Lecture 3
Review on Digital Logic (Part 2)

Xuan ‘Silvia’ Zhang
Washington University in St. Louis

http://classes.engineering.wustl.edu/ese461/

Circuit Optimization

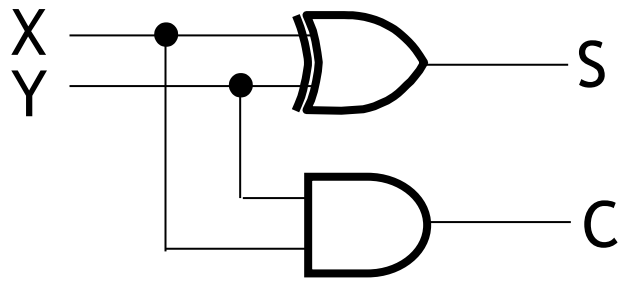
- Simplest implementation
- Cost criterion
  - literal cost (L)
  - gate input cost (G)
  - gate input cost with NOTs (GN)
- Examples (all the same function):
  - F = BD + AB'C + AC'D' L =
  - F = BD + AB'C + AB'D' + ABC' L =
  - F = (A + B)(A + D)(B + C + D')(B' + C' + D) L =
  - Which solution is best?

Half Adder

- Truth Table

X	Y	C	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

- Expression
  - $S = X \oplus Y$
  - $C = X \cdot Y$
- Logic Implementation

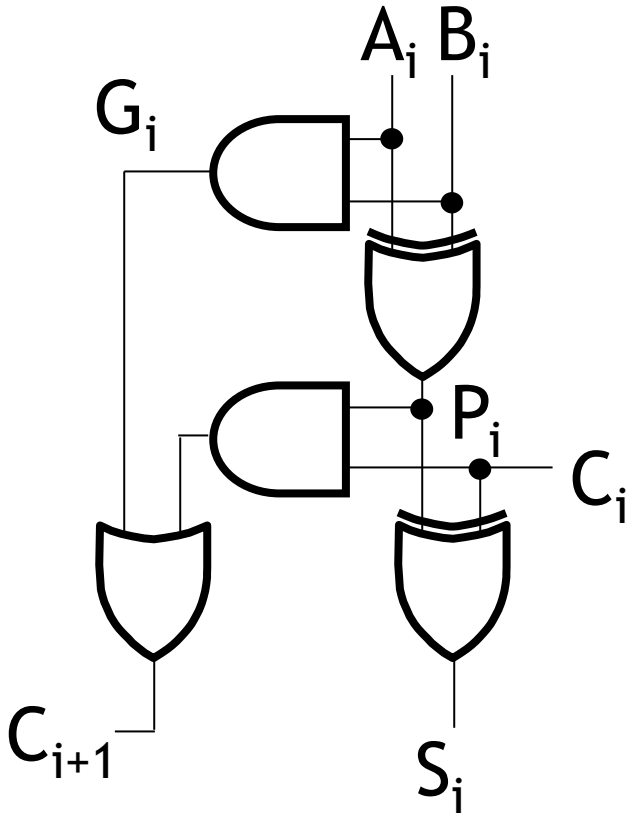


Full Adder

- Expression
  - $S_i = A_i \oplus B_i \oplus C_i$
  - or $C_{i+1} = A_iB_i + (A_i \oplus B_i)C_i$
  - $C_o = AB + (A \oplus B)C_i$

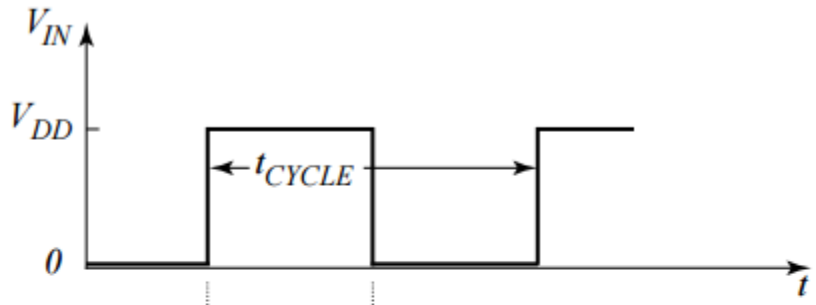
Alternatively

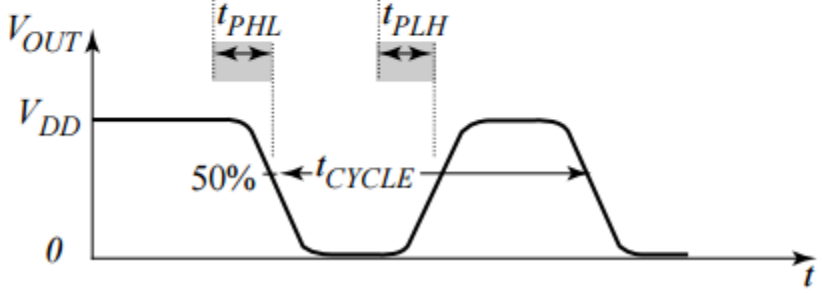
- G = generate (=AB) and
- P = propagate (=A ⊕ B)



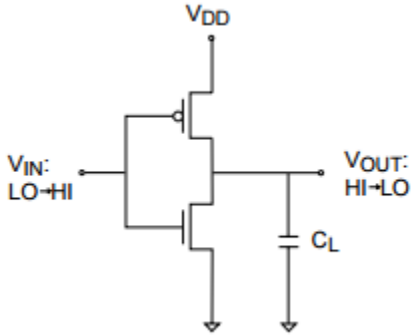
  - $C_{i+1} = G_i + P_i \cdot C_i$
  - or C_o= (G = Generate) OR (P =Propagate AND C_i = Carry In)

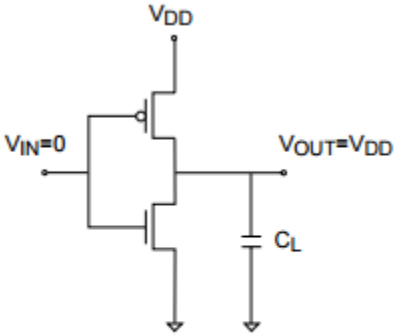
Propagation Delay



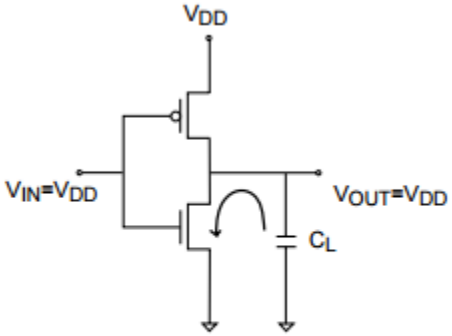


$$t_p = \frac{1}{2}(t_{PHL} + t_{PLH})$$

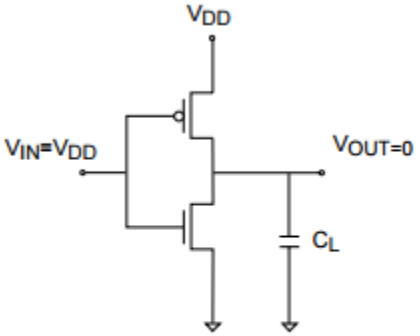




t=0-



t=0+



t->infty

Delay of a Full Adder

- Delay of the Sum and Carry bit
  - $S_0 = A_0 \oplus B_0 \oplus C_0$

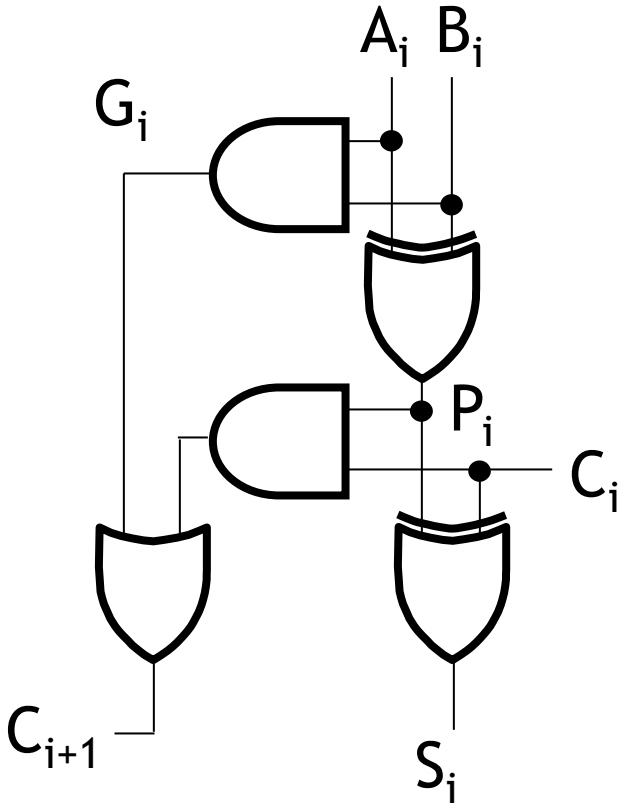
    $A_0 \oplus B_0$: 2 delays

    2+2=4 delays
  - $C_1 = A_0B_0 + (A_0 \oplus B_0)C_0$

    $A_0 \oplus B_0$: @2

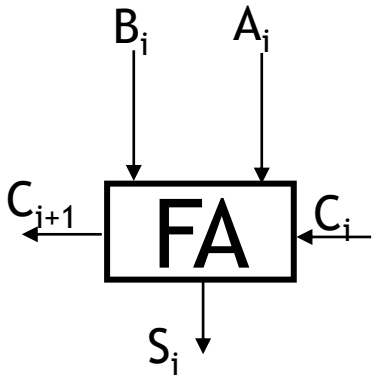
    $(A_0 \oplus B_0)C_0$: @3

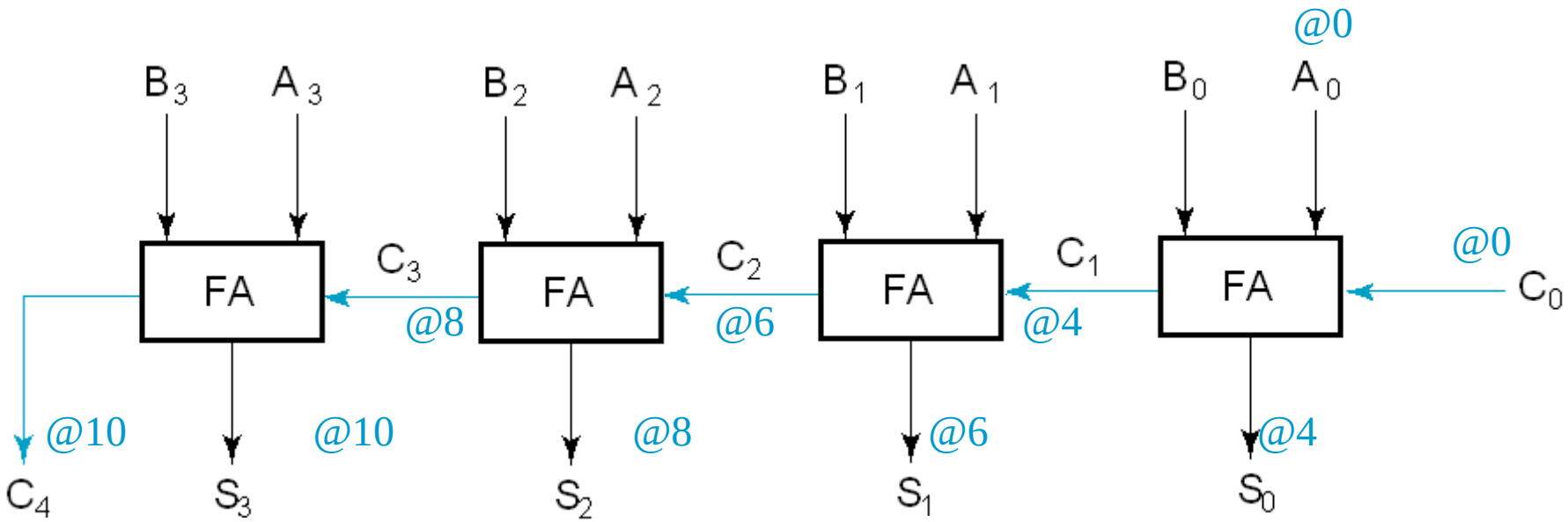
    2+2=4 delays
  - C_n?
  - S_n?



Ripple Carry Adder

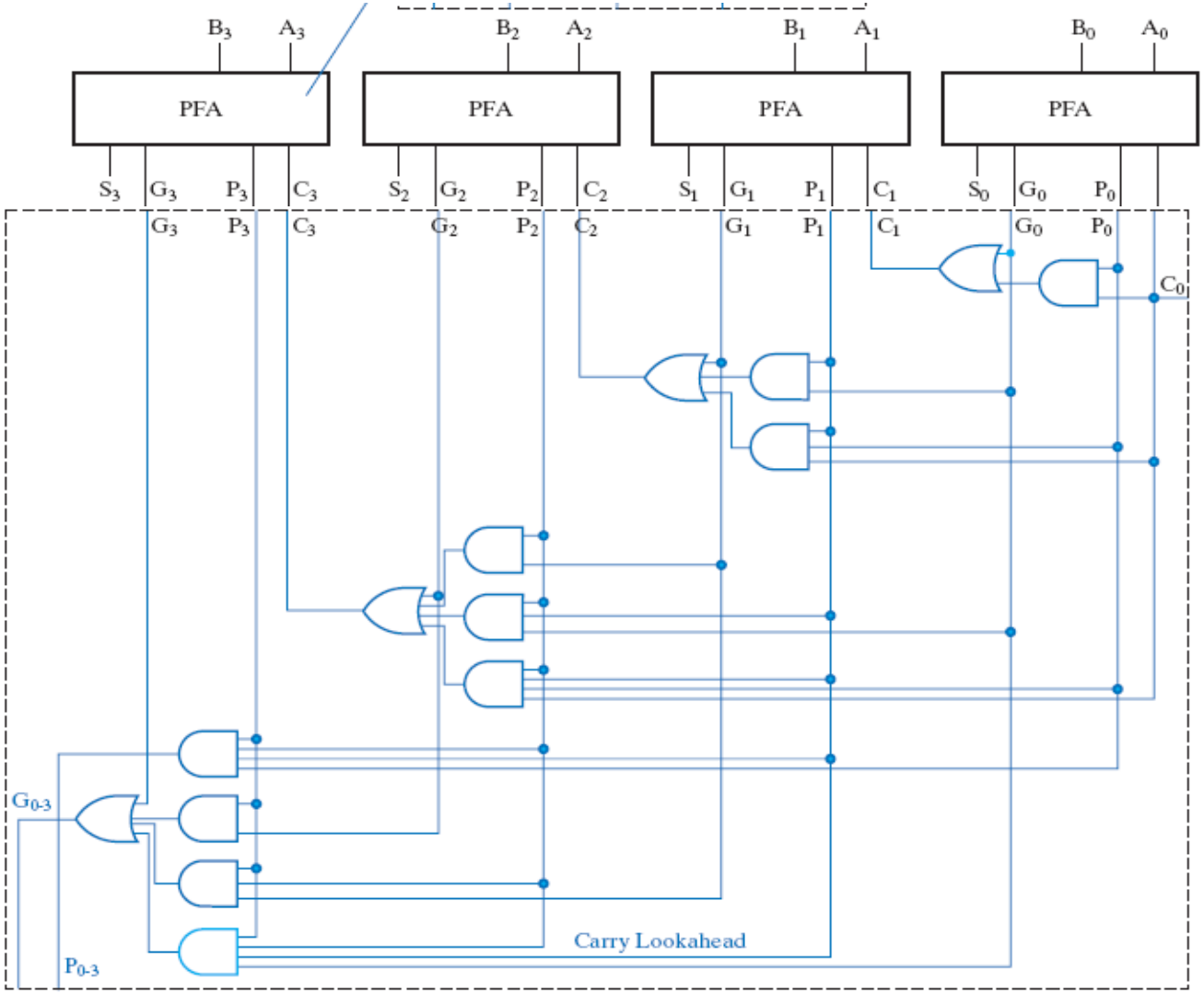
- Cascade four 1-bit full adders





Carry Lookahead Adder

- Separate circuit to calculate the carry



Revisit Number Representation

- Represent real numbers in hardware
  - accuracy
  - implementation complexity
  - robustness to errors
- Fixed-point
  - e.g. $(11.0101)_2$
  - 2's complement fractional binary number

Fract. binary format	Number of integer bits (including sign bit)	Number of fractional bits	Maximum positive decimal value	Maximum negative decimal value	Lsb decimal value
8.0	8	0	127.0	–128.0	1.0
7.1	7	1	63.5	–64.0	0.5
6.2	6	2	31.75	–32.0	0.25
5.3	5	3	15.875	–16.0	0.125
4.4	4	4	7.9375	–8.0	0.0625
3.5	3	5	3.96875	–4.0	0.03125
2.6	2	6	1.984375	–2.0	0.015625
1.7	1	7	0.9921875	–1.0	0.0078125

Fractional Binary Numbers

- 16-bit DSP
  - digital signal processor (DSP)
  - 1.15 (Q15) format

16-bit binary format	Decimal value	Binary	Hex
Two's complement	0.9999694824...	0◇111 1111 1111 1111	7FFF
1.15 (Q15) format	0.5	0◇100 0000 0000 0000	4000
	0.25	0◇010 0000 0000 0000	2000
resolution →	0.0000305175...	0◇000 0000 0000 0001	0001
	0	0◇000 0000 0000 0000	0000
	−0.0000305175...	1◇111 1111 1111 1111	FFFF
	−0.25	1◇110 0000 0000 0000	E000
	−0.5	1◇100 0000 0000 0000	C000
	−1.0	1◇000 0000 0000 0000	8000

  - Two Q15 number multiply = Q30 (2.30 format)

Fixed Point Multiplication

- Two Q15 number multiply
  - Q15 × Q15 = Q30
  - 2.30 format, 32 bits, two sign bits
  - MSB: extended sign bit
  - need to truncate back to 1.15 format
  - left shift by one bit, storing upper 16 bits
  - right shift by 15 bits, storing lower 16 bits

- Dynamic range
  - in a b-bit system

$$\text{dynamic range}_{\text{linear}} = \frac{\text{largest positive word value}}{\text{smallest positive word value}}$$

$$= \frac{2^b - 1}{1} = 2^b - 1.$$

Outline

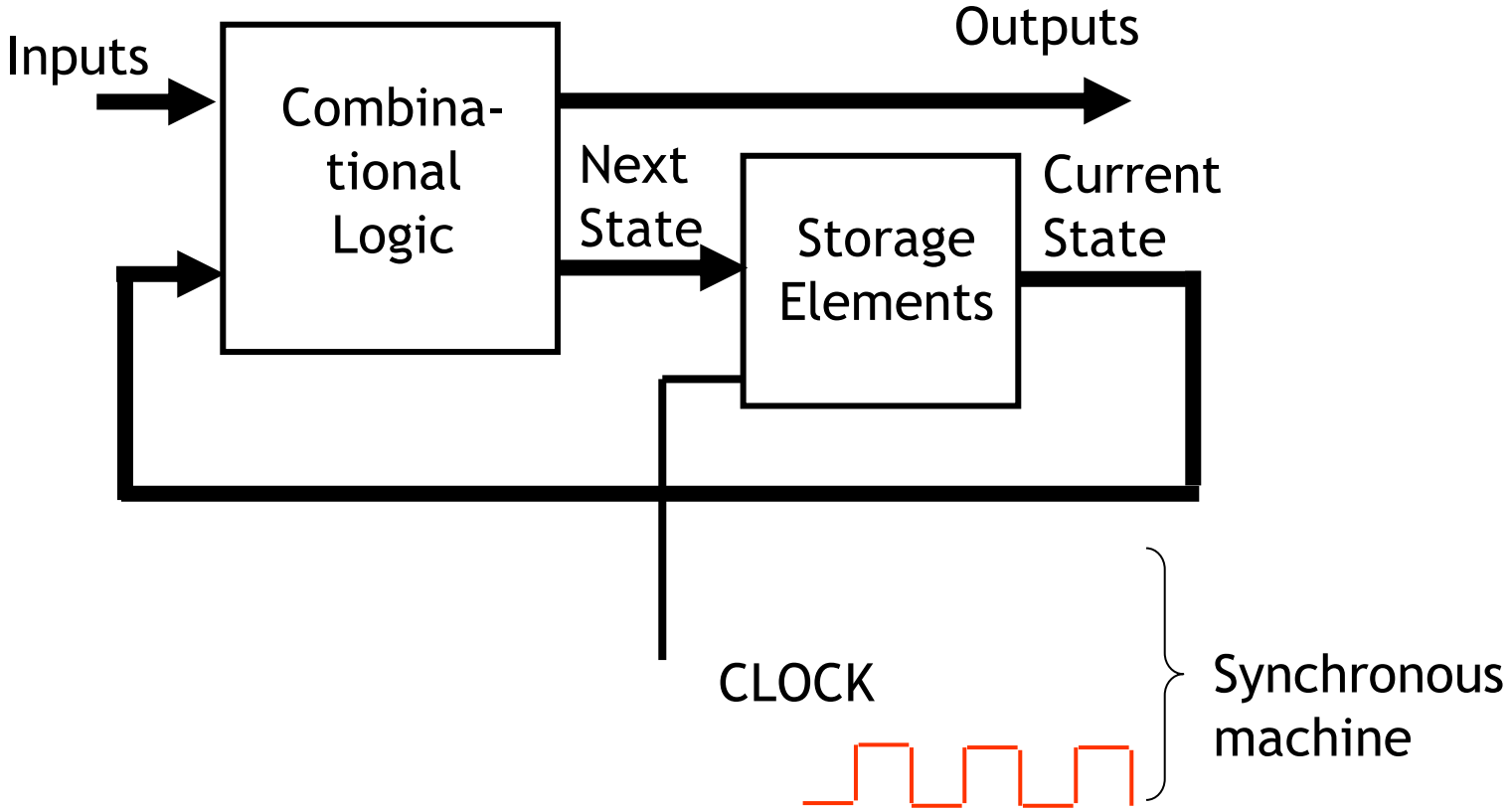
Arithmetic Logic

Sequential Logic

Memory Circuit

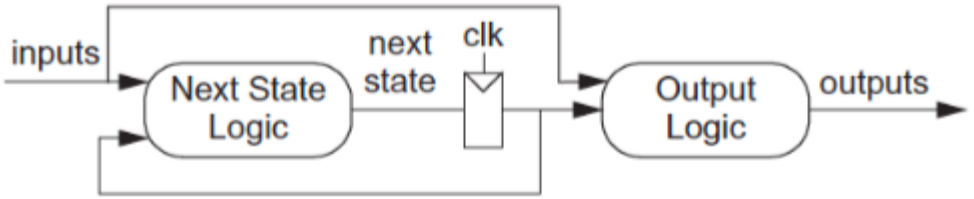
Sequential Logic

- Next state function
  - Next State = f(Inputs, State)
- Output function

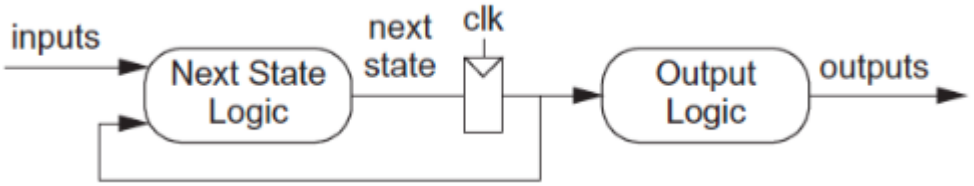


Finite State Machine (FSM)

- Mathematical model of computation
  - consist of a finite number of “states”
  - only one state at a time (Current State)
  - change state at a trigging event or condition
- Mealy machine
  - Outputs = g(Inputs, State)

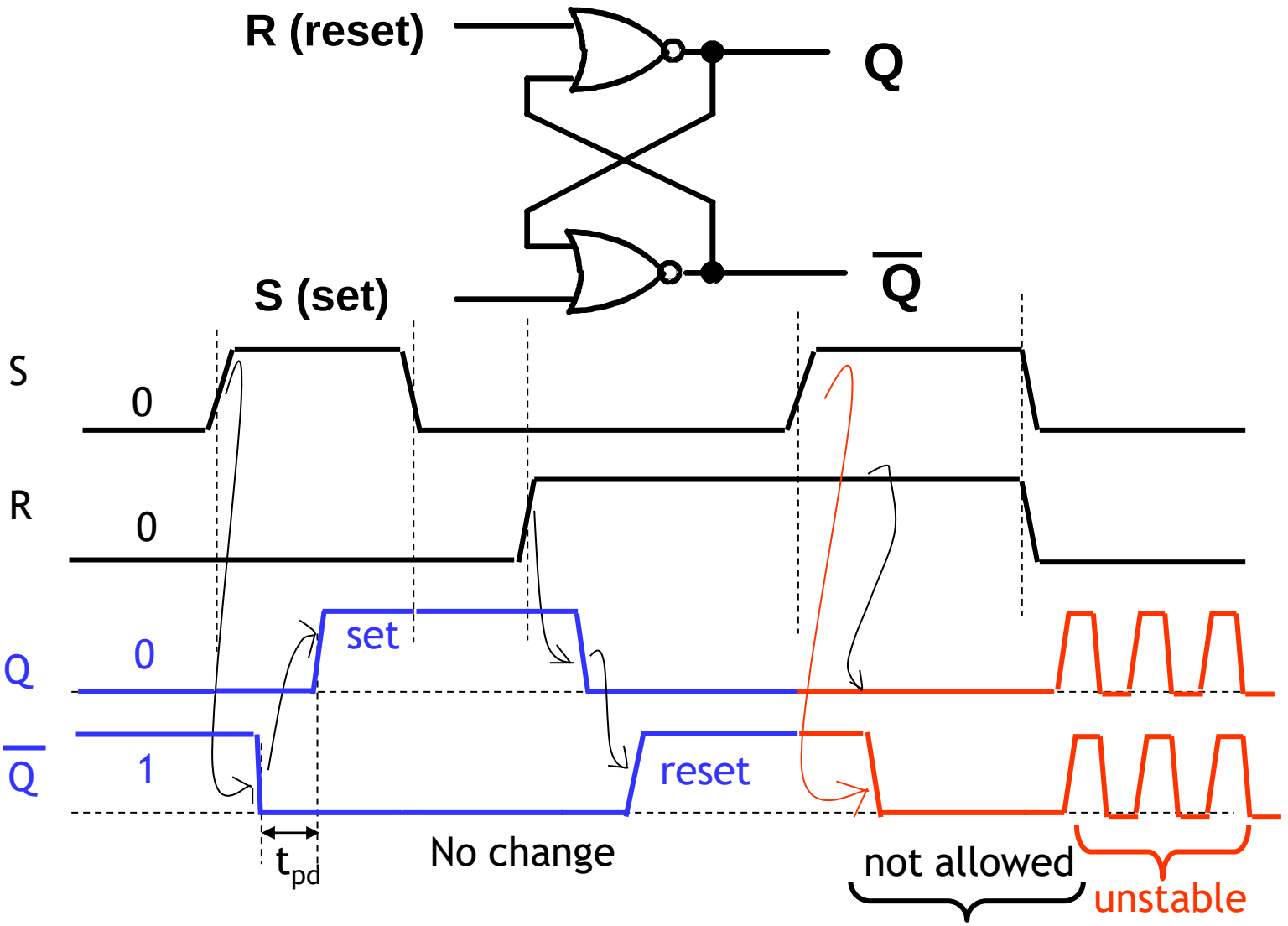


- Moore machine
  - Outputs = h(State)



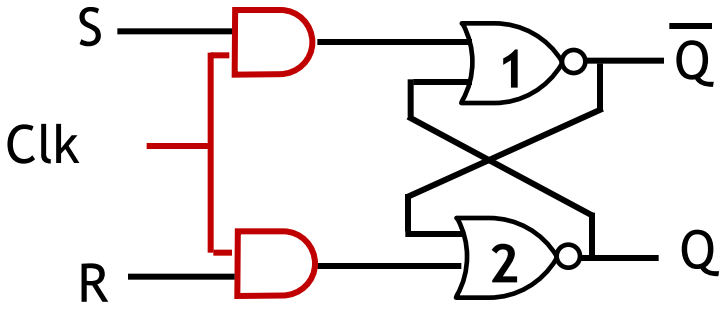
SR Latch

- Basic NOR latch

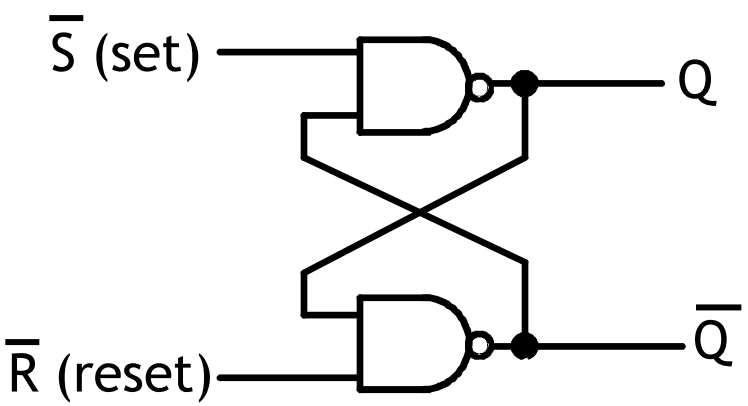


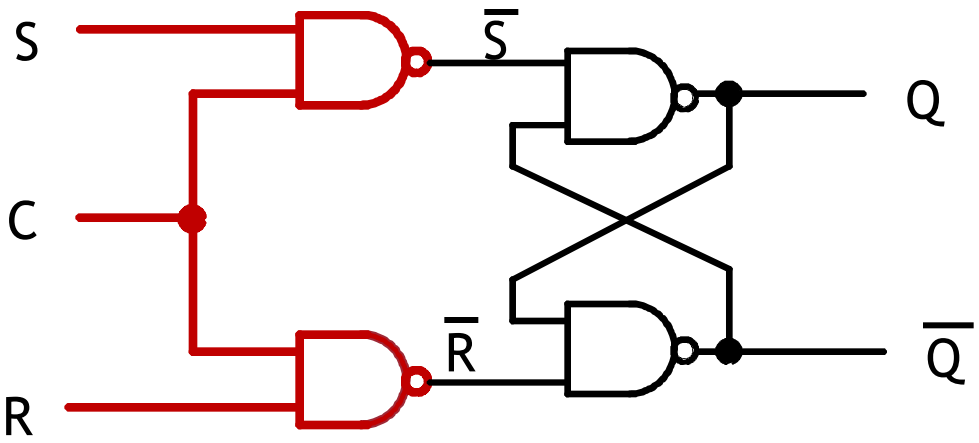
Other SR Latches

- Clocked



- NAND SR latch



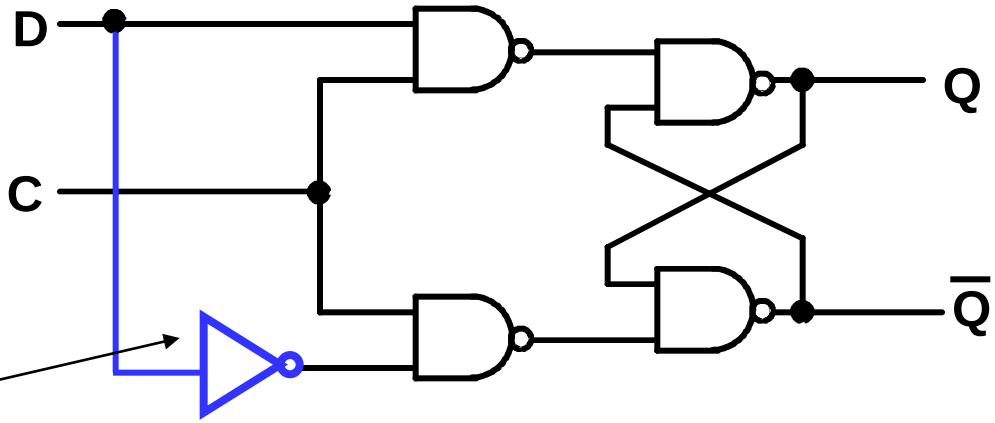


D Latch

- Truth table

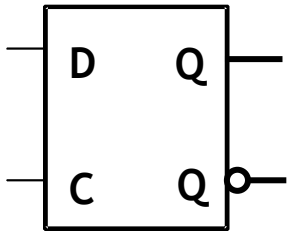
SR latch:

S	R	Q^+	$\overline{Q^+}$
0	0	hold,	
0	1	0	1
1	0	1	0
~~1~~	~~1~~	0	0



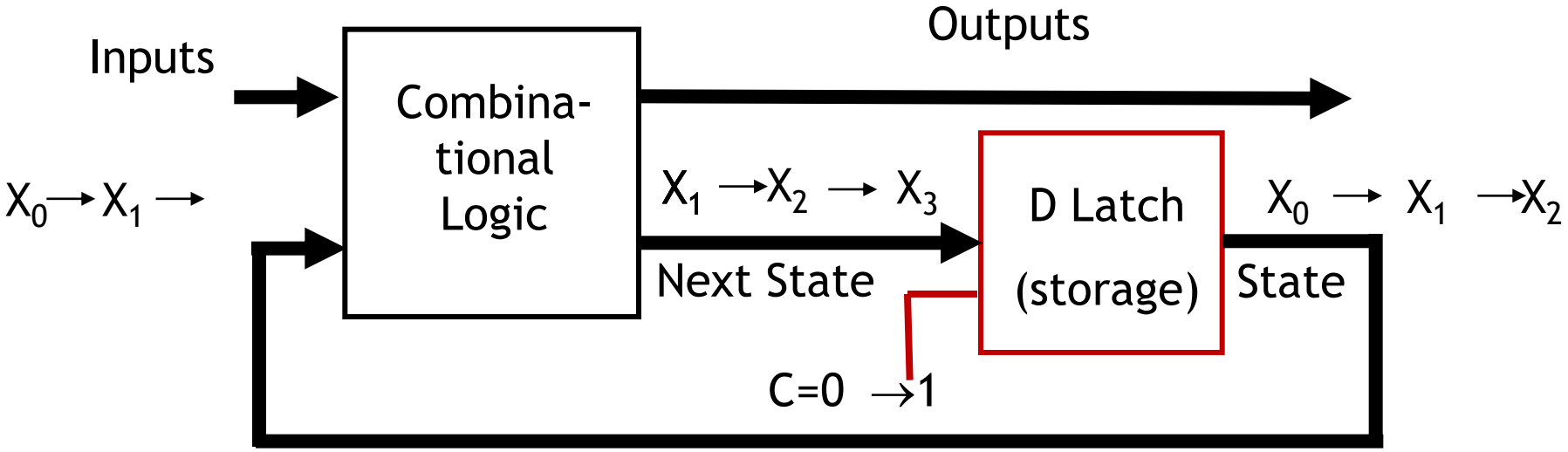
D latch

D	Q(t+1)
0	0
1	1

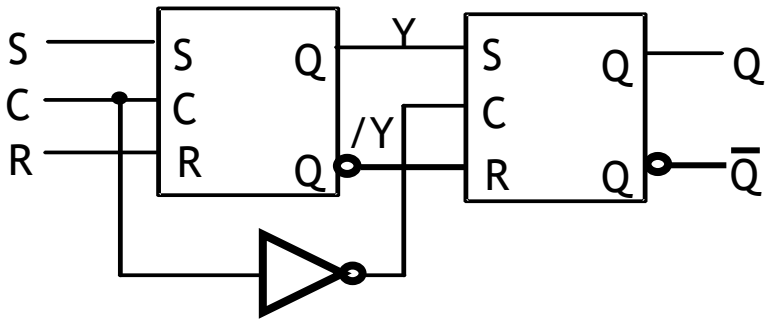


Flip-Flop

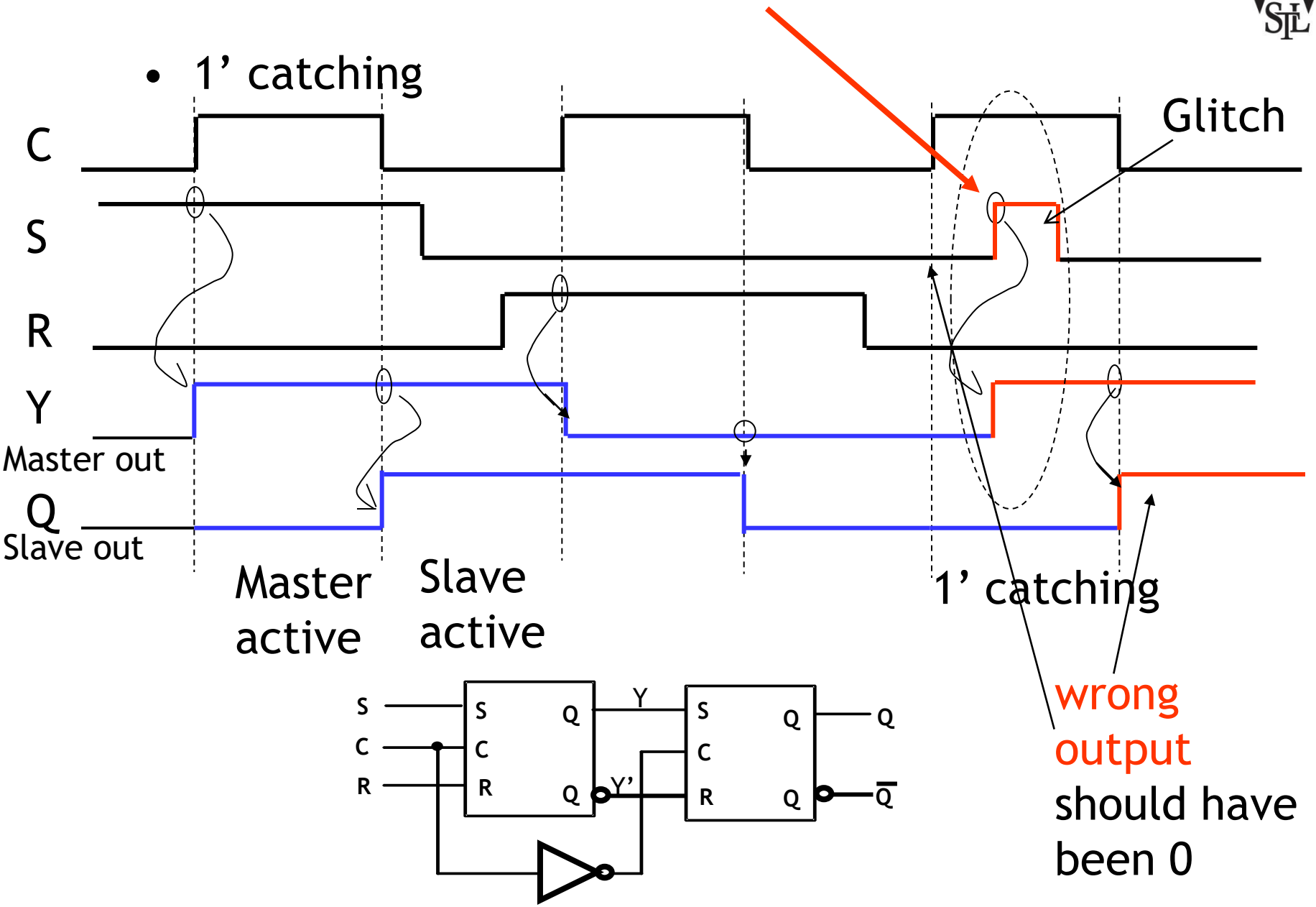
- Latch timing issue
  - transparent when C = 1
  - state should change only once every new clock cycle



- Master-slave flip flop
  - break feedthrough

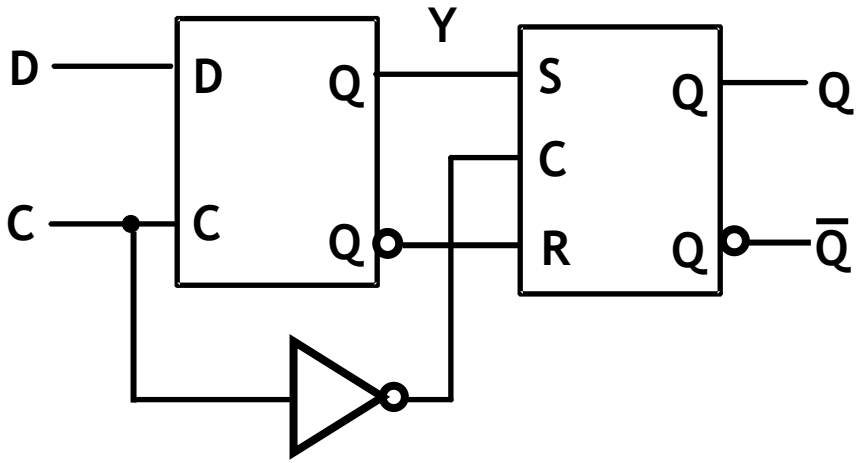


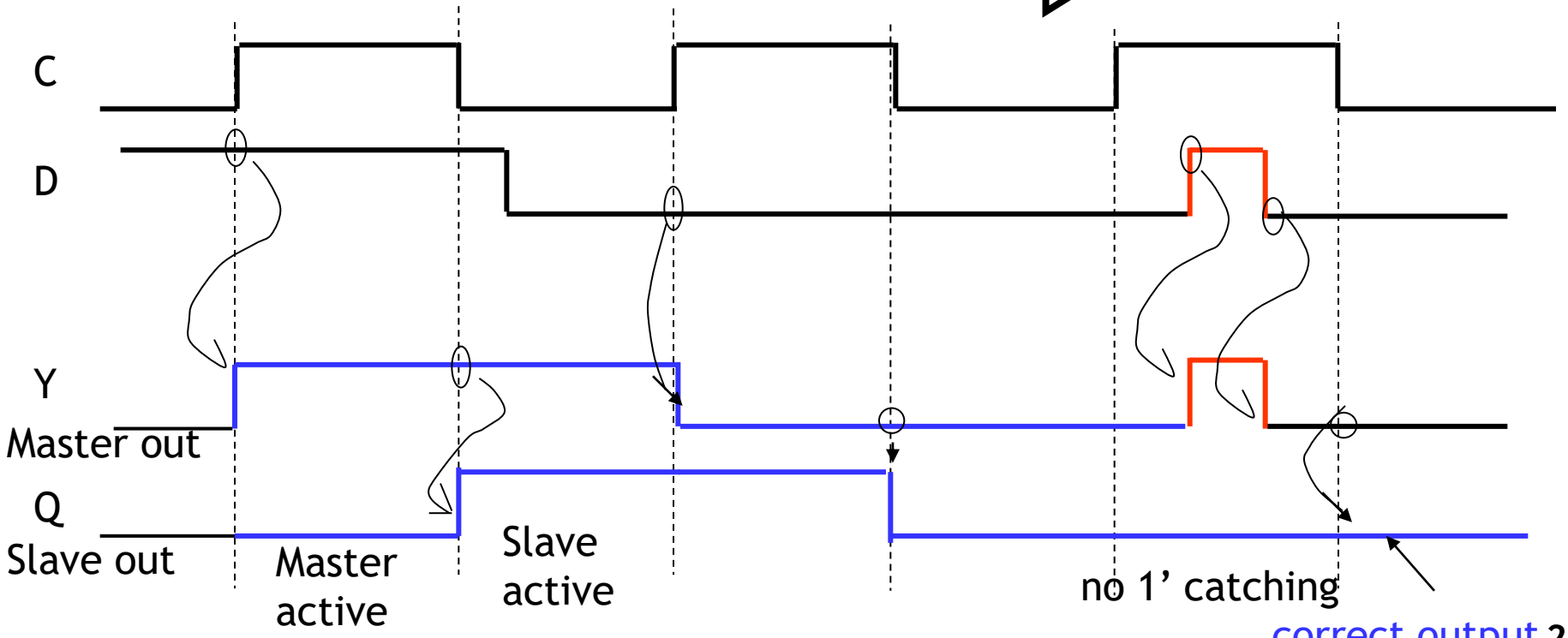
Flip-Flop Timing Issue



Edge-Triggered D Flip-Flop (DFF)

- Why edge trigger?
- D replace S and R input

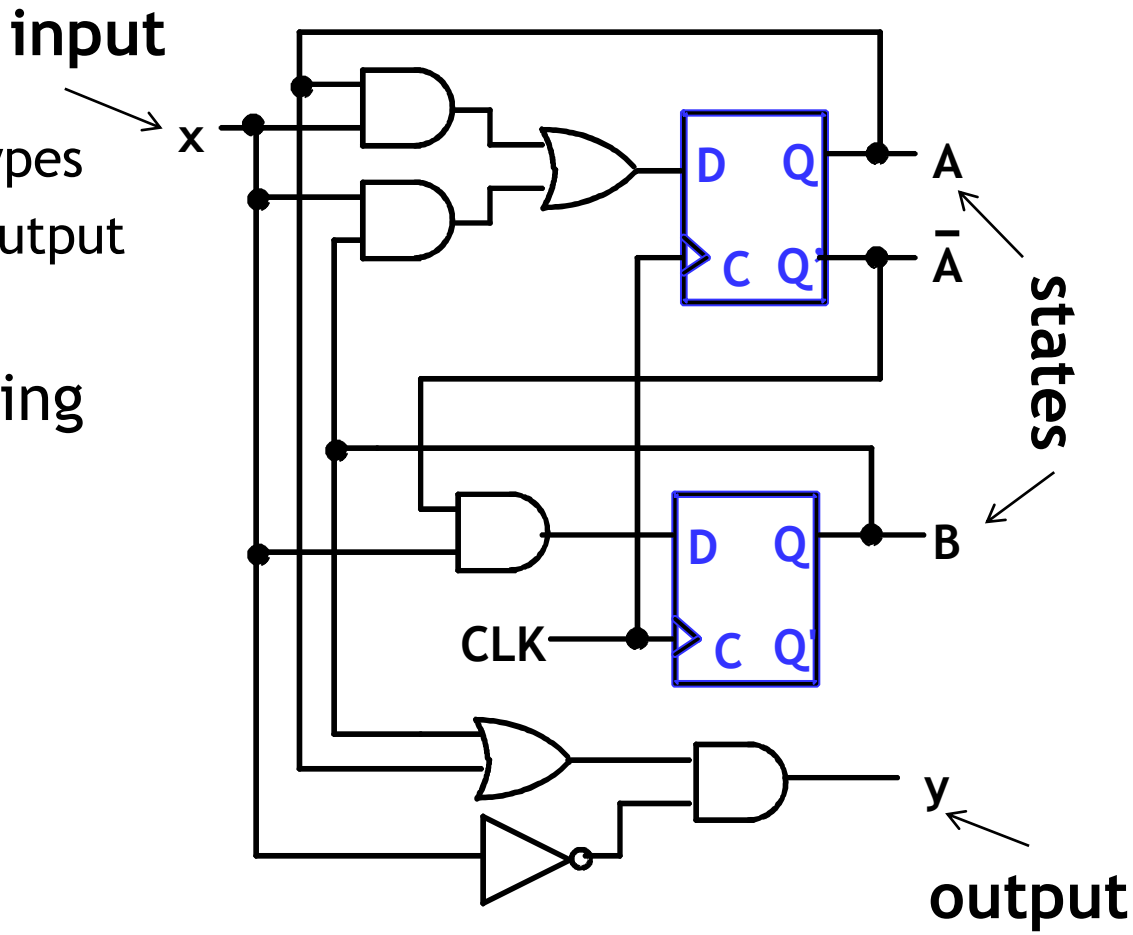




correct output

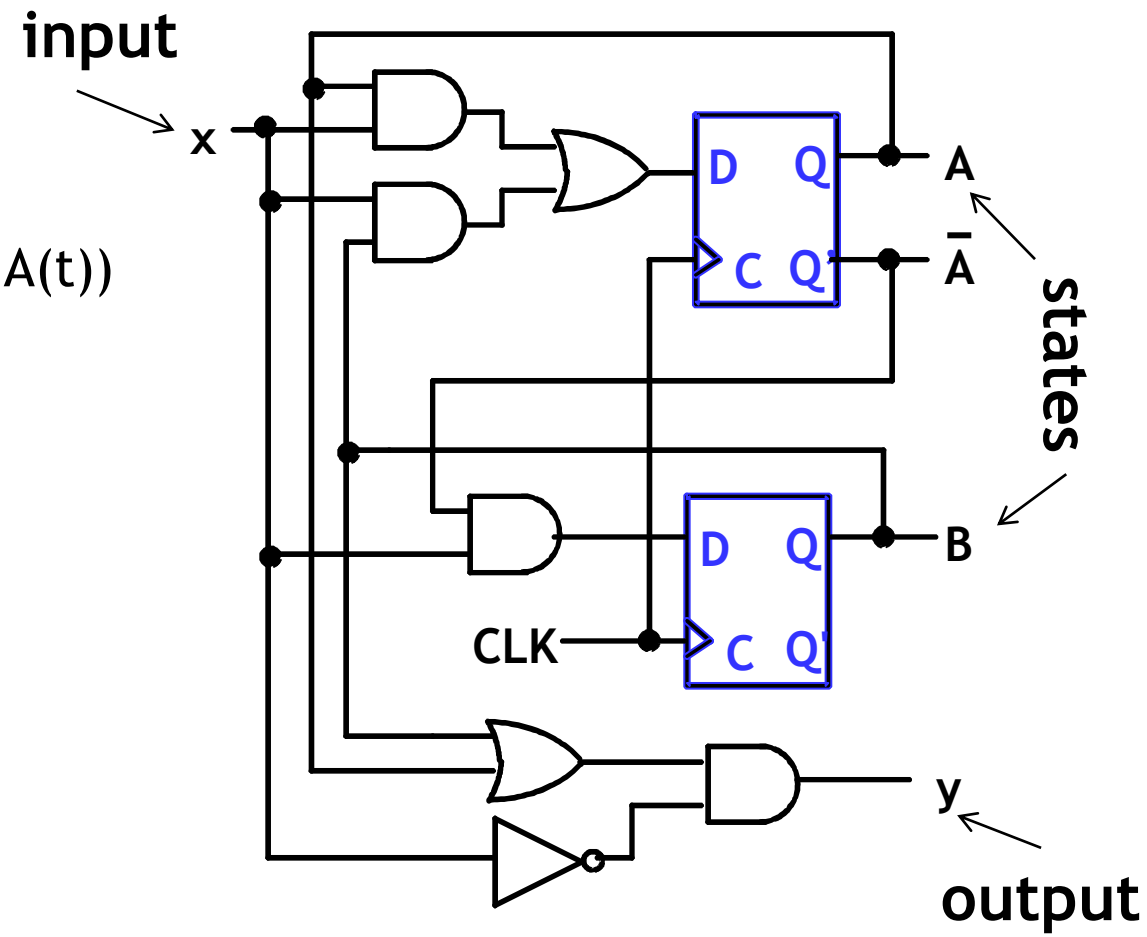
Sequential Circuit Analysis

- Design steps
  - word description
  - state diagram
  - state table
  - select flip-flop types
  - input to FF and output
  - verification
- Reverse engineering



Input Equations

- To flip-flops
  - $D_A = A(t)x(t)+B(t)x(t)$
  - $D_B = \bar{A}(t)x(t)$
- Output y
  - $y(t) = x(t)(B(t) + A(t))$



State Table

- For the example: $A(t+1) = A(t)x(t) + B(t)x(t)$

$B(t+1) = \bar{A}(t)x(t)$

$y(t) = \bar{x}(t)(B(t) + A(t))$

Inputs of the table | Outputs of the table

Present State		Input	Next State		Output
A(t)	B(t)	x(t)	A(t+1)	B(t+1)	y(t)
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	0	1
0	1	1	1	1	0
1	0	0	0	0	1
1	0	1	1	0	0
1	1	0	0	0	1
1	1	1	1	0	0

2^3 rows

(2^{m+n}) rows

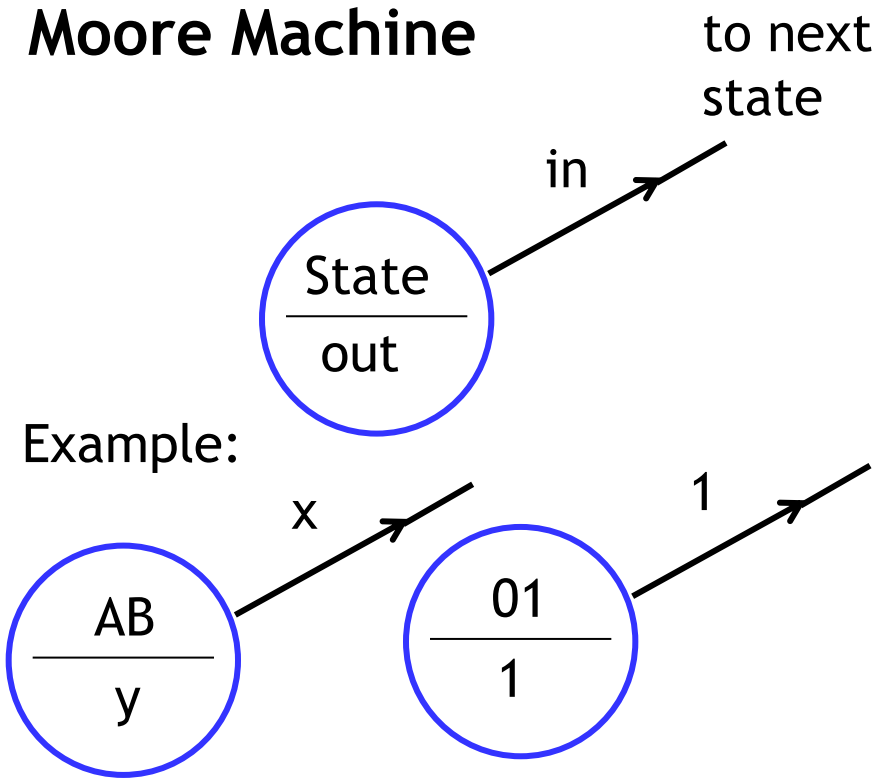
m: no. of FF

n: no. of inputs

State Diagram

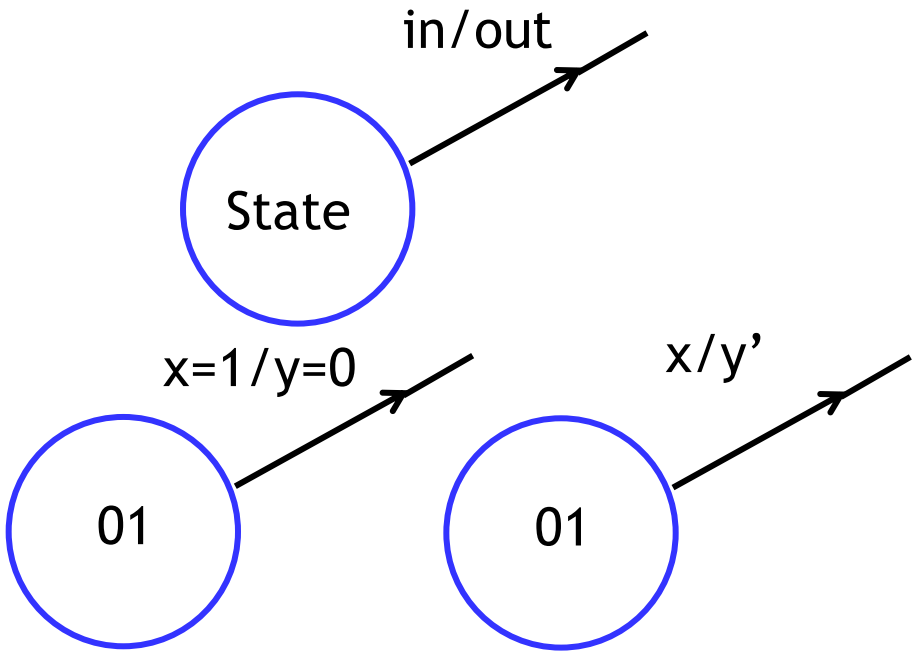
- Conventions

Moore Machine



Moore type output depends only on state

Mealy Machine



Mealy type output depends on state and input

Outline

Arithmetic Logic

Sequential Logic

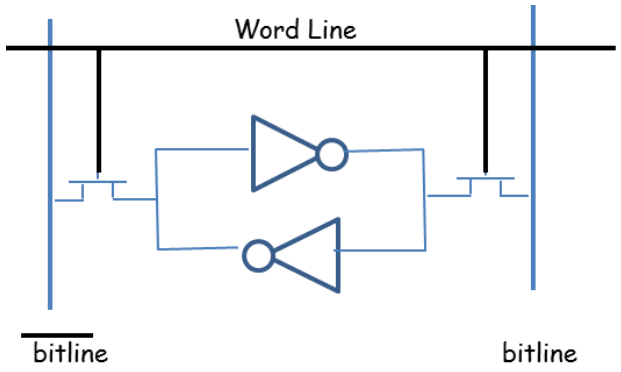
Memory Circuit

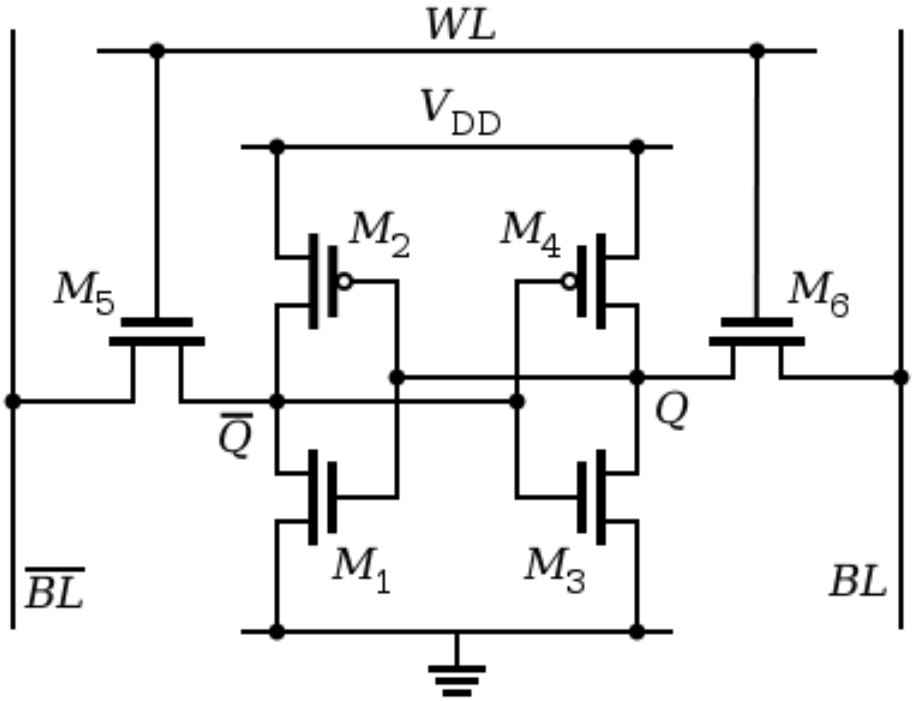
Static RAM

- Applications
  - CPU register file, cache, embedded memory, DSP
- Characteristics
  - 6 transistor per cell, other topologies
  - no need to refresh
  - access time ~ cycle time
  - no charge to leak
  - faster, more area, more expensive

SRAM Operation

- Standby
  - word line de-asserted
- Read
  - precharge bit lines
  - assert WL
  - BL rise/drop slightly
- Write
  - apply value to BL
  - assert WL
  - input drivers stronger





Summary

Number Representation

Boolean Logic and Gates

Combinational Logic

Arithmetic Logic

Sequential Logic

Memory Circuit

Questions?

Comments?

Discussion?

Homework #2

- Download problem sets from class website
- Due 09/12 (Monday) in class
- 2-day grace period