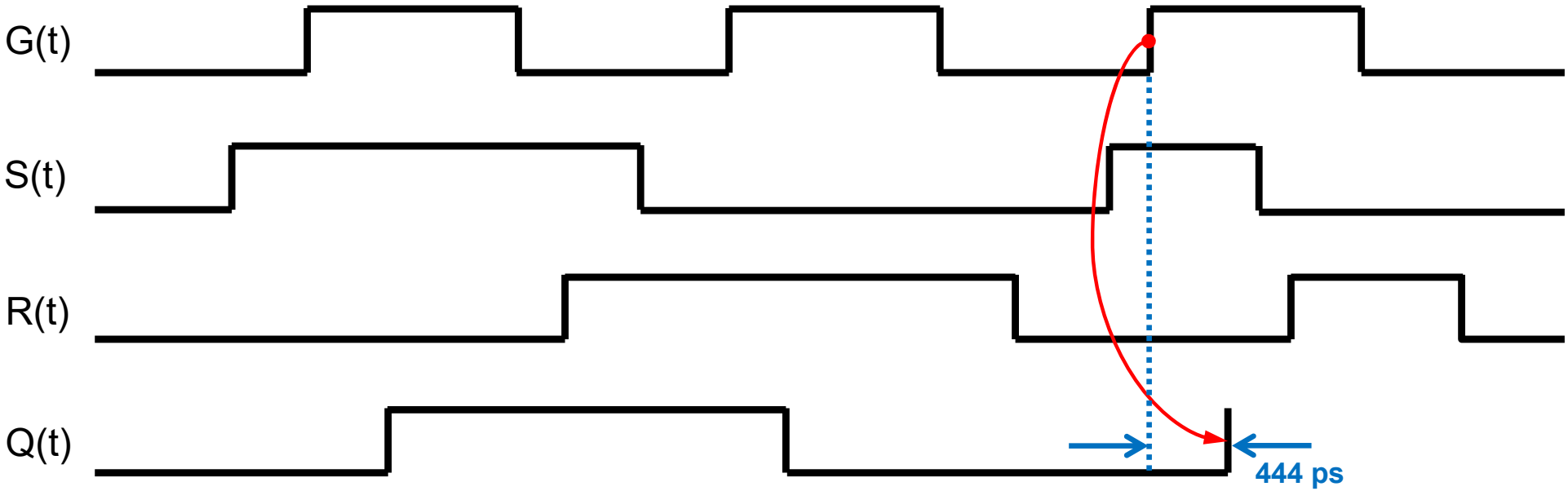
R(t)	S(t)	G(t)	Q(t+Δt)
X	X	0	Q(t)
0	0	1	Q(t)
0	1	1	1
1	0	1	0
1	1	1	prohibido



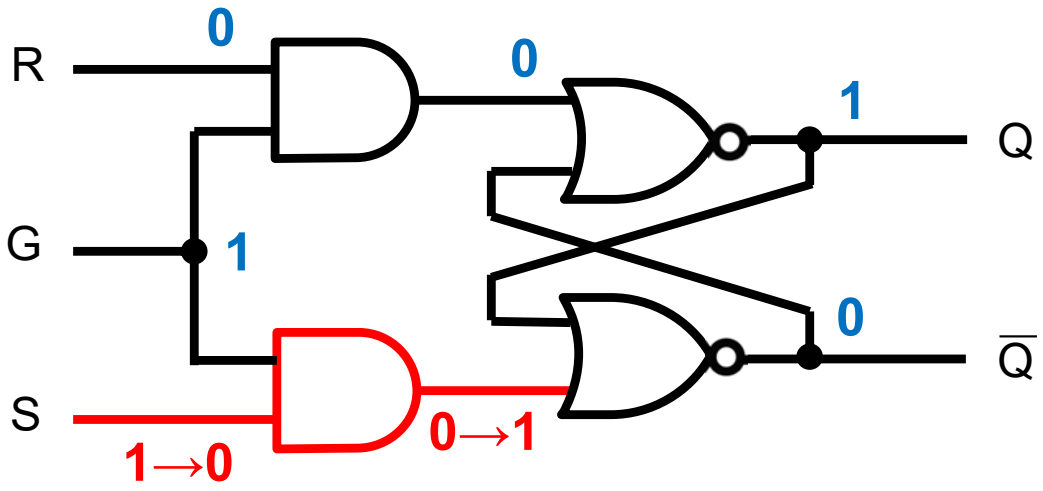
Biestable SR síncrono (por nivel)



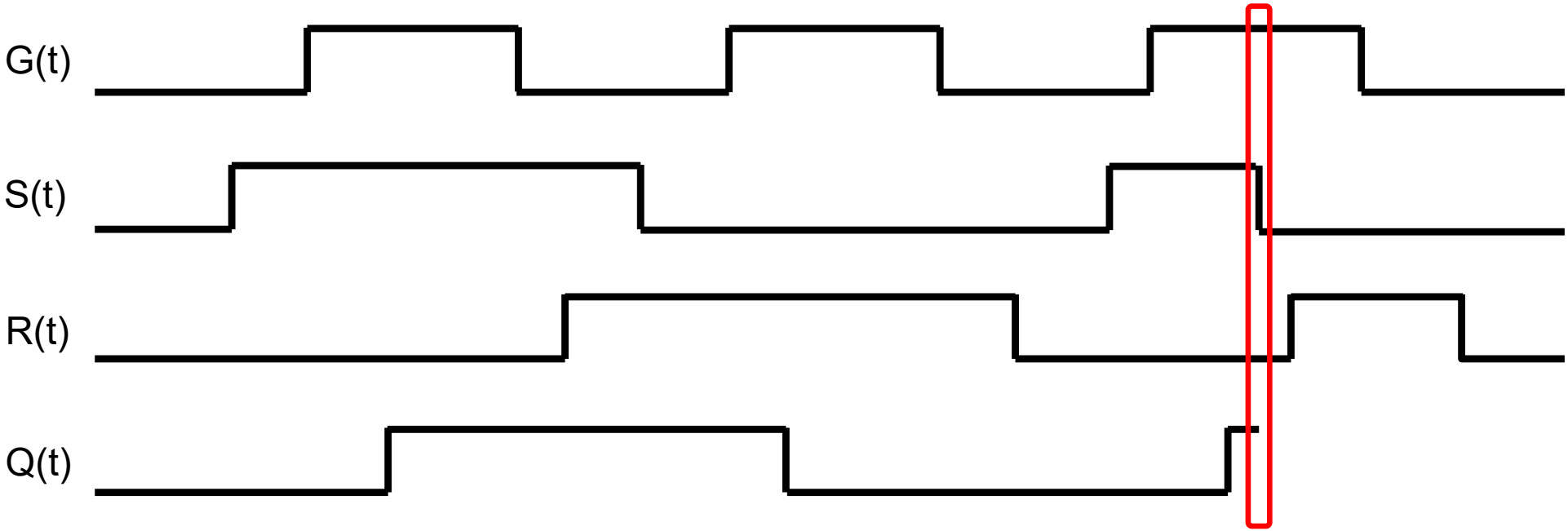
R(t)	S(t)	G(t)	Q(t+Δt)
X	X	0	Q(t)
0	0	1	Q(t)
0	1	1	1
1	0	1	0
1	1	1	prohibido



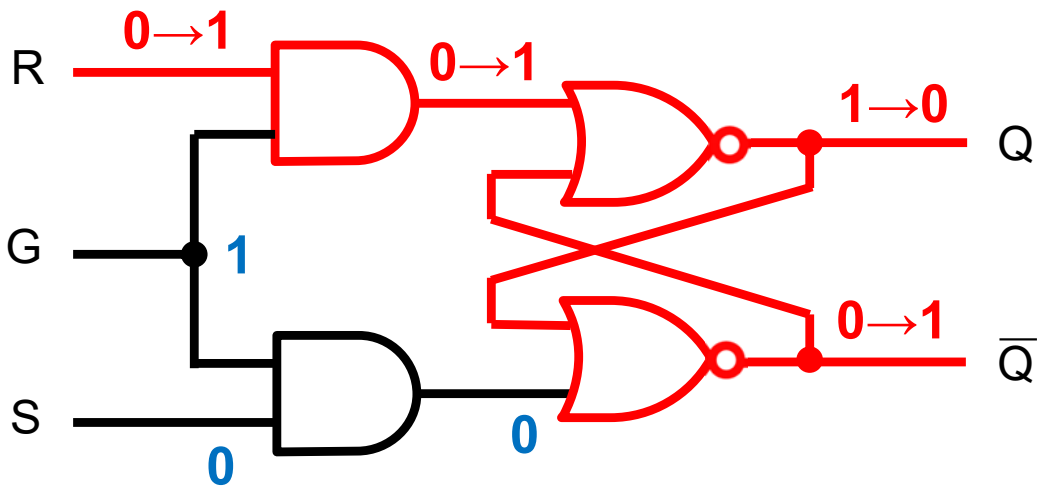
Biestable SR síncrono (por nivel)



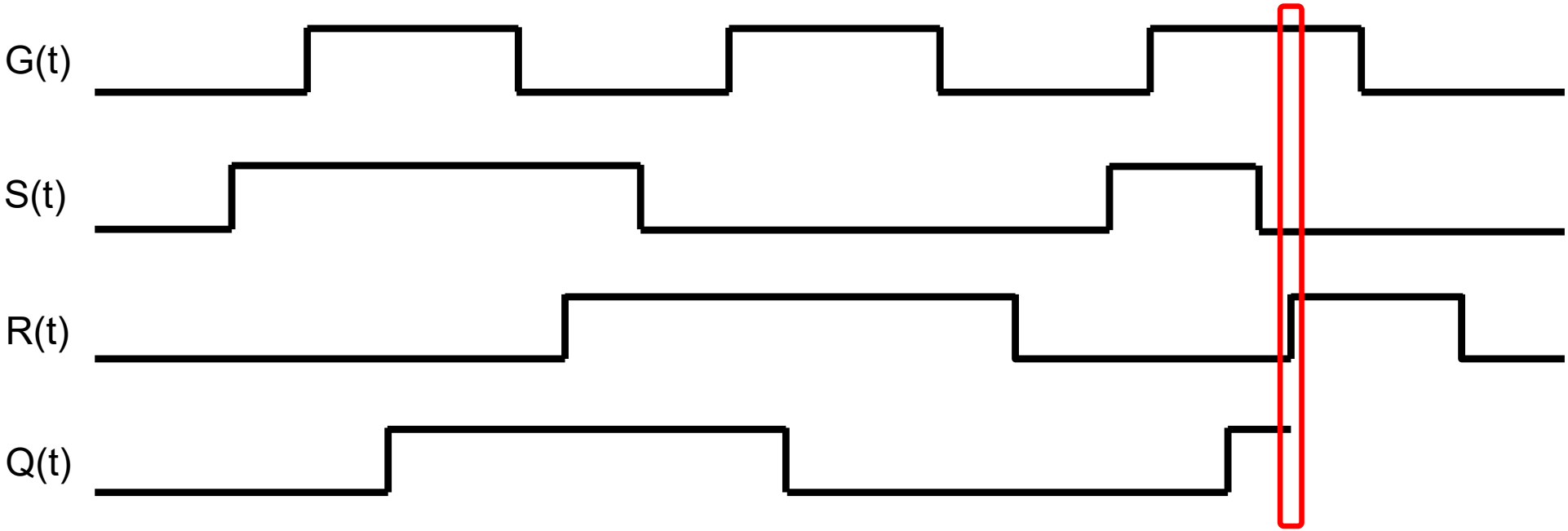
R(t)	S(t)	G(t)	Q(t+Δt)
X	X	0	Q(t)
0	0	1	Q(t)
0	1	1	1
1	0	1	0
1	1	1	prohibido



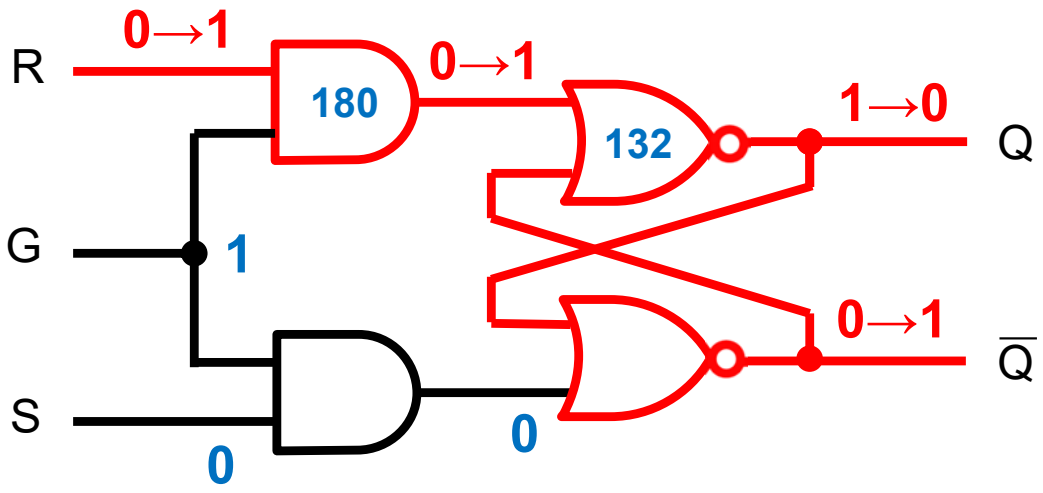
Biestable SR síncrono (por nivel)



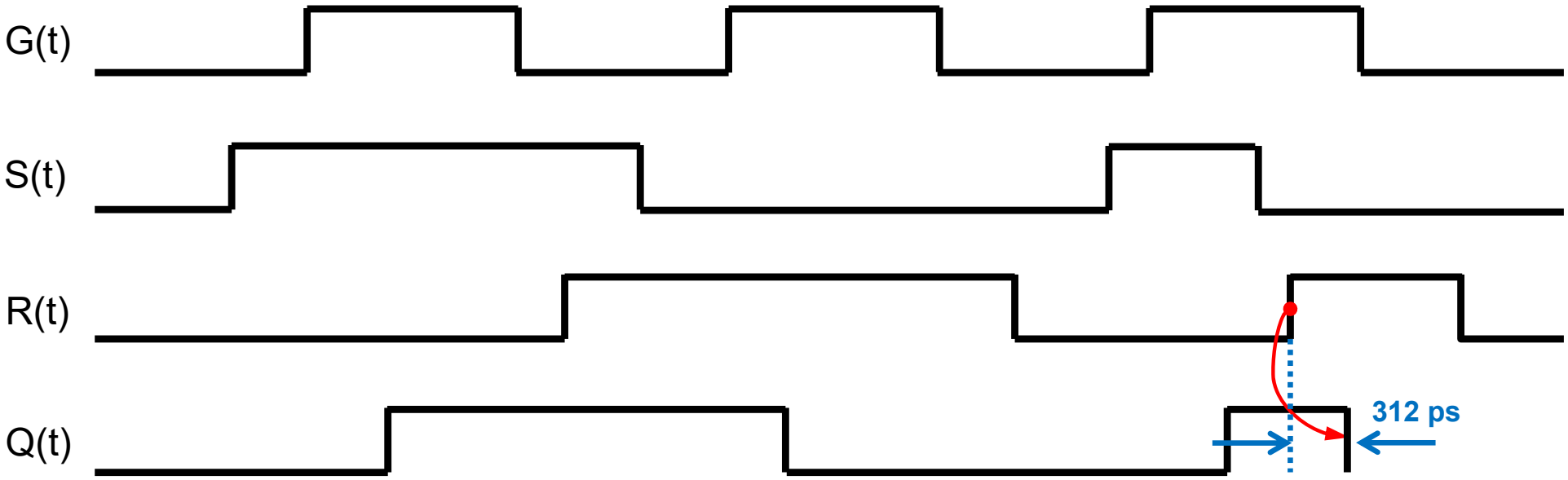
R(t)	S(t)	G(t)	Q(t+Δt)
X	X	0	Q(t)
0	0	1	Q(t)
0	1	1	1
1	0	1	0
1	1	1	prohibido



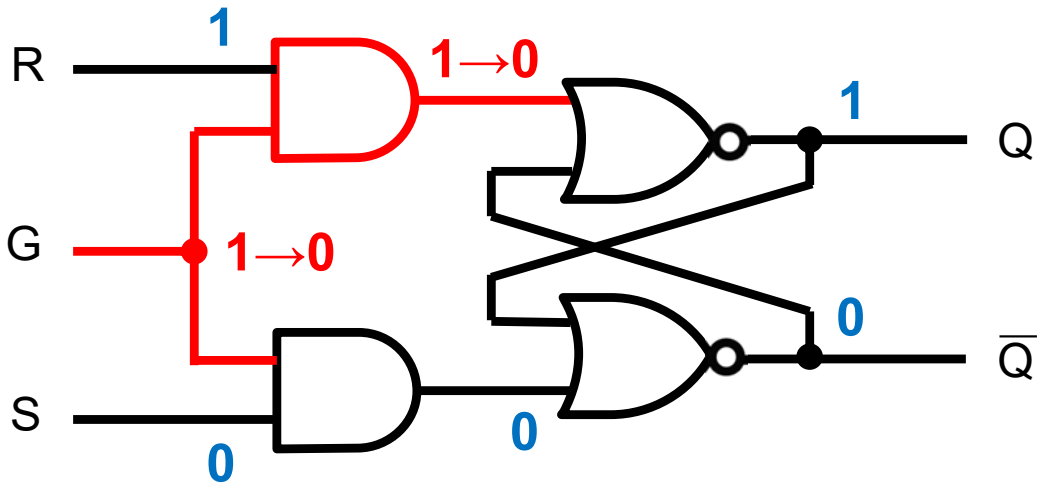
Biestable SR síncrono (por nivel)



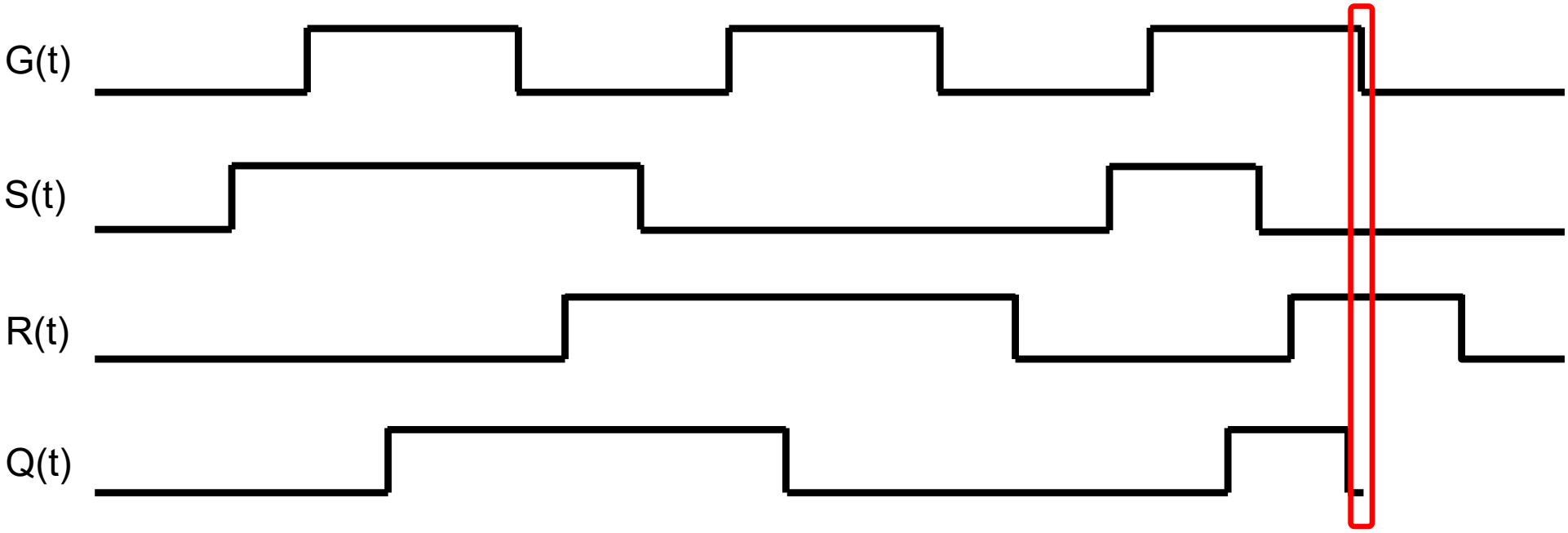
R(t)	S(t)	G(t)	Q(t+Δt)
X	X	0	Q(t)
0	0	1	Q(t)
0	1	1	1
1	0	1	0
1	1	1	prohibido



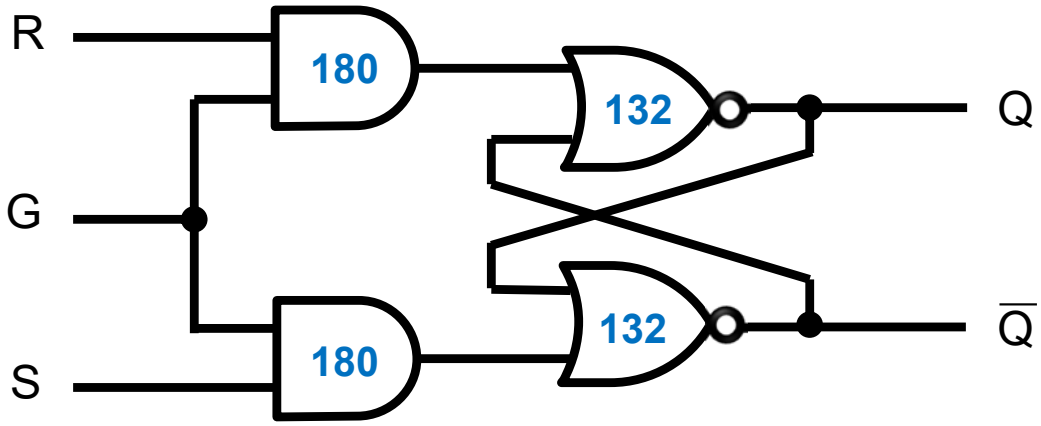
Biestable SR síncrono (por nivel)



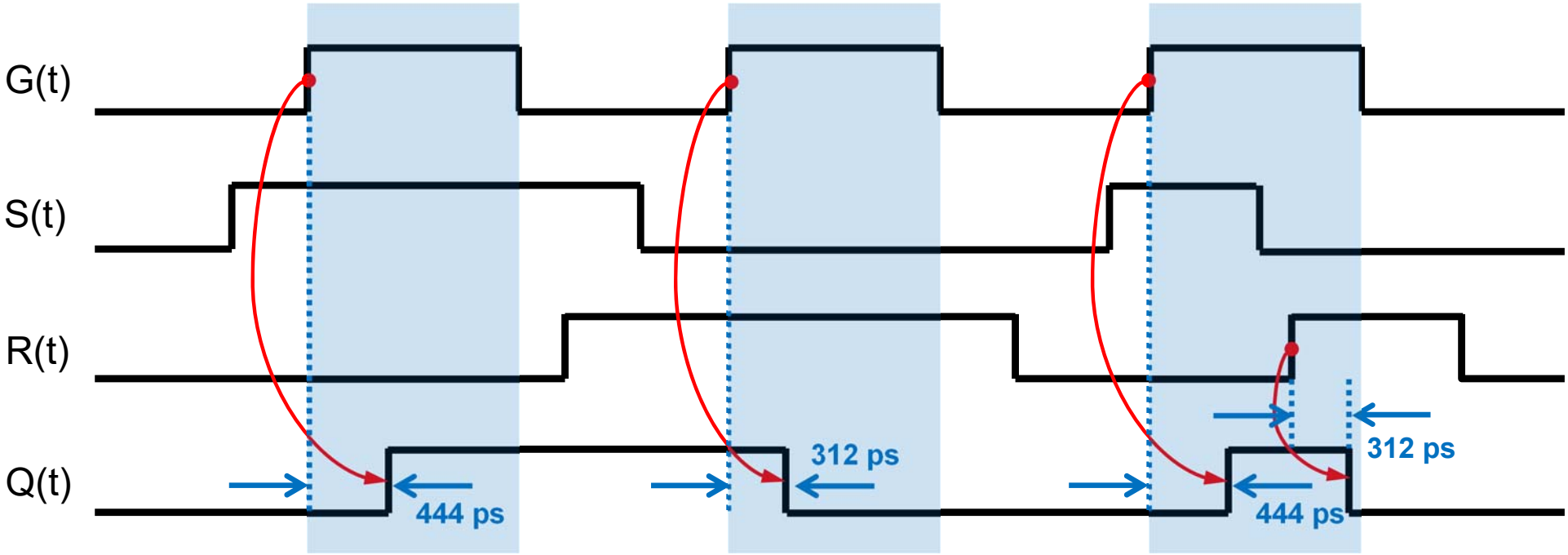
R(t)	S(t)	G(t)	Q(t+Δt)
X	X	0	Q(t)
0	0	1	Q(t)
0	1	1	1
1	0	1	0
1	1	1	prohibido



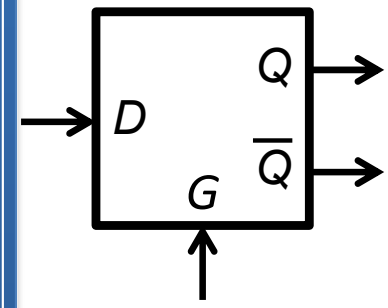
Biestable SR síncrono (por nivel)



R(t)	S(t)	G(t)	Q(t+Δt)
X	X	0	Q(t)
0	0	1	Q(t)
0	1	1	1
1	0	1	0
1	1	1	prohibido

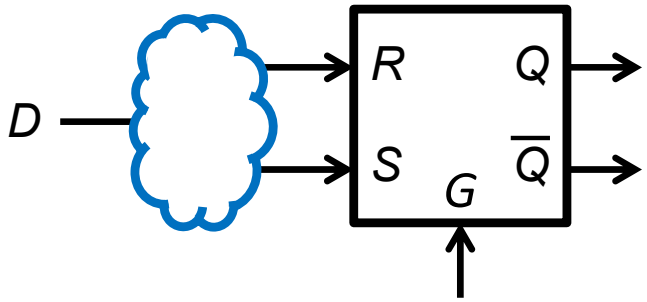


Biestable D síncrono (por nivel)



G(t)	D(t)	Q(t+Δt)
0	X	Q(t)
1	0	0
1	1	1

Biestable D síncrono (por nivel)

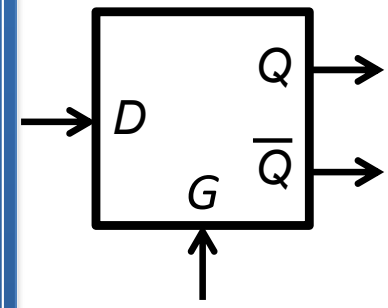


D	R
0	1
1	0

$R = \overline{D}$

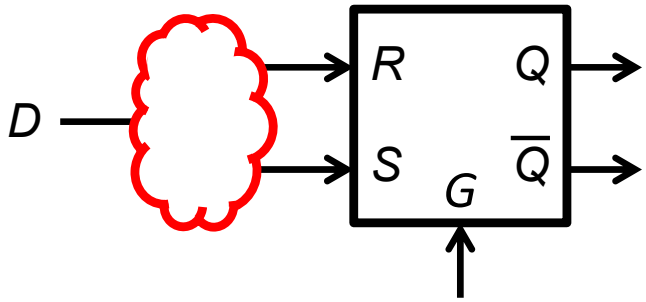
D	S
0	0
1	1

$S = D$



G(t)	D(t)	Q(t+Δt)
0	X	Q(t)
1	0	0
1	1	1

Biestable D síncrono (por nivel)

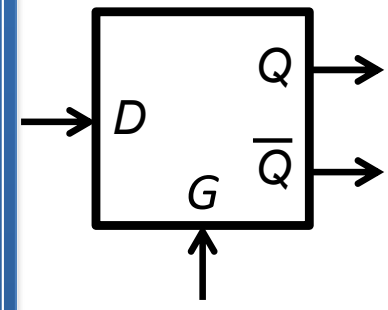


D	R
0	1
1	0

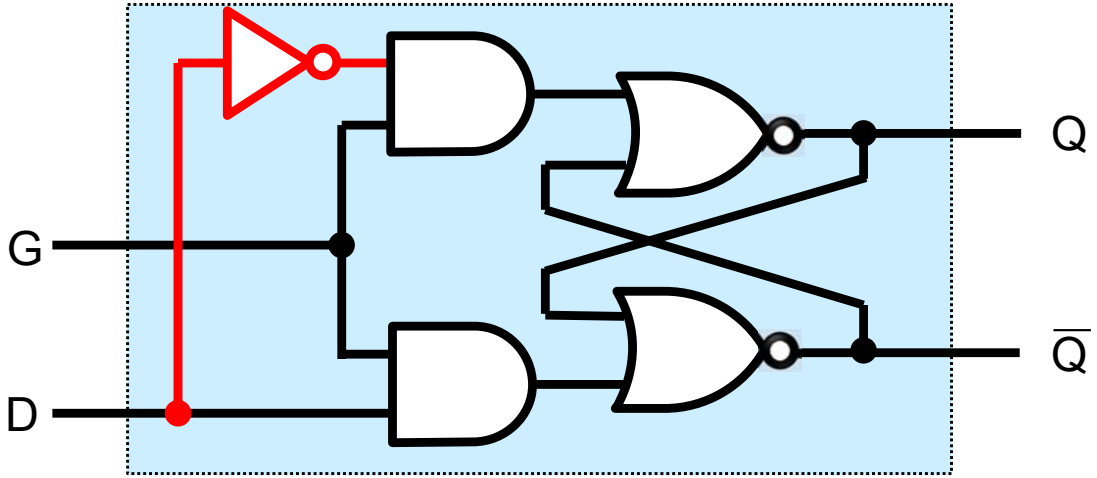
$R = \overline{D}$

D	S
0	0
1	1

$S = D$

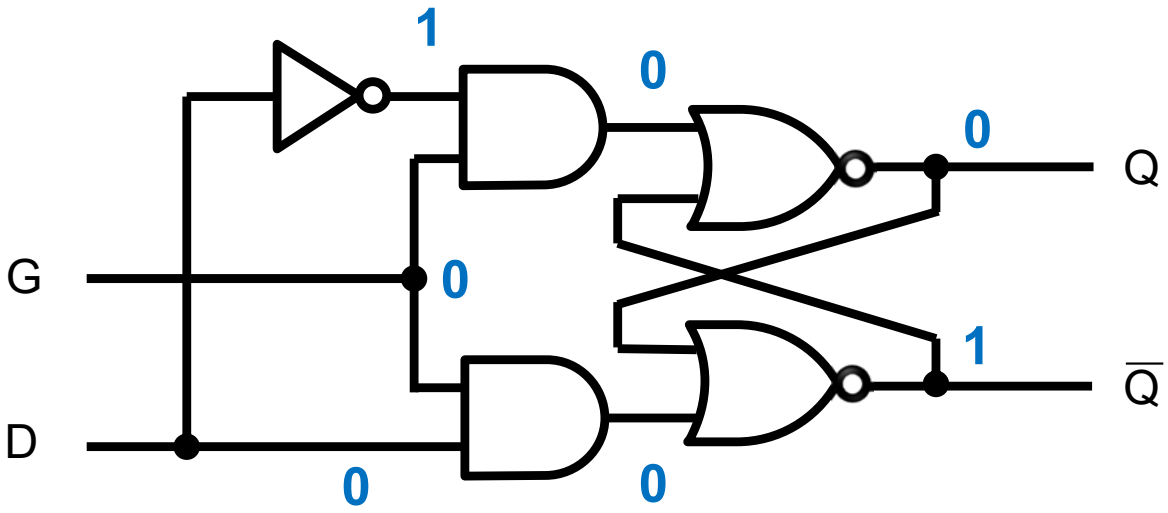


G(t)	D(t)	Q(t+Δt)
0	X	Q(t)
1	0	0
1	1	1

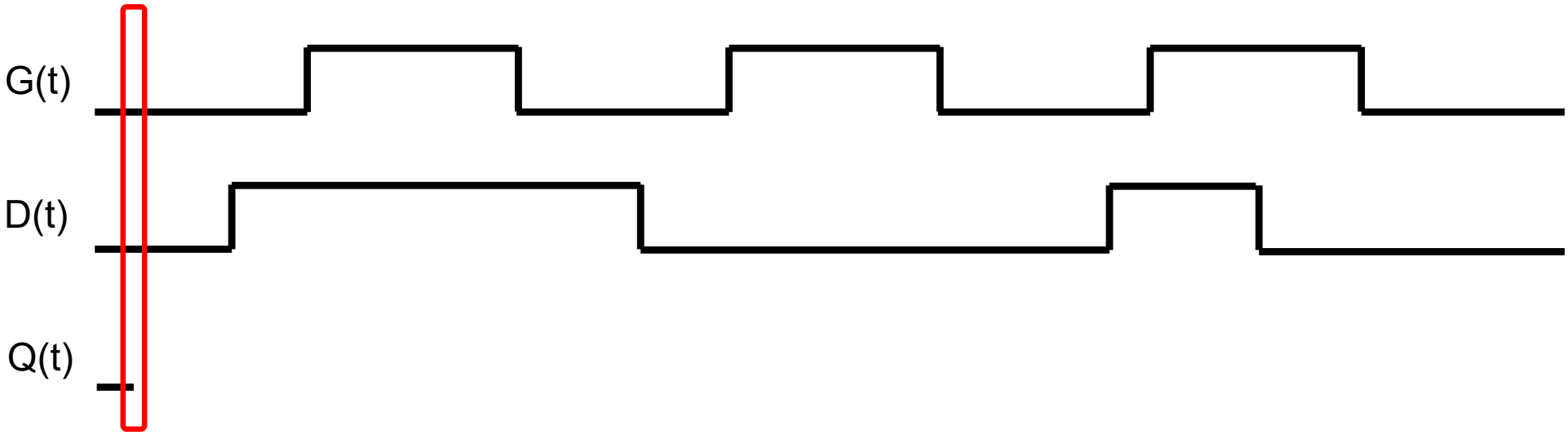


Biestable D síncrono disparado por nivel
(Latch D)

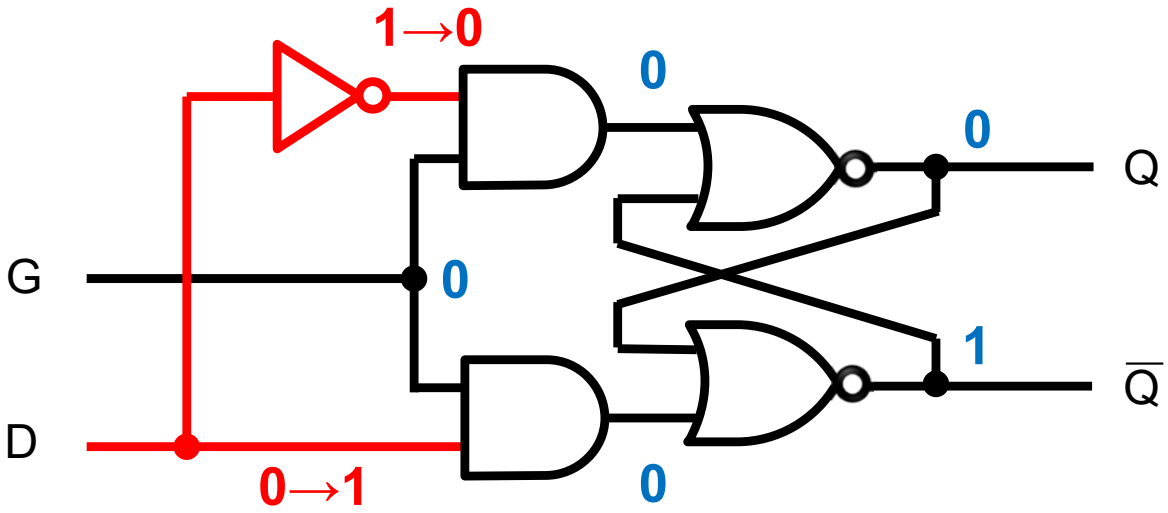
Biestable D síncrono (por nivel)



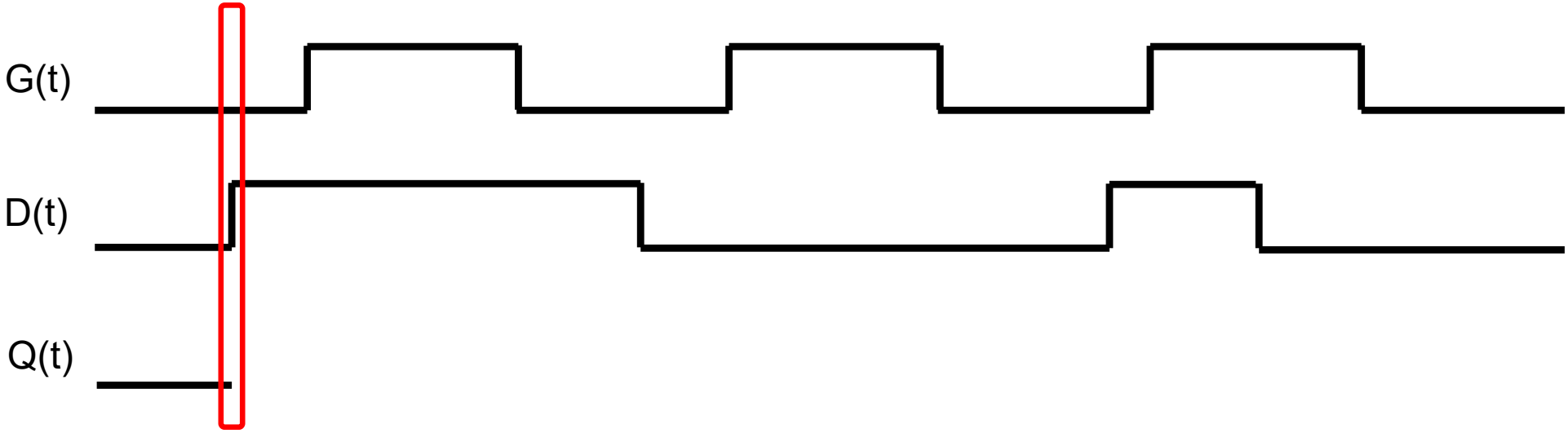
G(t)	D(t)	Q(t+Δt)
0	X	Q(t)
1	0	0
1	1	1



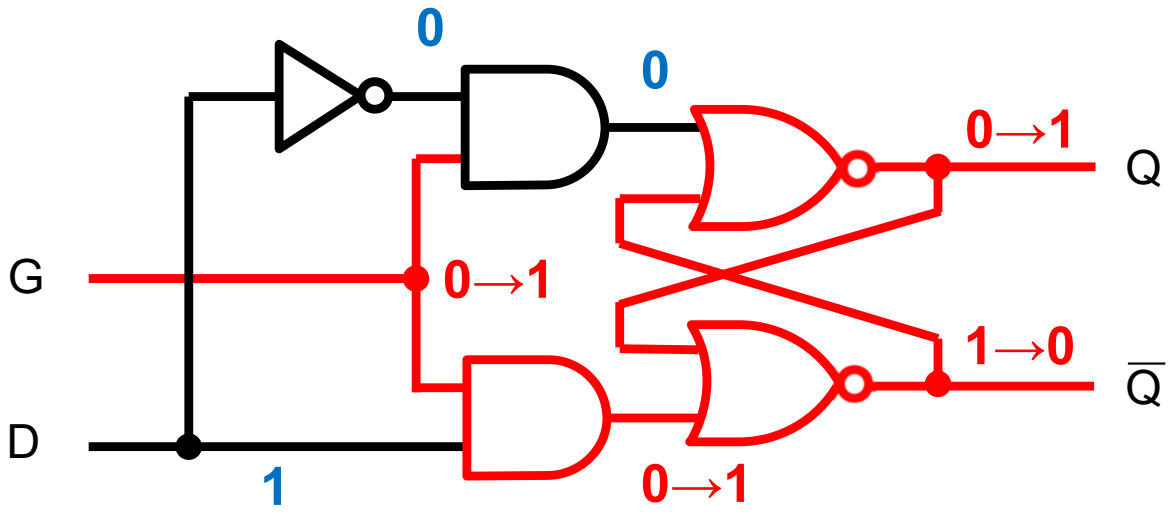
Biestable D síncrono (por nivel)



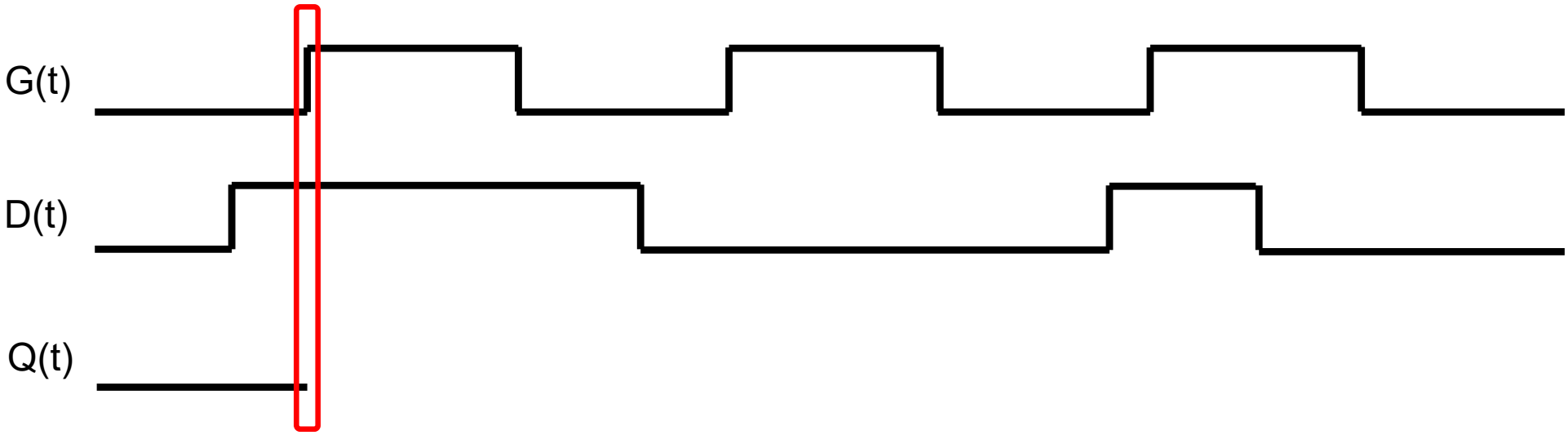
G(t)	D(t)	Q(t+Δt)
0	X	Q(t)
1	0	0
1	1	1



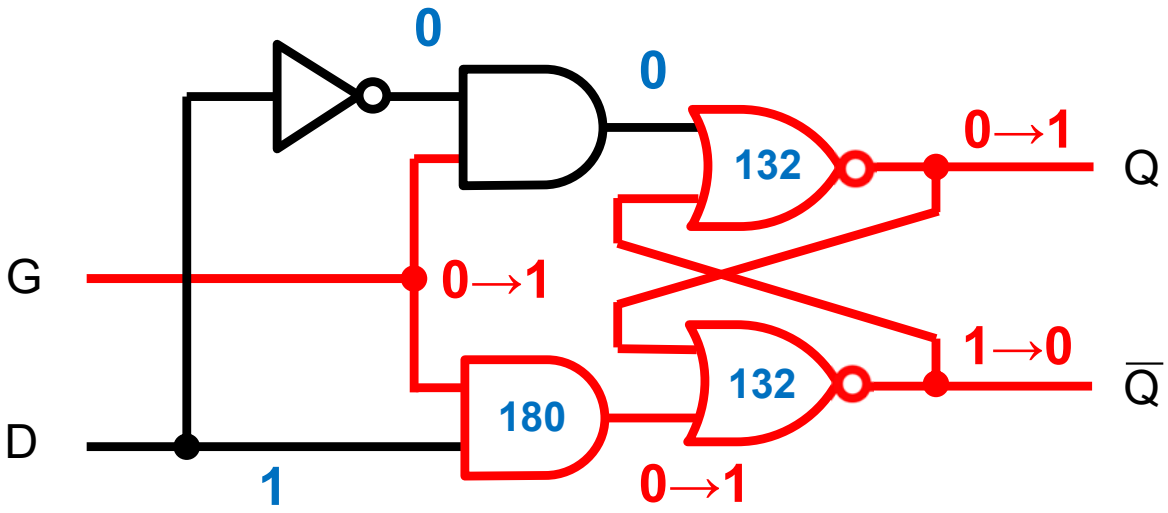
Biestable D síncrono (por nivel)



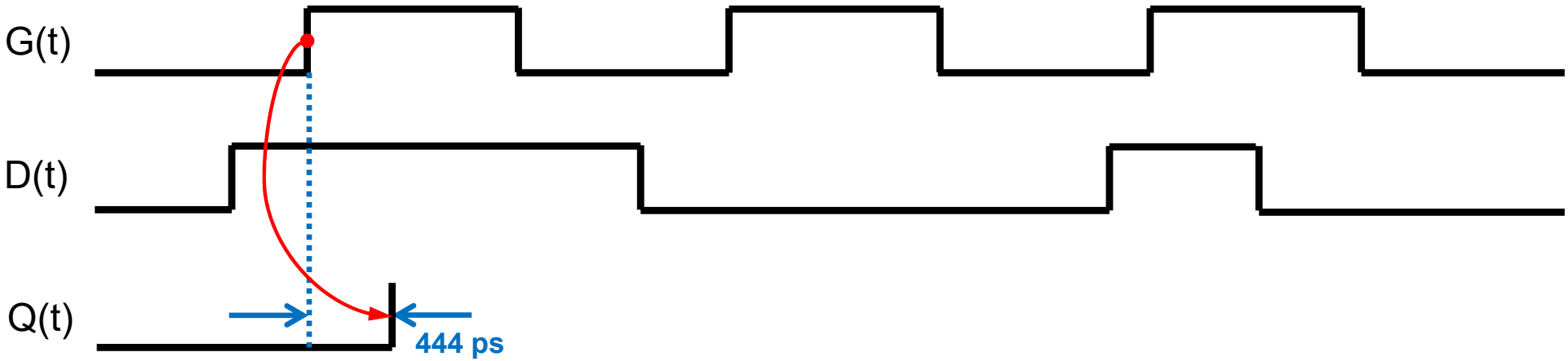
G(t)	D(t)	Q(t+Δt)
0	X	Q(t)
1	0	0
1	1	1



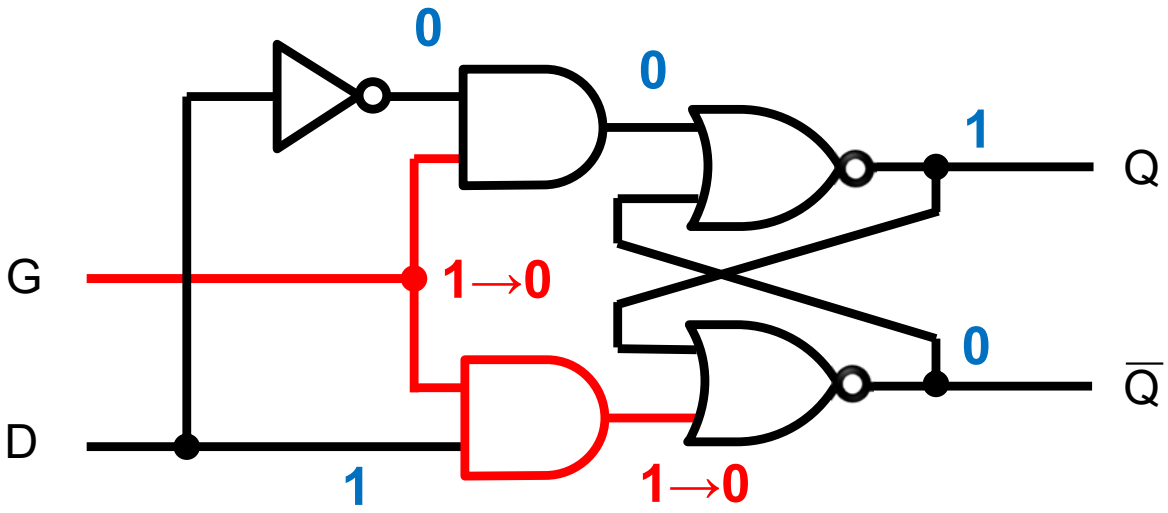
Biestable D síncrono (por nivel)



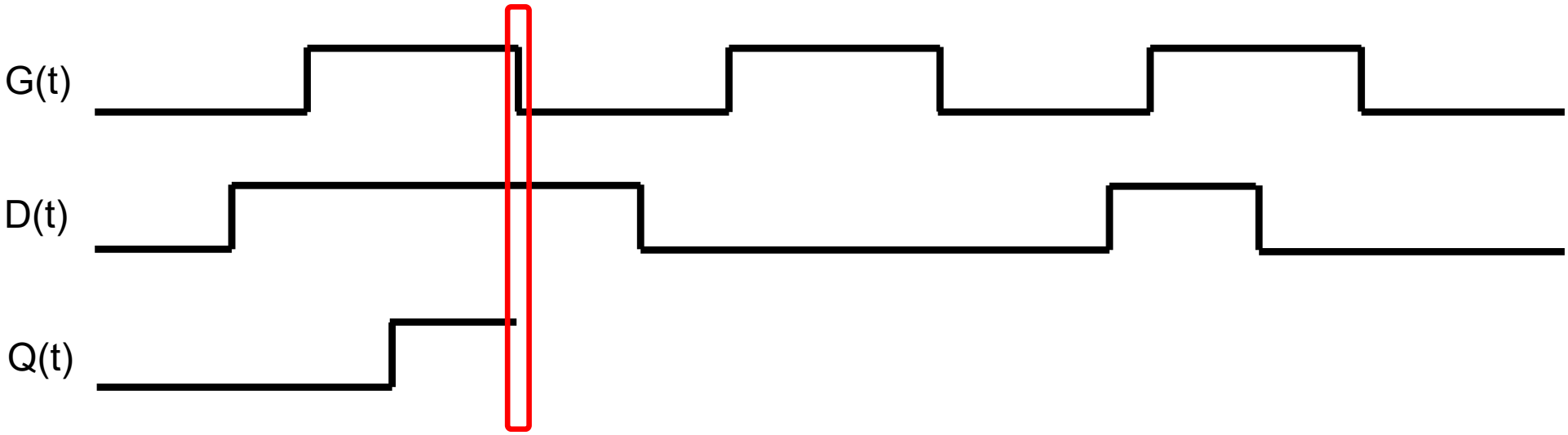
G(t)	D(t)	Q(t+Δt)
0	X	Q(t)
1	0	0
1	1	1



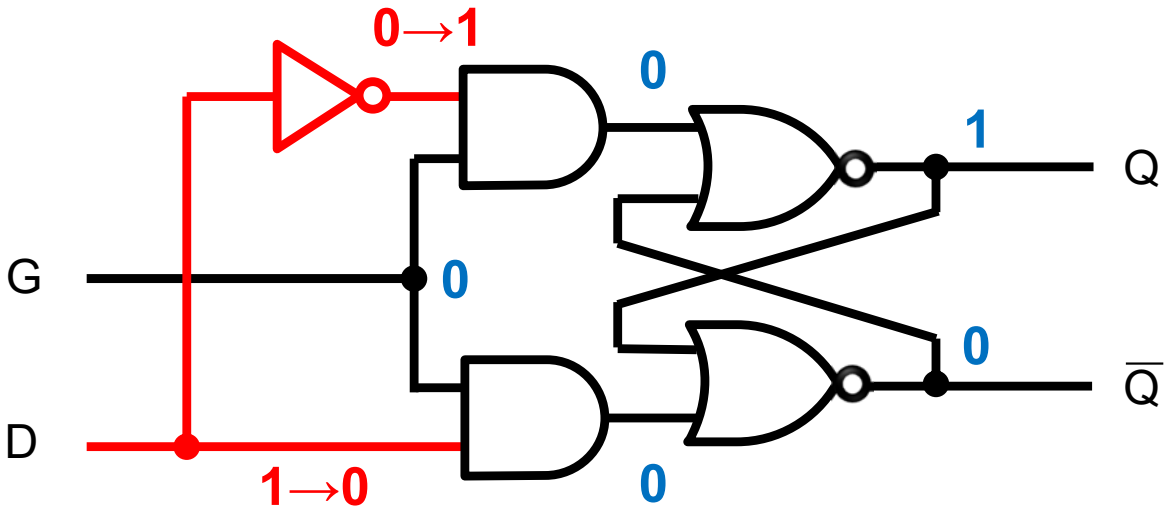
Biestable D síncrono (por nivel)



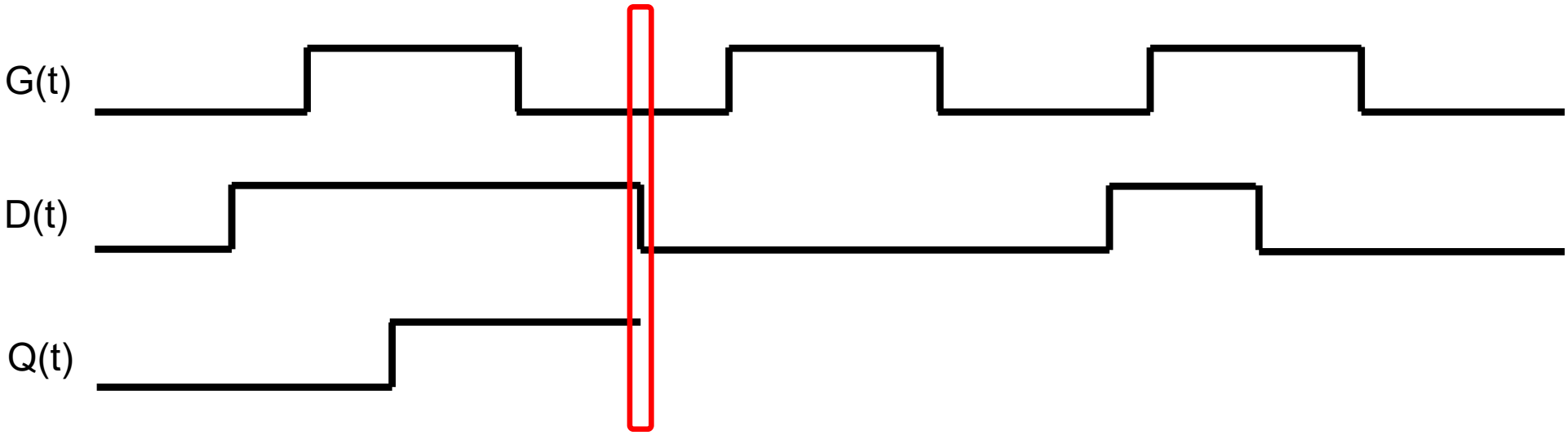
G(t)	D(t)	Q(t+Δt)
0	X	Q(t)
1	0	0
1	1	1



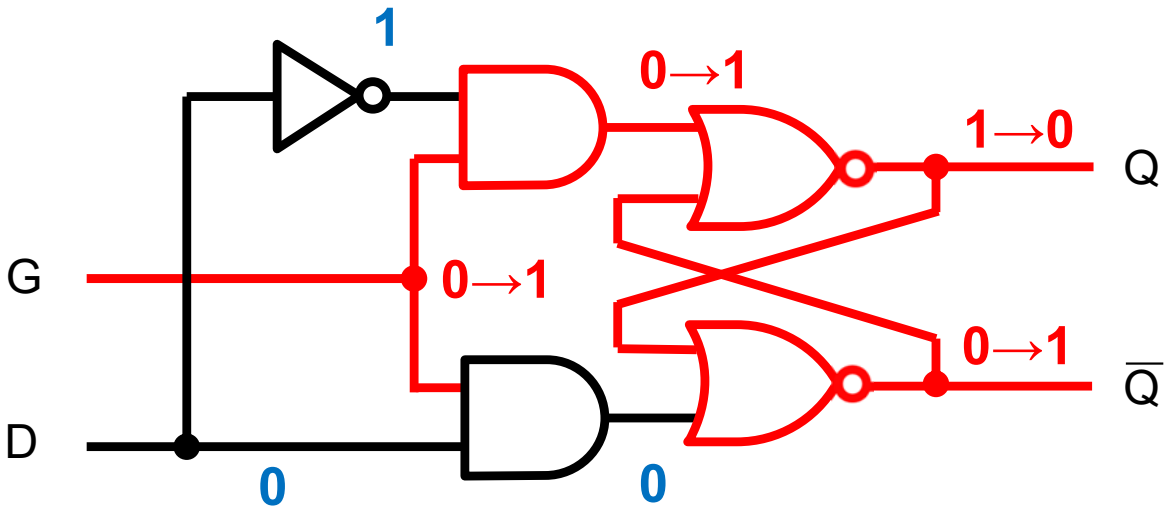
Biestable D síncrono (por nivel)



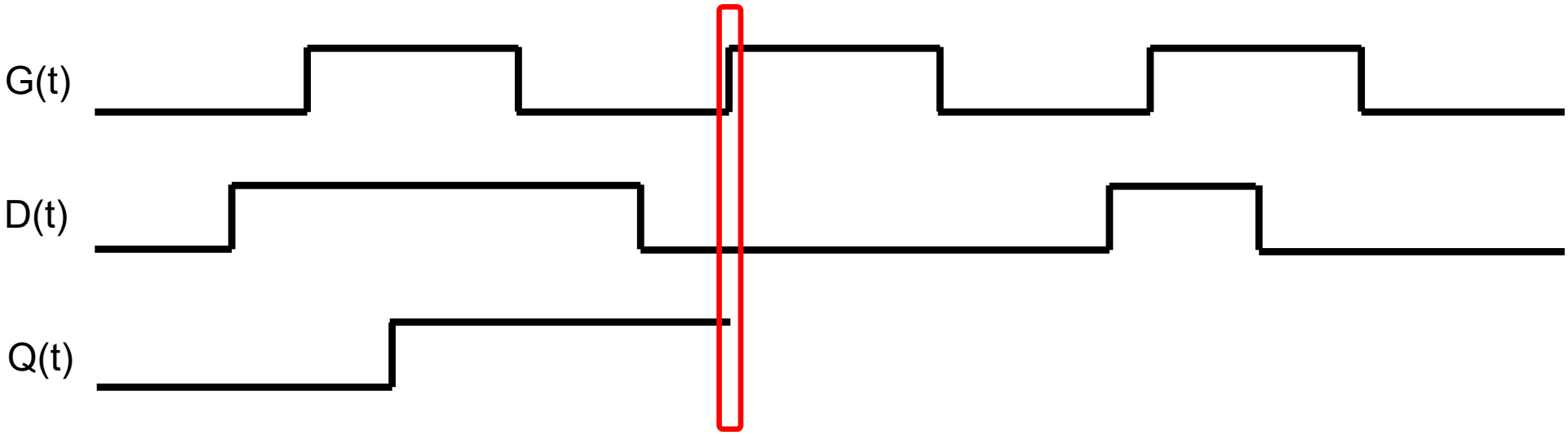
G(t)	D(t)	Q(t+Δt)
0	X	Q(t)
1	0	0
1	1	1



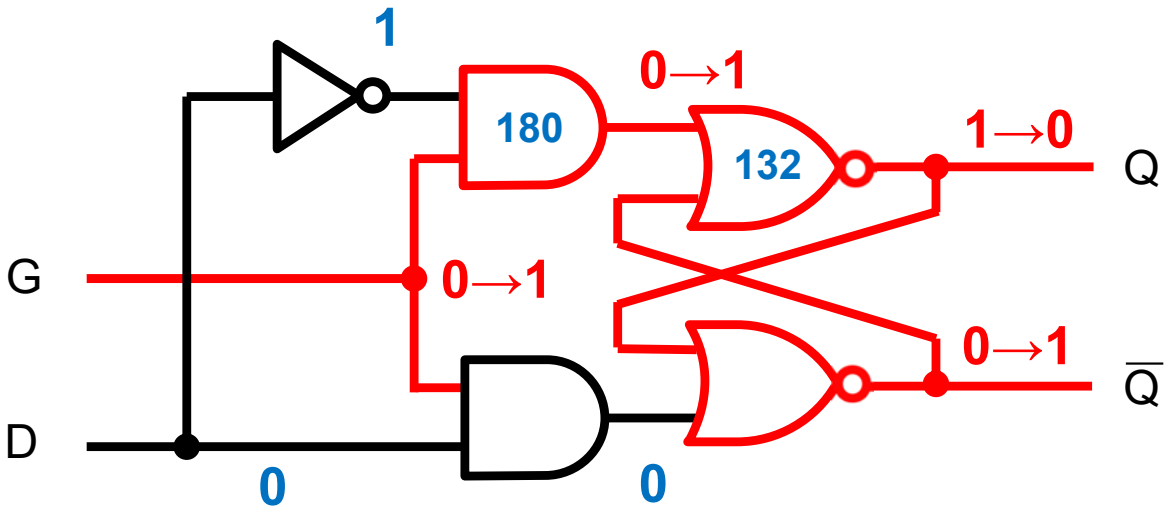
Biestable D síncrono (por nivel)



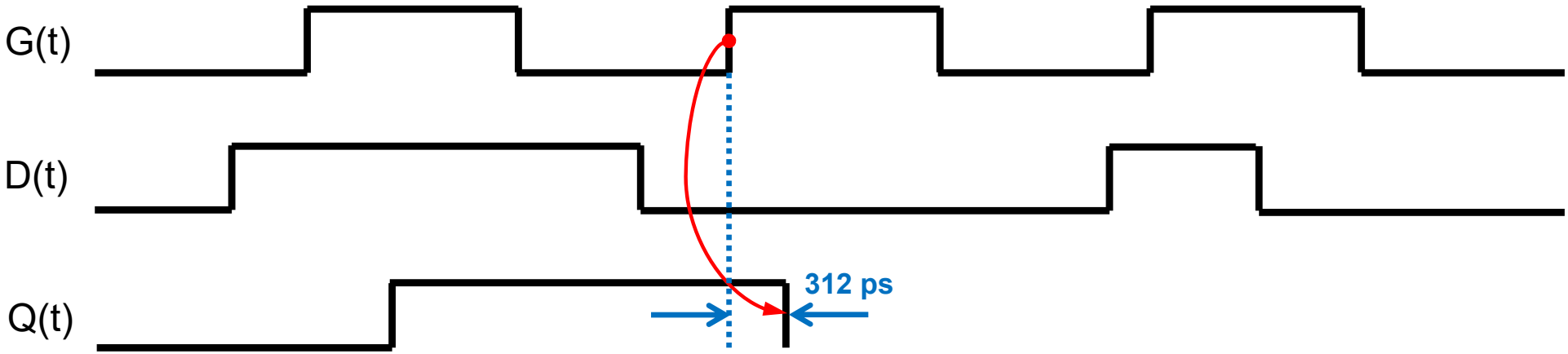
G(t)	D(t)	Q(t+Δt)
0	X	Q(t)
1	0	0
1	1	1



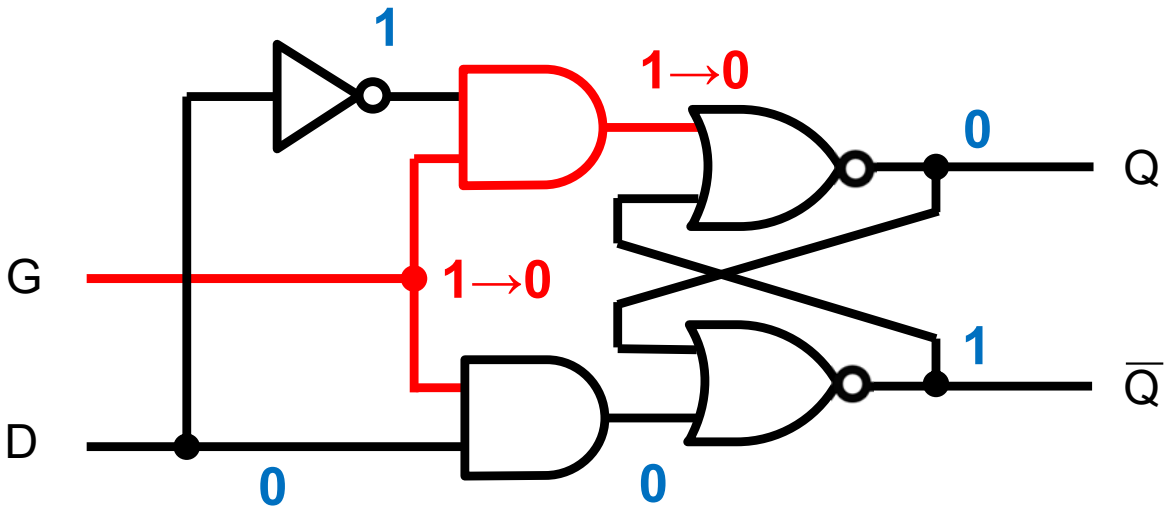
Biestable D síncrono (por nivel)



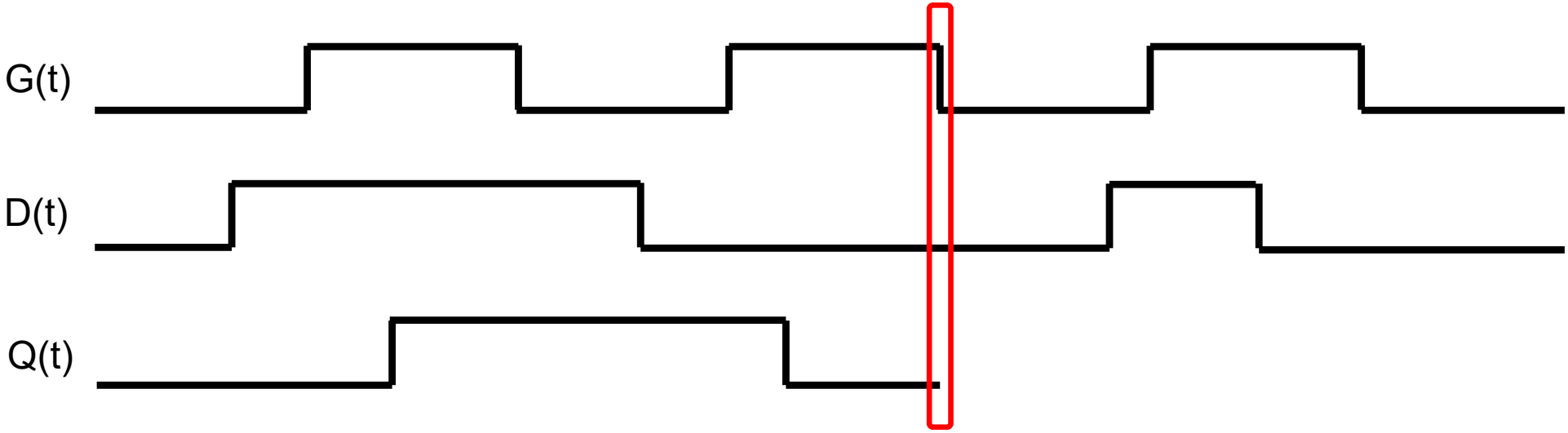
G(t)	D(t)	Q(t+Δt)
0	X	Q(t)
1	0	0
1	1	1



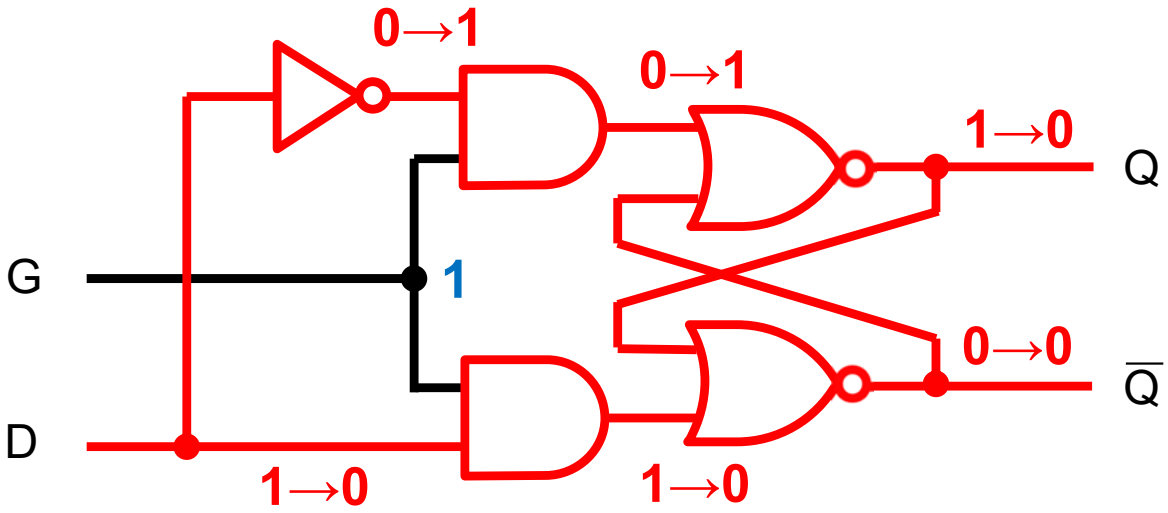
Biestable D síncrono (por nivel)



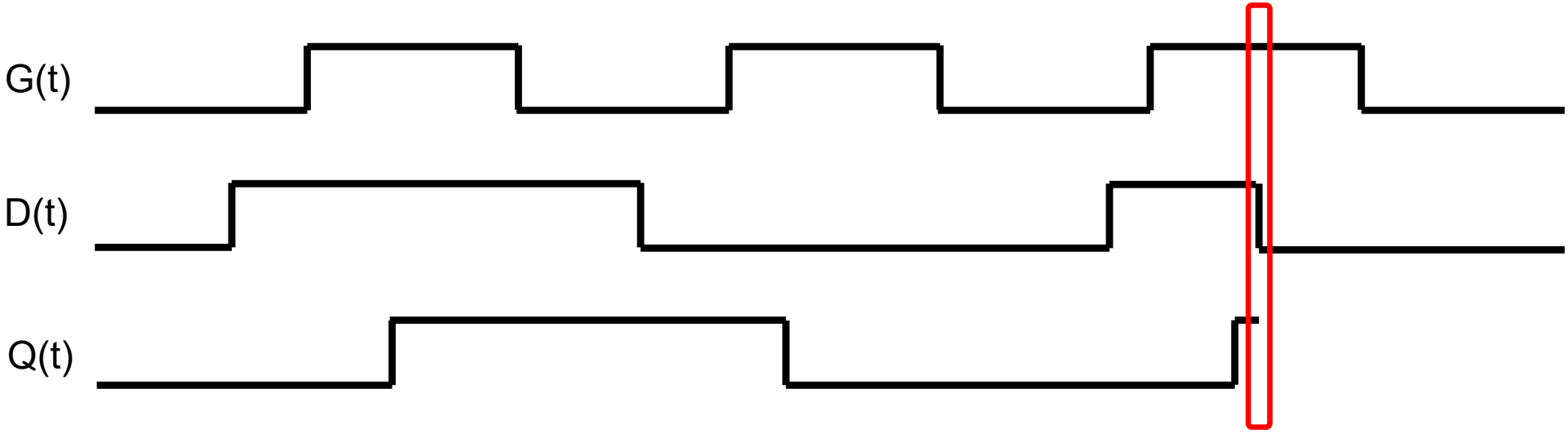
G(t)	D(t)	Q(t+Δt)
0	X	Q(t)
1	0	0
1	1	1



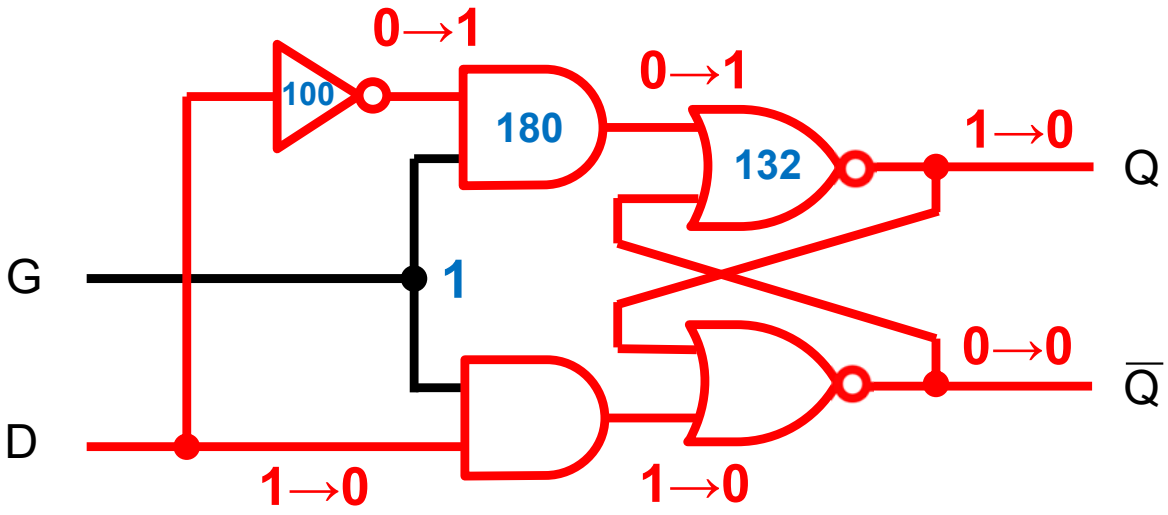
Biestable D síncrono (por nivel)



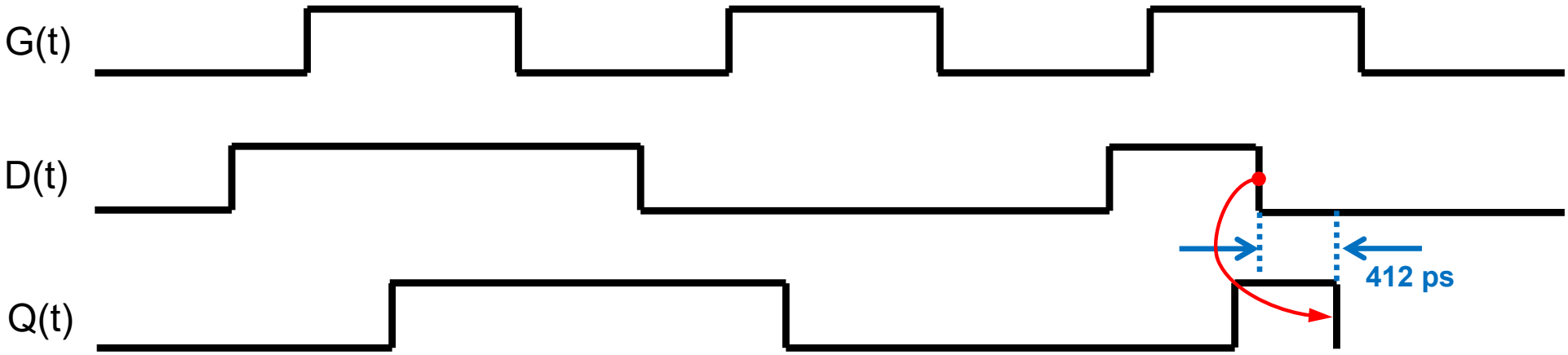
G(t)	D(t)	Q(t+Δt)
0	X	Q(t)
1	0	0
1	1	1



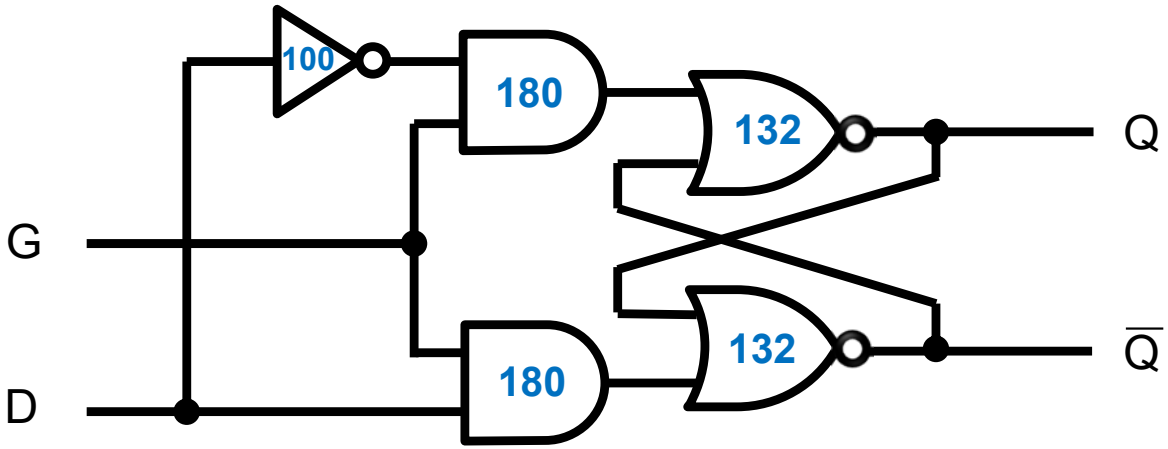
Biestable D síncrono (por nivel)



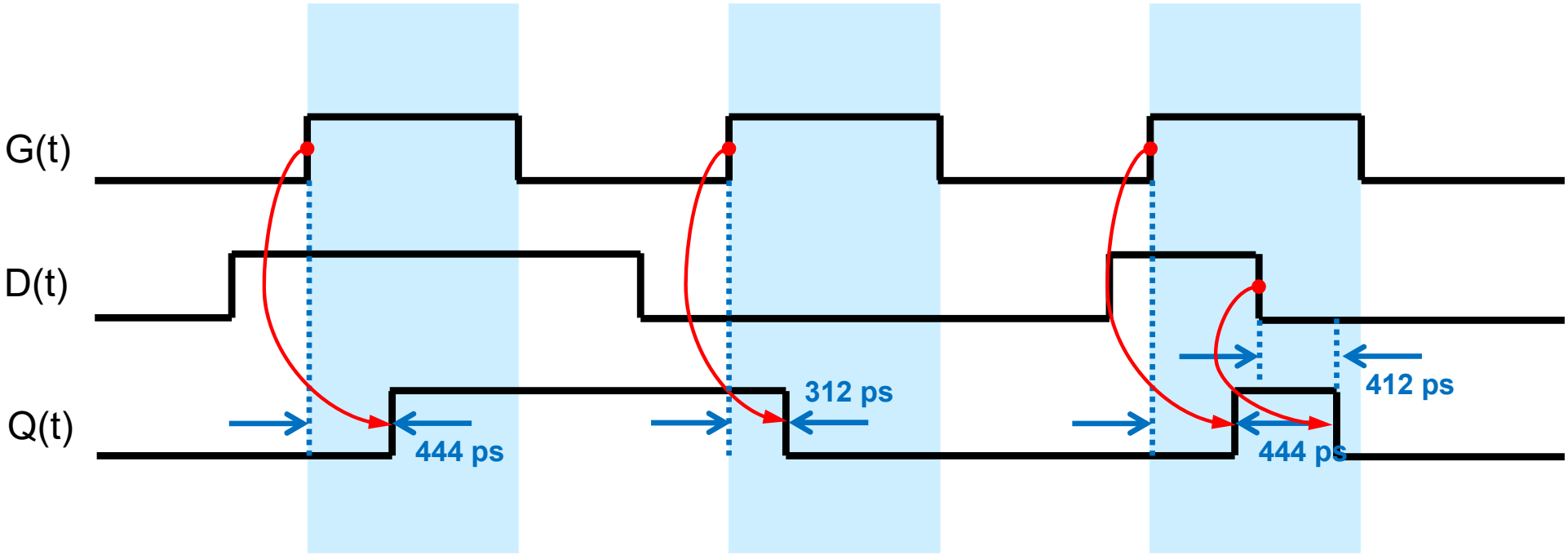
G(t)	D(t)	Q(t+Δt)
0	X	Q(t)
1	0	0
1	1	1



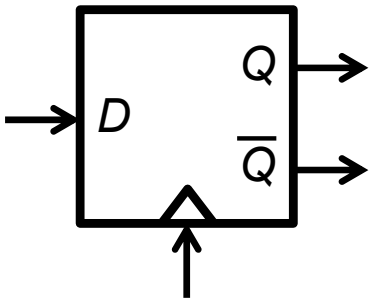
Biestable D síncrono (por nivel)



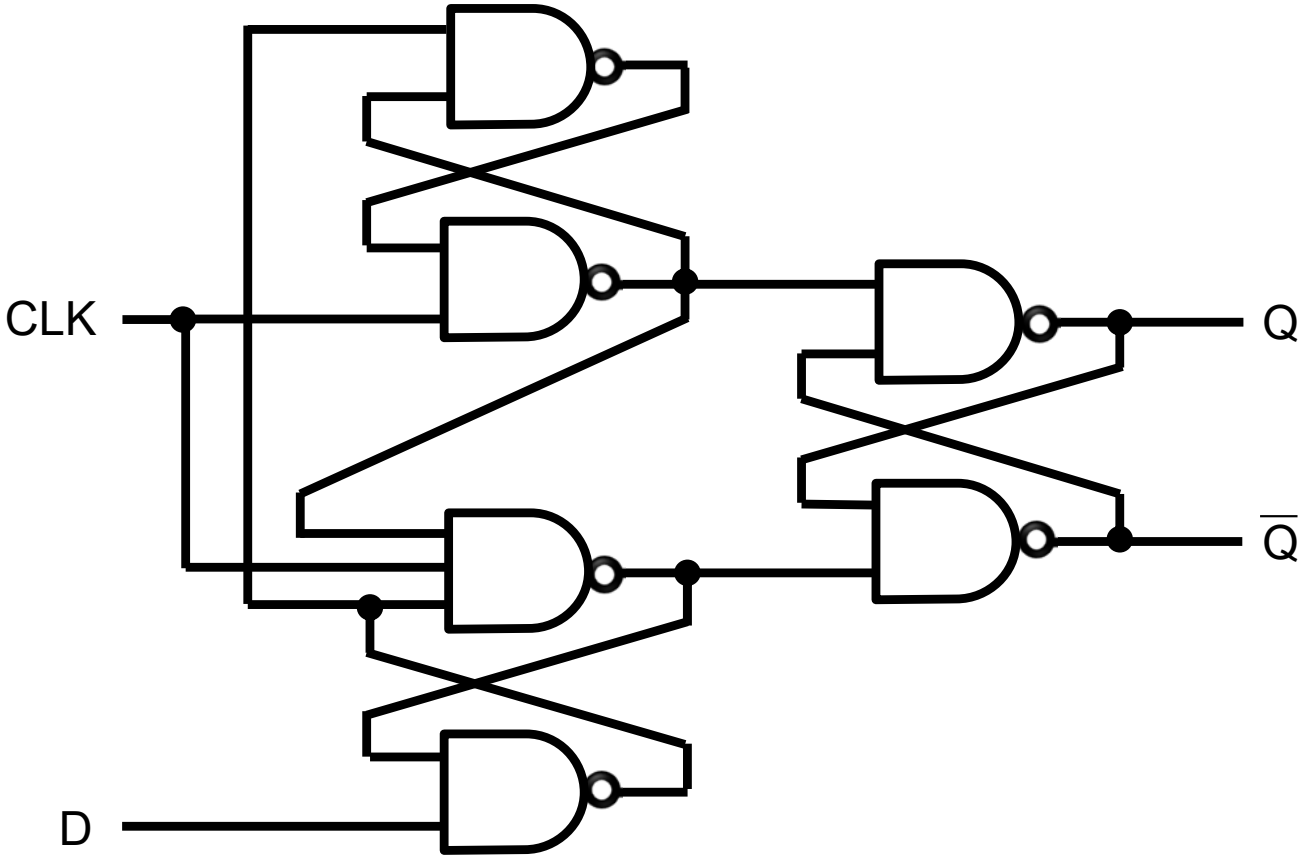
G(t)	D(t)	Q(t+Δt)
0	X	Q(t)
1	0	0
1	1	1



Biestable D síncrono (por flanco)

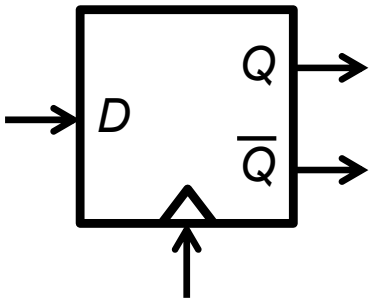


D(t)	CLK	Q(t+1)
0	↑	0
1	↑	1
resto		Q(t)

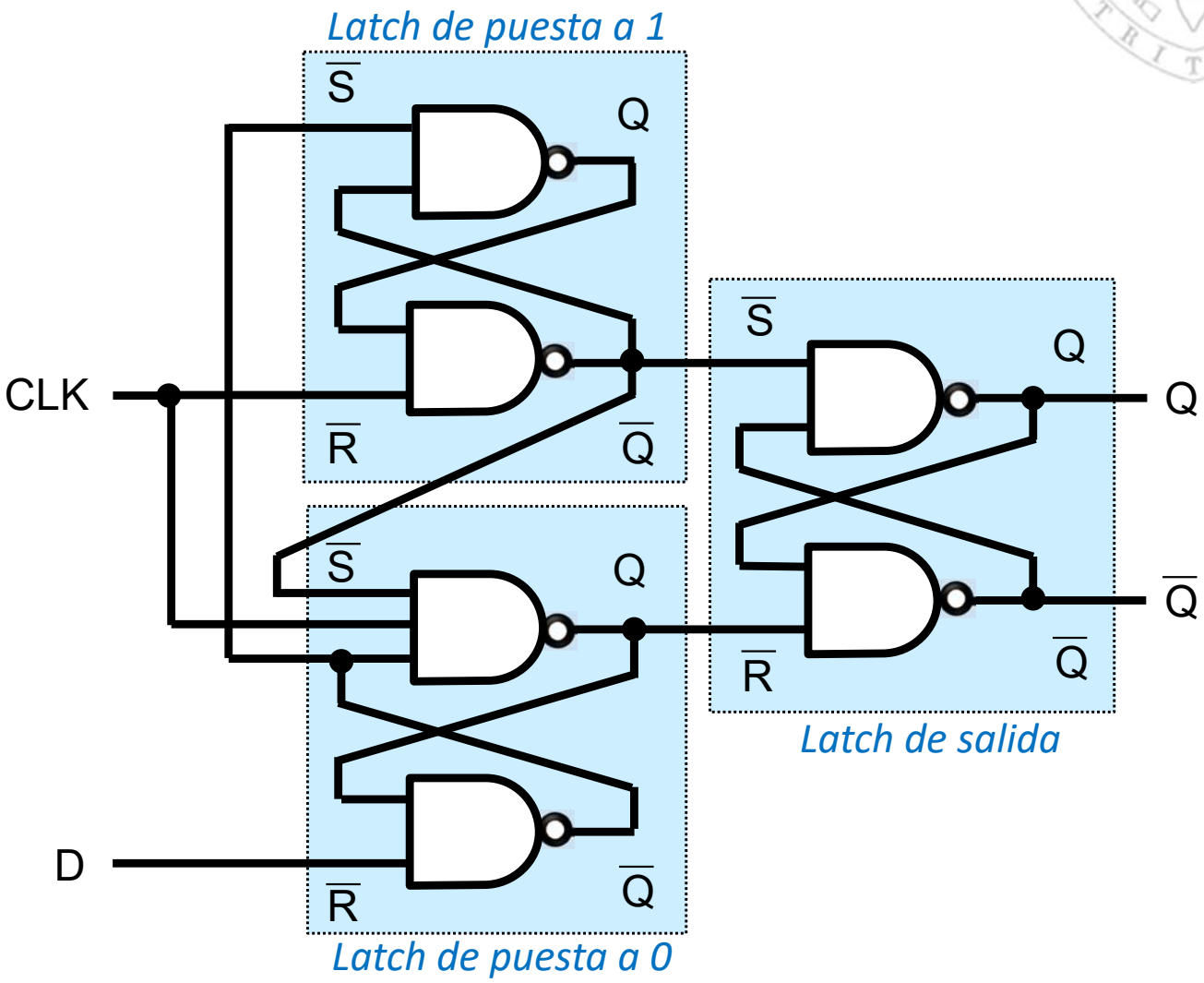


Biestable D síncrono disparado por flanco de subida
(Flip-flop D, implementación con NAND)

Biestable D síncrono (por flanco)

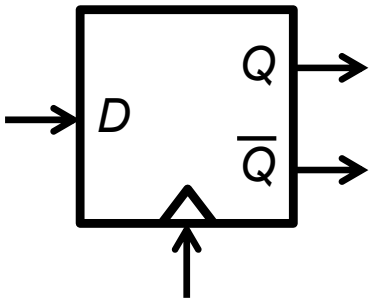


D(t)	CLK	Q(t+1)
0	↑	0
1	↑	1
resto		Q(t)

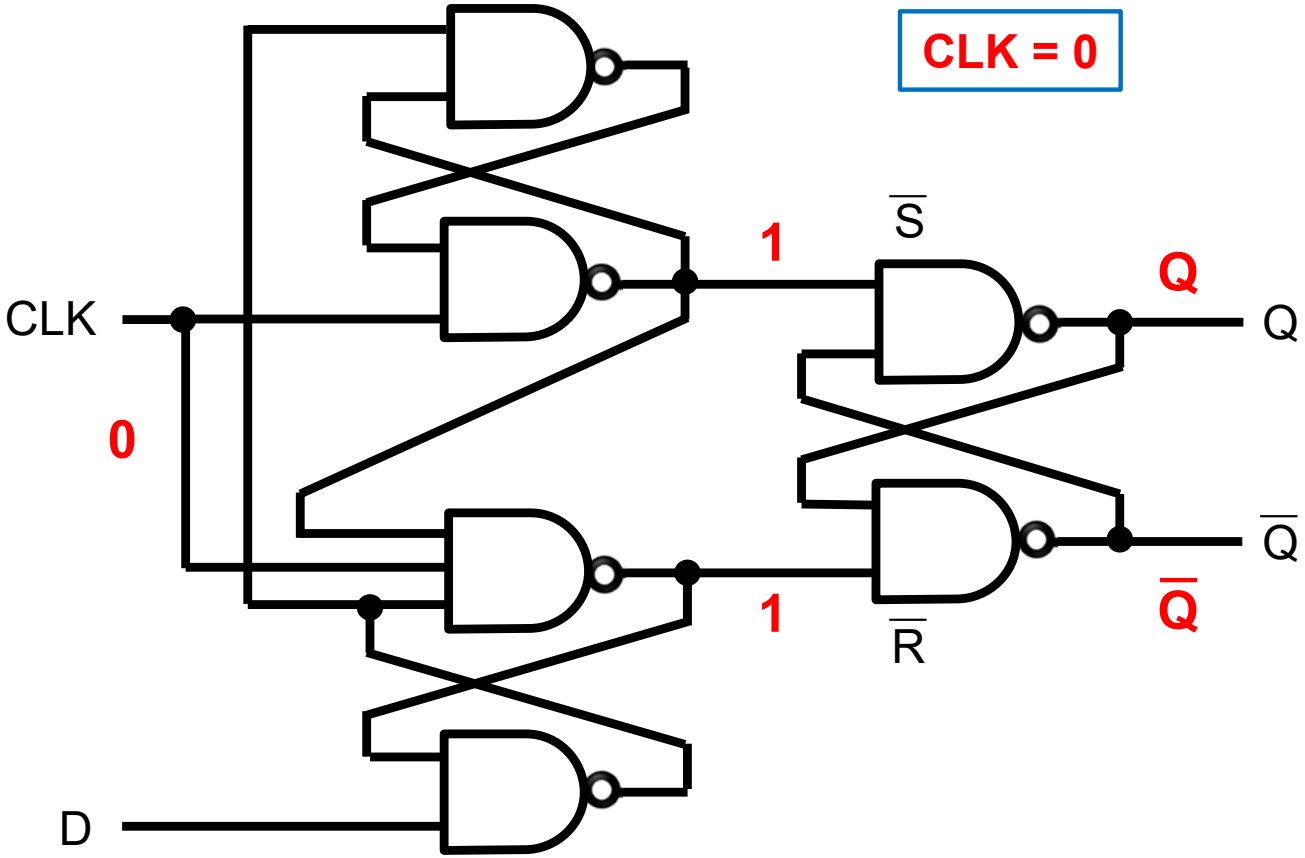


Biestable D síncrono disparado por flanco de subida
(Flip-flop D, implementación con NAND)

Biestable D síncrono (por flanco)

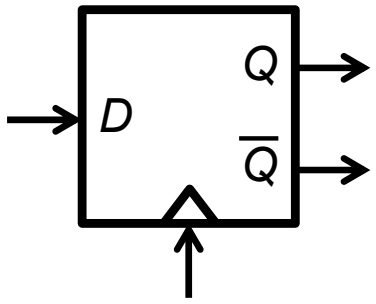


D(t)	CLK	Q(t+1)
0	↑	0
1	↑	1
resto		Q(t)

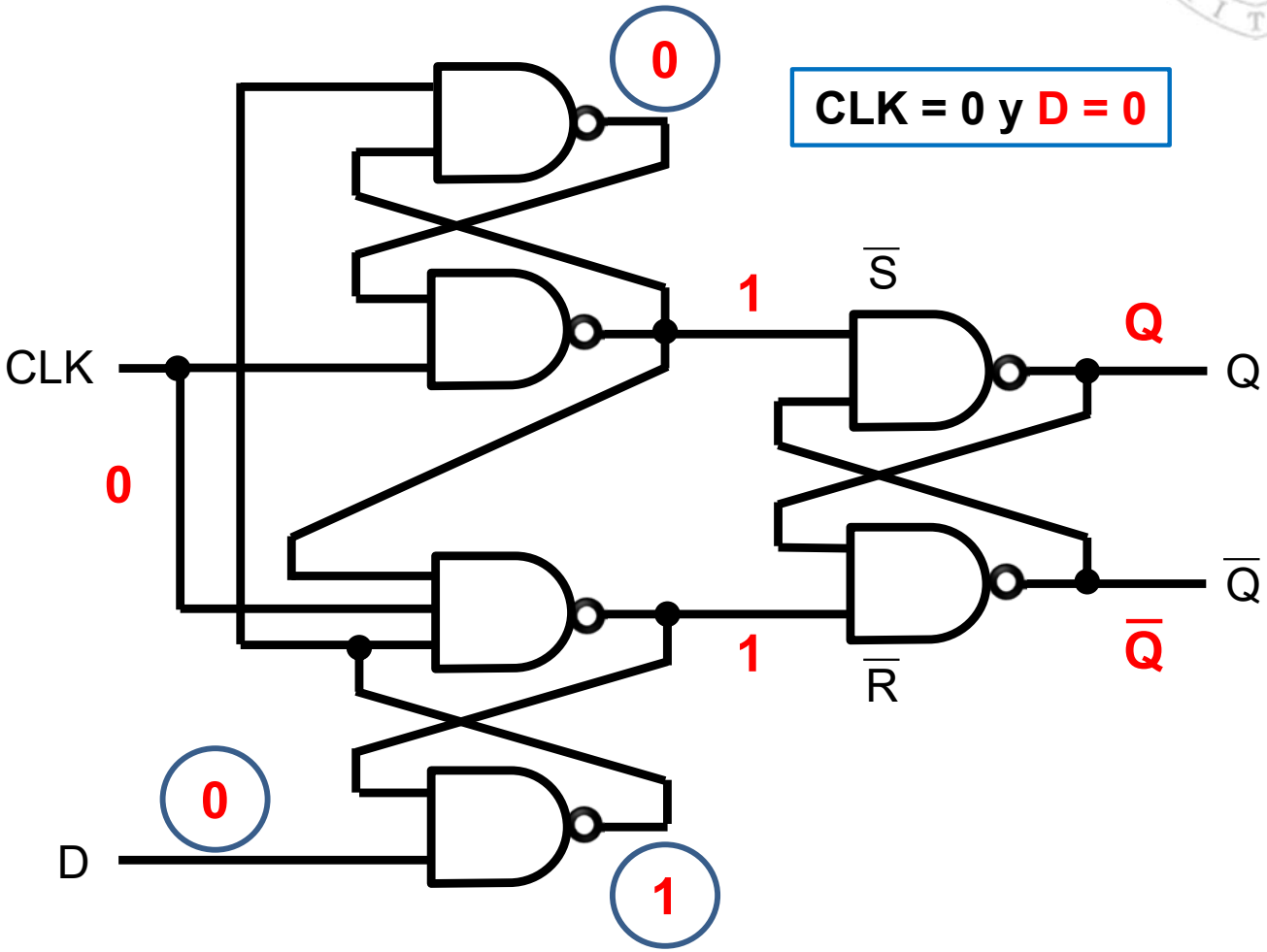


Biestable D síncrono disparado por flanco de subida
(Flip-flop D, implementación con NAND)

Biestable D síncrono (por flanco)

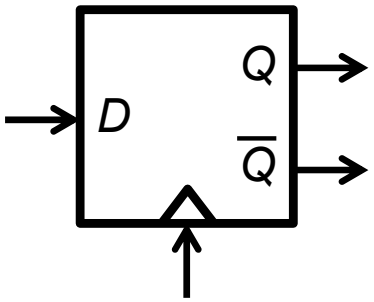


D(t)	CLK	Q(t+1)
0	↑	0
1	↑	1
resto		Q(t)

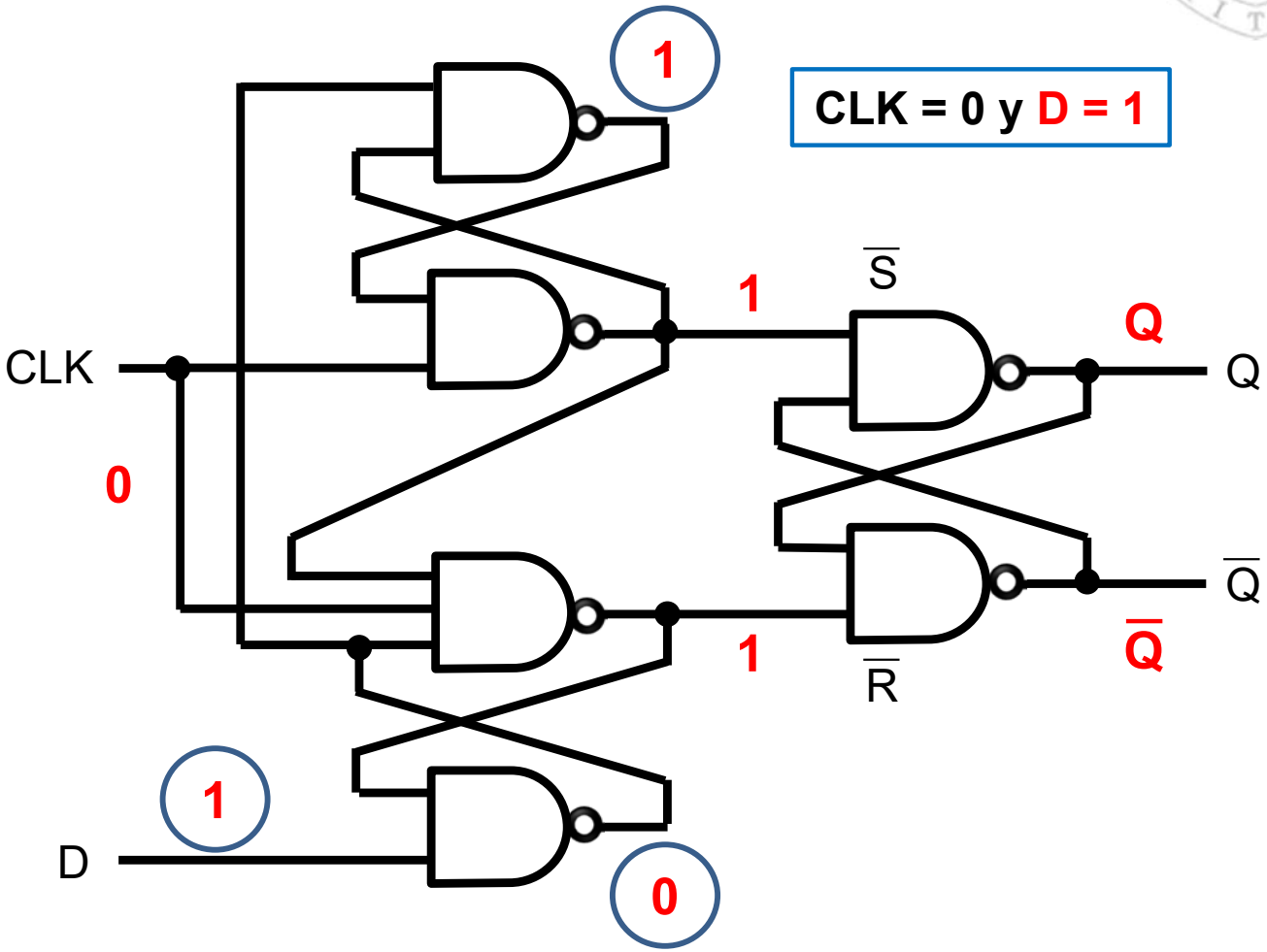


Biestable D síncrono disparado por flanco de subida
(Flip-flop D, implementación con NAND)

Biestable D síncrono (por flanco)

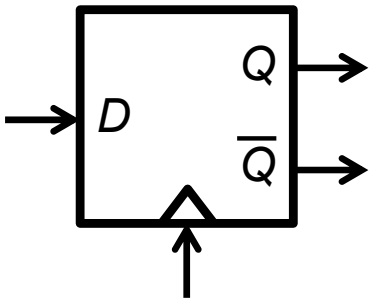


D(t)	CLK	Q(t+1)
0	↑	0
1	↑	1
resto		Q(t)

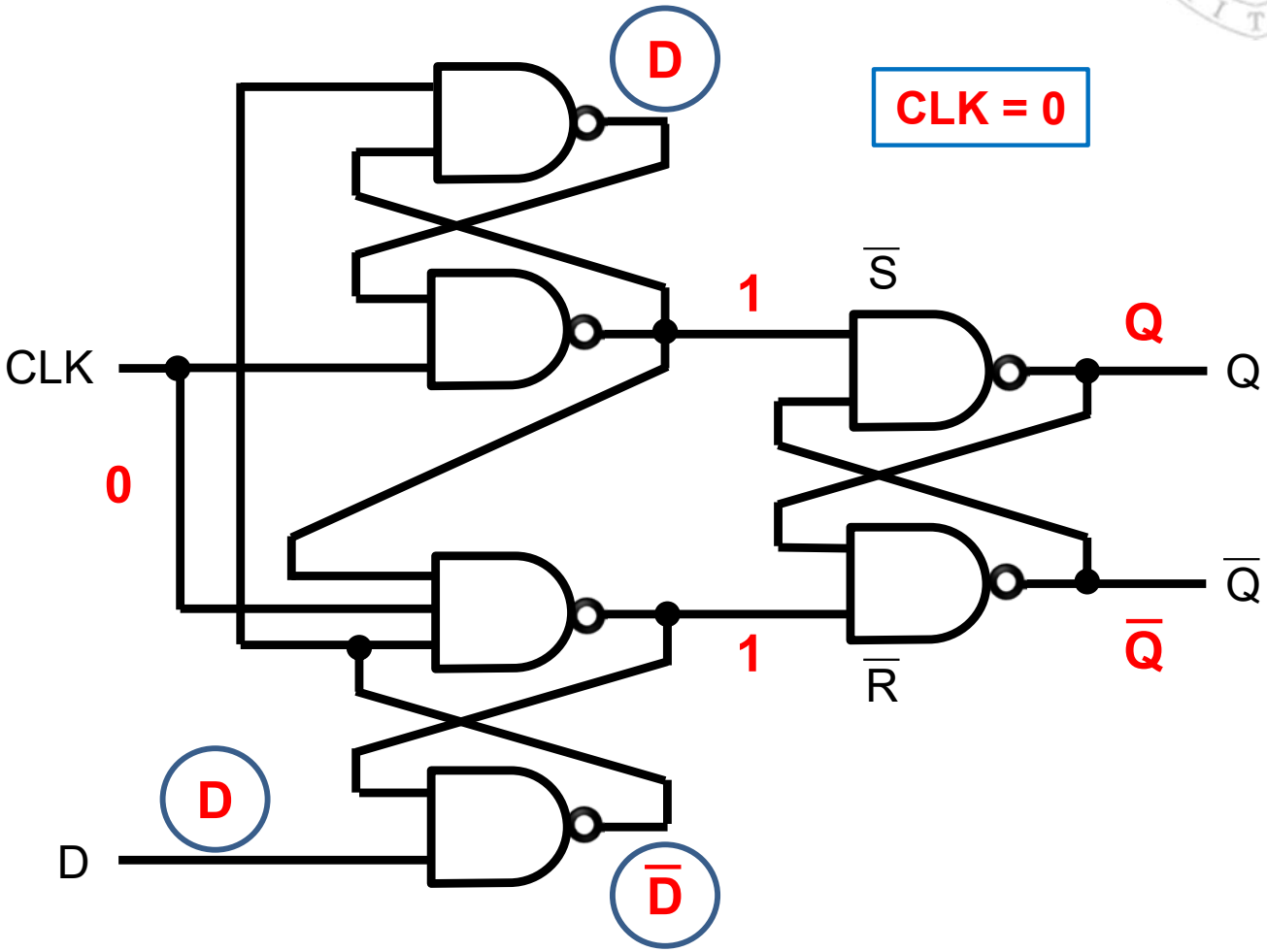


Biestable D síncrono disparado por flanco de subida
(Flip-flop D, implementación con NAND)

Biestable D síncrono (por flanco)

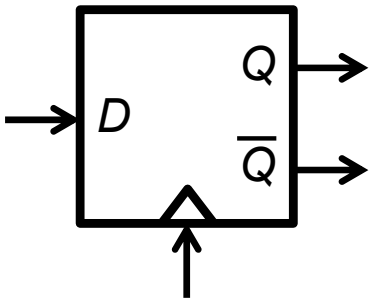


D(t)	CLK	Q(t+1)
0	↑	0
1	↑	1
resto		Q(t)

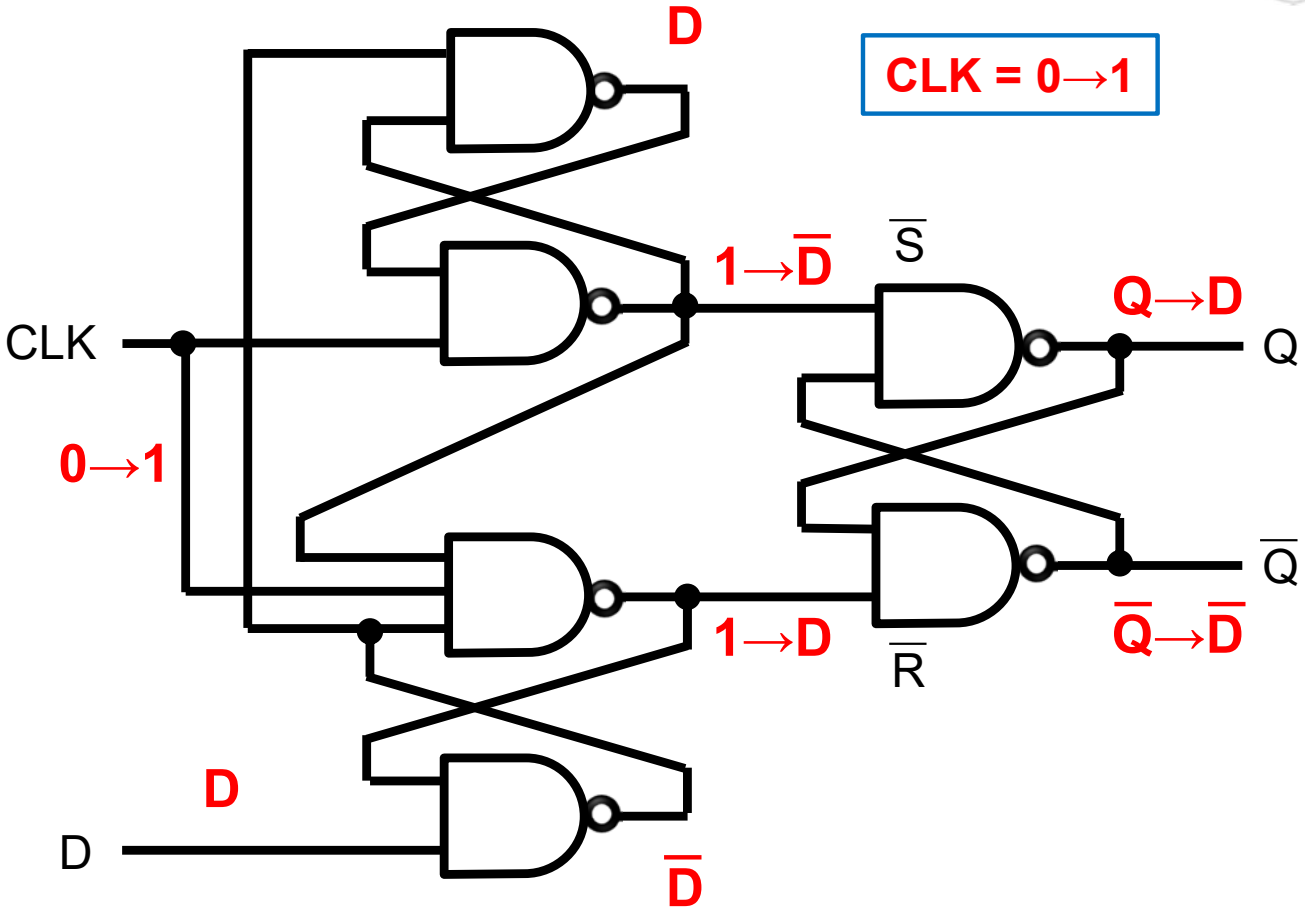


Biestable D síncrono disparado por flanco de subida
(Flip-flop D, implementación con NAND)

Biestable D síncrono (por flanco)

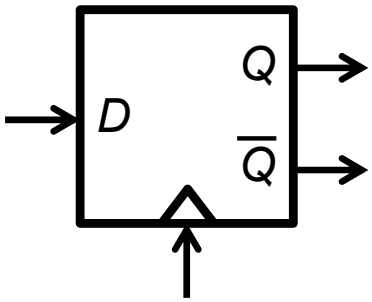


D(t)	CLK	Q(t+1)
0	↑	0
1	↑	1
resto		Q(t)

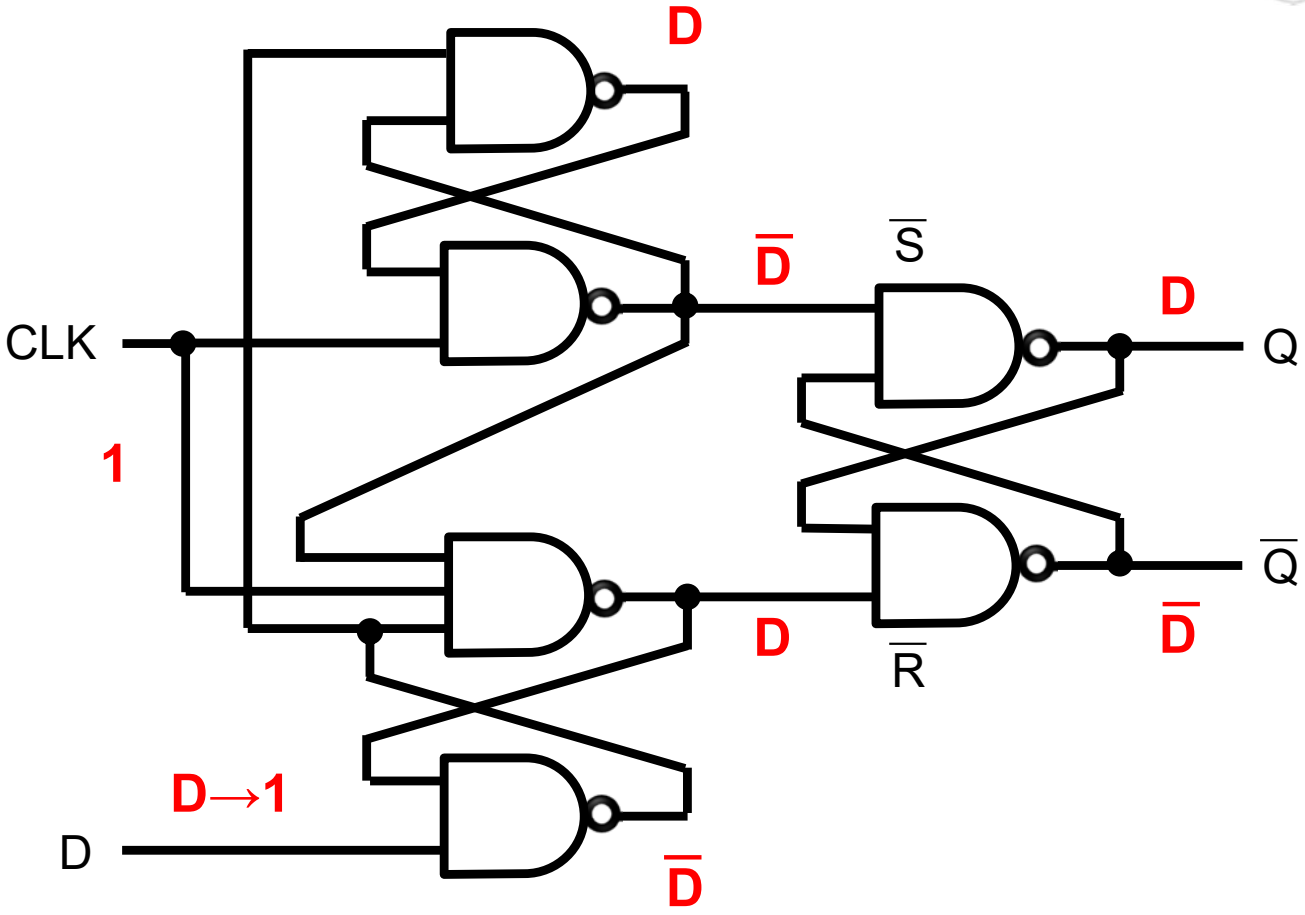


Biestable D síncrono disparado por flanco de subida
(Flip-flop D, implementación con NAND)

Biestable D síncrono (por flanco)

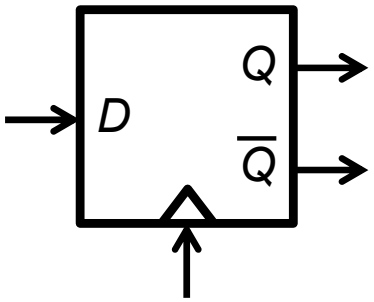


D(t)	CLK	Q(t+1)
0	↑	0
1	↑	1
resto		Q(t)

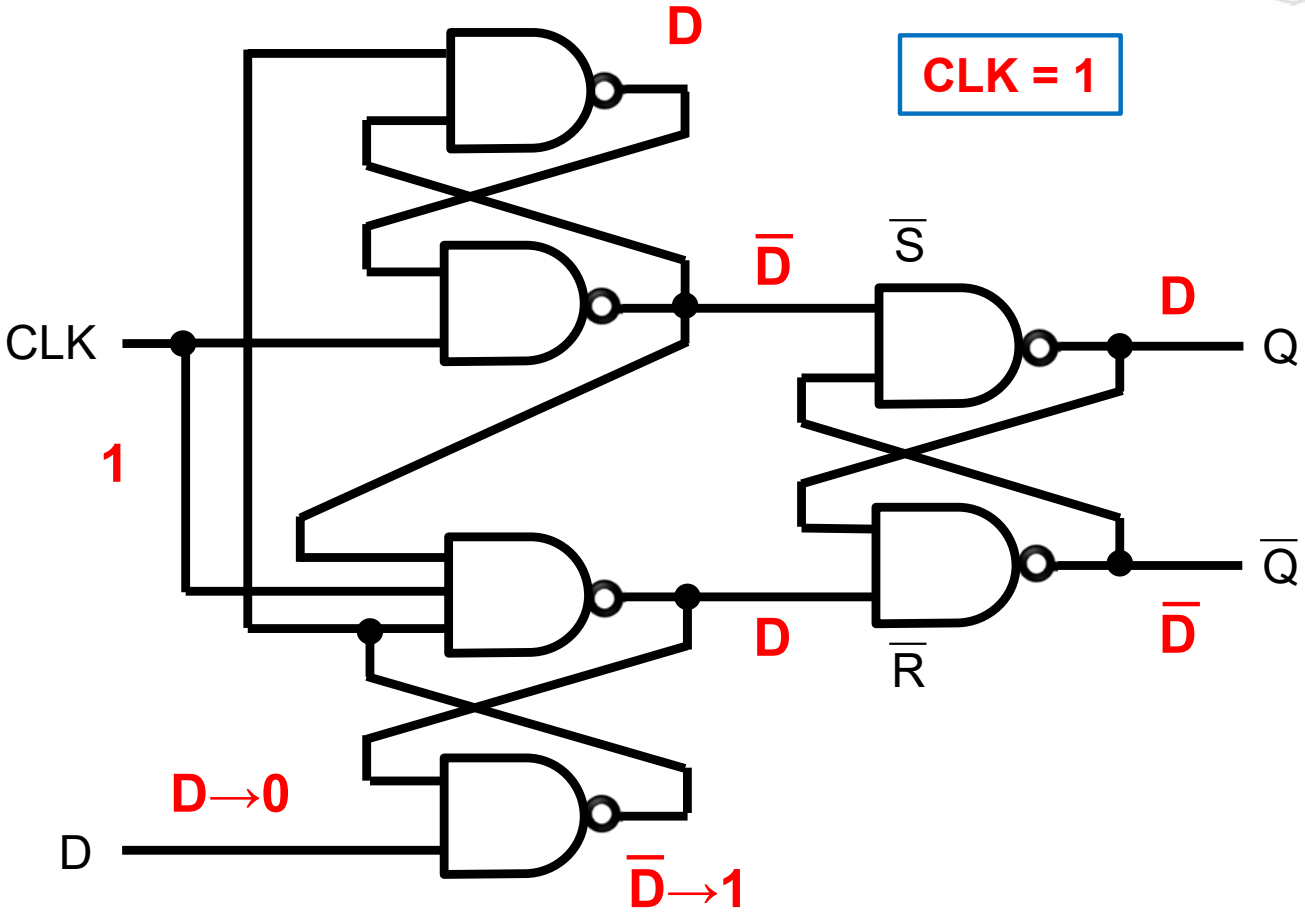


Biestable D síncrono disparado por flanco de subida
(Flip-flop D, implementación con NAND)

Biestable D síncrono (por flanco)

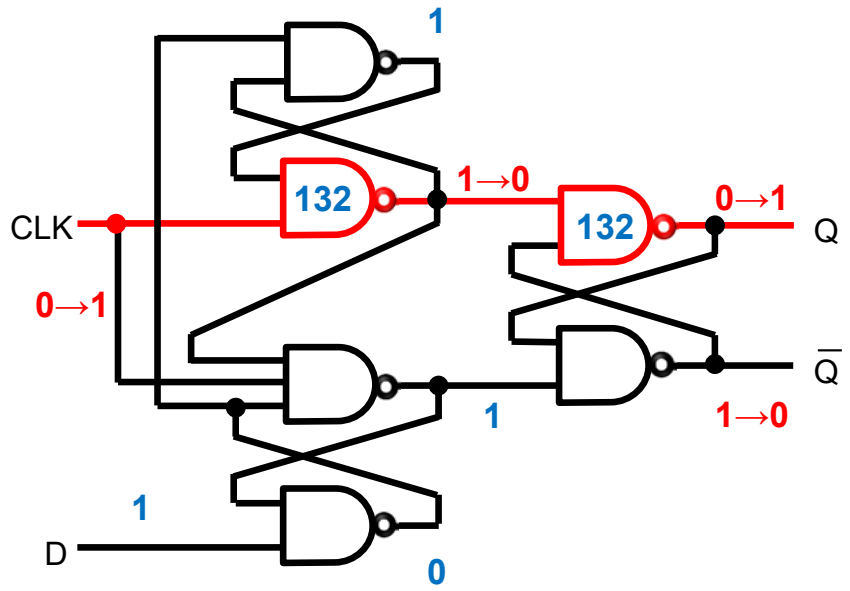


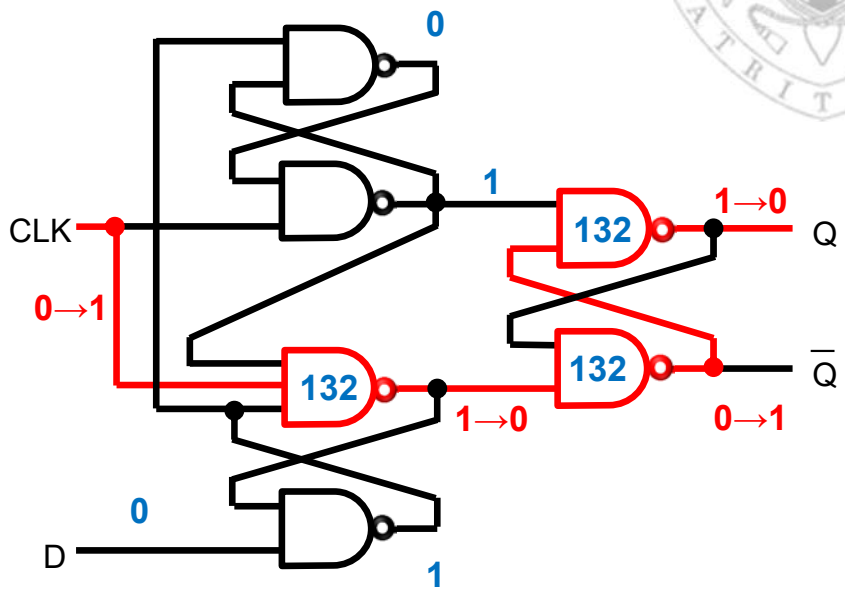
D(t)	CLK	Q(t+1)
0	↑	0
1	↑	1
resto		Q(t)

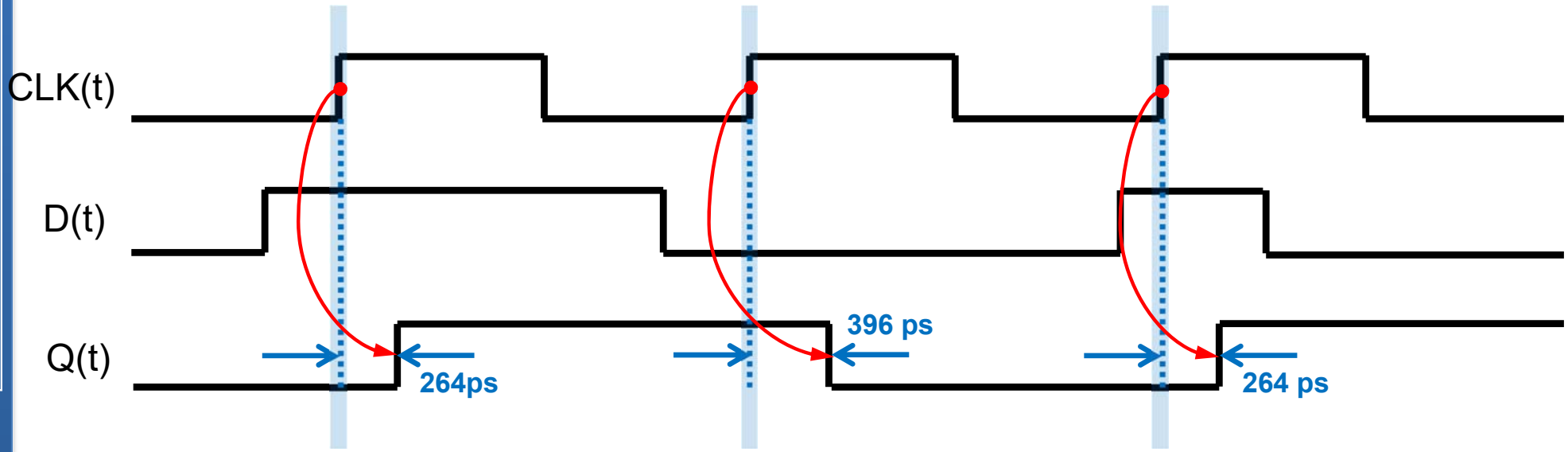


Biestable D síncrono disparado por flanco de subida
(Flip-flop D, implementación con NAND)

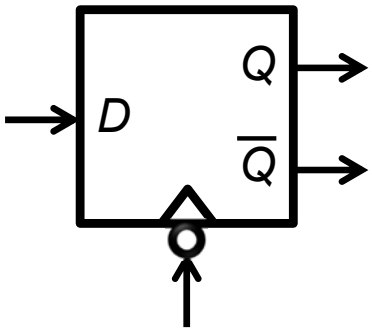
Biestable D síncrono (por flanco)



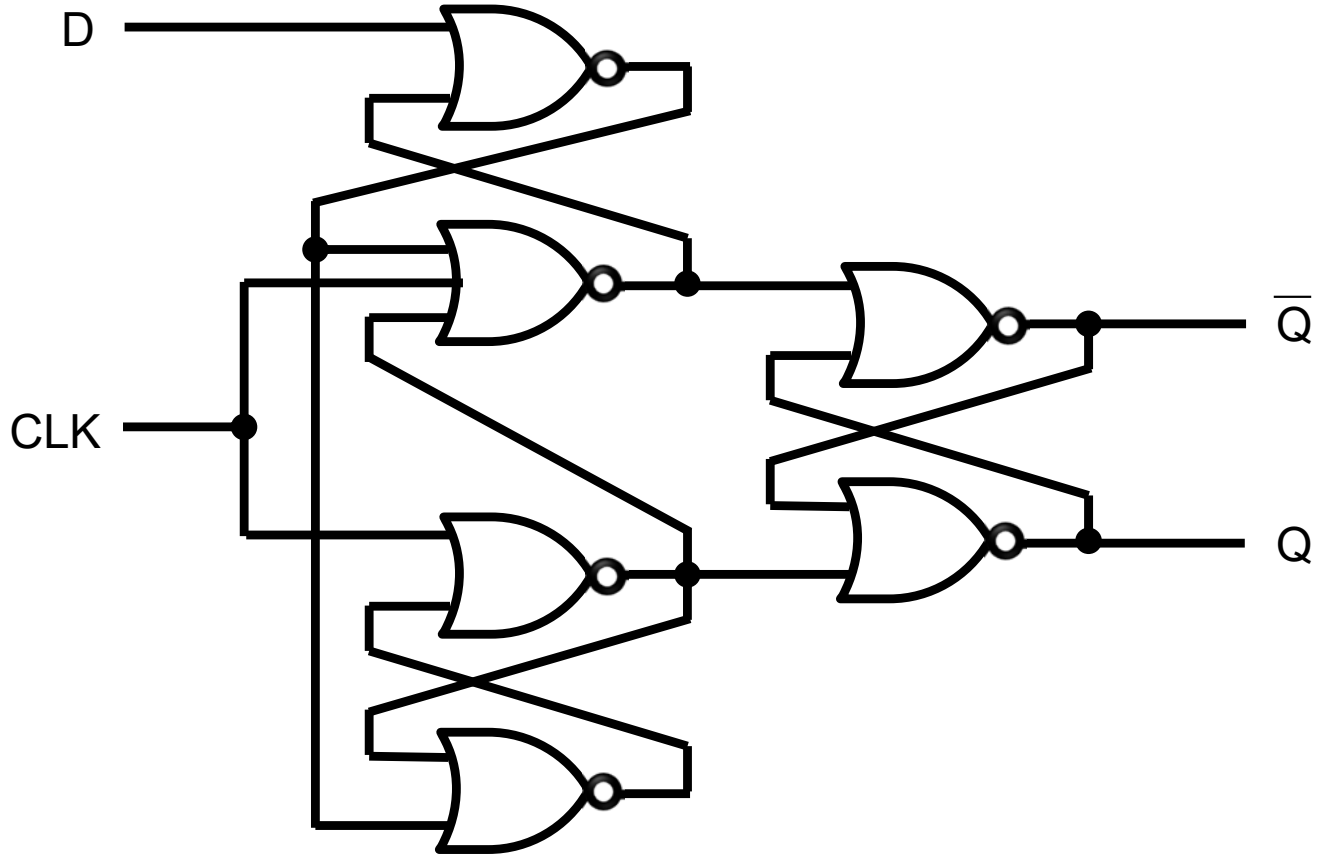




Biestable D síncrono (por flanco)



D(t)	CLK	Q(t+1)
0	↓	0
1	↓	1
resto		Q



Biestable D síncrono disparado por flanco de bajada
(Flip-flop D, implementación con NOR)

Síntesis con biestables D

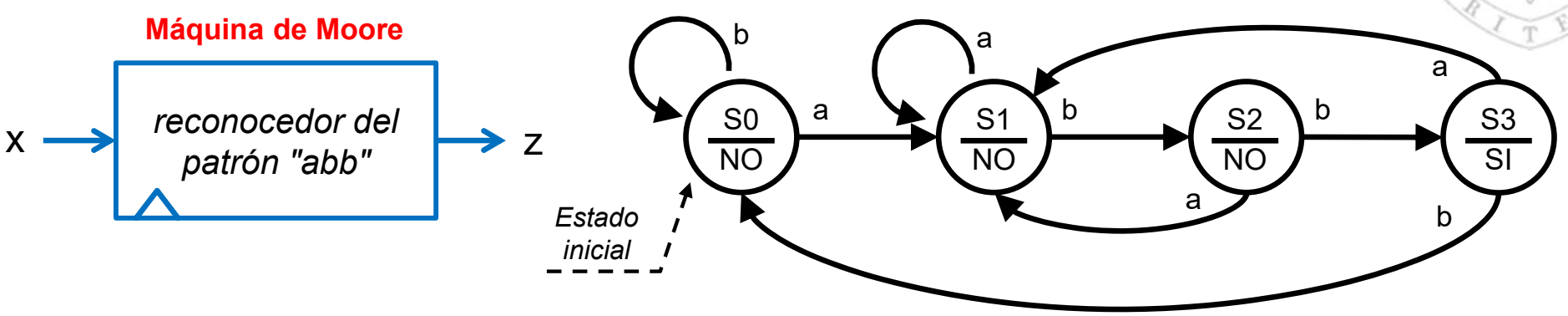
- Dada una especificación de una conducta secuencial implementarla como una red de módulos combinacionales y biestables D, en donde:
  - Todos los biestables se conectan a una señal de reloj periódica.
  - Todos los biestables se disparan por flancos de la misma polaridad.
  - Toda realimentación incluye al menos un biestable.

- Implementación canónica: realización directa de un diagrama de estados:
  - El registro de estado se implementa como un array de biestables D disparados por flanco (todos de la misma polaridad) con reloj común.
  - 2 bloques de lógica combinacional implementan las funciones de salida y de transición de estados.

Síntesis con biestables D

- Este método de síntesis sigue un modelo de temporización síncrona por flanco de reloj global, en donde:
  - Los cálculos que realiza el sistema se realizan ciclo a ciclo.
  - Las fronteras del ciclo están marcadas por las transiciones de igual polaridad en el reloj común.
  - Al comienzo del ciclo, el sistema hace un cambio de estado mediante la actualización simultánea de todos los biestables.
  - El nuevo estado provoca transiciones en las entradas de los módulos combinacionales que a su vez provocarán transiciones en sus salidas.
  - El cálculo a realizar en el ciclo finaliza cuando todos los sistemas combinacionales han alcanzado su régimen permanente.
  - Los valores permanentes a la salida de los módulos combinacionales son utilizados para actualizar los biestables al comienzo del ciclo siguiente.

Síntesis con biestables D



- Codificación domino: { a → 0, b → 1 }
- Codificación codominio: { NO→ 0, SI → 1 }
- Codificación estados: { S0 → (00), S1 → (01), S2 → (10), S3 → (11) }

Función de transición de estados

x	q_1	q_0	q_1'	q_0'
0	0	0	0	1
0	0	1	0	1
0	1	0	0	1
0	1	1	0	1
1	0	0	0	0
1	0	1	1	0
1	1	0	1	1
1	1	1	0	0

Función de salida

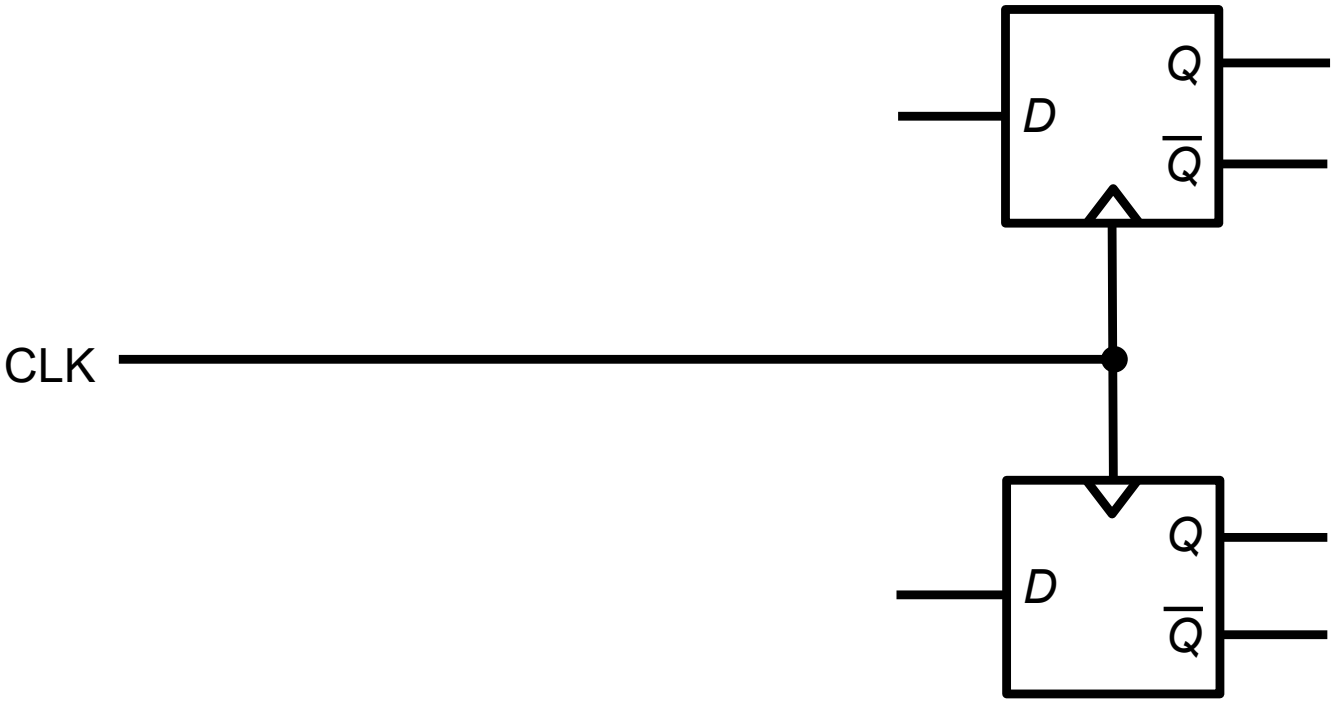
q_1	q_0	z
0	0	0
0	1	0
1	0	0
1	1	1

$$q_1' = x(q_1 \oplus q_0)$$
$$q_0' = \bar{x} + q_1\overline{q_0}$$
$$z = q_1 q_0$$

Síntesis con biestables D

$$q_1{}' = x(q_1 \oplus q_0)$$
$$q_0{}' = \bar{x} + q_1\overline{q_0}$$
$$z = q_1 q_0$$

Síntesis con biestables D

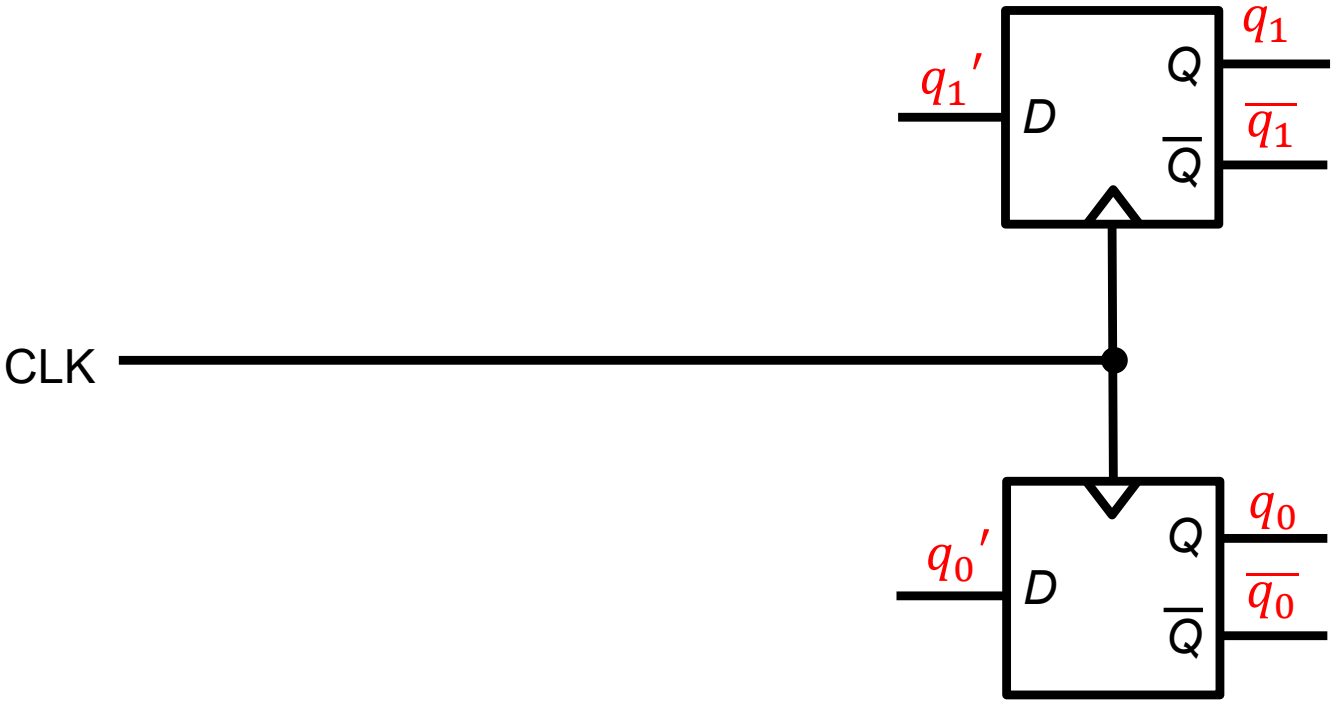


$$q_1{}' = x(q_1 \oplus q_0)$$

$$q_0{}' = \bar{x} + q_1\overline{q_0}$$

$$z = q_1 q_0$$

Síntesis con biestables D

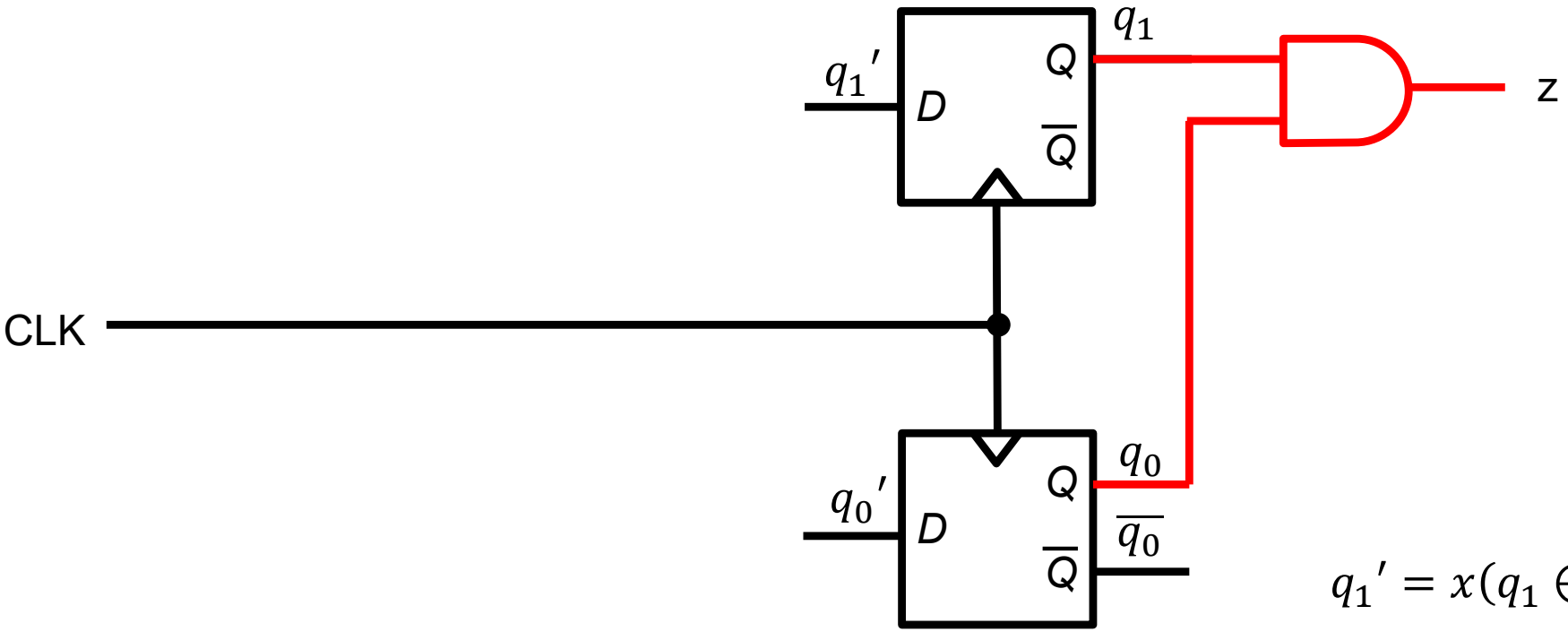


$$q_1{}' = x(q_1 \oplus q_0)$$

$$q_0{}' = \bar{x} + q_1\overline{q_0}$$

$$z = q_1 q_0$$

Síntesis con biestables D

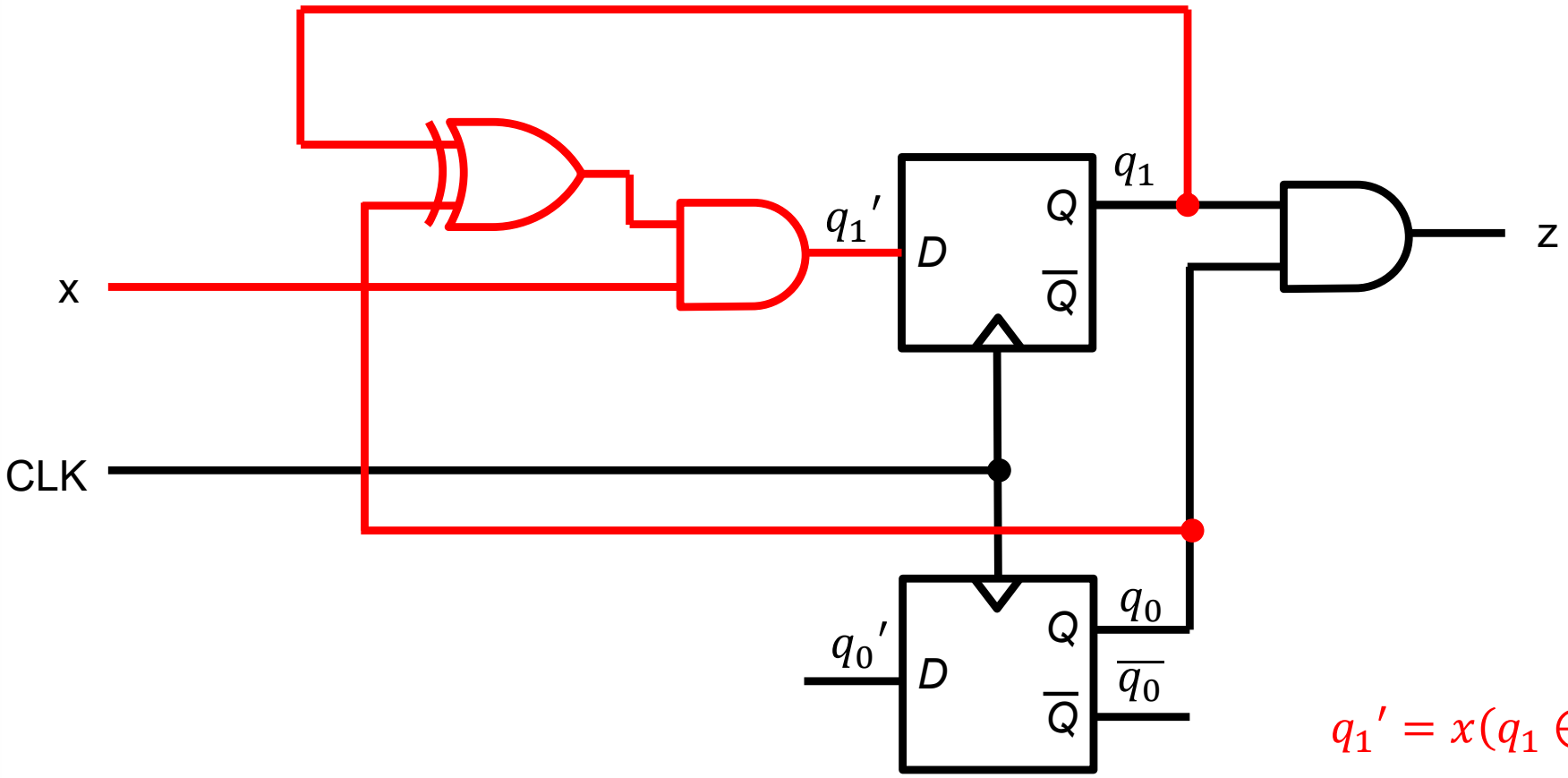


$$q_1{}' = x(q_1 \oplus q_0)$$

$$q_0{}' = \bar{x} + q_1\overline{q_0}$$

$$z = q_1 q_0$$

Síntesis con biestables D

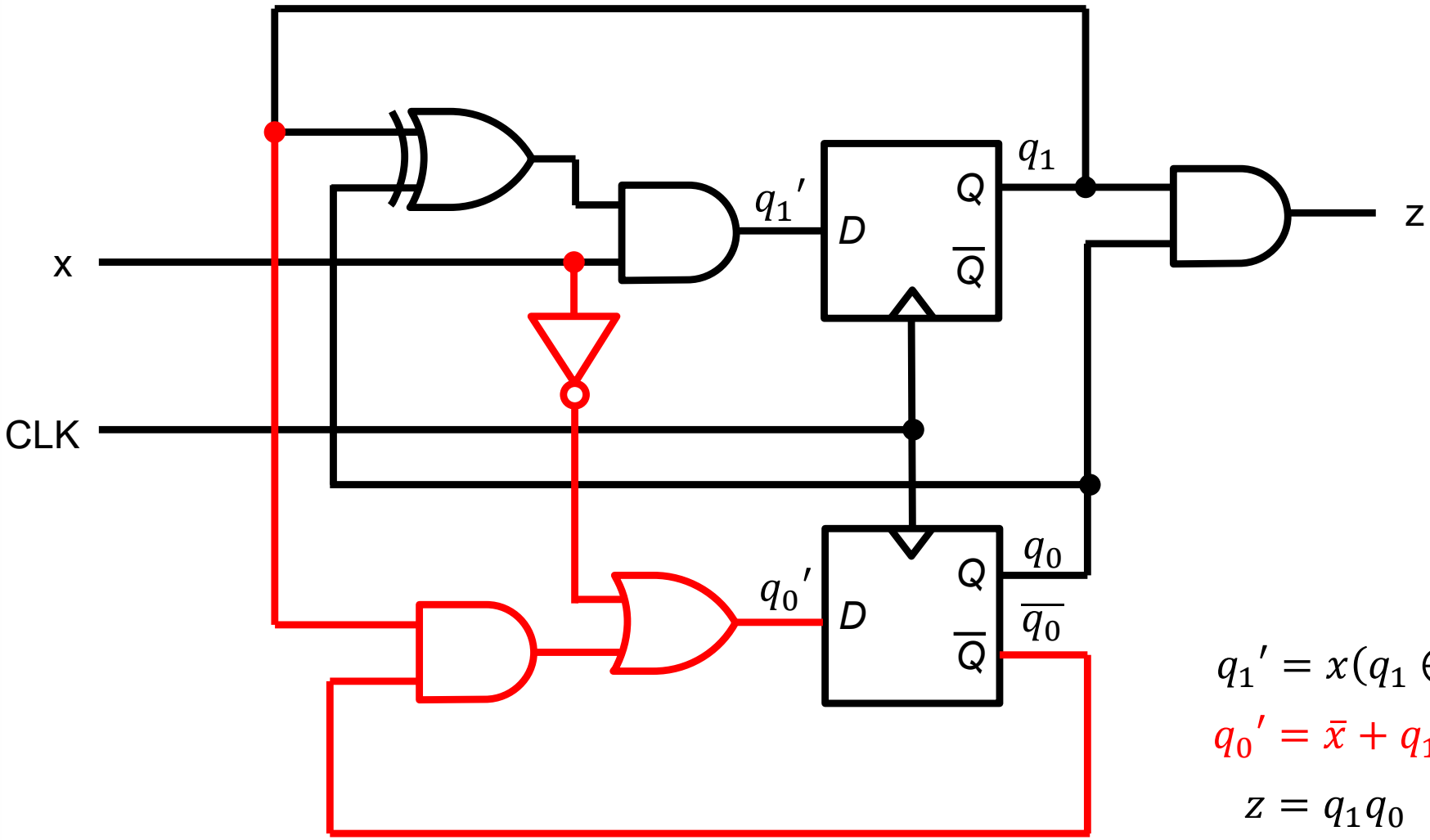


$$q_1' = x(q_1 \oplus q_0)$$

$$q_0' = \bar{x} + q_1\overline{q_0}$$

$$z = q_1 q_0$$

Síntesis con biestables D

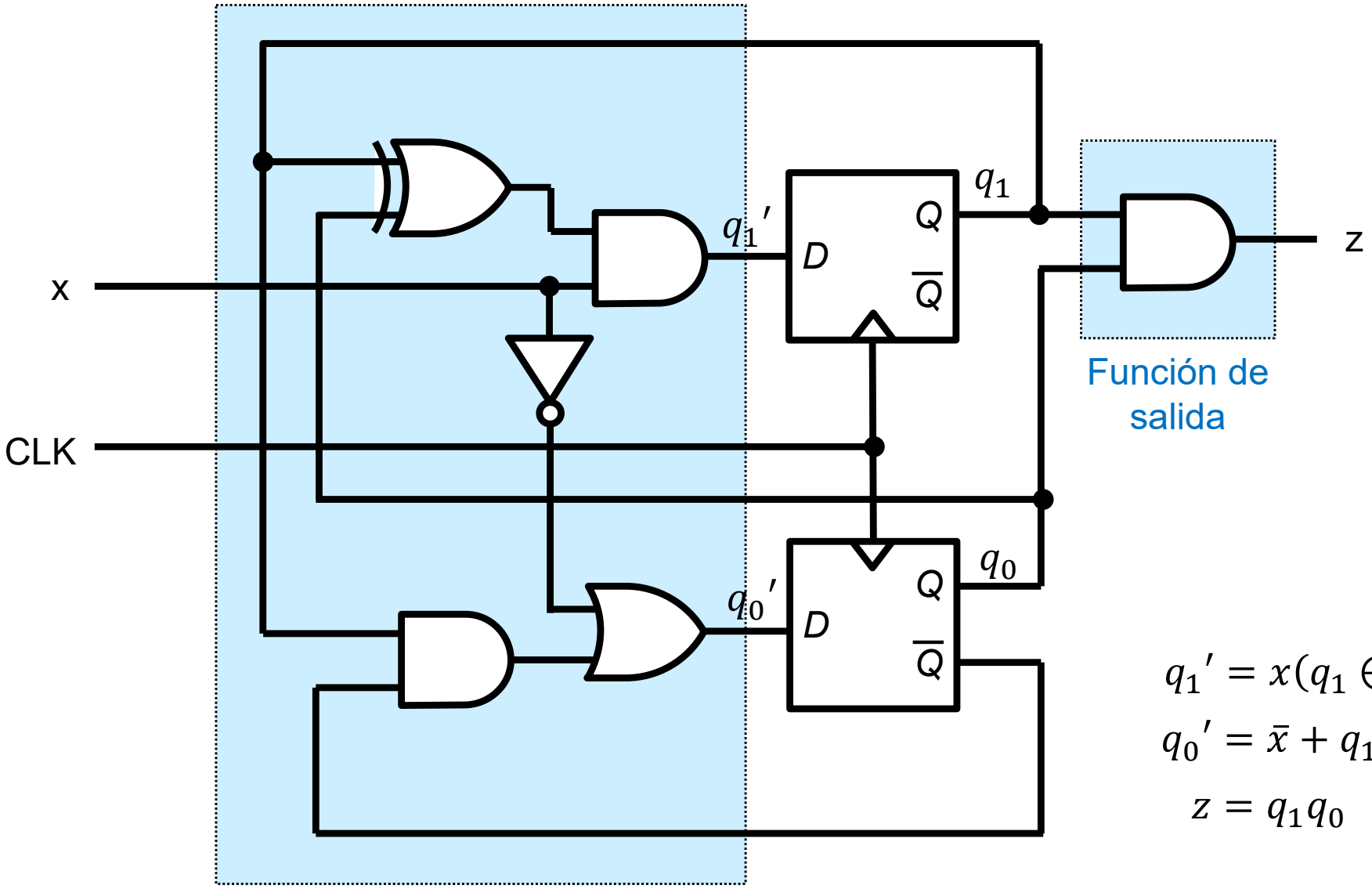


$$q_1{}' = x(q_1 \oplus q_0)$$

$$q_0{}' = \bar{x} + q_1\overline{q_0}$$

$$z = q_1 q_0$$

Síntesis con biestables D



$$q_1' = x(q_1 \oplus q_0)$$

$$q_0' = \bar{x} + q_1\overline{q_0}$$

$$z = q_1 q_0$$

Síntesis con biestables D

- Codificaciones distintas dan lugar a implementaciones diferentes de la misma máquina de estados.
  - Por ello es interesante elegir aquella codificación que reduzca al máximo el coste/retardo de los circuitos de transición y salida.

- Codificación domino: { a → 0, b → 1 }
- Codificación codominio: { NO→ 0, SI → 1 }
- Codificación estados: { S0 → (01), S1 → (00), S2 → (10), S3 → (11) }

Función de transición de estados

x	q_1	q_0	q_1'	q_0'
0	0	0	0	0
0	0	1	0	0
0	1	0	0	0
0	1	1	0	0
1	0	0	1	0
1	0	1	0	1
1	1	0	1	1
1	1	1	0	1

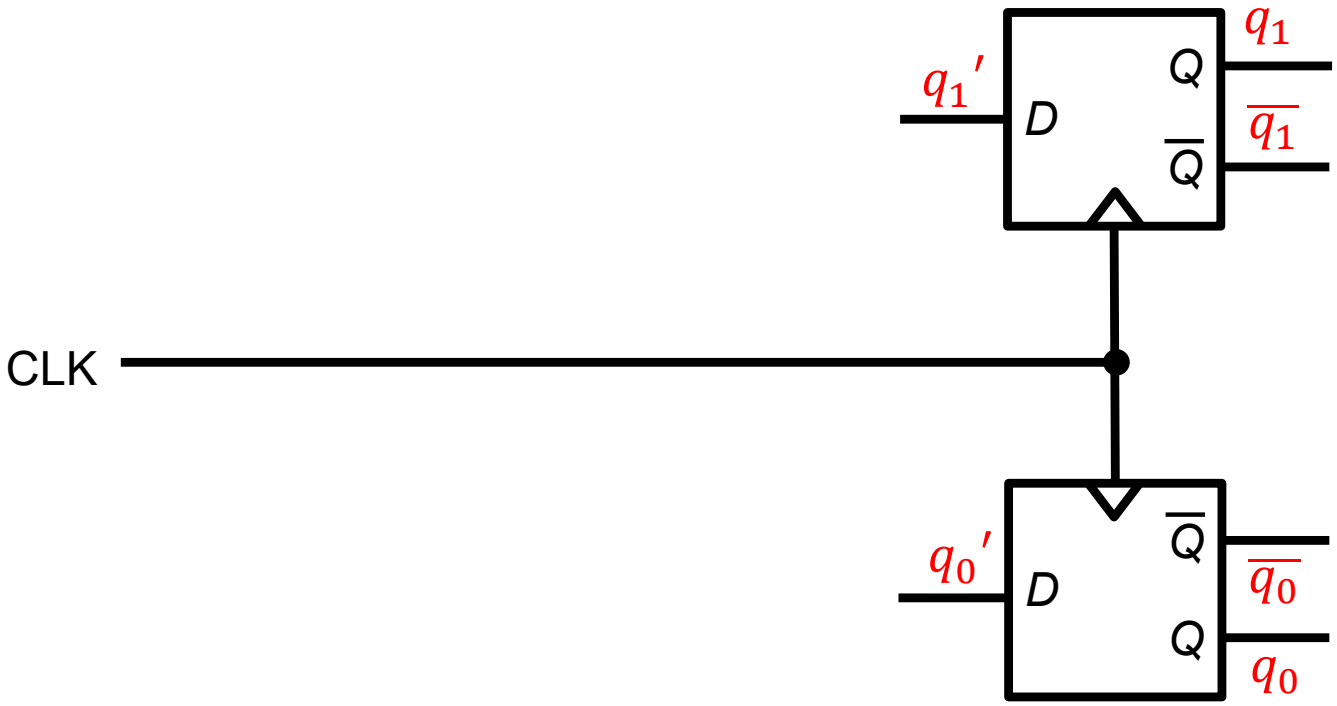
Función de salida

q_1	q_0	z
0	0	0
0	1	0
1	0	0
1	1	1

requiere 2 puertas menos

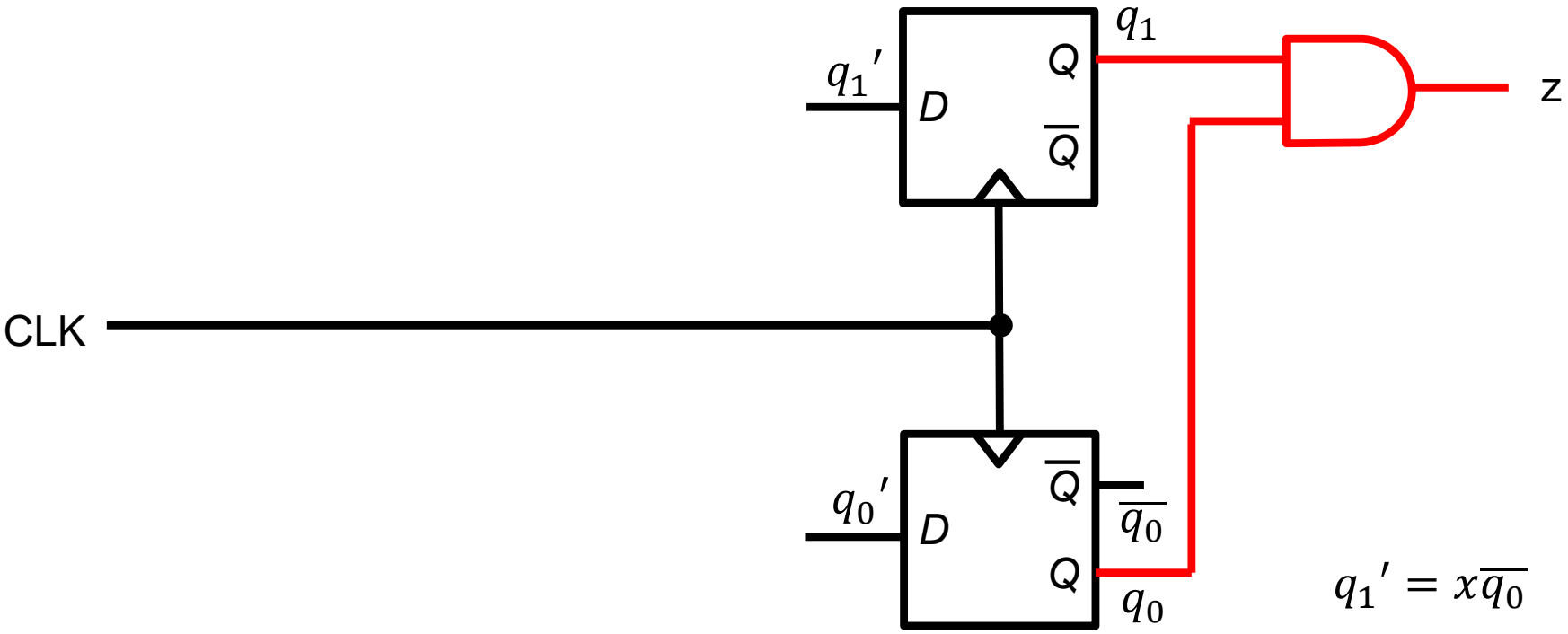
$$q_1' = x\overline{q_0}$$
$$q_0' = x(q_0+q_1)$$
$$z = q_1 q_0$$

Síntesis con biestables D



$$q_1{}' = x\overline{q_0}$$
$$q_0{}' = x(q_0+q_1)$$
$$z = q_1 q_0$$

Síntesis con biestables D

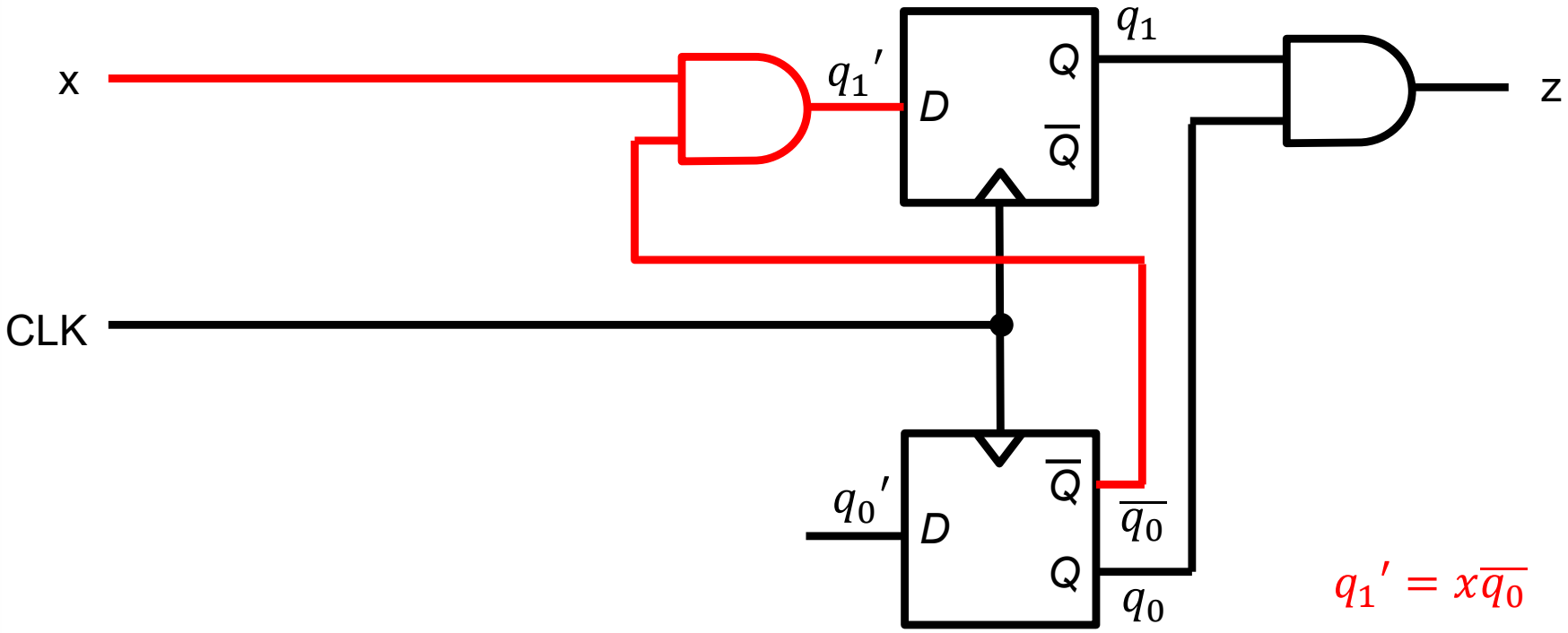


$$q_1{}' = x\overline{q_0}$$

$$q_0{}' = x(q_0 + q_1)$$

$$z = q_1 q_0$$

Síntesis con biestables D

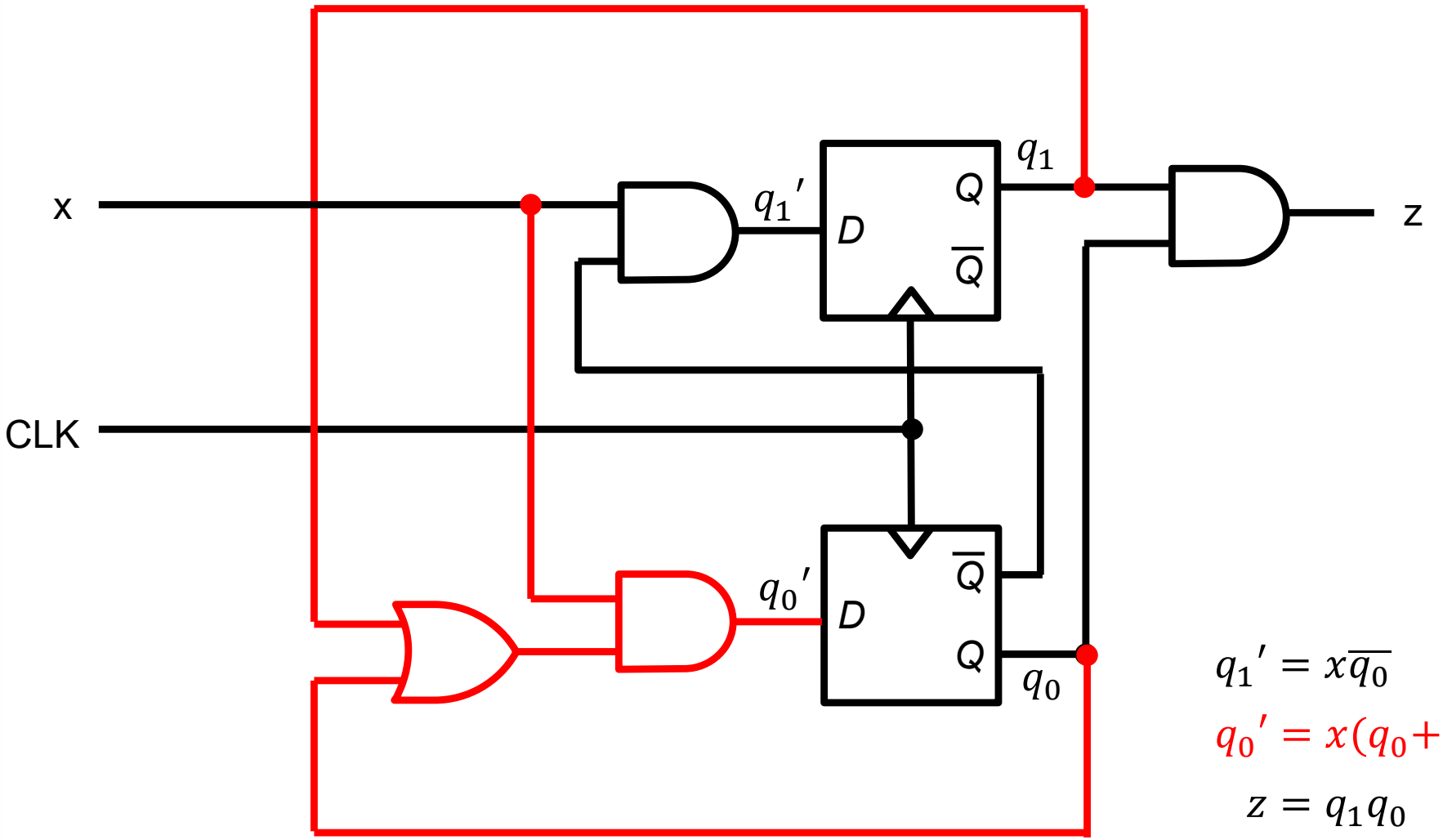


$$q_1' = x\overline{q_0}$$

$$q_0' = x(q_0 + q_1)$$

$$z = q_1 q_0$$

Síntesis con biestables D

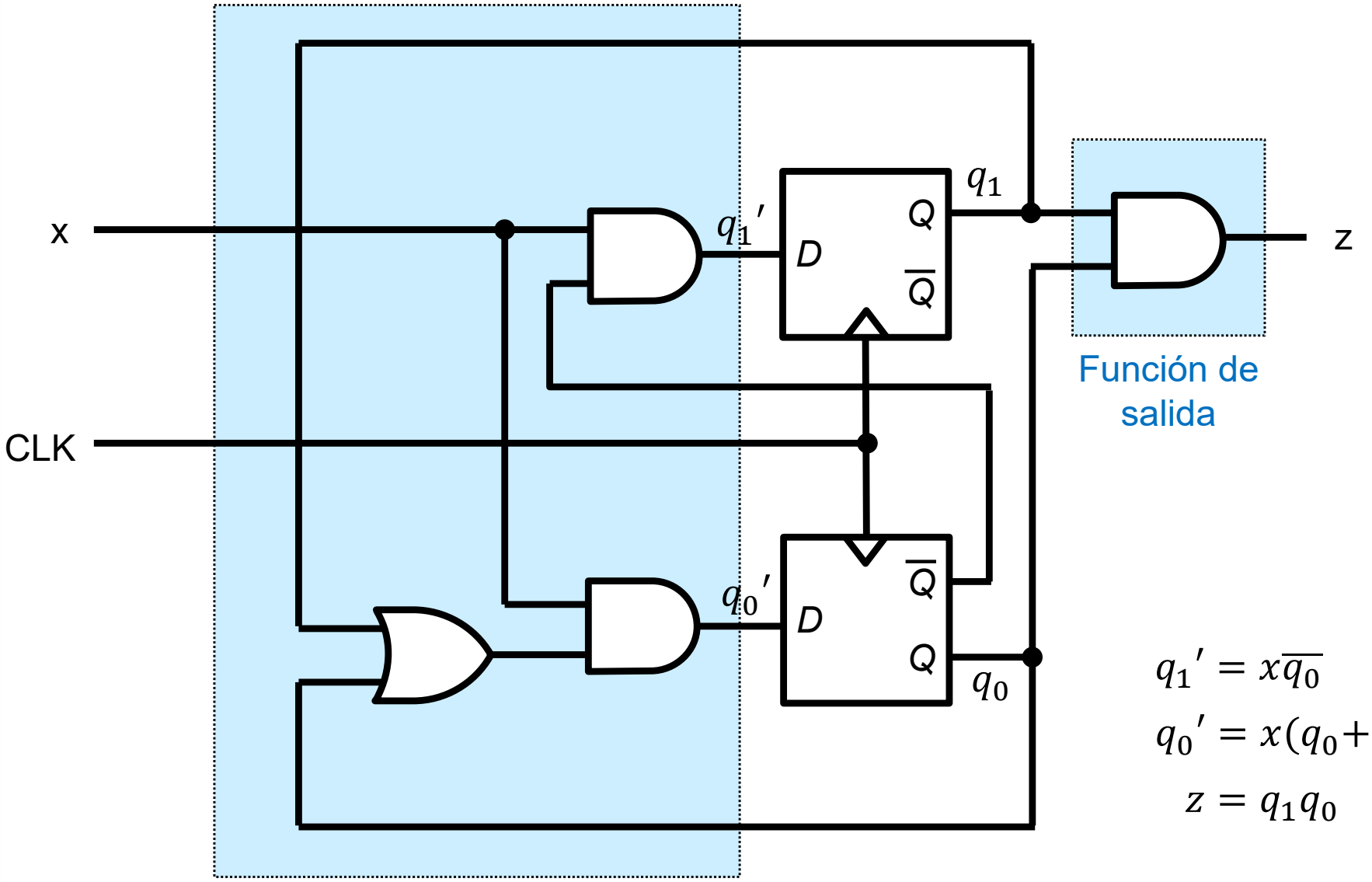


$$q_1' = x\overline{q_0}$$

$$q_0' = x(q_0+q_1)$$

$$z = q_1 q_0$$

Síntesis con biestables D



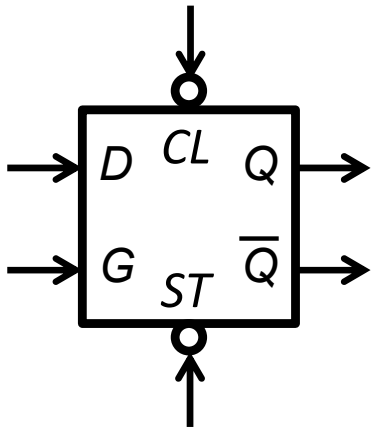
$$q_1' = x\overline{q_0}$$
$$q_0' = x(q_0+q_1)$$
$$z = q_1 q_0$$

Inicialización

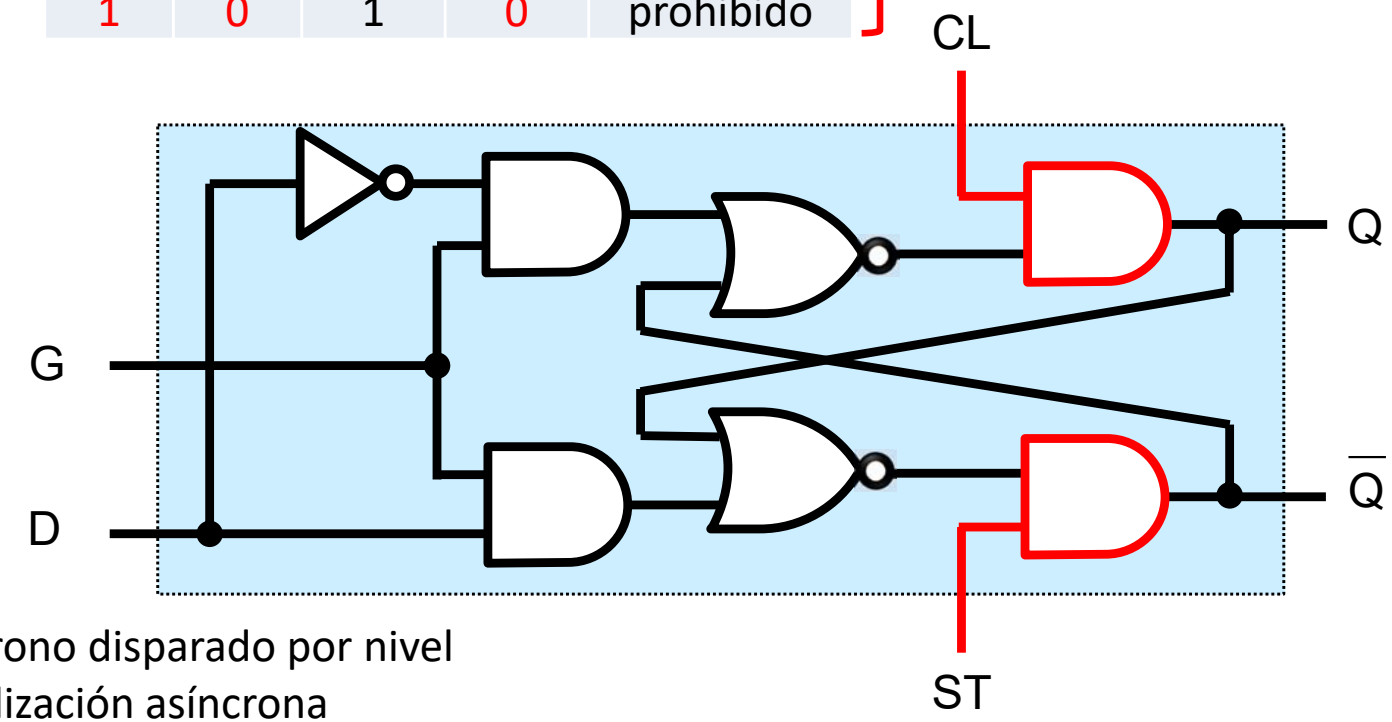
- ¿Cual es el estado de un biestable al encenderlo?
  - Cualquiera de los 2 posibles.
- ¿Cual es el estado de un sistema secuencial al encenderlo?
  - Cualquiera de los posibles (incluyendo prohibidos).

- Todos los sistemas secuenciales tienen una entrada de inicialización para llevarlos asíncronamente a un estado inicial.
  - Esta entrada global de reset deberá conectarse según la codificación del estado inicial a la entrada de inicialización que corresponda de cada uno de los biestables del circuito.

Inicialización



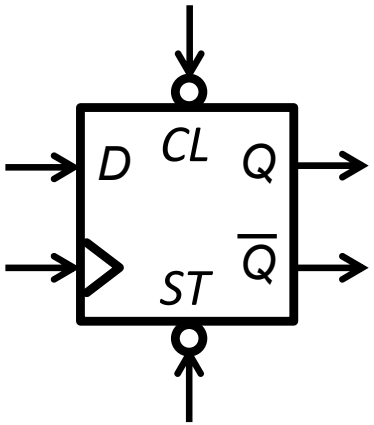
G(t)	D(t)	CL(t)	ST(t)	Q(t+Δt)	
1	0	X	1	0	*funcionamiento normal*
1	1	1	X	1	
0	X	1	1	Q(t)	
0	X	0	1	0	*inicialización*
0	X	1	0	1	
X	X	0	0	prohibido	*entradas contradictorias*
1	1	0	1	prohibido	
1	0	1	0	prohibido	

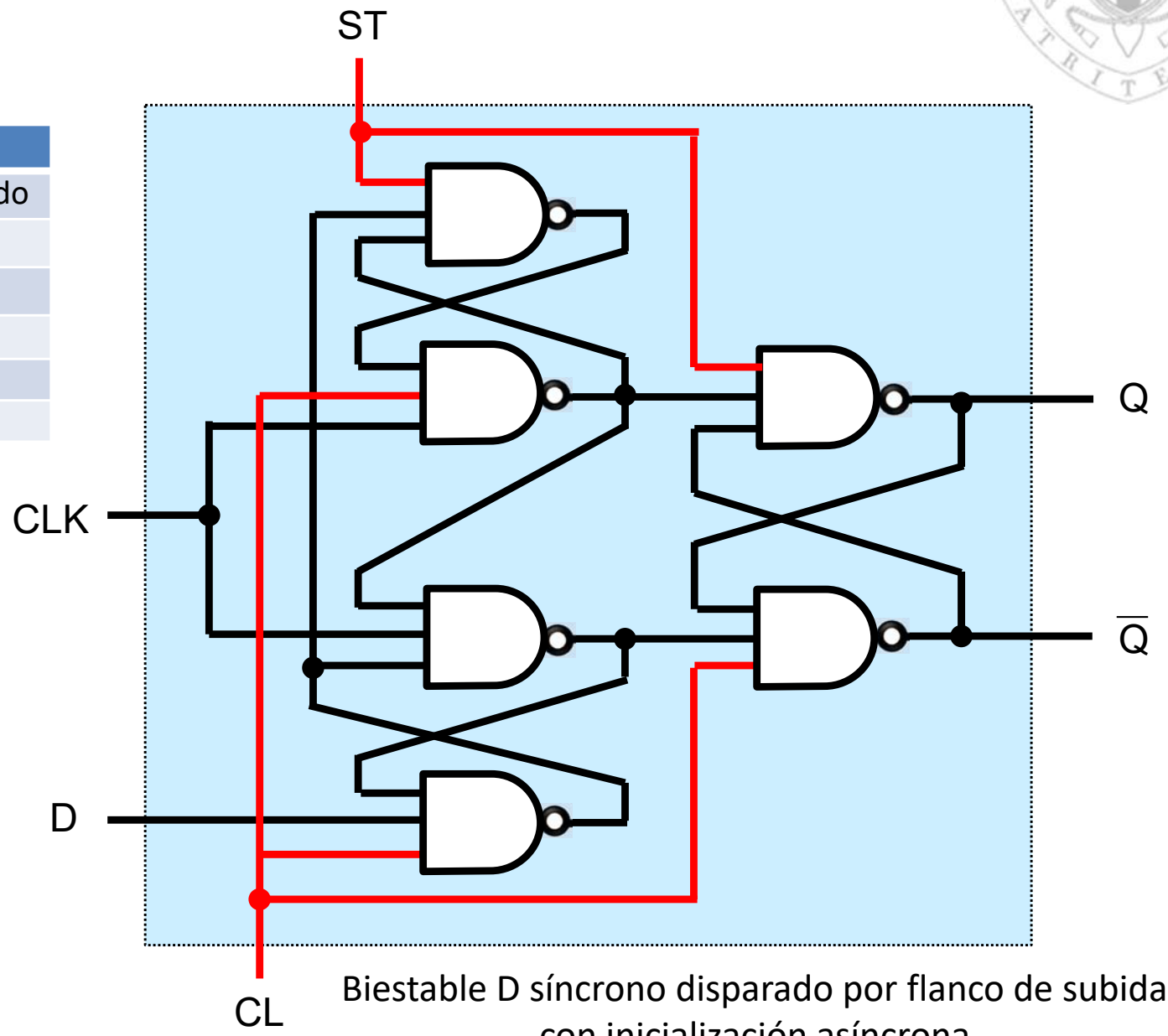


Biestable D síncrono disparado por nivel con inicialización asíncrona

Inicialización

D	CLK	CL	ST	Q'
X	X	0	0	prohibido
X	X	0	1	0
X	X	1	0	1
0	↑	1	1	0
1	↑	1	1	1
resto		1	1	Q

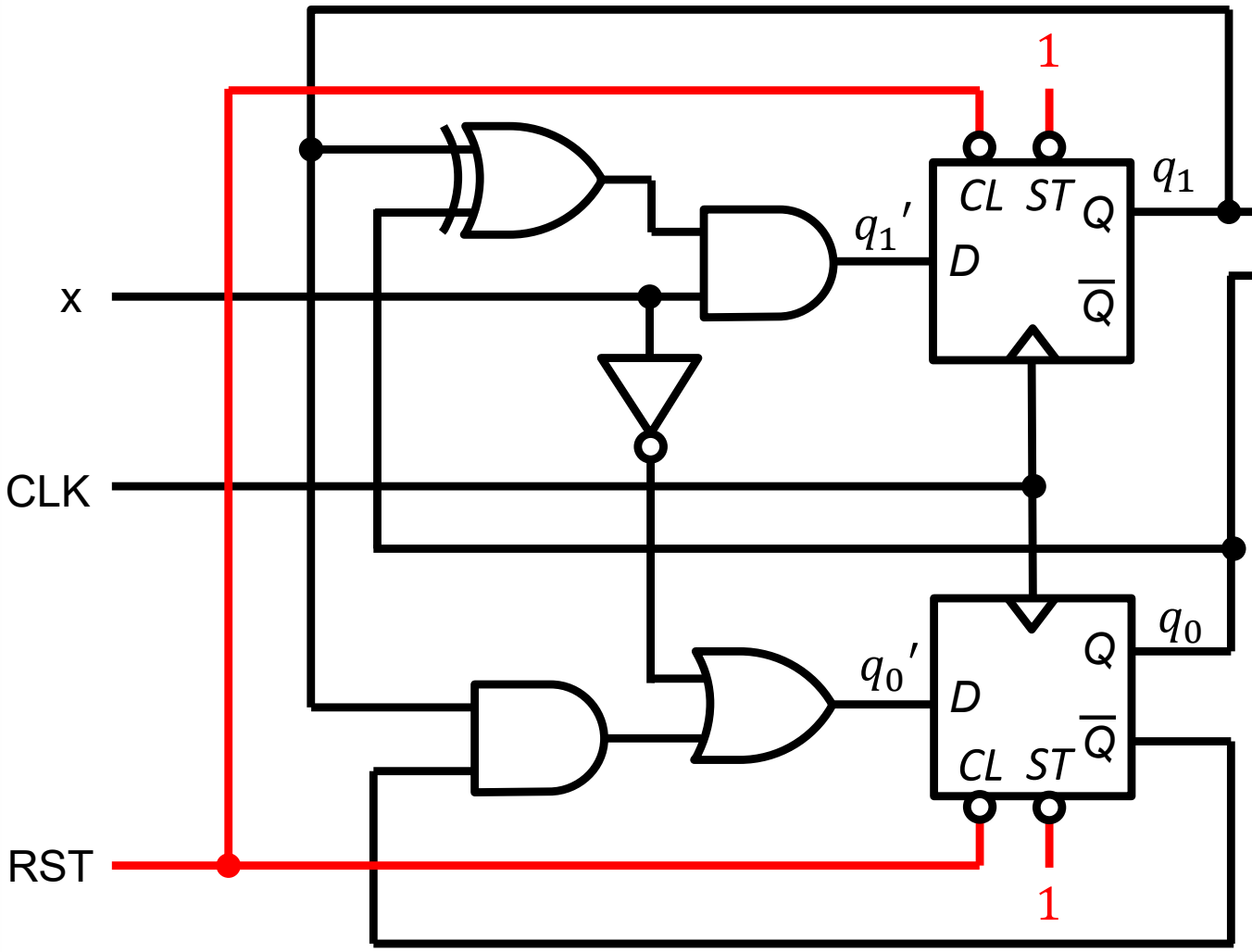




Biestable D síncrono disparado por flanco de subida con inicialización asíncrona

Inicialización

- El reset se distribuye según la codificación del estado inicial:



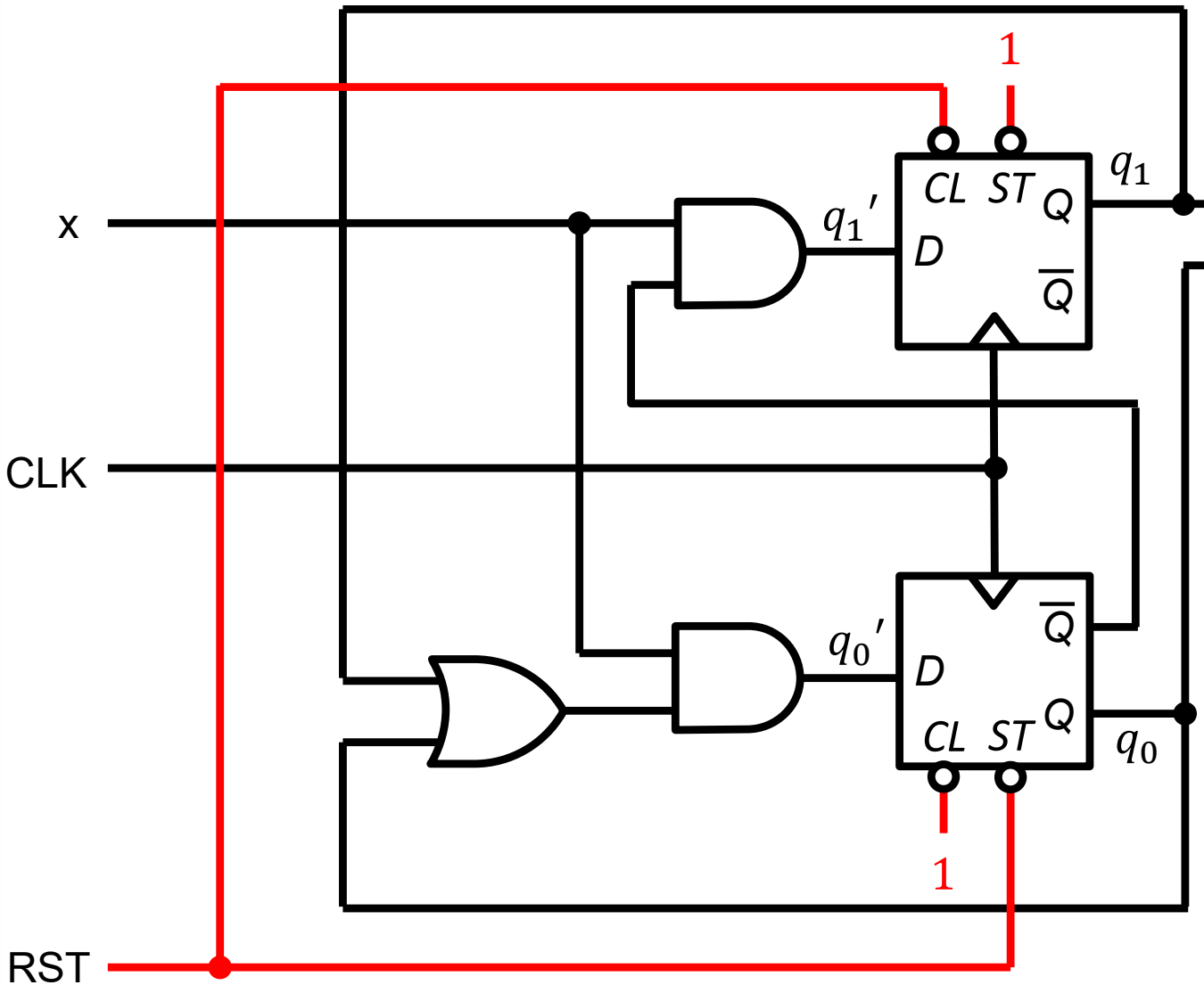
$$q_1' = x(q_1 \oplus q_0)$$

$$q_0' = \bar{x} + q_1\overline{q_0}$$

$$z = q_1 q_0$$

Inicialización

- El reset se distribuye según la codificación del estado inicial.



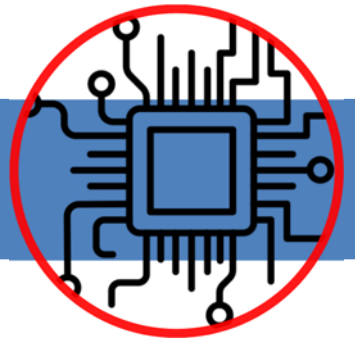
$$q_1' = x\overline{q_0}$$

$$q_0' = x(q_0 + q_1)$$

$$z = q_1 q_0$$

- Tecnología CMOS.
- Retardo.
- Metaestabilidad.
- Reglas de diseño.
- Generación de reloj.
- Generación de reset.
- Biblioteca de celdas.
- Cálculo del tiempo de ciclo.
- Simulación.

Apéndice tecnológico

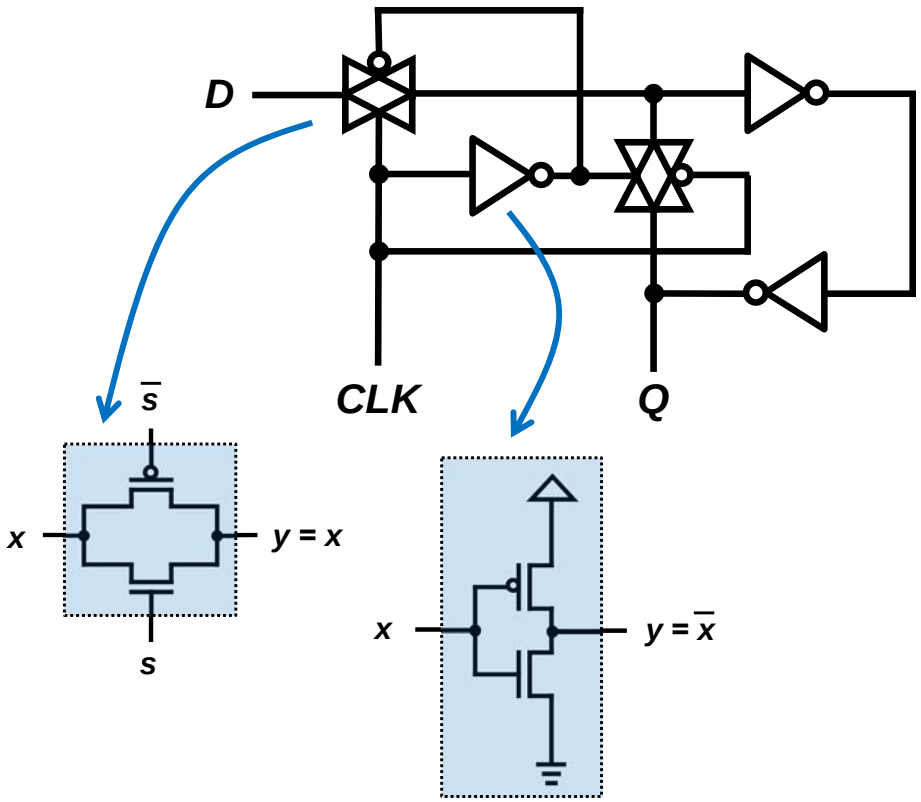


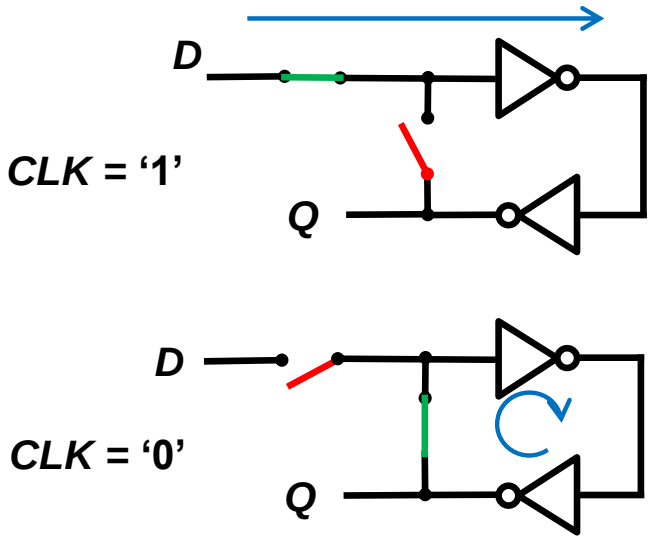
Tecnología CMOS

Biestables

- Un **biestable CMOS** suele implementarse realimentando parejas de inversores a través de puestas de paso.

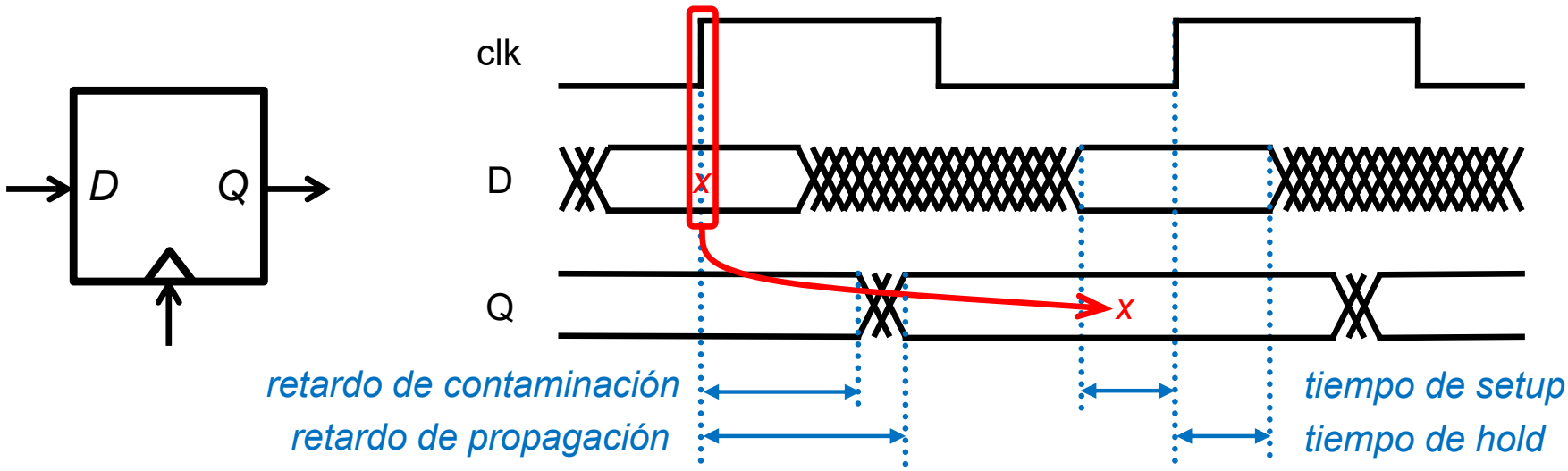
Latch D CMOS





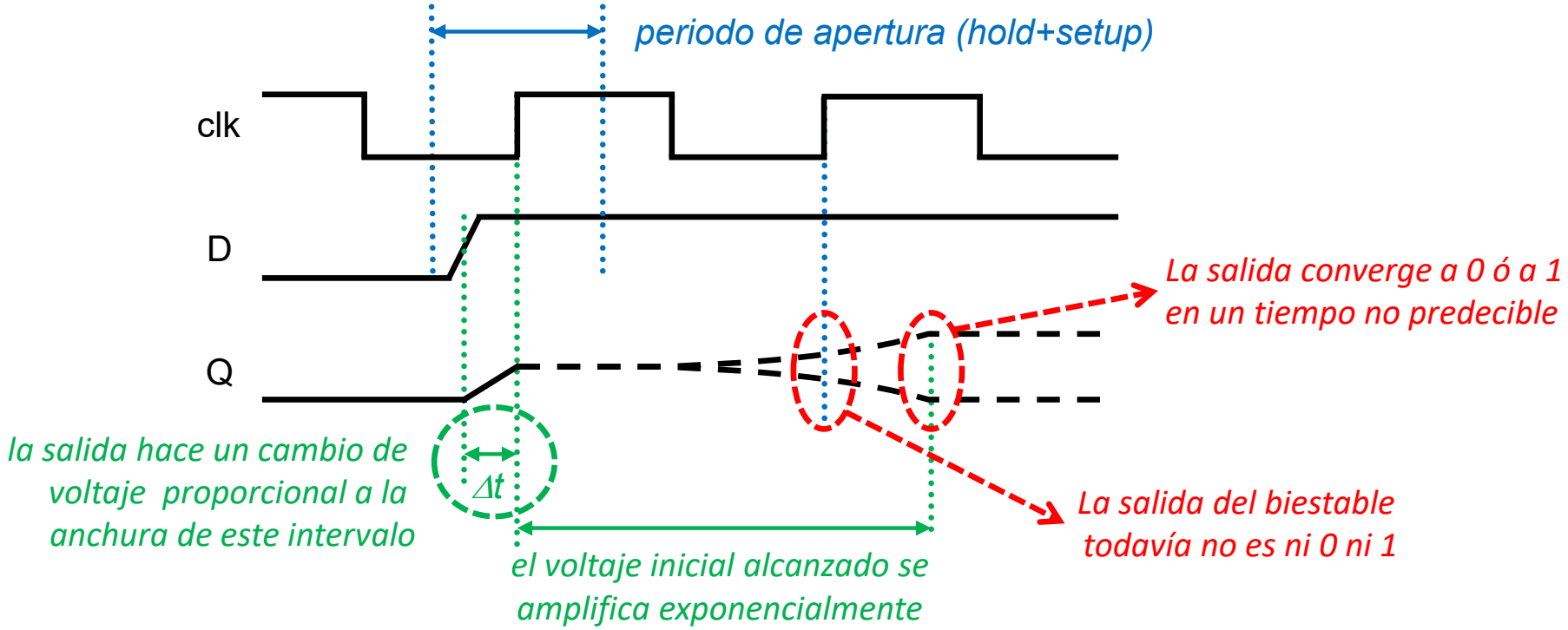
Retardo

- En un biestable disparado por flanco, las salidas cambian en respuesta al flanco del reloj y NO al cambio de la entrada.
  - los retardos se miden desde dicho flanco, el máximo se denomina retardo de propagación y el mínimo, retardo de contaminación.
- Además, para que tenga un comportamiento predecible, la entrada debe estar estable en las proximidades del flanco:
  - Como mínimo debe estar estable durante el tiempo de setup (antes del flanco) y durante tiempo de hold (después del flanco).



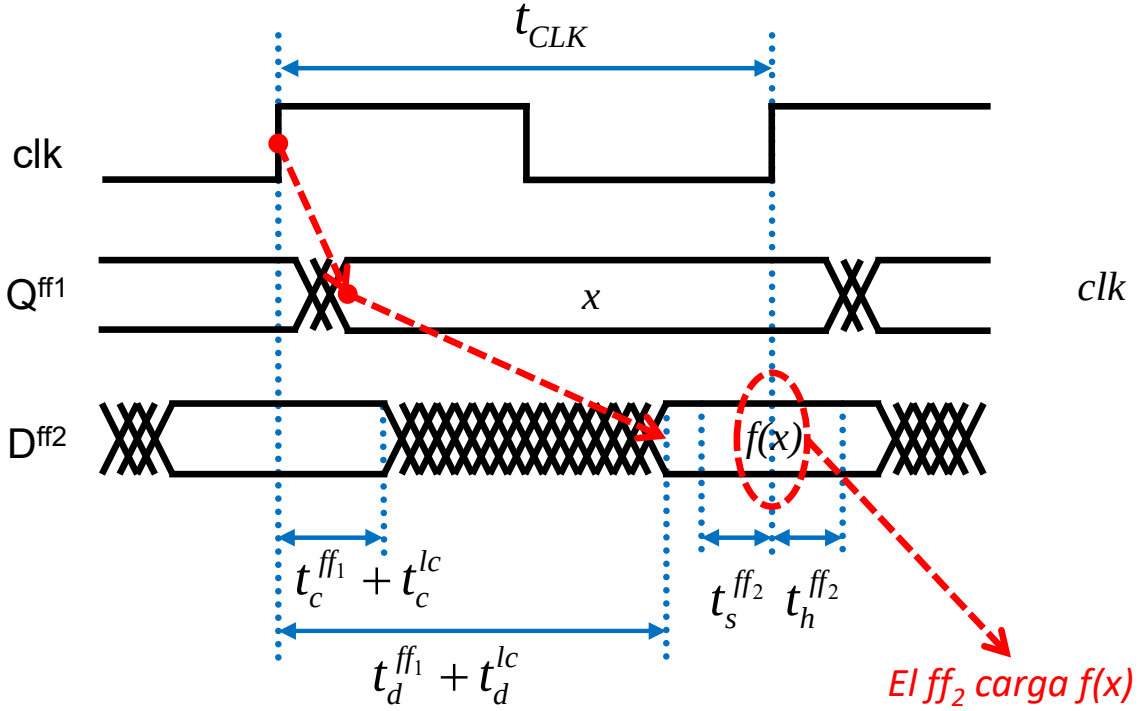
Metaestabilidad

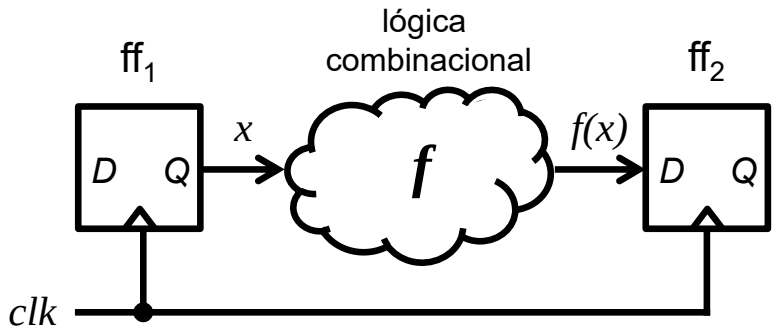
- Cuando se viola el tiempo de hold o el de setup, el biestable entra en un estado metaestable caracterizado por:
  - El retardo de propagación del biestable no está acotado.
  - El valor de salida del biestable es impredecible y, por tanto, se propagan en cadena valores inconsistentes por todo el circuito.



Reglas de diseño

- Por tanto, un sistema secuencial con temporización por flanco tendrá un comportamiento correcto si:
  - El tiempo de ciclo del reloj es lo suficientemente largo para que todos los sistemas combinacionales alcancen su régimen permanente.
  - Las entradas de todos los biestables permanecen estables durante su periodo de apertura (hold+setup).





ligadura de retardo máximo:

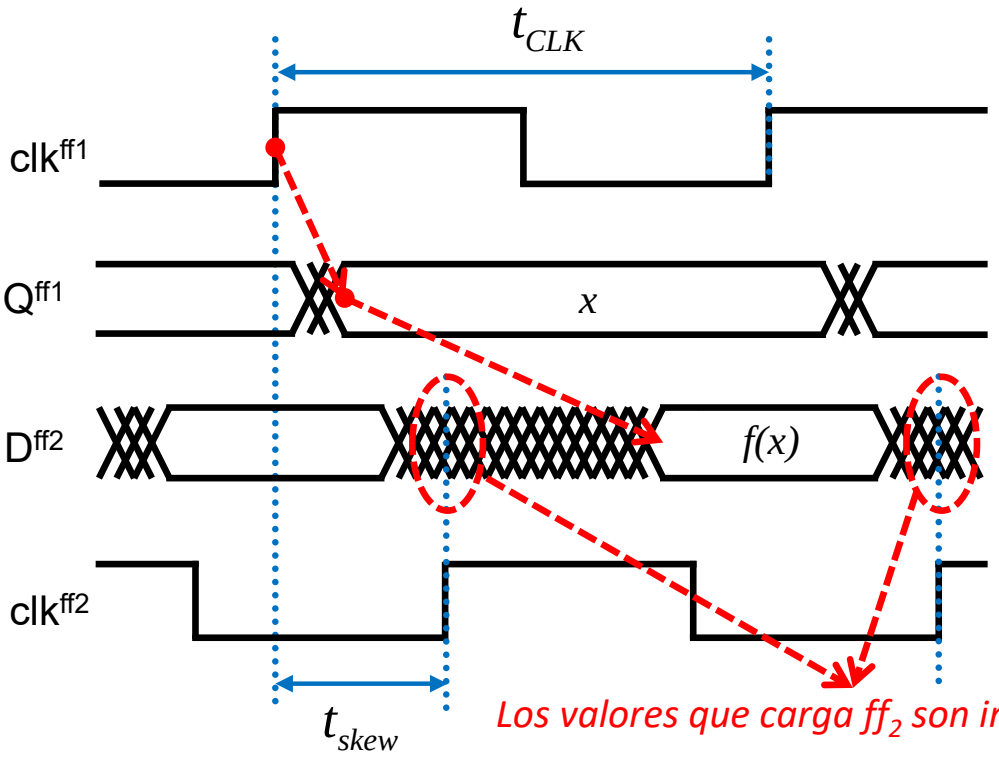
$$t_{CLK} \geq (t_d^{ff_1} + t_d^{lc} + t_s^{ff_2})$$

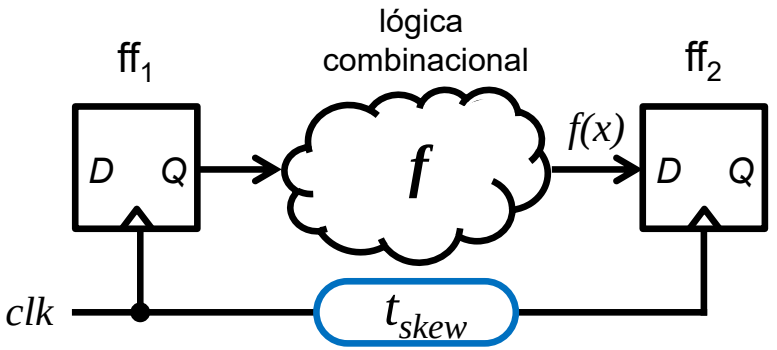
ligadura de retardo mínimo:

$$(t_c^{ff_1} + t_c^{lc}) \geq t_h^{ff_2}$$

Reglas de diseño

- No obstante, un sistema correctamente temporizado puede fallar si la señal de reloj no se distribuye adecuadamente:
  - Si la señal de reloj llega con retraso (skew) a las entradas de reloj de algunos flip-flops, el sistema se desincroniza.
  - Ídem si la frecuencia del reloj no es perfectamente regular (jitter).



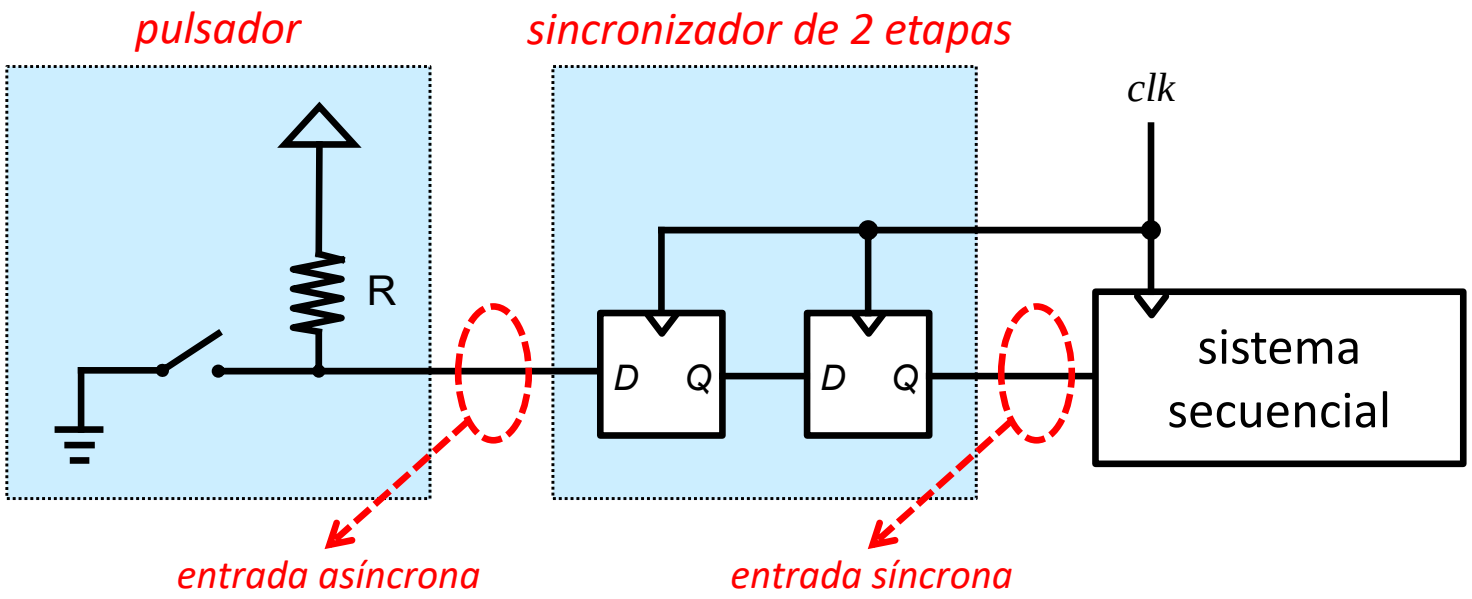


- Distinta longitud de cable
- Ruido (interferencias)
- Diferente carga local
- Variaciones locales de temperatura
- etc...

Los valores que carga ff_2 son impredecibles

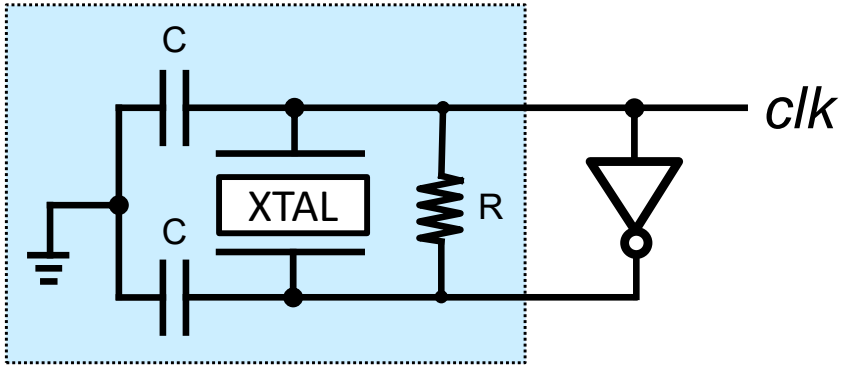
Reglas de diseño

- Asimismo, un sistema también puede fallar si sus entradas cambian asíncronamente porque están conectadas a:
  - Las salidas de otro sistema con distinto reloj.
  - Un dispositivo puramente asíncrono (i.e pulsador)
- En ambos casos es necesario utilizar sincronizadores.
  - El más básico consiste en conectar varios biestables D en cascada.

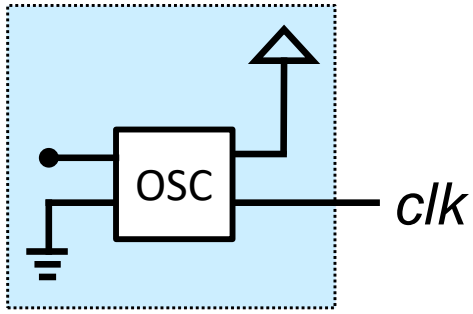


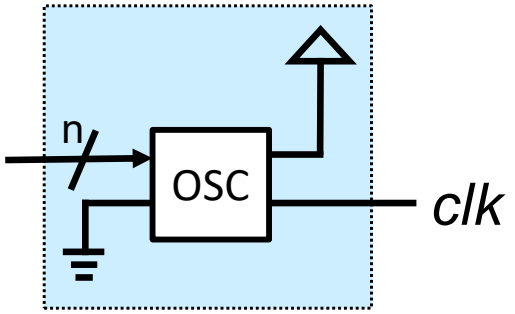
Generación de reloj

- La señal de reloj primaria se genera externamente usando:
  - Cristal de cuarzo



  - Oscilador integrado con frecuencia fija o programable





- Dentro del chip esta señal se acondiciona y se multiplica/divide para obtener la frecuencia deseada.

Generación de reloj

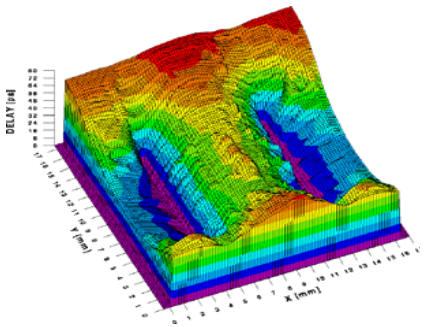
- La frecuencia de reloj aumenta en cada generación tecnológica (el periodo disminuye)

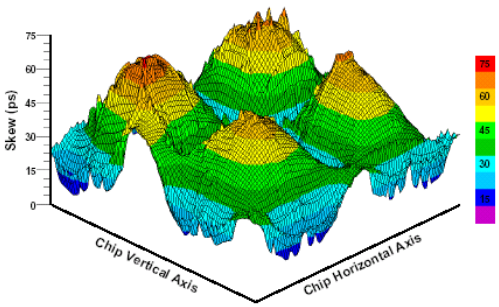
Modelo (Intel)	Año	Frecuencia	Periodo	La luz recorre en ese periodo de tiempo
4004	1971	108 KHz	9.25 μs	2.8 Km
8086	1978	4.77 MHz	0.21 μs	63 m
80386	1985	16 MHz	62.5 ns	19 m
Pentium	1993	66 MHz	15.2 ns	5 m
Pentium 4	2000	1.5 GHz	667 ps	20 cm
Intel Core 2 Quad	2007	2.4 GHz	417 ps	12 cm
Intel Core I7-4770	2013	3.4 GHz	294 ps	9 cm
Intel Core I9-11900X	2021	5.3 GHz	189 ps	6 cm

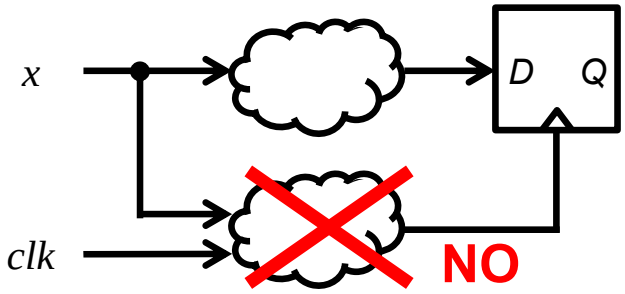
- Si la velocidad de los coches hubiera crecido tan rápido:
  - Coche (1971): 100 Km/h (Madrid-Barcelona: 6h)
  - Coche (2021): 4 900 000 Km/h (Madrid-Barna: 0.5s, Madrid-Plutón: 1h)

Generación de reloj

- El diseño (físico) de una la red de distribución de reloj en un chip es un proceso complejo:
  - La señal de reloj tiene mucha conectividad (tiene mucha carga) y las interconexiones son largas (muy resistivas y vulnerables al ruido).
  - Requiere trazar un árbol /red equilibrada de interconexiones, buffers y otros elementos correctores de desfase (PLL, DLL).
- En general es mala práctica que el reloj atraviese lógica:
  - Introduce un skew variable debido a la incertidumbre de la red.
  - Puede producir glitches que provoquen cambios espurios de estado.
    - Esto también es aplicable al reset, si se desea evitar inicializaciones espurias.

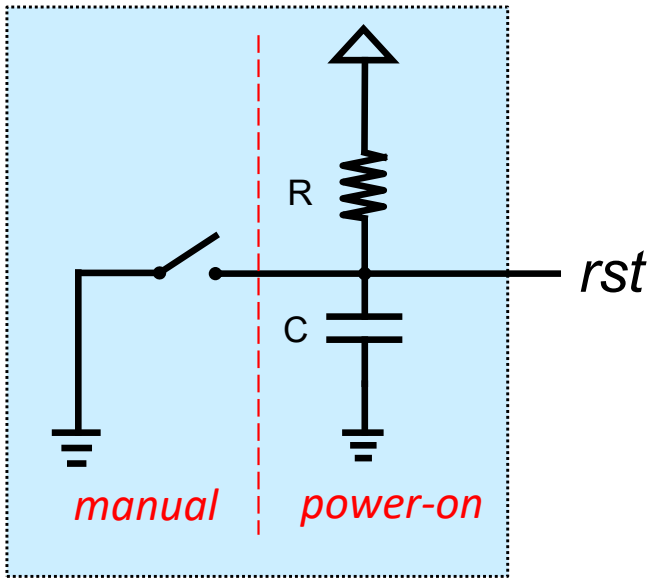




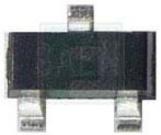


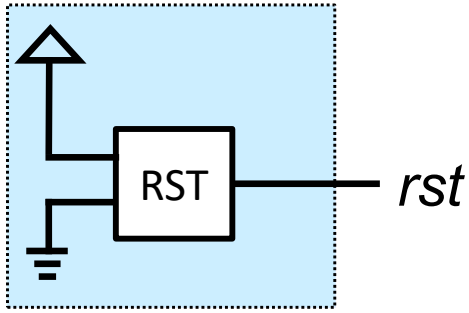
Generación de reset

- La señal de reset se genera externamente usando:
  - Un circuito RC



  - Generador de reset integrado

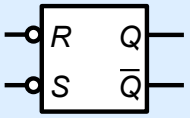
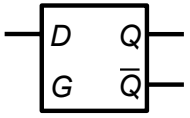
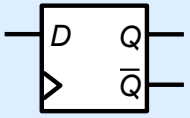
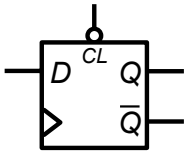
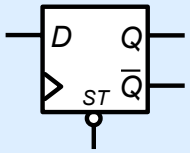
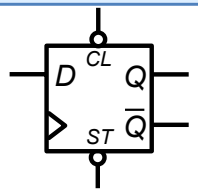




Biblioteca de celdas

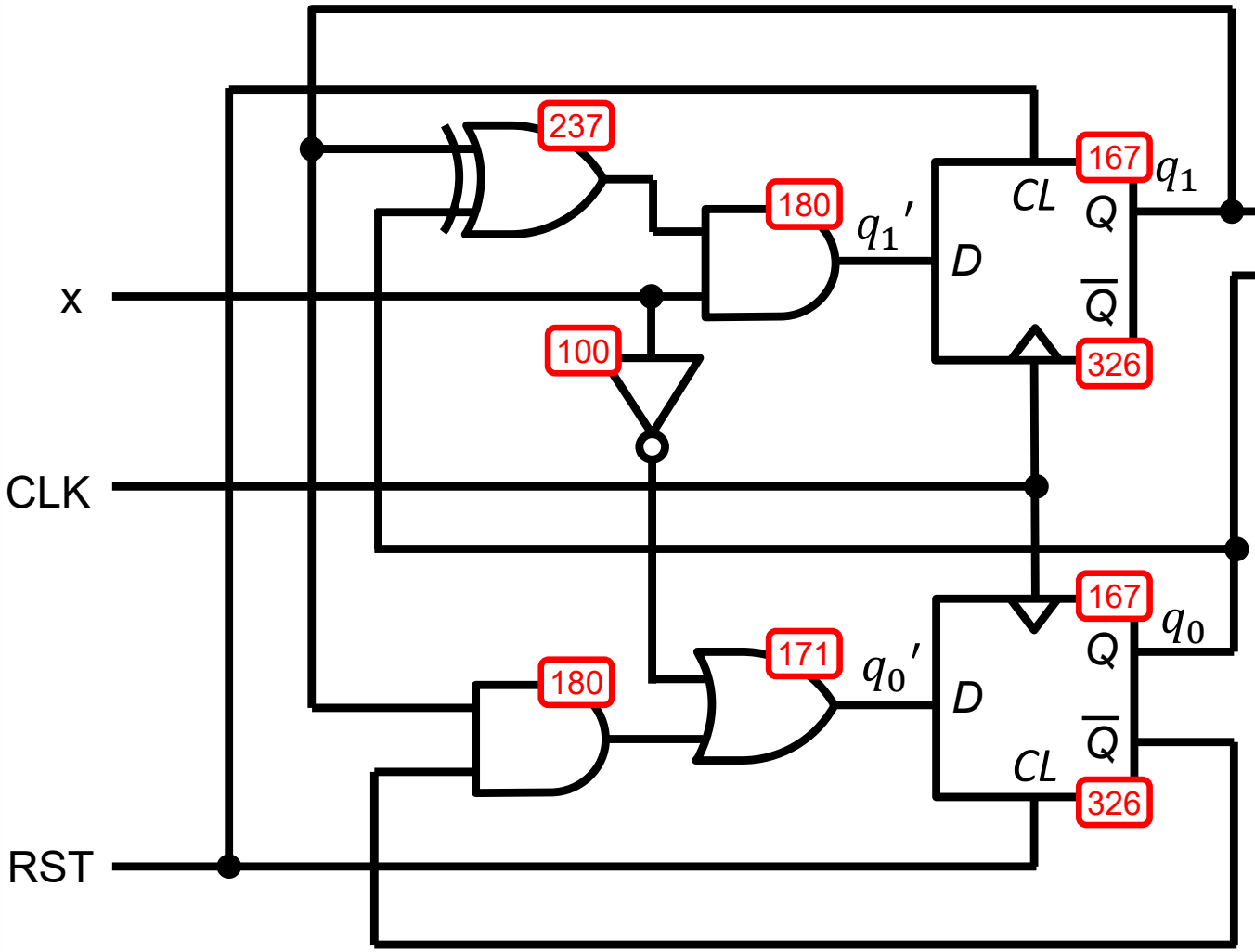
CMOS 90 nm

fuente: Synopsys (SAED EDK 90 nm)

Biestable	Área (μm^2)	Retardo (ps)	Consumo estático (nW)	Consumo dinámico (nW/MHz)
	10.1376	221 (Q) 386 (QN)	621	2359
	22.1184	219 (Q) 234 (QN)	144	463
	24.8832	217 (Q) 193 (QN)	140	284
	32.2560	167 (Q) 326 (QN)	164	281
	31.3344	412 (Q) 372 (QN)	152	161
	35.0208	212 (Q) 365 (QN)	167	215

Cálculo del tiempo de ciclo

CMOS 90 nm

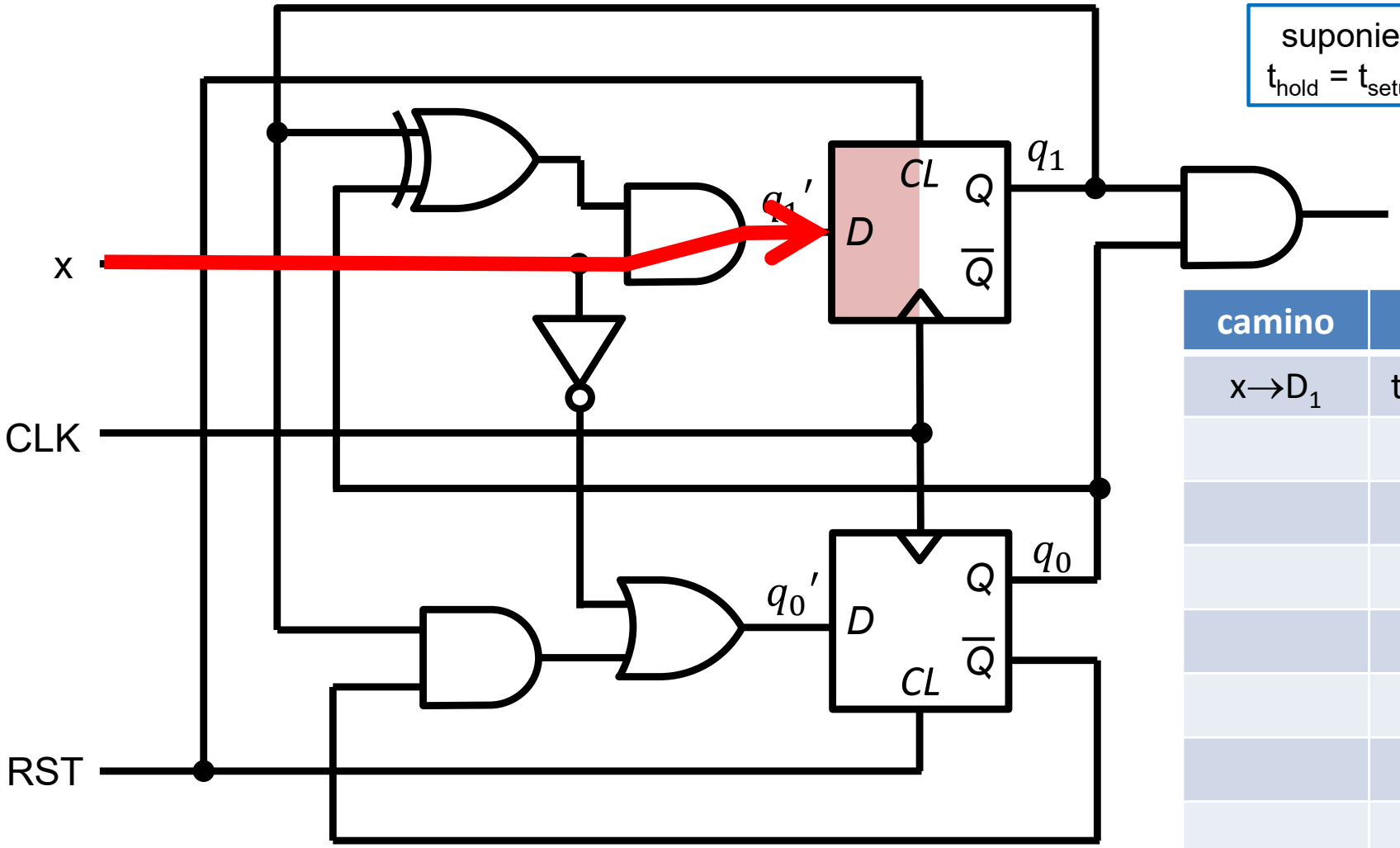


suponiendo:
$t_{hold} = t_{setup} = 0$

camino	retardo

Cálculo del tiempo de ciclo

CMOS 90 nm

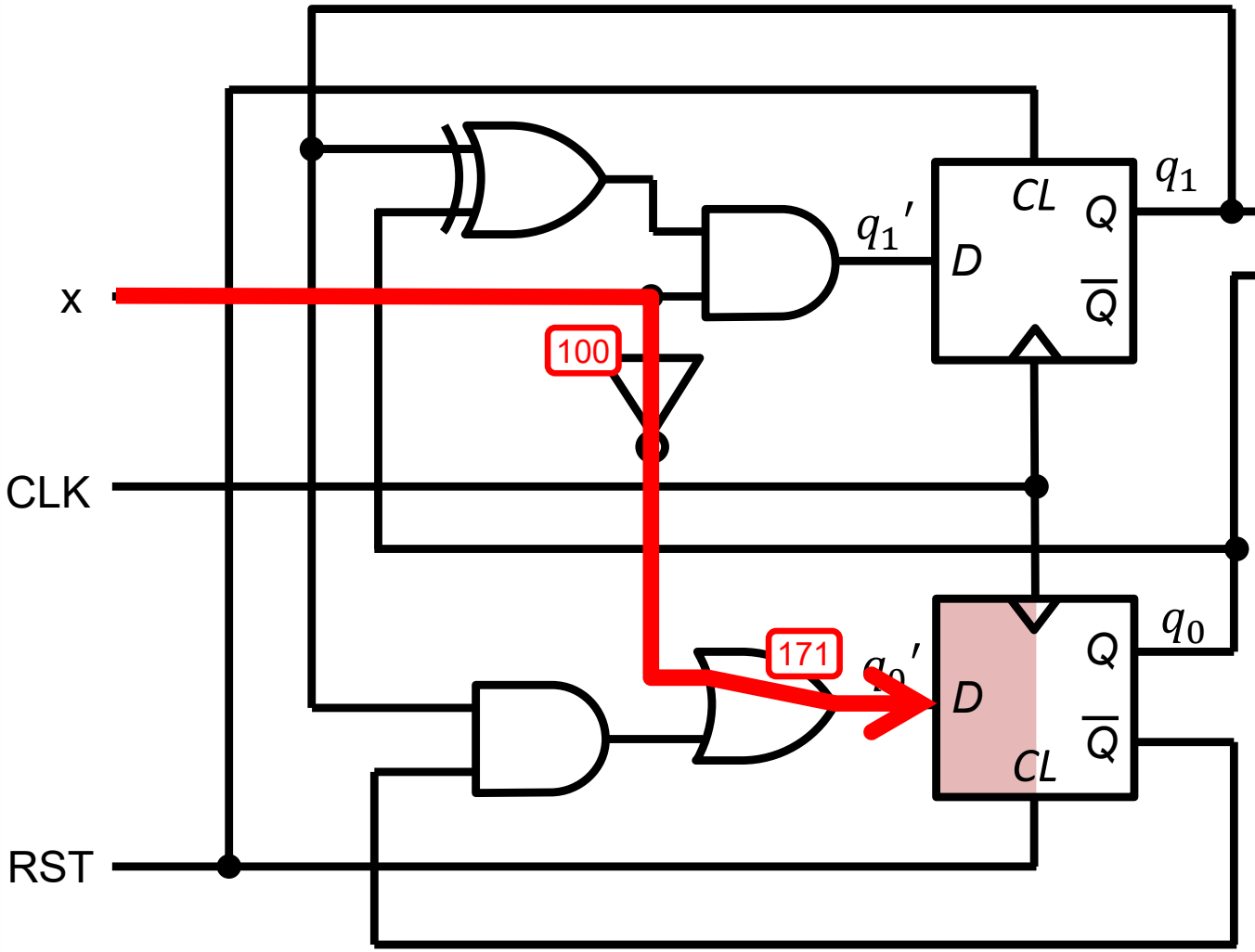


suponiendo:
$t_{hold} = t_{setup} = 0$

camino	retardo
$x \rightarrow D_1$	t_x + 180 ps

Cálculo del tiempo de ciclo

CMOS 90 nm

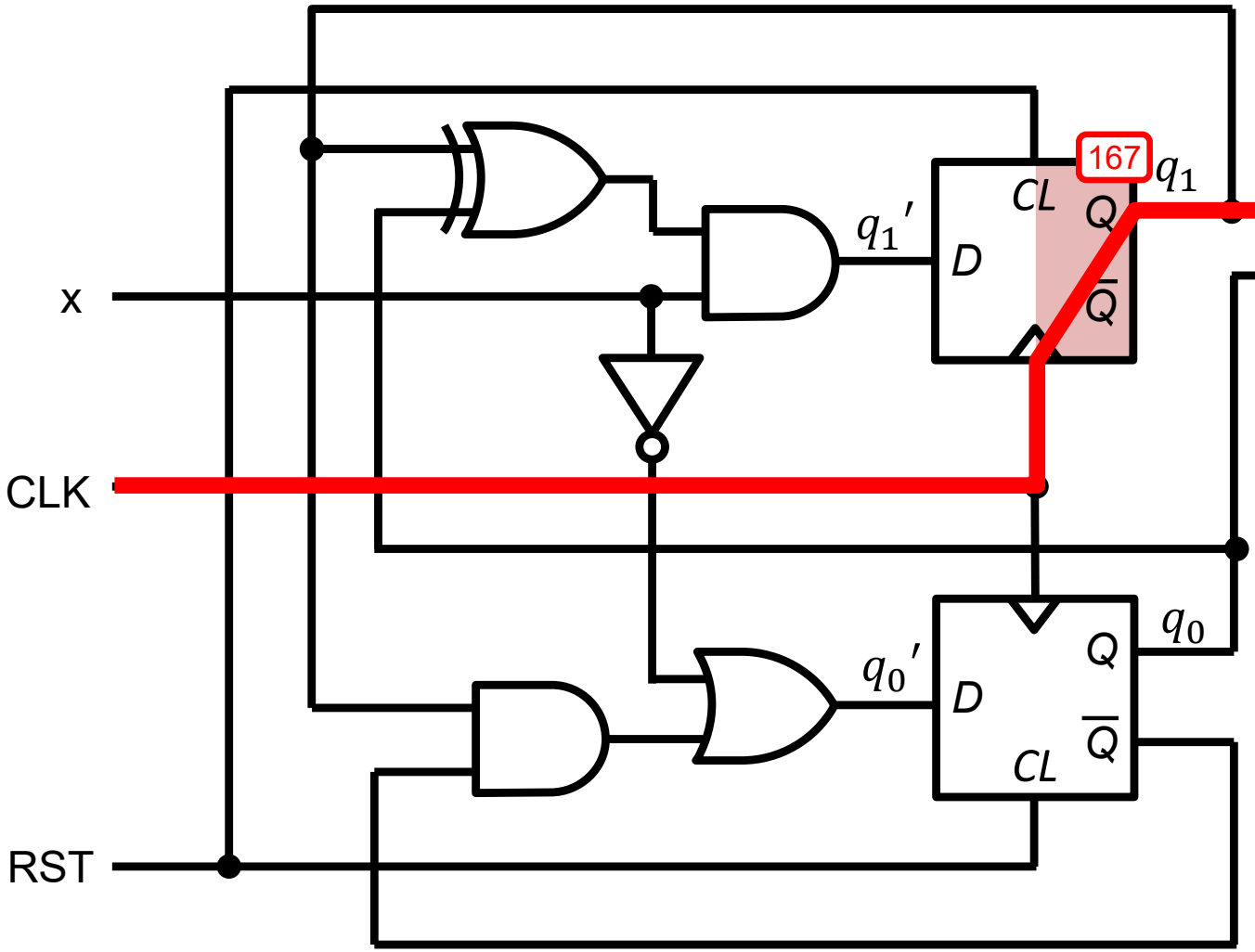


suponiendo:
$t_{hold} = t_{setup} = 0$

camino	retardo
$x \rightarrow D_1$	t_x + 180 ps
$x \rightarrow D_0$	t_x + 271 ps

Cálculo del tiempo de ciclo

CMOS 90 nm

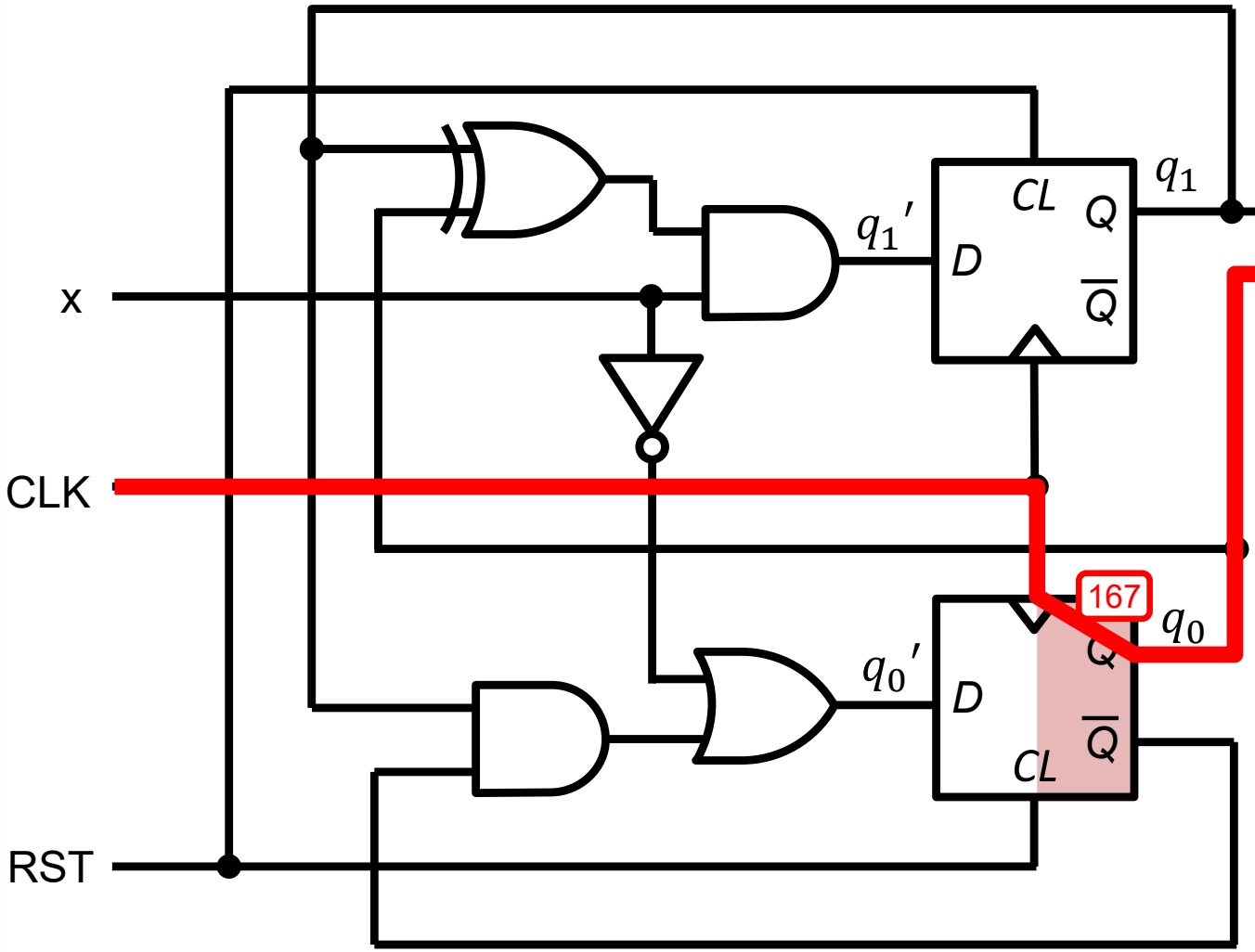


suponiendo:
$t_{hold} = t_{setup} = 0$

camino	retardo
$x \rightarrow D_1$	t_x + 180 ps
$x \rightarrow D_0$	t_x + 271 ps
$CLK_1 \rightarrow z$	347 ps

Cálculo del tiempo de ciclo

CMOS 90 nm

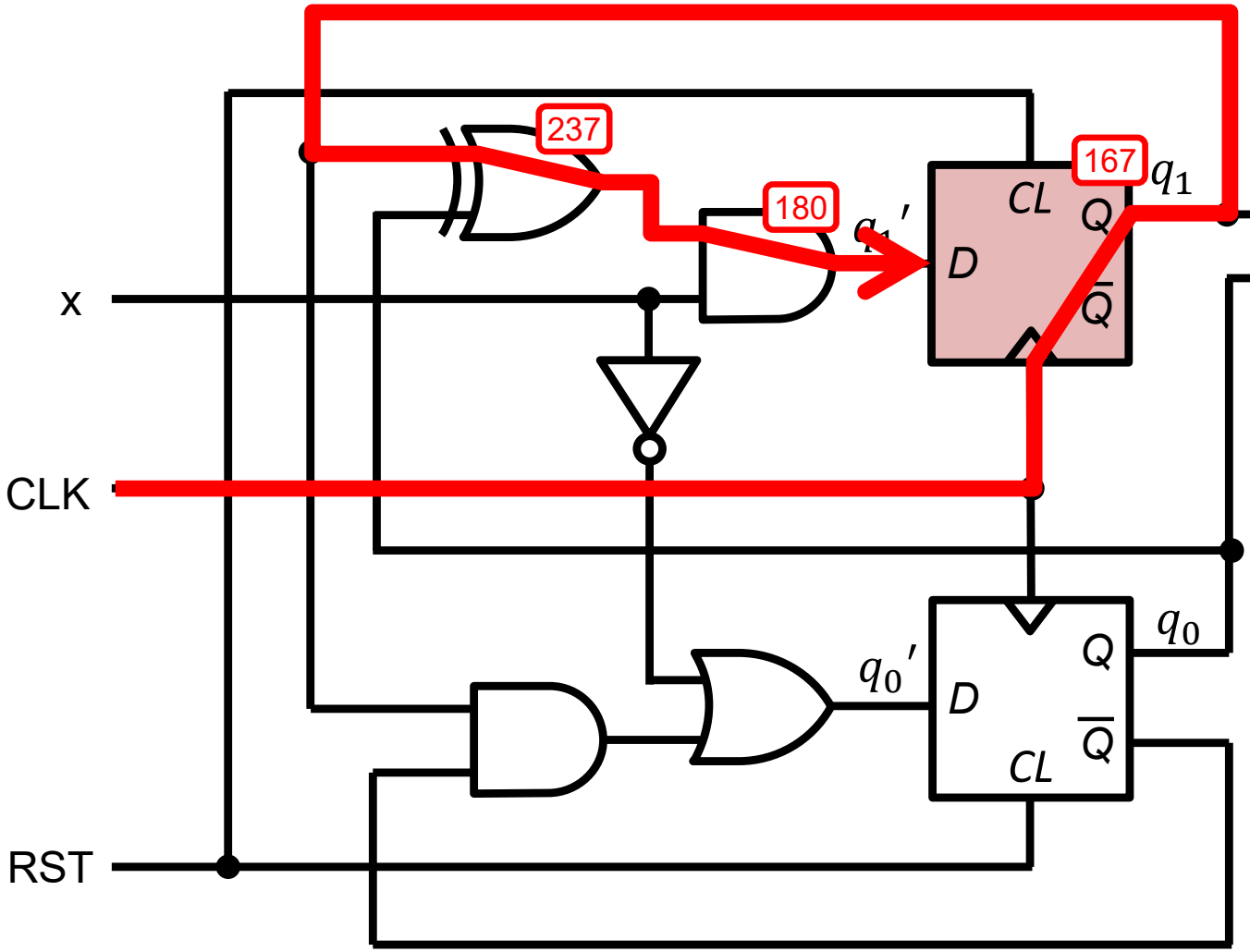


suponiendo:
$t_{hold} = t_{setup} = 0$

camino	retardo
$x \rightarrow D_1$	t_x + 180 ps
$x \rightarrow D_0$	t_x + 271 ps
$CLK_1 \rightarrow z$	347 ps
$CLK_0 \rightarrow z$	347 ps

Cálculo del tiempo de ciclo

CMOS 90 nm

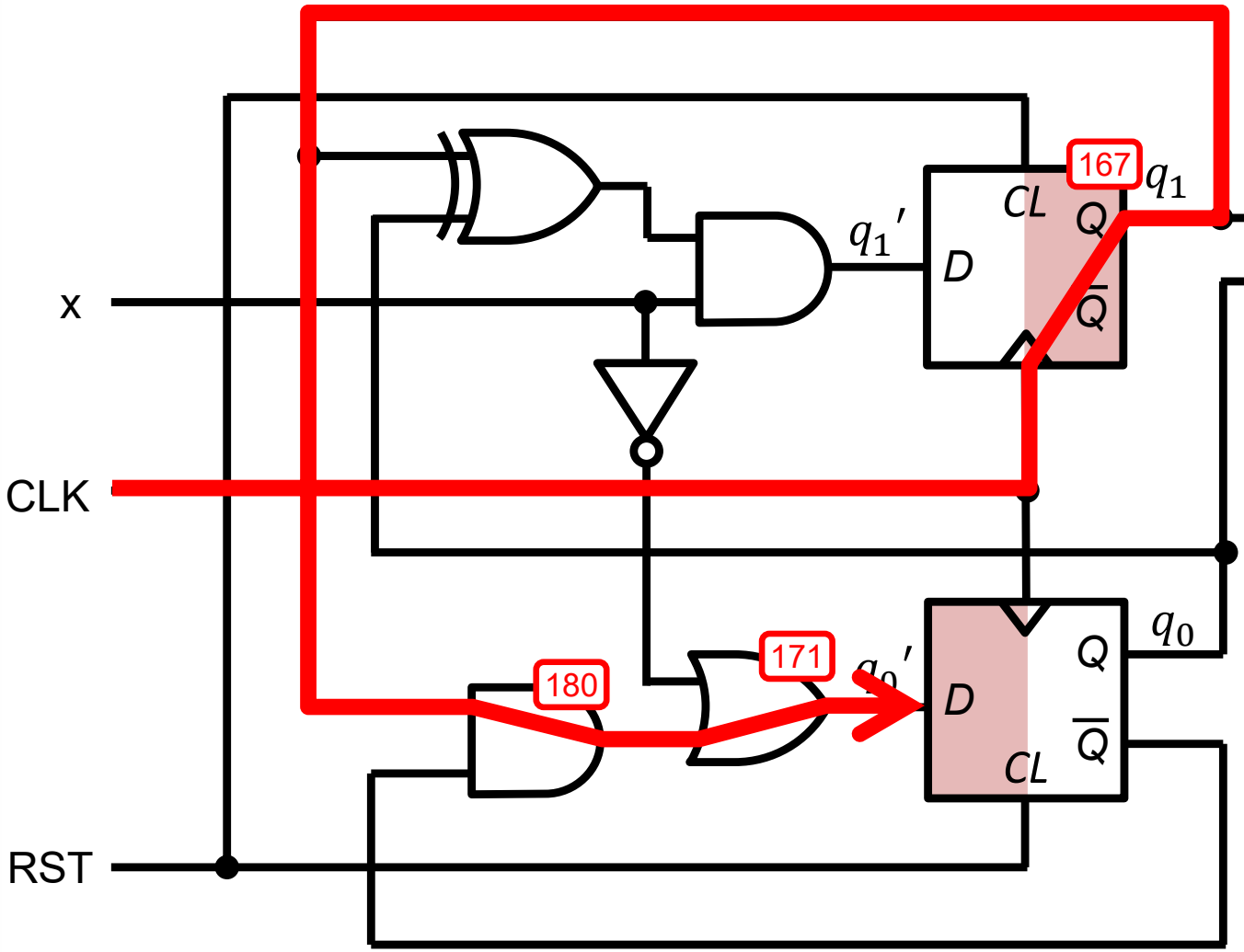


suponiendo:
$t_{hold} = t_{setup} = 0$

camino	retardo
$x \rightarrow D_1$	t_x + 180 ps
$x \rightarrow D_0$	t_x + 271 ps
$CLK_1 \rightarrow z$	347 ps
$CLK_0 \rightarrow z$	347 ps
$CLK_1 \rightarrow D_1$	584 ps

Cálculo del tiempo de ciclo

CMOS 90 nm

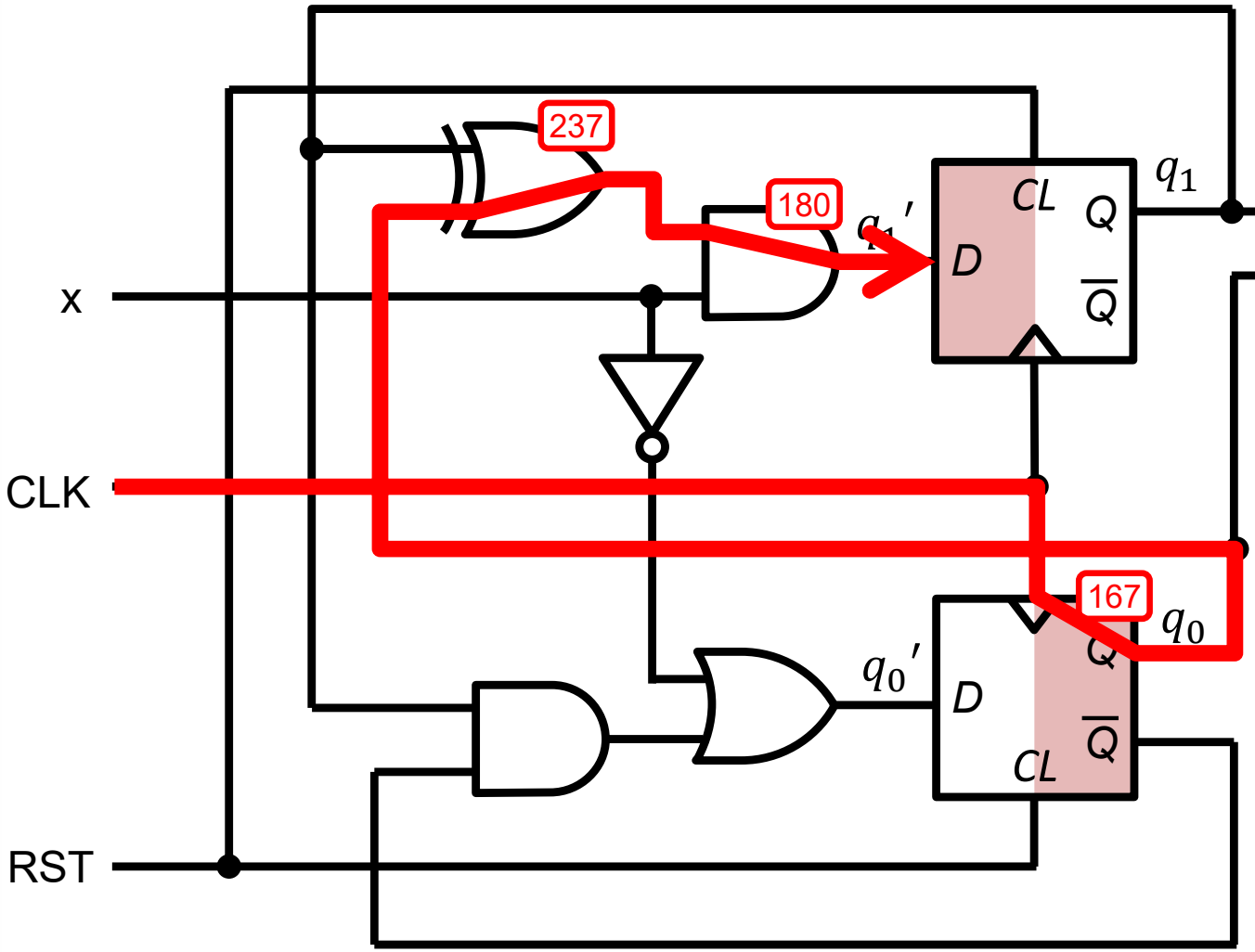


suponiendo:
$t_{hold} = t_{setup} = 0$

camino	retardo
$x \rightarrow D_1$	t_x + 180 ps
$x \rightarrow D_0$	t_x + 271 ps
$CLK_1 \rightarrow z$	347 ps
$CLK_0 \rightarrow z$	347 ps
$CLK_1 \rightarrow D_1$	584 ps
$CLK_1 \rightarrow D_0$	518 ps

Cálculo del tiempo de ciclo

CMOS 90 nm

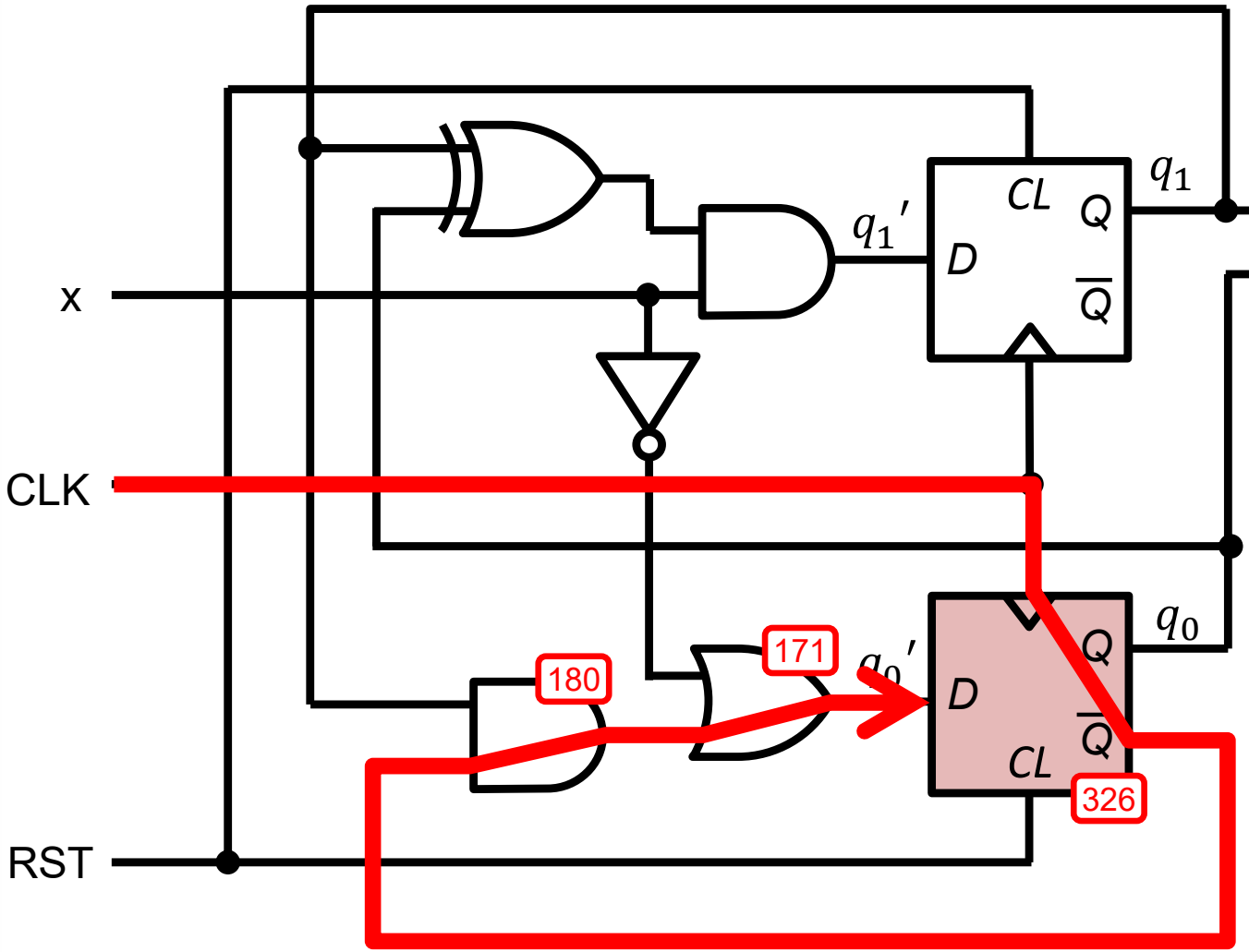


suponiendo:
$t_{hold} = t_{setup} = 0$

camino	retardo
$x \rightarrow D_1$	t_x + 180 ps
$x \rightarrow D_0$	t_x + 271 ps
$CLK_1 \rightarrow z$	347 ps
$CLK_0 \rightarrow z$	347 ps
$CLK_1 \rightarrow D_1$	584 ps
$CLK_1 \rightarrow D_0$	518 ps
$CLK_0 \rightarrow D_1$	584 ps

Cálculo del tiempo de ciclo

CMOS 90 nm

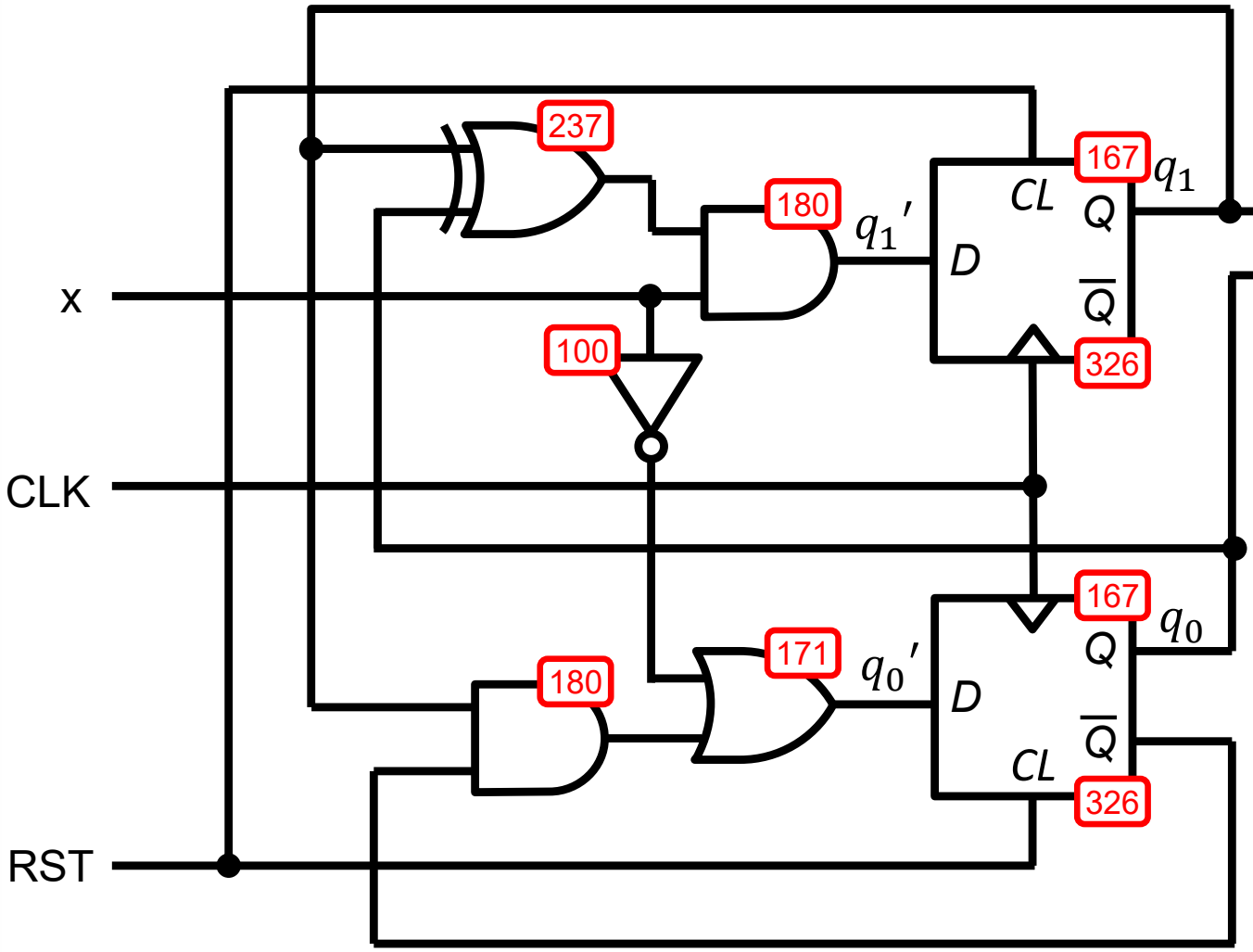


suponiendo:
$t_{hold} = t_{setup} = 0$

camino	retardo
$x \rightarrow D_1$	t_x + 180 ps
$x \rightarrow D_0$	t_x + 271 ps
$CLK_1 \rightarrow z$	347 ps
$CLK_0 \rightarrow z$	347 ps
$CLK_1 \rightarrow D_1$	584 ps
$CLK_1 \rightarrow D_0$	518 ps
$CLK_0 \rightarrow D_1$	584 ps
$CLK_0 \rightarrow D_0$	677 ps

Cálculo del tiempo de ciclo

CMOS 90 nm



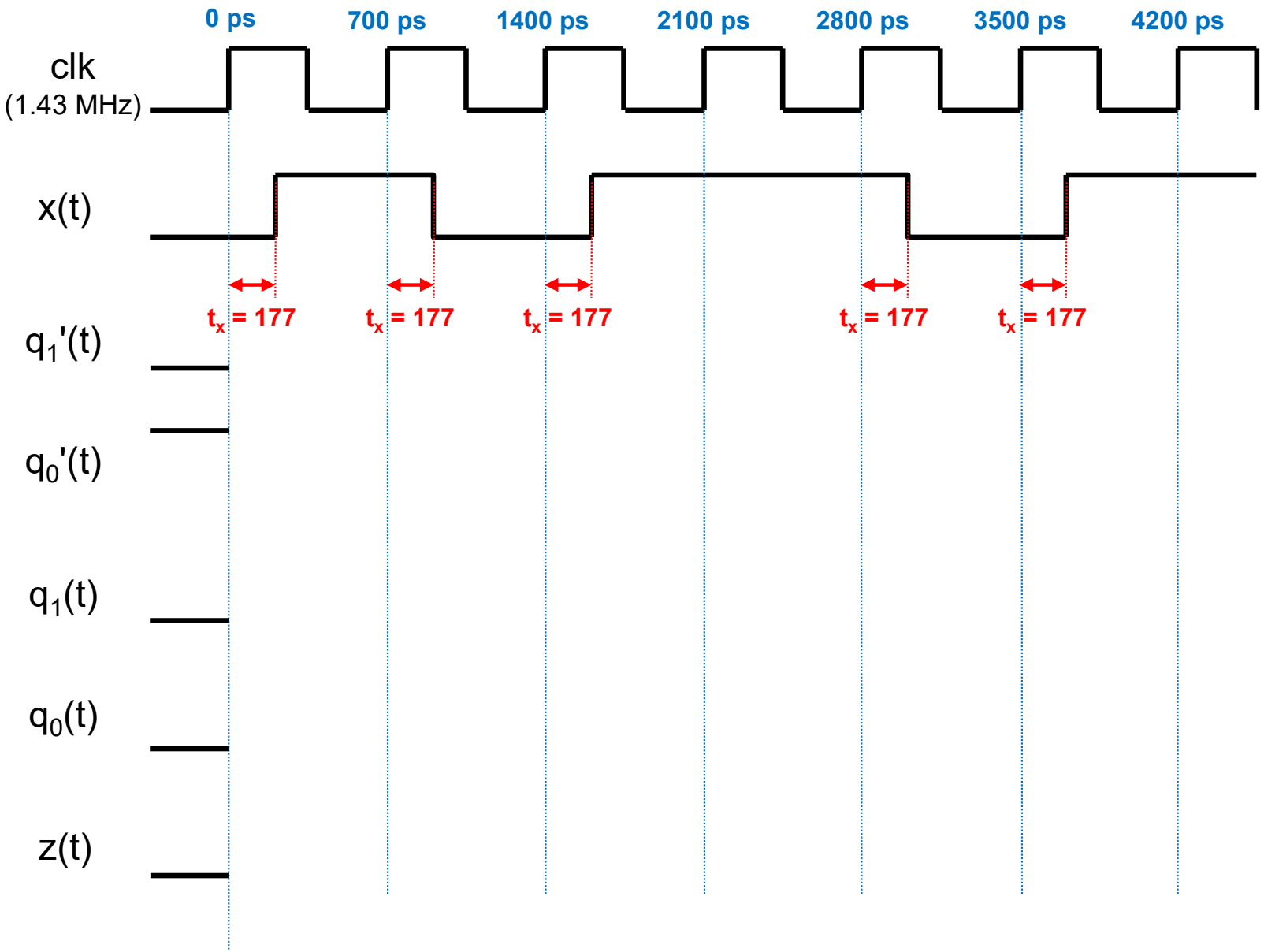
suponiendo:
$t_{hold} = t_{setup} = 0$

camino	retardo
$x \rightarrow D_1$	t_x + 180 ps
$x \rightarrow D_0$	t_x + 271 ps
$CLK_1 \rightarrow z$	347 ps
$CLK_0 \rightarrow z$	347 ps
$CLK_1 \rightarrow D_1$	584 ps
$CLK_1 \rightarrow D_0$	518 ps
$CLK_0 \rightarrow D_1$	584 ps
$CLK_0 \rightarrow D_0$	**677 ps**

$$t_{clk} \geq \max(\,retardo\,) = 677 \text{ ps} \Rightarrow \boldsymbol{f_{clk}} = \frac{1}{t_{clk}} \leq \frac{1}{677 \cdot 10^{-12}\text{s}} = \mathbf{1.47\ GHz}$$

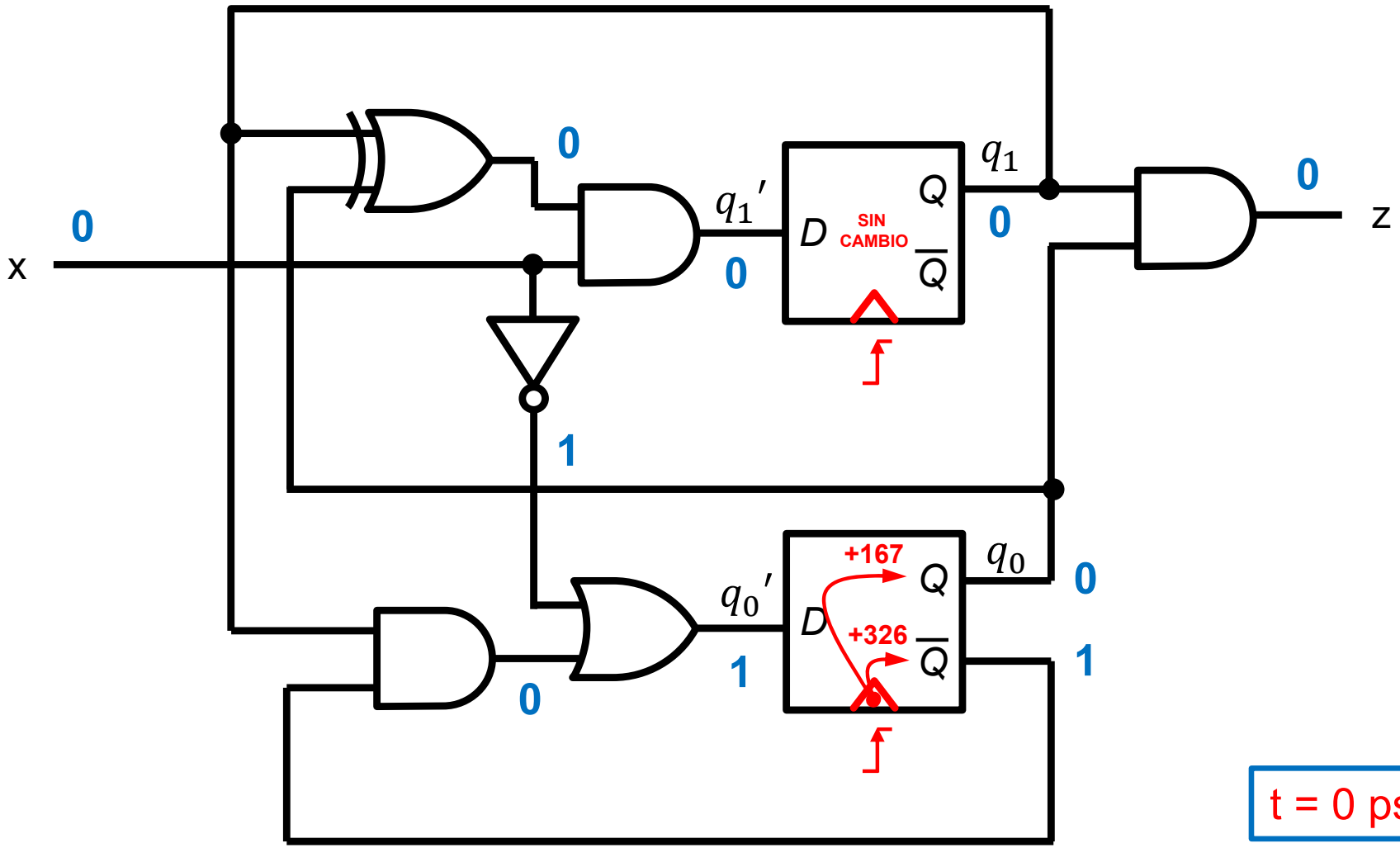
Simulación

CMOS 90 nm



Simulación

CMOS 90 nm

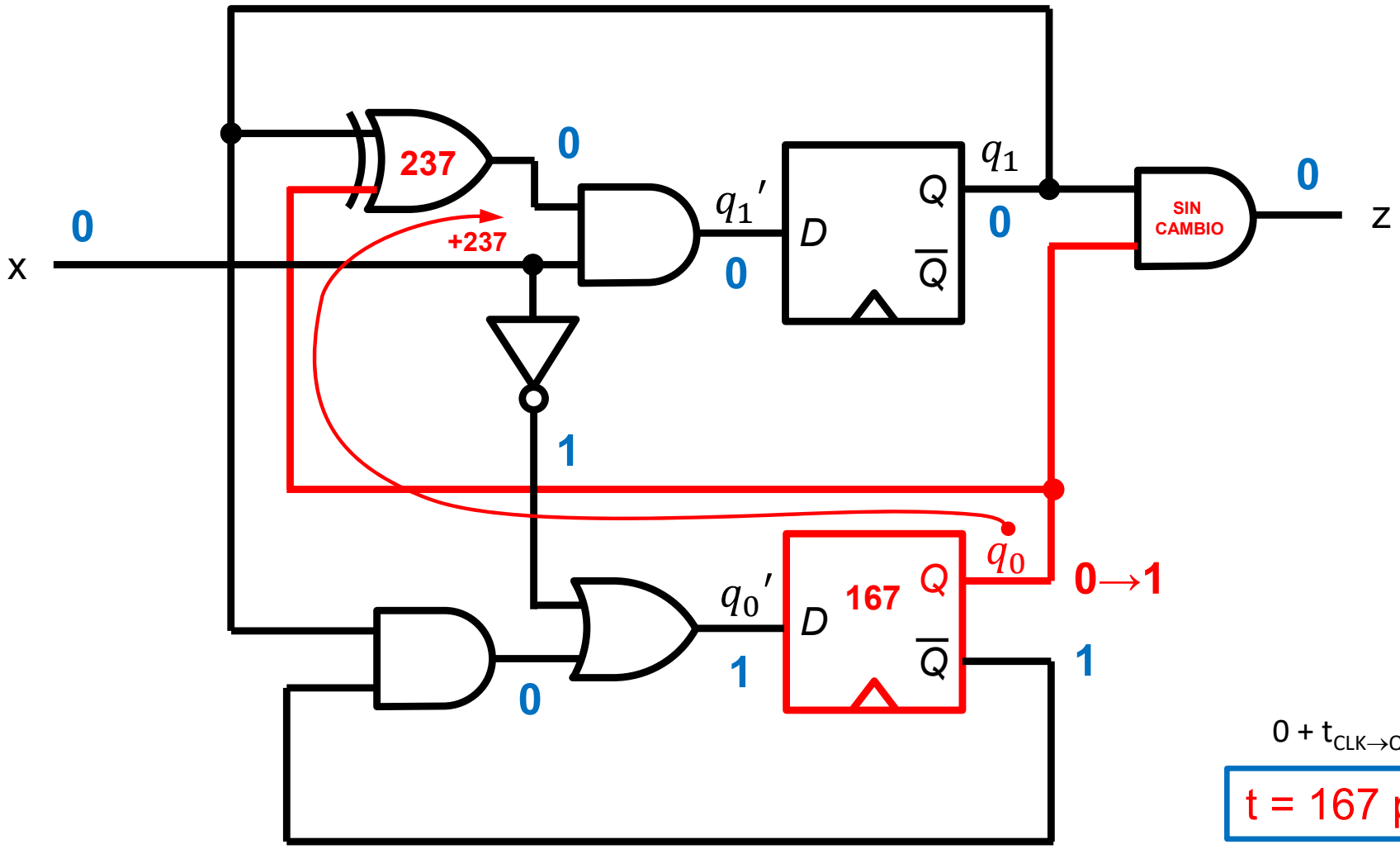


t = 0 ps

1er. ciclo de reloj

Simulación

CMOS 90 nm



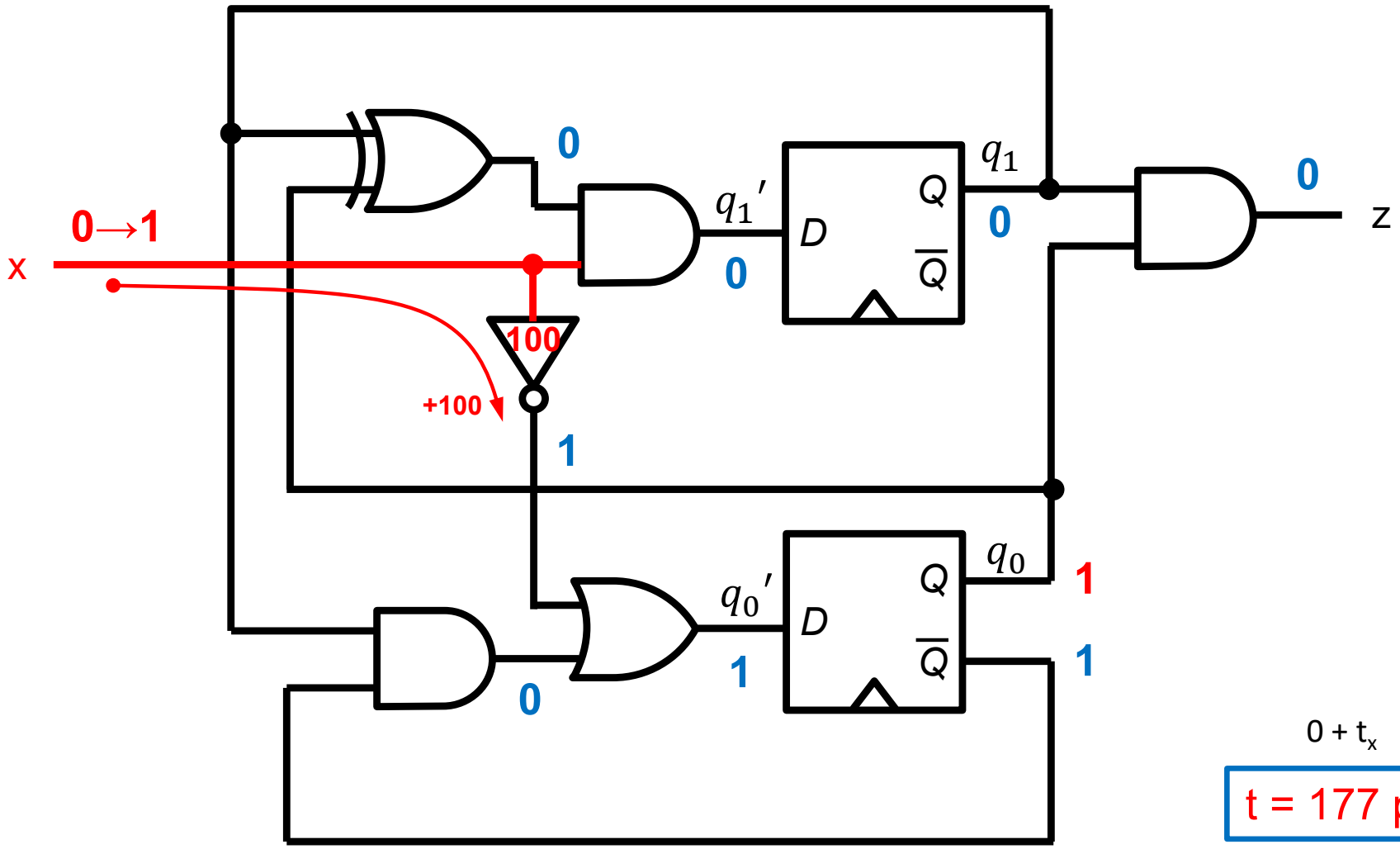
$0 + t_{CLK \rightarrow Q}$

t = 167 ps

1er. ciclo de reloj

Simulación

CMOS 90 nm



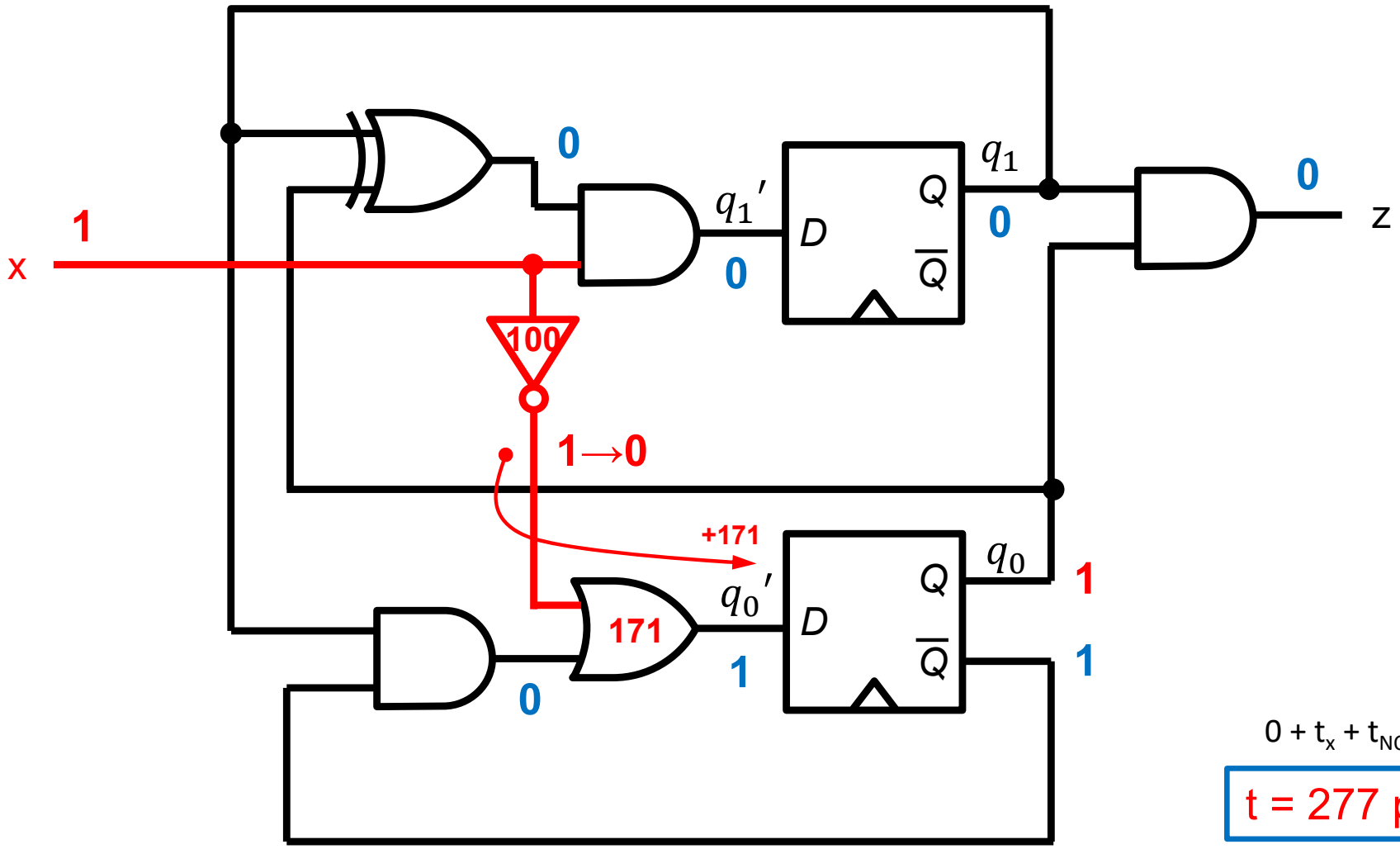
$0 + t_x$

t = 177 ps

1er. ciclo de reloj

Simulación

CMOS 90 nm



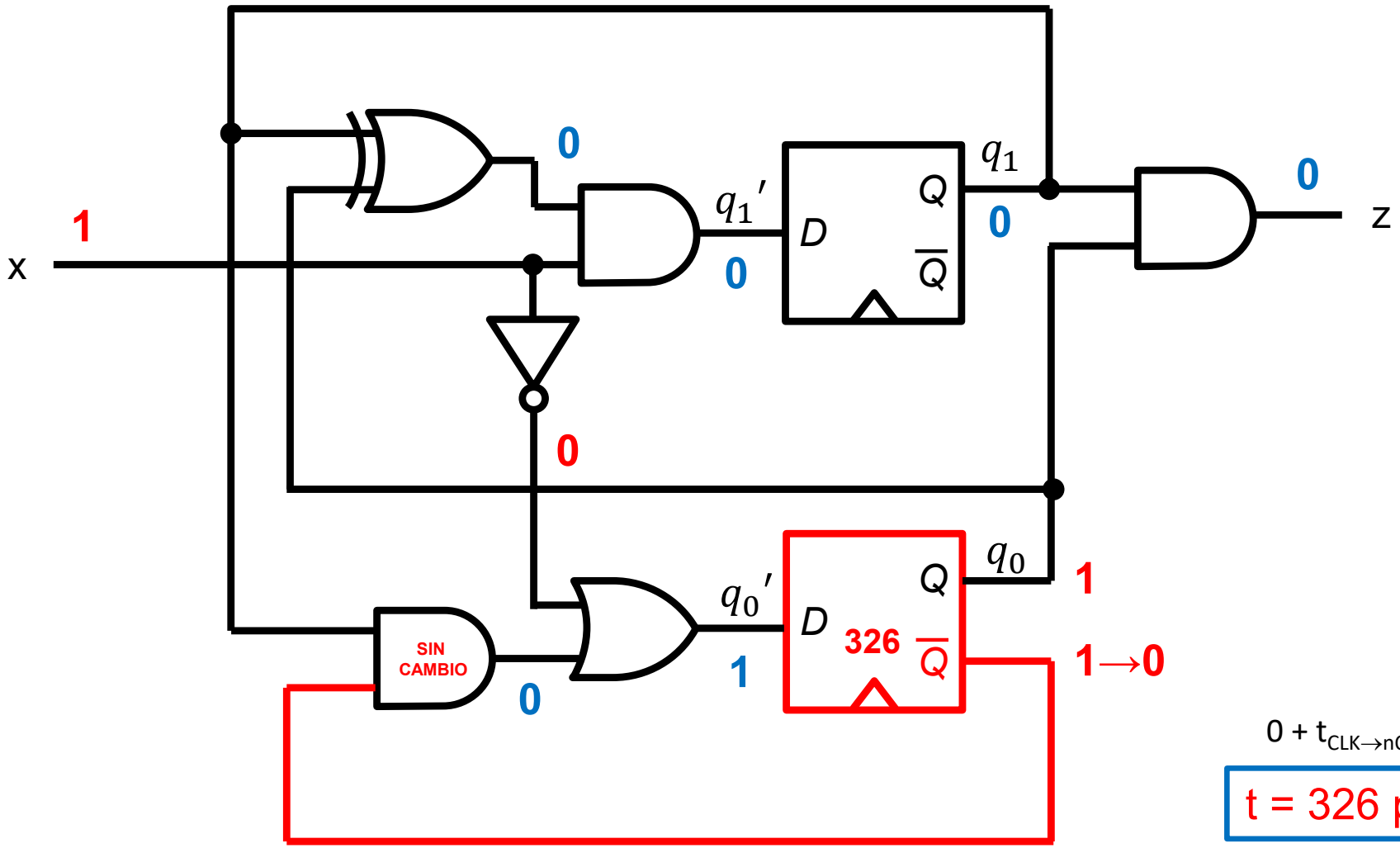
$0 + t_x + t_{NOT}$

t = 277 ps

1er. ciclo de reloj

Simulación

CMOS 90 nm



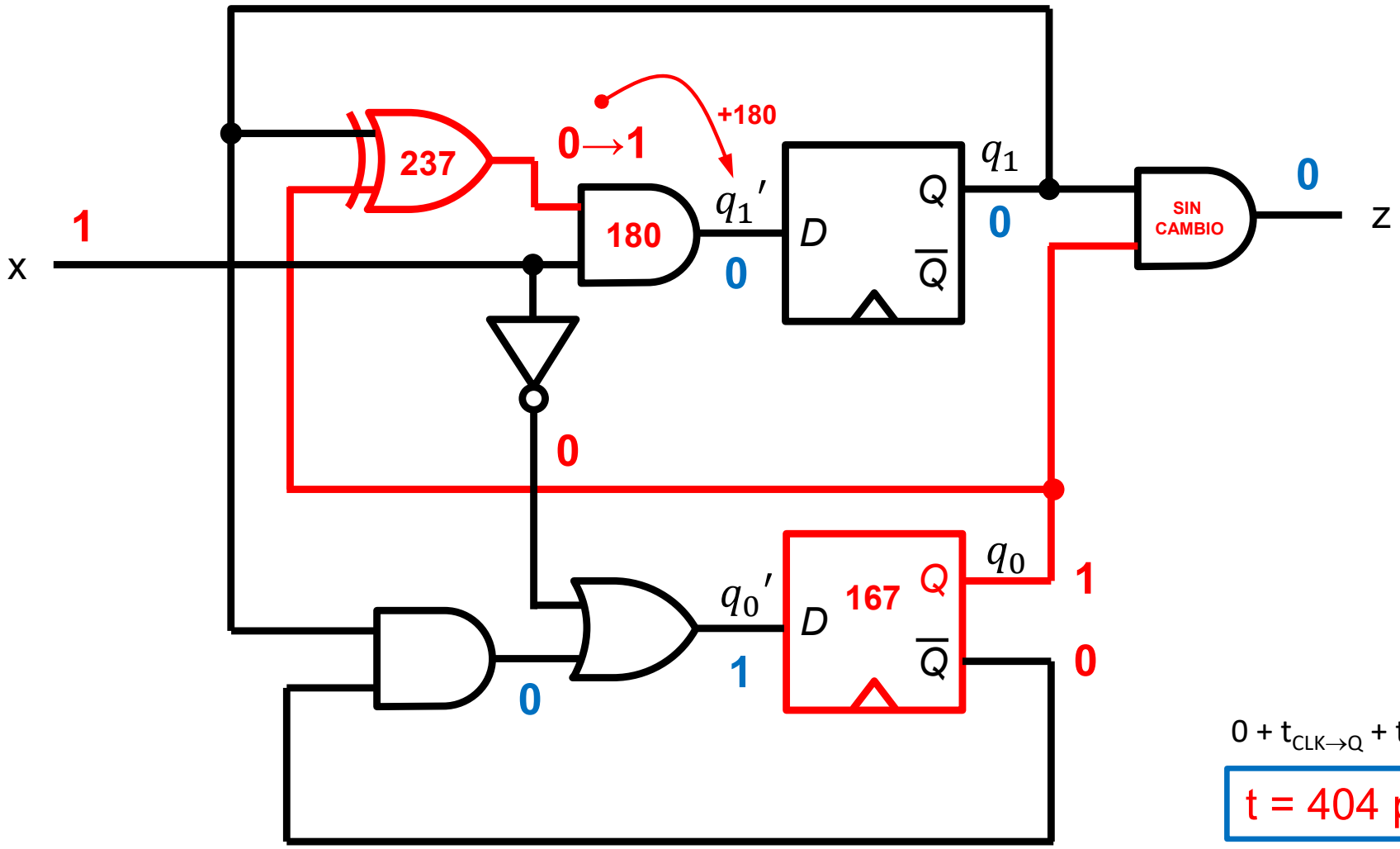
$0 + t_{CLK \rightarrow nQ}$

t = 326 ps

1er. ciclo de reloj

Simulación

CMOS 90 nm



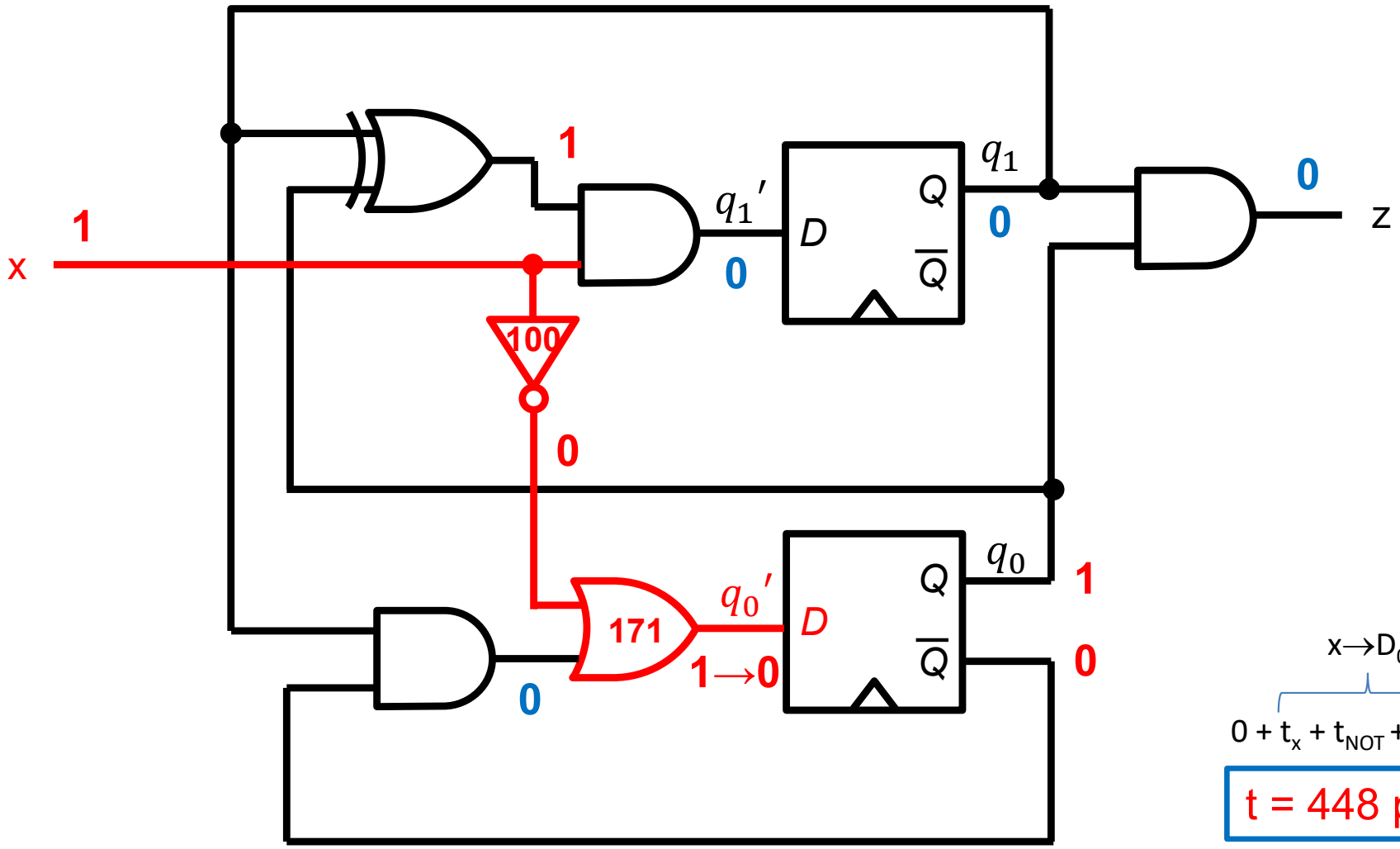
$0 + t_{CLK \rightarrow Q} + t_{XOR}$

t = 404 ps

1er. ciclo de reloj

Simulación

CMOS 90 nm



$x \rightarrow D_0$

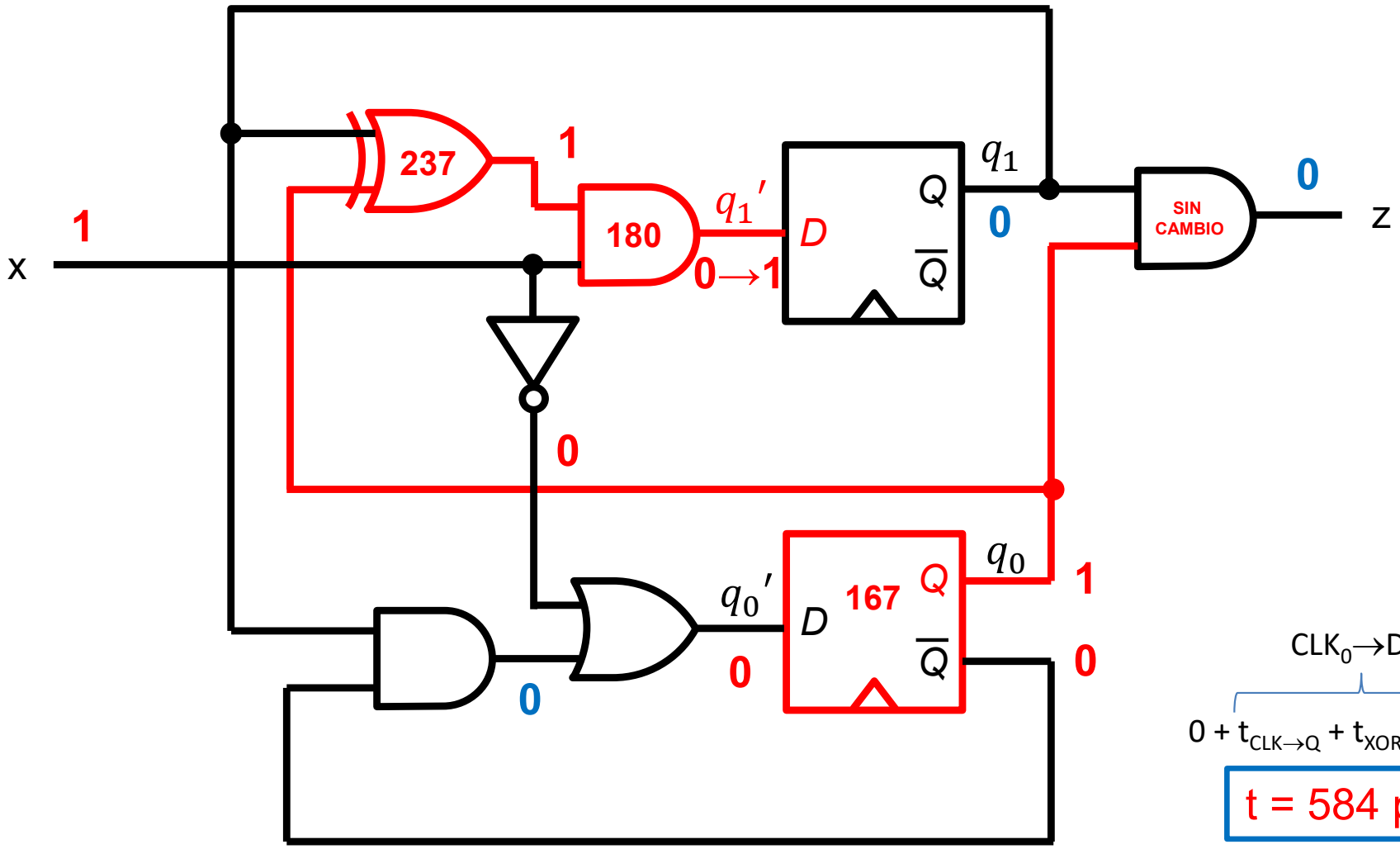
$0 + t_x + t_{NOT} + t_{OR}$

t = 448 ps

1er. ciclo de reloj

Simulación

CMOS 90 nm



$CLK_0 \rightarrow D_1$

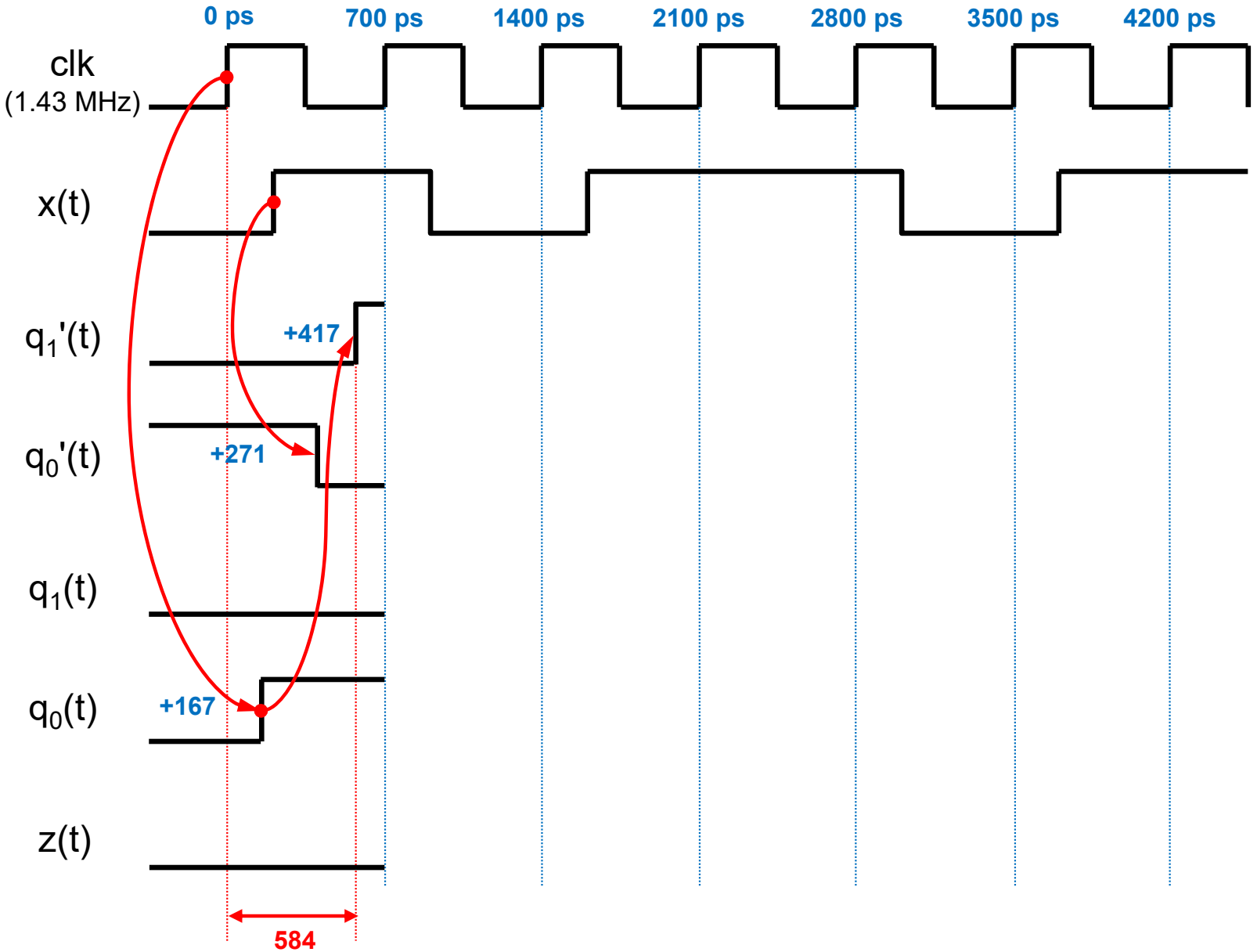
$0 + t_{CLK \rightarrow Q} + t_{XOR} + t_{AND}$

t = 584 ps

1er. ciclo de reloj

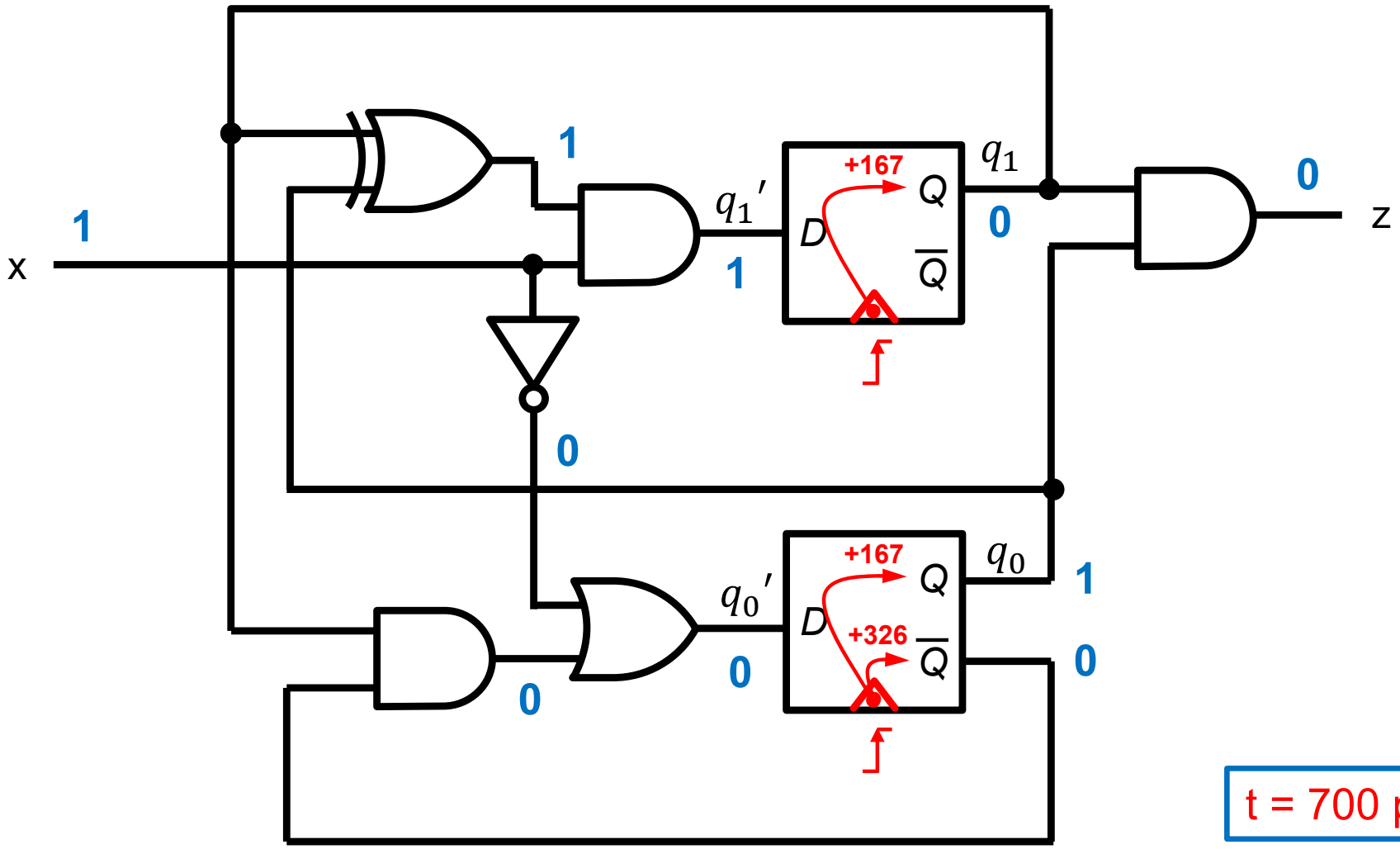
Simulación

CMOS 90 nm



Simulación

CMOS 90 nm

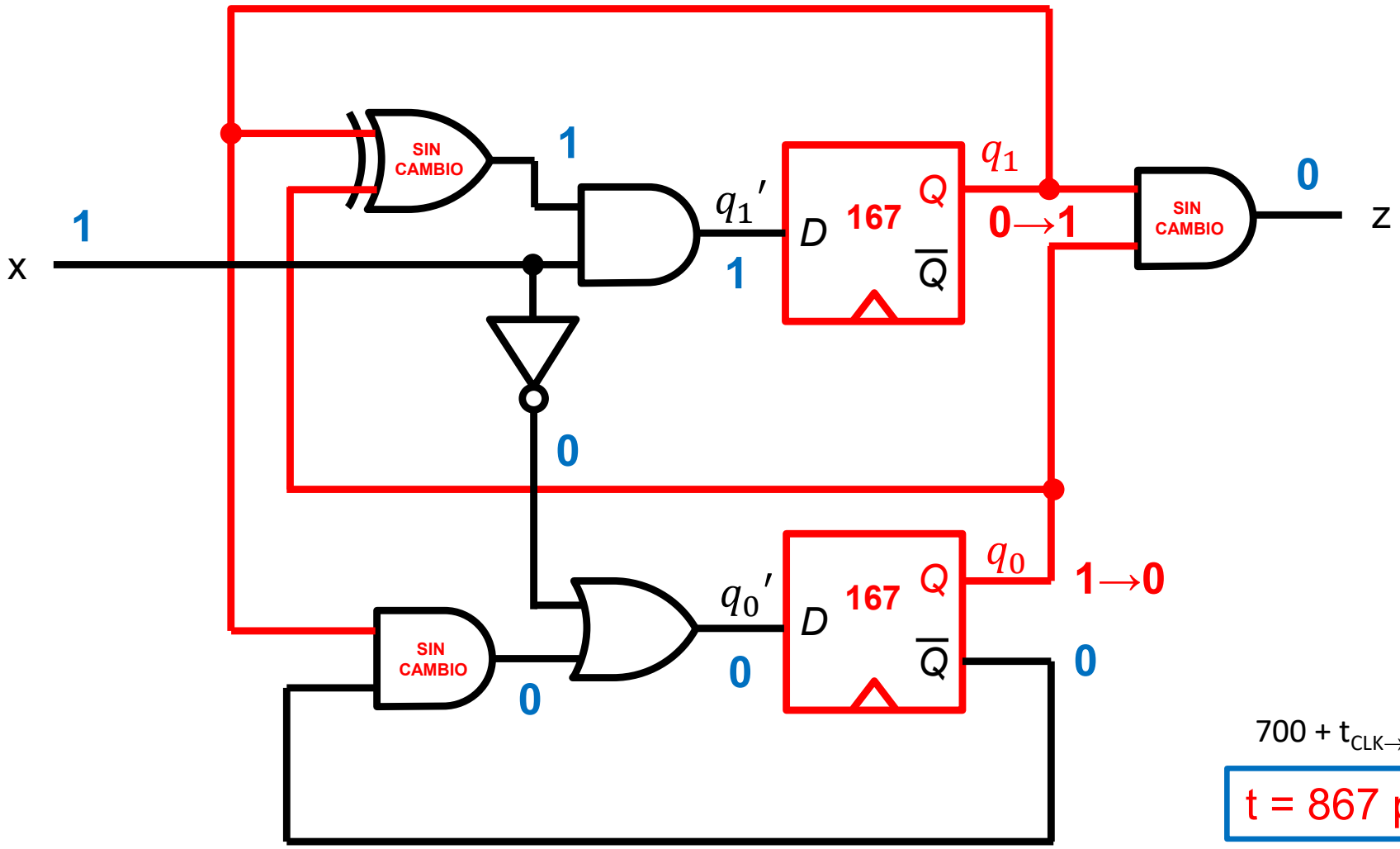


t = 700 ps

2do. ciclo de reloj

Simulación

CMOS 90 nm



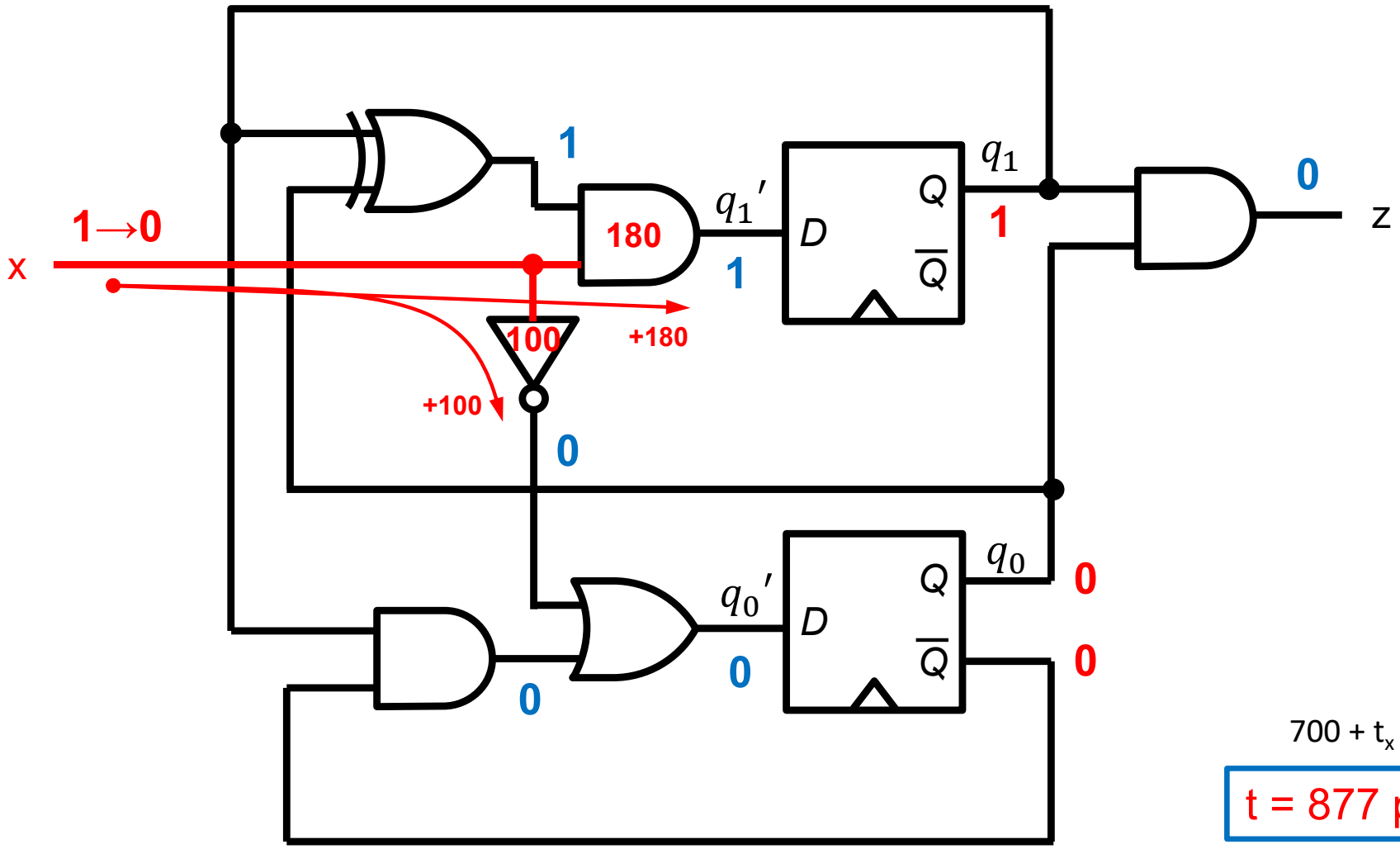
$700 + t_{CLK \rightarrow Q}$

t = 867 ps

2do. ciclo de reloj

Simulación

CMOS 90 nm



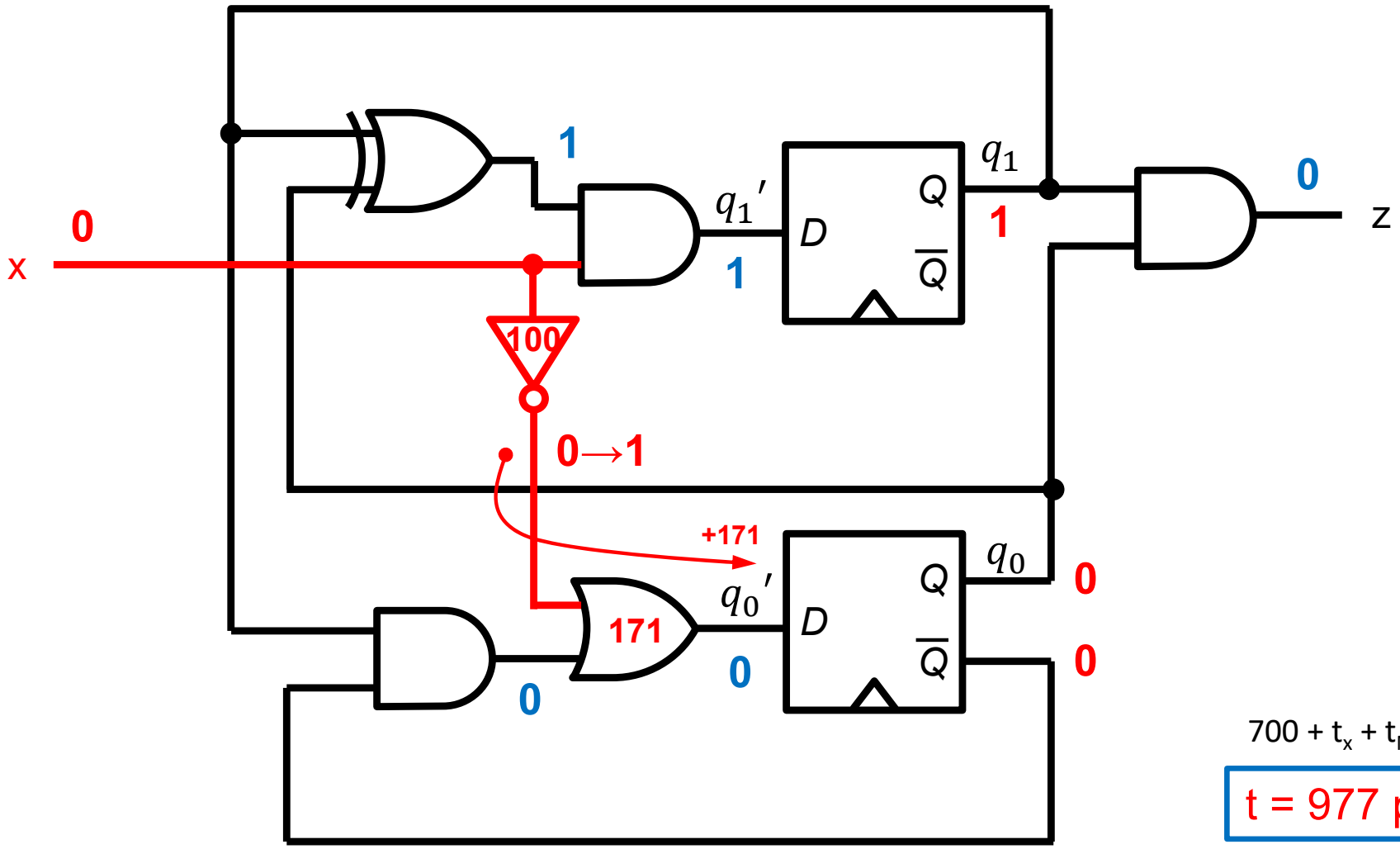
$700 + t_x$

t = 877 ps

2do. ciclo de reloj

Simulación

CMOS 90 nm



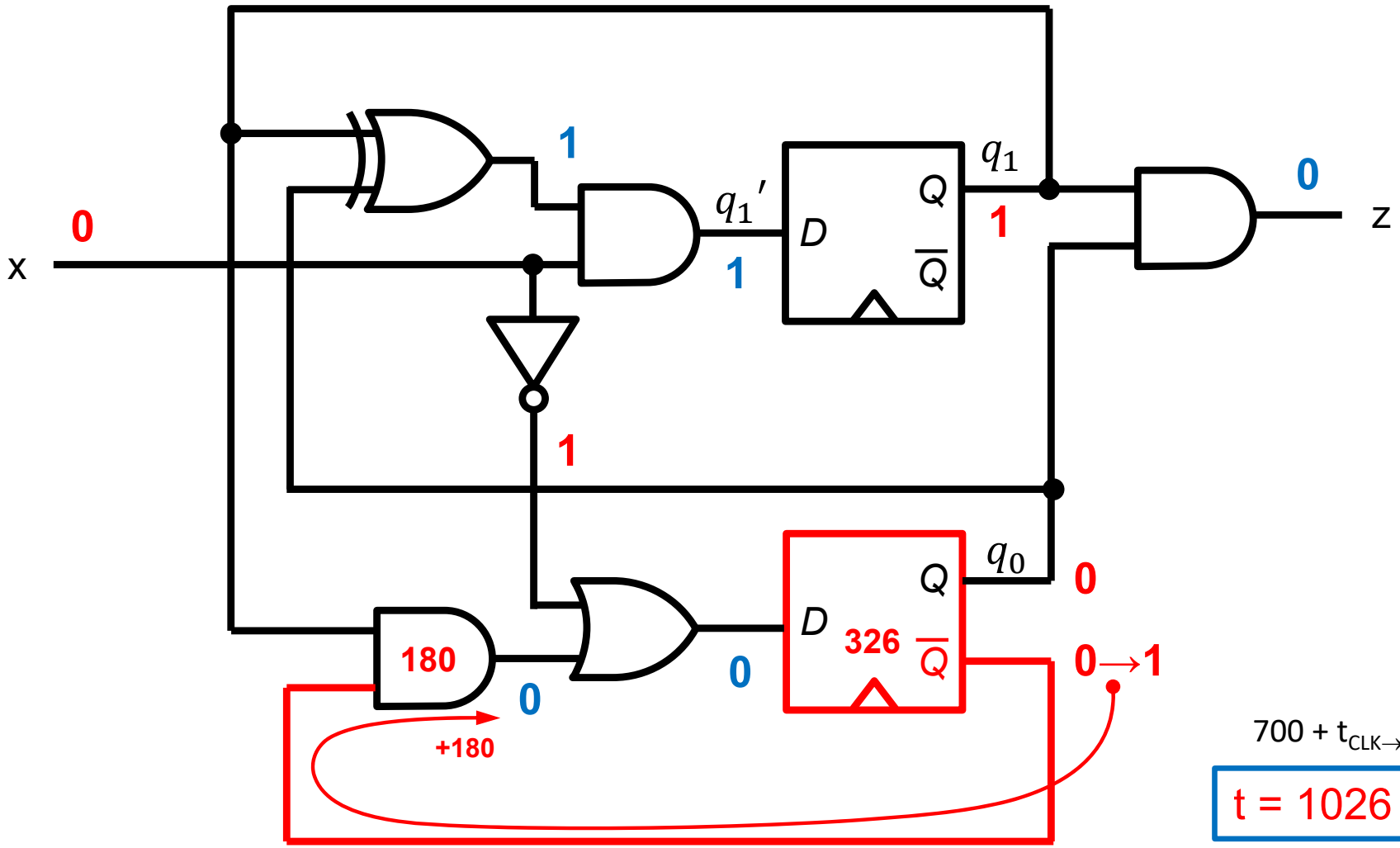
$700 + t_x + t_{NOT}$

t = 977 ps

2do. ciclo de reloj

Simulación

CMOS 90 nm



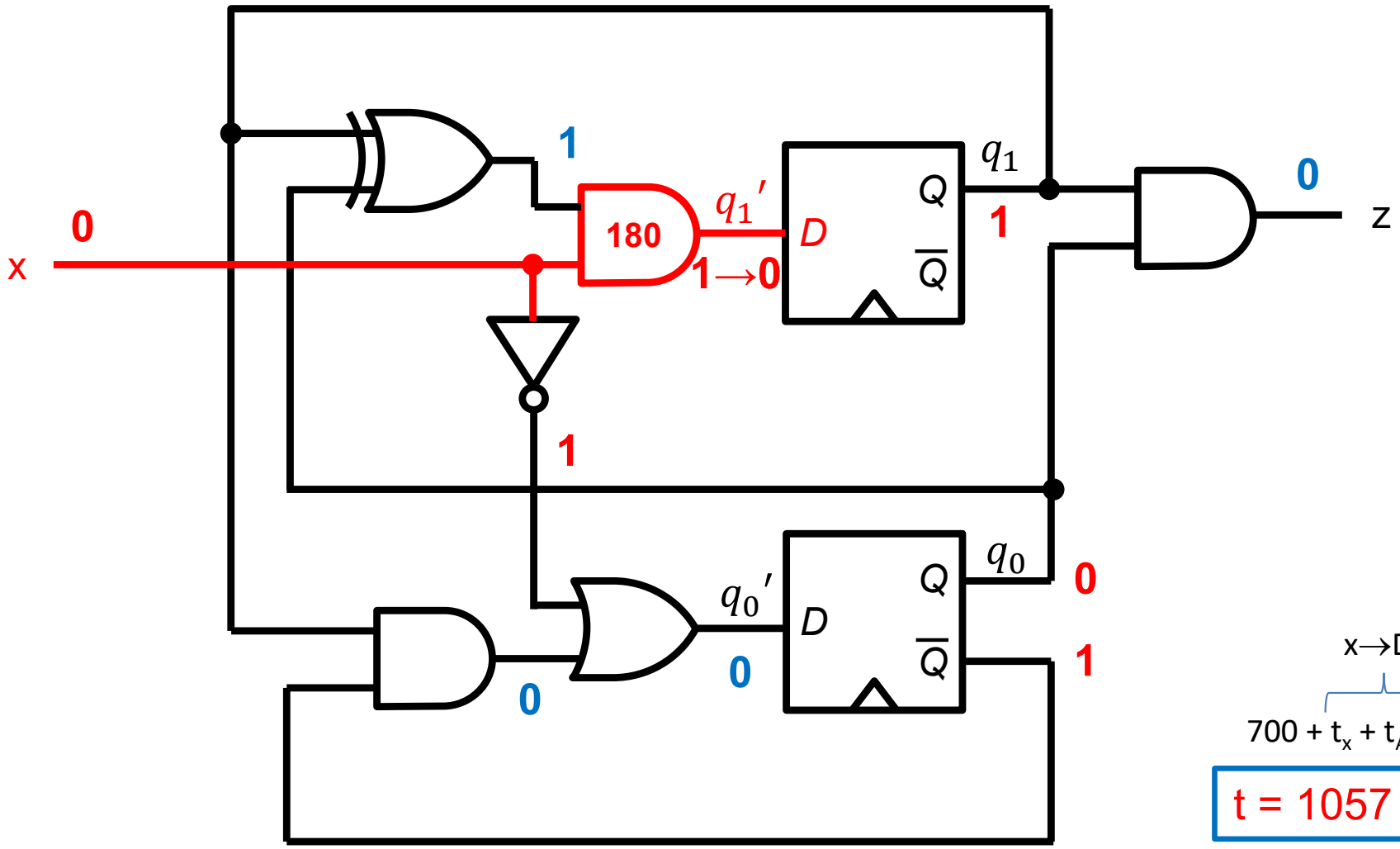
$700 + t_{CLK \rightarrow nQ}$

t = 1026 ps

2do. ciclo de reloj

Simulación

CMOS 90 nm



$x \rightarrow D_1$

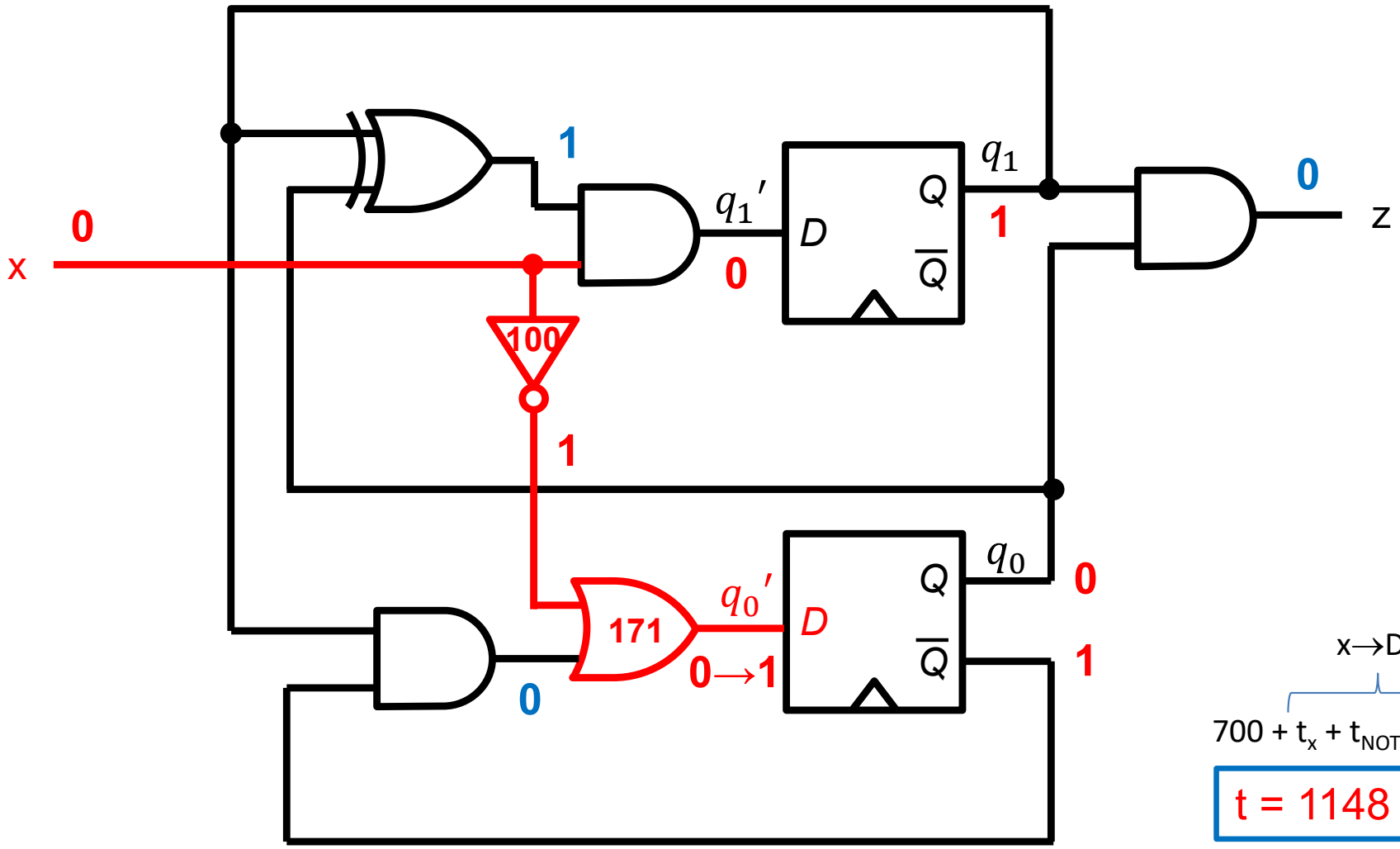
$700 + t_x + t_{AND}$

t = 1057 ps

2do. ciclo de reloj

Simulación

CMOS 90 nm



$x \rightarrow D_0$

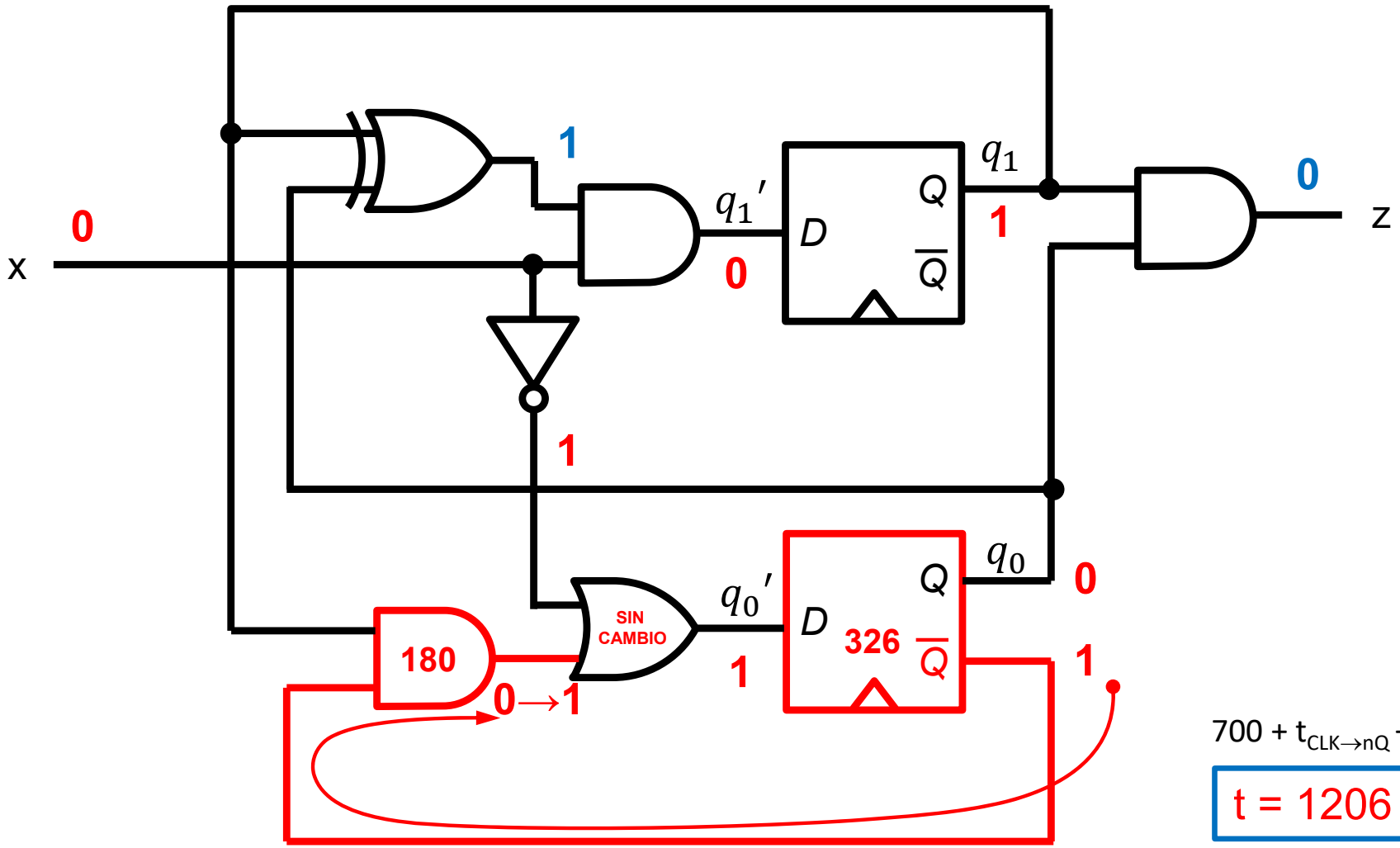
$700 + t_x + t_{NOT} + t_{OR}$

t = 1148 ps

2do. ciclo de reloj

Simulación

CMOS 90 nm



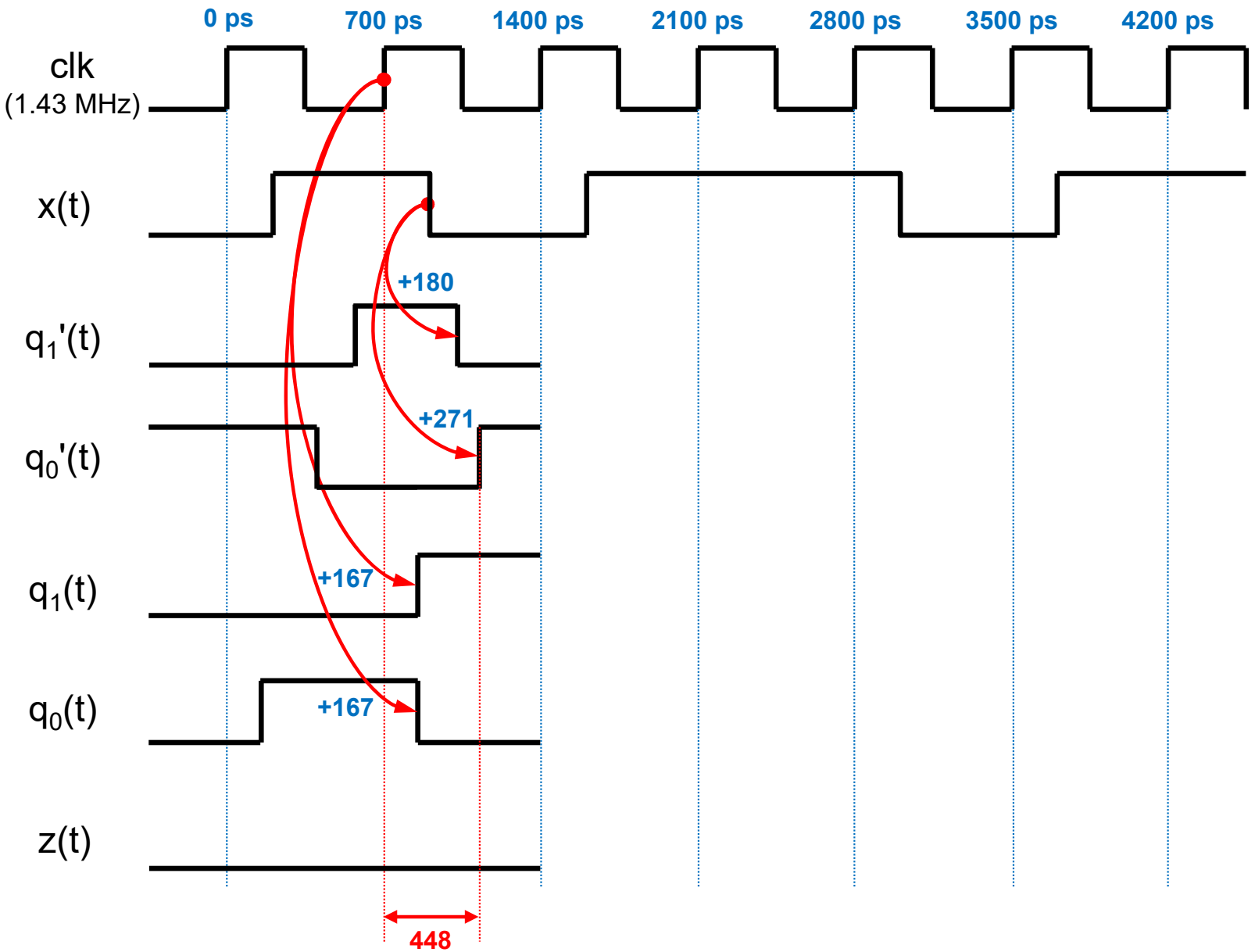
$700 + t_{CLK \rightarrow nQ} + t_{AND}$

t = 1206 ps

2do. ciclo de reloj

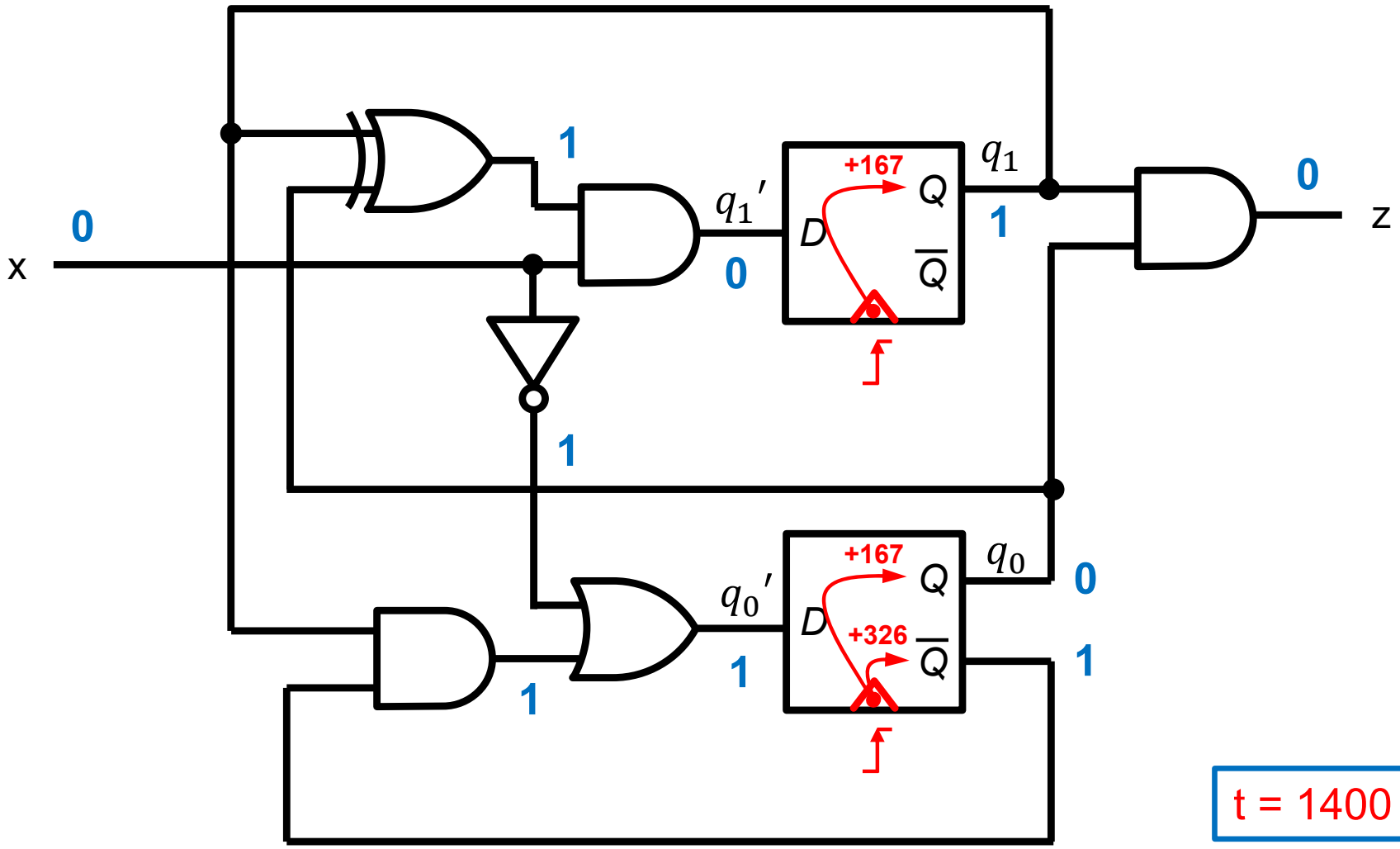
Simulación

CMOS 90 nm



Simulación

CMOS 90 nm

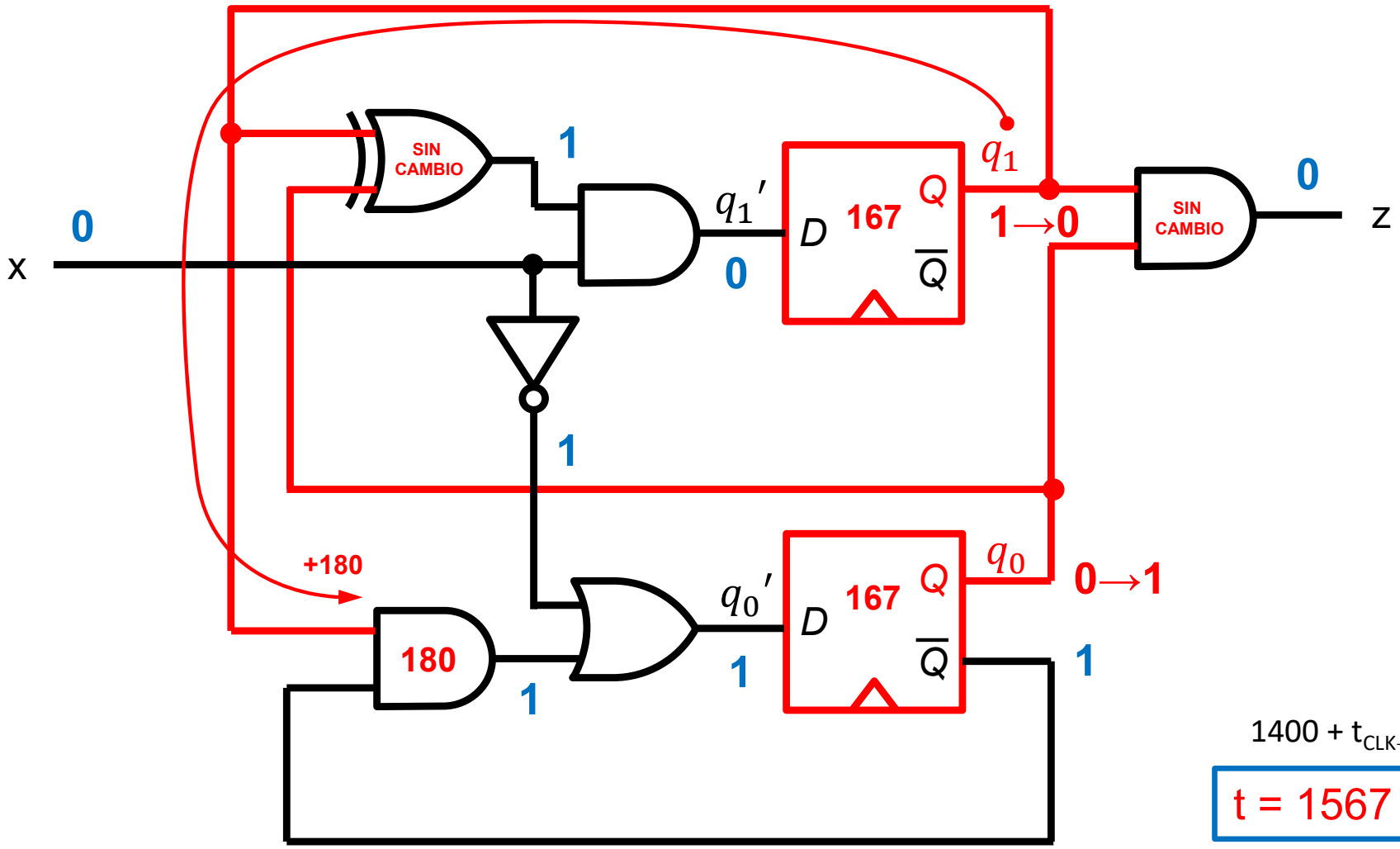


t = 1400 ps

3er. ciclo de reloj

Simulación

CMOS 90 nm



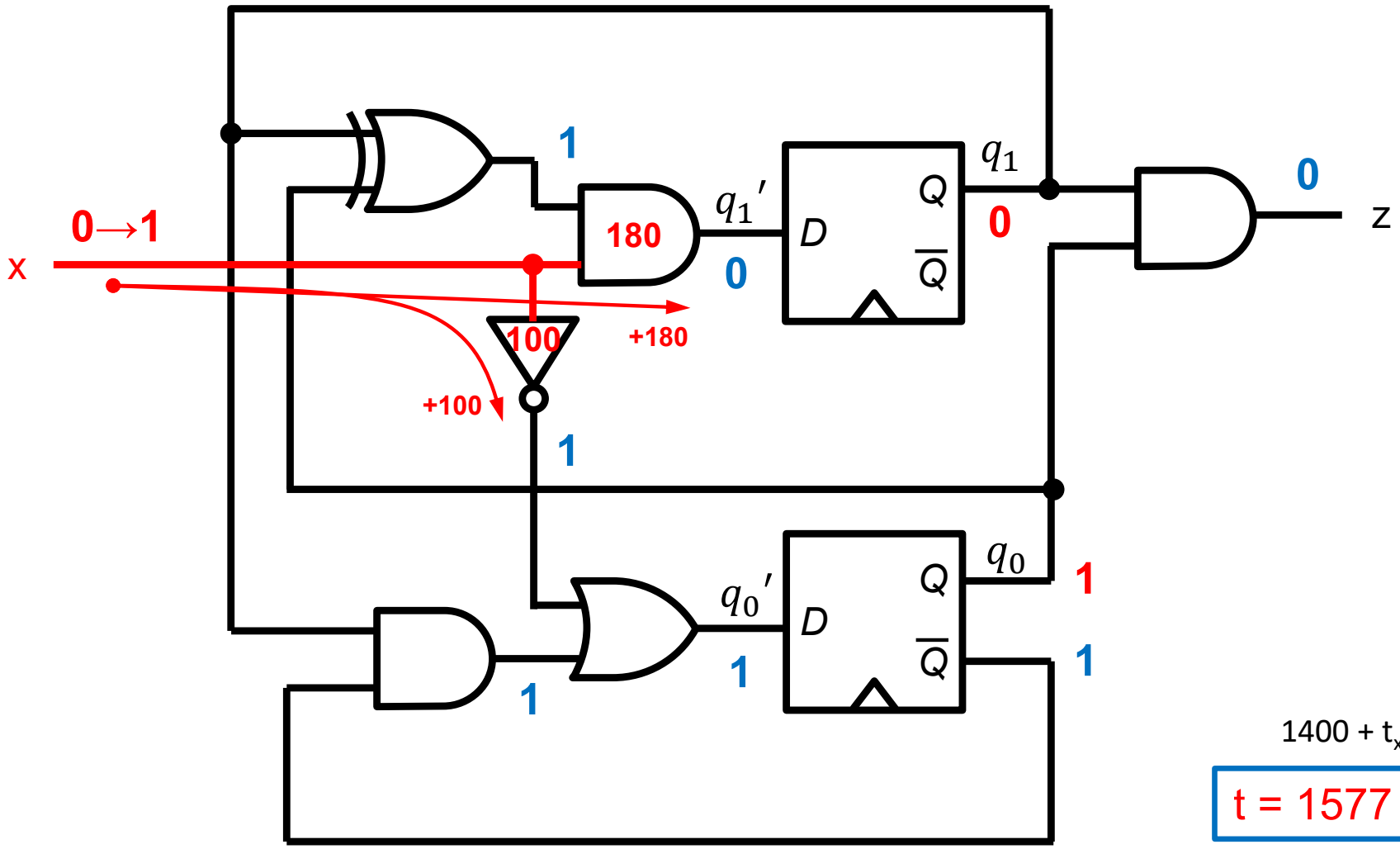
$1400 + t_{CLK \rightarrow Q}$

t = 1567 ps

3er. ciclo de reloj

Simulación

CMOS 90 nm



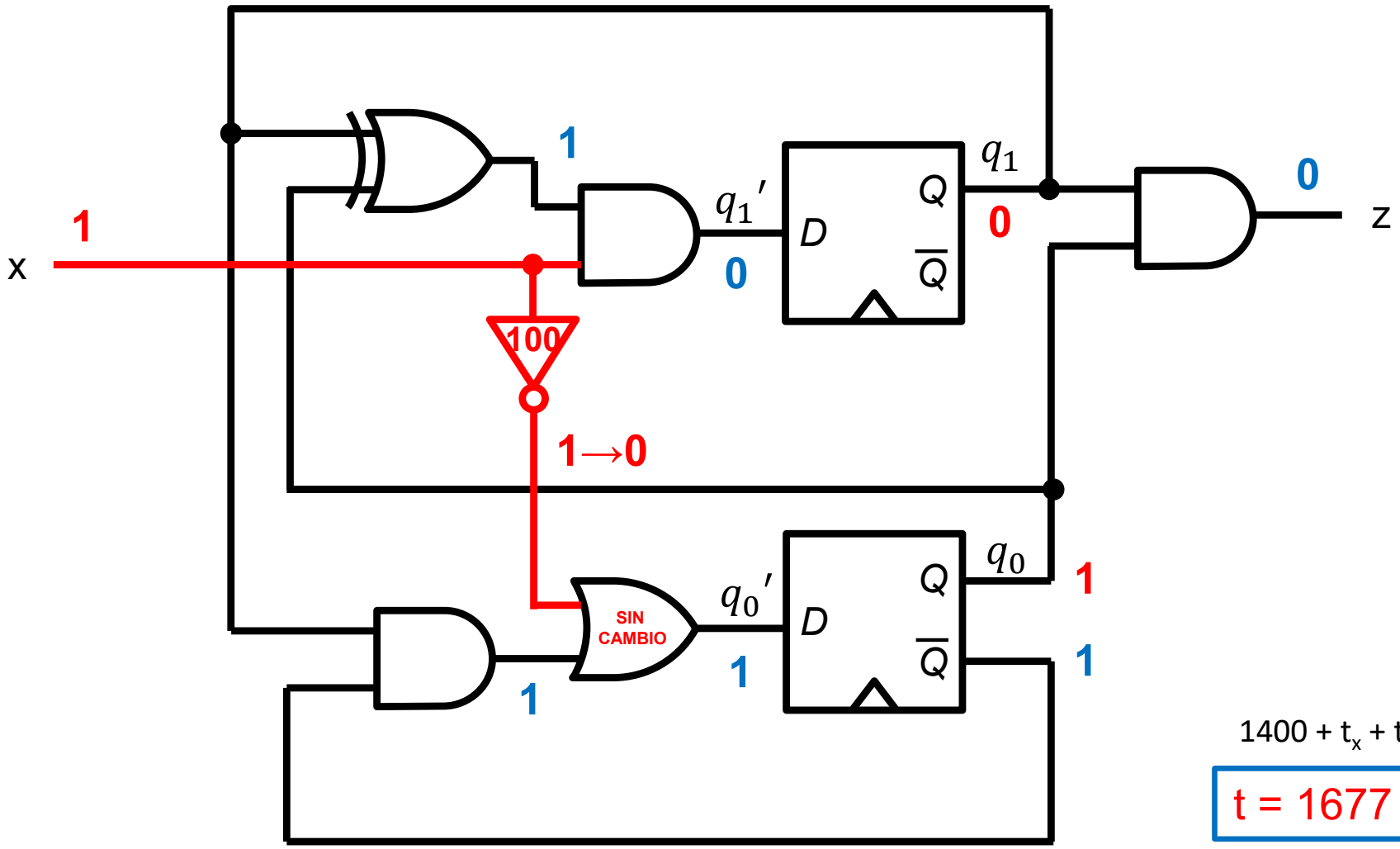
1400 + t_x

t = 1577 ps

3er. ciclo de reloj

Simulación

CMOS 90 nm



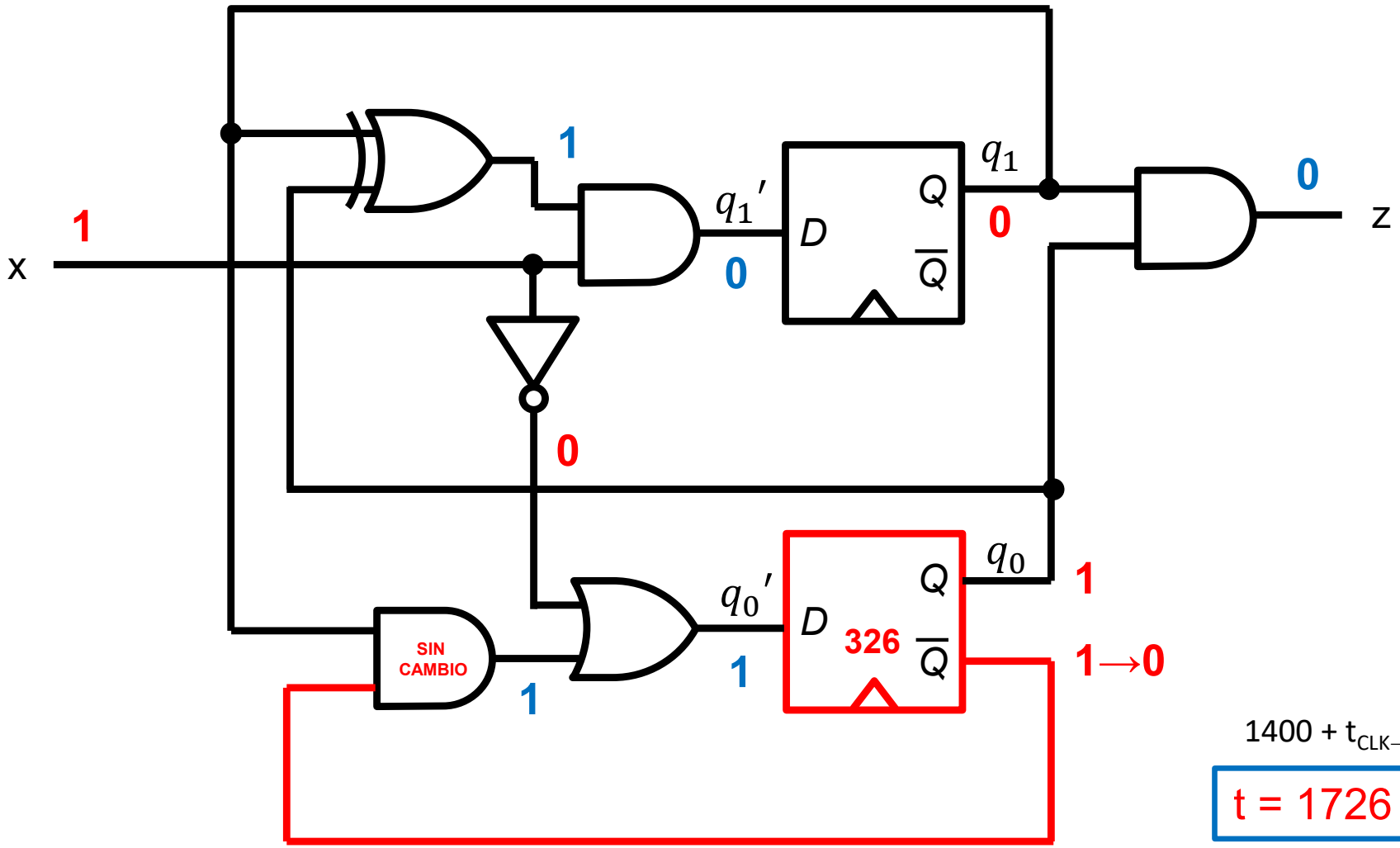
$1400 + t_x + t_{NOT}$

t = 1677 ps

3er. ciclo de reloj

Simulación

CMOS 90 nm



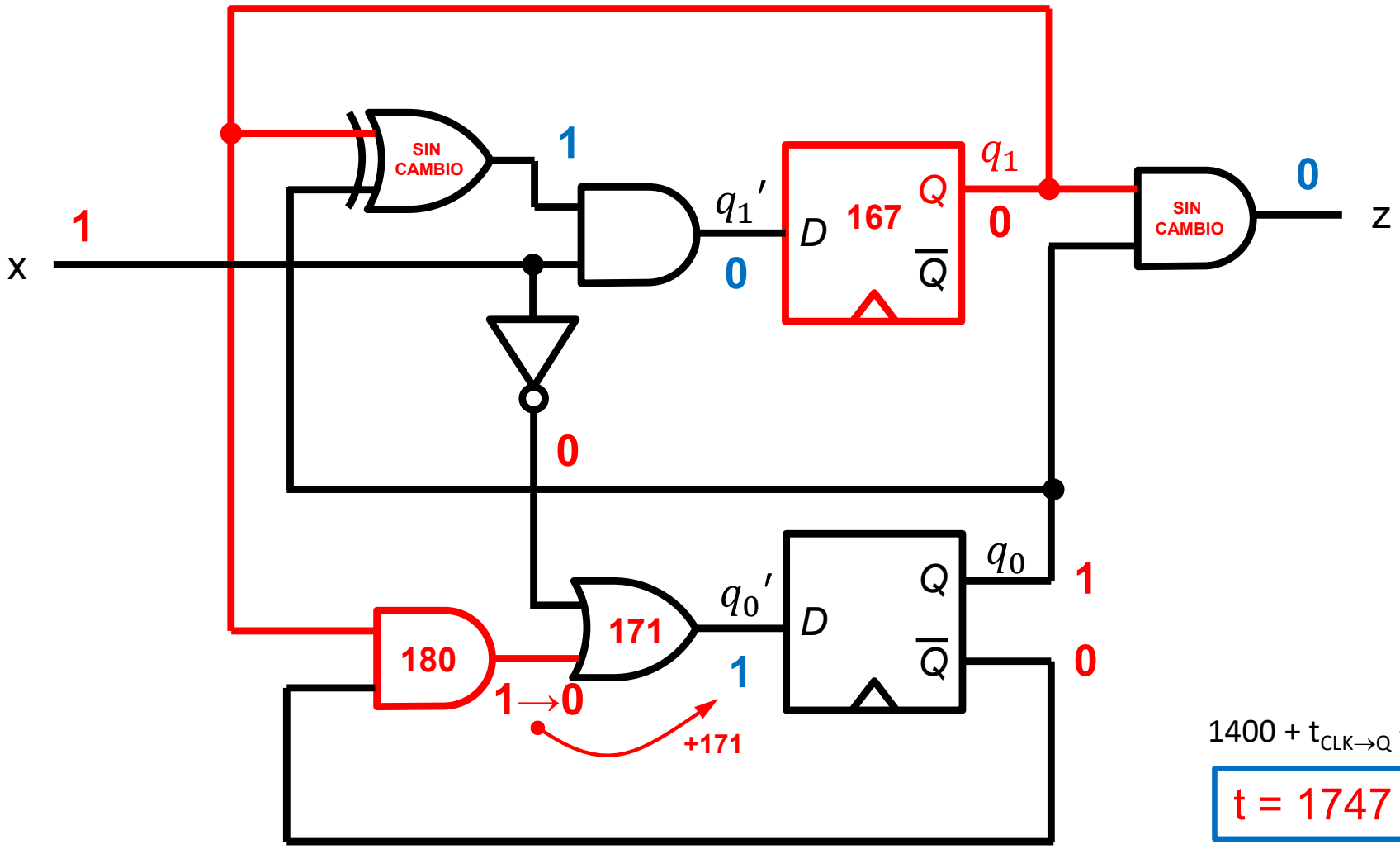
$1400 + t_{CLK \rightarrow nQ}$

t = 1726 ps

3er. ciclo de reloj

Simulación

CMOS 90 nm



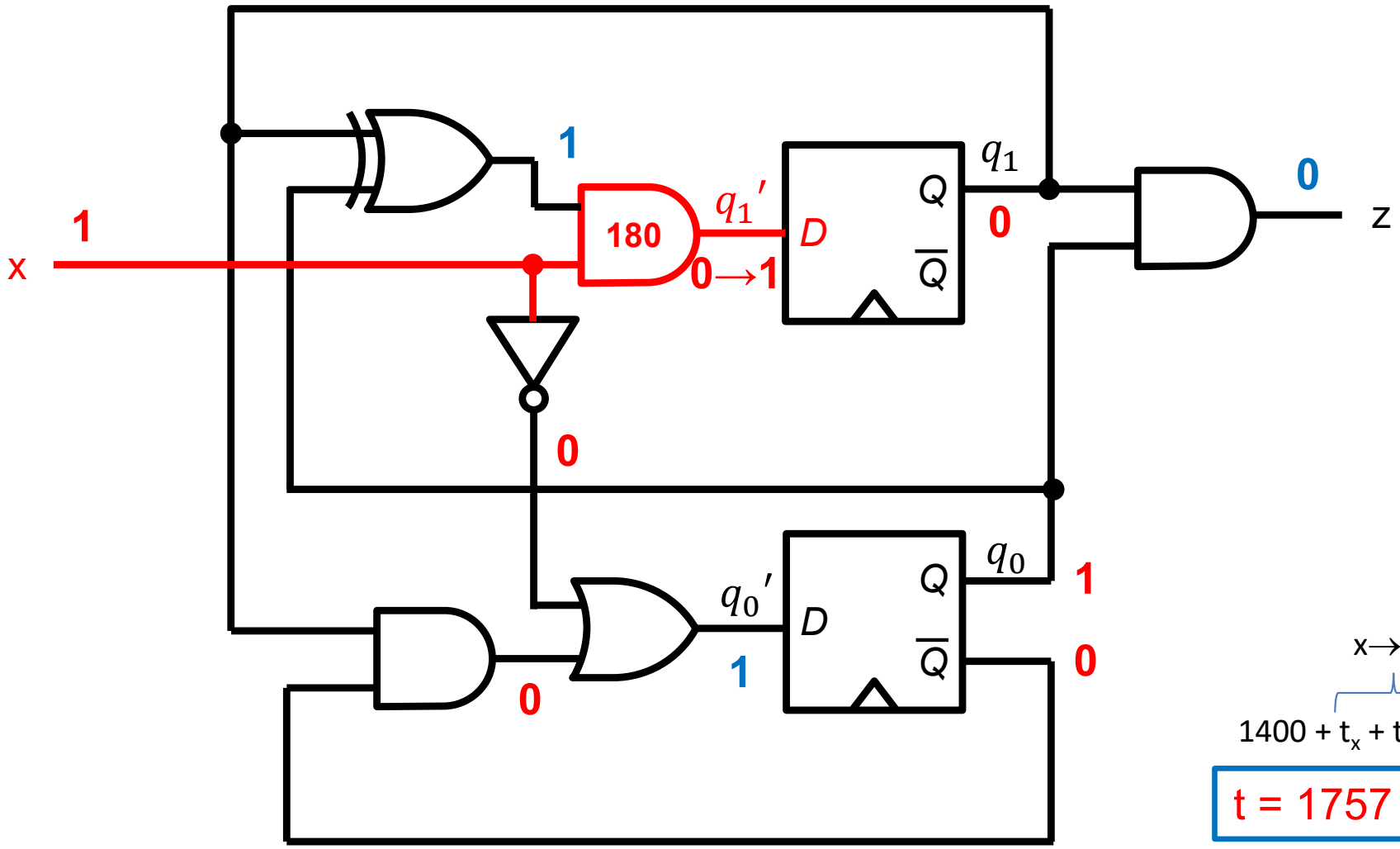
$1400 + t_{CLK \rightarrow Q} + t_{AND}$

t = 1747 ps

3er. ciclo de reloj

Simulación

CMOS 90 nm



$x \rightarrow D_1$

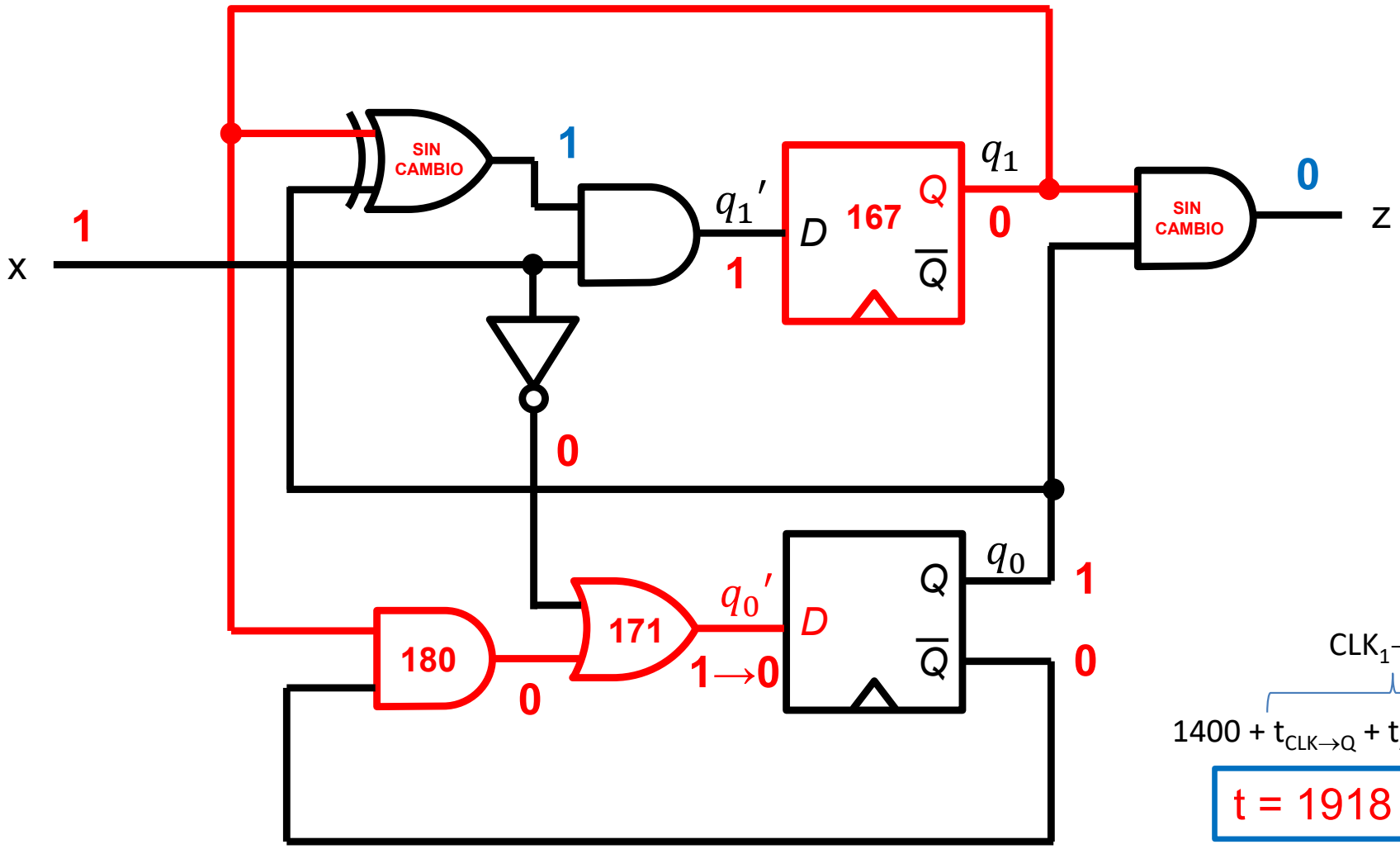
$1400 + t_x + t_{AND}$

t = 1757 ps

3er. ciclo de reloj

Simulación

CMOS 90 nm



$CLK_1 \rightarrow D_0$

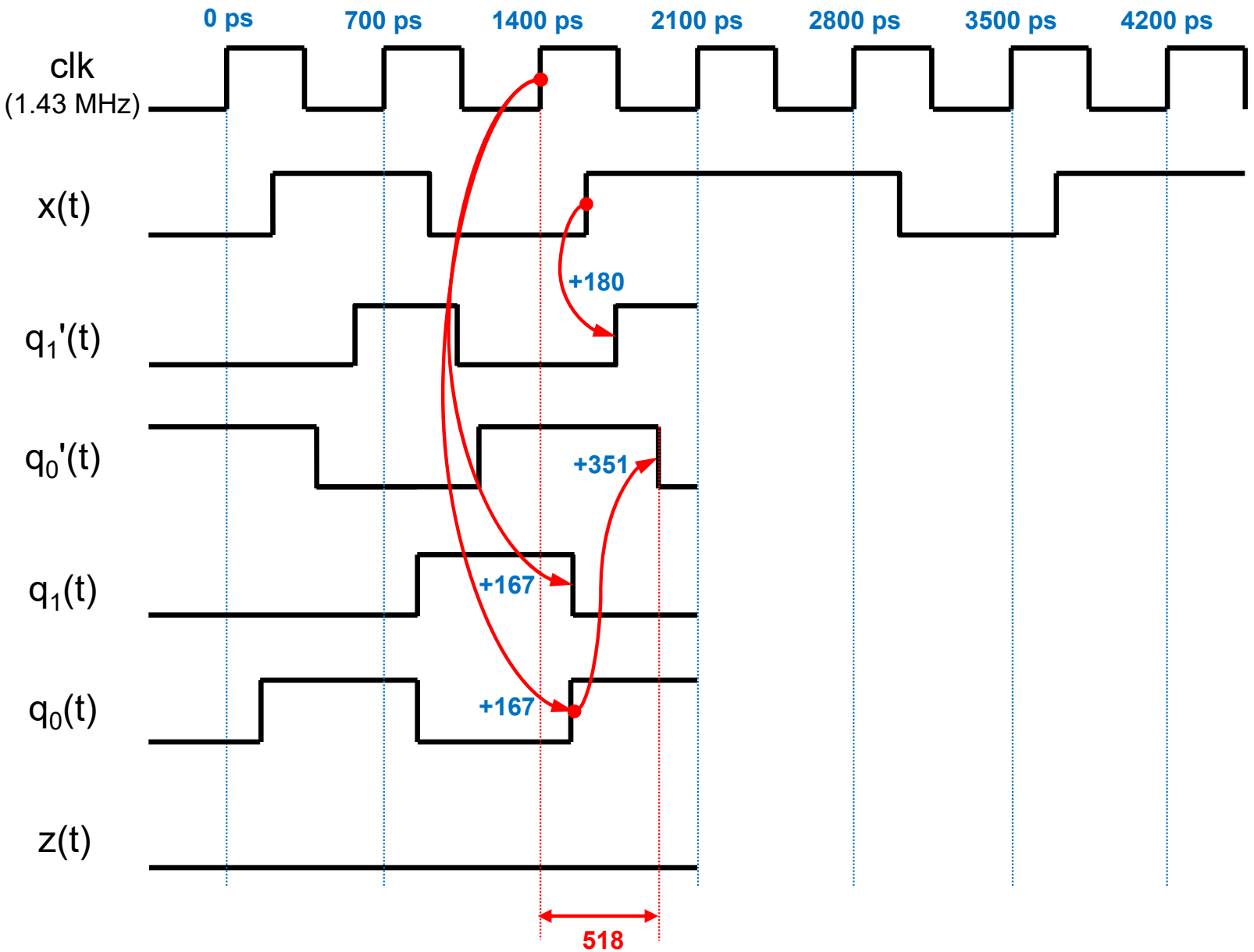
$1400 + t_{CLK \rightarrow Q} + t_{AND} + t_{OR}$

t = 1918 ps

3er. ciclo de reloj

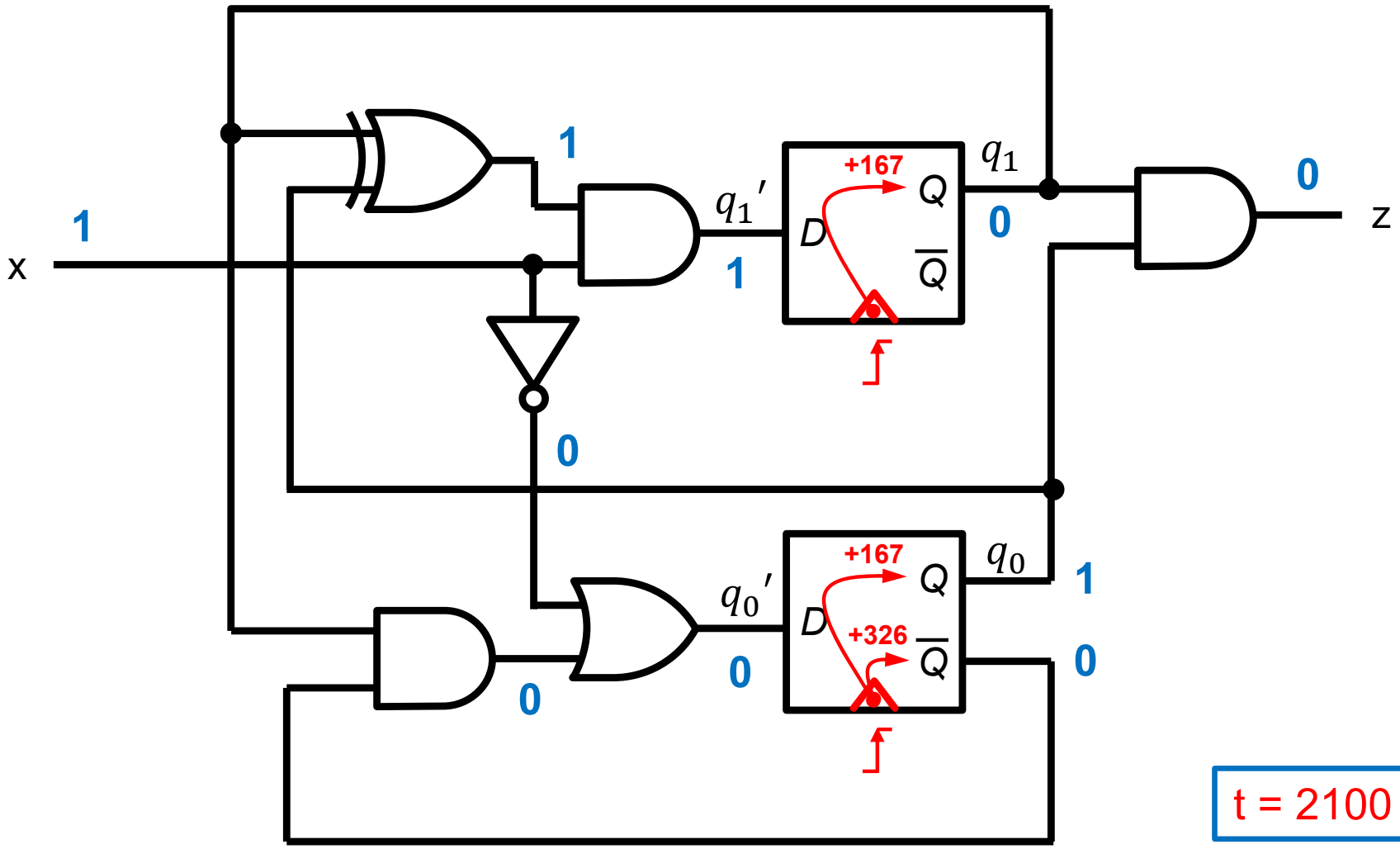
Simulación

CMOS 90 nm



Simulación

CMOS 90 nm

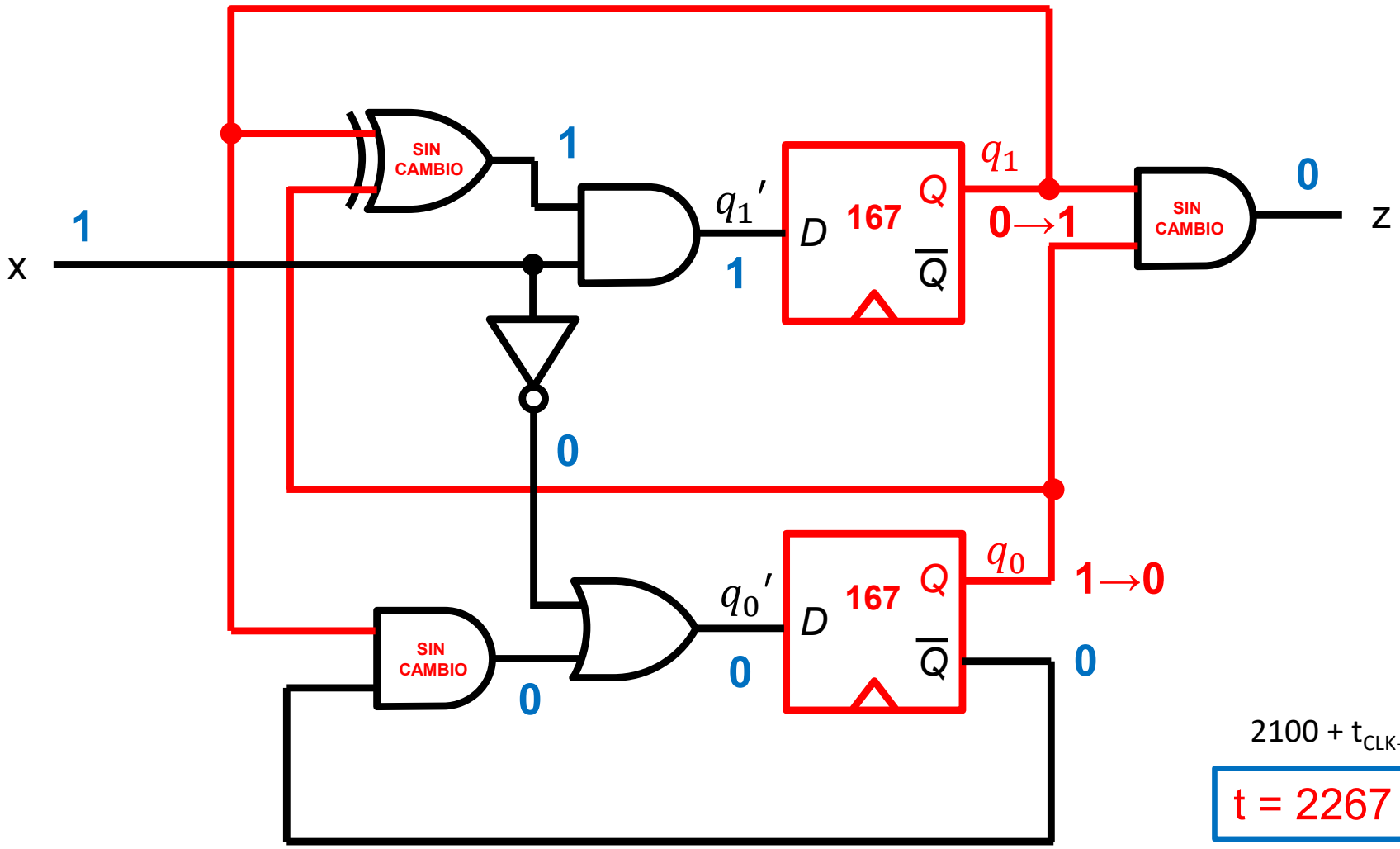


t = 2100 ps

4o. ciclo de reloj

Simulación

CMOS 90 nm



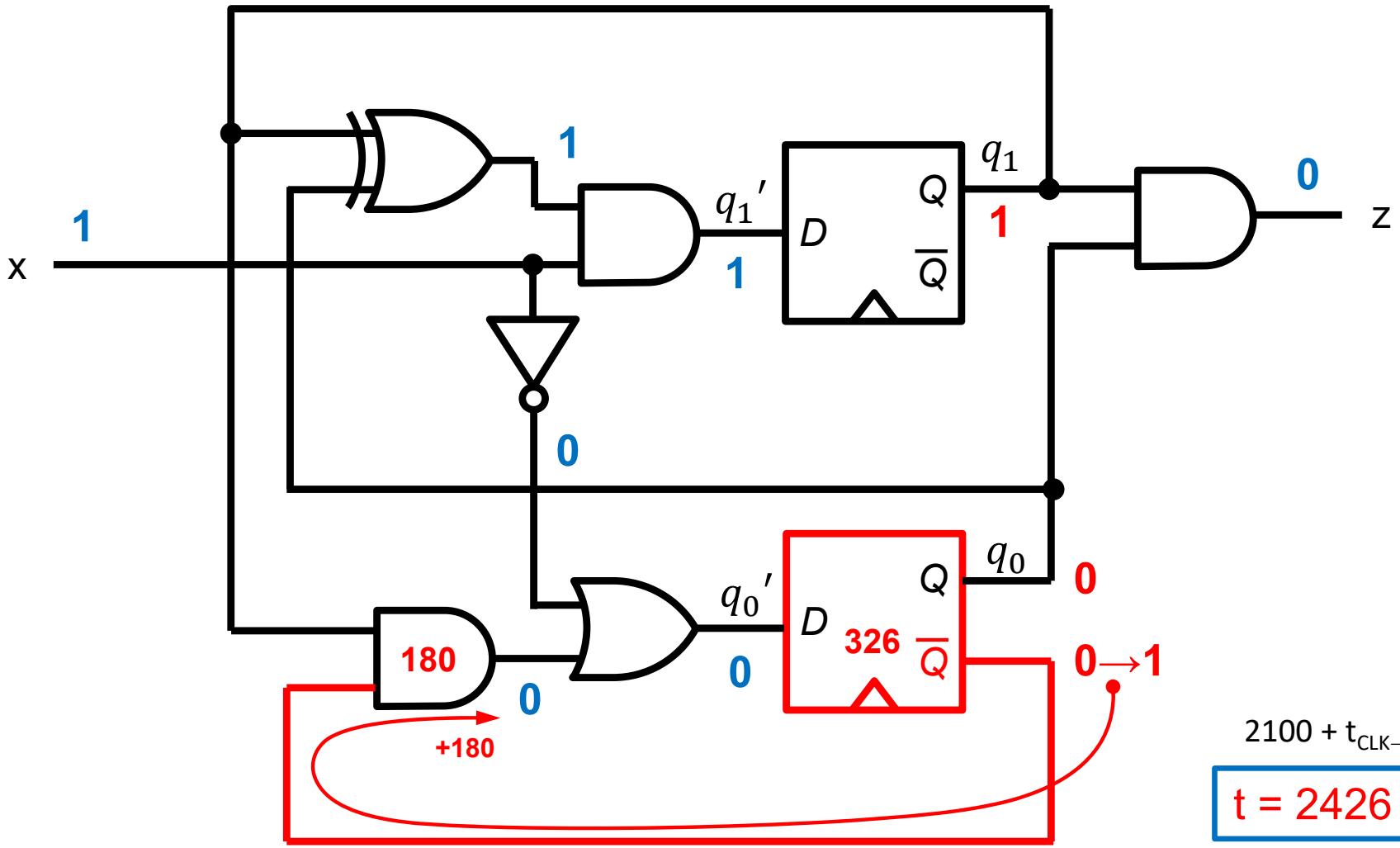
2100 + $t_{CLK \rightarrow Q}$

t = 2267 ps

4o. ciclo de reloj

Simulación

CMOS 90 nm



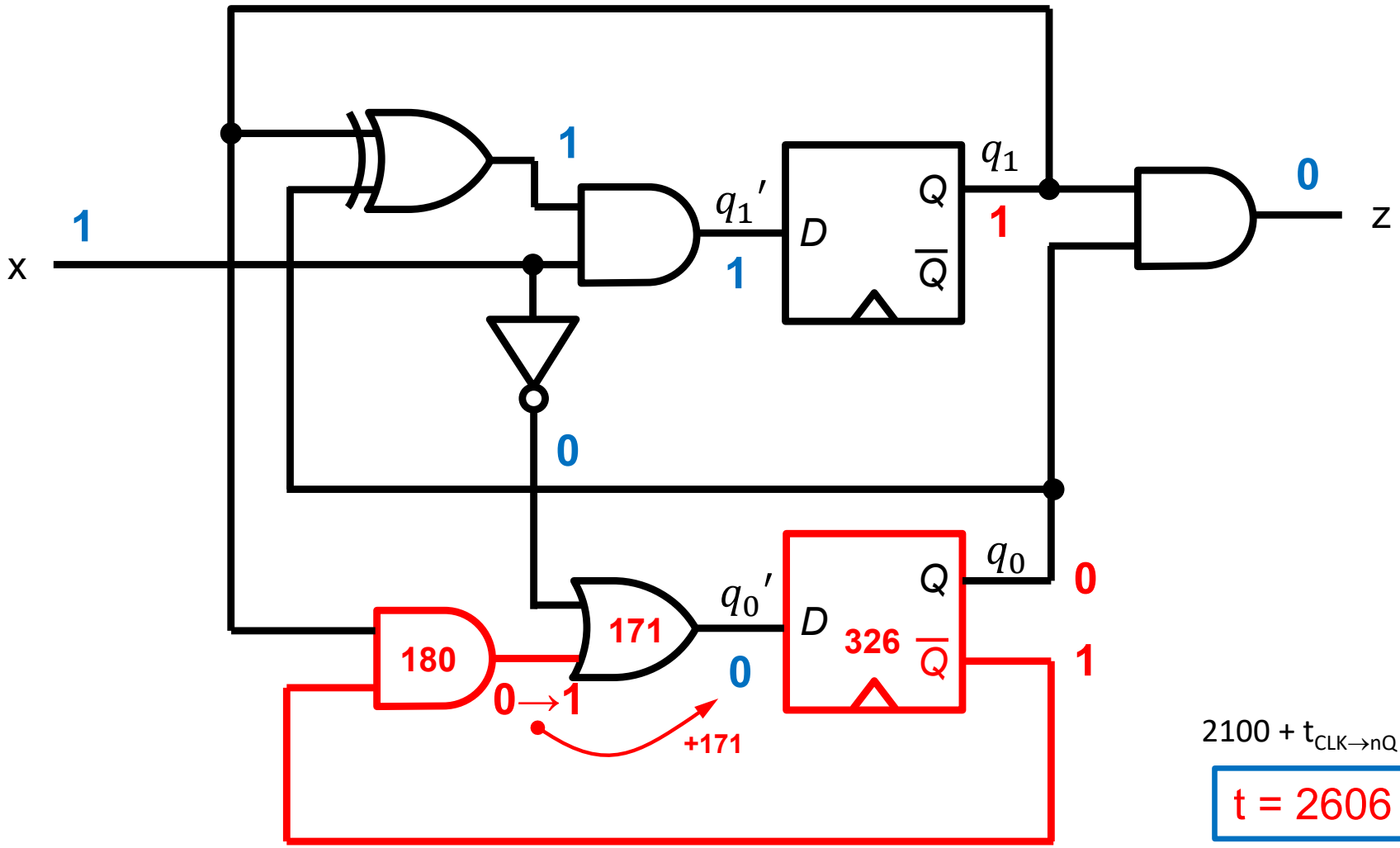
$2100 + t_{CLK \rightarrow nQ}$

t = 2426 ps

4o. ciclo de reloj

Simulación

CMOS 90 nm



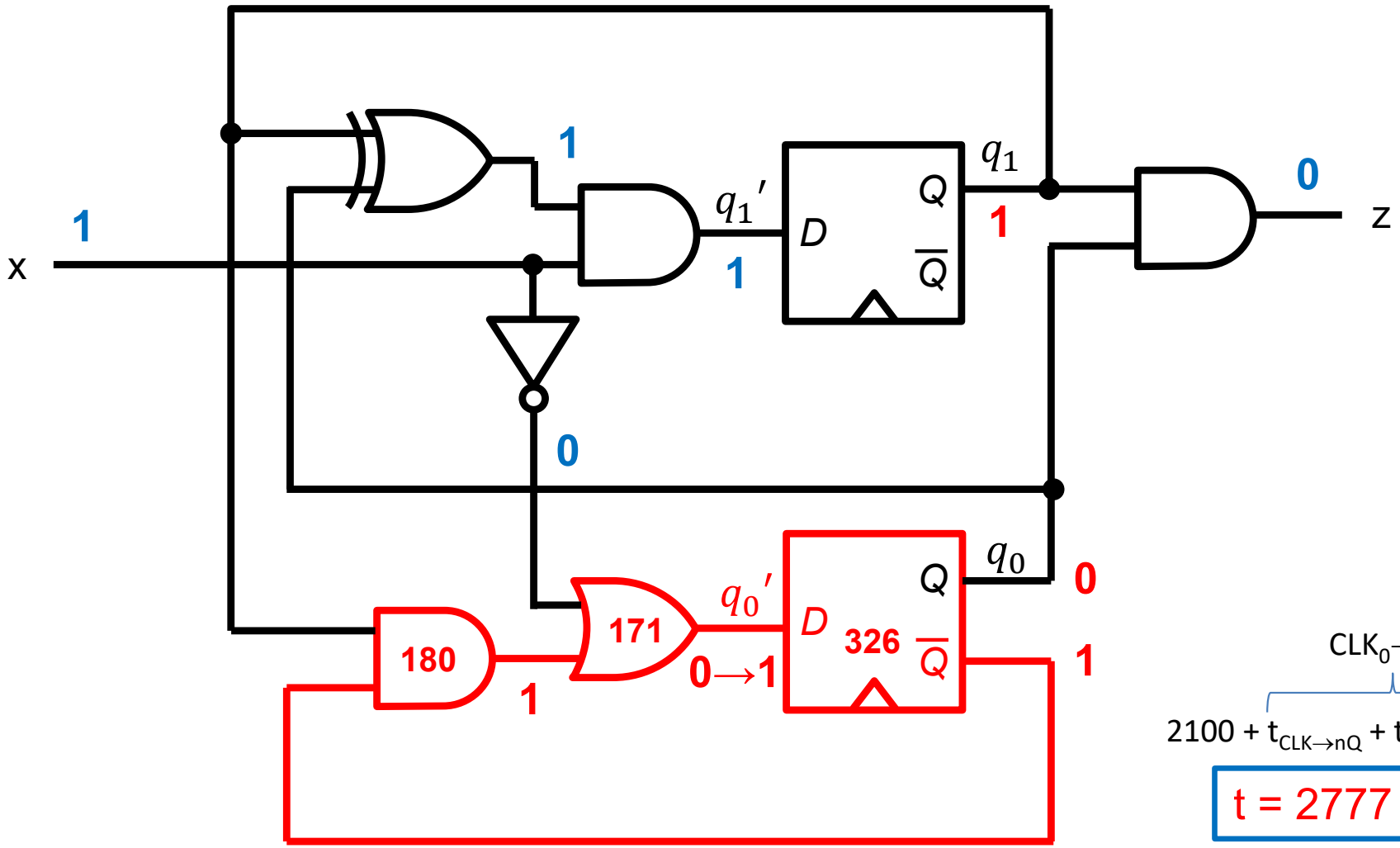
$2100 + t_{CLK \rightarrow nQ} + t_{AND}$

t = 2606 ps

4o. ciclo de reloj

Simulación

CMOS 90 nm



$CLK_0 \rightarrow D_0$

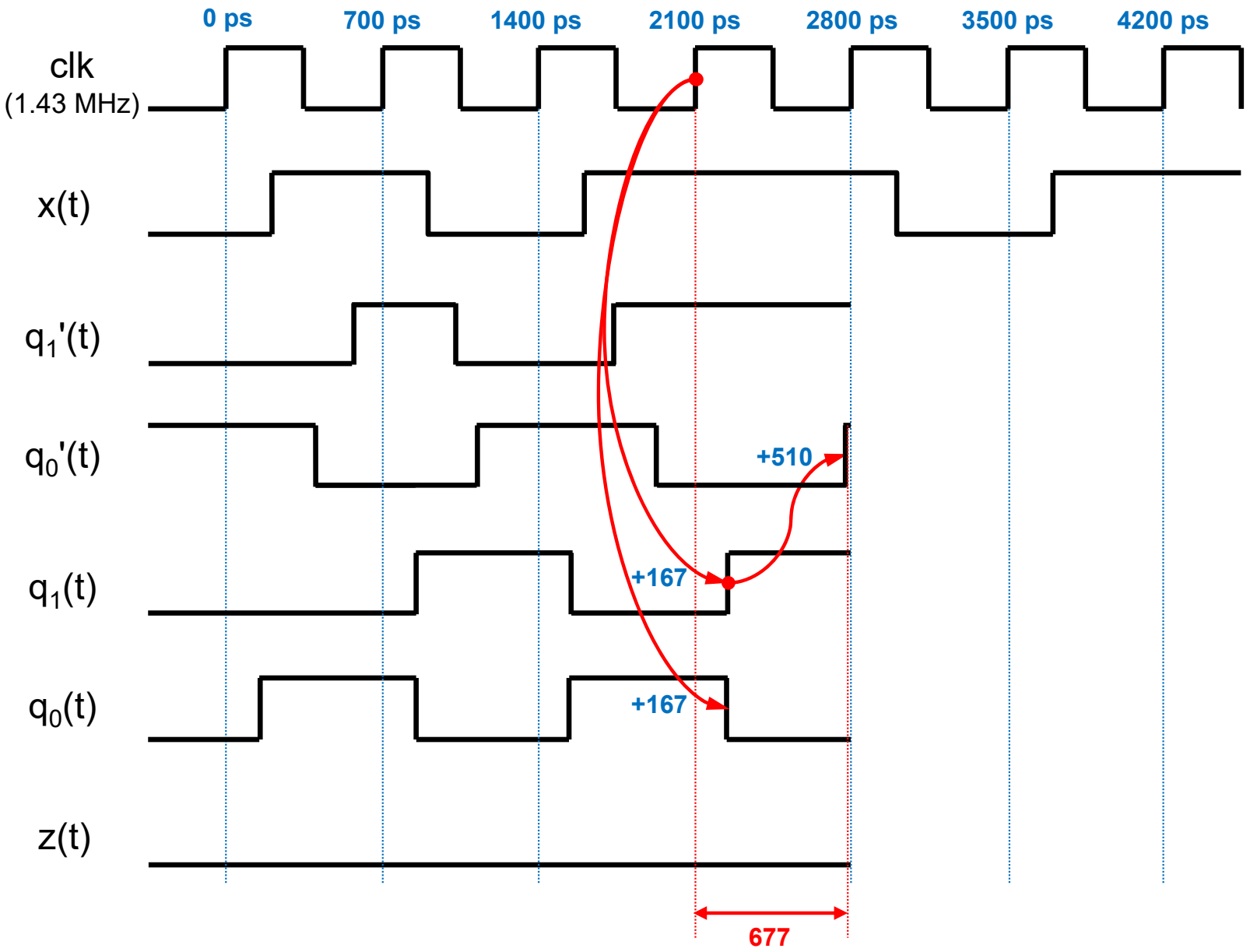
$2100 + t_{CLK \rightarrow nQ} + t_{AND} + t_{OR}$

t = 2777 ps

4o. ciclo de reloj

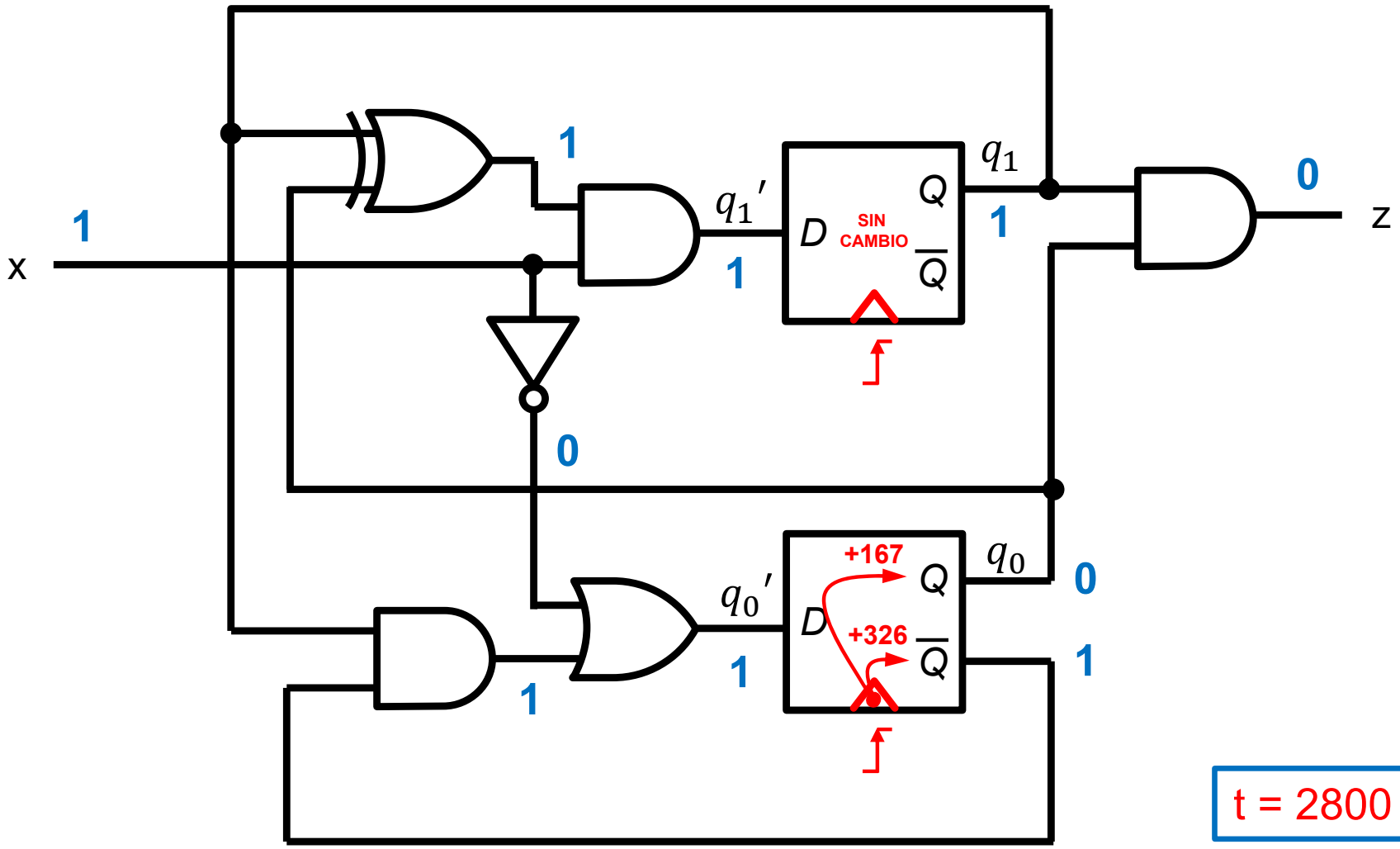
Simulación

CMOS 90 nm



Simulación

CMOS 90 nm

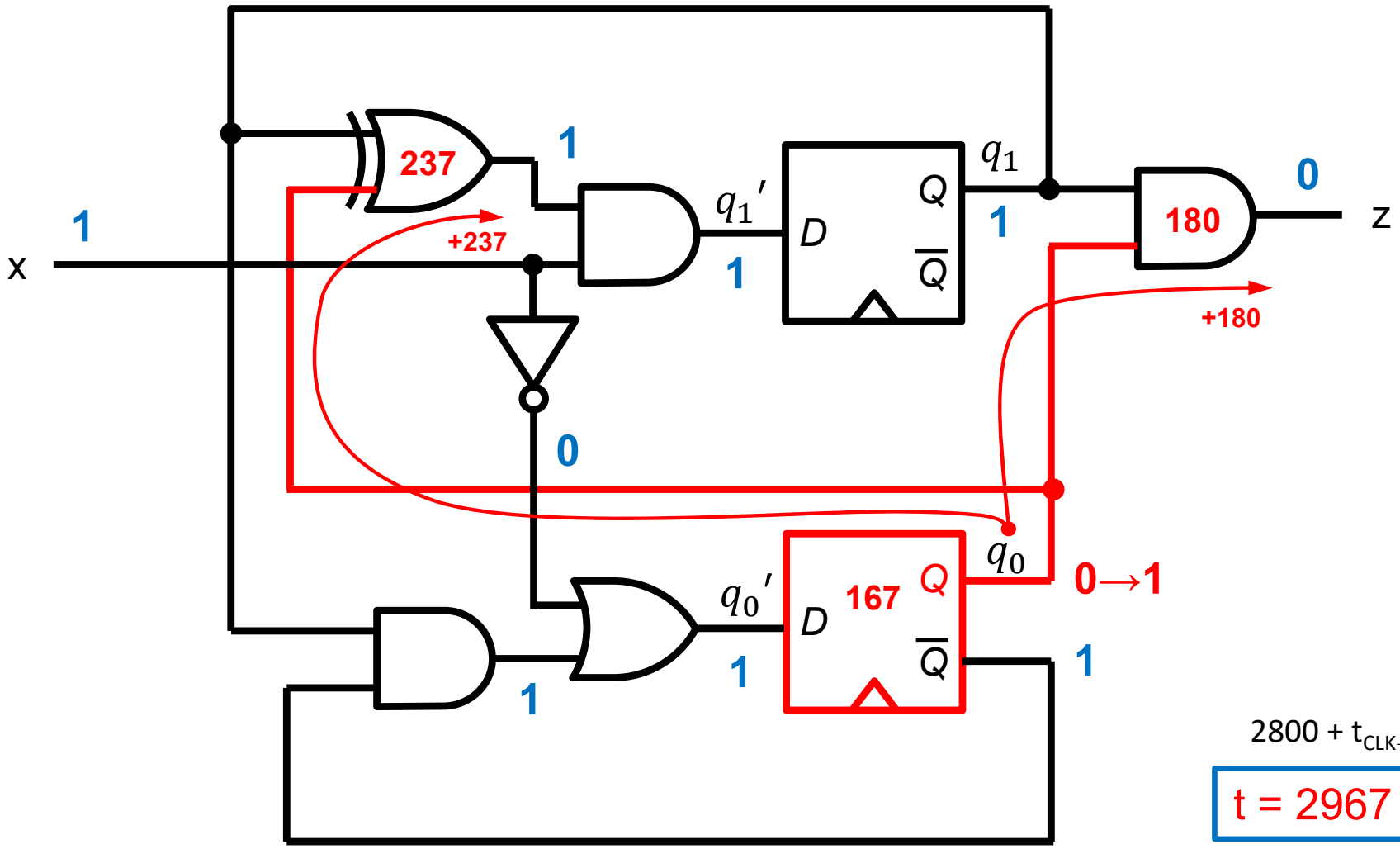


t = 2800 ps

5o. ciclo de reloj

Simulación

CMOS 90 nm



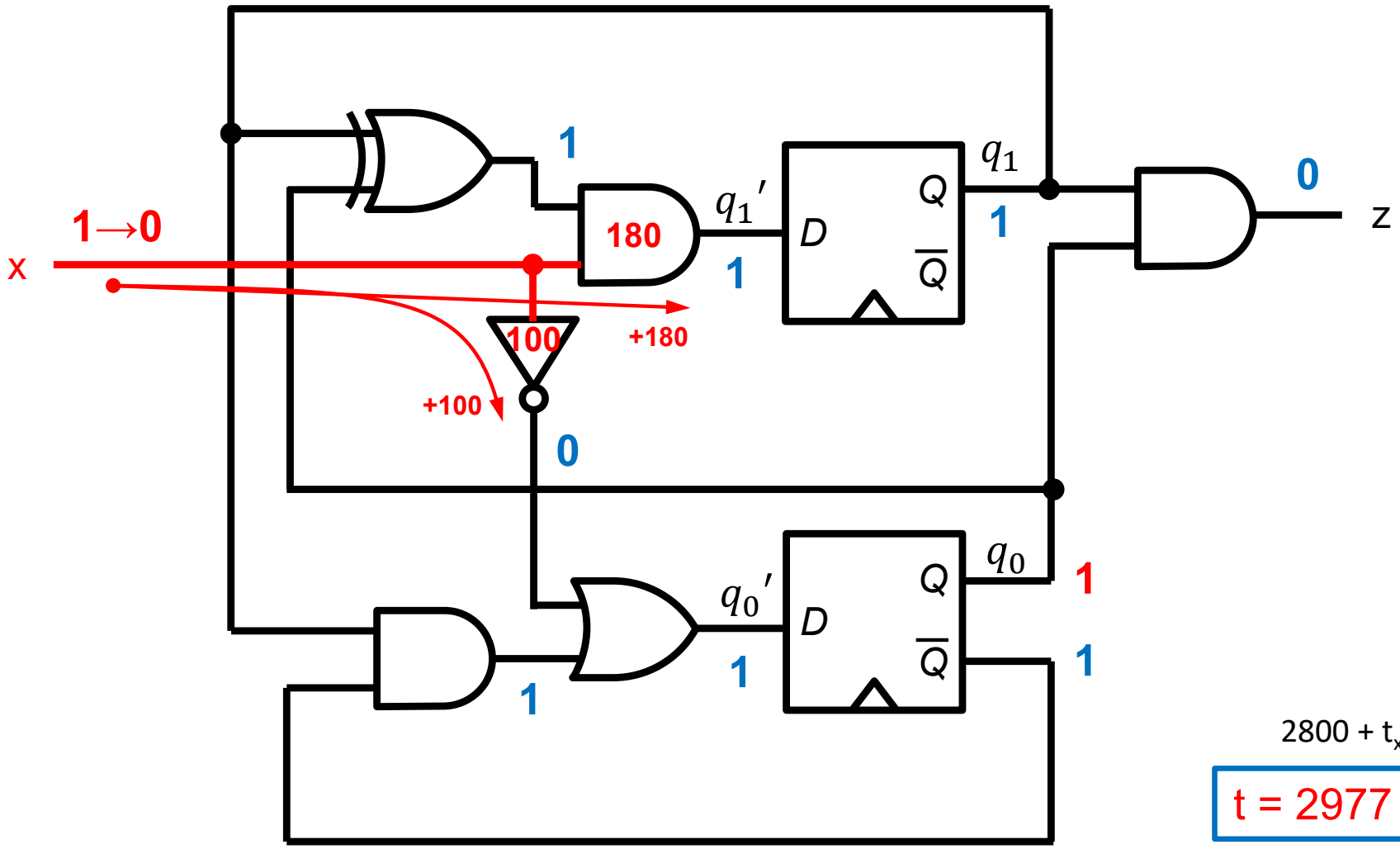
$2800 + t_{CLK \rightarrow Q}$

t = 2967 ps

5o. ciclo de reloj

Simulación

CMOS 90 nm



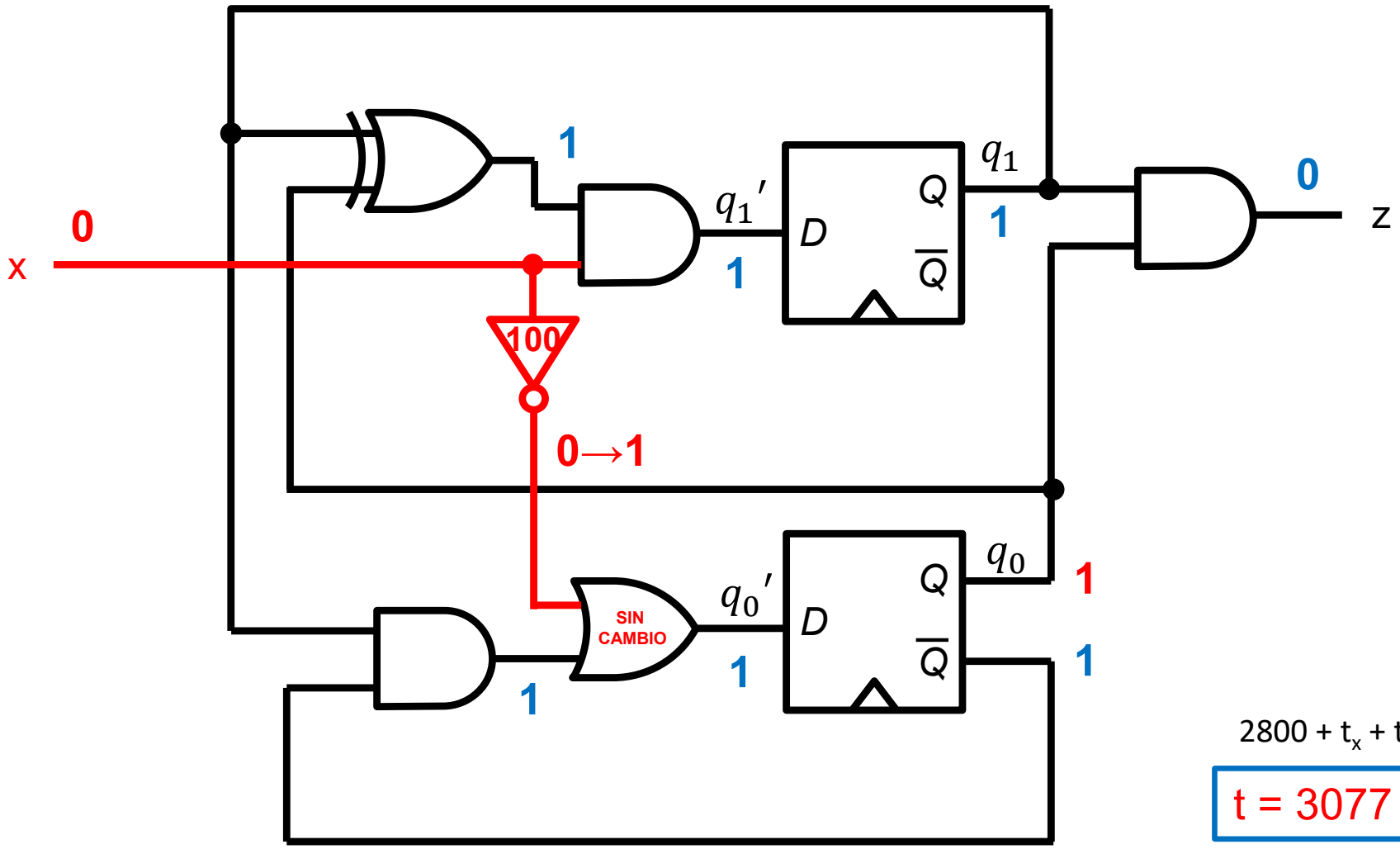
2800 + t_x

t = 2977 ps

5o. ciclo de reloj

Simulación

CMOS 90 nm



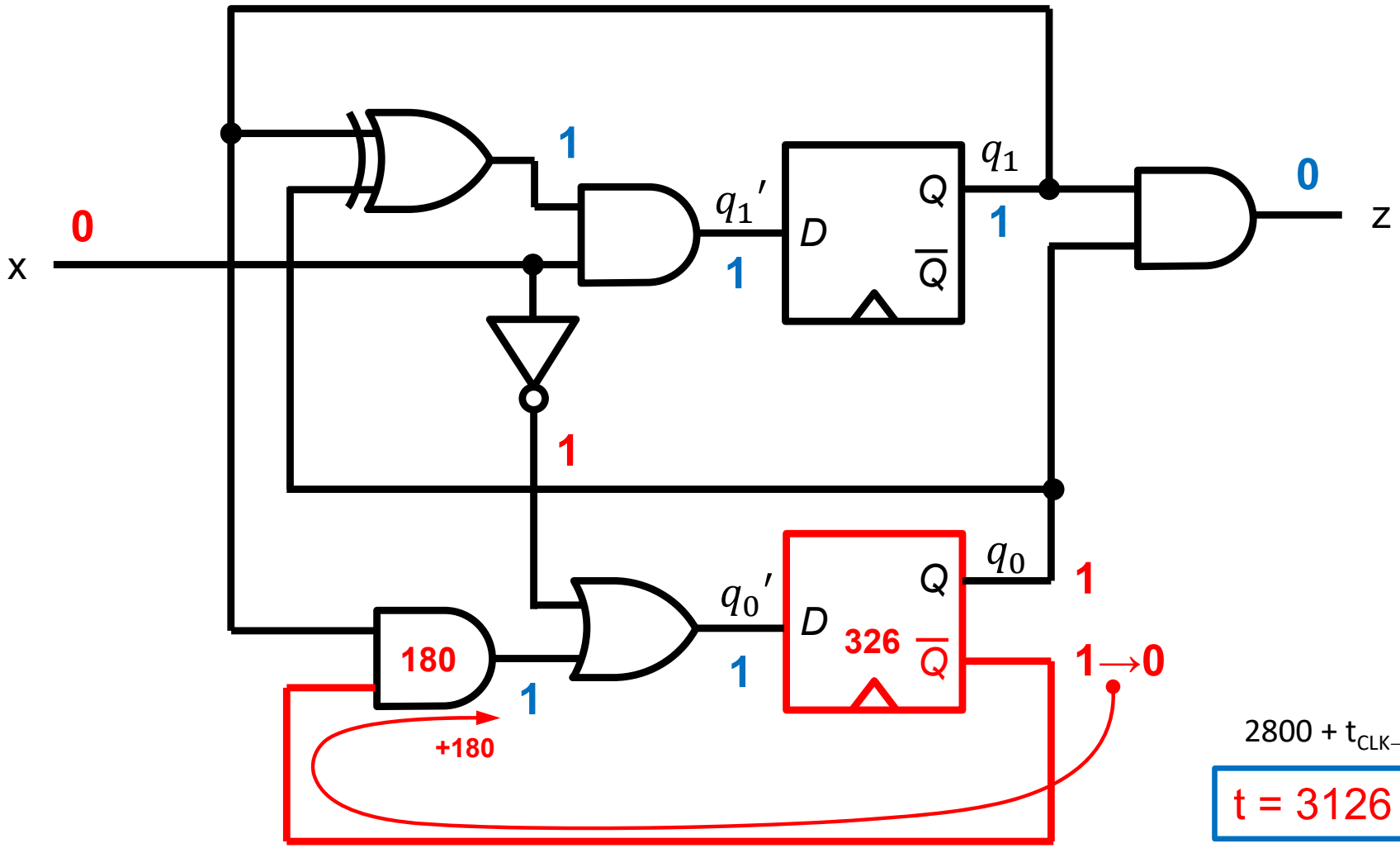
$2800 + t_x + t_{NOT}$

t = 3077 ps

5o. ciclo de reloj

Simulación

CMOS 90 nm



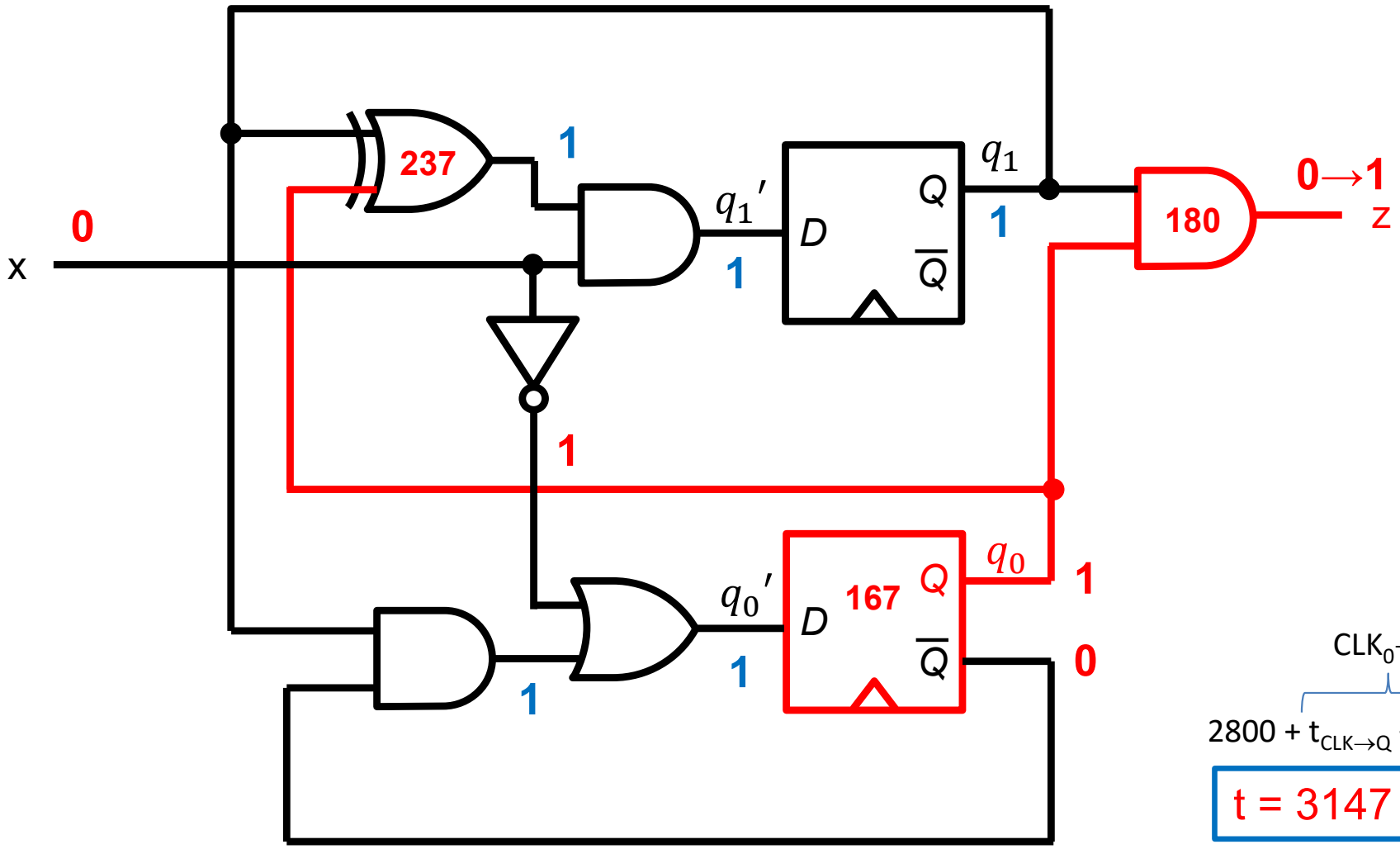
$2800 + t_{CLK \rightarrow nQ}$

t = 3126 ps

5o. ciclo de reloj

Simulación

CMOS 90 nm



$CLK_0 \rightarrow z$

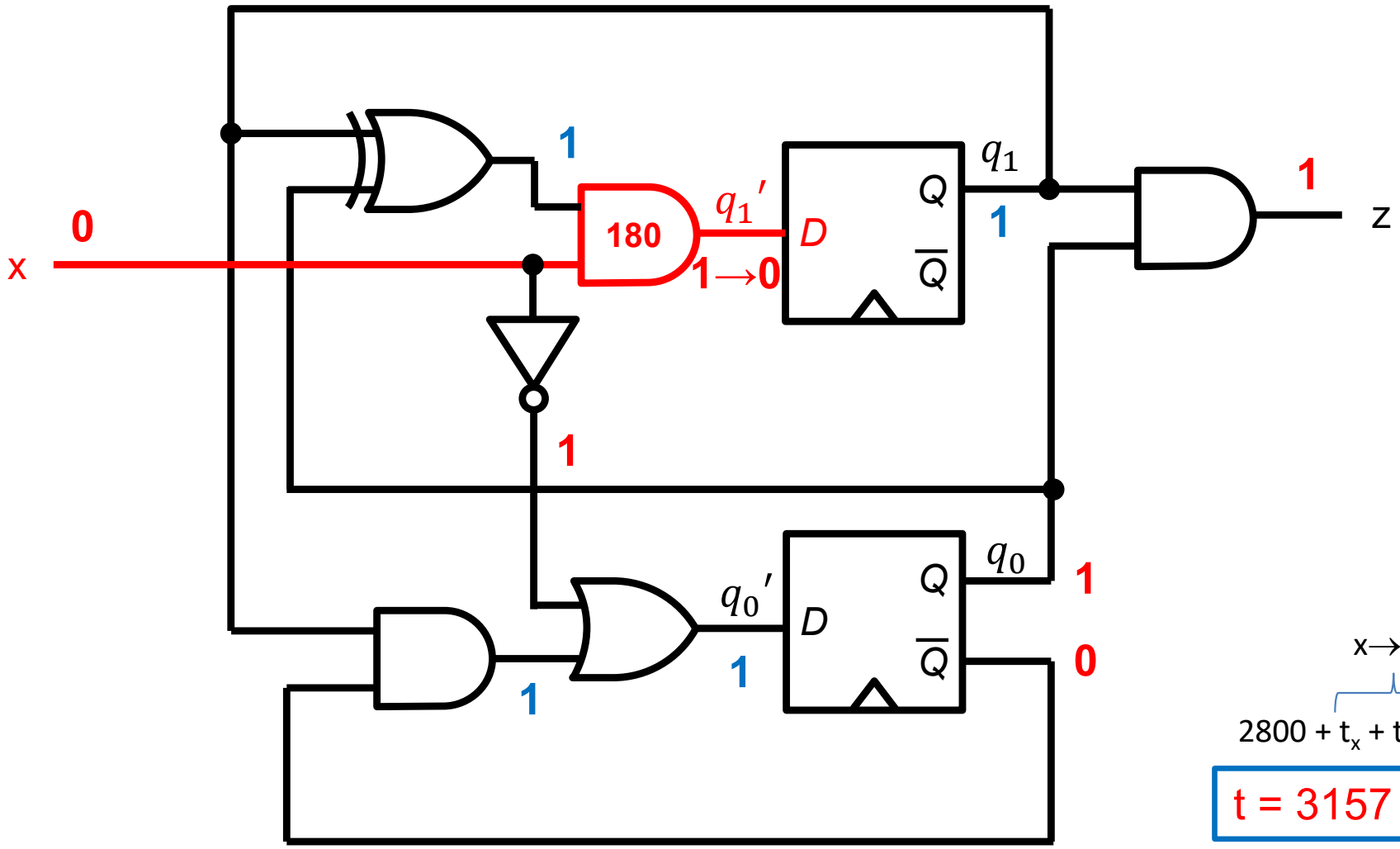
$2800 + t_{CLK \rightarrow Q} + t_{AND}$

t = 3147 ps

5o. ciclo de reloj

Simulación

CMOS 90 nm



$x \rightarrow D_1$

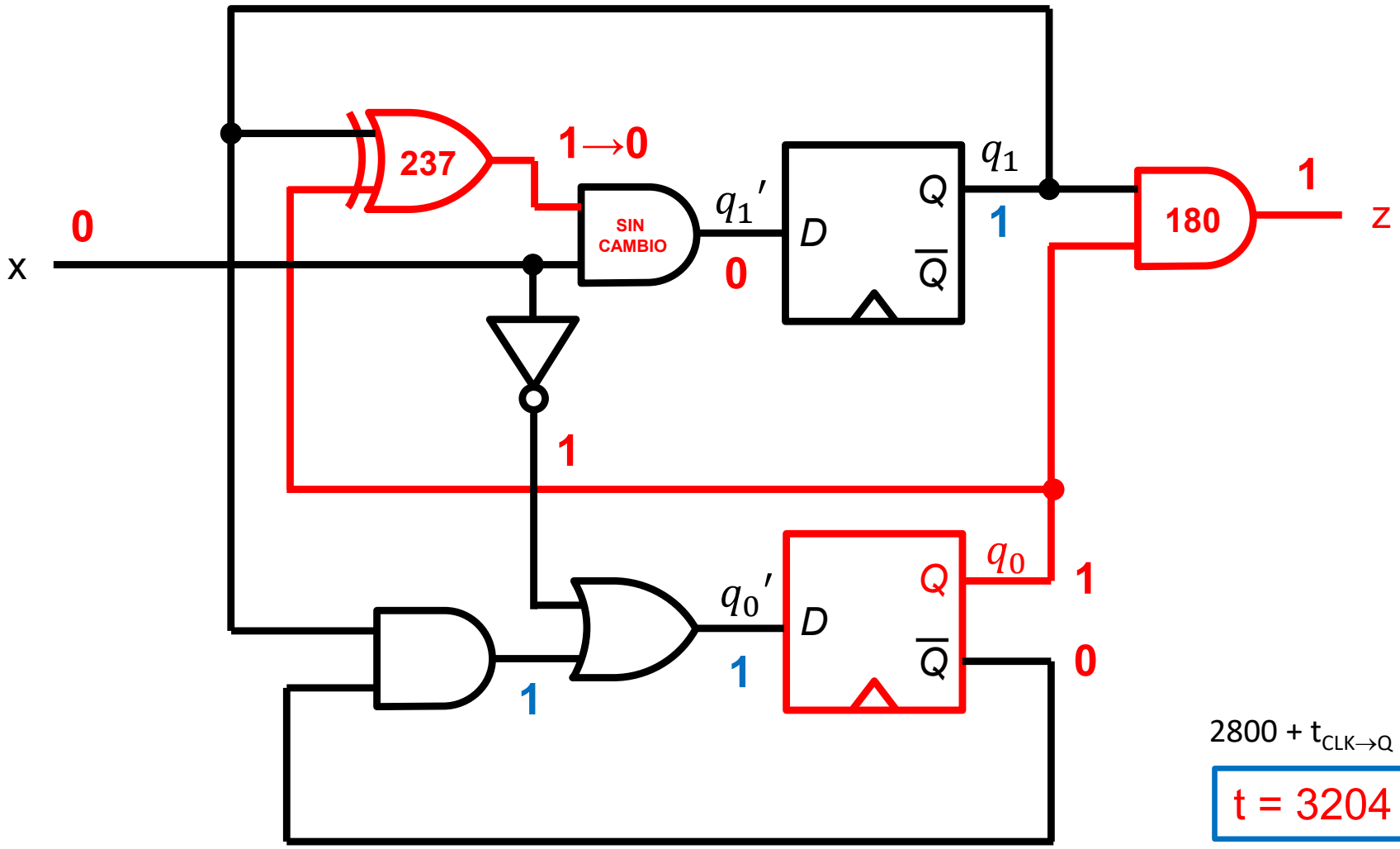
$2800 + t_x + t_{AND}$

t = 3157 ps

5o. ciclo de reloj

Simulación

CMOS 90 nm



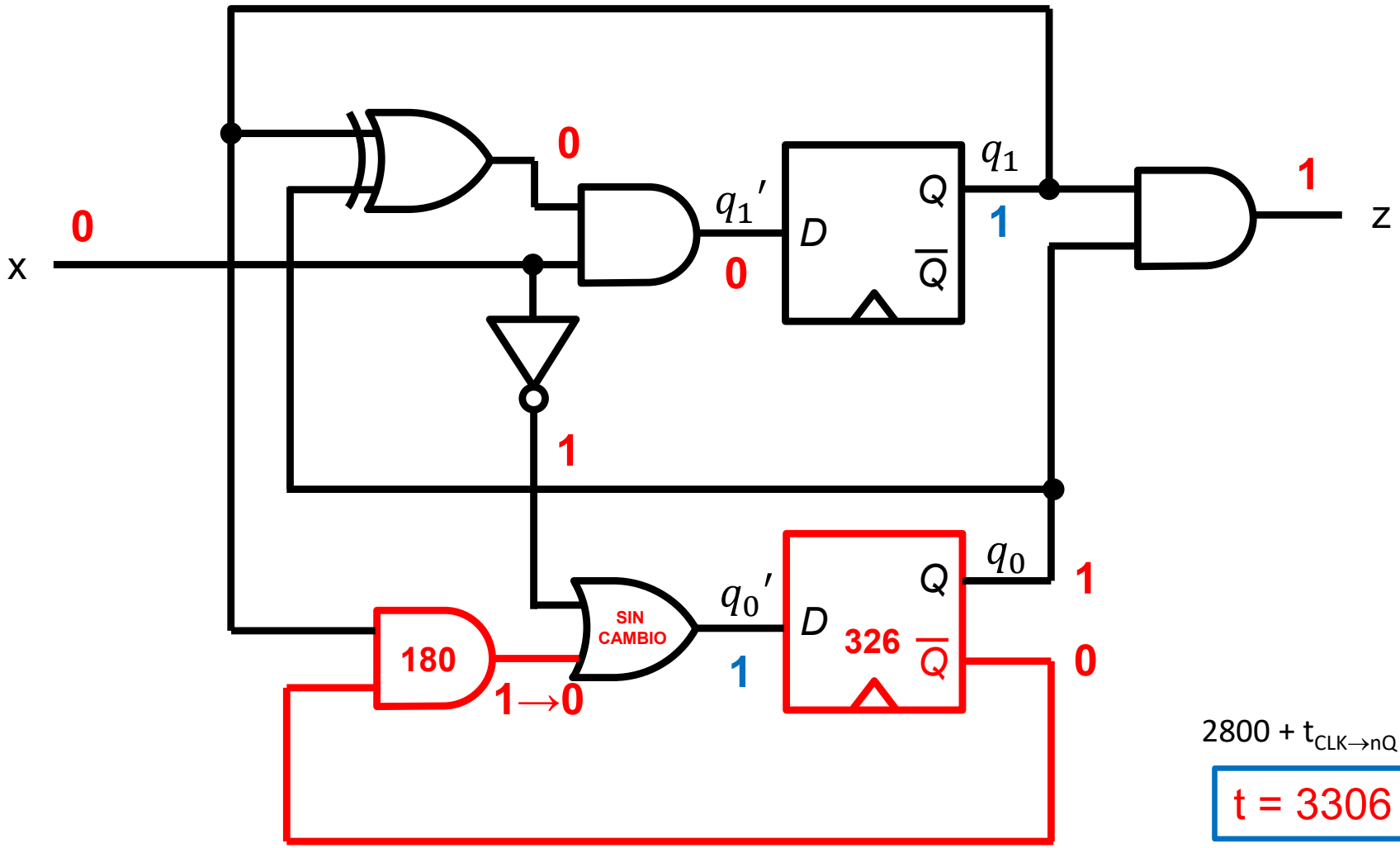
$2800 + t_{CLK \rightarrow Q} + t_{XOR}$

t = 3204 ps

5o. ciclo de reloj

Simulación

CMOS 90 nm



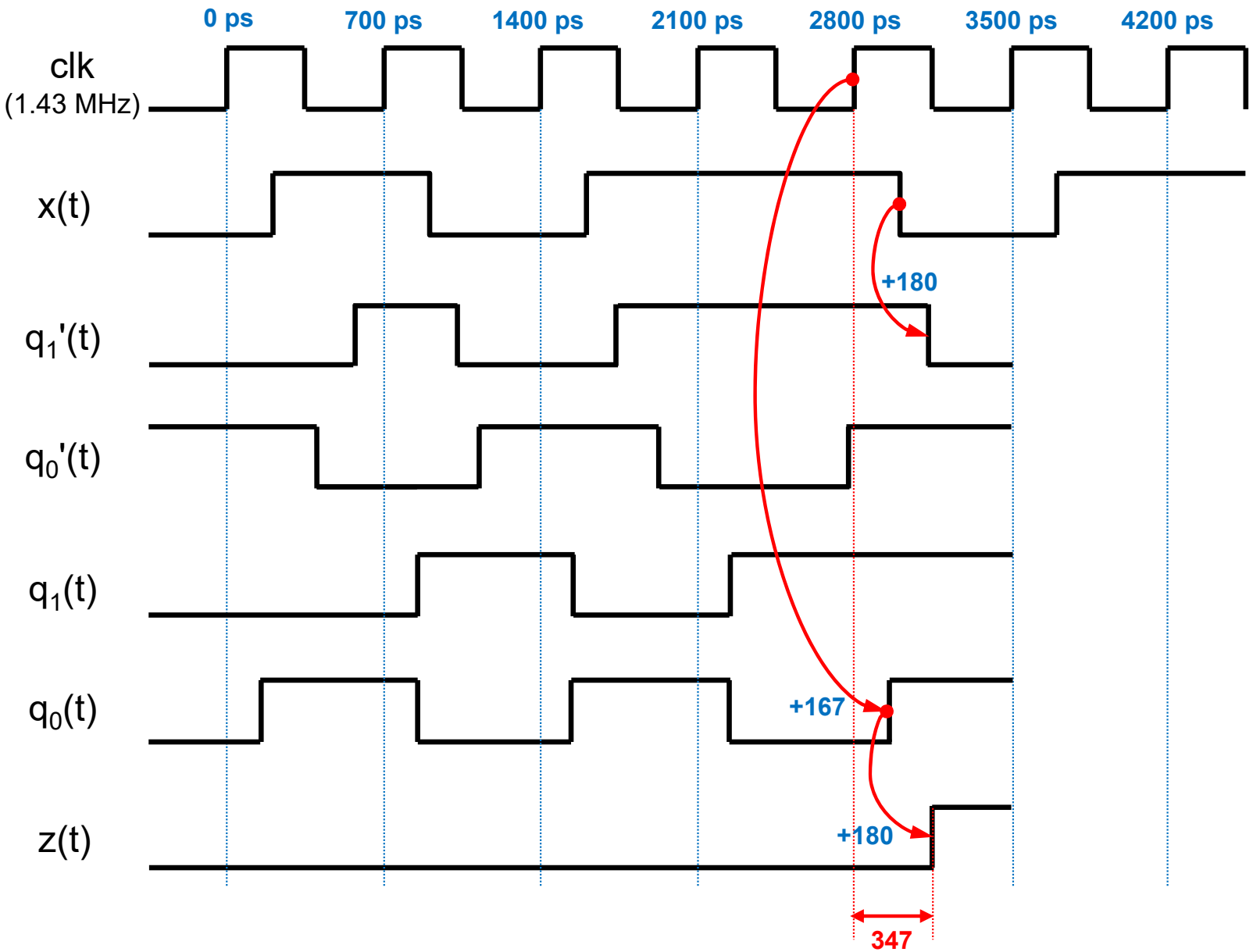
$2800 + t_{CLK \rightarrow nQ} + t_{AND}$

t = 3306 ps

5o. ciclo de reloj

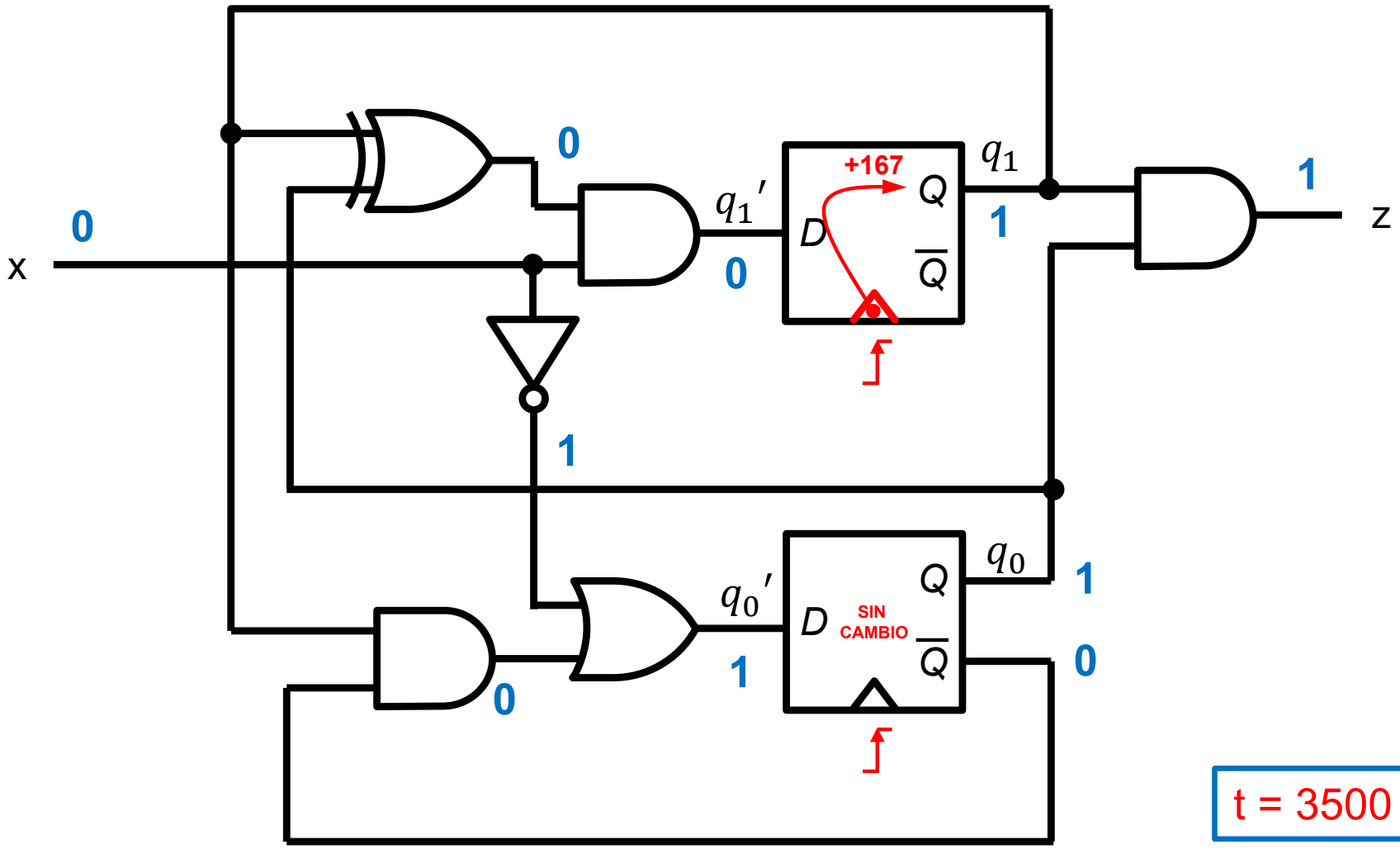
Simulación

CMOS 90 nm



Simulación

CMOS 90 nm

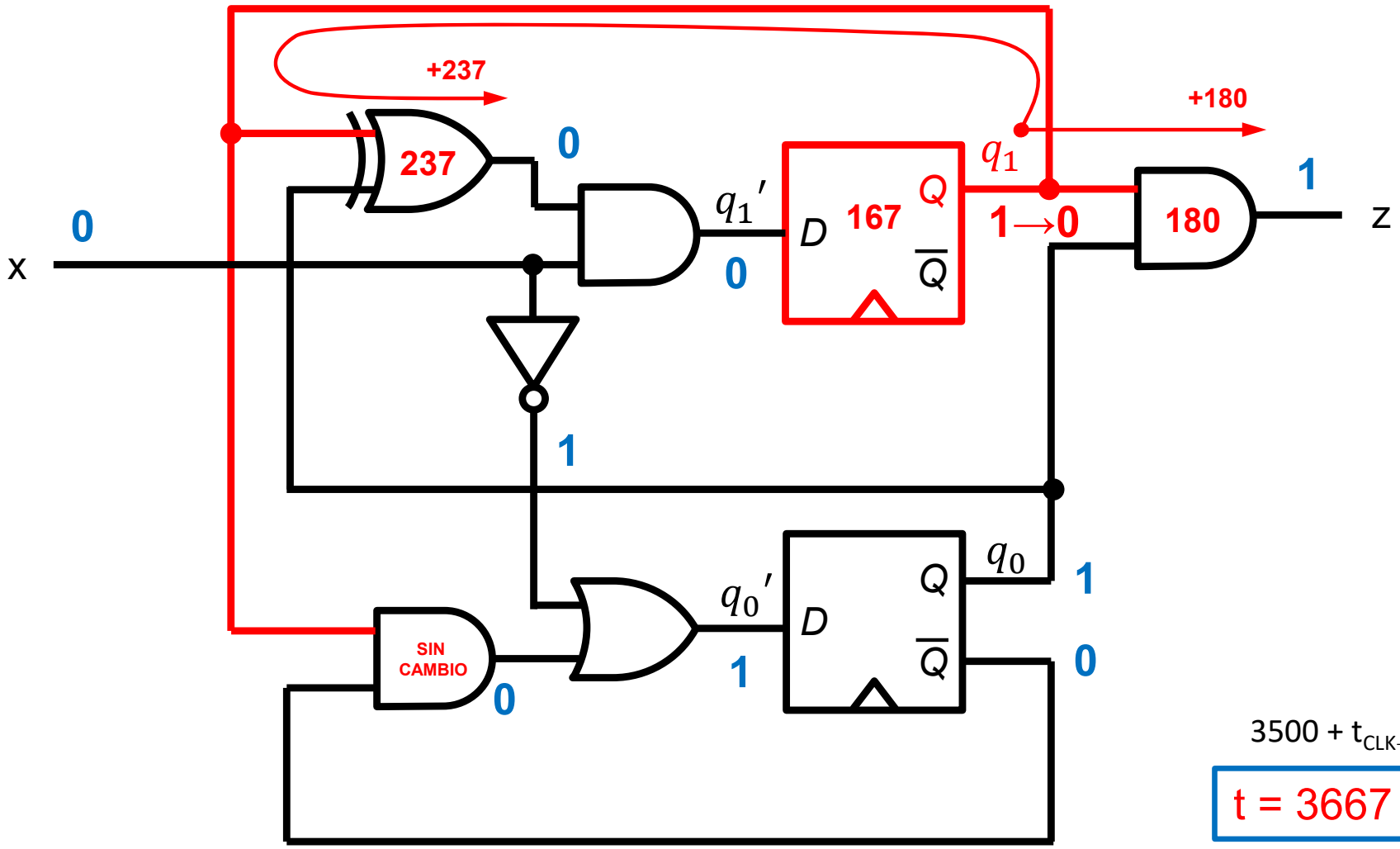


t = 3500 ps

6o. ciclo de reloj

Simulación

CMOS 90 nm



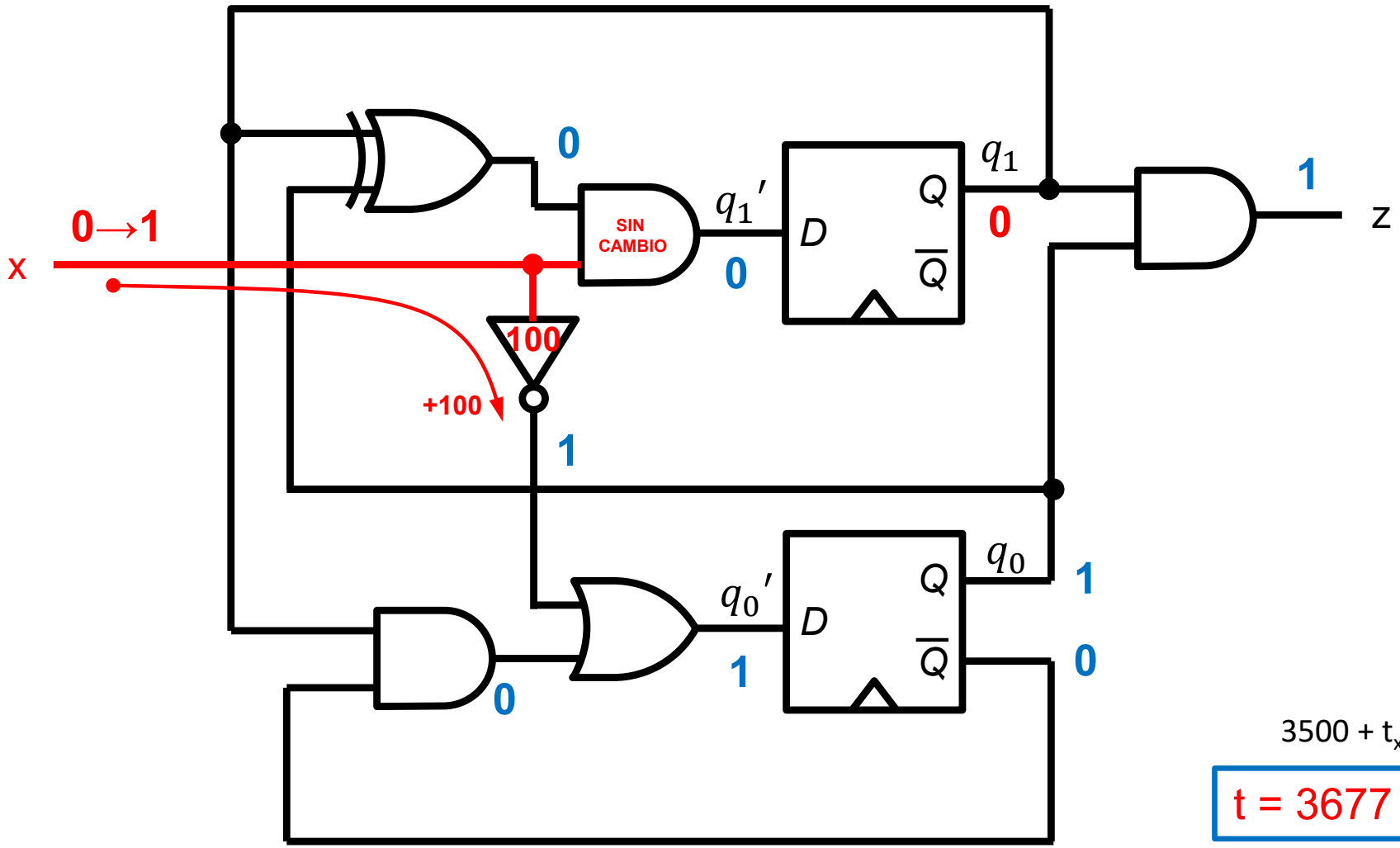
3500 + $t_{CLK \rightarrow Q}$

t = 3667 ps

6o. ciclo de reloj

Simulación

CMOS 90 nm



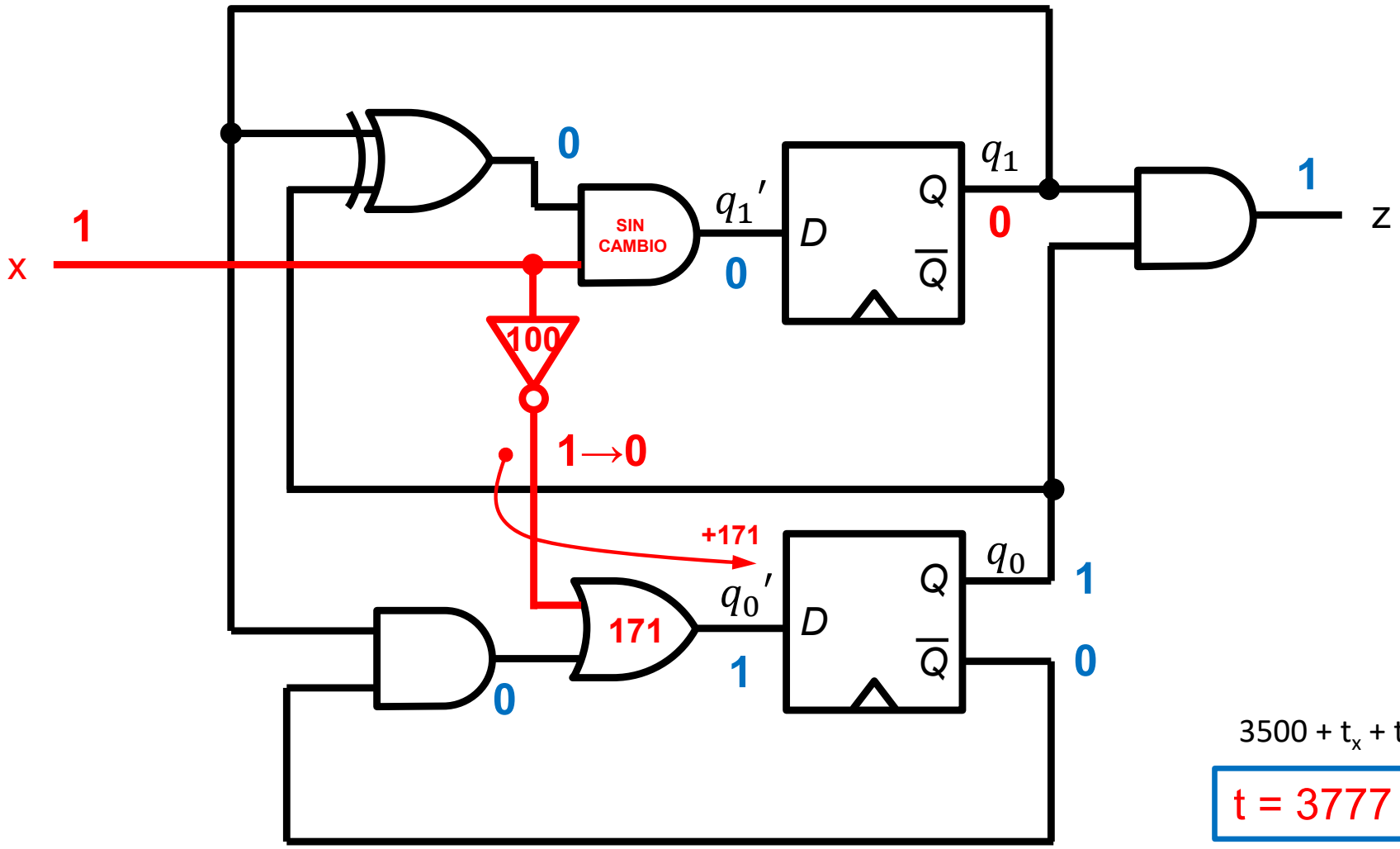
3500 + t_x

t = 3677 ps

6o. ciclo de reloj

Simulación

CMOS 90 nm



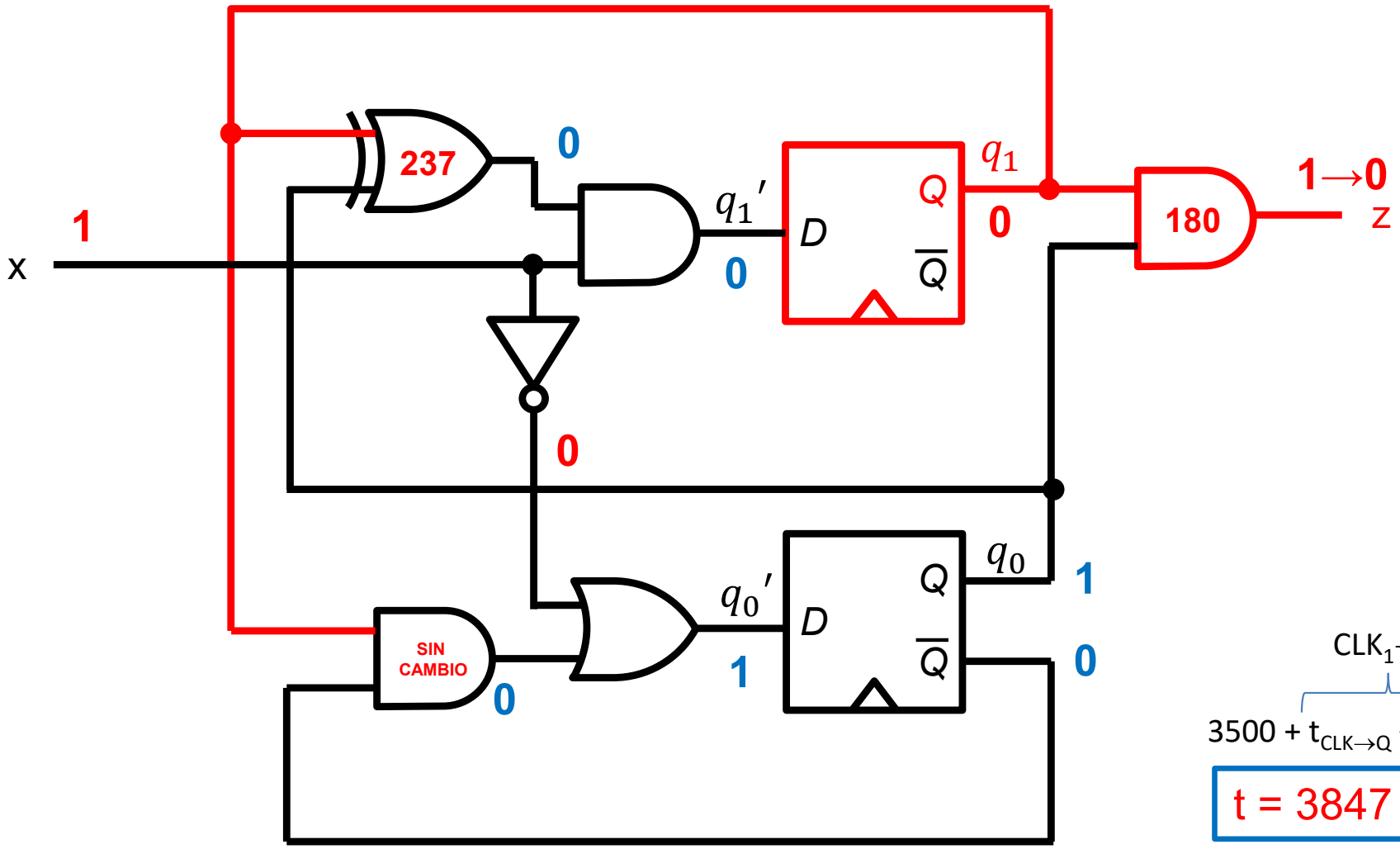
$3500 + t_x + t_{NOT}$

t = 3777 ps

6o. ciclo de reloj

Simulación

CMOS 90 nm



$CLK_1 \rightarrow z$

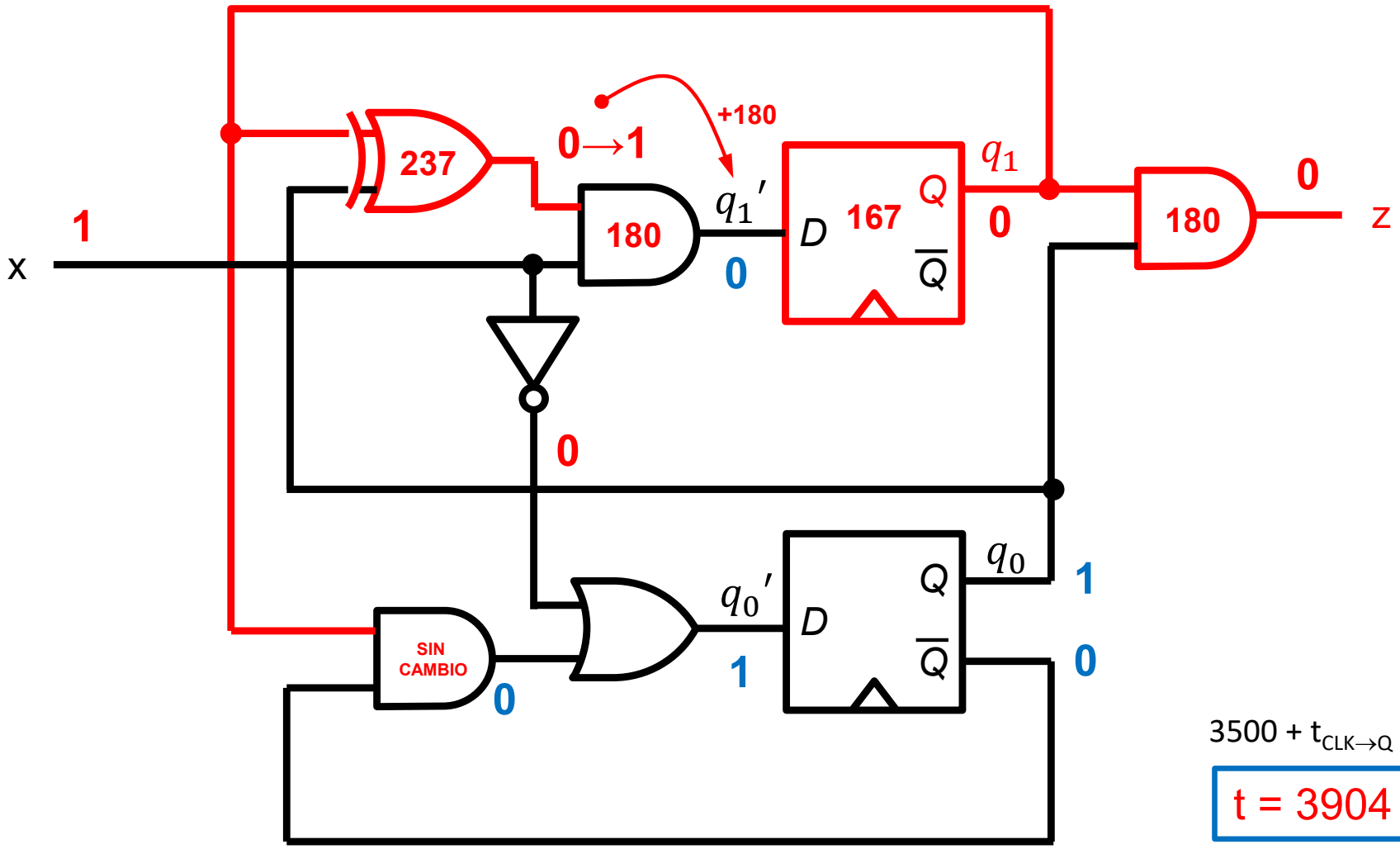
$3500 + t_{CLK \rightarrow Q} + t_{AND}$

t = 3847 ps

6o. ciclo de reloj

Simulación

CMOS 90 nm



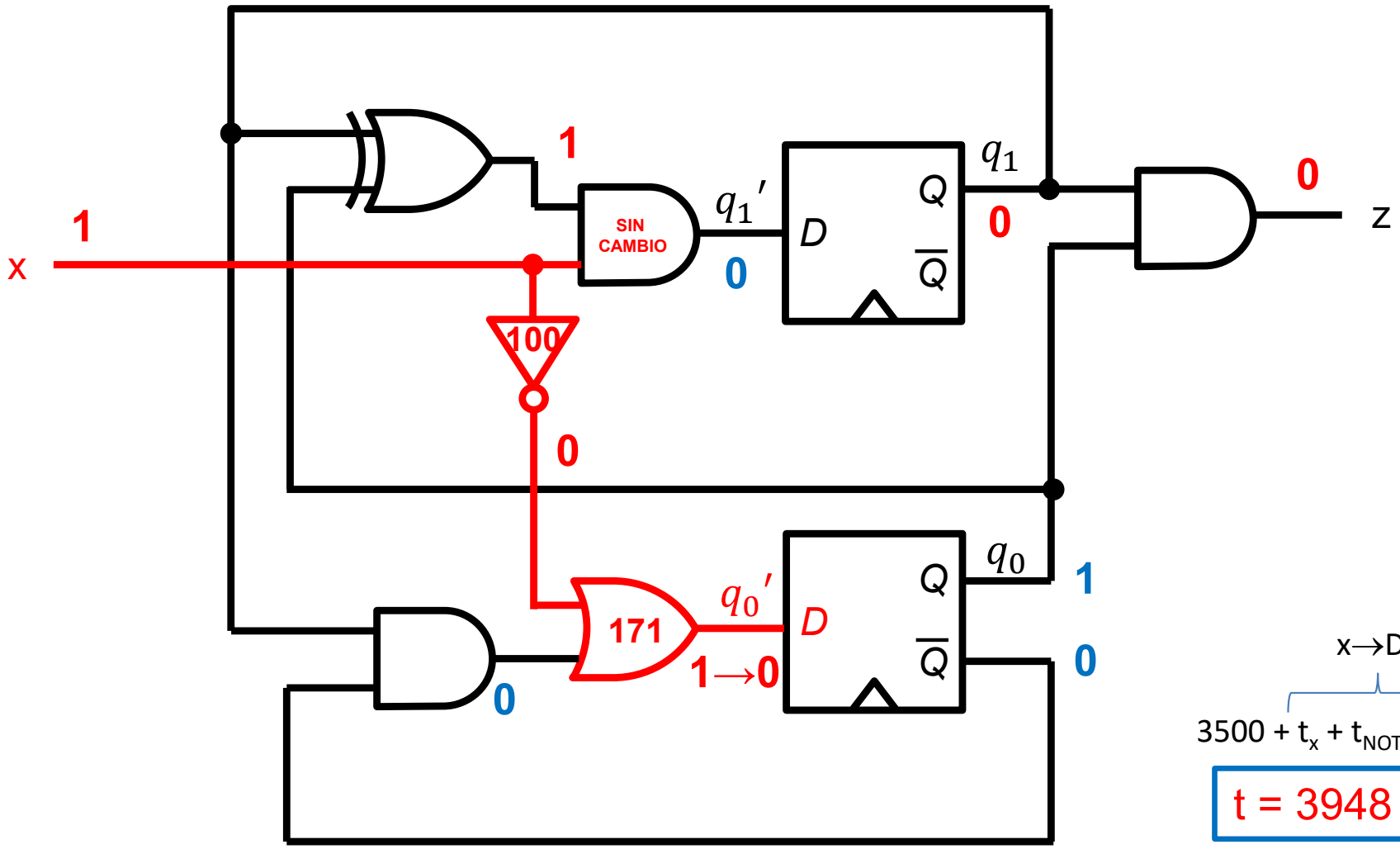
$3500 + t_{CLK \rightarrow Q} + t_{XOR}$

t = 3904 ps

6o. ciclo de reloj

Simulación

CMOS 90 nm



$x \rightarrow D_0$

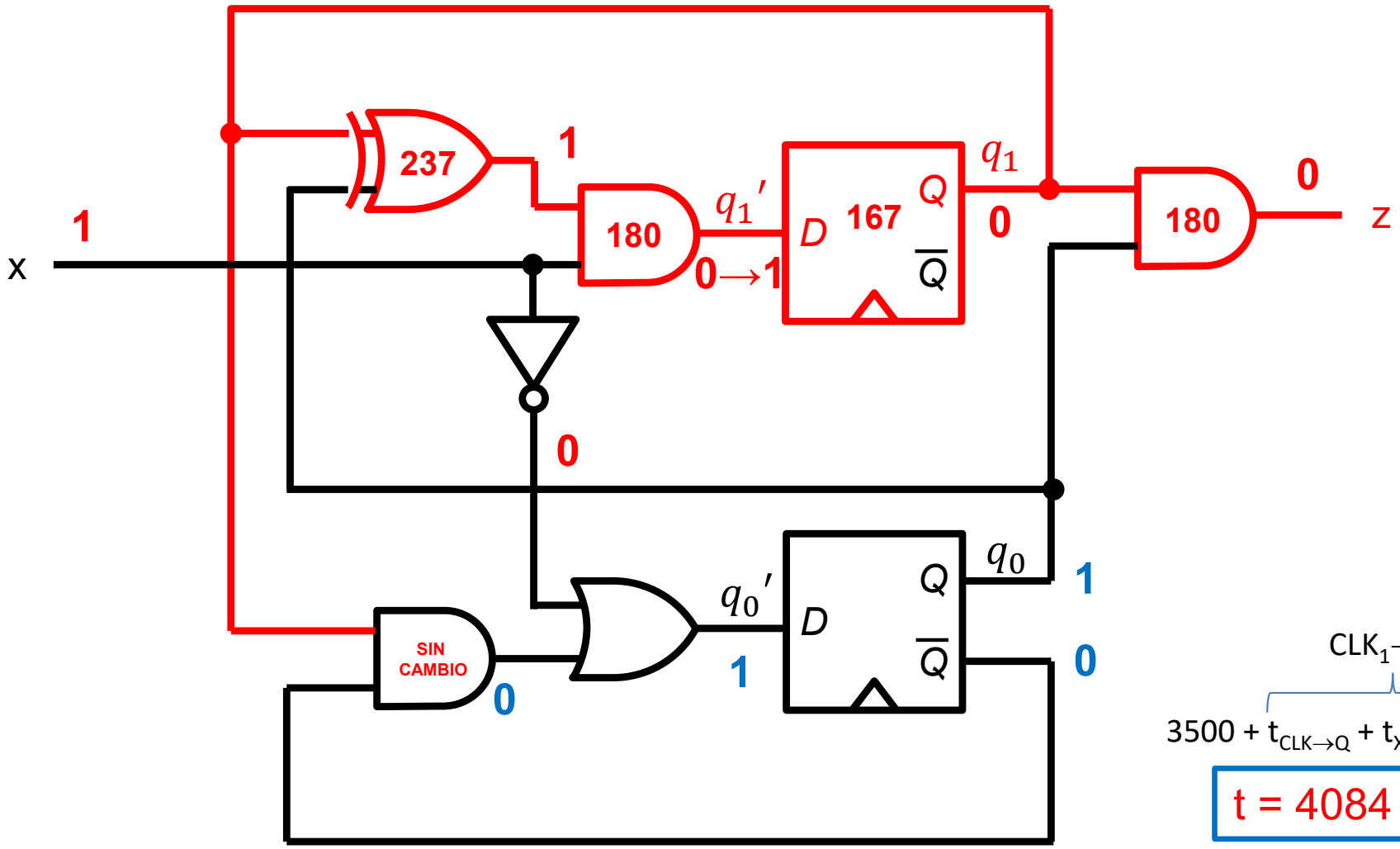
$3500 + t_x + t_{NOT} + t_{AND}$

t = 3948 ps

6o. ciclo de reloj

Simulación

CMOS 90 nm



$$\text{CLK}_1 \rightarrow \text{D}_1$$

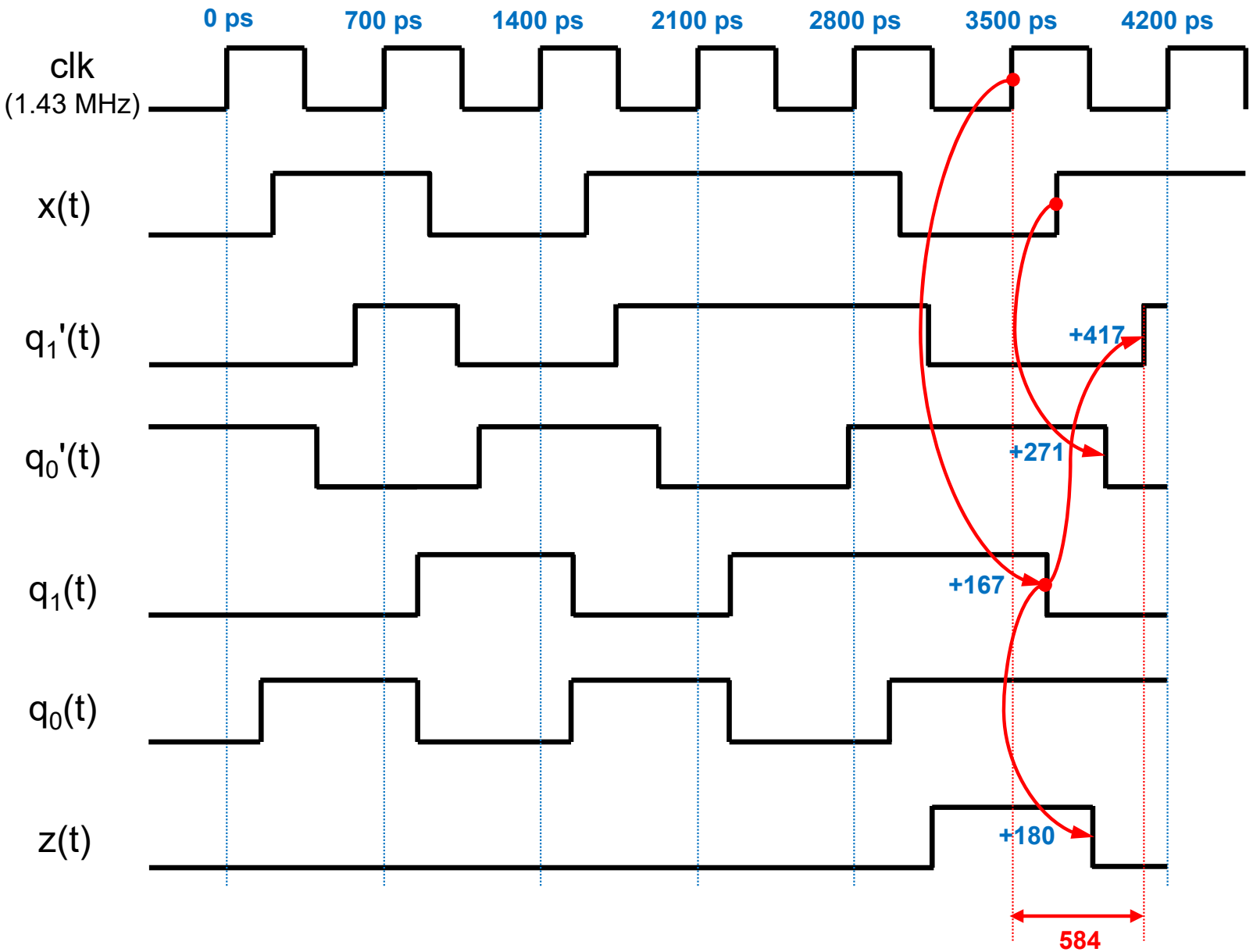
$$3500 + t_{CLK \rightarrow Q} + t_{XOR} + t_{AND}$$

t = 4084 ps

6o. ciclo de reloj

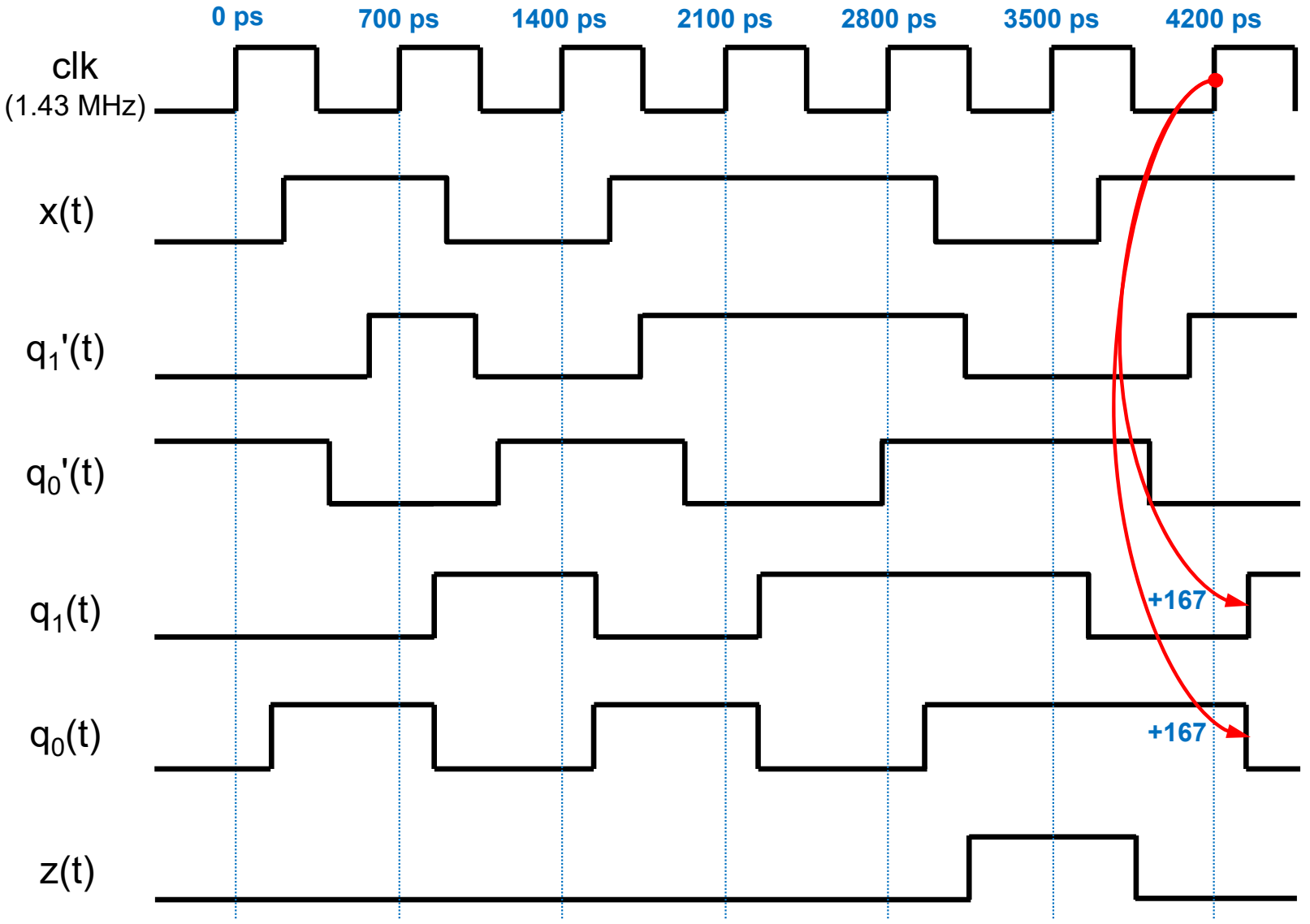
Simulación

CMOS 90 nm



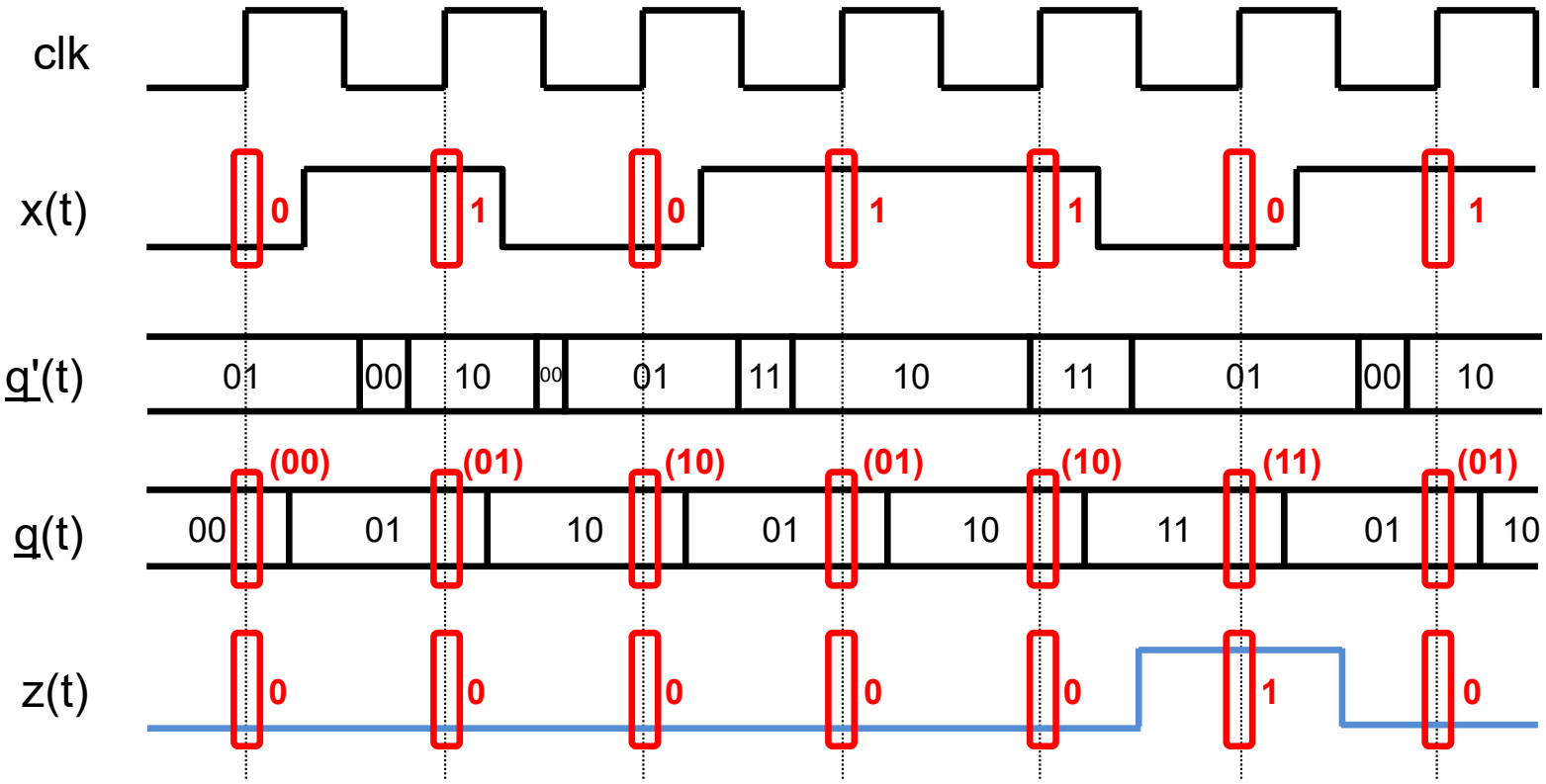
Simulación

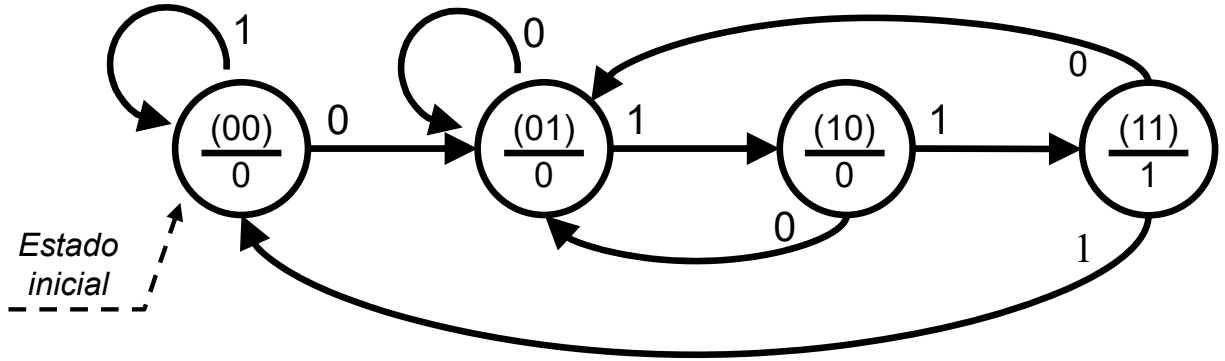
CMOS 90 nm



Simulación

CMOS 90 nm





Acerca de *Creative Commons*

- Licencia CC (Creative Commons)
  - Ofrece algunos derechos a terceras personas bajo ciertas condiciones. Este documento tiene establecidas las siguientes:

Reconocimiento (*Attribution*):
En cualquier explotación de la obra autorizada por la licencia hará falta reconocer la autoría.

No comercial (*Non commercial*):
La explotación de la obra queda limitada a usos no comerciales.

Compartir igual (*Share alike*):
La explotación autorizada incluye la creación de obras derivadas siempre que mantengan la misma licencia al ser divulgadas.

Más información: https://creativecommons.org/licenses/by-nc-sa/4.0/