08.07.20 12:33

CSCI 150
Introduction to Digital and Computer System Design
Lecture 4: Sequential Circuit I

Jetic Gū
2020 Summer Semester (S2)

Overview

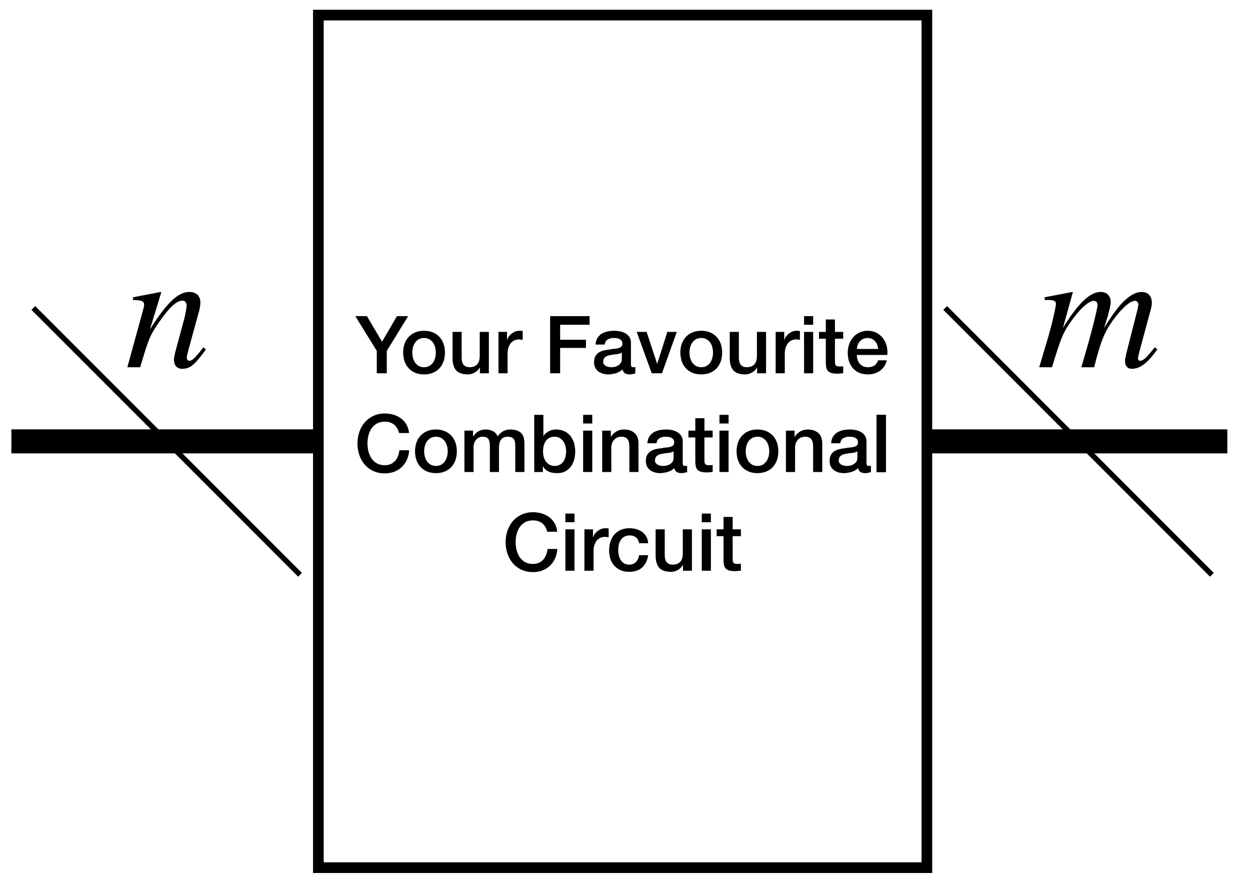
- Focus: Basic Information Retaining Blocks
- Architecture: Sequential Circuit
- Textbook v4: Ch5 5.1, 5.2; v5: Ch4 4.1 4.2
- Core Ideas:

1. Introduction
2. SR and $\overline{SR}$ Latches, D Latch

P0
Review

Combinational Logic Circuit Design

- Design Principles
  - Knows: fixed-Length input and output
  - Knows: input/output mapping relations
  - Optimisation: Minimise overall delay

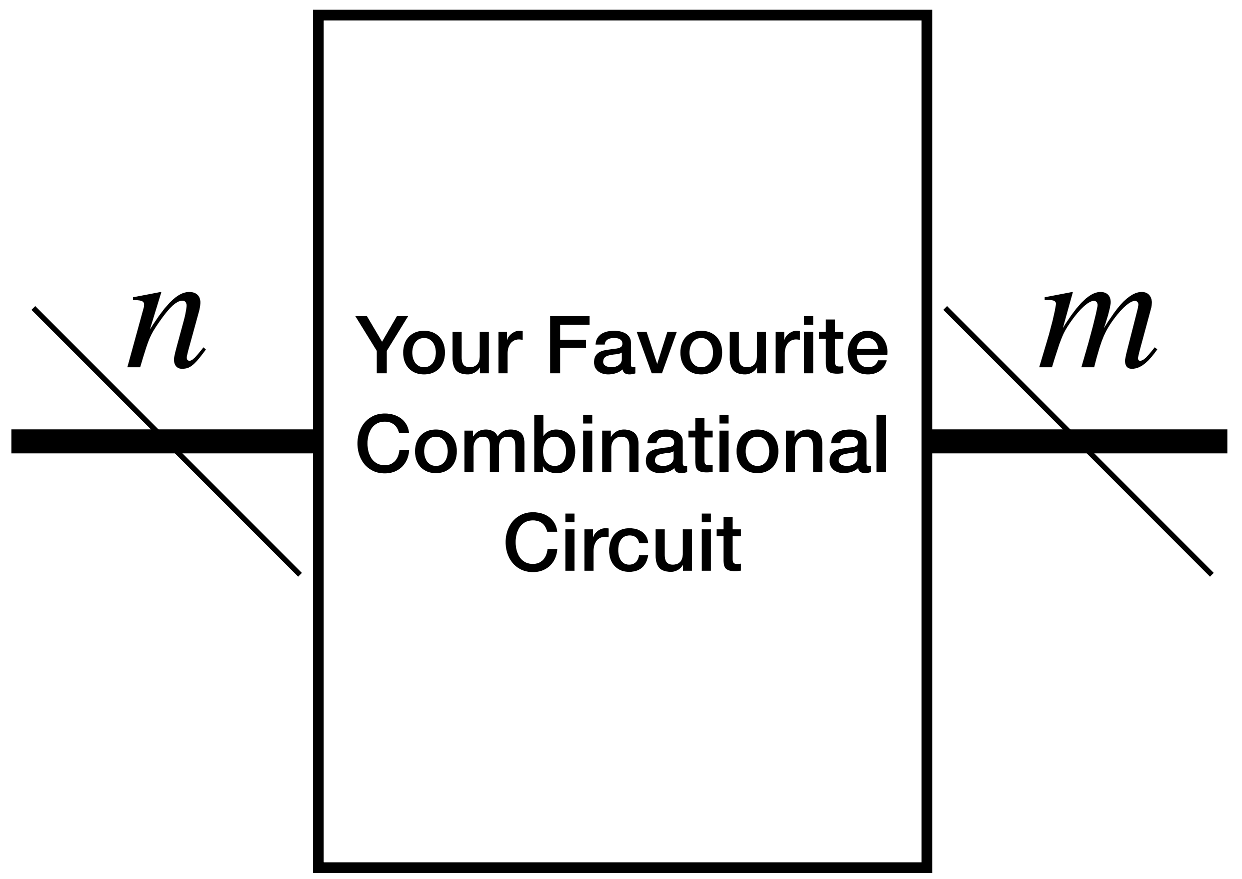


Review

P0
Review

Combinational Logic Circuit Design

- Features
  - Fixed-Length input and output
  - The same input will always give the same output
  - Operations are simultaneous with minimum delay

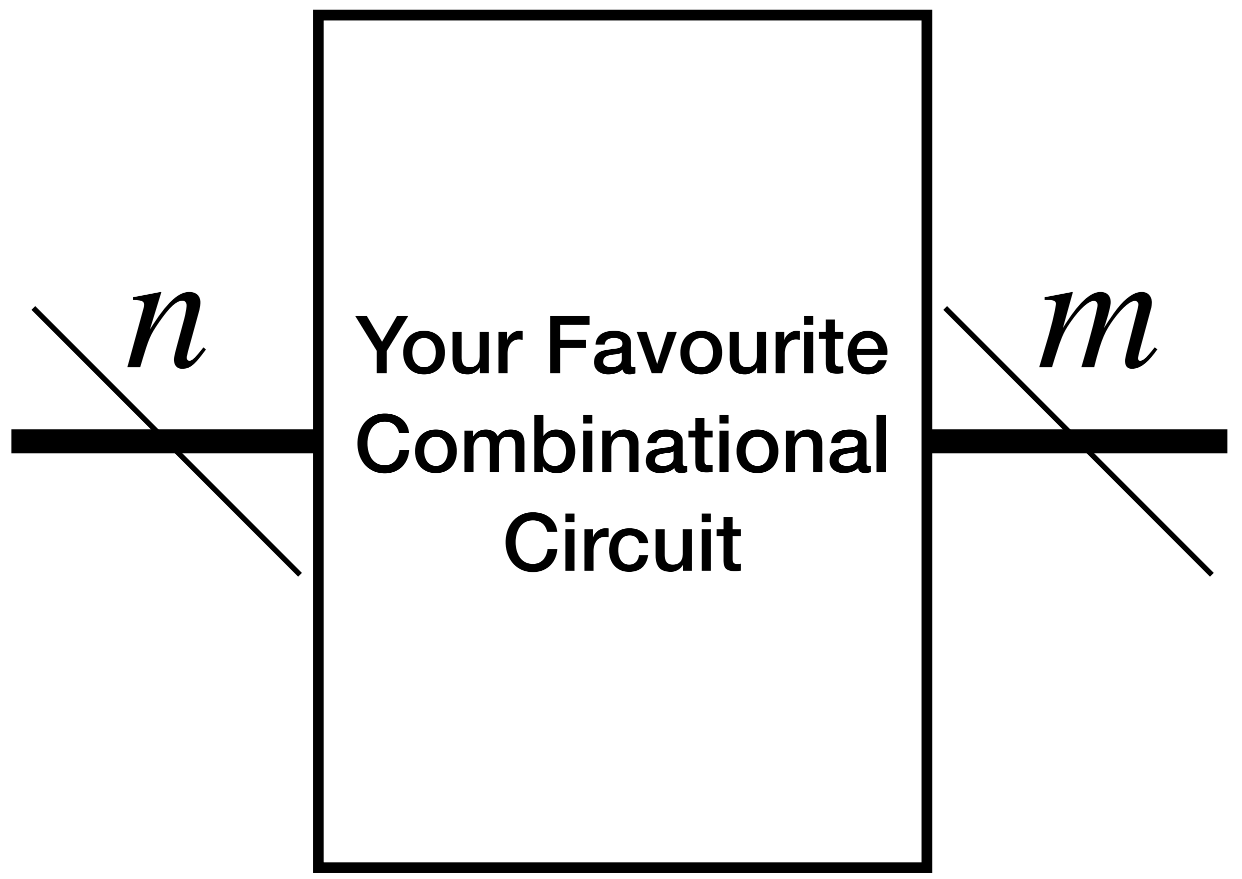


Review

P0
Review

Combinational Logic Circuit Design

- Cannot handle variable length input
- Cannot store information
- Cannot perform multi-step tasks

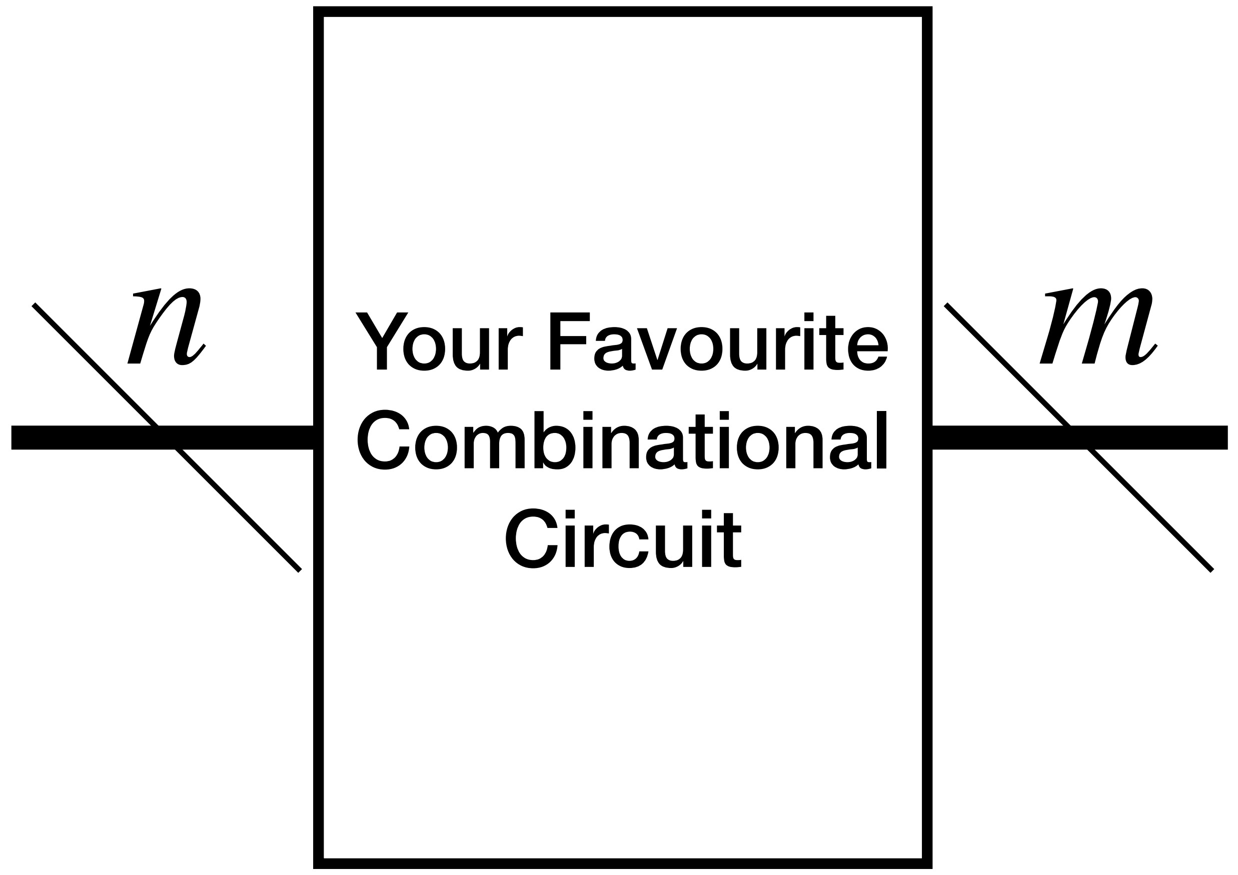


Review

Introduction to Sequential Circuit

Solution

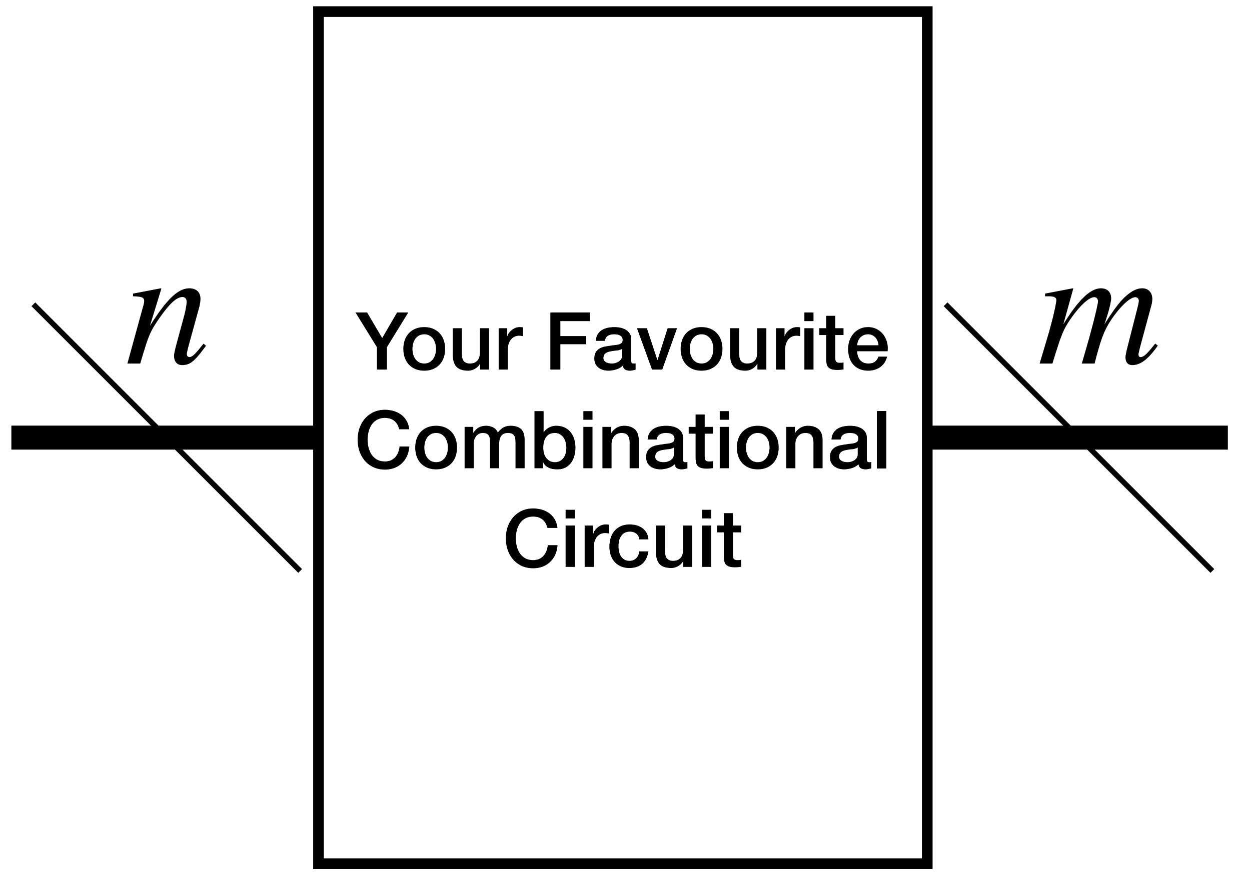
- Cannot handle variable length input
- Cannot store information
- Cannot perform multi-step tasks



P1
Introduction

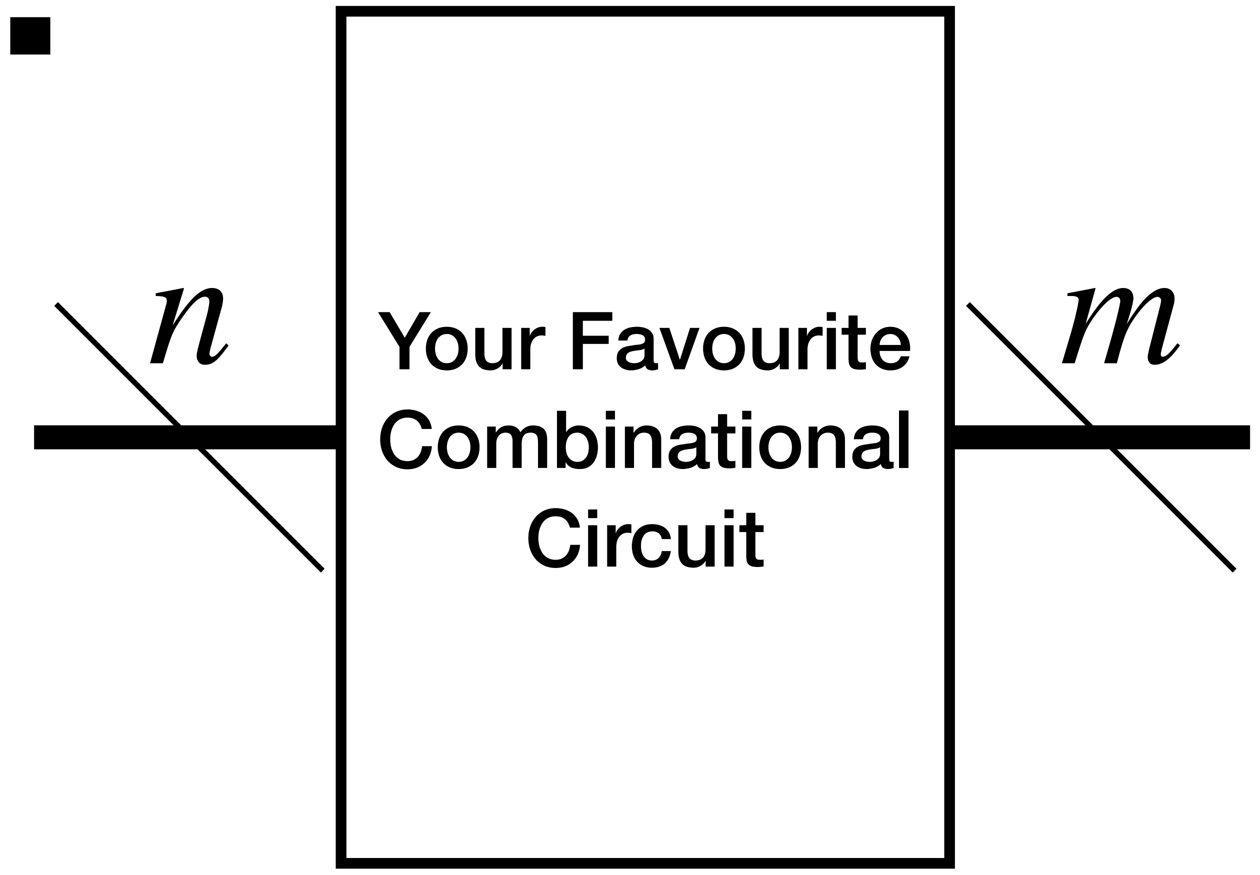
Solution: Storage!

- Cannot handle variable length input
- Cannot store information
- Cannot perform multi-step tasks



Concept

Solution: Storage!



Your Favourite Combinational Circuit

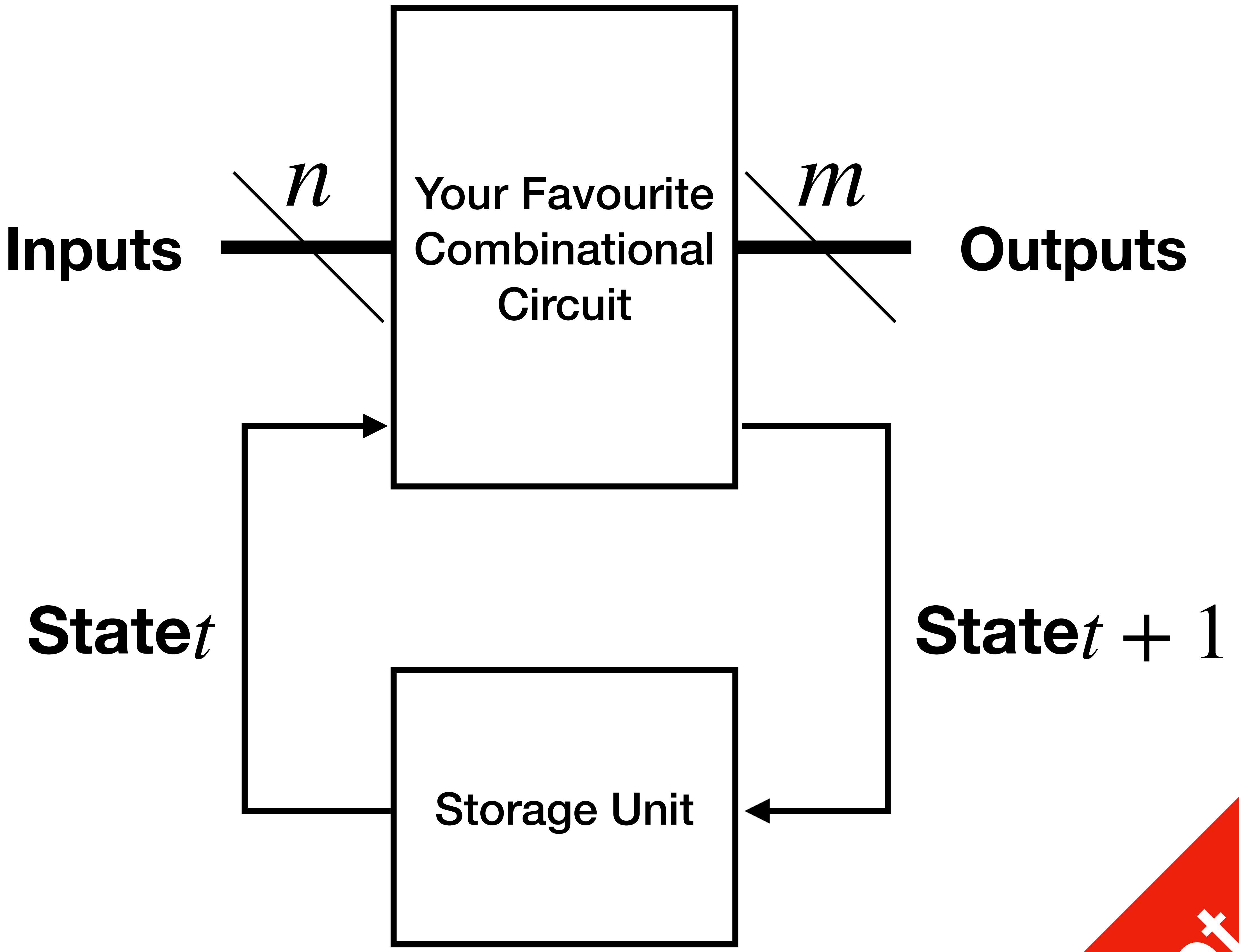
- Cannot handle variable length input
- Cannot store information
- Cannot perform multi-step tasks

- Storage of partial input
- Storage of partial results and states
- Storage of instructions

Concept

Sequential Circuits

- Handle variable-length input
- Store information
- Multi-step tasks
- **State Transition** from time t to $t + 1$



Concept

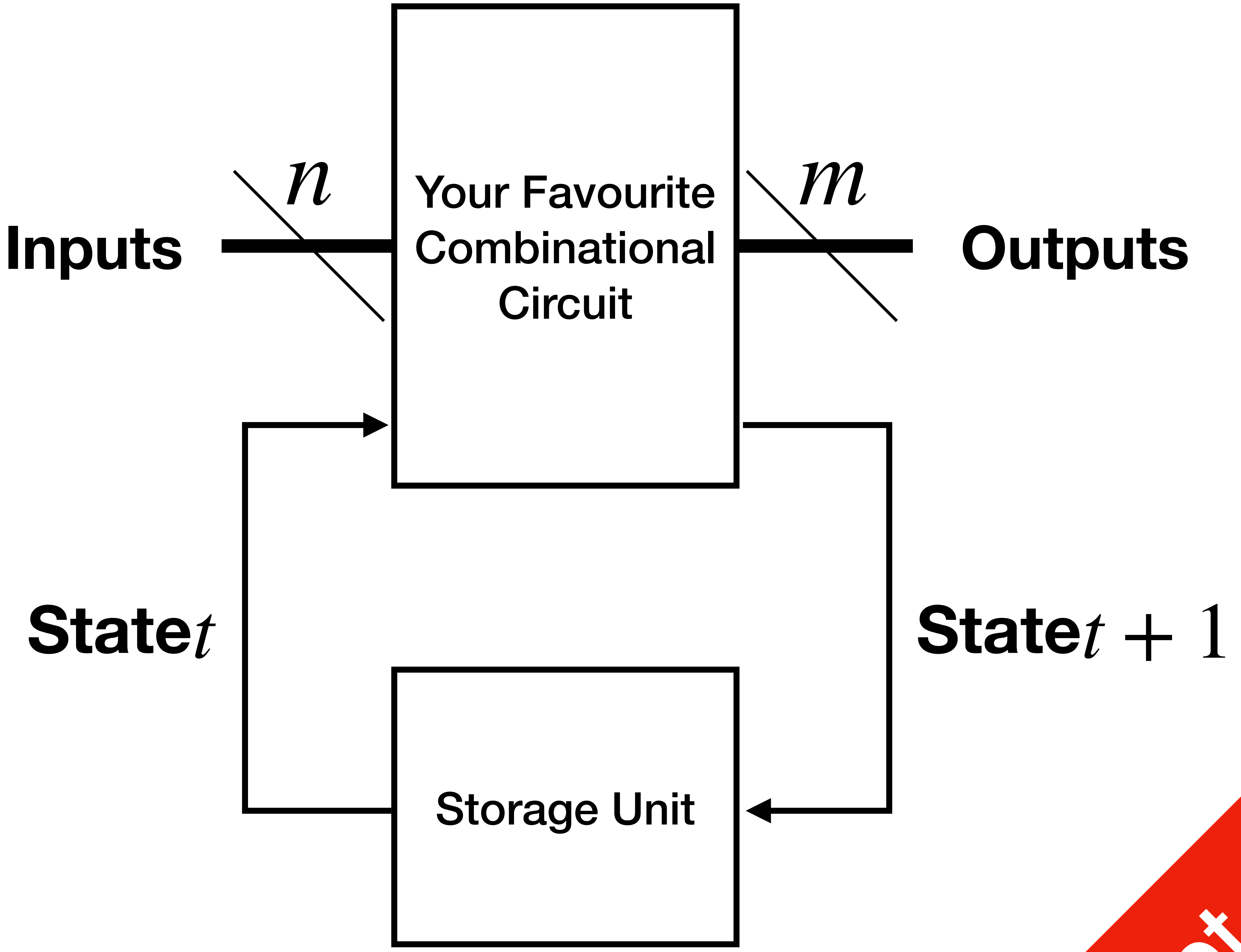
Definitions

1. **Storage Elements**
   circuits that can store binary information

2. **State**
   partial results, instructions, etc.

3. **Synchronous Sequential Circuit**
   Signals arrive at discrete instants of time, outputs at next time step

4. **Asynchronous Sequential Circuit**
   Signals arrive at any instant of time, outputs when ready



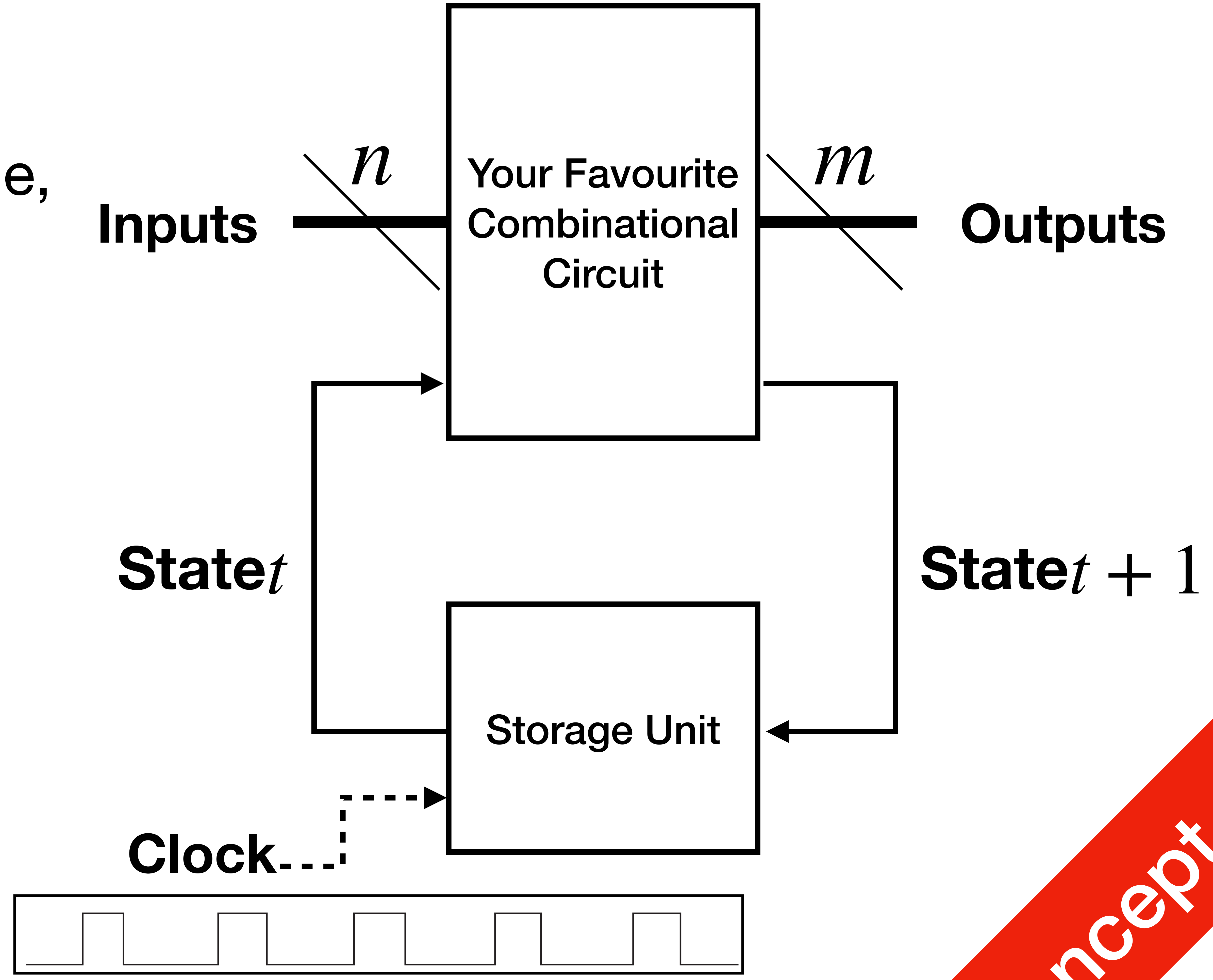
Definitions

3. **Synchronous Sequential Circuit**
   Signals arrive at discrete instants of time, outputs at next time step

- Has Clock

4. **Asynchronous Sequential Circuit**
   Signals arrive at any instant of time, outputs when ready

- May not have Clock



Question

1. Are calculators designed using **Combinational Circuits** or **Sequential Circuits**?

- What about your microwave and toaster?

- What about your digital watch (not smart)?

- What about computers/smartphones?

Question

Exercises

Question

2. Is this calculator using **Asynchronous Sequential Circuit** or **Synchronous Sequential Circuit**?

Question

2. Is this calculator using **Asynchronous Sequential Circuit** or **Synchronous Sequential Circuit**?

- What are the states for this calculator?

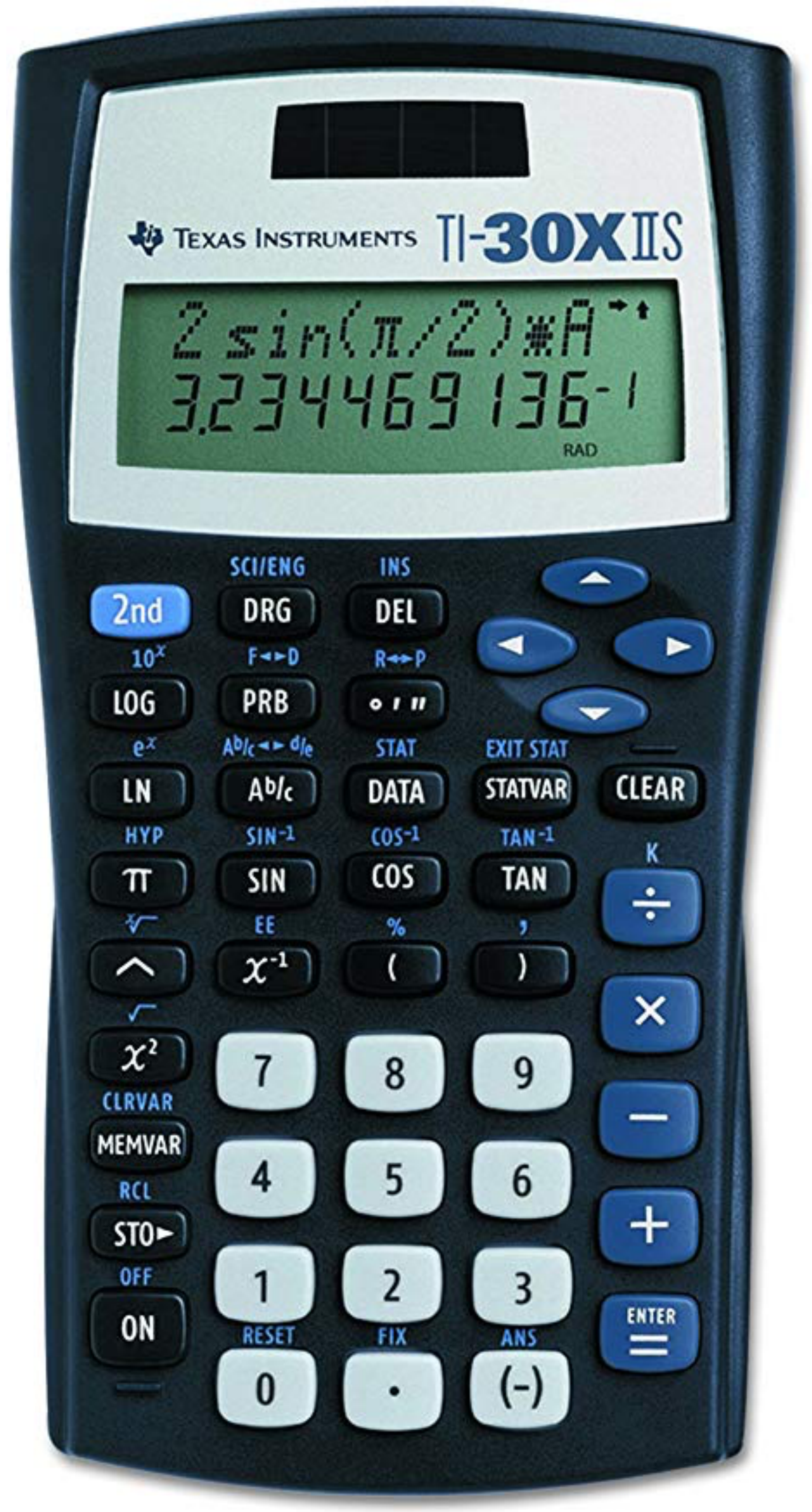
Question

2. Is this calculator using **Asynchronous Sequential Circuit** or **Synchronous Sequential Circuit**?

- What are the states for this calculator?

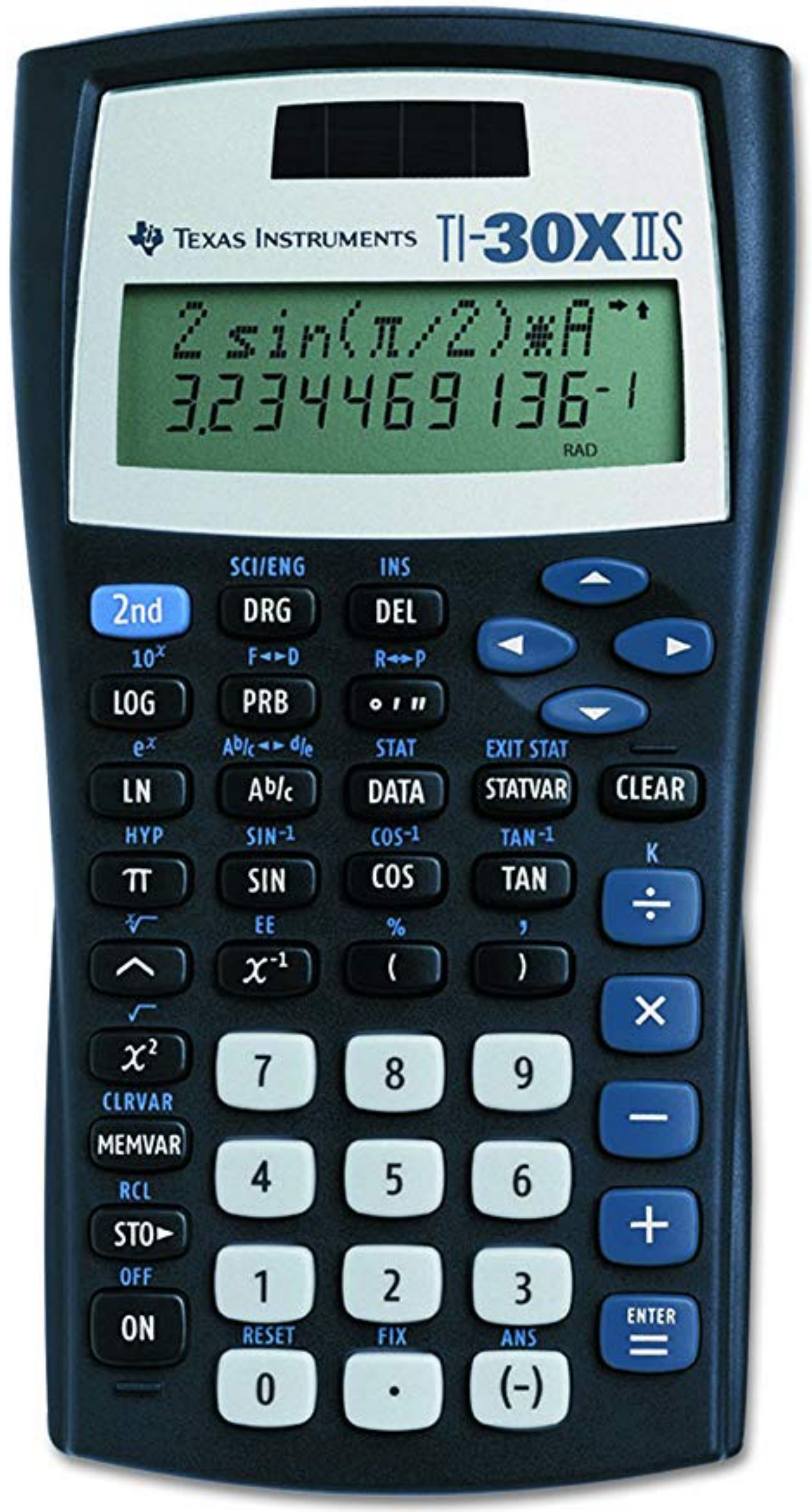
- What kind of information is stored?

Question



Question

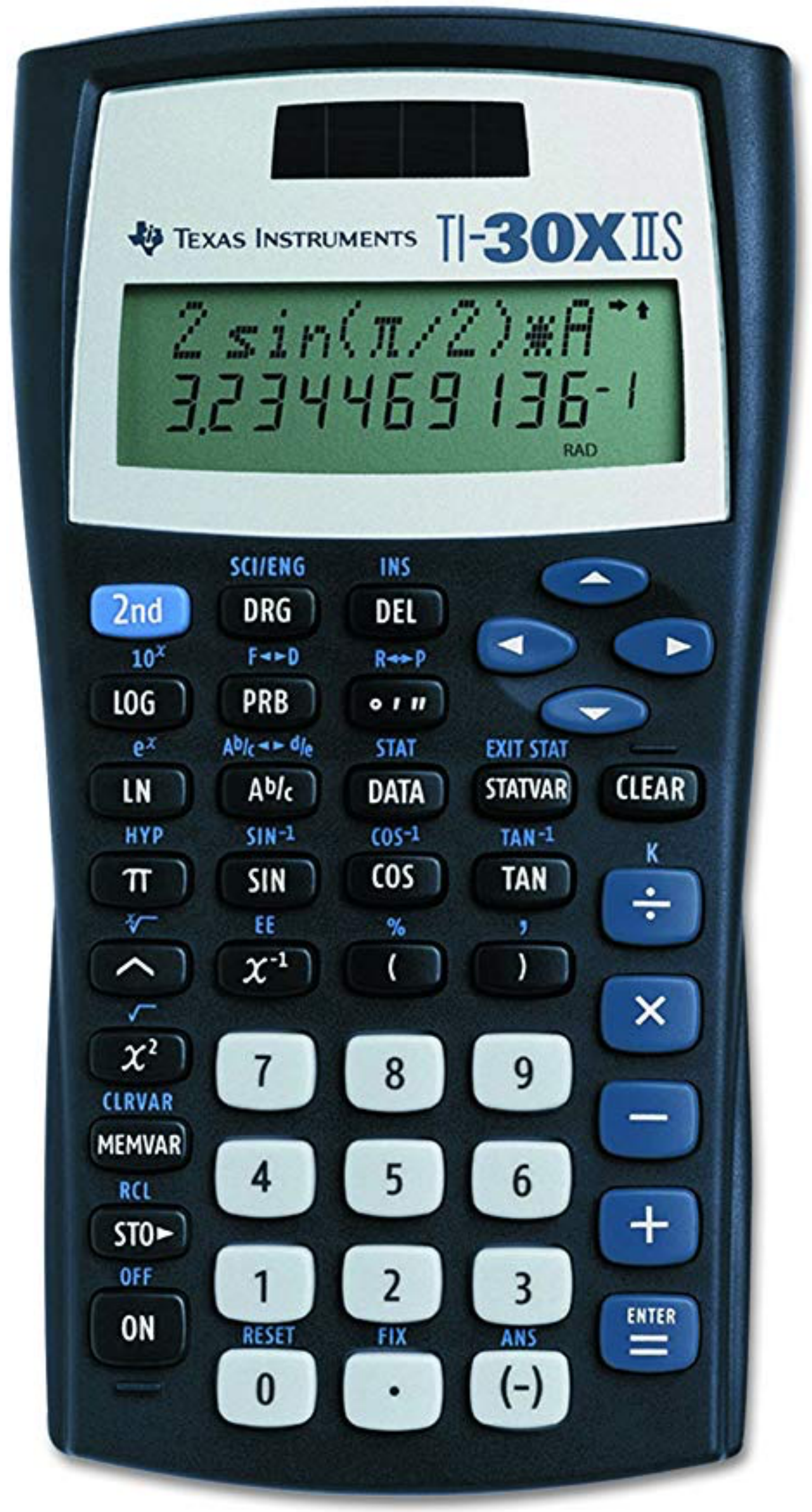
3. Is this calculator using **Asynchronous Sequential Circuit** or **Synchronous Sequential Circuit**?



Question

3. Is this calculator using **Asynchronous Sequential Circuit** or **Synchronous Sequential Circuit**?

   - What are the states for this calculator?

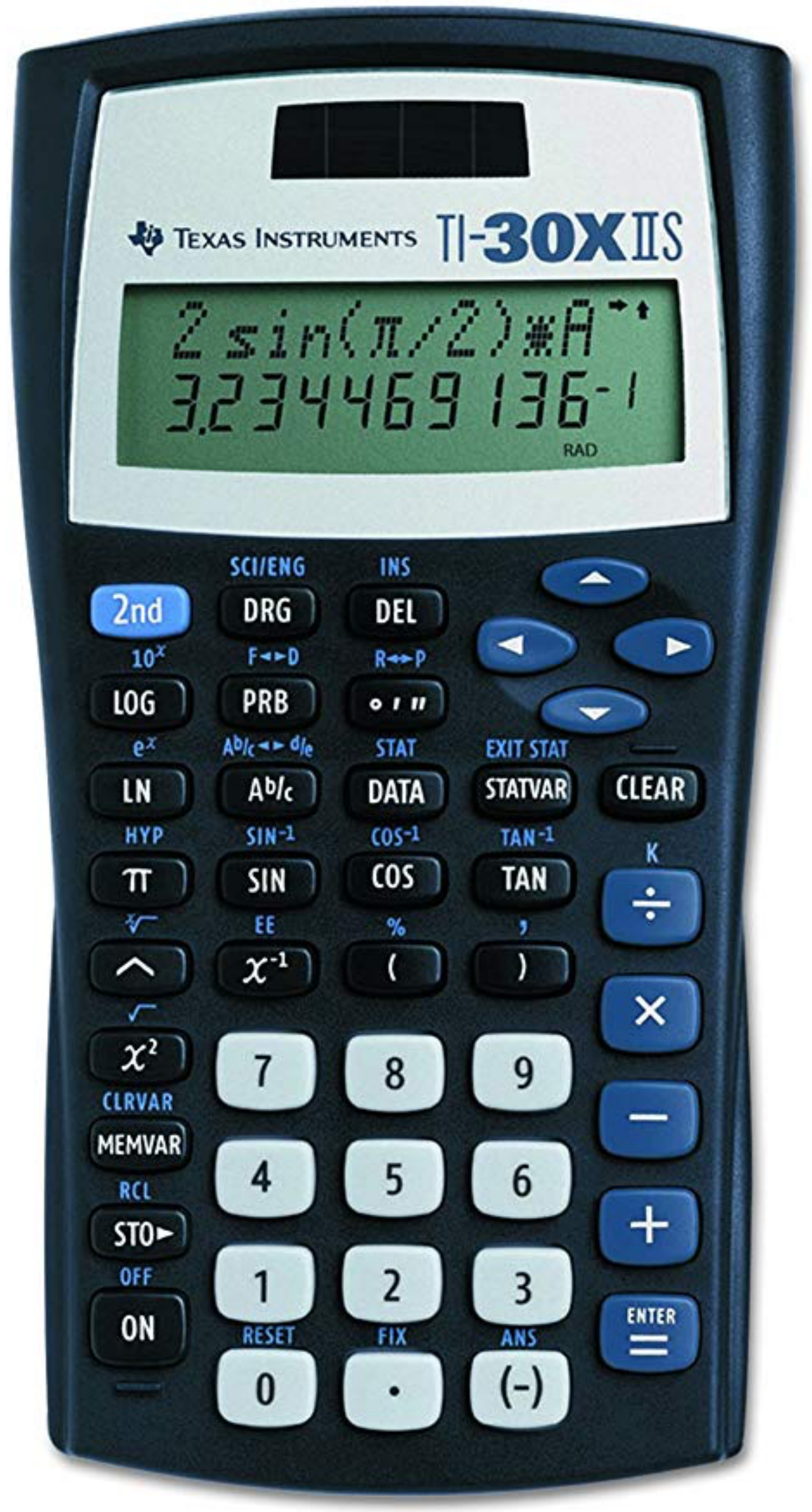


Question

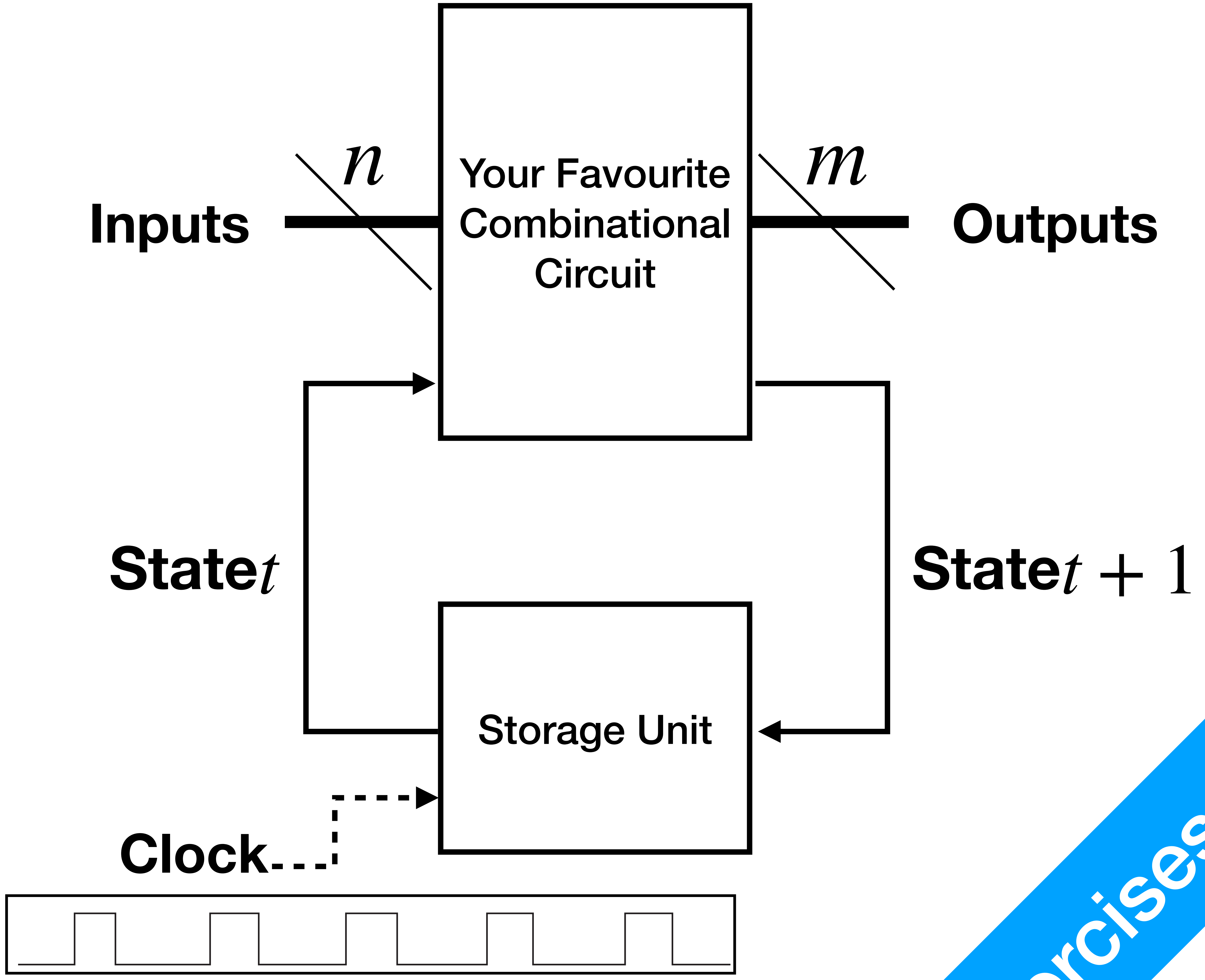
3. Is this calculator using **Asynchronous Sequential Circuit** or **Synchronous Sequential Circuit**?

- What are the states for this calculator?

- What kind of information is stored?

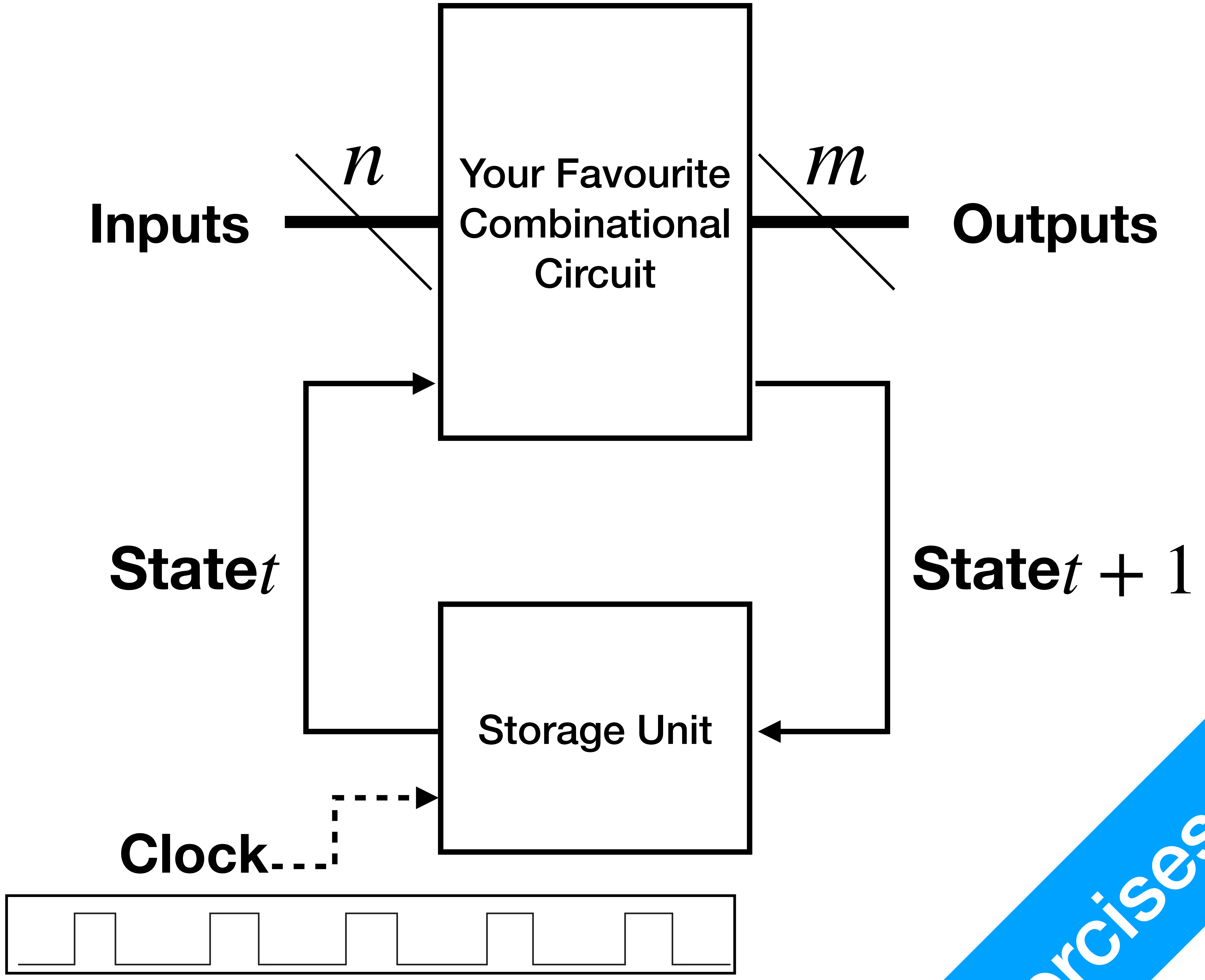


Question


Inputs
n
Your Favourite Combinational Circuit
m
Outputs
State$t$
State$t+1$
Storage Unit
Clock

Question

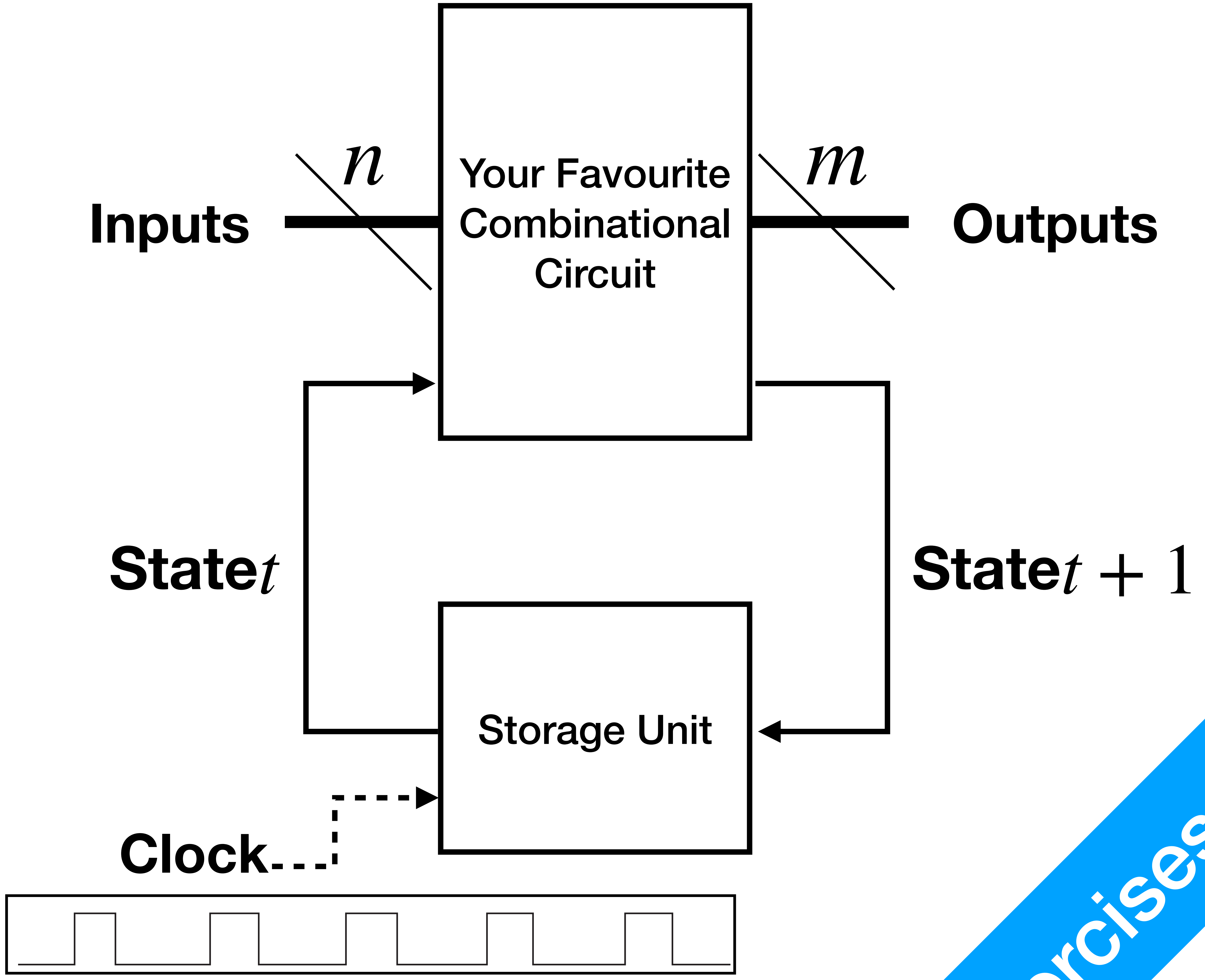
4. Is your laptop/PC/smartphone using **Asynchronous Sequential Circuit** or **Synchronous Sequential Circuit**?



Question

4. Is your laptop/PC/smartphone using **Asynchronous Sequential Circuit** or **Synchronous Sequential Circuit**?

- What are the Input/Output devices of these computers?

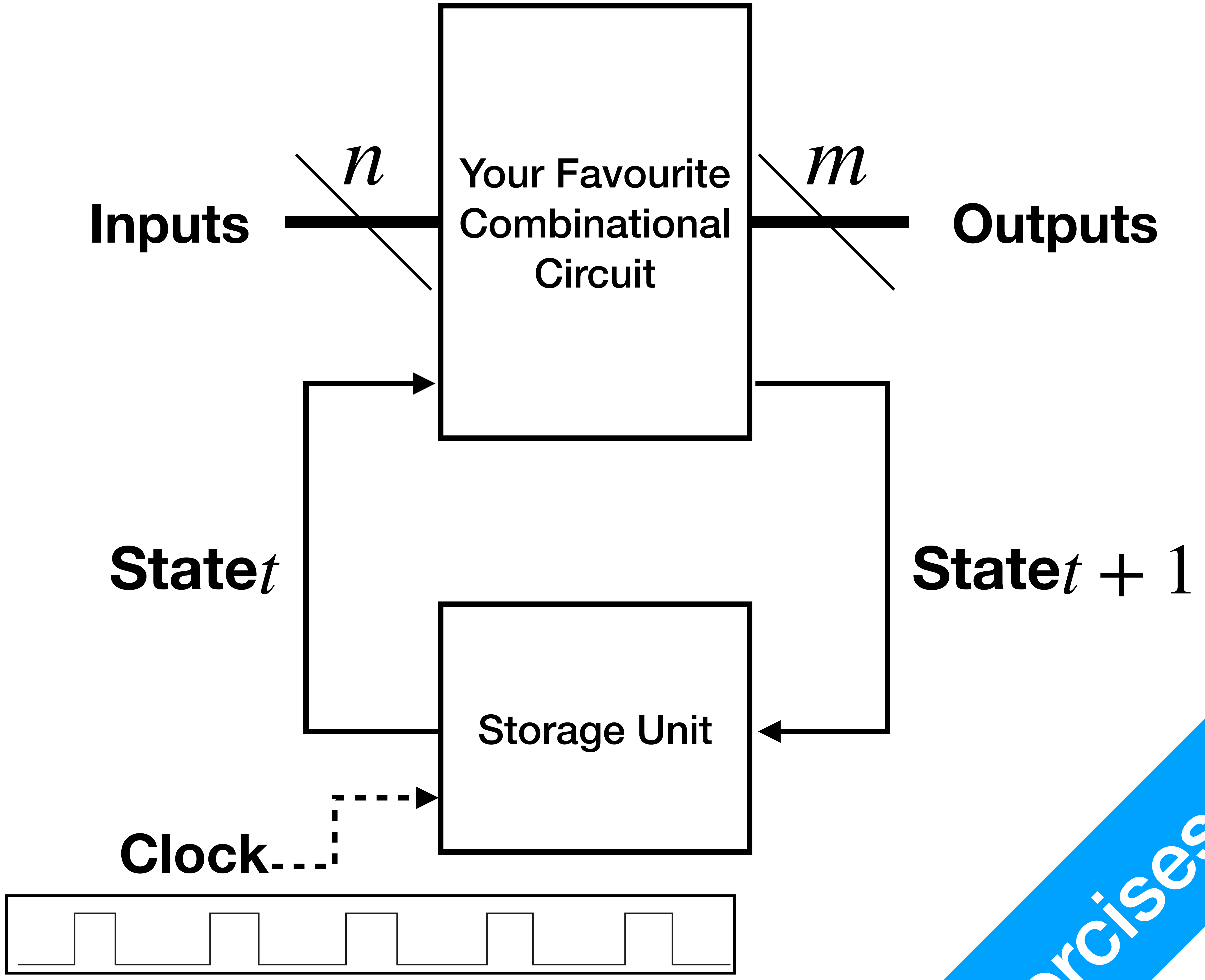


Question

4. Is your laptop/PC/smartphone using **Asynchronous Sequential Circuit** or **Synchronous Sequential Circuit**?

- What are the Input/Output devices of these computers?

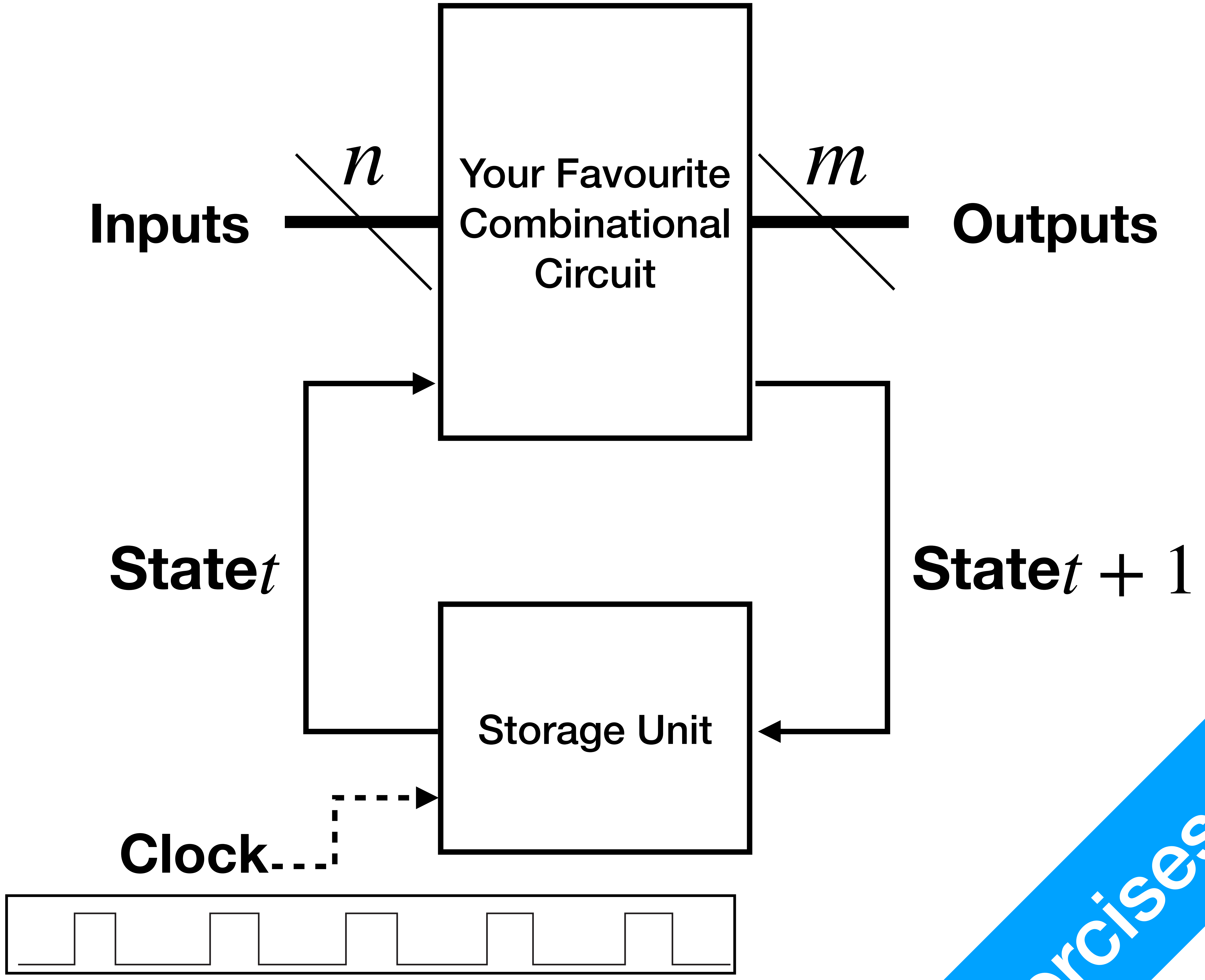
- What are the storage devices?



Question

4. Is your laptop/PC/smartphone using **Asynchronous Sequential Circuit** or **Synchronous Sequential Circuit**?

- What are the Input/Output devices of these computers?
- What are the storage devices?
- What about CPU?



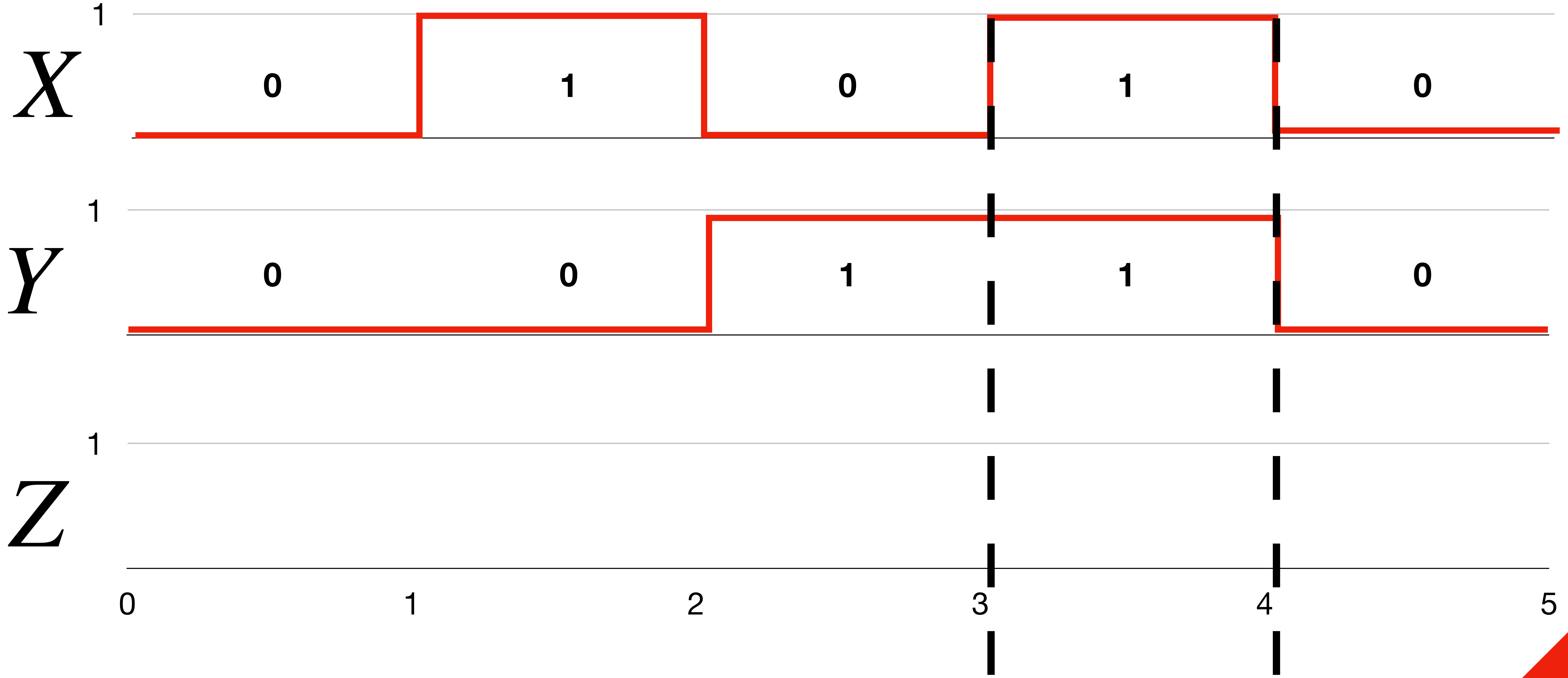
Latches

SR and $\overline{SR}$ Latches, D Latch

Summary

Stability

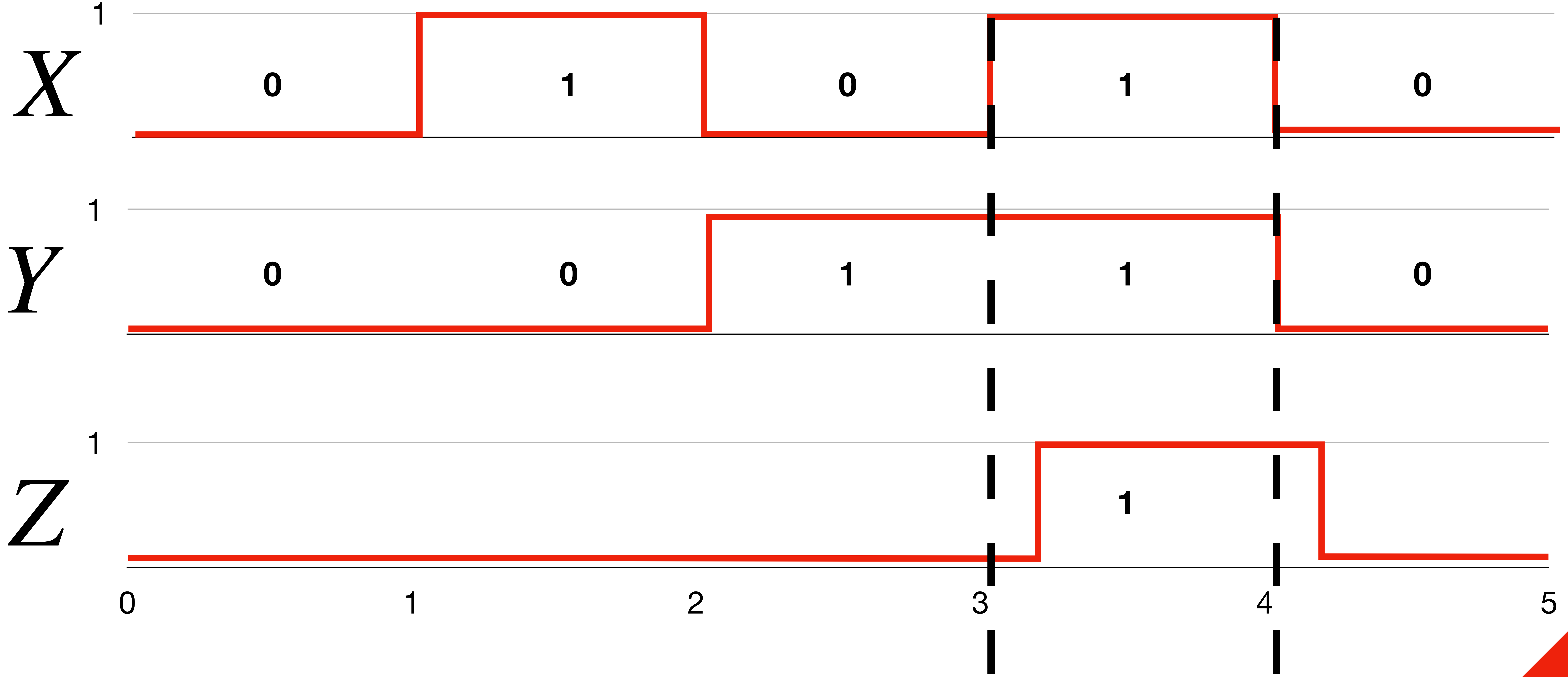
- Stable State: the values in a circuit after brief changing due to delay in passing information, reaches a state where it doesn't change anymore



P2
Latches

Stability

- Stable State: the values in a circuit after brief changing due to delay in passing information, reaches a state where it doesn't change anymore

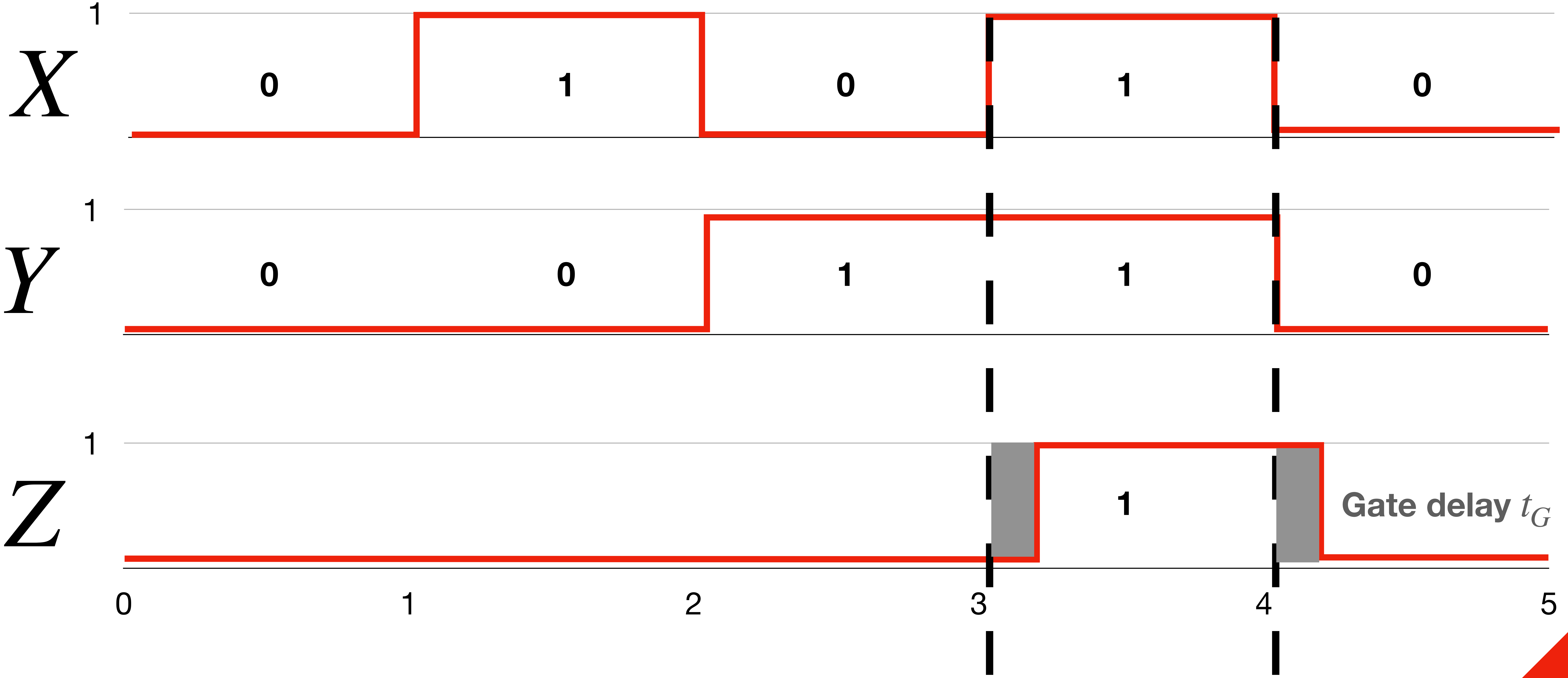


Concept

P2
Latches

Stability

- Stable State: the values in a circuit after brief changing due to delay in passing information, reaches a state where it doesn't change anymore

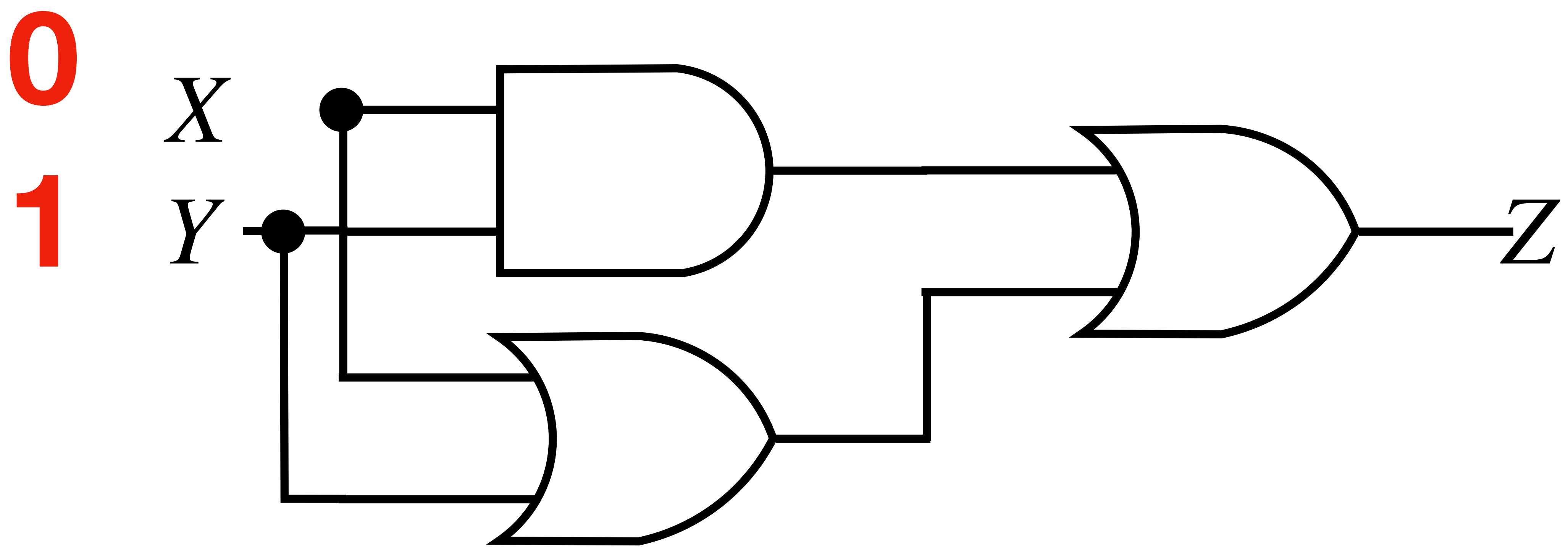


Concept

P2
Latches

Stability

- Stable State: all values in a circuit after some changes due to delay in passing signals, reach a state where they don't anymore

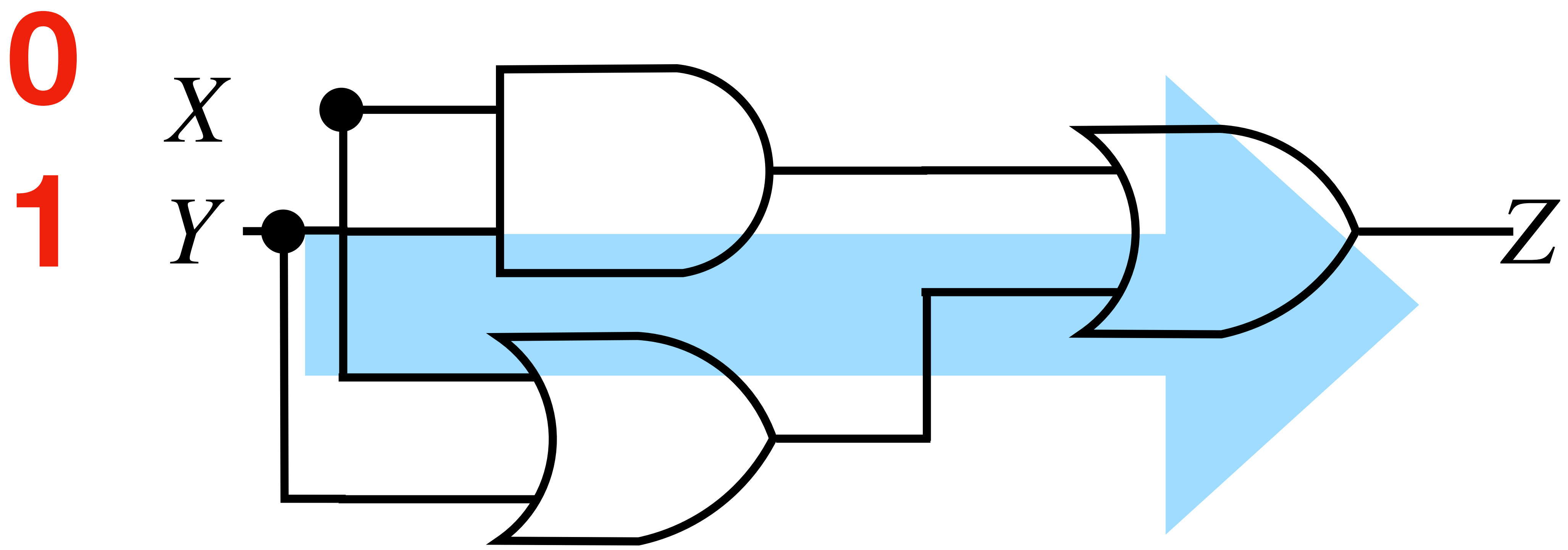


Concept

P2
Latches

Stability

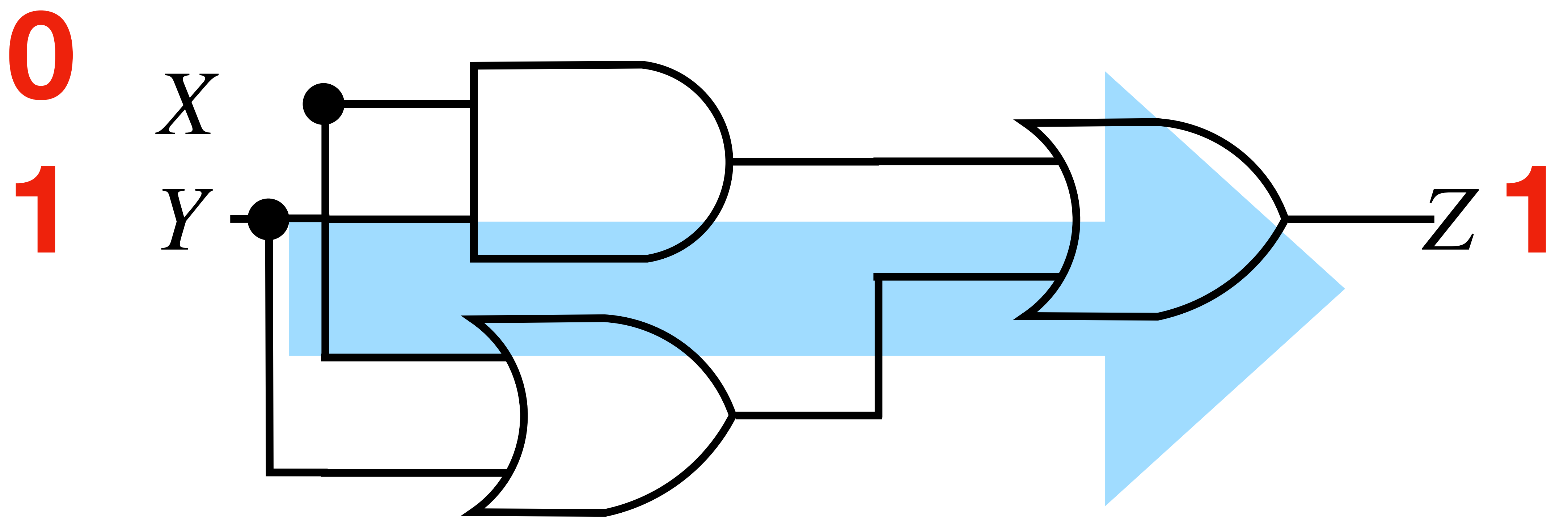
- Stable State: all values in a circuit after some changes due to delay in passing signals, reach a state where they don't anymore



Concept

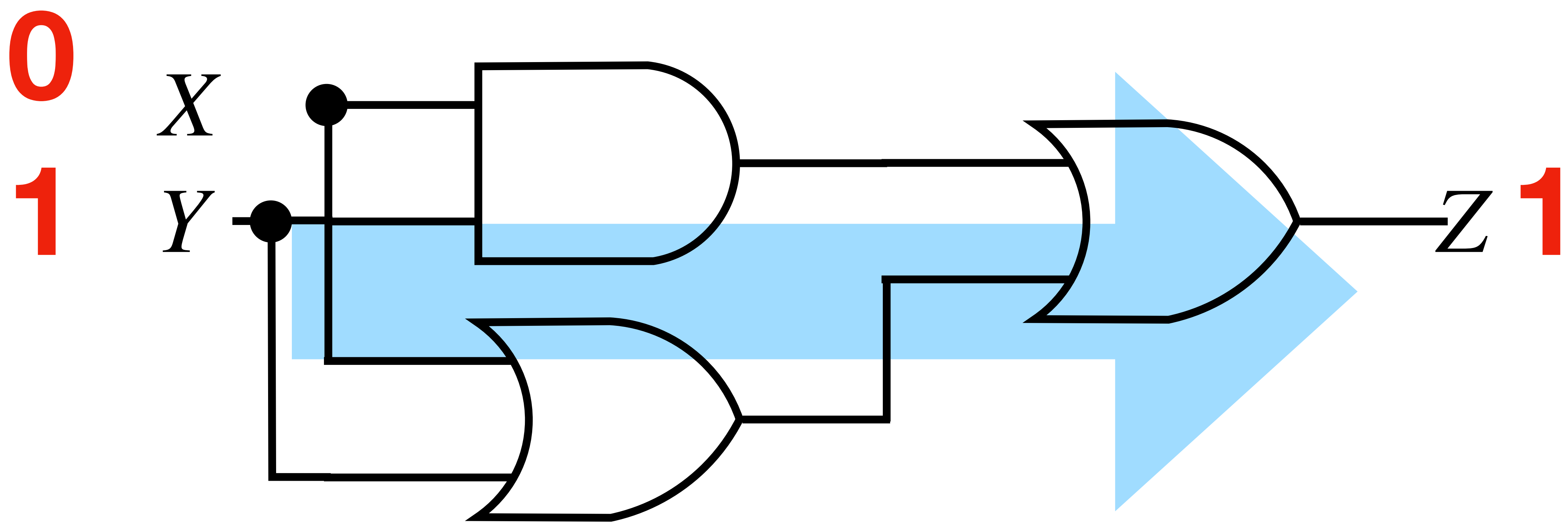
Stability

- Stable State: all values in a circuit after some changes due to delay in passing signals, reach a state where they don't anymore



Stability

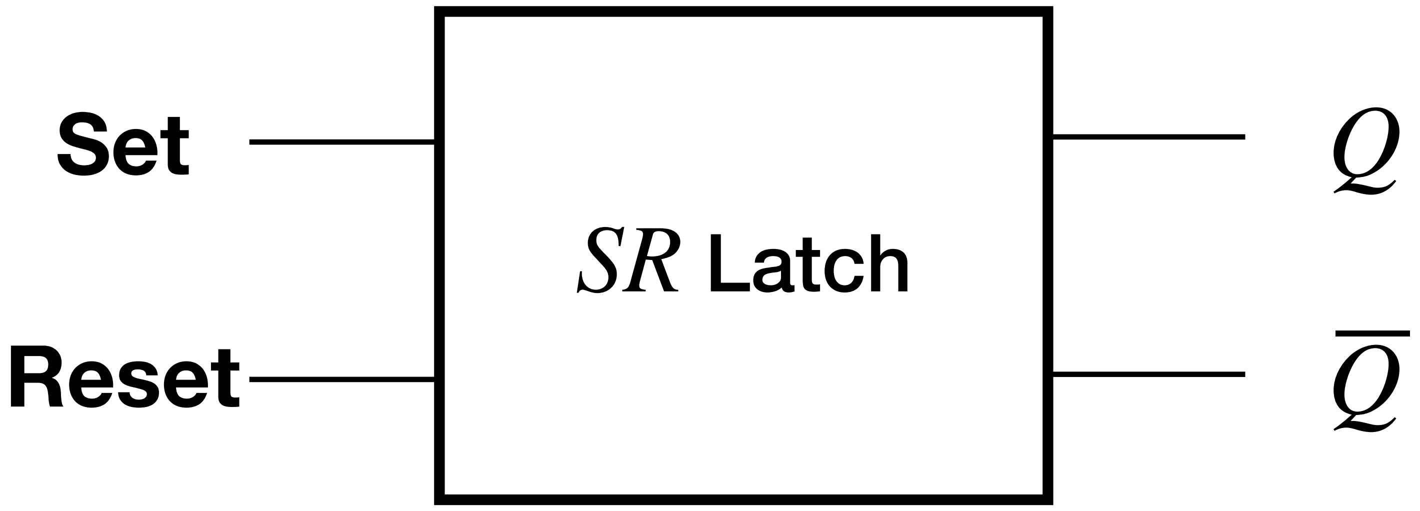
- Stable State: all values in a circuit after some changes due to delay in passing signals, reach a state where they don't anymore



- What other scenarios might bring about these instabilities?

Latches

- Basic Storage Elements
  - Maintain a binary state indefinitely, as long as there's power
  - The binary state inside can be changed

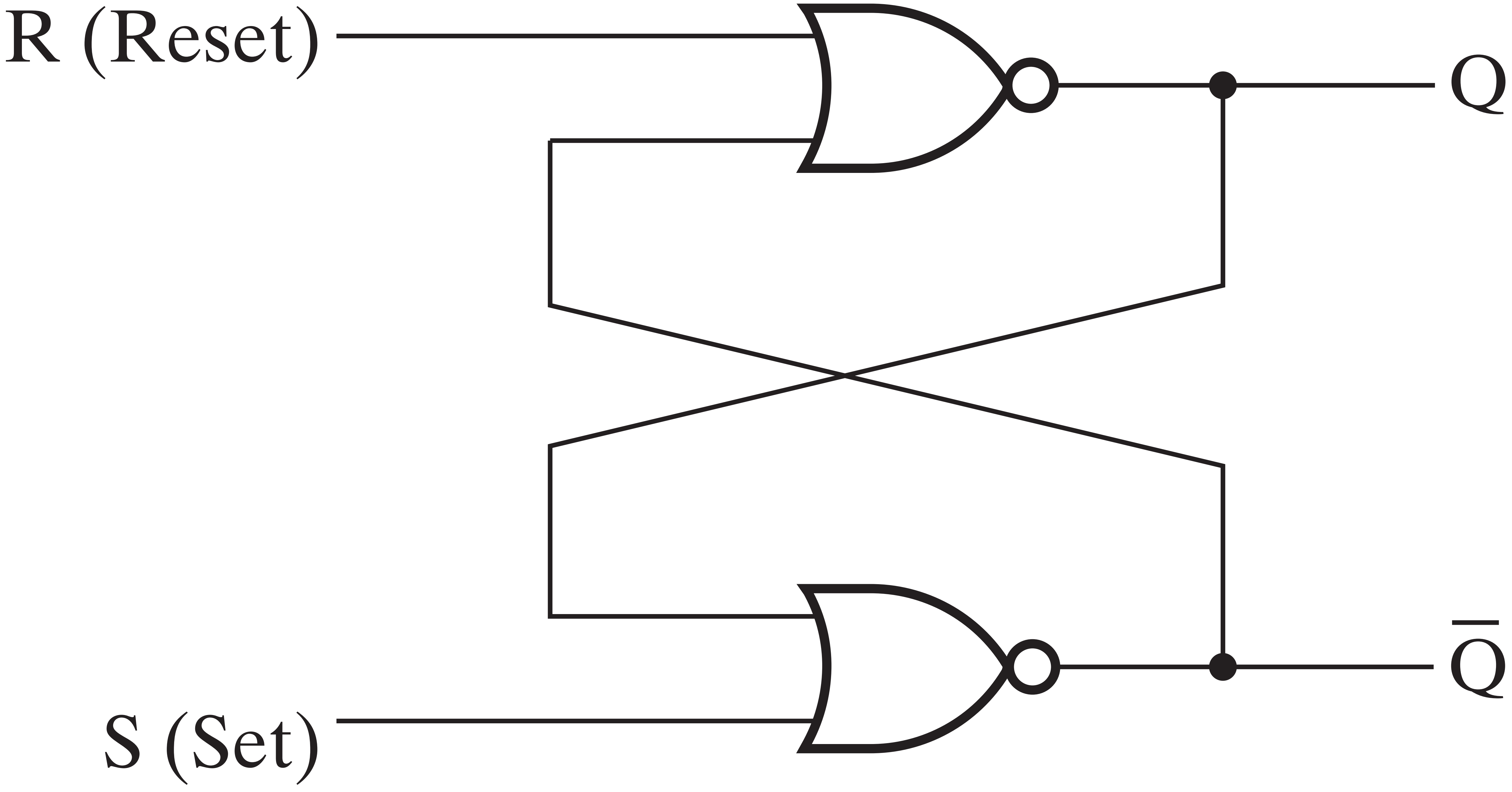


Summary

- SR Latches and $\overline{SR}$ Latches
- D Latches

Review

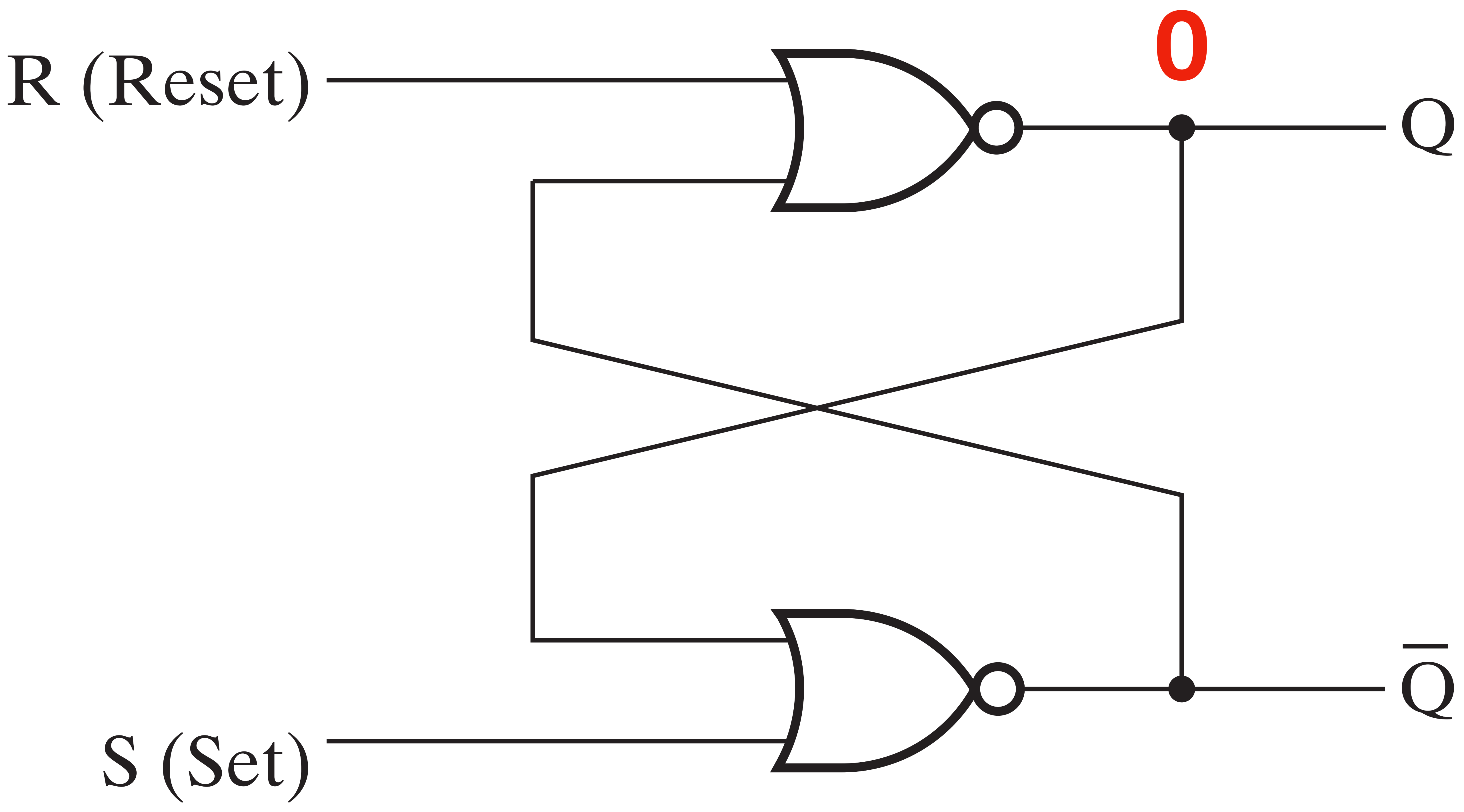
SR Latch: Stored value = 0



Demo

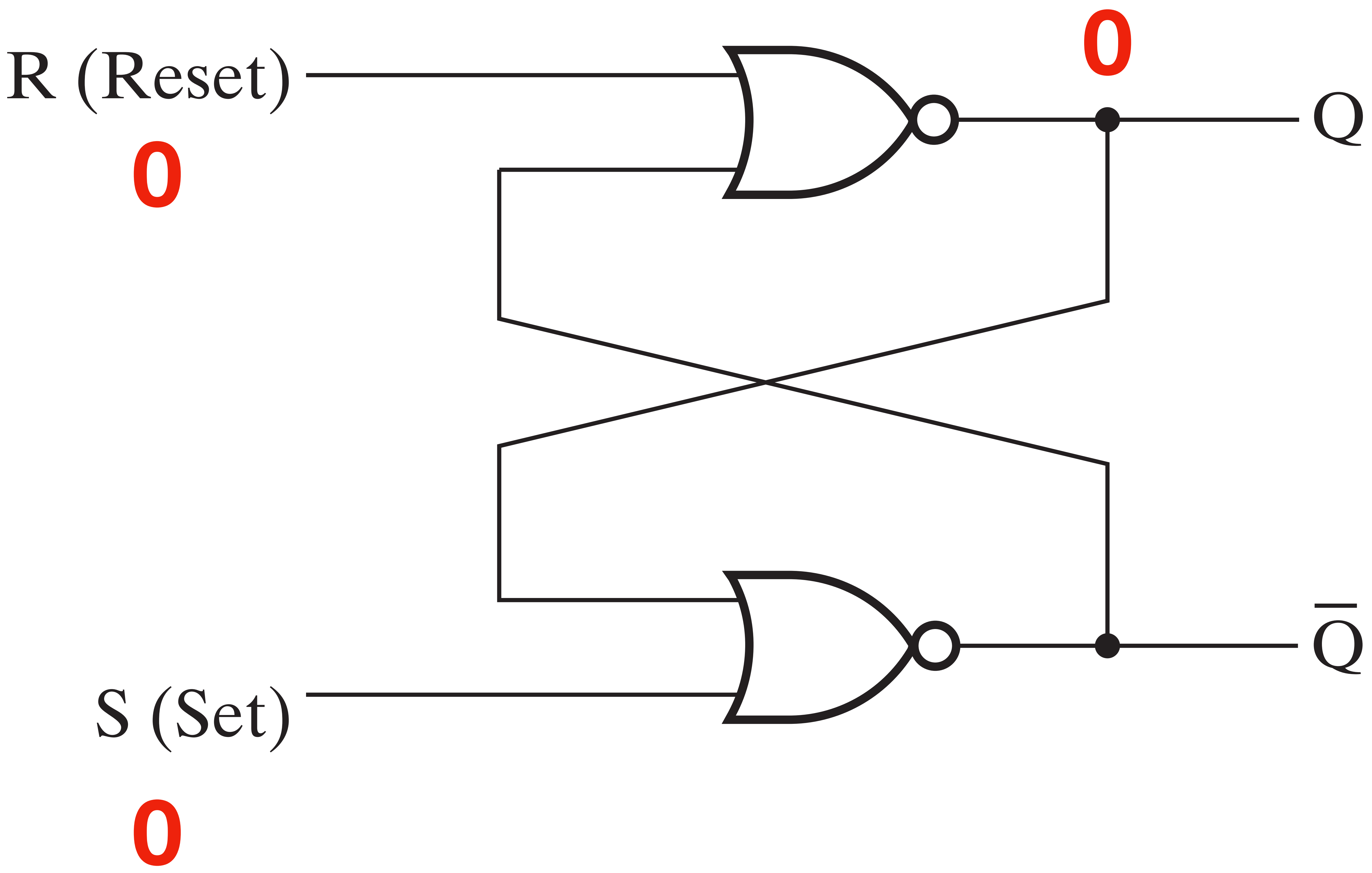
P2
Latches

SR Latch: Stored value = 0



Demo

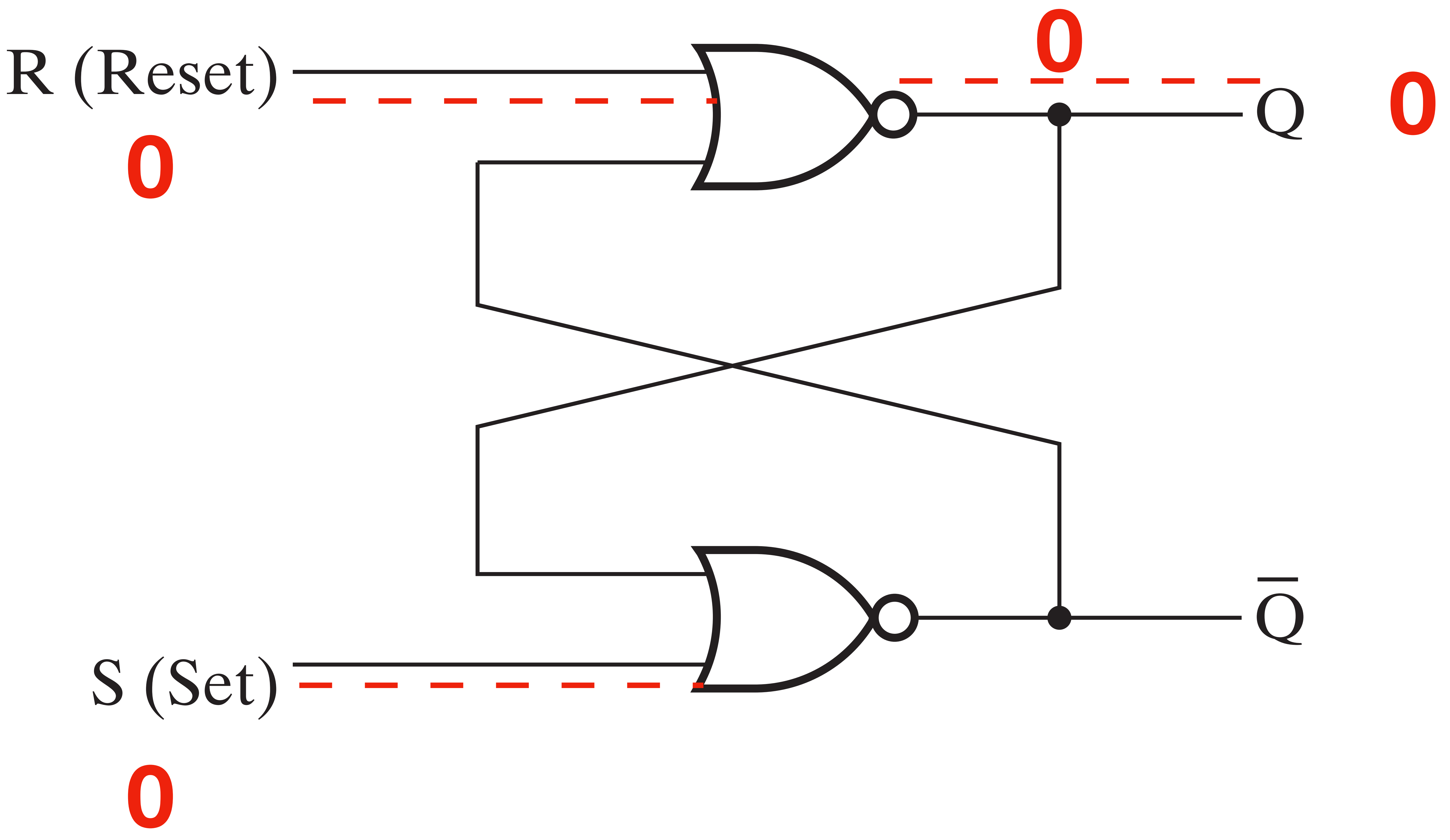
SR Latch: Stored value = 0



Demo

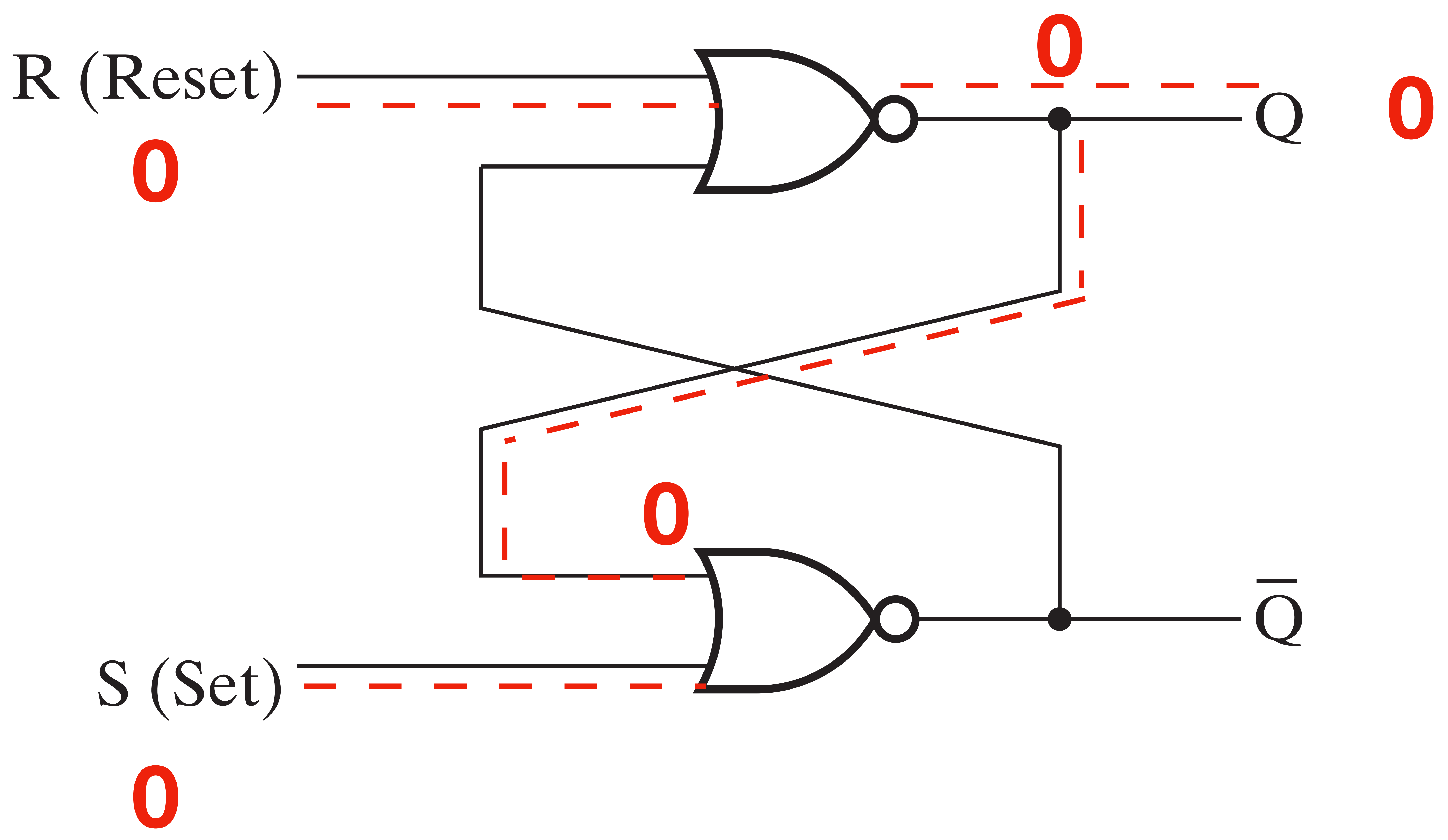
P2
Latches

SR Latch: Stored value = 0

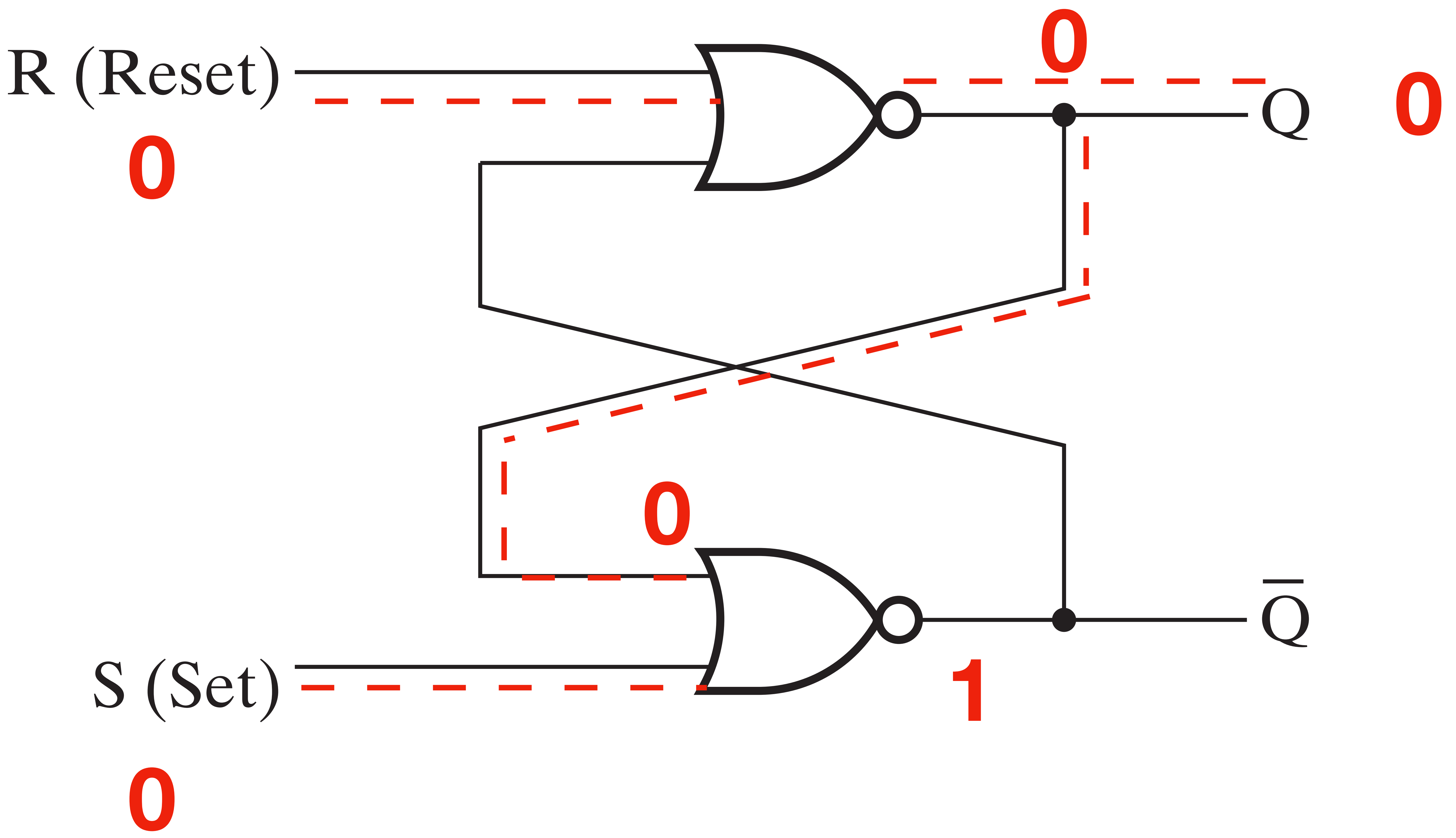


Demo

SR Latch: Stored value = 0

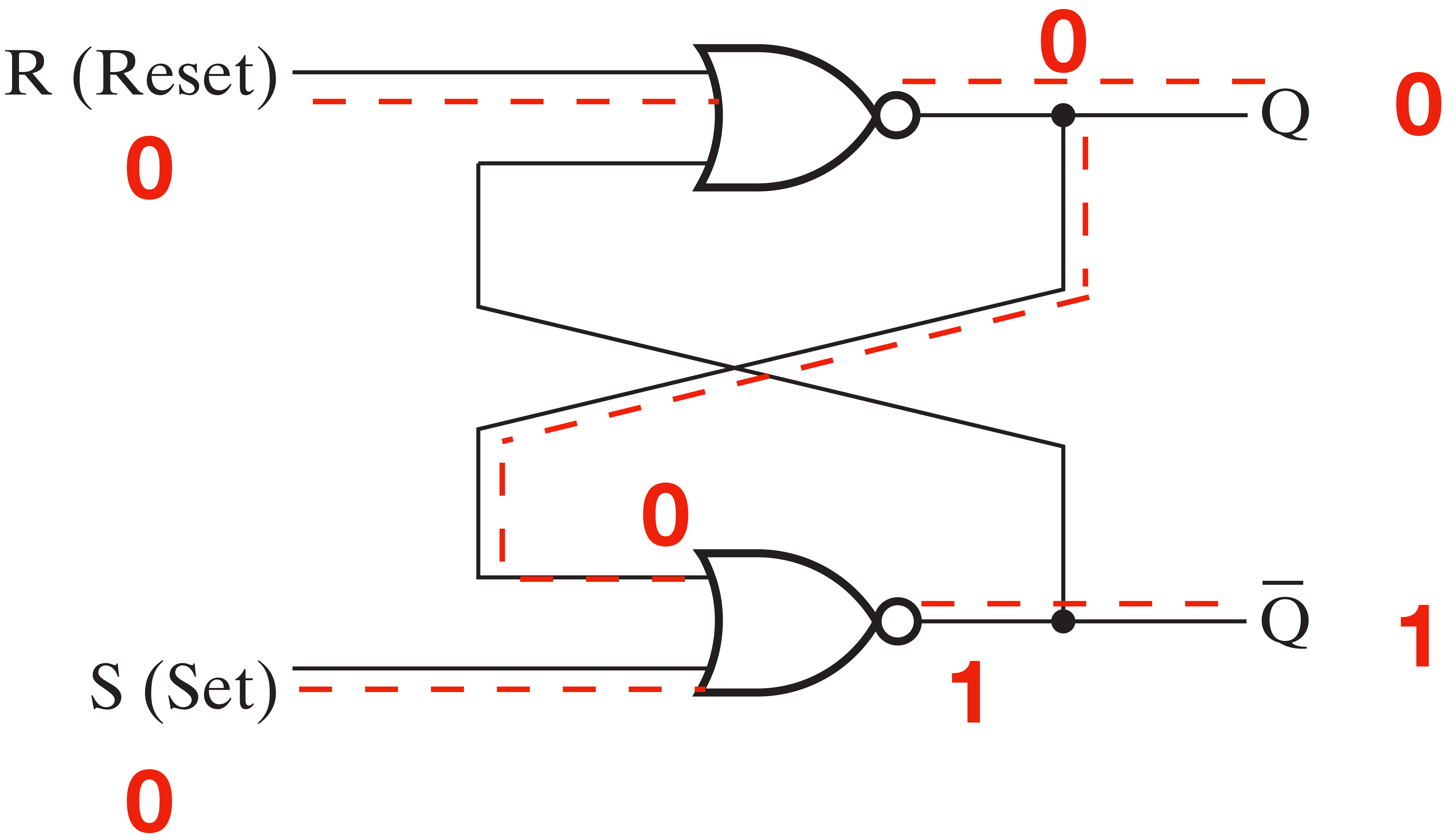


SR Latch: Stored value = 0



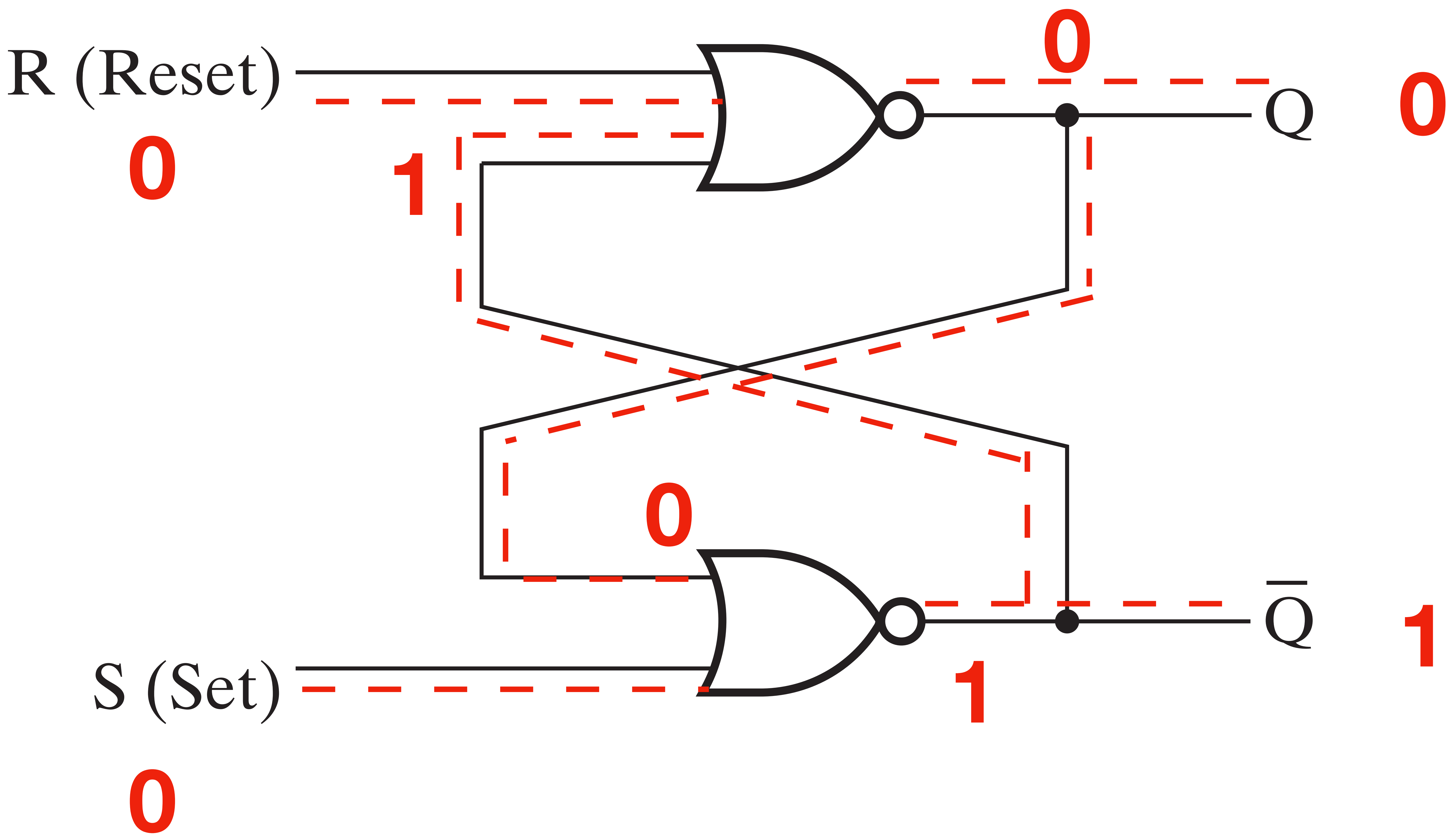
P2
Latches

SR Latch: Stored value = 0



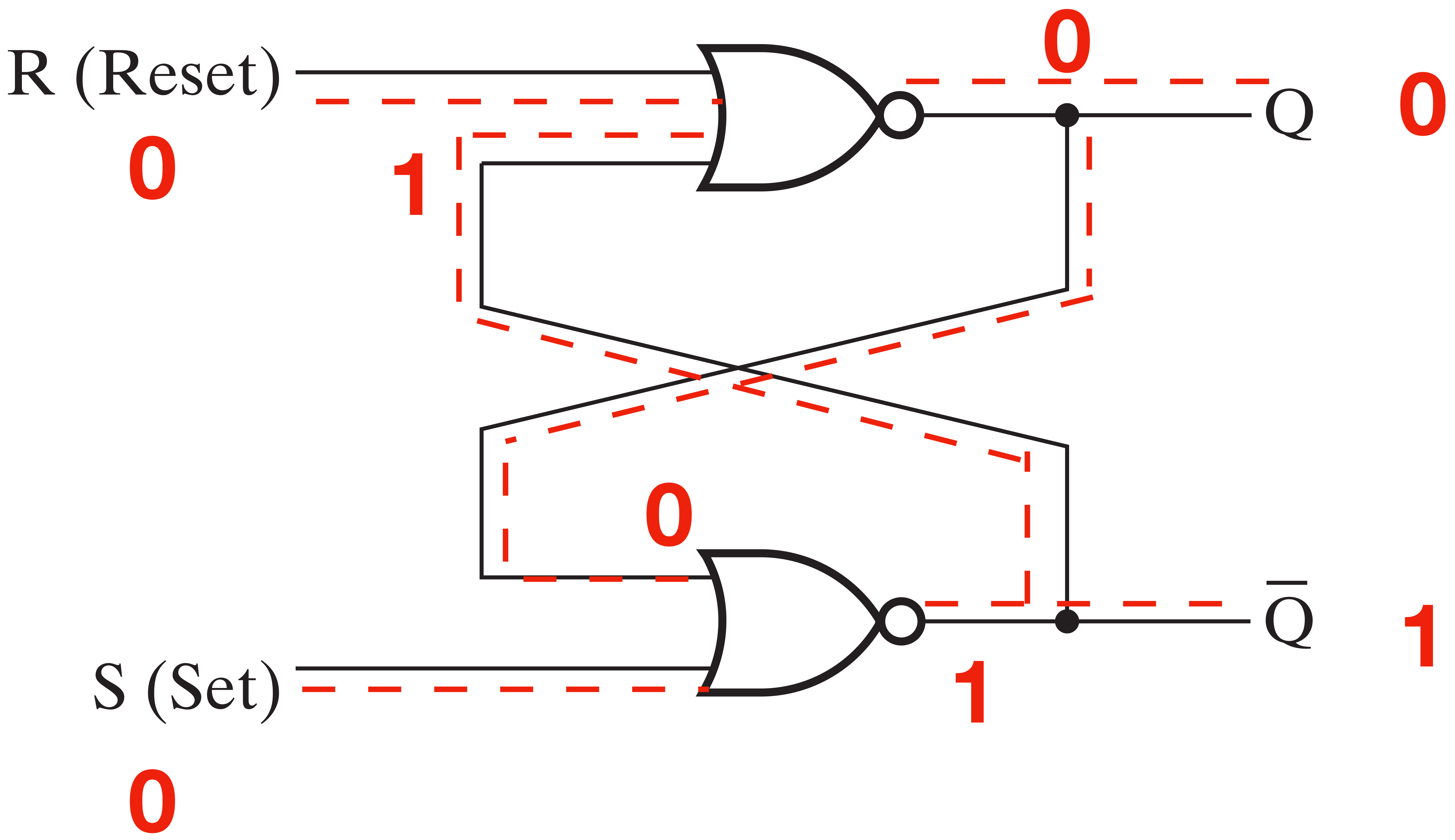
Demo

SR Latch: Stored value = 0



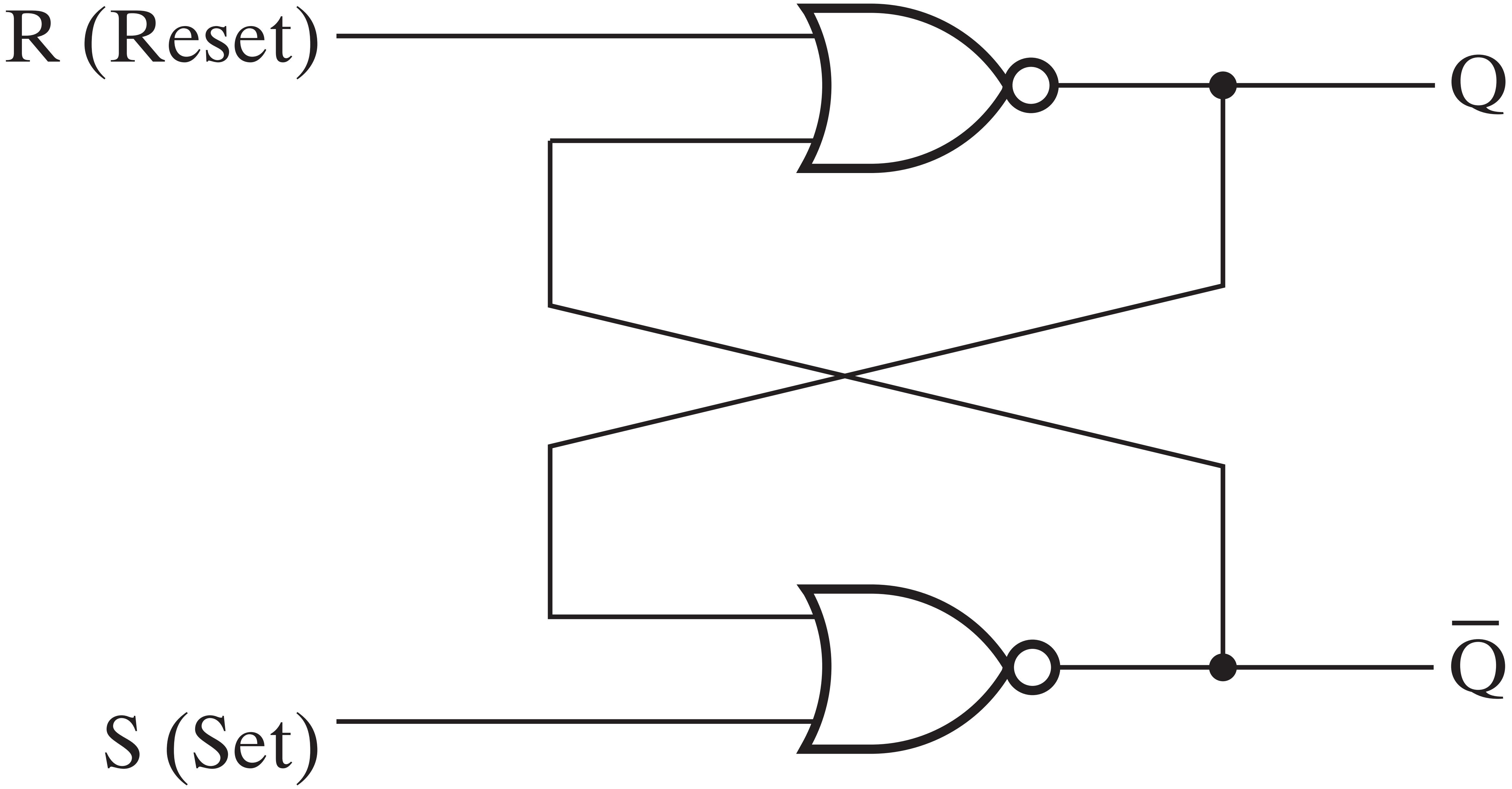
P2
Latches

SR Latch: Stored value = 0



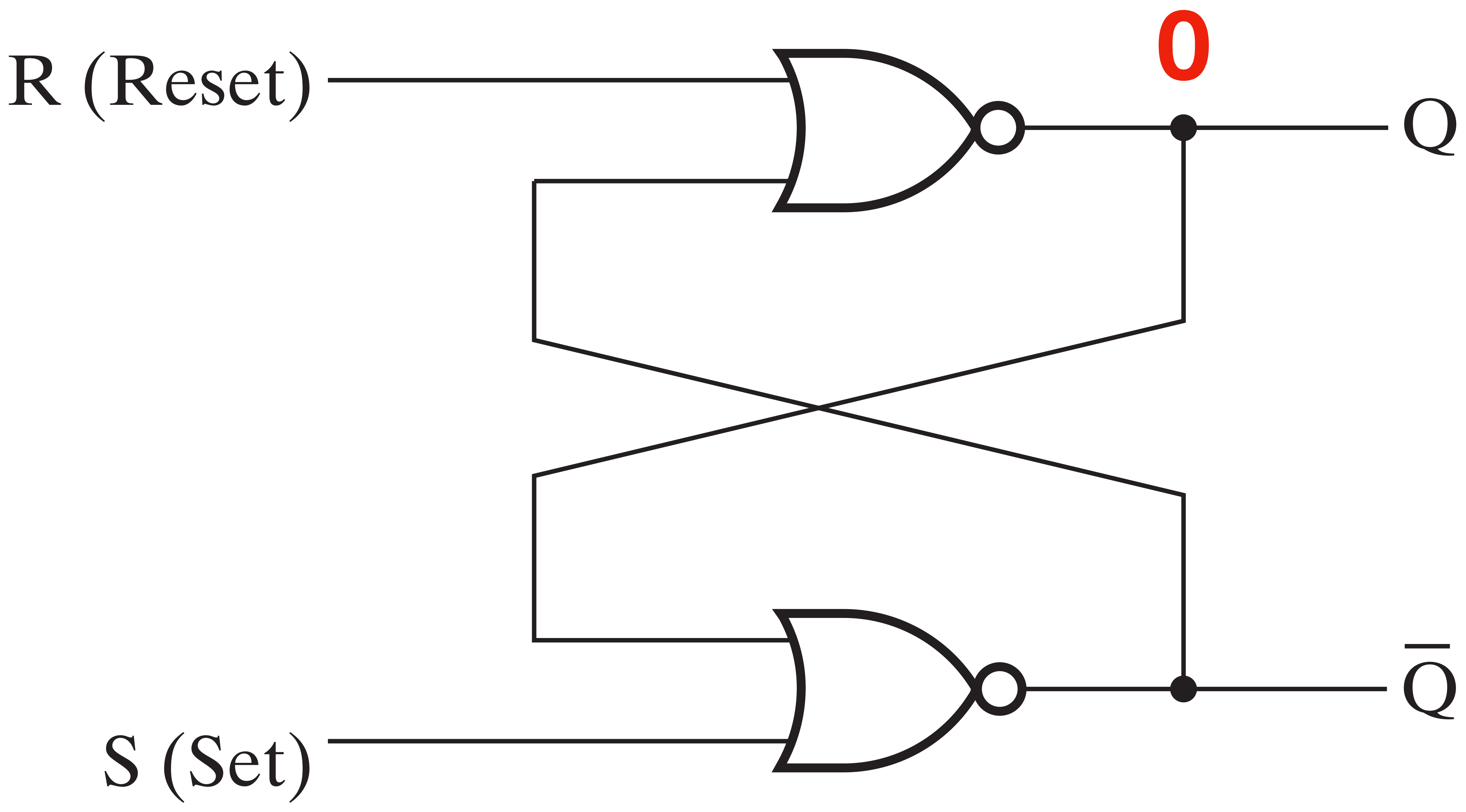
Demo

SR Latch: Stored value = 0



P2
Latches

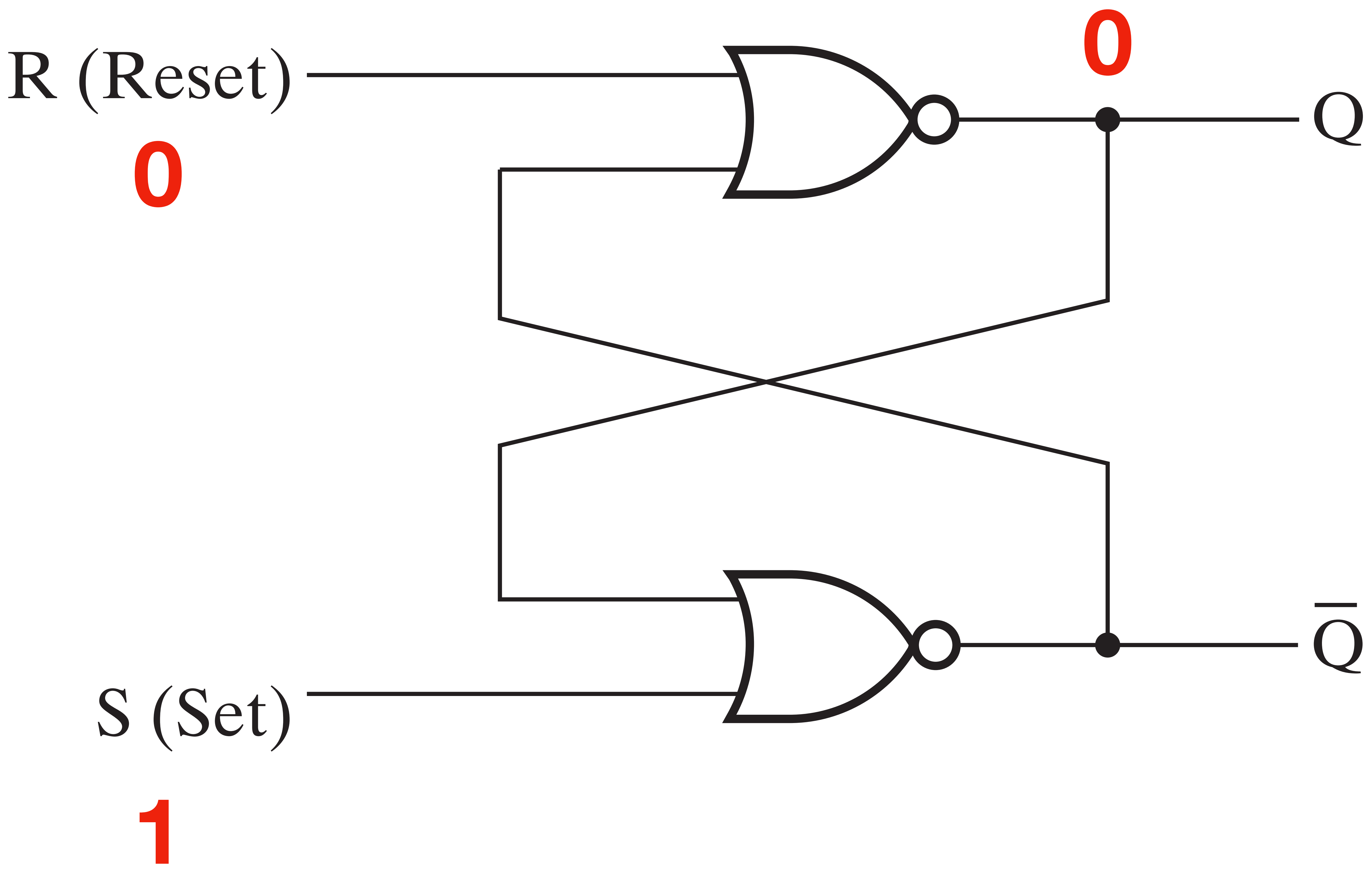
SR Latch: Stored value = 0



Demo

P2
Latches

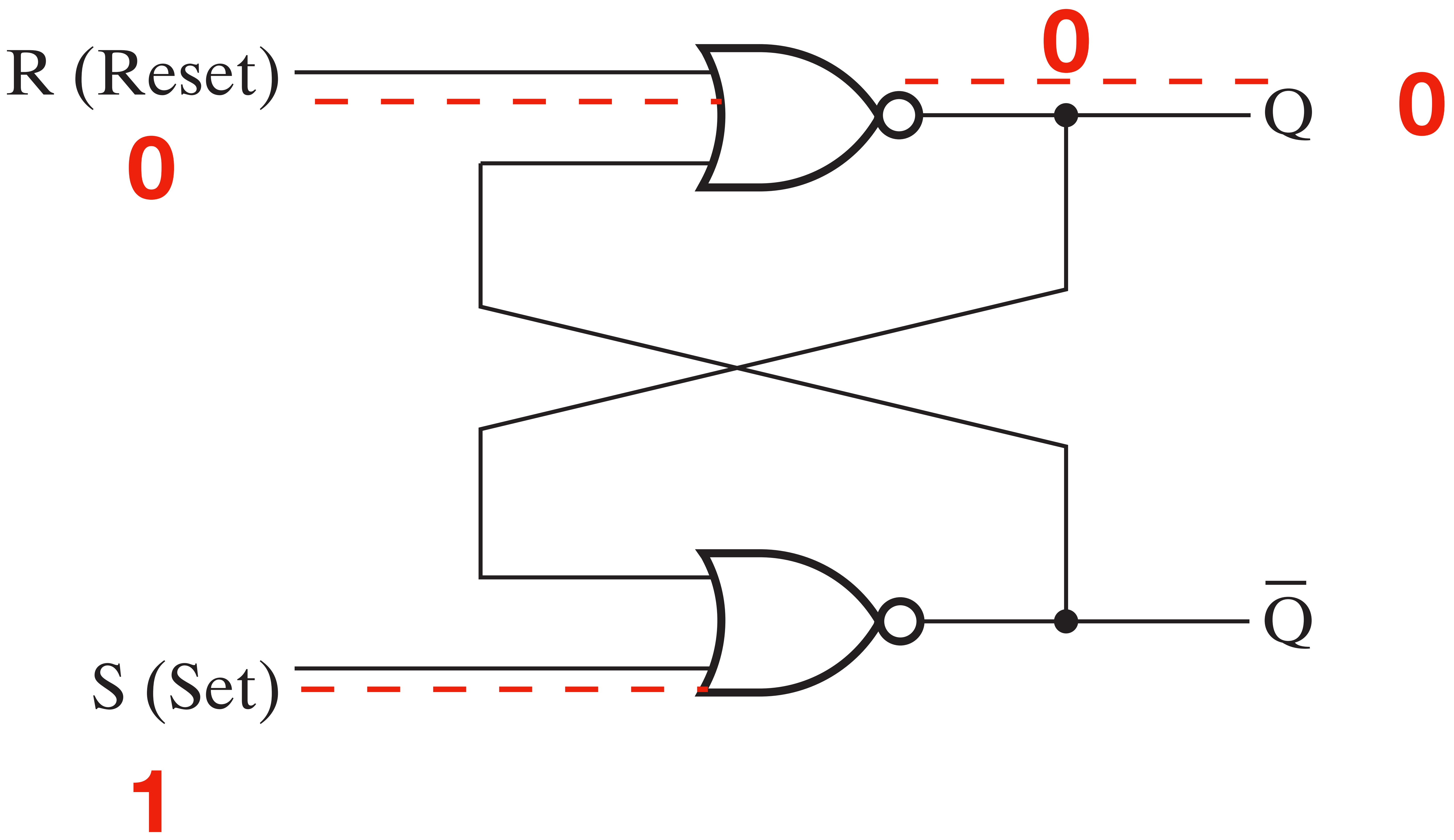
SR Latch: Stored value = 0



Demo

P2
Latches

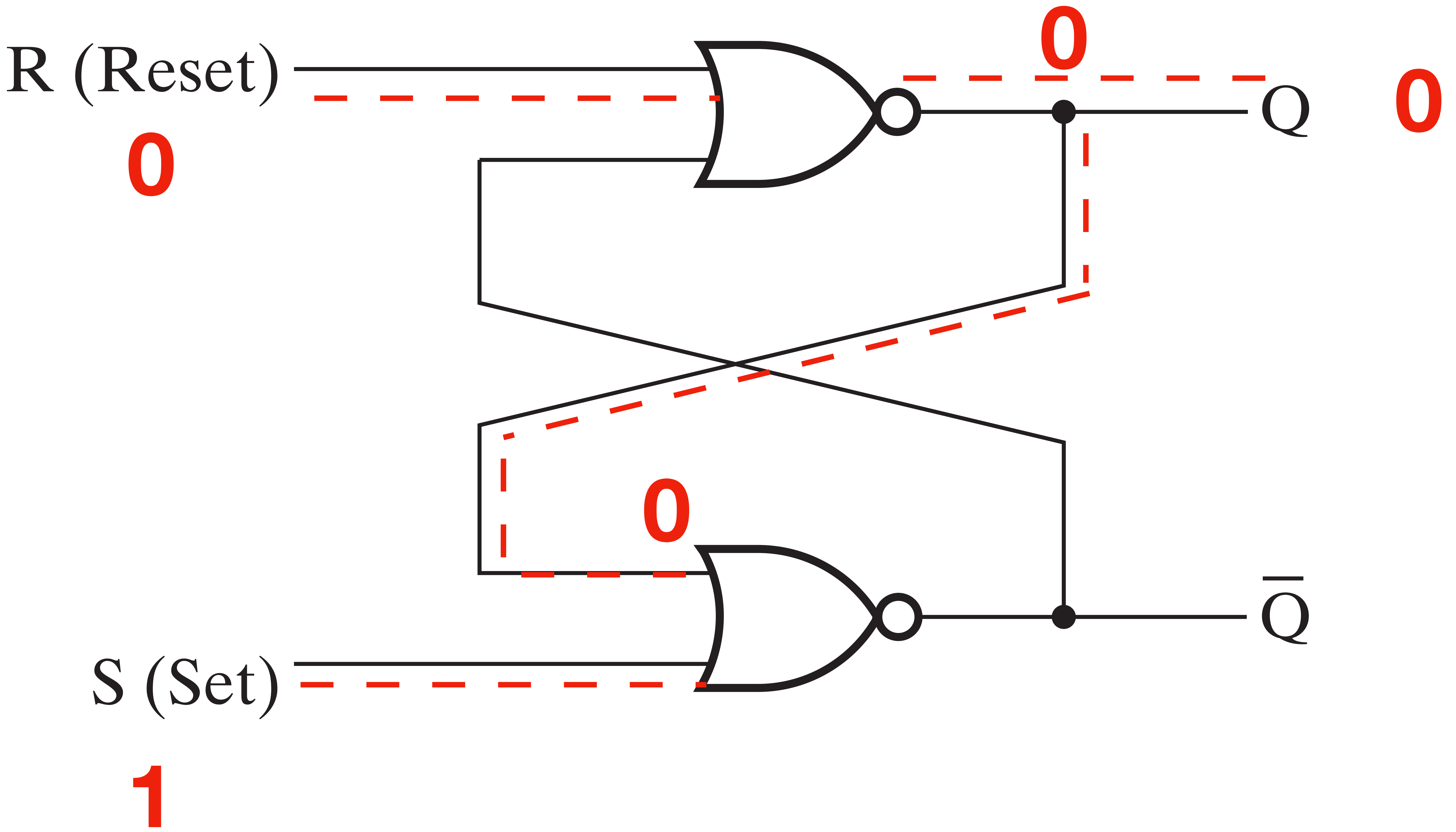
SR Latch: Stored value = 0



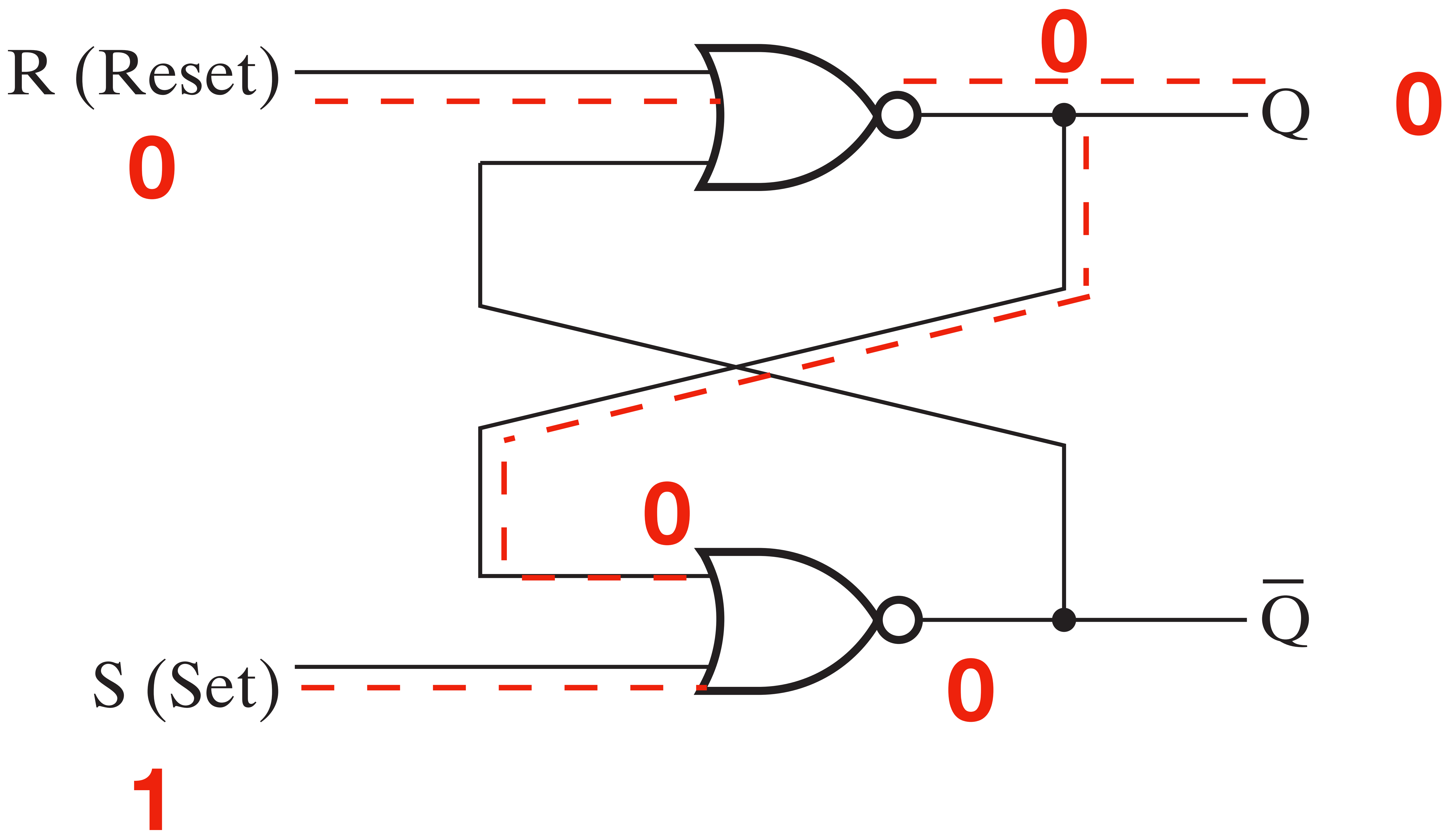
Demo

P2
Latches

SR Latch: Stored value = 0

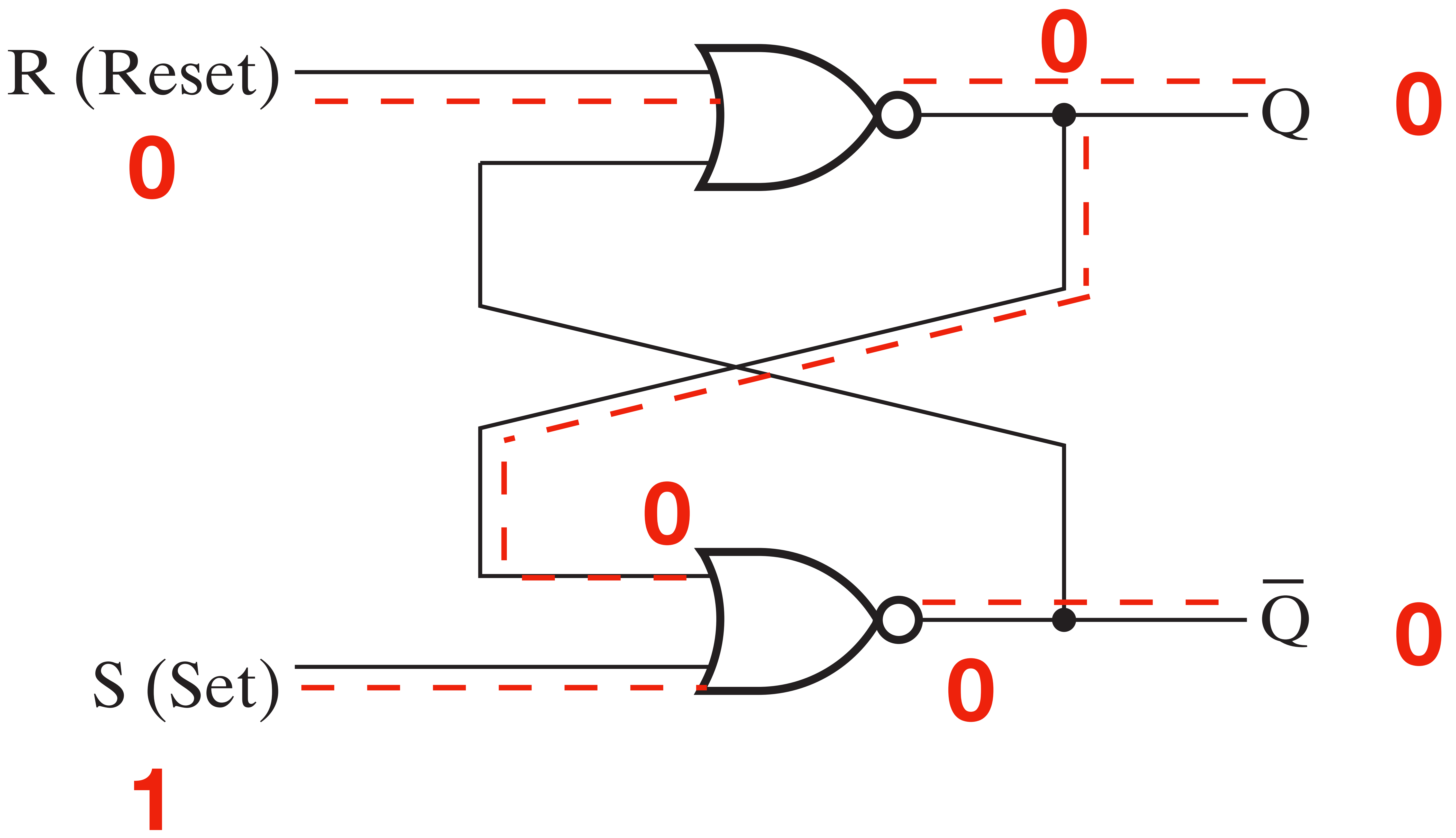


SR Latch: Stored value = 0



P2
Latches

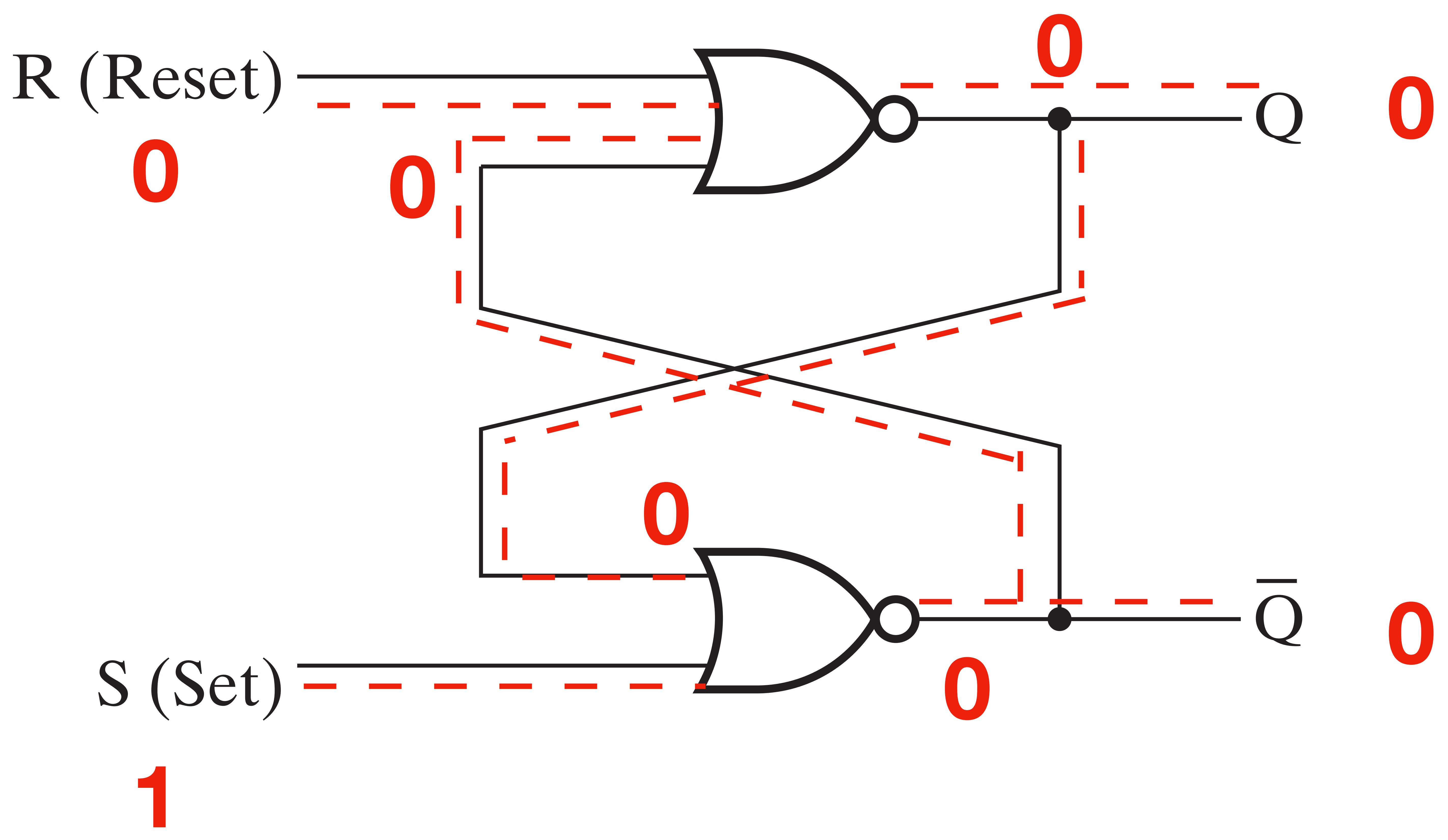
SR Latch: Stored value = 0



Demo

P2
Latches

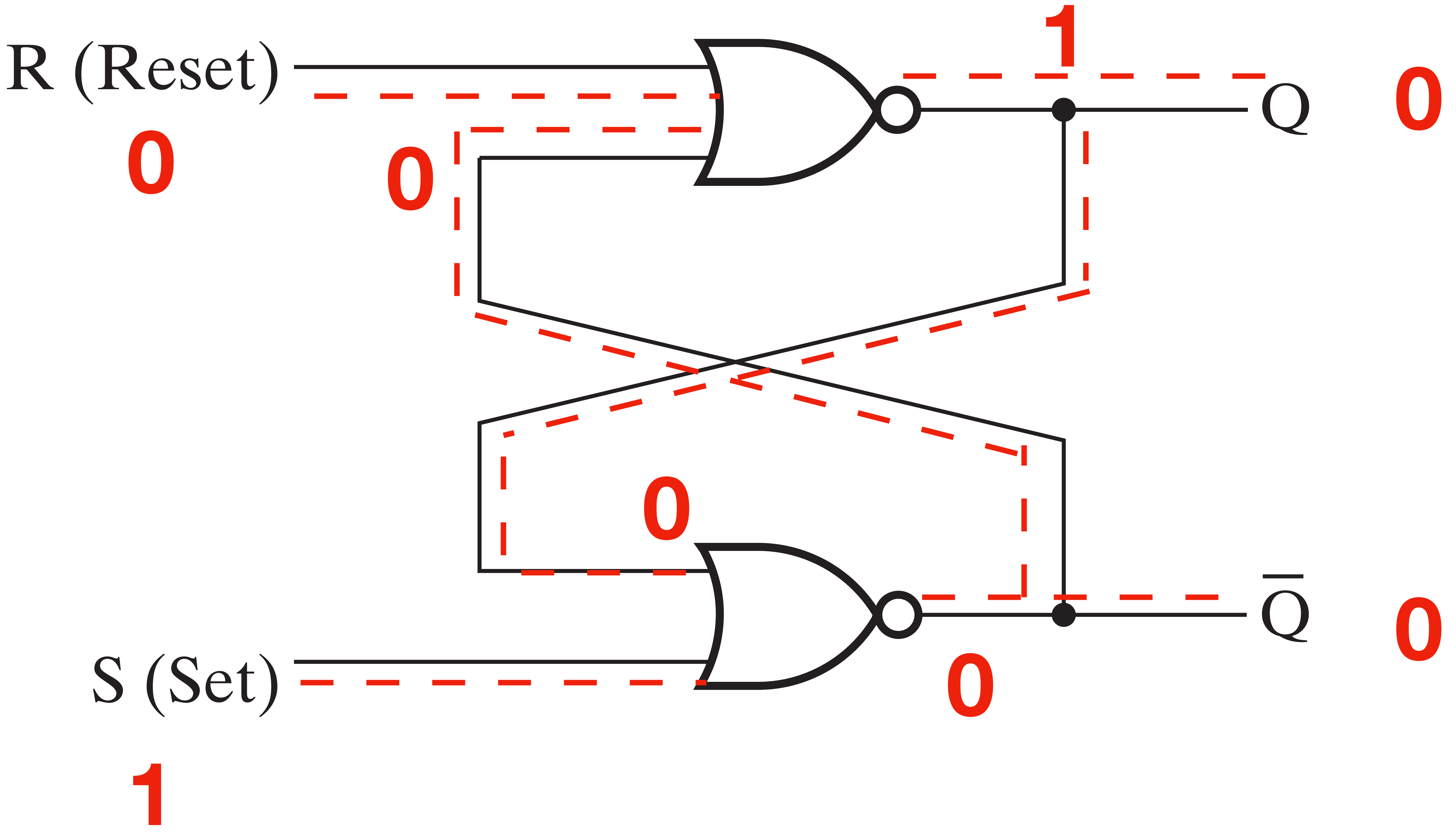
SR Latch: Stored value = 0



Demo

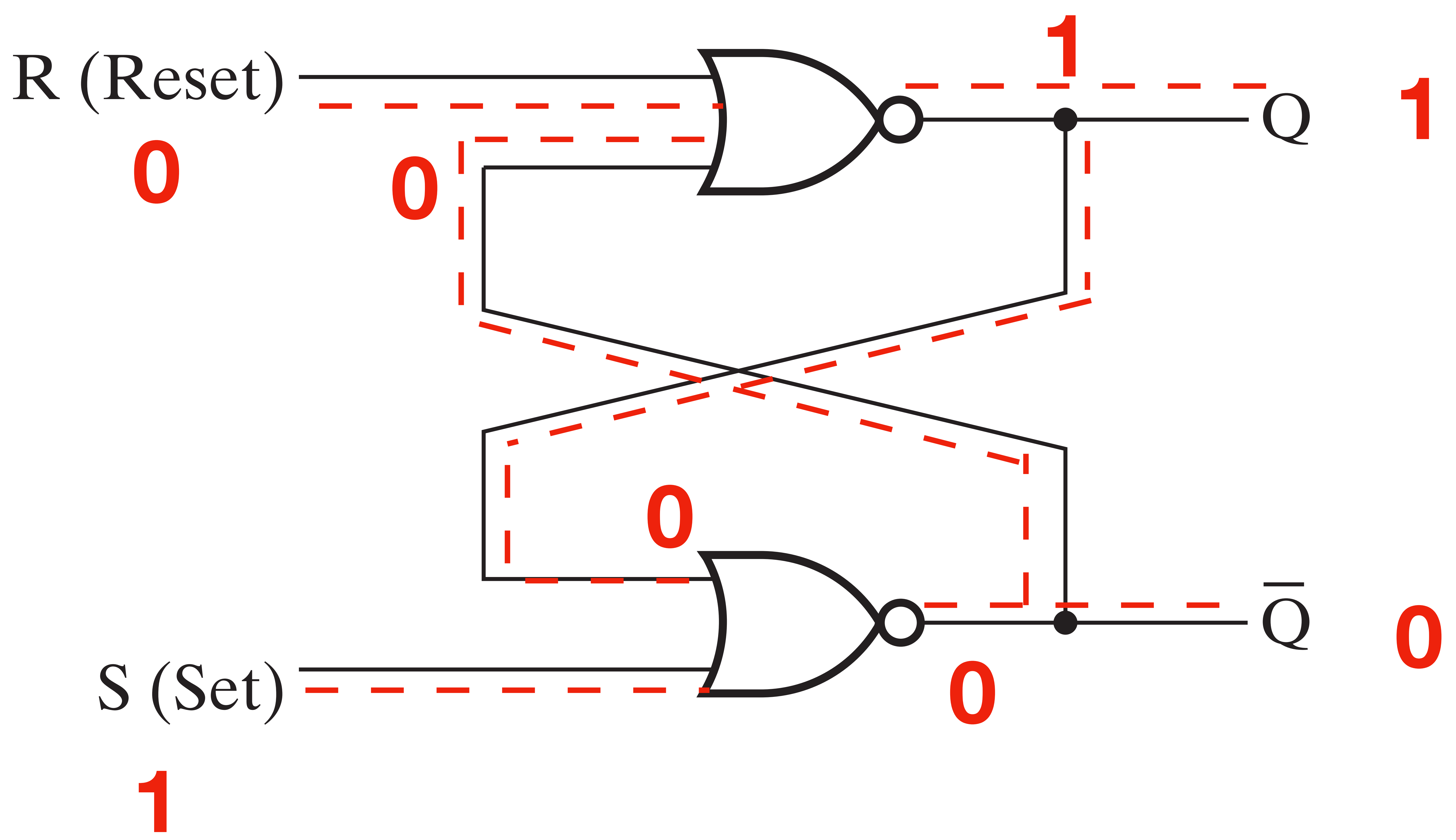
P2
Latches

SR Latch: Stored value = 0

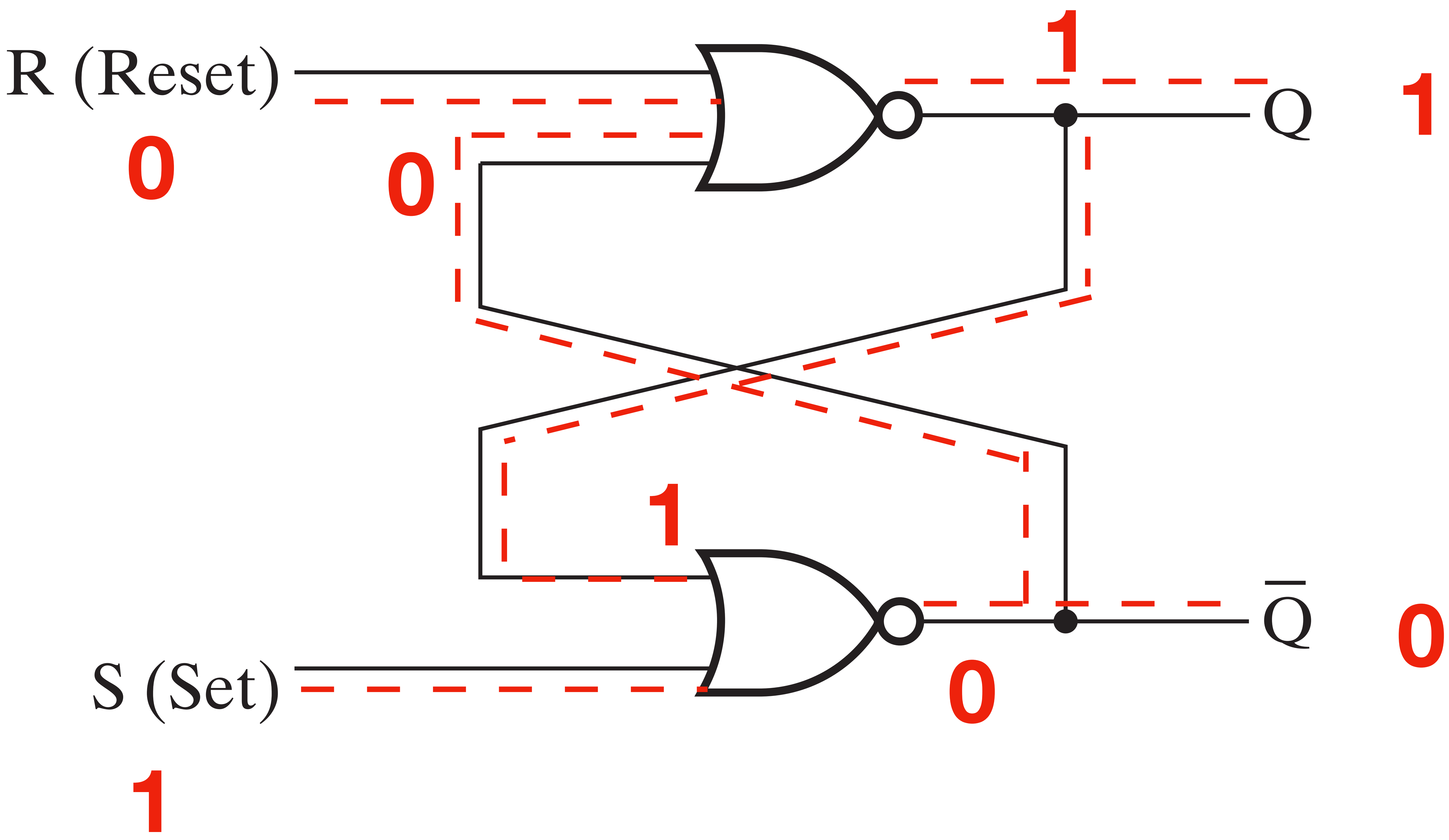


Demo

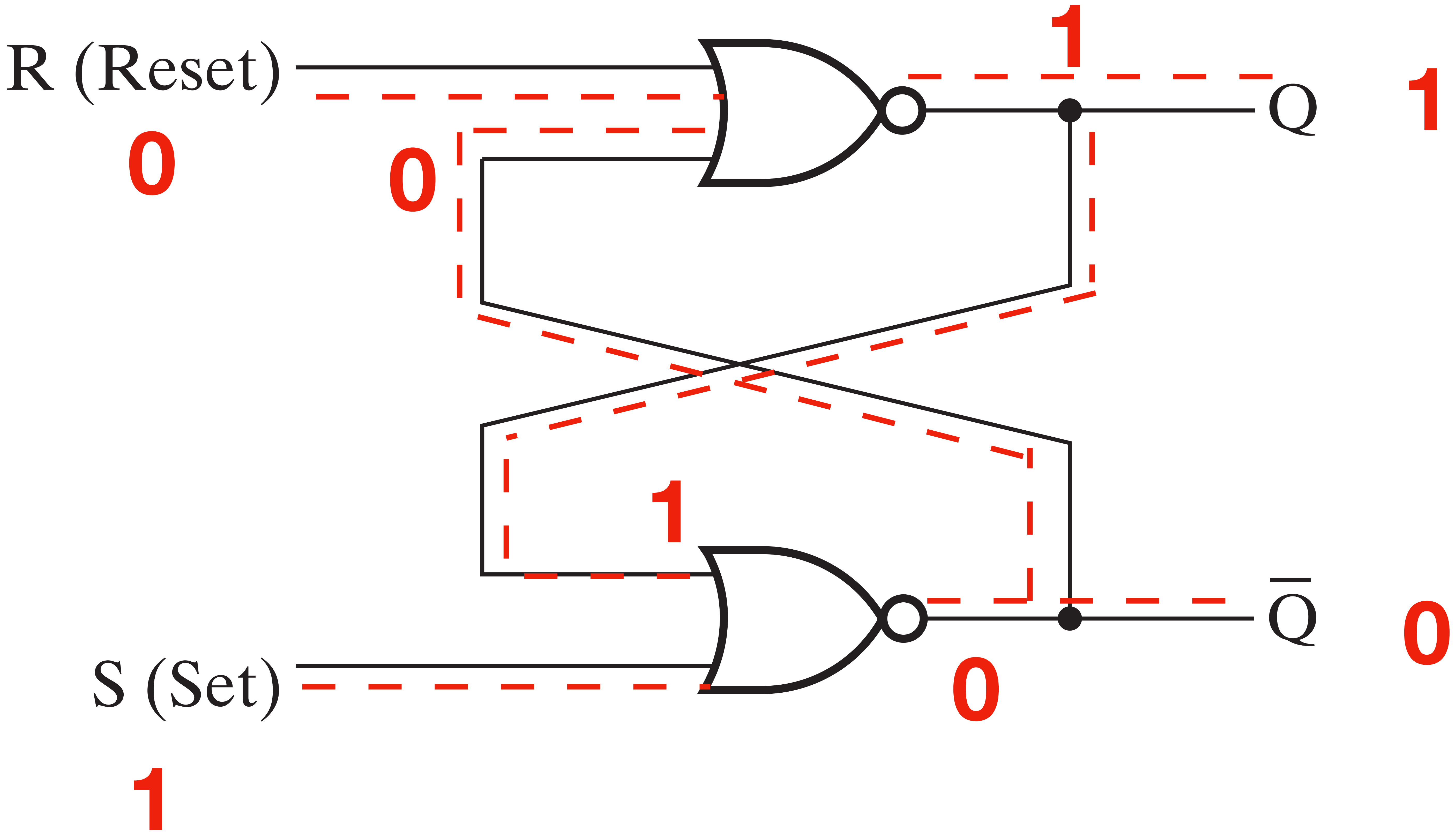
SR Latch: Stored value = 0



SR Latch: Stored value = 0

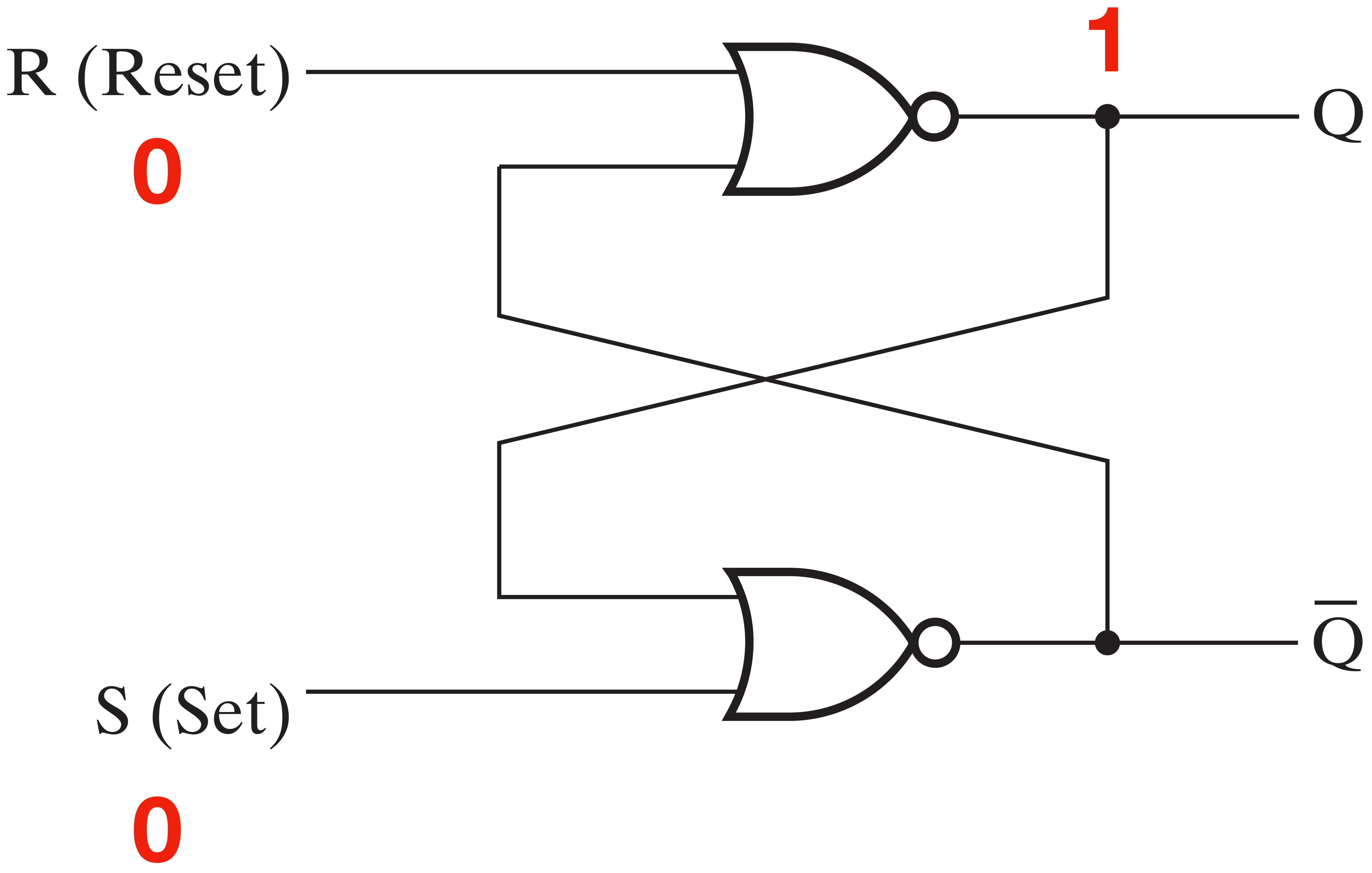


SR Latch: Stored value = 0



P2
Latches

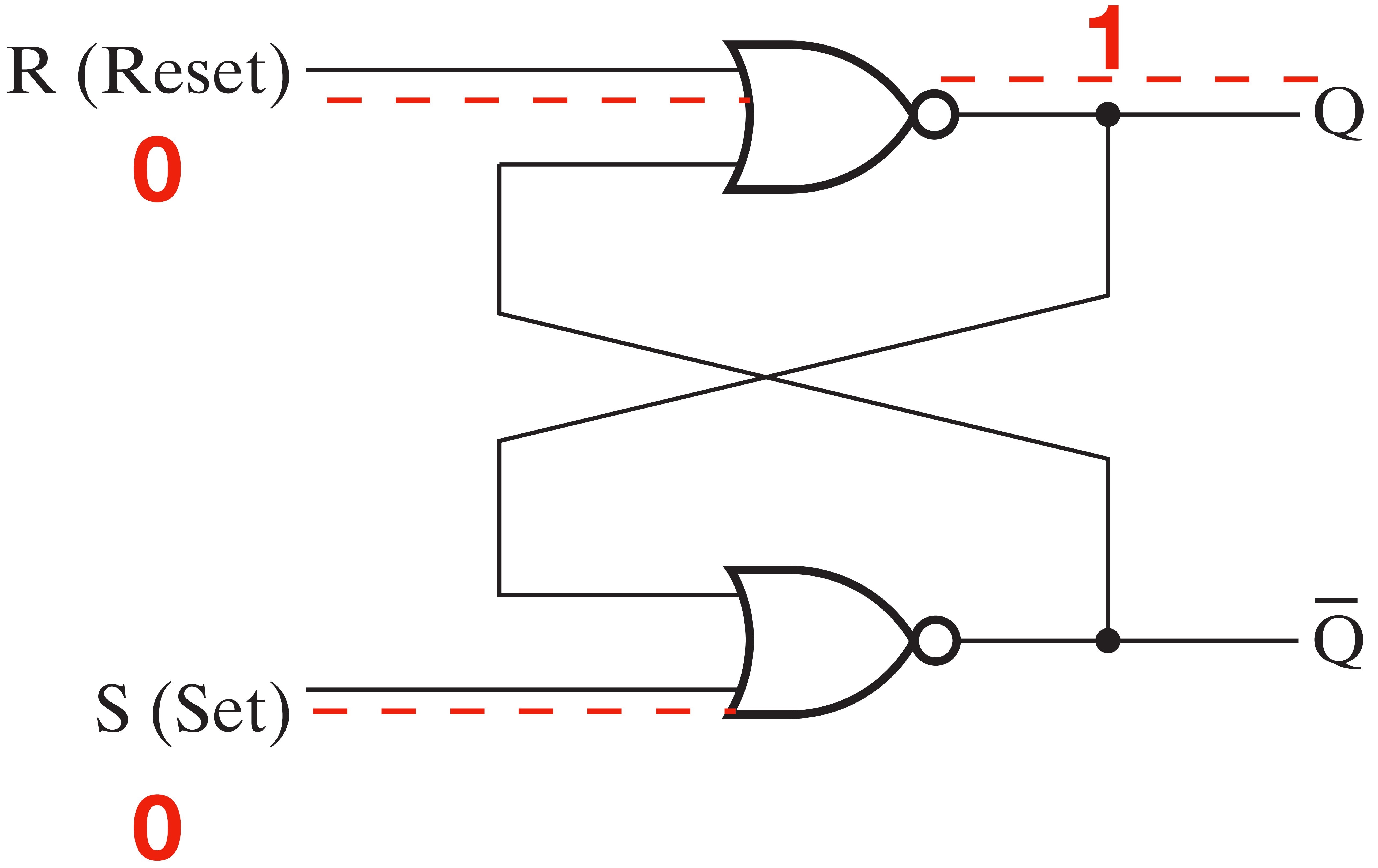
SR Latch: Stored value = 1



Demo

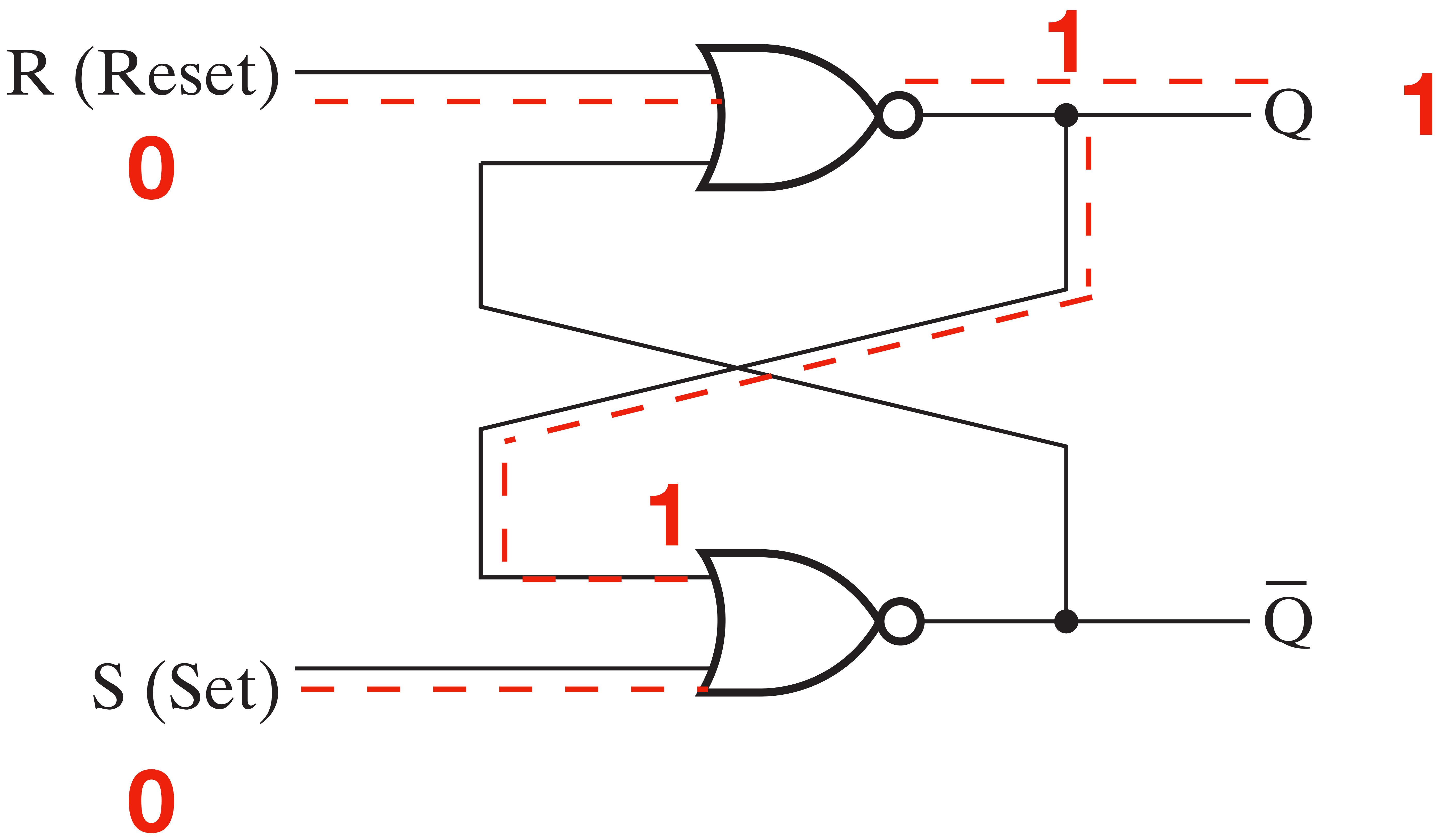
P2
Latches

SR Latch: Stored value = 1

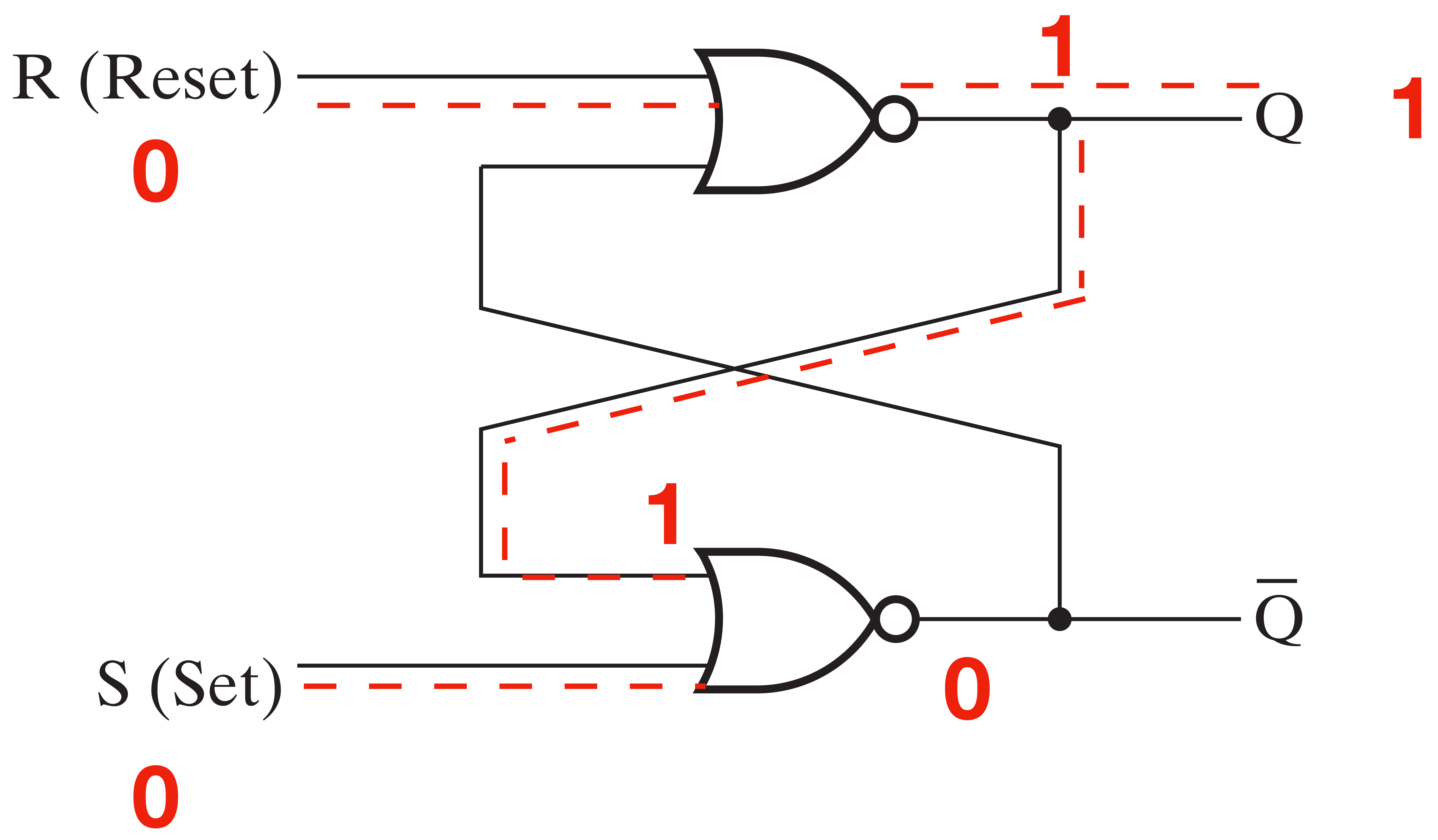


Demo

SR Latch: Stored value = 1



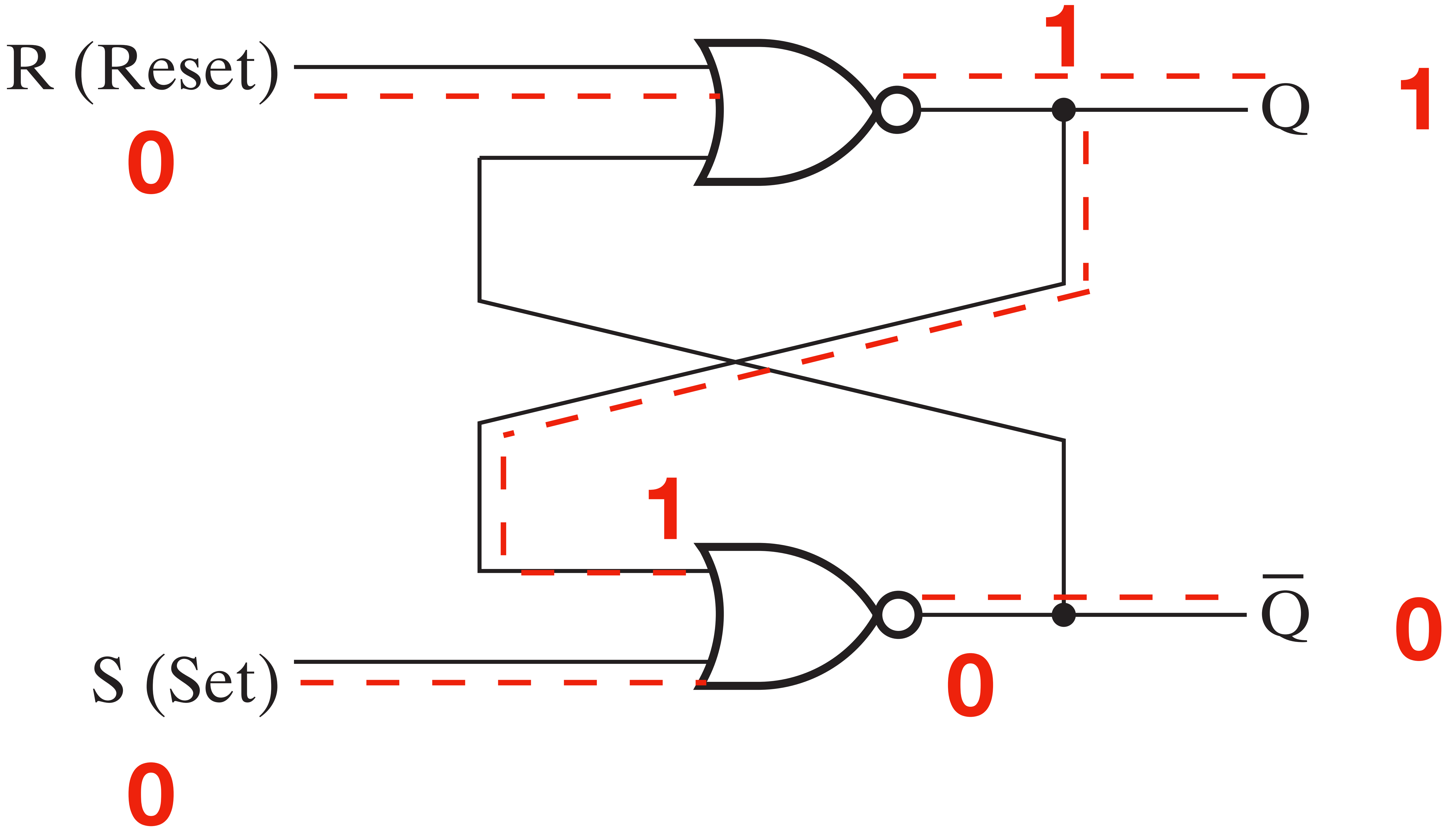
SR Latch: Stored value = 1



Demo

P2
Latches

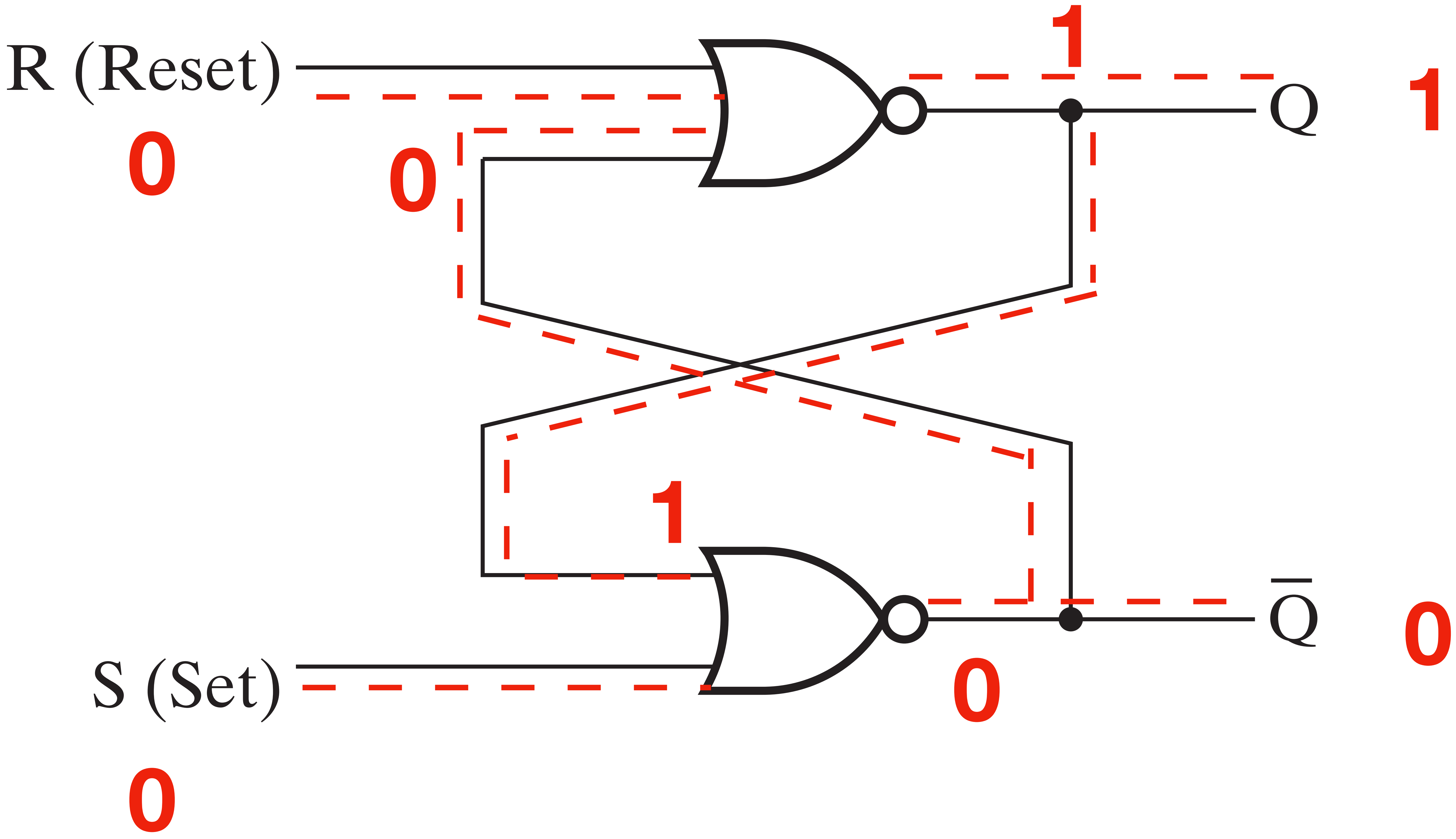
SR Latch: Stored value = 1



Demo

P2
Latches

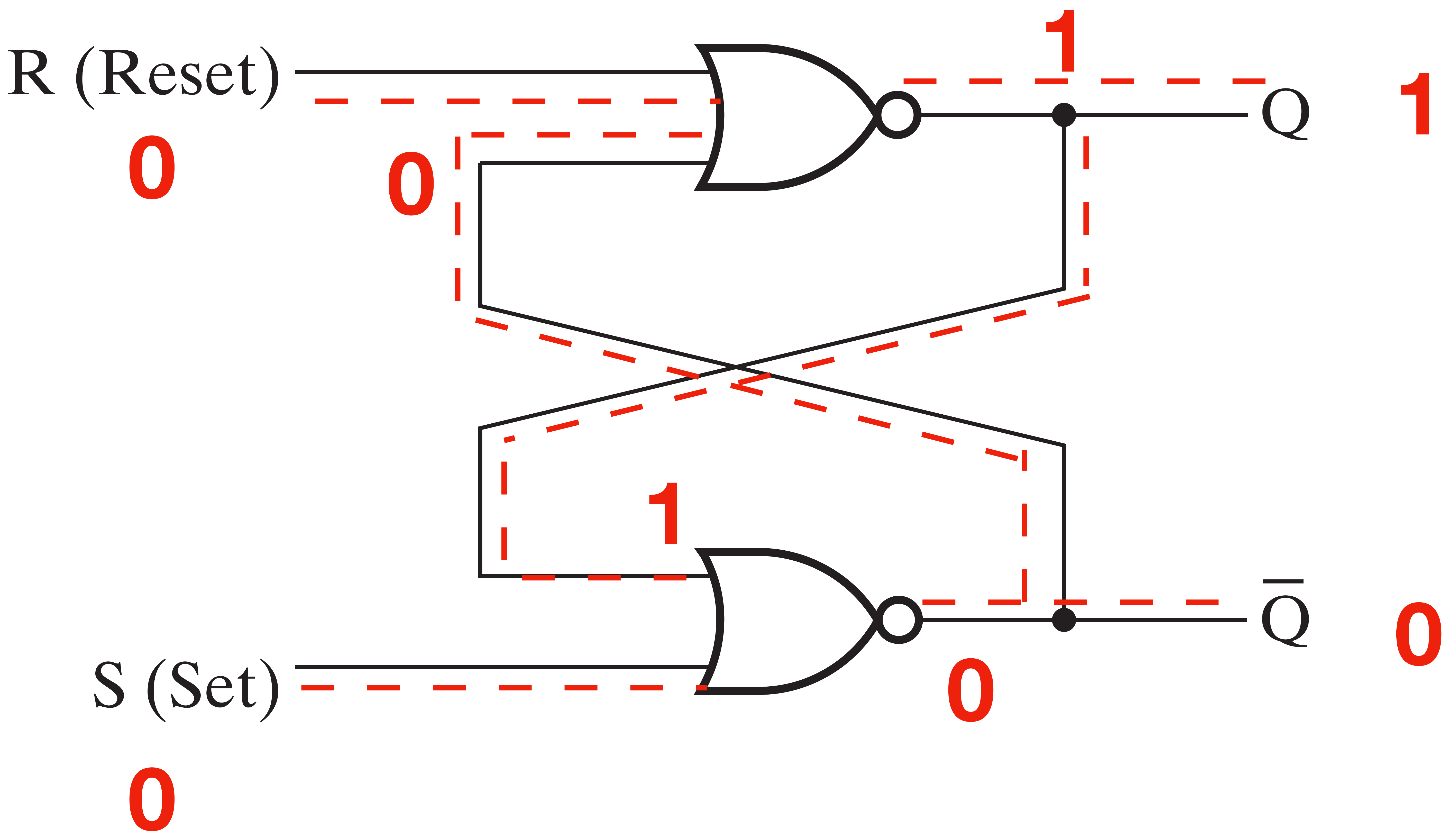
SR Latch: Stored value = 1



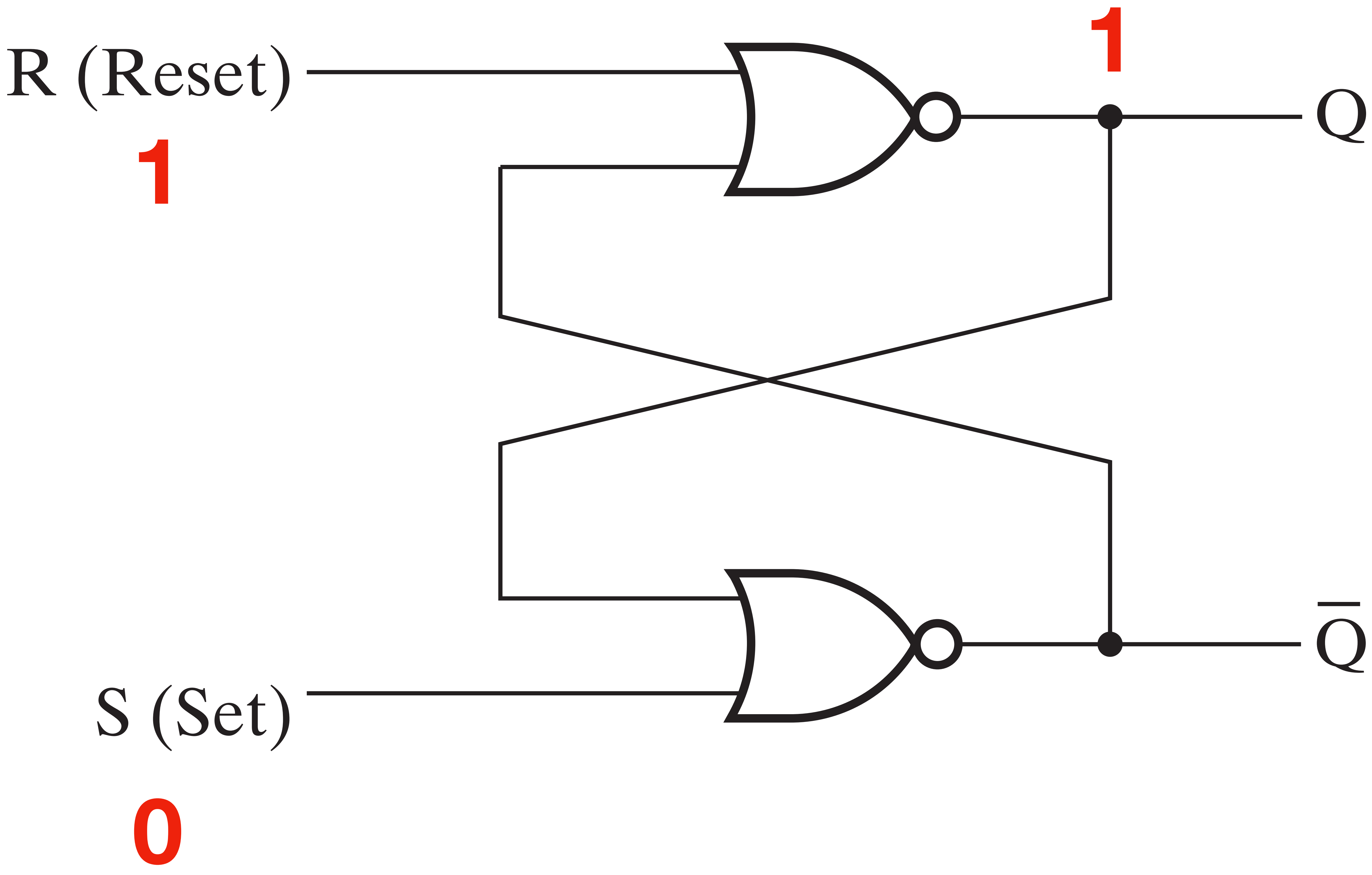
Demo

P2
Latches

SR Latch: Stored value = 1



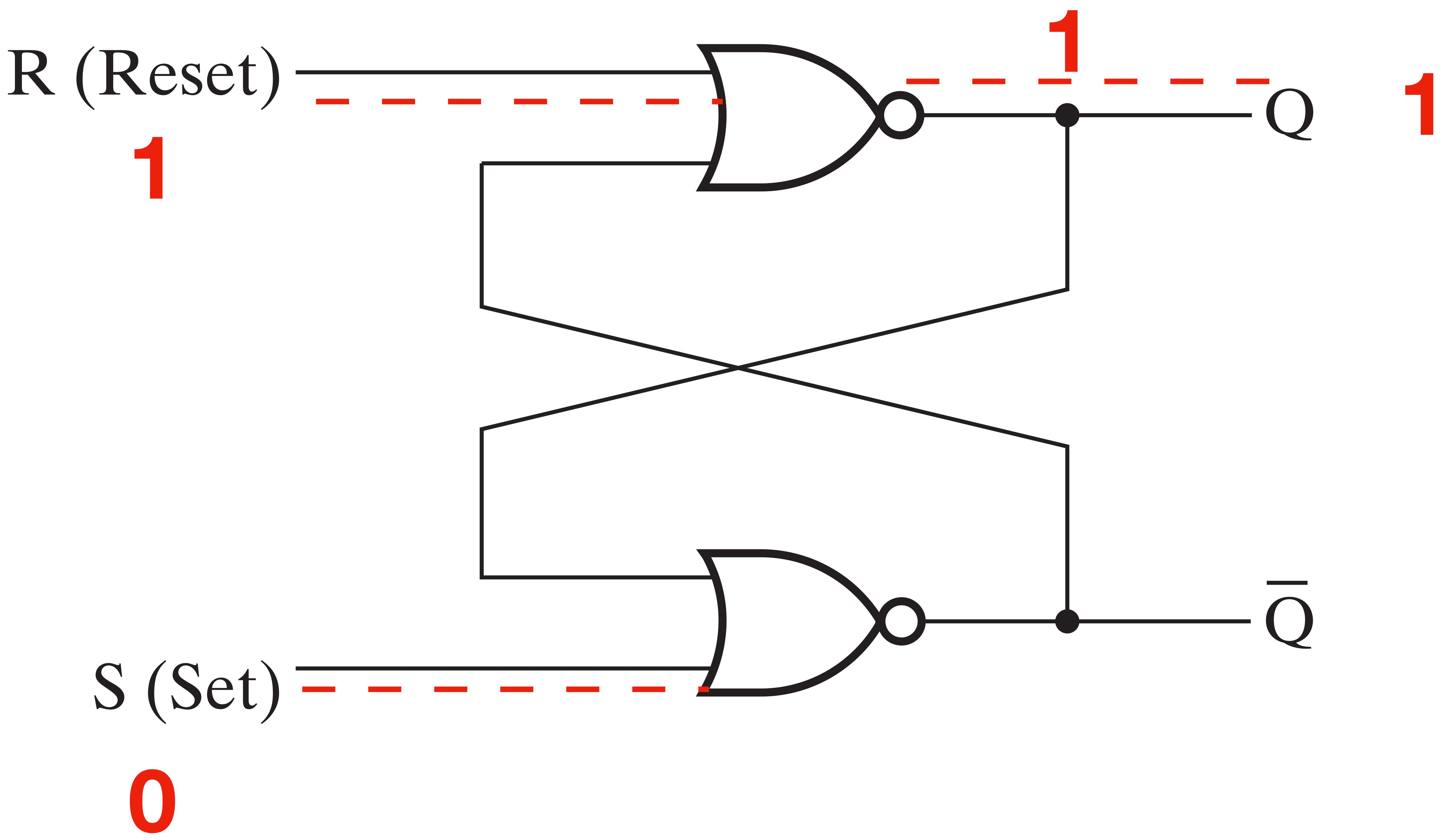
SR Latch: Stored value = 1



Demo

P2
Latches

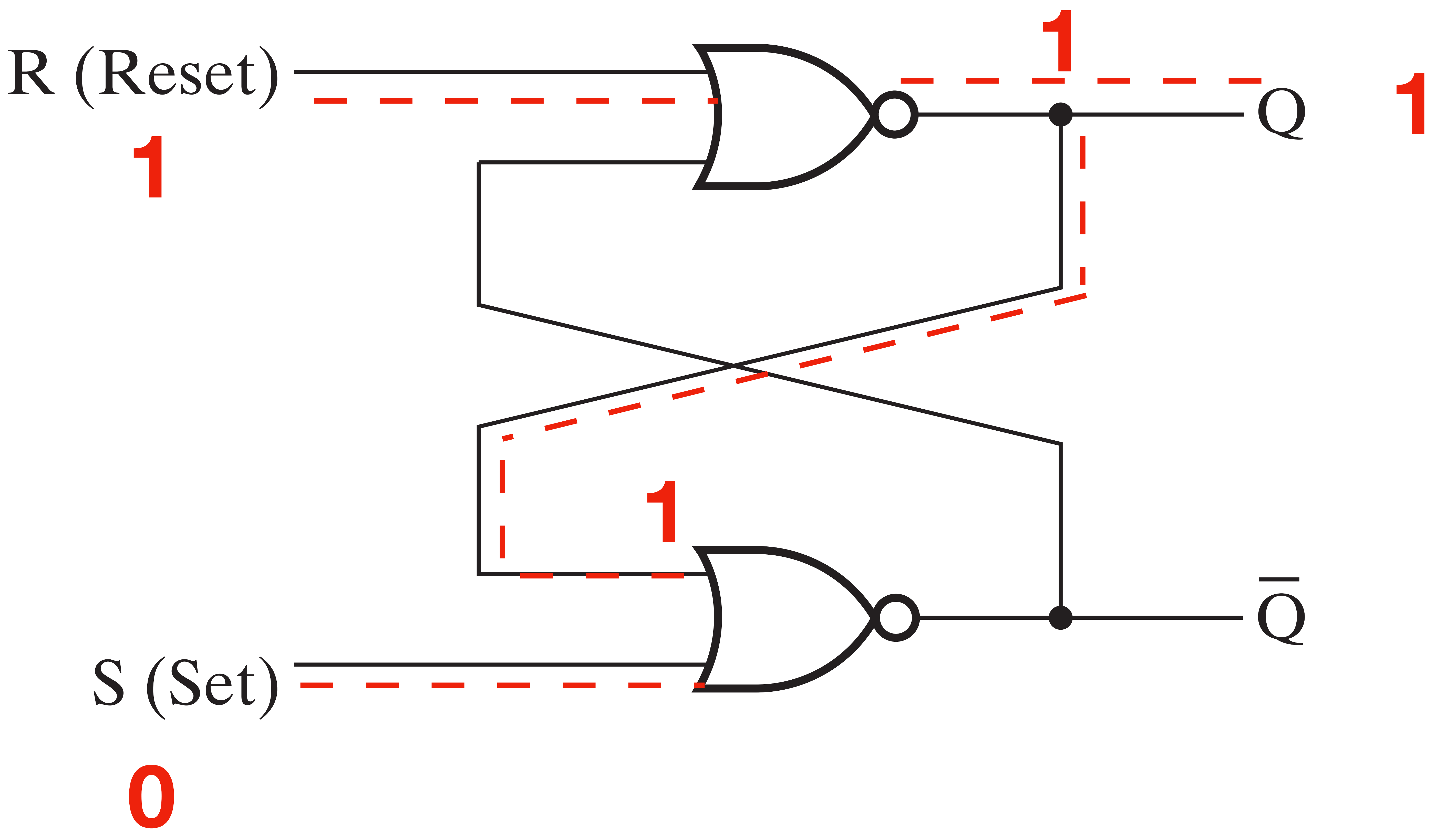
SR Latch: Stored value = 1



Demo

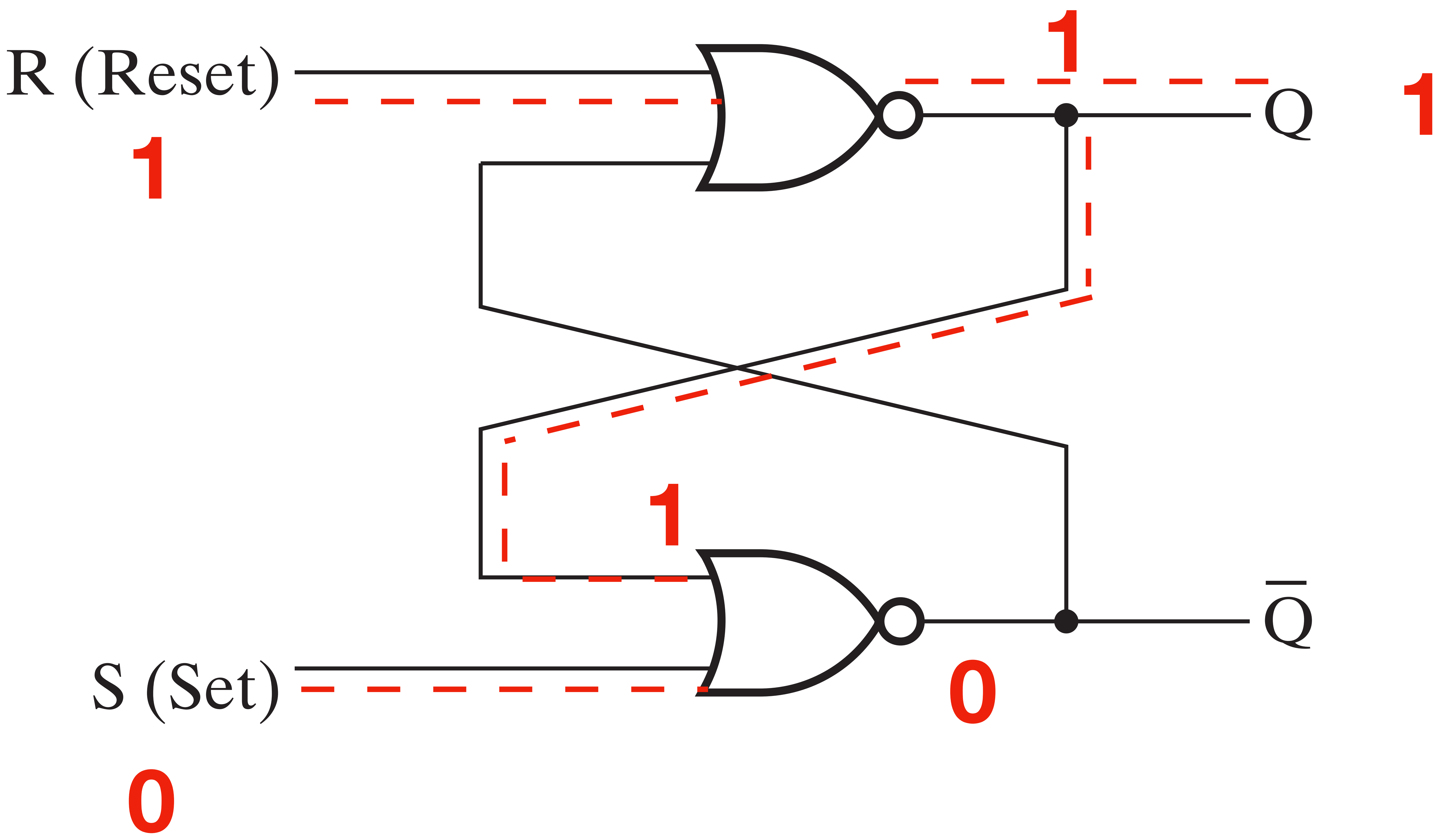
P2
Latches

SR Latch: Stored value = 1



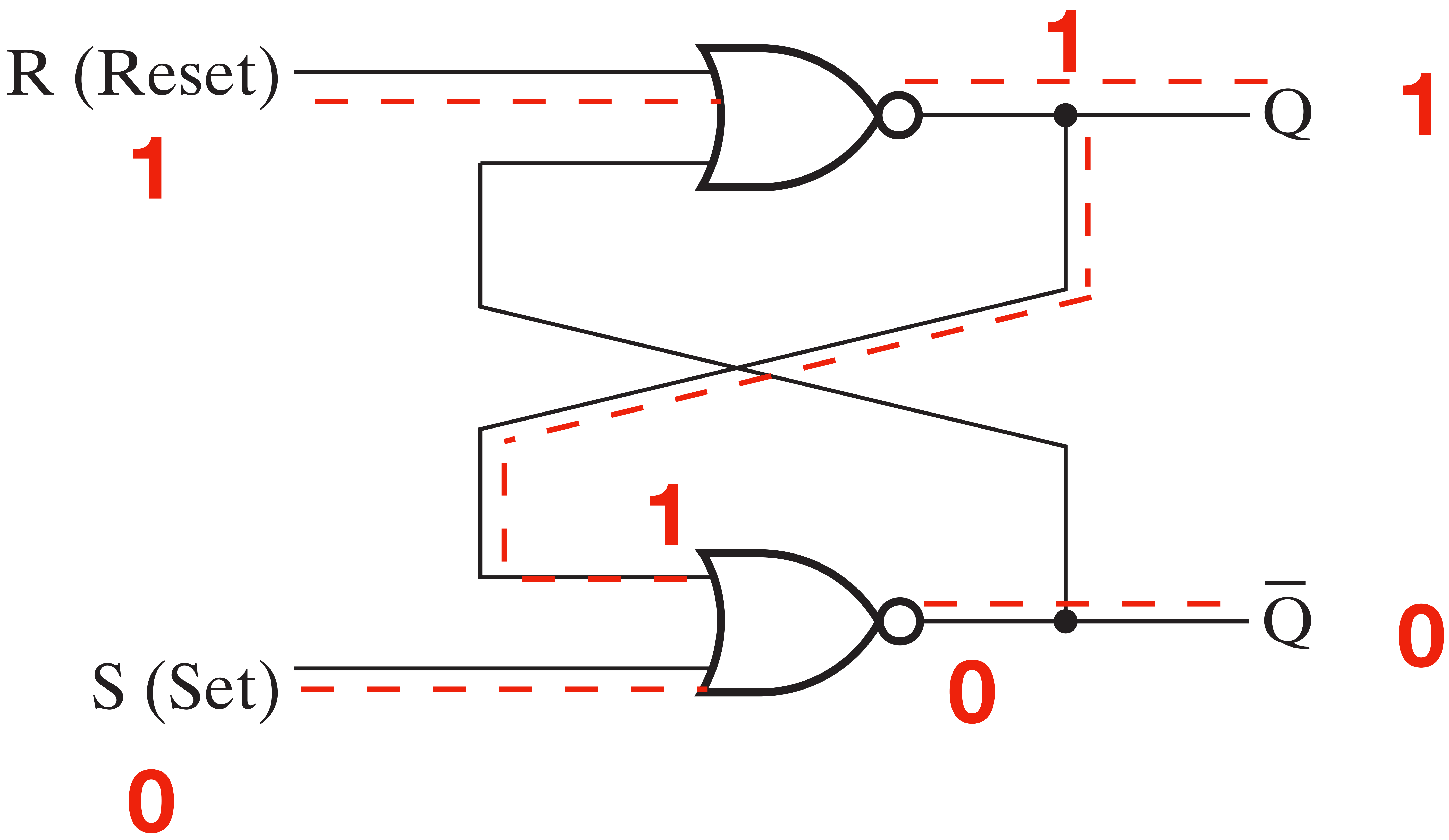
P2
Latches

SR Latch: Stored value = 1



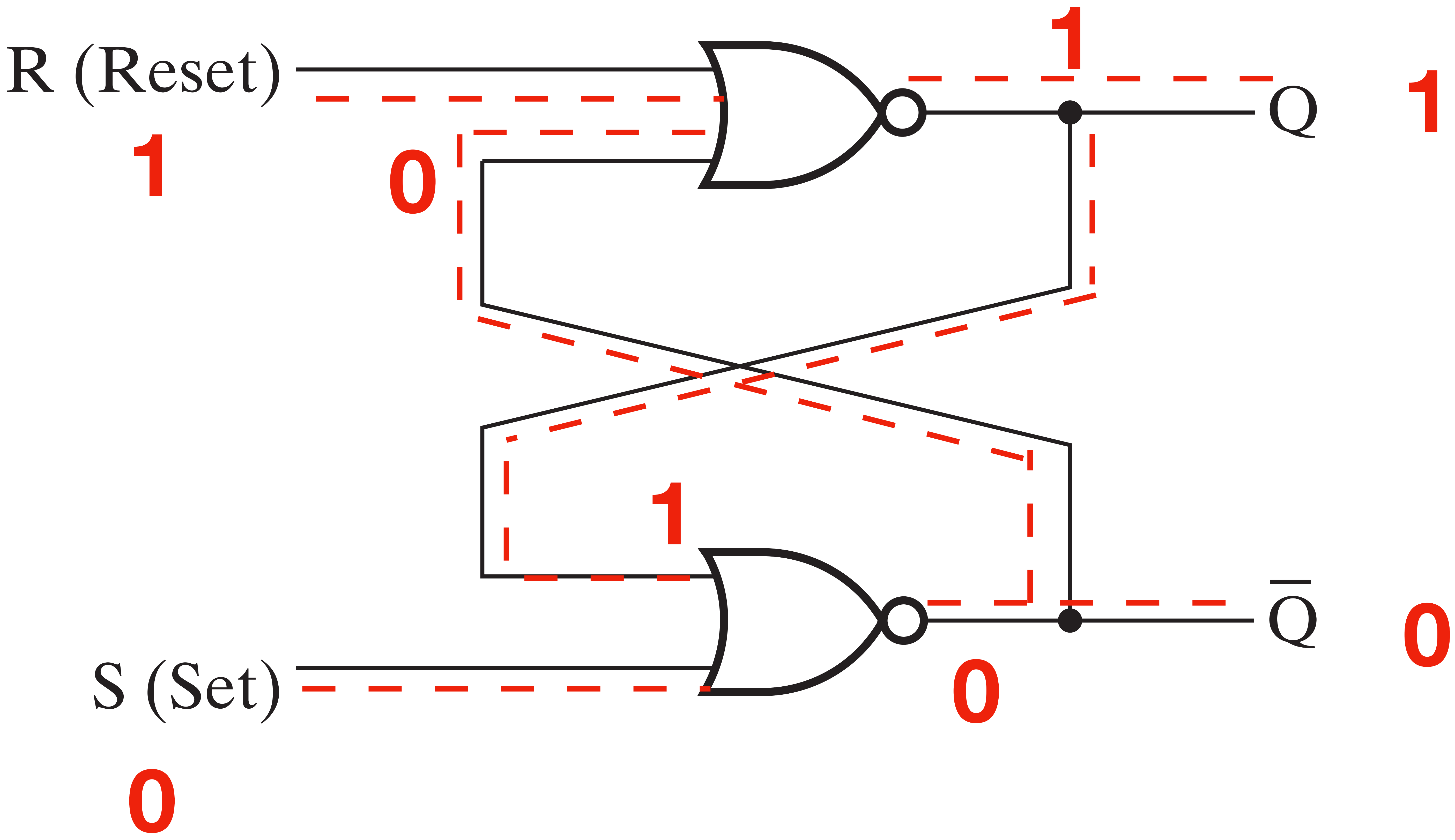
Demo

SR Latch: Stored value = 1



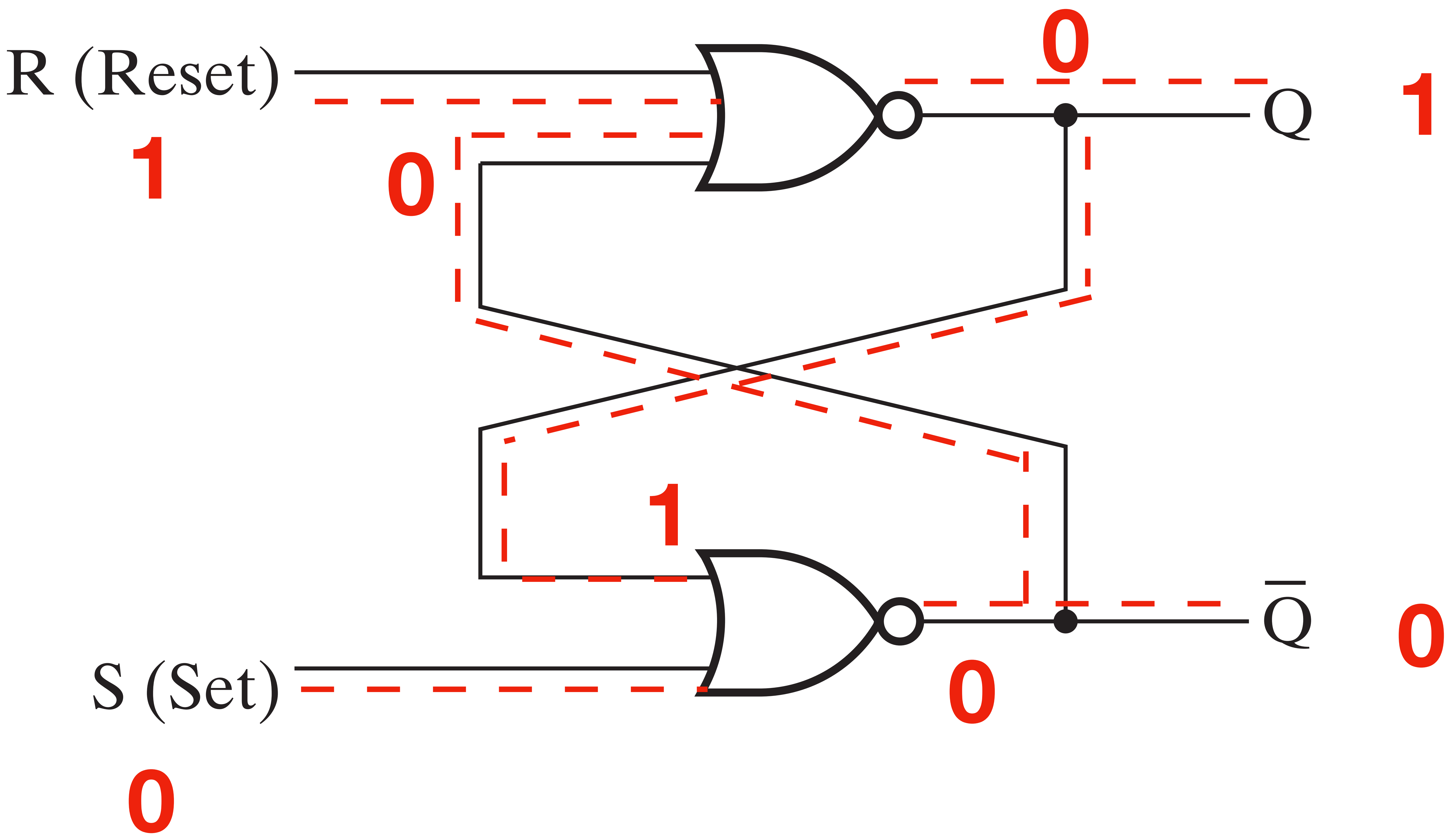
P2
Latches

SR Latch: Stored value = 1

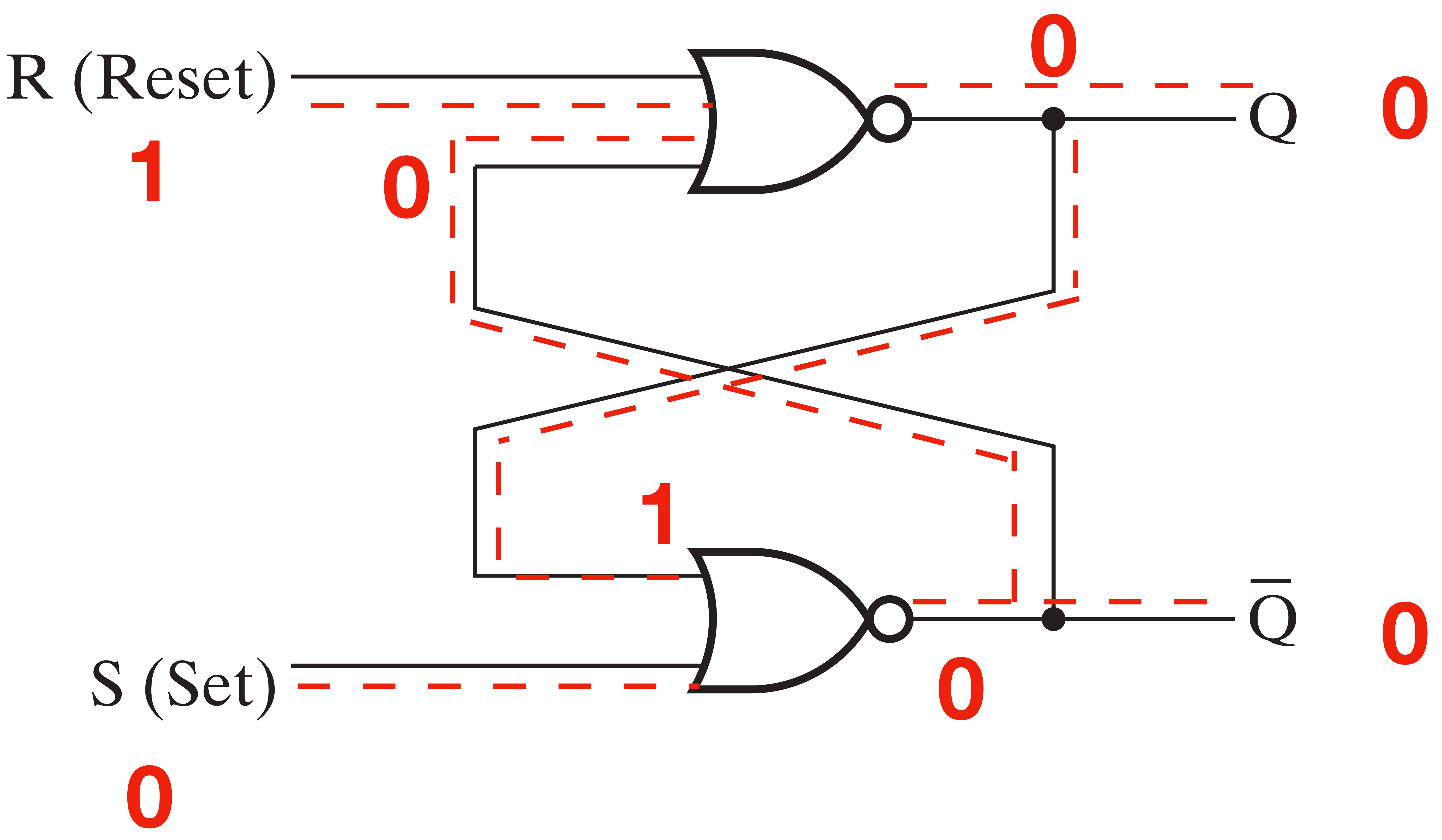


Demo

SR Latch: Stored value = 1

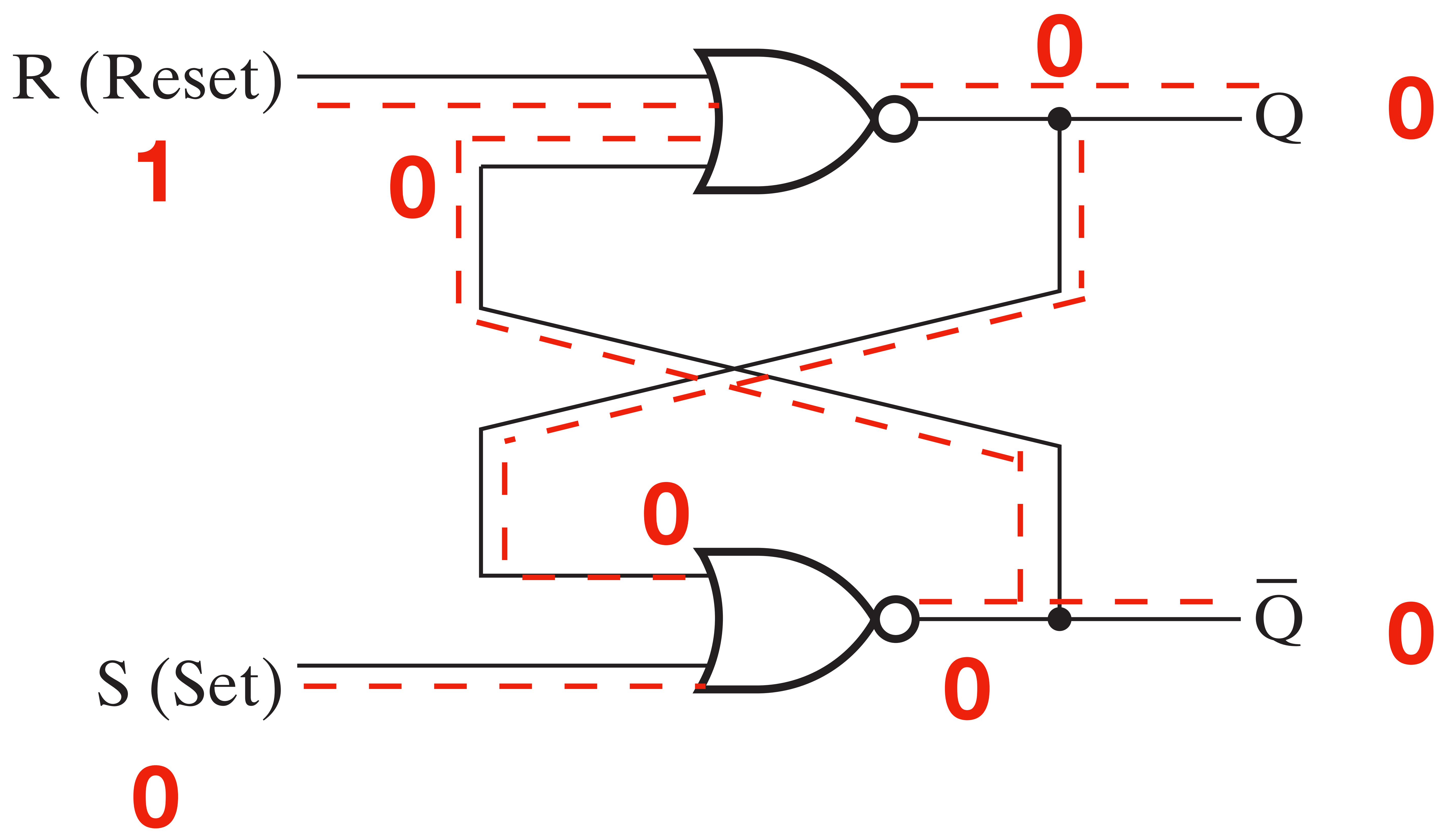


SR Latch: Stored value = 1



P2
Latches

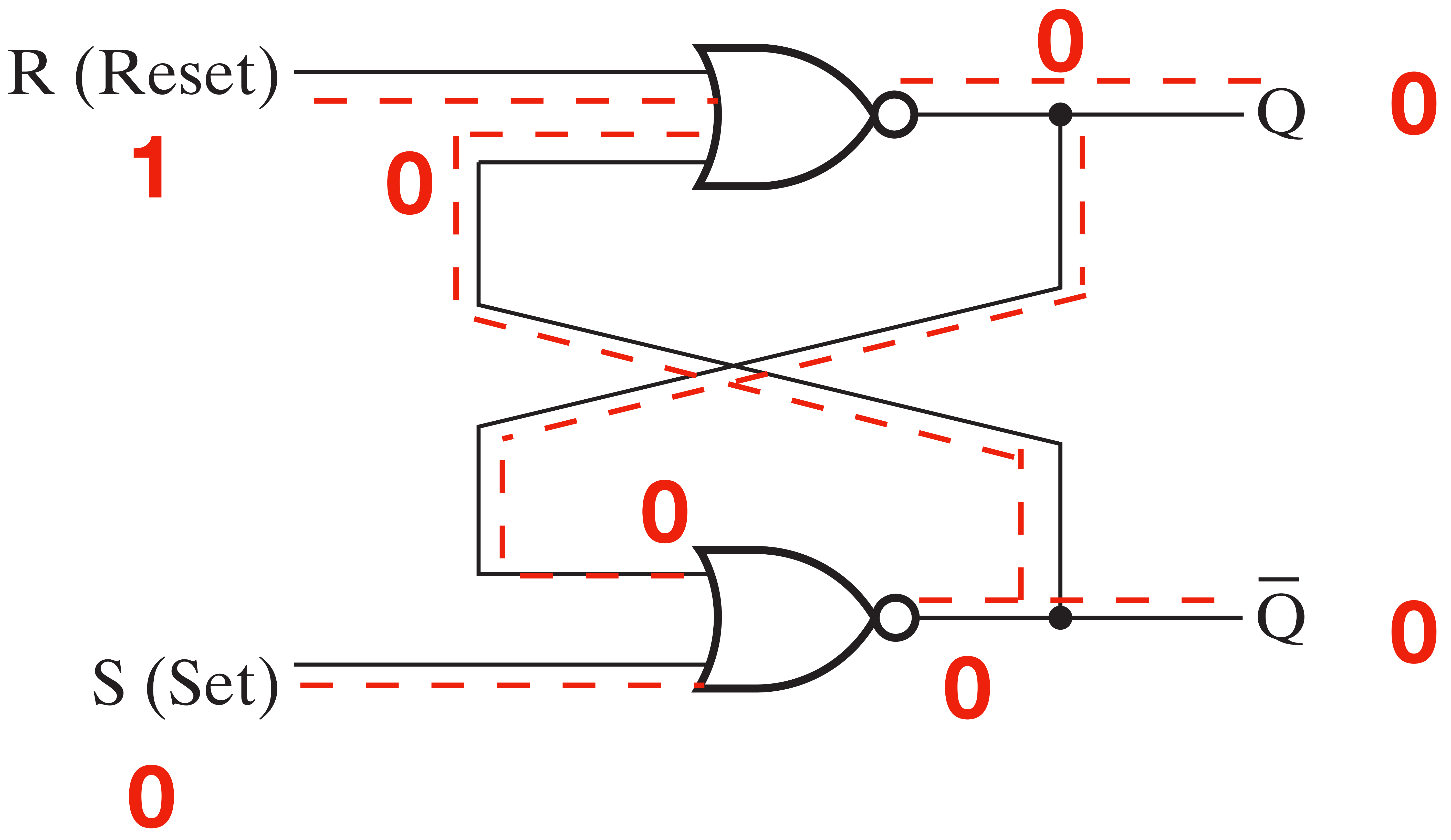
SR Latch: Stored value = 1



Demo

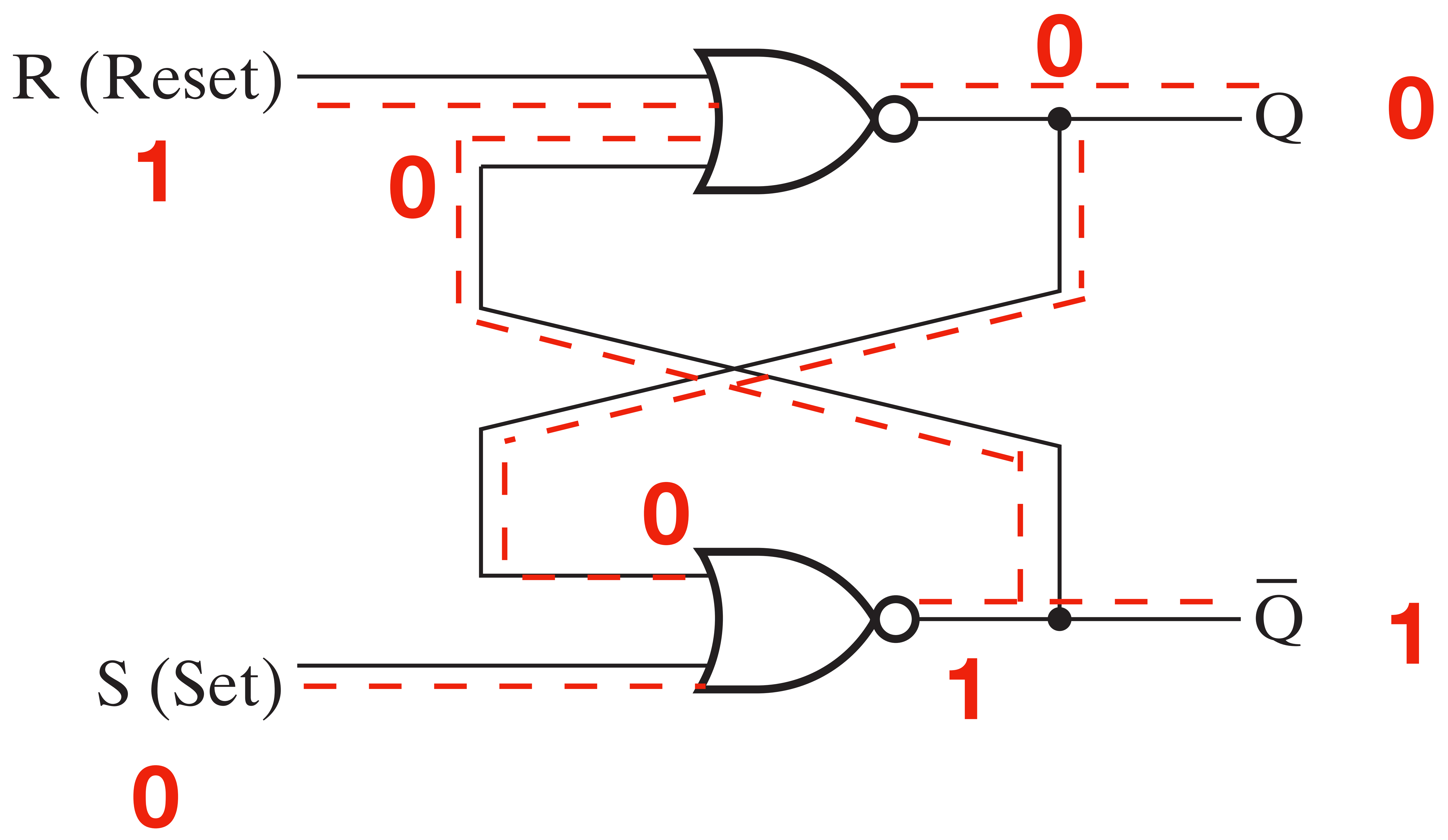
P2
Latches

SR Latch: Stored value = 1



P2
Latches

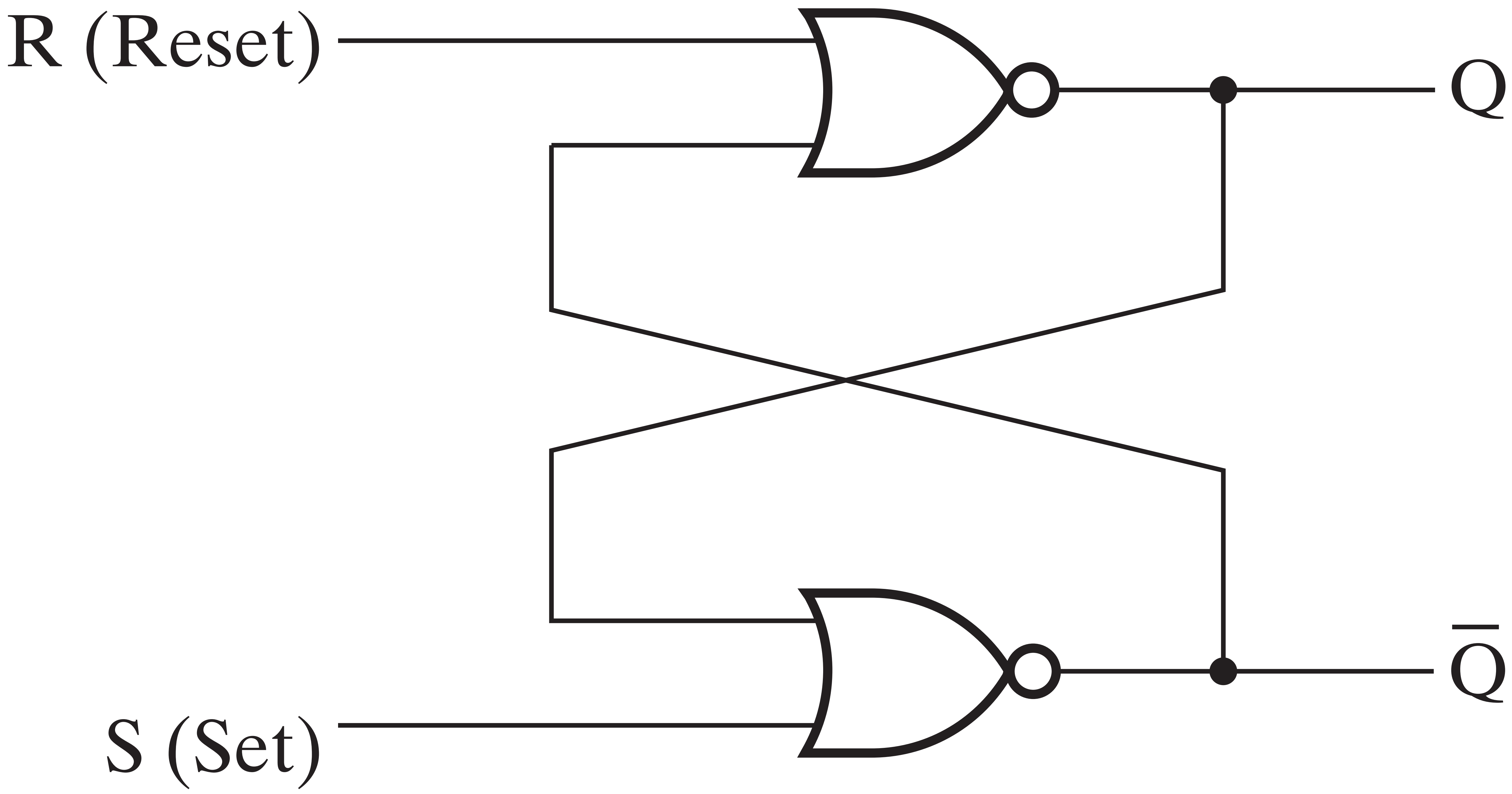
SR Latch: Stored value = 1



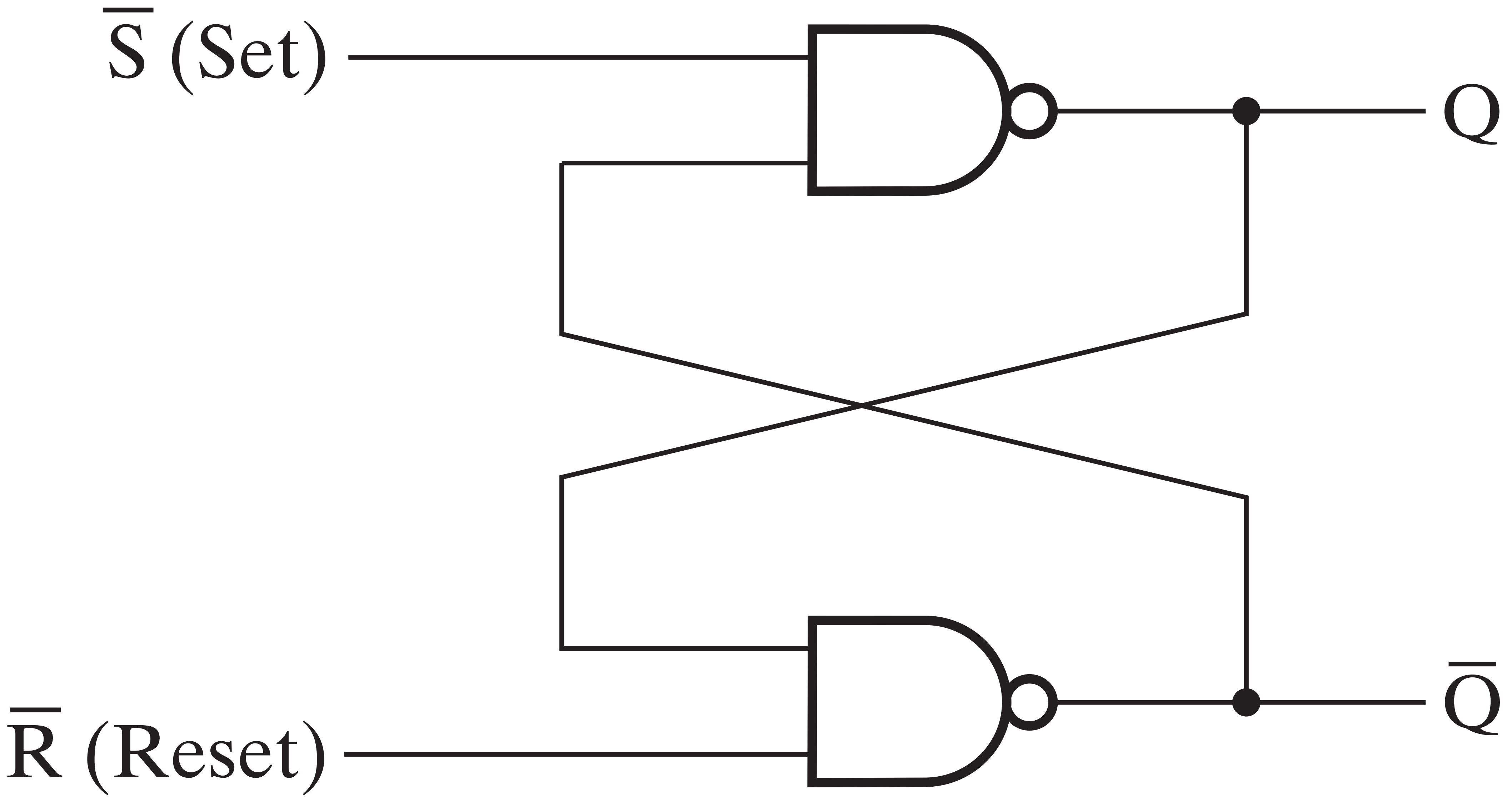
Demo

Exercise

- Draw the SR Latch in LogicWorks as below, make sure it works as intended



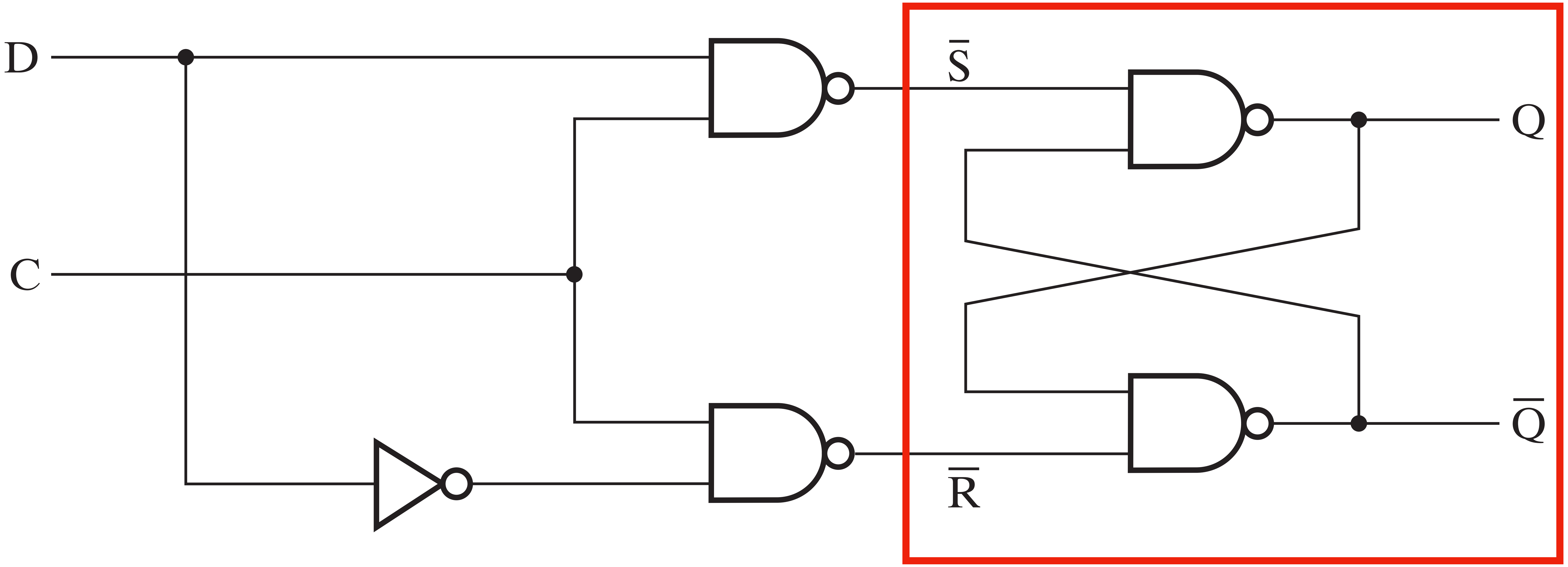
$\overline{SR}$ Latch


$\overline{\mathrm{S}}$ (Set)
Q
$\overline{\mathrm{R}}$ (Reset)
$\overline{\mathrm{Q}}$

$\overline{\mathrm{S}}$	$\overline{\mathrm{R}}$	Q	$\overline{\mathrm{Q}}$	
0	1	1	0	Set state
1	1	1	0	
1	0	0	1	Reset state
1	1	0	1	
0	0	1	1	Undefined

- Design similar to SR latches, but with NANDS
- Functions equivalent to SR latches with S and R inverted

D Latch



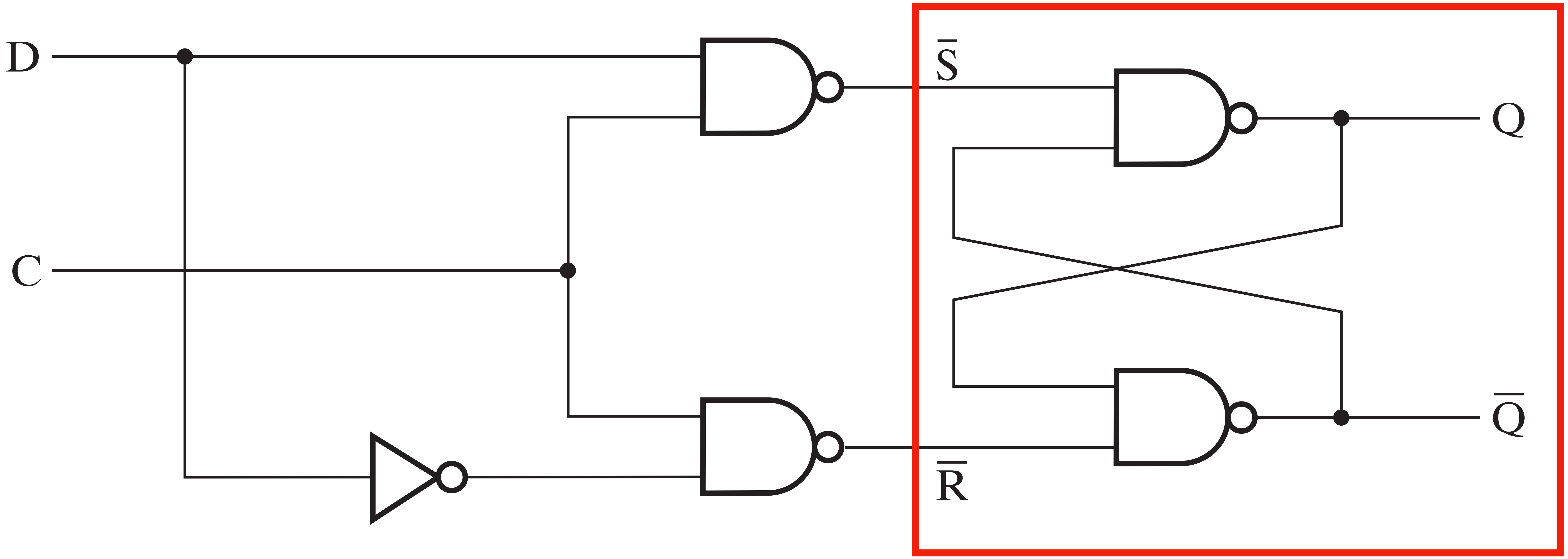
C	D	Next state of Q
0	X	No change
1	0	Q = 0; Reset state
1	1	Q = 1; Set state

- Implemented using $\overline{SR}$ latches
- C: Signals changes to the stored states; D the value to change to

Concept

Latches

- Implement $\overline{SR}$ latch, save as a component in your library
- Implement D latch, save as a component in your library



Exercise