جامعة بيرزيت

BIRZEIT UNIVERSITY

ANSWER BOOKLET

Student: Digital Number 11

Course: Department: .... Number: ....

Division: "ch5" Instructor: ....

Date: ....

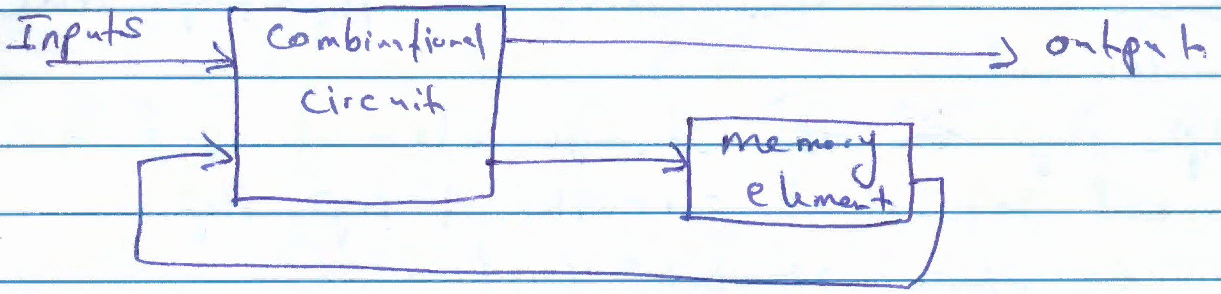
Day Month Year

For Instructor's Use

Question	Grade
1	
2	
3	
4	
5	
6	
7	
8	
9	
10	
11	
12	
Total	

Ⓐ Synchronous Sequential Logic

Ⓑ Sequential Circuit



- in combinational circuits, the outputs are entirely dependent on the current inputs.
- in sequential circuits, the outputs are function of the current inputs and the present state of the memory element (feedback.)

- also the next state of the ~~m~~storage element is a function of the external inputs and current state.

Ⓐ Two main types of sequential circuits (the classification depends on the timing of their signals).

1) synchronous sequential circuits: the behaviour can be defined from the knowledge of its signals at discrete instants of time

2) asynchronous sequential circuit: depends upon the input signals at any instant of time and the order of the inputs.

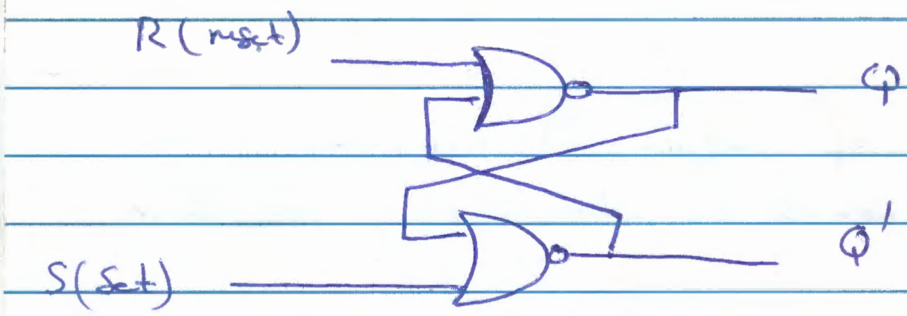
Ⓐ clocked sequential circuits

LATCHES

- Basic circuits from which all flip-flops are constructed
- flip_flop is ~~the~~ a storage element used in clocked sequential circuit. A flip_flop is able to store one bit of information.

- The major differences among various types of flip_flops (latches) are in the number of inputs they posses and in the manner in which inputs affect the binary state.

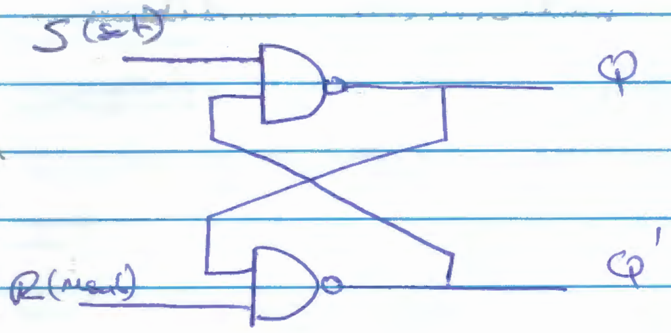
SR Latch (Set - Reset) Latch, using Couple NOR gates



(Stable state)

S	R	Q	Q'	
1	0	1	0	(set state)
0	0	1	0	after S=1, R=0
0	1	0	1	
0	0	0	1	after S=0, R=1
1	1	0	0	(undefined state or Indeterminate).

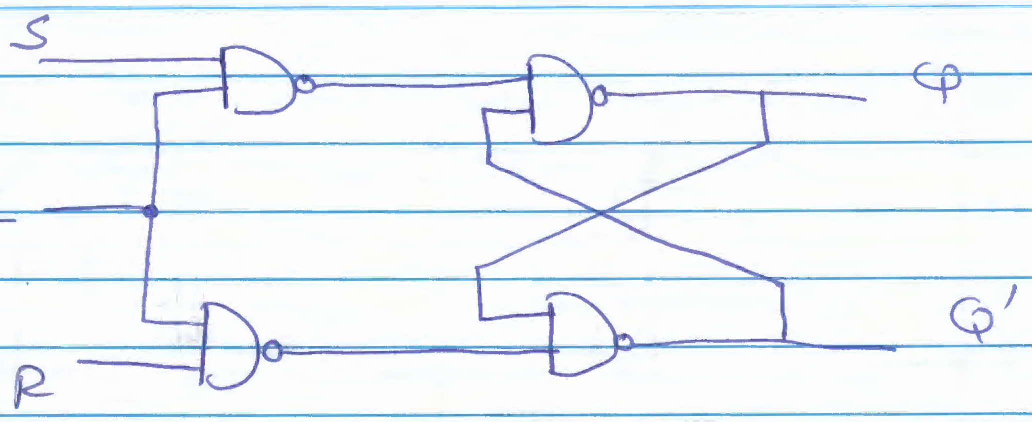
⊛ SR Latch using Coupl Nand gates:-



S	R	Q	Q'	
1	0	0	1	(set complement)
1	1	0	1	(after S=1, R=0)
0	1	1	0	
1	1	1	0	(after S=0, R=1)
0	0	1	1	(undefined)

(Sometimes refered as S'-R' latch)
because S=0 set Q=1
R=0 reset Q=0

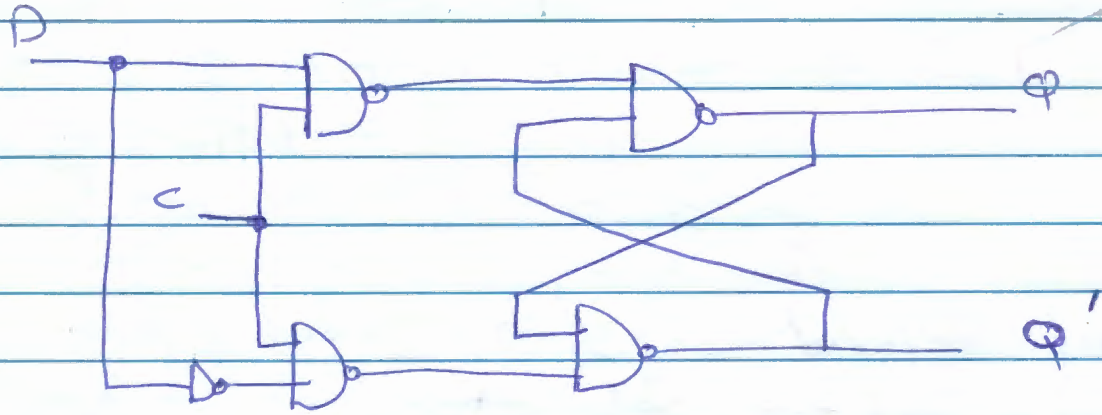
⊛ SR Latch with Control Input



C	S	R	Next state
0	X	X	No change
1	0	0	No change
1	0	1	Q=0 (Reset)
1	1	0	Q=1 (Set)
1	1	1	undefined

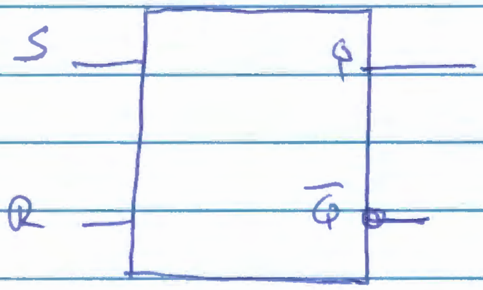
* D - Latch

- one way to eliminate the undesirable condition of the undefined state is SR latch is to ensure that inputs S and R are never equal to 1



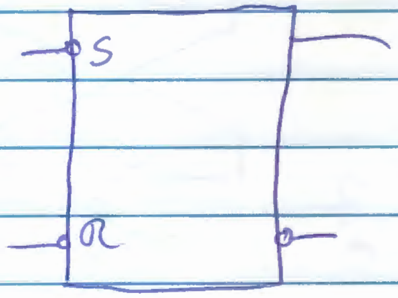
C	D	Next state of Q
0	X	No Change
1	0	Q = 0 (Reset)
1	1	Q = 1 (Set)

(transparent latch = D latch)



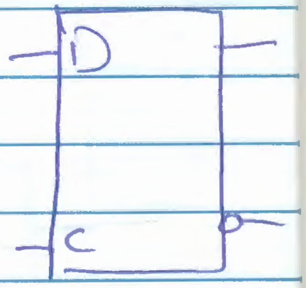
NOR

SR



NAND

$\overline{SR}$



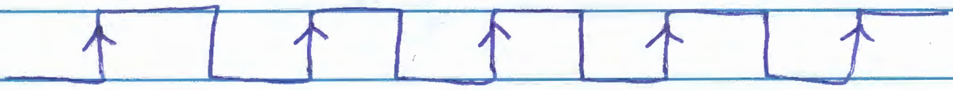
D

⊛ Flip-Flops:

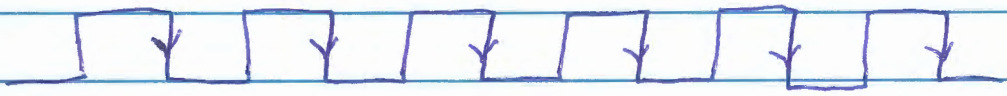
- Latch without control input, always respond to the change in the inputs.
- ~~Both~~ Latch with control input, responds to the change of inputs (eg S & R) only when the control is in the positive level

- unpredictable situation may occur since the state of the latches may keep changing for as long as the clock pulse stays in the active level.

⇒ Flip flop circuits are constructed in such a way as to make them operate properly when they are connected to a sequential circuit that employs a common clock.

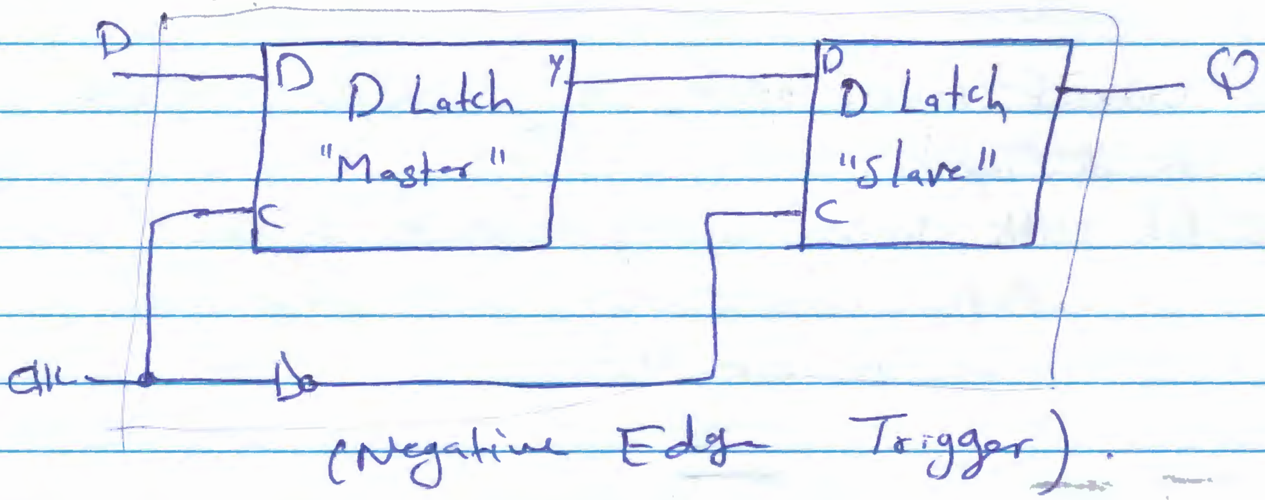


positive edge response (+ve edge trigger)

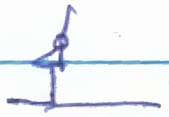


Negative edge response (-ve edge triggor)

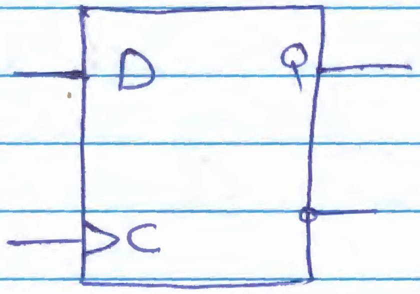
④ Master-Slave D-FF

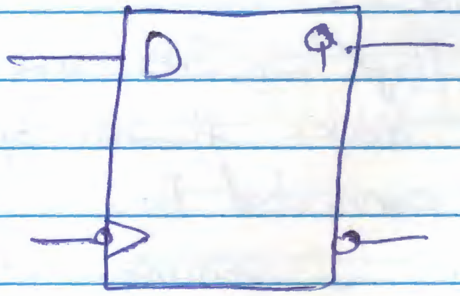


(Negative Edge Trigger).

 +ve Edge Triggered

⑤ D-type +ve-Edge-Triggered FF.





+ve-edge Triggered symbol

-ve-Edge Triggerd symbol

Types of FF

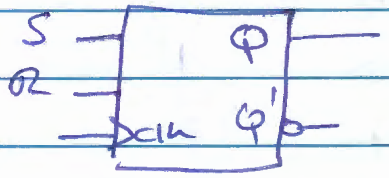
① SR

S	R	Q	Q'
1	0	1	0
0	0	1	0
0	1	0	1
0	0	0	1
1	1	undefined	

Next State

⇒ S	R	Q(t+1)
0	0	Q(t)
0	1	0
1	0	1
1	1	Undefined

characterestic Table



② ~~D FF~~

⇒ S	R	Q(t)	Q(t+1)
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	X
1	1	1	X

③ JK FF

J	K	Q(t+1)	
0	0	Q(t)	same as SR FF
0	1	0	〃 〃 〃 FF
1	0	1	〃 〃 〃 〃
1	1	Q'(t)	New feature

⟹

J	K	Q(t)	Q(t+1)	S	R	D
0	0	0	0	0	X	0
0	0	1	1	X	0	1
0	1	0	0	0	X	0
0	1	1	0	0	1	0
1	0	0	1	1	0	1
1	0	1	1	X	0	1
1	1	0	1	1	0	1
1	1	1	0	0	1	0

- implement JK using SR FF

JK \ Q	0	1
00	x	X
01	x	1
10	1	X
11	1	1

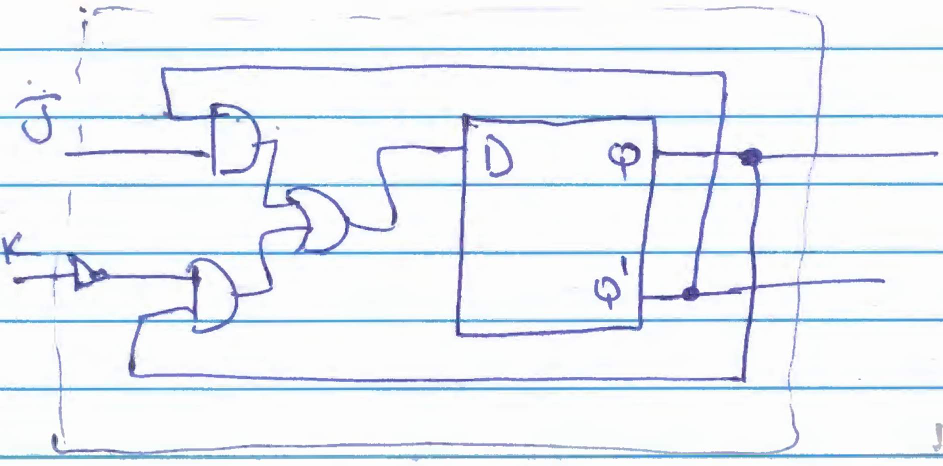
$S = JQ'$

$R = J'K + JKQ$

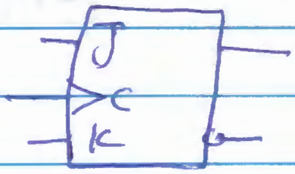
JK \ Q	0	1
00		1
01		
11	1	
10	1	1

$D = JQ' + K'Q$

⟹



$D = JQ' + K'Q$



characterestic equation
$Q(t+1) = JQ' + K'Q$

④ Toggle FF (T-FF)

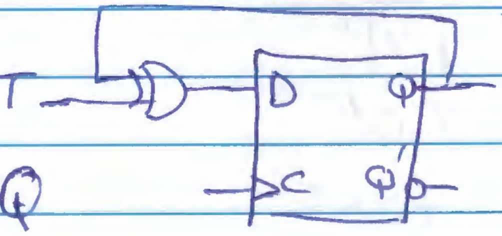
T	Q(t+1)
0	Q(t)
1	Q'(t)

(characterestic)

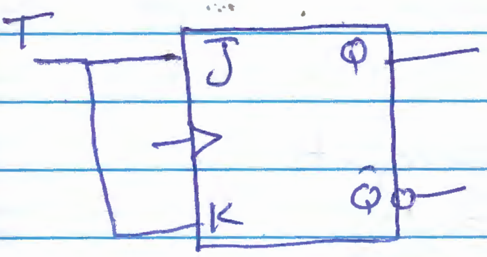
⇒

T	Q(t)	Q(t+1)	D
0	0	0	0
0	1	1	1
1	0	1	1
1	1	0	0

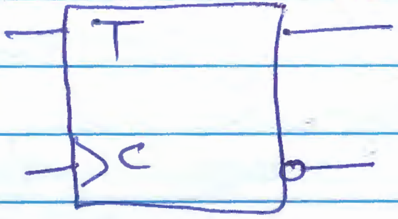
T \ Q	0	1
0		1
1	1	



$D = TQ' + T'Q = T \oplus Q$



(derive this)



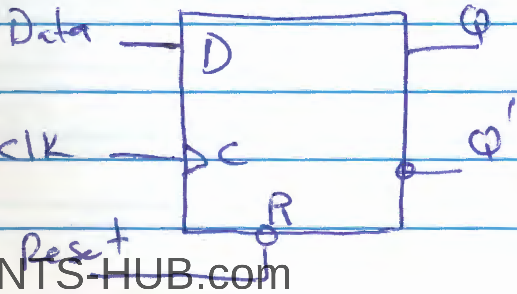
characterestic equation

$$Q(t+1) = T \oplus Q$$

* Asynchronous Inputs of Flip-Flops (direct input)

- Some FF have asynchronous inputs that are used to force the flip-flop to a particular state independent of the clock. (Initial state of the FF is unknown ??). (when powered is turned on.)
- input that sets flip-flop to 1 is preset or direct set
- input that reset flip-flop to 0 is called clear or direct reset).

* D-FF with direct reset (clear)



R	C	D	Q	Q'
0	X	X	0	1
1	↑	0	0	1
1	↑	1	1	0

*

⊛ HDL for sequential Circuits:

⊛ Behavioural Modelling

⊛ Two kinds of behavioral statements in Verilog HDL : initial and always

- initial: executes once beginning at time=0
- always: executes repeatedly until the simulation terminate.

⊛ generation of clock for simulation

①

```
initial
  begin
    clock = 1'b0;     // clock is a variable nam
    repeat (30)
      #10 clock = ~clock;
  end
```

②

```
initial
    begin
      clock = 1'b0;
        # 300 $finish;
    end
always
      #10 clock = ~clock;
```

(*) always @ (event control expression)
procedural assignments

(*) two kinds of events

(1) Level sensitive events (mainly in combinational circuits).

always @ (A or B or select)

(2) Edge-triggered events

Two keywords : posedge and negedge

always @ (posedge clock or negedge reset)

(*) A procedural assignment is an assignment within an <ins>initial</ins> or <ins>always</ins> statement.

(*) Description of D-Latch with control input

```
module D-Latch (Q, D, Control);
 output Q;
 input D, control;
 reg Q;
 always @ (control or D)
  if (Control ==1) Q = D;
endmodule
```

*

⊛ D flip flop

```
module D-FF (Q, D, clk);
   output Q;
   input D, CLK;
   reg Q;
 always @ (posedge CLK)
      Q = D;
 endmodule
```

⊛ Functional Description of JK flip-flop

```
module JK-FF (J, K, CLK, Q, Qnot);
  output Q, Qnot;
  input J, K, CLK;
   reg Q;
 assign Qnot = ~Q;
 always @ (posedge CLK)
      case ({J, k})
        2'b00 : Q = Q;
        2'b01 : Q = 1'b0;
        2'b10 : Q = 1'b1;
        2'b11 : Q = ~Q;
      endcase
endmodule
```

⊛ Functional Description of T-FF

```
module T-FF (T, clk, Q, Qnot)
 output Q, Qnot;
 input T, CLK;
  reg Q;
 assign Qnot = ~Q;
always @(posedge CLK)
    ~~if (T == 0) Q = Q;~~
    if (T == 1) Q = ~Q;
endmodule
```

⊛ Analysis of Clocked Sequential Circuits:

⊛ 3 important things in the analysis of sequential circuits: State equation, state table and state diagram.

- if you know one of these 3 things ⇒ you can derive the others.

① State equation (or transition equation): specifies the next state as a function of the present state and inputs.

Ex.

Figure 5-15

- since the D input of a flip-flop determines the value of the next state ⇒

$$A(t+1) = A(t)\,x(t) + B(t)\,x(t)$$
$$B(t+1) = A'(t)\,x(t)$$

or for simplicity:

$$\left.\begin{aligned} A(t+1) &= Ax + Bx \\ B(t+1) &= A'x \end{aligned}\right\}$$ next state

and the output

~~y(t)~~ $y = (A+B)\,x'$ (no clock

② State table (or transition table)

- specifies the time sequence of inputs, outputs and flip-flop states.
- The table consists of four sections labeled present state, input, next state, and output

⇒