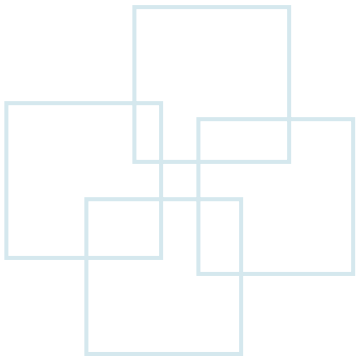


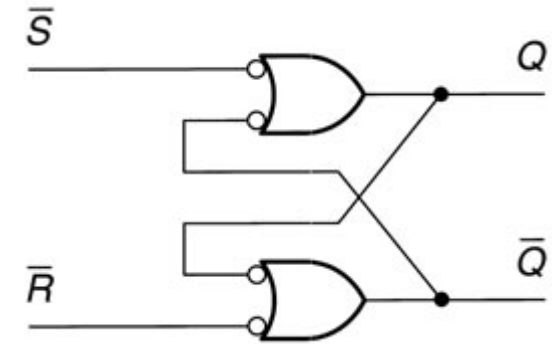
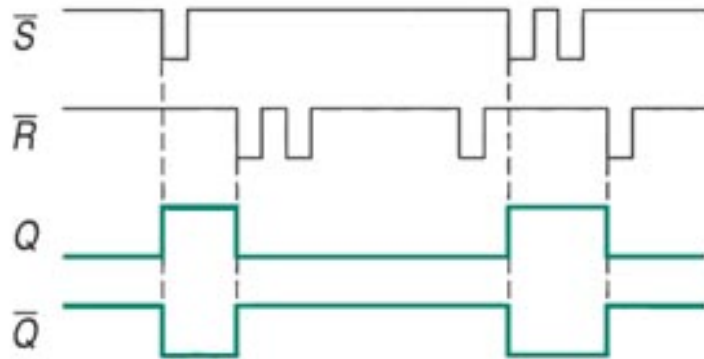
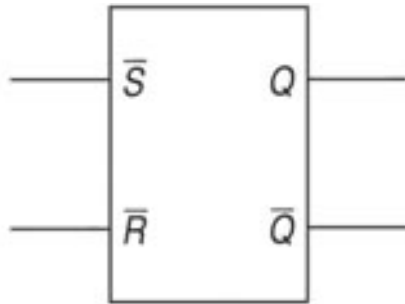
# Class 9

## Sequential Logic: Latch





# SR NAND Latch



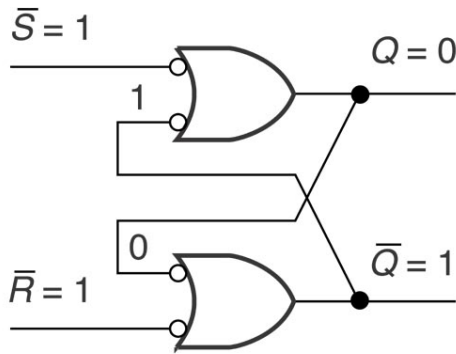
nS	nR	Q <sub>t+1</sub>	nQ <sub>t+1</sub>	Function
0	0	1	1	Forbidden
0	1	1	0	Set
1	0	0	1	Reset
1	1	Q <sub>t</sub>	nQ <sub>t</sub>	No Change

SR NAND Latch Function Table

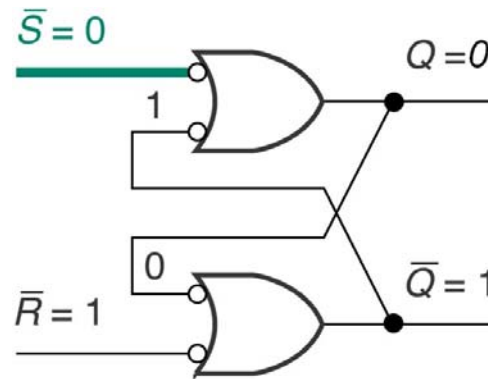


# Reset-to-Set Transition

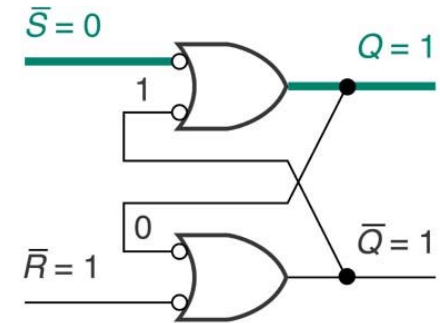
nS	nR	$Q_{t+1}$	$nQ_{t+1}$	Function
0	0	1	1	Forbidden
0	1	1	0	Set
1	0	0	1	Reset
1	1	$Q_t$	$nQ_t$	No Change



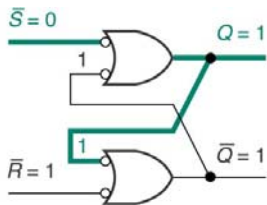
a) Stable in the RESET condition. Set and Reset inputs inactive.



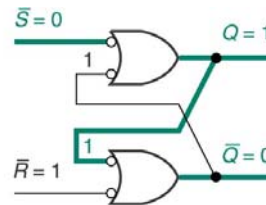
b) Set input activates.



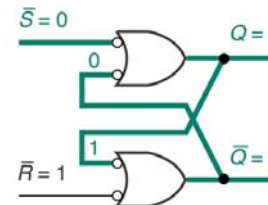
c) Change propagates through upper gate. (Either input LOW makes output HIGH.)



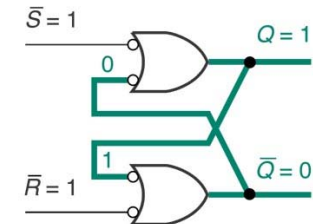
d) HIGH transfers across feedback line to lower gate, removing active input condition.



e) Change propagates through lower gate. (Both inputs HIGH, therefore output LOW.)



f) Feedback transfers LOW to upper gate, completing change to new state.

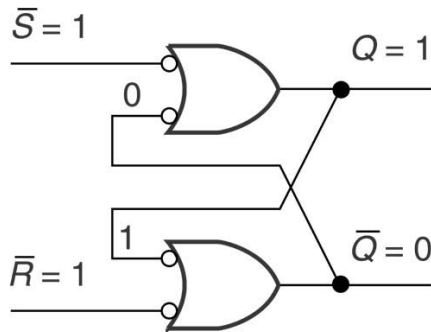


g) S input goes back to inactive state. SET state held by LOW at inner input of upper gate.

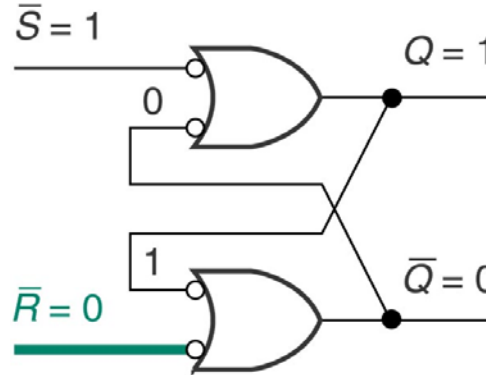


# Set-to-Reset Transition

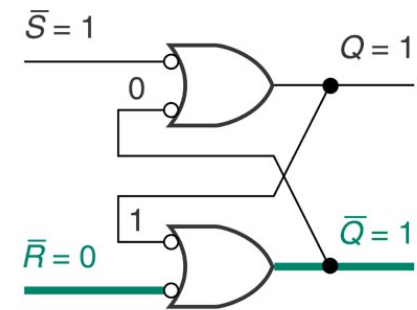
nS	nR	$Q_{t+1}$	$nQ_{t+1}$	Function
0	0	1	1	Forbidden
0	1	1	0	Set
1	0	0	1	Reset
1	1	$Q_t$	$nQ_t$	No Change



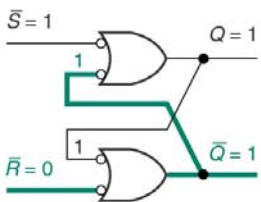
a) Stable in the SET condition. Set and Reset inputs inactive.



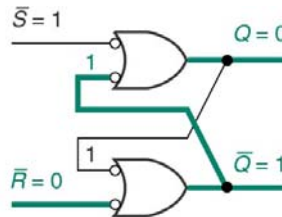
b) Reset input activates.



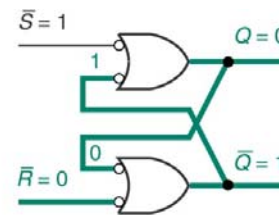
c) Change propagates through lower gate. (Either input LOW makes output HIGH.)



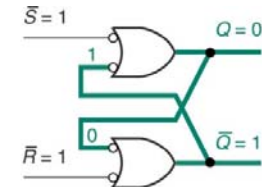
d) HIGH transfers across feedback line to upper gate, removing active input condition.



e) Change propagates through upper gate. (Both inputs HIGH, therefore output LOW.)



f) Feedback line transfers LOW to lower gate, completing transition to RESET state.

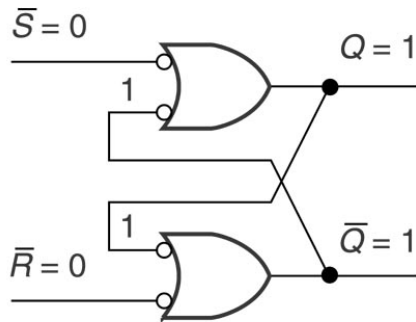


g) R goes back to inactive level. Latch is stable in new state, held by the 0 on inner input of lower gate.

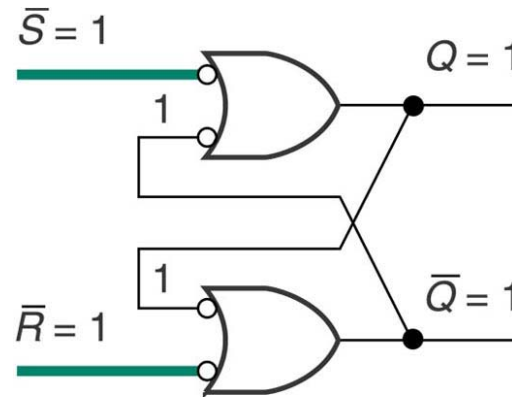


# Transition from Forbidden State

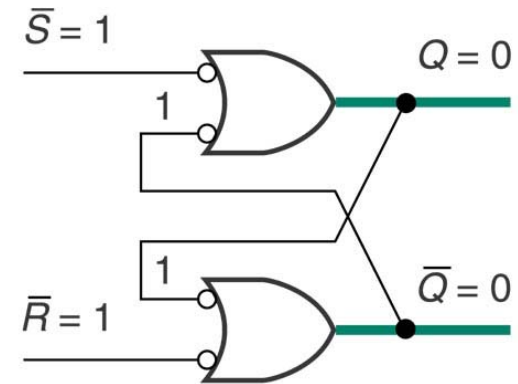
nS	nR	$Q_{t+1}$	$nQ_{t+1}$	Function
0	0	1	1	Forbidden
0	1	1	0	Set
1	0	0	1	Reset
1	1	$Q_t$	$nQ_t$	No Change



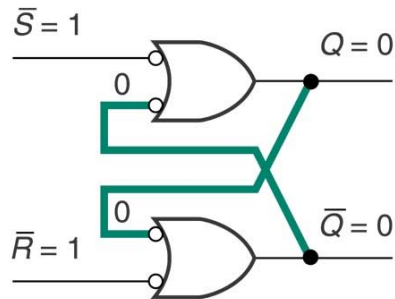
- a) Both Set and Reset inputs active. Either input LOW makes output HIGH. Therefore, both outputs HIGH.



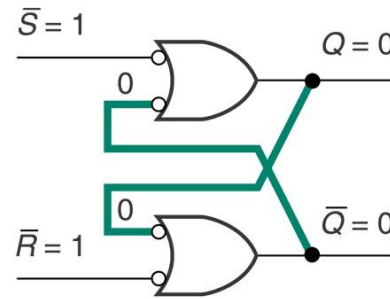
- b) Set and Reset inputs deactivate simultaneously.



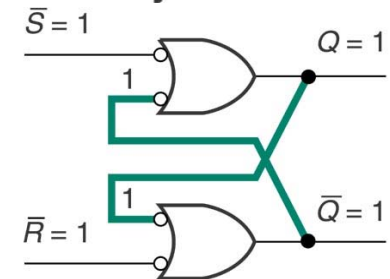
- c) Change propagate through gates simultaneously.



- d) New output levels cross circuit via feedback lines.



- d) New output levels cross circuit via feedback lines.

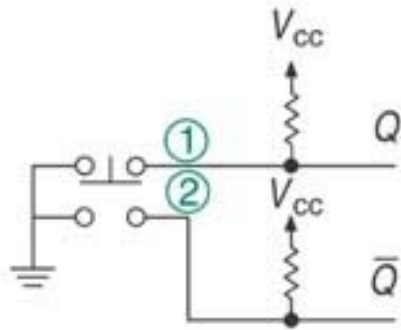


- f) Output logic levels cross circuit via feedback lines. Cycle repeats and circuit oscillates.

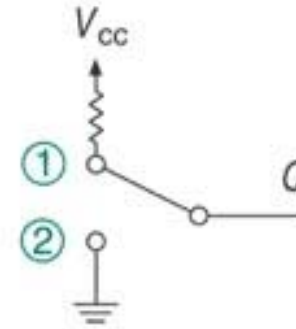


# Switch Bouncing

nS	nR	$Q_{t+1}$	$nQ_{t+1}$	Function
0	0	1	1	Forbidden
0	1	1	0	Set
1	0	0	1	Reset
1	1	$Q_t$	$nQ_t$	No Change



a. Pushbutton



b. Toggle



c. Ideal waveform

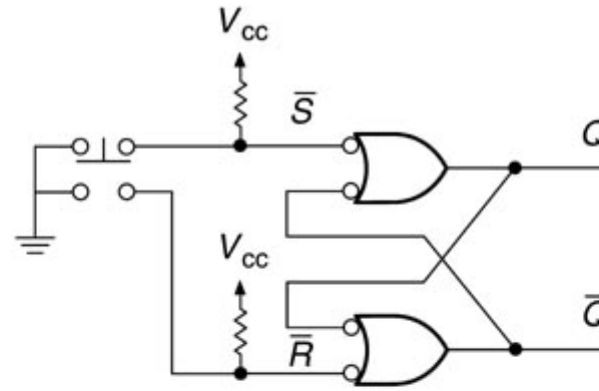


d. Effect of contact bounce

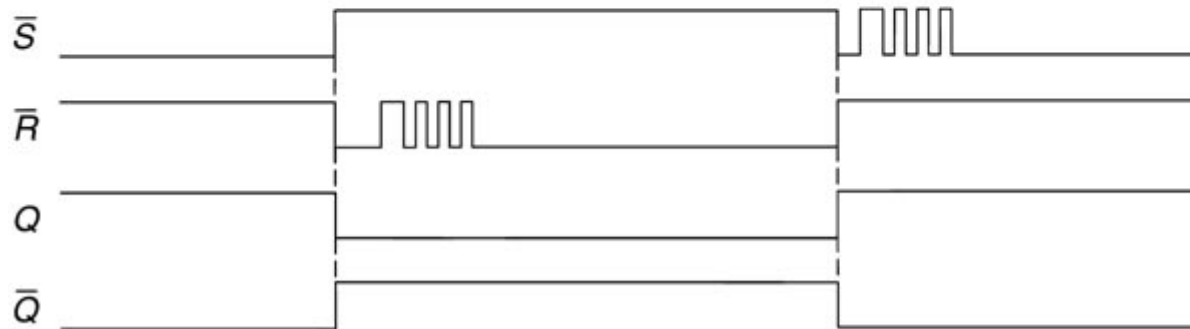


# Switch Debouncing

nS	nR	$Q_{t+1}$	$nQ_{t+1}$	Function
0	0	1	1	Forbidden
0	1	1	0	Set
1	0	0	1	Reset
1	1	$Q_t$	$nQ_t$	No Change



a. Switch debouncer



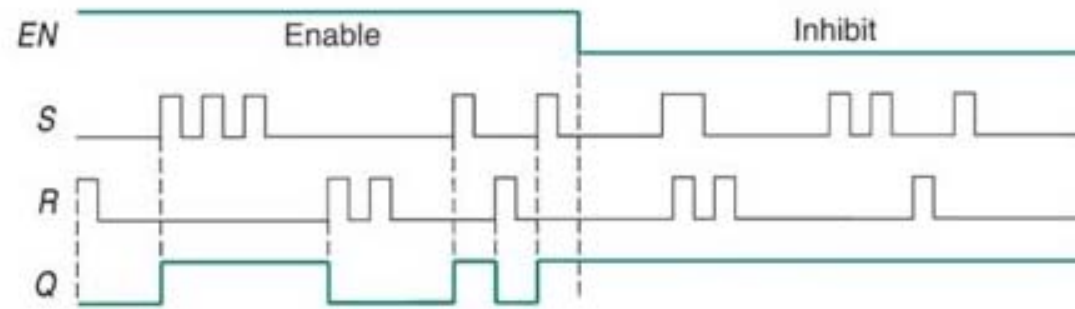
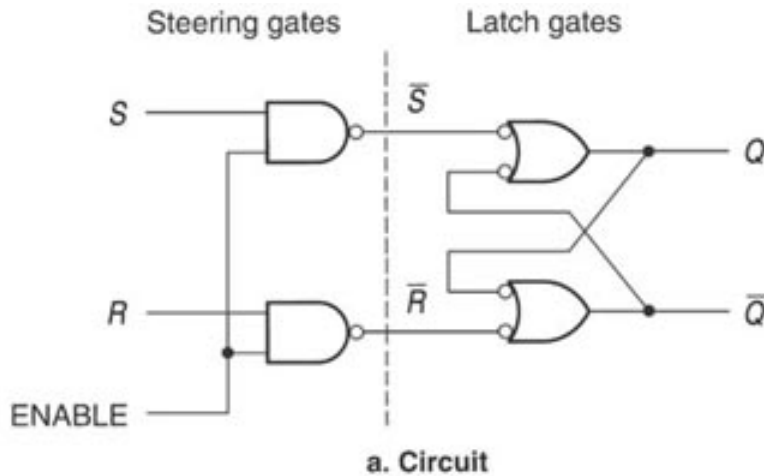
b. Timing diagram



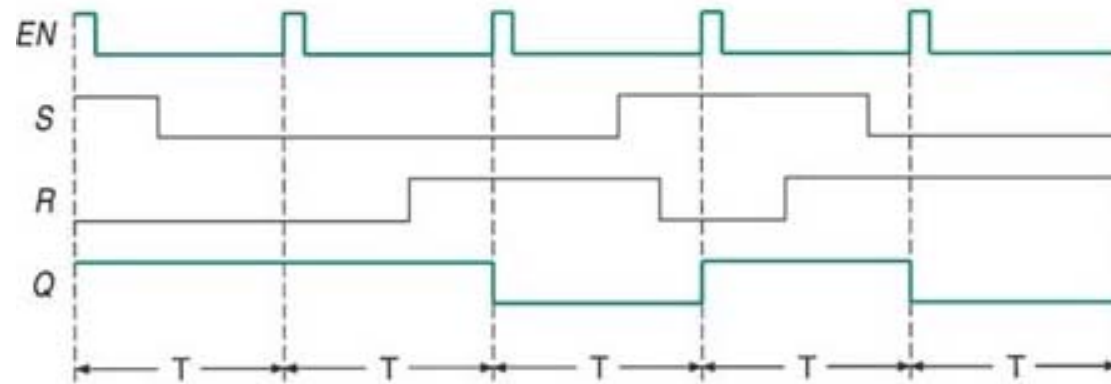
# Gated SR NAND Latch

nS	nR	$Q_{t+1}$	$nQ_{t+1}$	Function
0	0	1	1	Forbidden
0	1	1	0	Set
1	0	0	1	Reset
1	1	$Q_t$	$nQ_t$	No Change

SR NAND Latch Truth Table



b. ENABLE used as an ON/OFF signal



T = equal time interval

c. ENABLE used as a synchronizing signal

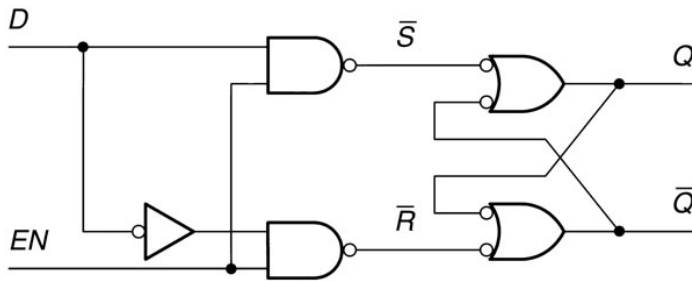
EN	S	R	$Q_{t+1}$	$nQ_{t+1}$	Function
1	0	0	$Q_t$	$nQ_t$	No Change
1	0	1	0	1	Reset
1	1	0	1	0	Set
1	1	1	1	1	Forbidden
0	X	X	$Q_t$	$nQ_t$	Inhibited

Gated NAND Latch Function Table



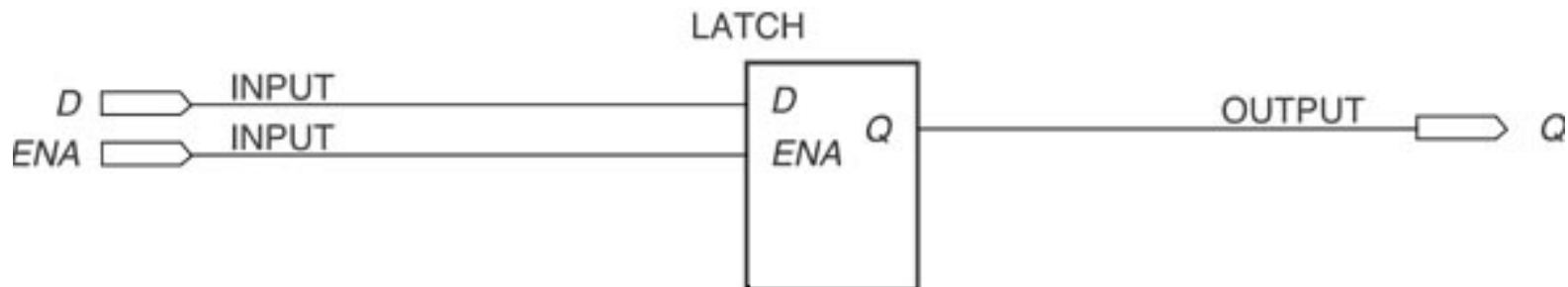


# Gated D Latch (Transparent Latch)



EN	D	$Q_{t+1}$	$nQ_{t+1}$	Function	Comment
1	1	1	0	Set	
1	0	0	1	Reset	Transparent
0	X	$Q_t$	$nQ_t$	No Change	Store

Gated NAND Latch Function Table

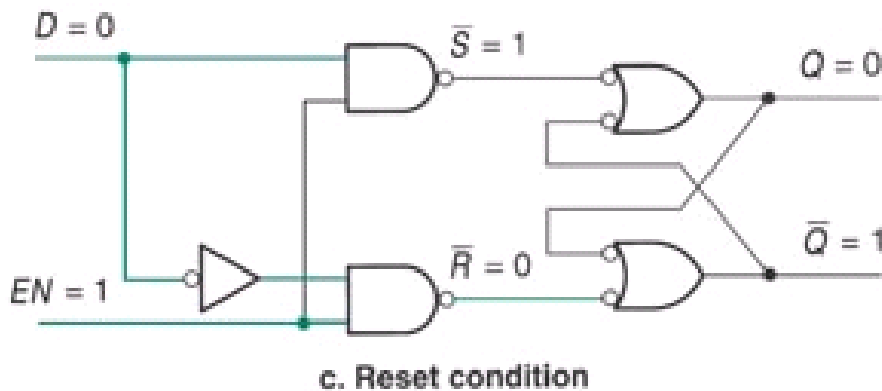
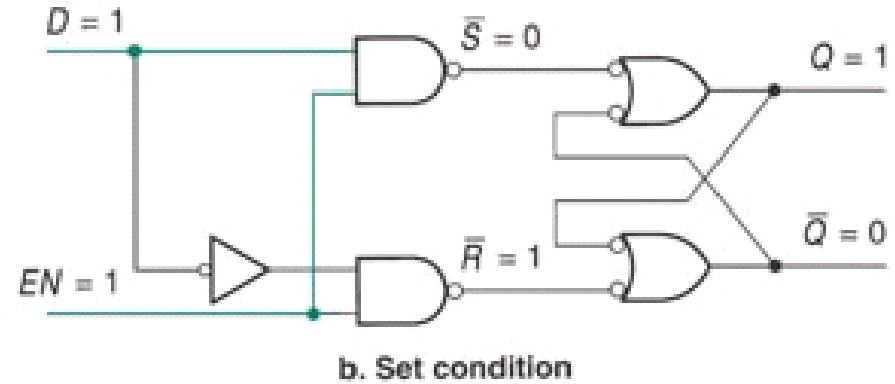
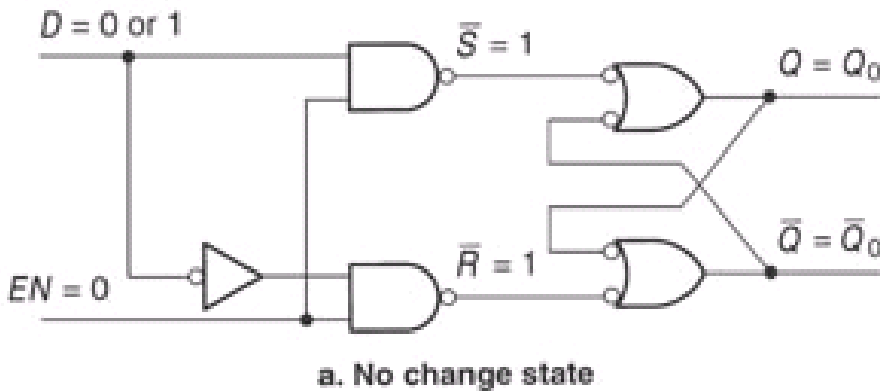




# Gated D Latch (Cont.)

nS	nR	$Q_{t+1}$	$nQ_{t+1}$	Function
0	0	1	1	Forbidden
0	1	1	0	Set
1	0	0	1	Reset
1	1	$Q_t$	$nQ_t$	No Change

SR NAND Latch Truth Table



EN	D	$Q_{t+1}$	$nQ_{t+1}$	Function	Comment
1	1	1	0	Set	
1	0	0	1	Reset	Transparent
0	X	$Q_t$	$nQ_t$	No Change	Store

Gated D Latch Truth Table



# SR NAND Latch with VHDL

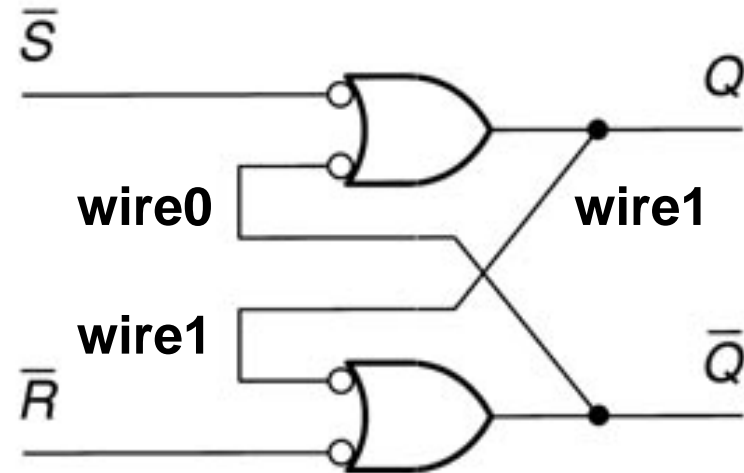
```
LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY nand_latch IS
  PORT
  (
    nS : IN STD_LOGIC;
    nR : IN STD_LOGIC;
    Q : OUT STD_LOGIC;
    nQ : OUT STD_LOGIC
  );
END nand_latch;

ARCHITECTURE bdf_type OF nand_latch IS
  SIGNAL wire0 : STD_LOGIC;
  SIGNAL wire1 : STD_LOGIC;
BEGIN

  Q <= wire1;
  nQ <= wire0;
  wire1 <= NOT(nS AND wire0);
  wire0 <= NOT(wire1 AND nR);

END bdf_type;
```





# Edge-Trigger Event

```

ENTITY counter IS
  PORT
  (
    Q: IN STD_LOGIC; -- reset counter
  );
END counter;

```

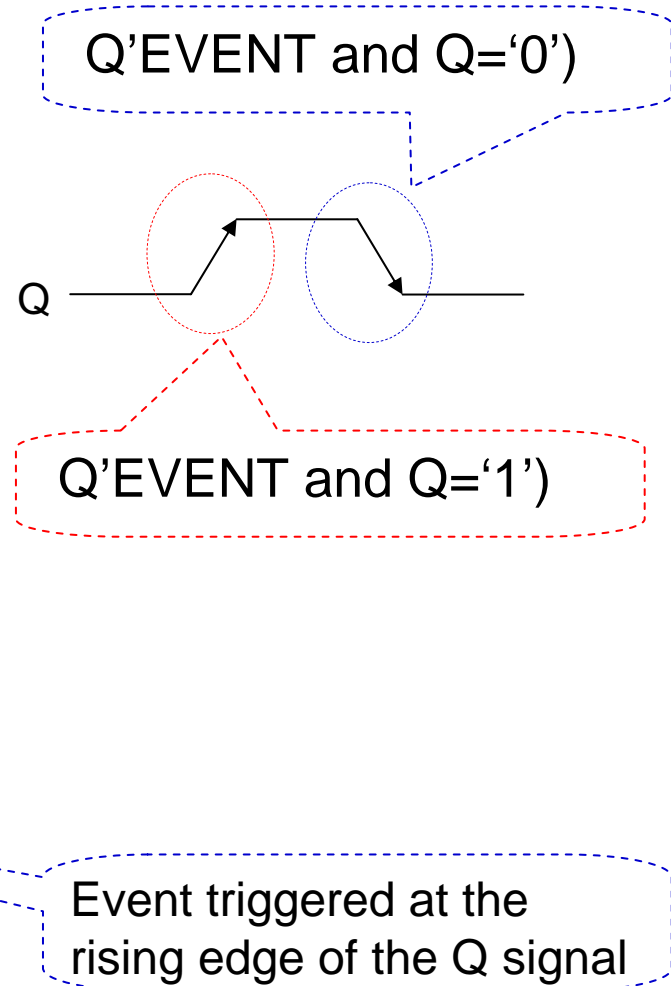
```

ARCHITECTURE a OF counter IS
  SIGNAL cnt: INTEGER RANGE 0 to 10;
BEGIN
  PROCESS(all)
  BEGIN
    IF (cnt > '9') THEN -- reset digit
      cnt <= 0;
    ELSIF (Q'EVENT and Q='1') THEN -- Handle counter
      cnt <= cnt + 1;
    END IF;
  END PROCESS;
END a;

```

Sense everything port and signal

Integer can be used in addition, division, and comparison.





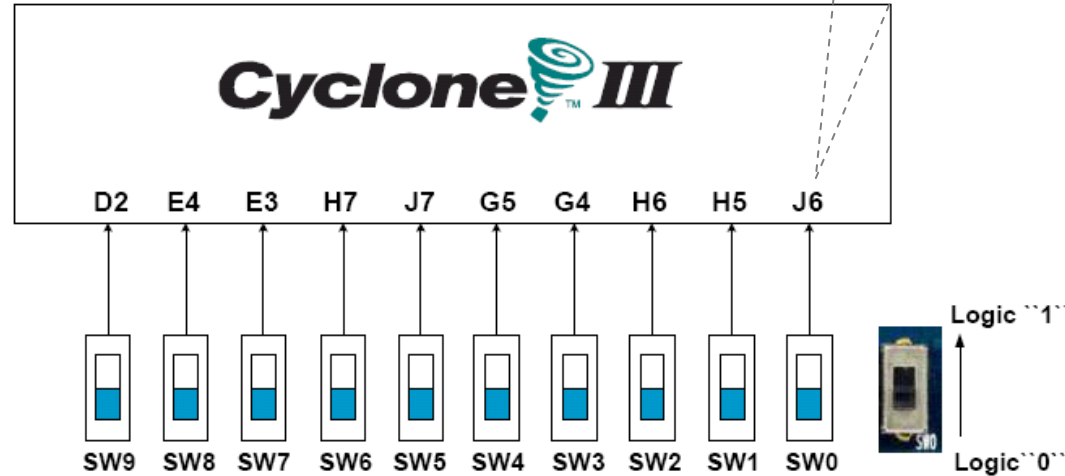
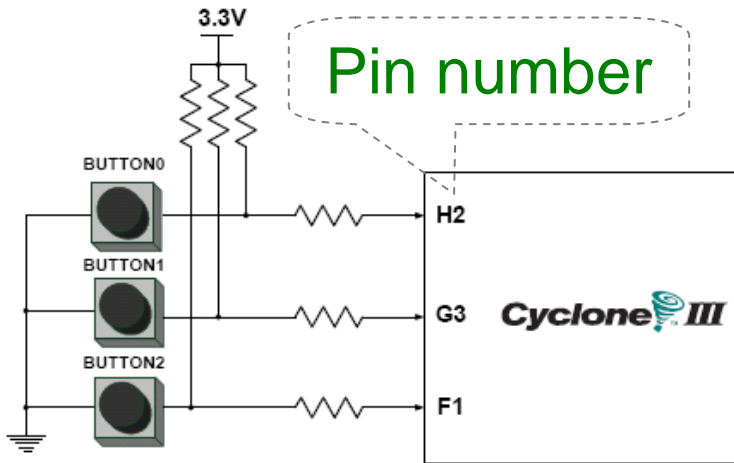
# Lab 9

- **Note: This lab does not allow to use any existing latch modules.**
- Part 1: Gated SR NAND latch
  - Design a gated SR NAND latch.
  - Create a vector waveform file (.vwf) to evaluate the output of the latch.
    - S: count value, binary, simulation period=4us, advanced by 1 every 100ns, **start from 50ns**
    - R: count value, binary, simulation period=4us, advanced by 1 every 200ns, start from 0ns
    - EN: count value, binary, simulation period 4us, advanced by 1 every 2us, start from 0ns
- Part 2: Gated SR NAND latch application: Design the on/off pushbuttons with a counter
  - I/O Functions:
    - PushButton2 (**S**) is to turn on the motor (i.e., LED0: **G**).
    - PushButton1 (**R**) is to turn off motor (i.e., LED0: **G**).
    - PushButton0 is to reset the 2-digit BCD (or decimal) counter to 00.
    - Hex1 and Hex0 shows the value of the 2-digit BCD counter.
    - SW0 is to enable/disable the pushbuttons.
      - When SW0 is Logic-High, enable the pushbuttons . Otherwise, the pushbuttons are disabled.
  - The 2-digit BCD counter is advanced by one whenever the motor is turned ON from OFF (**rising-edge trigger**).
- Report:
  - Write down what you have learned from this lab. (實驗心得)



# Pushbutton and Slide Switches

Pin number



3 Pushbutton switches:  
 Not pressed → Logic High  
 Pressed → Logic Low

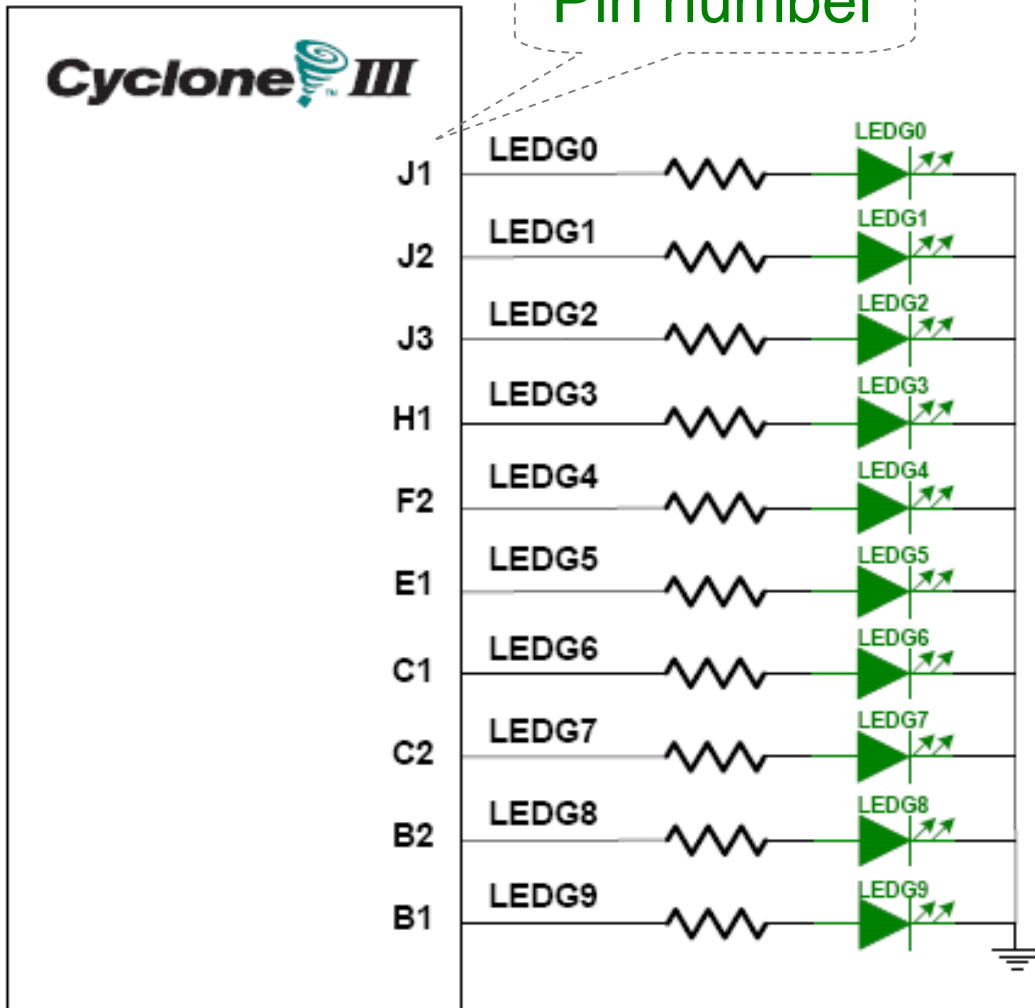
Signal Name	FPGA Pin No.
BUTTON [0]	PIN_ H2
BUTTON [1]	PIN_ G3
BUTTON [2]	PIN_ F1

10 Slide switches (Sliders):  
 Up → Logic High  
 Down → Logic

SW[0]	PIN_J6	SW[5]	PIN_J7
SW[1]	PIN_H5	SW[6]	PIN_H7
SW[2]	PIN_H6	SW[7]	PIN_E3
SW[3]	PIN_G4	SW[8]	PIN_E4
SW[4]	PIN_G5	SW[9]	PIN_D2



# LEDs



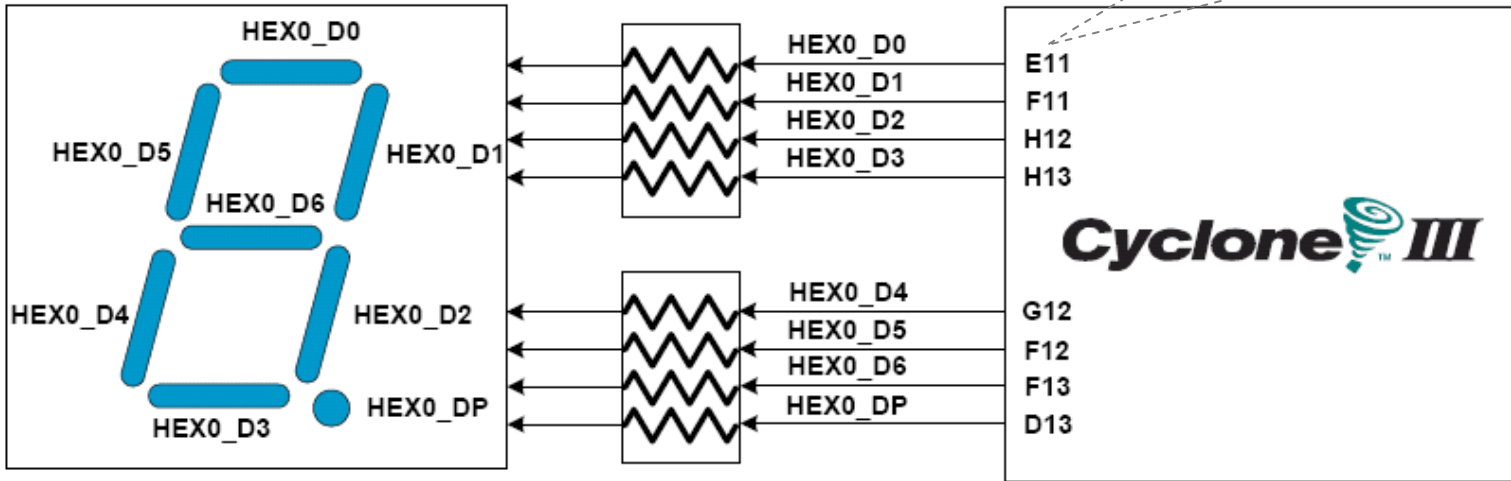
10 LEDs  
 Output high → LED on  
 Output low → LED off

Signal Name	FPGA Pin No.
LEDG[0]	PIN_J1
LEDG[1]	PIN_J2
LEDG[2]	PIN_J3
LEDG[3]	PIN_H1
LEDG[4]	PIN_F2
LEDG[5]	PIN_E1
LEDG[6]	PIN_C1
LEDG[7]	PIN_C2
LEDG[8]	PIN_B2
LEDG[9]	PIN_B1



# 7-Segment Displays

Pin number  
(active-low)



Signal Name	FPGA Pin No.
-------------	--------------

HEX0_D[0]	PIN_E11
HEX0_D[1]	PIN_F11
HEX0_D[2]	PIN_H12
HEX0_D[3]	PIN_H13
HEX0_D[4]	PIN_G12
HEX0_D[5]	PIN_F12
HEX0_D[6]	PIN_F13
HEX0_DP	PIN_D13

HEX1_D[0]	PIN_A13
HEX1_D[1]	PIN_B13
HEX1_D[2]	PIN_C13
HEX1_D[3]	PIN_A14
HEX1_D[4]	PIN_B14
HEX1_D[5]	PIN_E14
HEX1_D[6]	PIN_A15
HEX1_DP	PIN_B15

HEX2_D[0]	PIN_D15
HEX2_D[1]	PIN_A16
HEX2_D[2]	PIN_B16
HEX2_D[3]	PIN_E15
HEX2_D[4]	PIN_A17
HEX2_D[5]	PIN_B17
HEX2_D[6]	PIN_F14
HEX2_DP	PIN_A18

HEX3_D[0]	PIN_B18
HEX3_D[1]	PIN_F15
HEX3_D[2]	PIN_A19
HEX3_D[3]	PIN_B19
HEX3_D[4]	PIN_C19
HEX3_D[5]	PIN_D19
HEX3_D[6]	PIN_G15
HEX3_DP	PIN_G16