# ECE 2214

## Digital Logic Design

## L-21 Flip-Flops

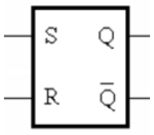
Fall 2022

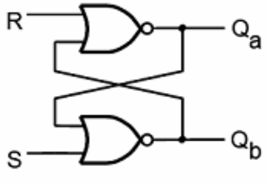
# Outline

- Master-slave D flip-flops
- Edge-triggered D flip-flops
- Additional features of D flip-flops
- T flip-flop
- JK Flip flops
- Flip-flops in VHDL

# Summary of previous lecture

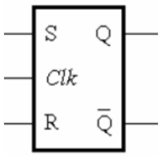
- Basic memory elements
  - Basic SR latch:
    - Can store 1-bit of information but …
    - Limited by asynchronous outputs
    - S = R = 1 can’t occur otherwise Qa ≠ Qb

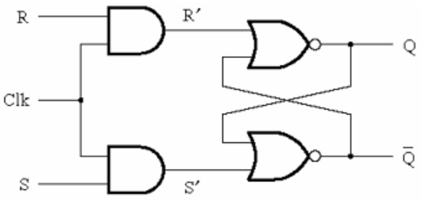




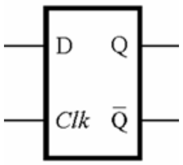
Circuit

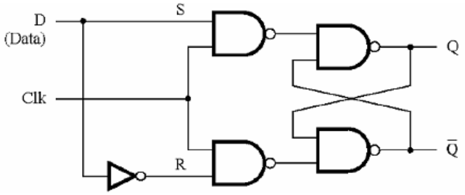
  - Gated SR latch:
    - Include a clock terminal to synchronize the output but…
    - S = R = 1 can’t occur otherwise Qa ≠ Qb
    - Asynchronous output while CLK = 1





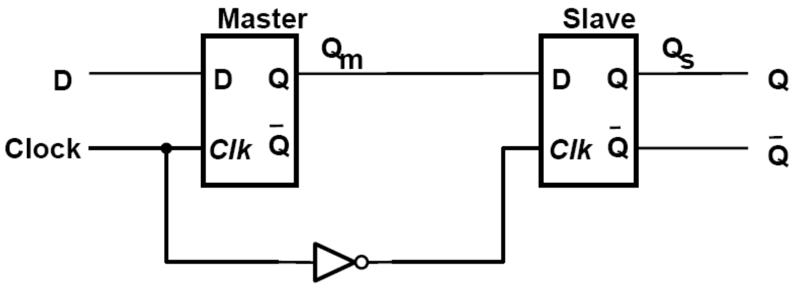
  - Gated D latch:
    - Provide complemented outputs at any time
    - S = R = 1 problem solved with a not gate
    - Asynchronous out while CLK = 1 is not solved



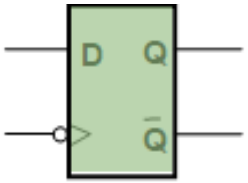


# Master-Slave D flip-flops

- To solve the clock synchronism problem of the latches
  - Need to device circuits that …
    - Can memorize digital information
    - Provide complemented outputs and…
    - Work synchronously with the CLK signal at all time
- Master slave flip-flops were conceived to overcome this limitation!



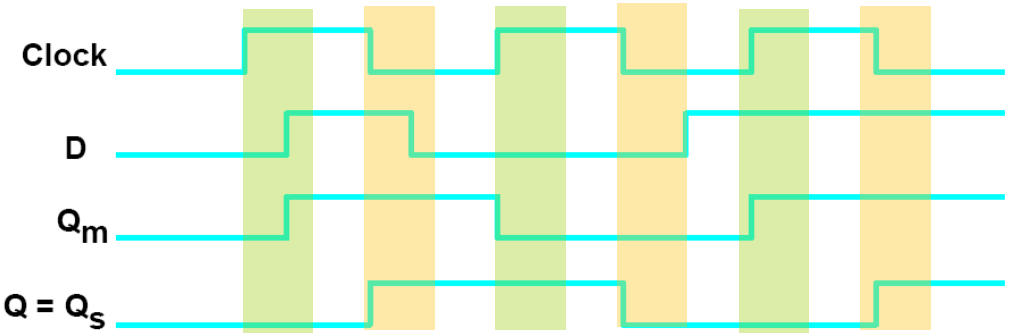
Circuit



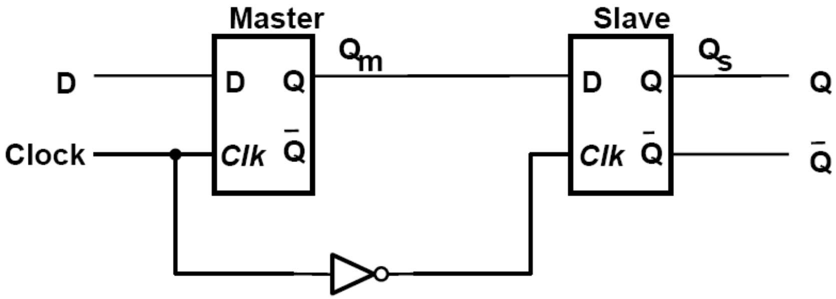
Graphical symbol

  - Two D latches governed by level edge resulting in edge control
    - Can be positive or negative clock edges
      - Figure corresponds to a Master-Slave active with negative clock edge

# Master-Slave D flip-flops



**Timing diagram**



**Circuit**

From Fundamentals of Digital Logic with VHDL Design 2nd Ed. Stephen Brown, Zvonko Vranesic; McGraw-Hill, 2005

- Master latch changes when the clock changes to ‘1’
- Slave changes when clock is ‘0’
- The output of circuit changes with negative clock edge
  - The NOT gate guarantees edge synchronous behavior
  - Provide half of CLK to allow the output change

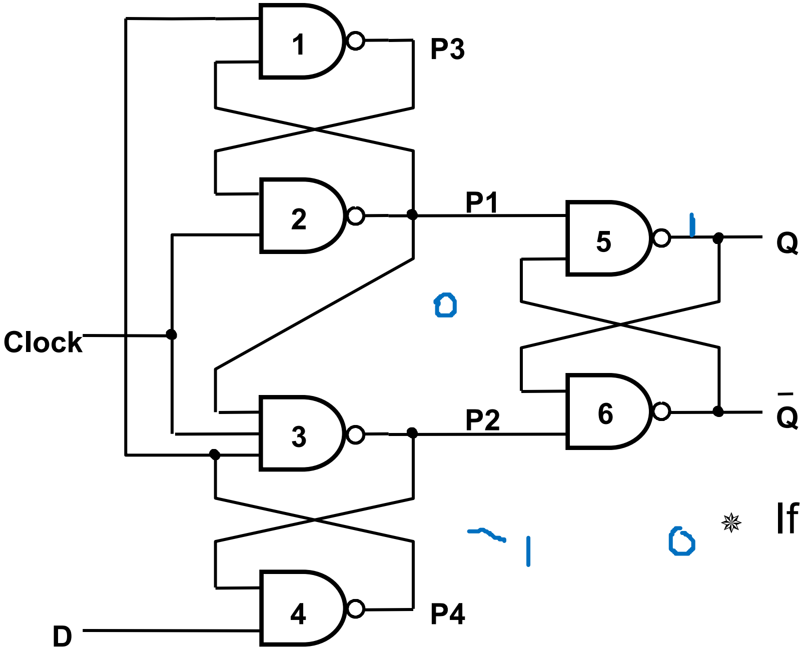
- Flip-flops are **edge sensitive** rather that level sensitive

| D | CLK | $Q_n$ | $\overline{Q}_n$ |
|---|---|---|---|
| 0 | ↓ | 0 | 1 |
| 1 | ↓ | 1 | 0 |
| x | 0 | $Q_{n-1}$ | $\overline{Q}_{n-1}$ |
| x | 1 | $Q_{n-1}$ | $\overline{Q}_{n-1}$ |

Characteristic table

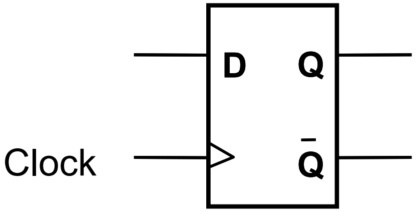
# Positive edge D flip-flops

- Similar to a Master-slave flip-flop but with additional logic to ensure the output of the flip-flop will change with the positive edge of the clock signal
  - Show an optimized NAND gate implementation



Circuit

From Fundamentals of Digital Logic with VHDL Design 2nd Ed.
Stephen Brown, Zvonko Vranesic; McGraw-Hill, 2005



Graphical symbol

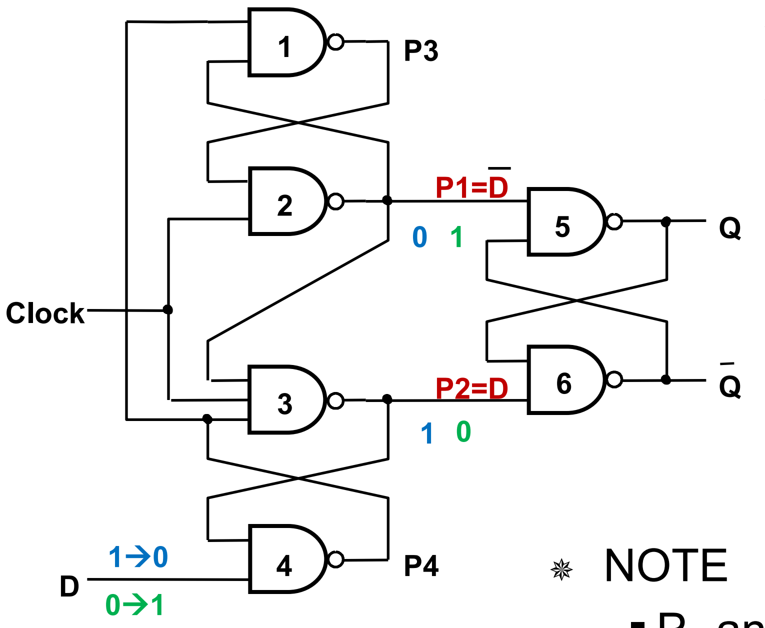
| D | CLK | $Q_n$ | $\overline{Q}_n$ |
|---|---|---|---|
| 0 | ↑ | 0 | 1 |
| 1 | ↑ | 1 | 0 |
| x | 0 | $Q_{n-1}$ | $\overline{Q}_{n-1}$ |
| x | 1 | $Q_{n-1}$ | $\overline{Q}_{n-1}$ |

Characteristic table

- If CLK = 0, $P_1 = P_2 = 1$, $P_4 = \overline{D}$, & $P_3 = D$,
  - $Q_n = Q_{n-1}$ (output unchanged)
- If CLK= 1, $P_1 = \overline{D}$ & $P_2 = D \rightarrow Q_n = D$

- After CLK changes to 1, any change in D will not affect Q. Why?

# Edge-triggered D flip-flops



Circuit

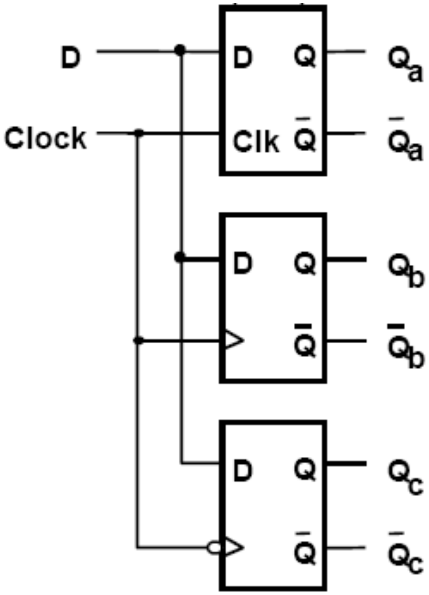
- After CLK changes to 1, $P_1 = \overline{D}$ and $P_2 = D$
- Any change in D will not affect Q because:
  - If before: D = 1 ➔$P_1$ = 0 and $P_2$ = 1

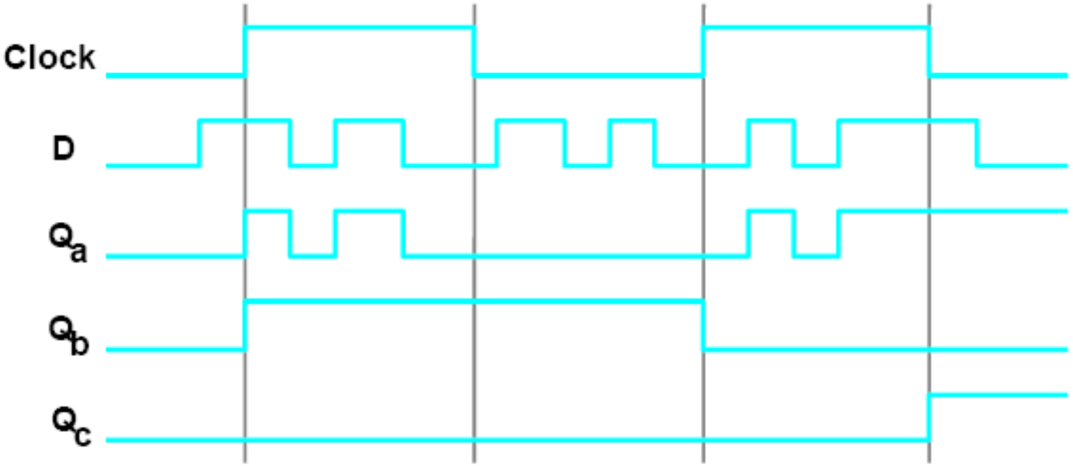
    D→0 while $P_1$=0, $P_2$ = 1 regardless of D

- NOTE
  - $P_3$ and $P_4$ must be stable when CLK ↑
  - $T_{SU}$ (stability of D value before CLK changes) of the F-F configuration is determined by the delay introduced by gates 4 and 1 when the signal propagates from input D to $P_3$
    - Longest data path
  - $T_H$ (hold time) is determined by gate 3
    - Once $P_2$ is stable, a change in D no longer matters

# Level sensitive vs. edge-triggered D flip-flops

- A comparison between the two types of flip-flops and the D latch…



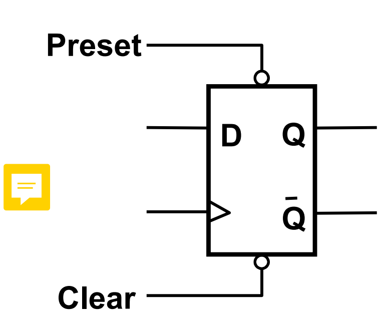


From Fundamentals of Digital Logic with VHDL Design 2nd Ed. Stephen Brown, Zvonko Vranesic; McGraw-Hill, 2005

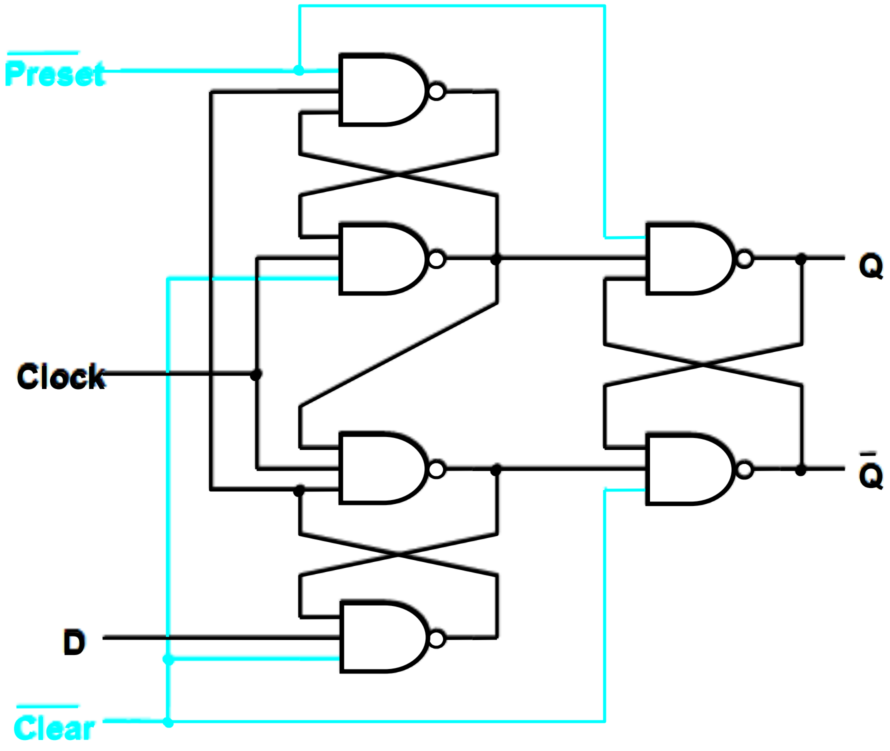
- Observation
  - Any input change in D, is reflected in the output $Q_a$ because it is level sensitive
  - The output $Q_b$ will reflect the state of input D with positive clock edge
  - The output $Q_c$ will reflect the state of input D with negative clock edge

# Clear and preset features of D flip-flops

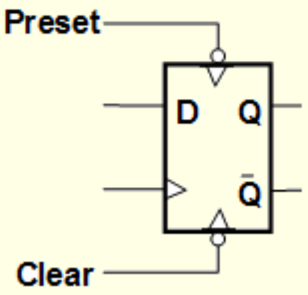
- The clear/preset input is used to set/reset the output of the flip-flop
  - Activated preset → sets Q = 1
  - Activated clear → reset Q = 0
- Feature is important for many applications i.e. counters requiring resetting/presetting values



| D | CL | Cr | Pr | $Q_n$ | $\overline{Q}_n$ |
|---|---|---|---|---|---|
| 0 | ↑ | 1 | 1 | 0 | 1 |
| 1 | ↑ | 1 | 1 | 1 | 0 |
| x | D | 1 | 1 | $Q_{n-}$ | $\overline{Q}_{n-1}$ |
| x | D | 1 | 1 | $Q_{n-}$ | $\overline{Q}_{n-1}$ |
| x | x | 0 | 1 | 0 | 1 |
| x | x | 1 | 0 | 1 | 0 |



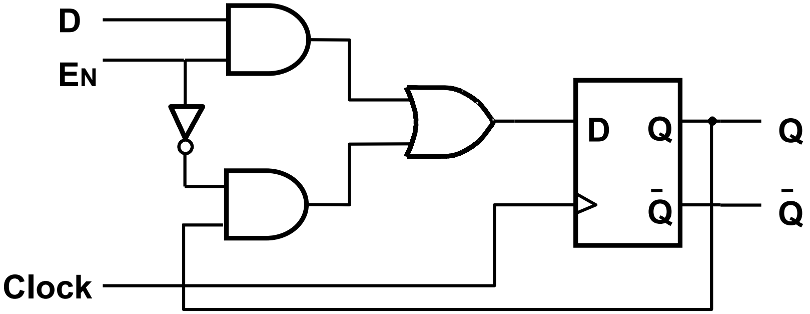
- Observation
  - Cr and Pr are active low
  - Cr and Pr can't be active at the same time
  - Cr and Pr are level sensitive
  - Cr-Pr can be converted to edge sensitive

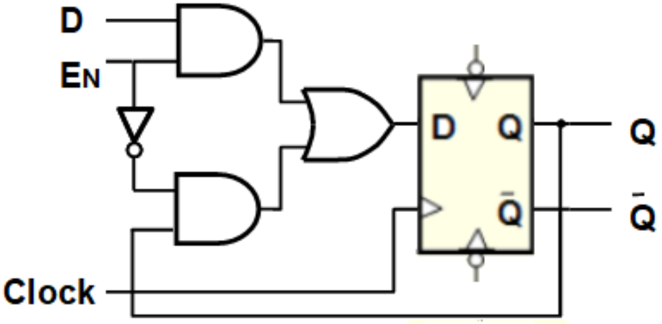


| D | CL | Cr | Pr | $Q_n$ | $\overline{Q}_n$ |
|---|---|---|---|---|---|
| 0 | ↑ | 1 | 1 | 0 | 1 |
| 1 | ↑ | 1 | 1 | 1 | 0 |
| x | D | 1 | 1 | $Q_{n-}$ | $\overline{Q}_{n-1}$ |
| x | D | 1 | 1 | $Q_{n-}$ | $\overline{Q}_{n-1}$ |
| x | ↑ | 0 | 1 | 0 | 1 |
| x | ↑ | 1 | 0 | 1 | 0 |

# D flip-flops with clock enable

- Clock enable input allows preserving certain output state for a period of time determined by the enable pin

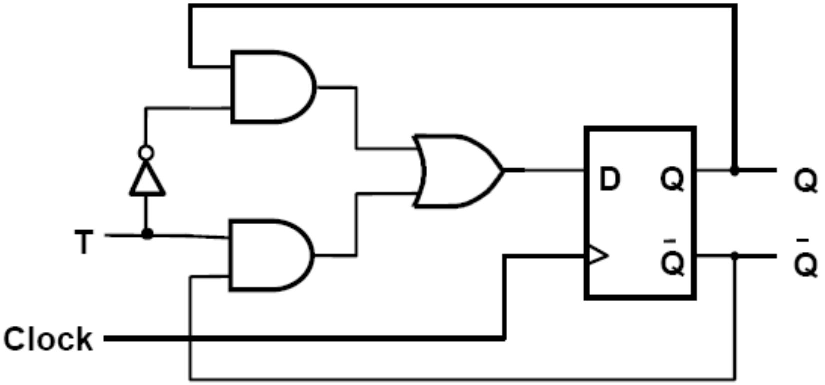




- Observation
  - The circuit does not control the clock
  - D input is controlled instead
  - $E_N = 1$, $Q_{N+1} = D$ with CLK ↑
  - $E_N = 0$, $Q_{N+1} = Q_N$ when CLK ↑ regardless of the value of D
    - Achieved Q with a feedback of $Q_N$
  - Both Cr and Pr inputs are applicable here (synchronous or asynchronous)

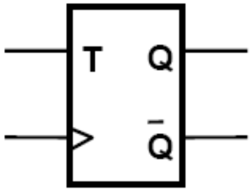
# T flip-flop

- A special type of flip-flop that is also made by adding logic to a D flip-flop
- It toggles the outputs with the active edge of the clock



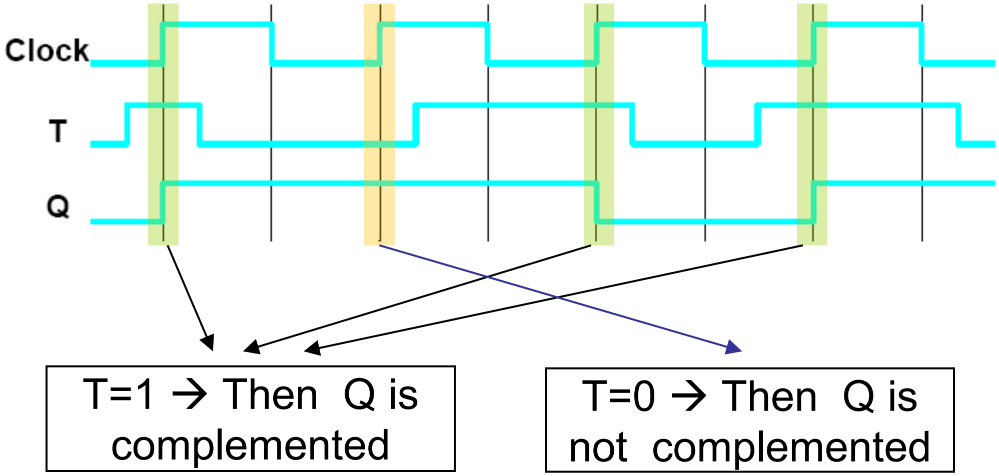
| T | Q (t + 1) |
|---|---|
| 0 | Q (t) |
| 1 | $\overline{Q}$ (t) |

Truth table



Graphical symbol

- Observation
  - Output Q is toggled with T = 1 and a positive edge of the clock
  - Output Q does not change with T = 0
  - Both Cr and Pr inputs are applicable here too (synchronous or asynchronous)



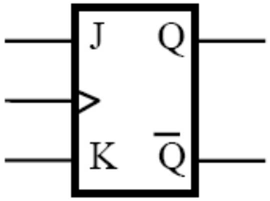
T=1 → Then Q is complemented

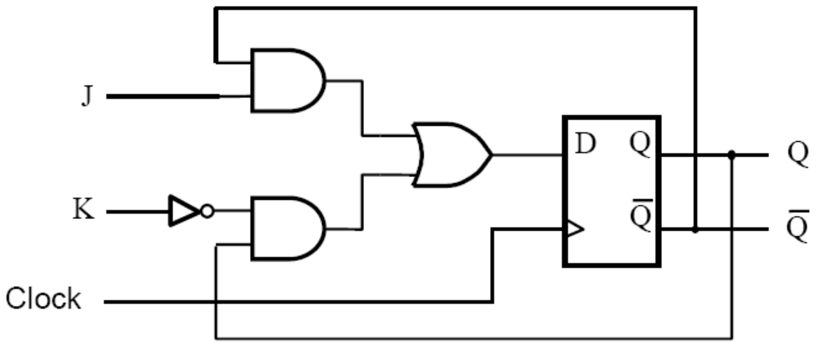
T=0 → Then Q is not complemented

From Fundamentals of Digital Logic with VHDL Design 2nd Ed. Stephen Brown, Zvonko Vranesic; McGraw-Hill, 2005

# J-K flip-flop

- Similar to T F-F but they have 2 control inputs J and K
- Combine the characteristics of the D and T F-Fs
- Edge sensitive device
- The equation describing D is: $D = J\overline{Q} + \overline{K}Q$





| CLK | J | K | $Q_n$ | $\overline{Q}_n$ |
|---|---|---|---|---|
| ↑ | 0 | 0 | $Q_{n-1}$ | $\overline{Q}_{n-1}$ |
| ↑ | 0 | 1 | 0 | 1 |
| ↑ | 1 | 0 | 1 | 0 |
| ↑ | 1 | 1 | $\overline{Q}_{n-1}$ | $Q_{n-1}$ |
| 0 | x | x | $Q_{n-1}$ | $\overline{Q}_{n-1}$ |
| 1 | x | x | $Q_{n-1}$ | $\overline{Q}_{n-1}$ |

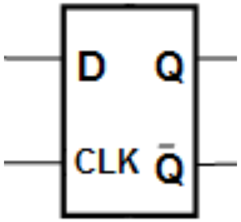
⇨ D-FF

⇨ T-FF

- J-K F-F is versatile
  - Behaves like a D-FF when J = D and $K = \overline{D}$
  - Behaves as a T F-F if J=K=T
    - It toggles the output in every CLK ↑

# Latches and flip-flops in VHDL

- VHDL implementation of Latches and F-F is simple
- Can be implemented with sequential assignment
  - **IF** and **Case** statements
- Examples of implementations:
- Example 1: Code for a gated D latch



From Fundamentals of Digital Logic with VHDL Design 2nd Ed. Stephen Brown, Zvonko Vranesic; McGraw-Hill, 2005

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

ENTITY GDlatch IS
    PORT (  D, Clock  : IN     STD_LOGIC ;
            Q         : OUT    STD_LOGIC) ;
END GDlatch ;

ARCHITECTURE Behavior OF GDlatch IS
BEGIN
    PROCESS ( Clock )
    BEGIN
        IF Clock = '1' THEN
            Q <= D ;
        END IF ;
    END PROCESS ;
END Behavior ;
```

- Notice that when Clock = 1 , any change in D will be reflected in Q
- When clock = 0, Q will not change with a change in D

# Flip-flops in VHDL

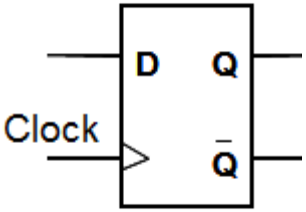
- EXAMPLE 2: Code for a D flip-flop

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

ENTITY flipflop IS
    PORT (    D, Clock   : IN      STD_LOGIC ;
              Q          : OUT     STD_LOGIC) ;
END flipflop ;

ARCHITECTURE Behavior OF flipflop IS
BEGIN
    PROCESS ( Clock, D )
    BEGIN
        IF rising_edge( Clock) THEN
            Q <= D ;
        END IF ;
    END PROCESS ;
END Behavior ;
```

From Fundamentals of Digital Logic with VHDL Design 2nd Ed. Stephen Brown, Zvonko Vranesic; McGraw-Hill, 2005



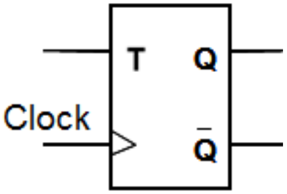
- Rising_edge can also be implemented with (Clock’EVENT AND Clock = ‘1’ )
- Clock’EVENT is a VHDL construct called attribute
- It is a property of the object SIGNAL (Clock in this case)
- Clock’EVENT refers to any change in clock signal
- Clock = ‘1’ refers to positive clock edge
- Falling_edge(Clock) or
  Clock’EVENT AND Clock = ‘0’
  refers to negative clock edge

# Flip-flops in VHDL

- EXAMPLE 3: Code for a T flip-flop

```
LIBRARY ieee ;
USE ieee.std_logic_1164.all ;
ENTITY flipflop IS
      PORT (    T, Clock   : IN      STD_LOGIC ;
                Q, Qn      : OUT   STD_LOGIC) ;
END flipflop ;
ARCHITECTURE Behavior OF flipflop IS
SIGNAL  Qt :      STD_LOGIC;
BEGIN
   PROCESS ( Clock,T )
    BEGIN
         IF rising_edge(Clock) AND T='1' THEN
           Qt <= NOT(Qt) ;
         END IF ;
    END PROCESS ;
       Q <= Qt;
       Qn<=NOT(Qt);
END Behavior ;
```

From Fundamentals of Digital Logic with VHDL Design 2nd Ed. Stephen Brown, Zvonko Vranesic; McGraw-Hill, 2005

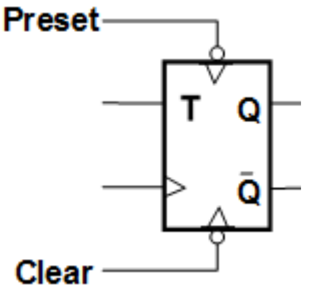


# Flip-flops in VHDL

- EXAMPLE 4: Code for a T flip-flop with synchronous clear and preset

```
LIBRARY ieee ;
USE ieee.std_logic_1164.all ;
ENTITY Tflipflop IS
      PORT (T, Cr,Pr,clk  : IN      STD_LOGIC ;
                Q,  Qn          : OUT   STD_LOGIC) ;
END Tflipflop ;
ARCHITECTURE Behavior OF Tflipflop IS
SIGNAL  Qt :       STD_LOGIC;
BEGIN
    PROCESS ( clk,T, Cr, Pr )
    BEGIN
       IF rising_edge(clk) THEN
          IF (Cr XOR Pr) = '1' THEN    Qt <= Cr;
          ELSE Qt <=  Qt XOR T;
          END IF ;
       END IF ;
    END PROCESS ;
    Q <= Qt; Qn<=NOT(Qt);
 END Behavior ;
```

| T | clk | Cr | Pr | $Q_n$ | $\overline{Q}_n$ |
|---|---|---|---|---|---|
| 0 | ↑ | 1 | 1 | $Q_{n-1}$ | $\overline{Q}_{n-1}$ |
| 1 | ↑ | 1 | 1 | $\overline{Q}_{n-1}$ | $Q_{n-1}$ |
| x | 0 | 1 | 1 | $Q_{n-1}$ | $\overline{Q}_{n-1}$ |
| x | 1 | 1 | 1 | $Q_{n-1}$ | $\overline{Q}_{n-1}$ |
| x | ↑ | 0 | 1 | 0 | 1 |
| x | ↑ | 1 | 0 | 1 | 0 |



# Summary

- Introduced
  - Master-slave and Edge-triggered D flip-flops
  - Additional features of D flip-flops: ***Clear***, ***Preset*** and ***Enable***
  - T and J-K flip-flops
- Revised VHDL implementation of flip-flops
- Recommended study from the textbook
  1. Study solved problems 7.13 & 7.14
  2. Answer problems 7.1- 7.12

# Reference

- Fundamentals of Digital Logic with VHDL Design 2nd Edition Stephen Brown, Zvonko Vranesic; McGraw-Hill, 2005. Chapter 7, pp. 391-400
- Digital Design; Principles and Practices. Fourth Edition. John F. Wakerly, Prentice Hall, 2006. ISBN 0-13-186389-4. Chapter 7, pp. 532-542

- Next lecture:
  - Registers
    - Fundamentals of Digital Logic with VHDL Design 3/e, Stephen Brown, Zvonko Vranesic; McGraw-Hill, 2009. Chapter 7, pp. 398-406
    - Digital Design; Principles and Practices. Fourth Edition. John F. Wakerly, Prentice Hall, 2006. ISBN 0-13-186389-4. Chapter 7, pp. 710-720, 727-752