

Hardware Design I Chap. 5 Memory elements

Computing Architecture Lab.
Hajime Shimada
E-mail: shimada@is.naist.jp

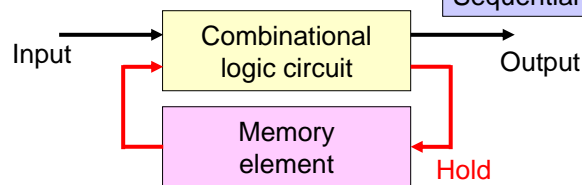
1

Why memory is required?

- To hold data which will be processed with designed hardware (for **storage**)
 - Main memory, cache, register, and so on.
- To achieve sequential circuit (for **holding temporal value**)
 - Combinational logic circuits do not permit cyclic data flow
- If we separate data flow via memory element, we can permit it

-> Chap. 2

Sequential circuit -> Chap. 6



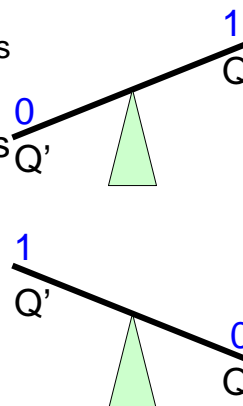
Outline

- Flip-flops and latches (for temporal value)
 - SR flip-flop and its variations
 - D flip-flop and its variations
 - D latch
- Random Access Memory (RAM) and related structure (for storage)
 - Basic organization of RAM
 - Static RAM (SRAM)
 - The other RAMs
 - Content Addressable Memory (CAM)



What's flip-flop?

- Assume seesaw
 - Outputs Q and Q' take opposite status
 - Q and Q' flip under some condition
- There are several types of flip-flops
 - SR flip-flop
 - Clocked SR flip-flop
 - Master-slave SR flip-flop
 - Master-slave D flip-flop
 - Edge trigger D flip-flop



The relations of flip-flops

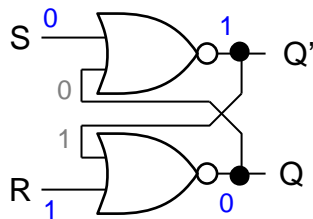
- **SR flip-flop**: set/reset procedure is complicated
 - ↓ Add clock which indicate timing of value set
- Clocked SR flip-flop: clock timing is severe
 - ↓ Accept slow clock
- Master-slave SR flip-flop: dual-rail logic is redundant
 - ↓ Simplify input to single "D"
- Master-slave D flip-flop: delay in flip-flop is large
 - A part of this organization becomes D latch
 - ↓ Reduce delay
- Edge trigger D flip-flop

Note that the latter organization requires much more gates



SR flip-flop

- Set Reset flip-flop (SR-FF)
 - "Set" means "output 1"
 - "Reset" means "output 0"
- e.g. reset status
 - The feedback loop creates stable status



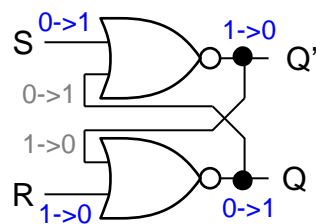
S	R	Q	Q'
0	0	Q	Q' (Memorize)
0	1	0	1
1	0	1	0
1	1	0	0 (Prohibit)



Flip of SR flip-flop

- Flip of reset status to set status

- If S becomes 1, Q' becomes 0
 - Inputs of the lower NOR becomes 0 and 1 (still outputs 0)
- After that, if R becomes 0, Q becomes 1
 - Inputs of the upper NOR becomes 1 and 1 (still outputs 0)



S	R	Q	Q'
0	0	Q	Q' (Memorize)
0	1	0	1
1	0	1	0
1	1	0	0 (Prohibit)



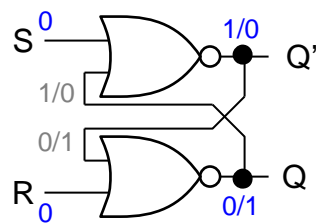
Computing Architecture Lab.
Hajime Shimada

Hardware Design I (Chap. 5)

7

Stable status of SR flip-flop

- SR flip-flop becomes stable if we input 0 to both inputs
 - It keeps prior status
- Usually, we treat this status as a basic status
 - From here, we set S or R to change status



S	R	Q	Q'
0	0	Q	Q' (Memorize)
0	1	0	1
1	0	1	0
1	1	0	0 (Prohibit)



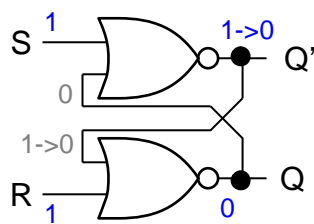
Computing Architecture Lab.
Hajime Shimada

Hardware Design I (Chap. 5)

8

What occurs when we input 1 to both inputs of SR flip-flop?

- Both outputs becomes 0
- Usually, we prohibit this status
 - It represents $Q = Q'$ which is conflictive status



S	R	Q	Q'
0	0	Q	Q' (Memorize)
0	1	0	1
1	0	1	0
1	1	0	0 (Prohibit)



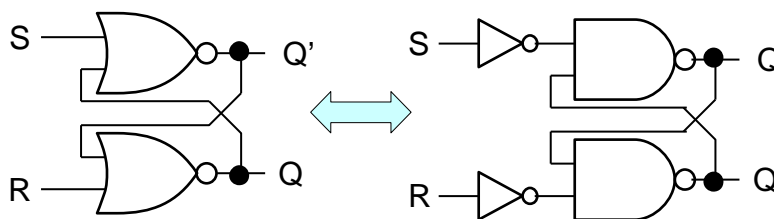
Computing Architecture Lab.
Hajime Shimada

Hardware Design I (Chap. 5)

9

The other implementation of SR flip-flop

- We can implement SR flip-flop by NAND and NOT gates
 - Note that Q and Q' are counterchanged in this implementation



Computing Architecture Lab.
Hajime Shimada

Hardware Design I (Chap. 5)

10

The relations of flip-flops

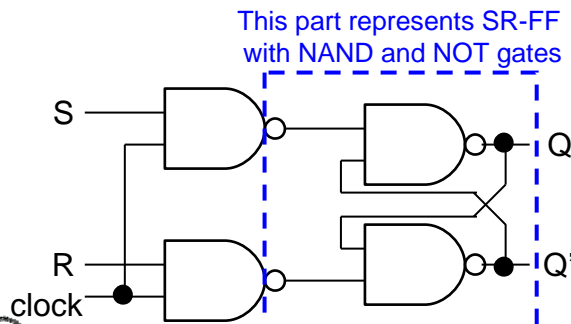
- SR flip-flop: set/reset procedure is complicated
 - ↓ Add clock which indicate timing of value set
- **Clocked SR flip-flop**: clock timing is severe
 - ↓ Accept slow clock
- Master-slave SR flip-flop: dual-rail logic is redundant
 - ↓ Simplify input to single "D"
- Master-slave D flip-flop: delay in flip-flop is large
 - A part of this organization becomes D latch
 - ↓ Reduce delay
- Edge trigger D flip-flop

Note that the latter organization requires much more gates



Clocked SR flip-flop

- A circuit which can enable set or reset input when clock=1
- If clock=0, inputs of blue rectangle becomes 0
- Also called SR latch

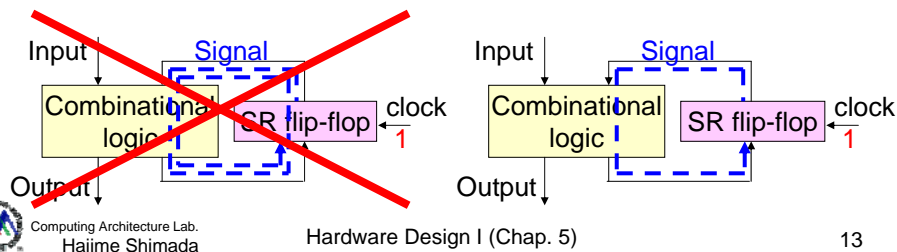


clock	S	R	Q	Q'
1	0	0	Q	Q'
1	0	1	0	1
1	1	0	1	0
1	1	1	0	0
0	*	*	Q	Q'



Signal through of clocked SR flip-flop

- It **put through signal** when clock=1
- In some case, transparent signal is unacceptable
- e.g. sequential circuit -> Chap. 6
 - There's possibility that the **signal loops for multiple times** through SR flip-flop
 - >Wrong operation from combinational logic viewpoint



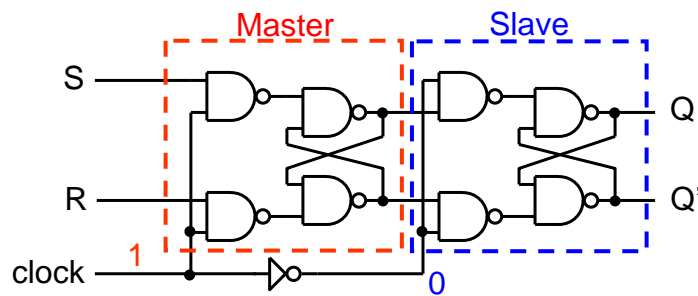
The relations of flip-flops

- SR flip-flop: set/reset procedure is complicated
 - ↓ Add clock which indicate timing of value set
- Clocked SR flip-flop: clock timing is severe
 - ↓ Accept slow clock
- **Master-slave SR flip-flop**: dual-rail logic is redundant
 - ↓ Simplify input to single "D"
- Master-slave D flip-flop: delay in flip-flop is large
 - A part of this organization becomes D latch
 - ↓ Reduce delay
- Edge trigger D flip-flop

Note that the latter organization requires much more gates

Master-slave flip-flop (1/2)

- When clock=1
 - Master captures values of S and R
 - Slave does not change status
- Multiple S and R flop is hidden



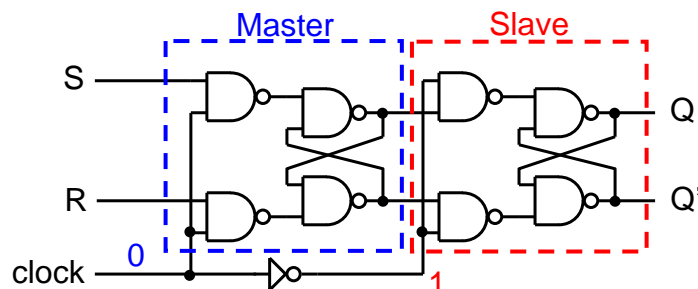
Computing Architecture Lab.
Hajime Shimada

Hardware Design I (Chap. 5)

15

Master-slave flip-flop (2/2)

- When clock=0
 - Master does not change status
 - Slave captures values from master
 - Outputs value which master captures
- The output becomes value in prior clock period



Computing Architecture Lab.
Hajime Shimada

Hardware Design I (Chap. 5)

16

The relations of flip-flops

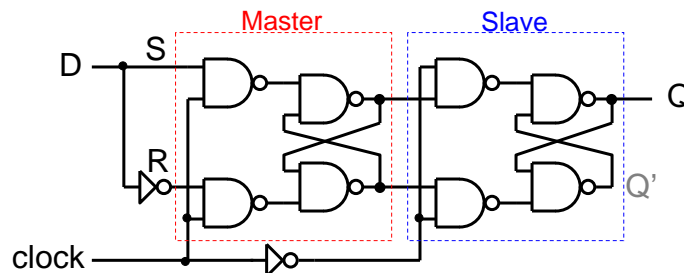
- SR flip-flop: set/reset procedure is complicated
 - ↓ Add clock which indicate timing of value set
- Clocked SR flip-flop: clock timing is severe
 - ↓ Accept slow clock
- Master-slave SR flip-flop: dual-rail logic is redundant
 - ↓ Simplify input to single "D"
- **Master-slave D flip-flop**: delay in flip-flop is large
 - A part of this organization becomes D latch
 - ↓ Reduce delay
- Edge trigger D flip-flop

Note that the latter organization requires much more gates

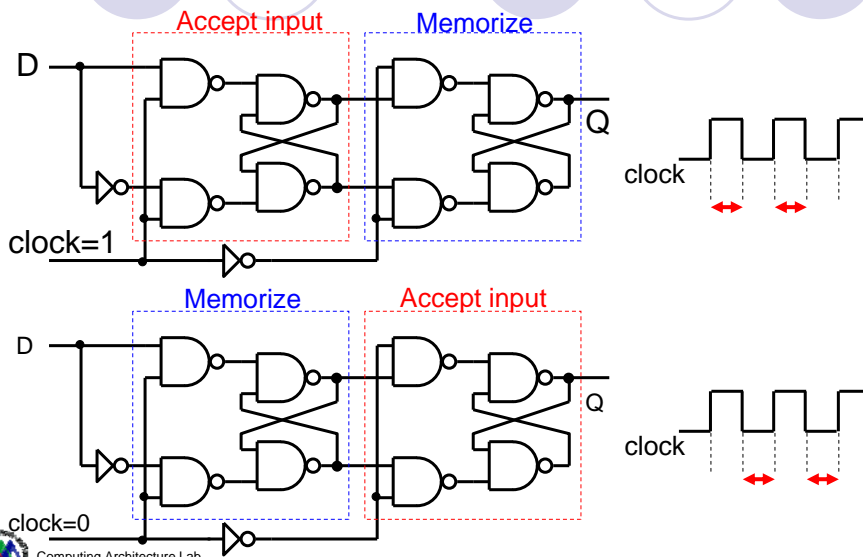


Master-slave D flip-flop

- Assuming $S=D$ and $R=D'$
- The function becomes "output D in prior clock period"
 - "D" means "delay"

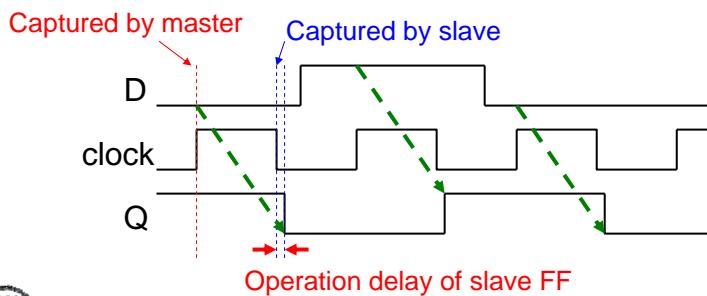


The operation of master slave D flip-flop



Timeline of D flip-flop operation

- Input value arrives Q after “half clock + alpha”
 - Alpha: operation time of slave flip-flop
- How to remove this delay?



The relations of flip-flops

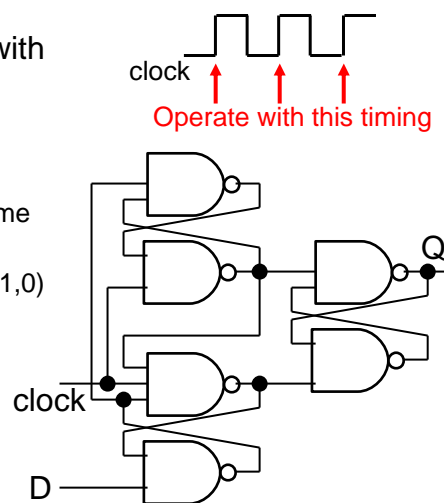
- SR flip-flop: set/reset procedure is complicated
 - ↓ Add clock which indicate timing of value set
- Clocked SR flip-flop: clock timing is severe
 - ↓ Accept slow clock
- Master-slave SR flip-flop: dual-rail logic is redundant
 - ↓ Simplify input to single "D"
- Master-slave D flip-flop: delay in flip-flop is large
 - A part of this organization becomes D latch
 - ↓ Reduce delay
- **Edge trigger D flip-flop**

Note that the latter organization requires much more gates



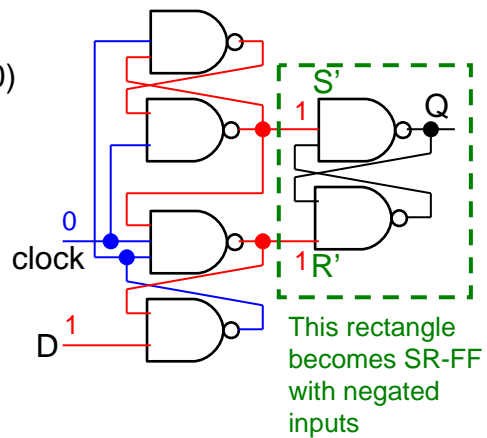
Edge trigger D flip-flop

- A flip-flop which operates with **edge of clock**
- It can output value **after a moment** of clock edge
 - A moment: state transition time of logic gates
 - Utilize (S,R)=(0,0) to (S,R)=(1,0) or (S,R)=(0,1) action in it



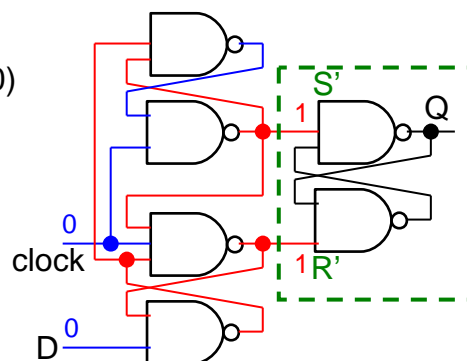
Operation of edge trigger D flip-flop (1/4)

- Assume clock=0, D=1
- It holds values
 - Assume that (S,R)=(0,0) state in SR flip-flop



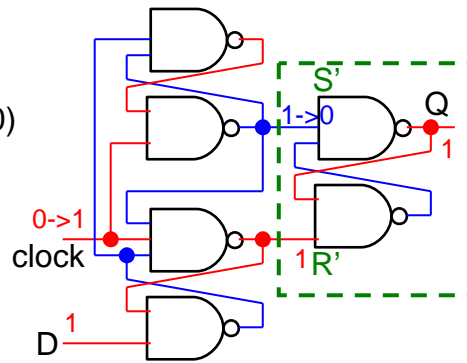
Operation of edge trigger D flip-flop (2/4)

- Assume clock=0, D=0
- Also it holds values
 - Assume that (S,R)=(0,0) state in SR flip-flop



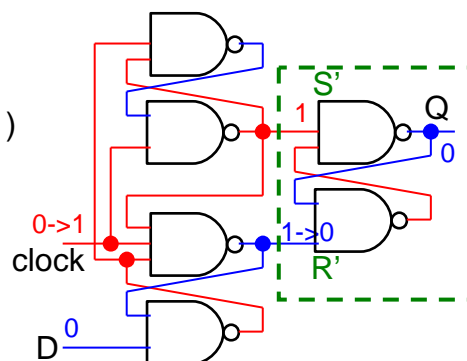
Operation of edge trigger D flip-flop (3/4)

- Assume clock=0->1 under D=1
- Q becomes 1
 - Assume that (S,R)=(1,0) state in SR flip-flop



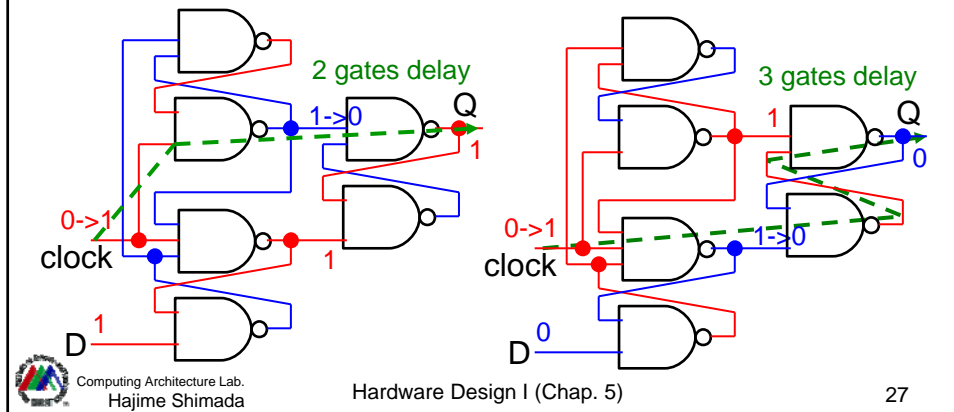
Operation of edge trigger D flip-flop (4/4)

- Assume clock=0->1 under D=0
- Q becomes 0
 - Assume that (S,R)=(0,1) state in SR flip-flop



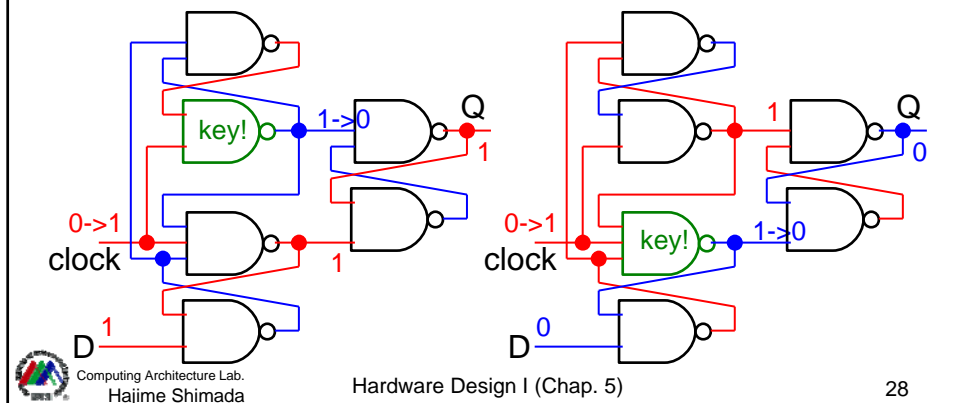
Operation delay of edge trigger D flip-flop

- It requires 3 gates operation delay in maximum



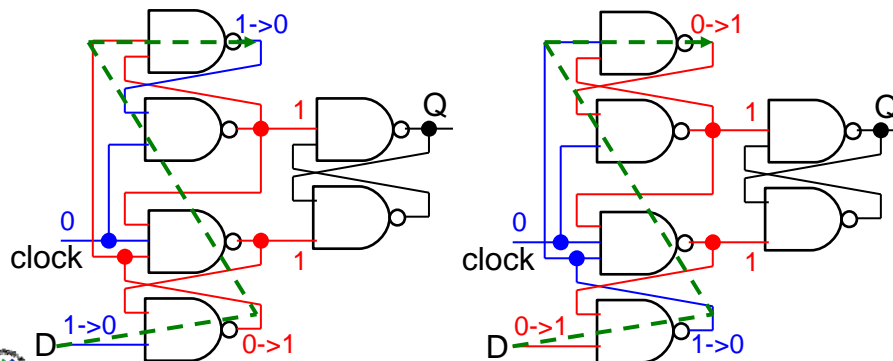
How long do we have to keep D value? (after clock has injected)

- After marked gate transition, internal state does not change even if D changes
- It is called “**Hold time** is 1 gate operation delay”



How long do we have to keep D value? (before clock has injected)

- When we translate D value, it requires 2 gate delay to become ready to accept clock pulse status
- It is called “**Setup time** is 2 gate operation delay”



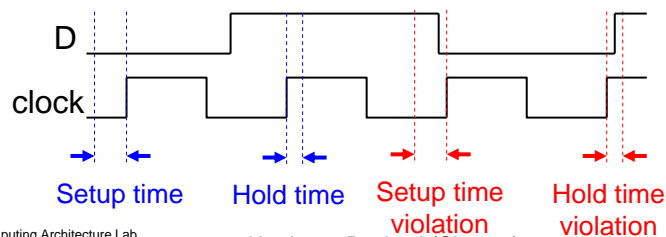
Computing Architecture Lab.
Hajime Shimada

Hardware Design I (Chap. 5)

29

Setup time and hold time

- Setup time
 - The restriction before clock pulse
 - Never change D in this term
- Hold time
 - The restriction after clock pulse
 - Never change D in this term



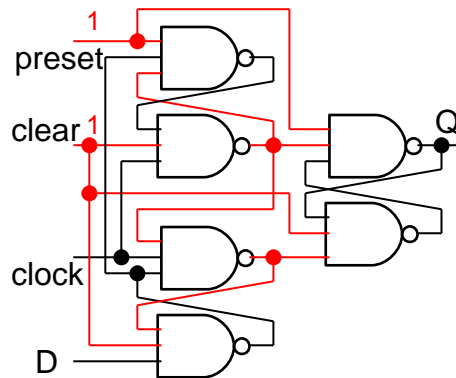
Computing Architecture Lab.
Hajime Shimada

Hardware Design I (Chap. 5)

30

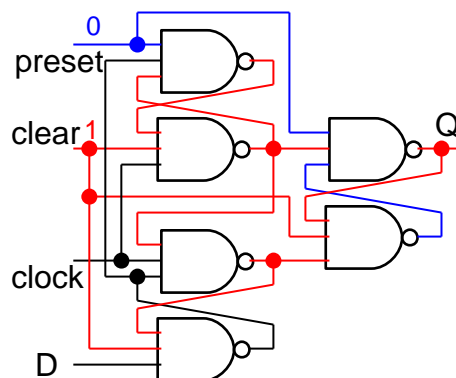
Edge trigger D flip-flop with preset and clear

- **Preset:** force output value to 1
 - Not that this signal is under negative logic
- **Clear:** force output value to 0
 - Not that this signal is under negative logic
- Used in circuit if you want to initialize values



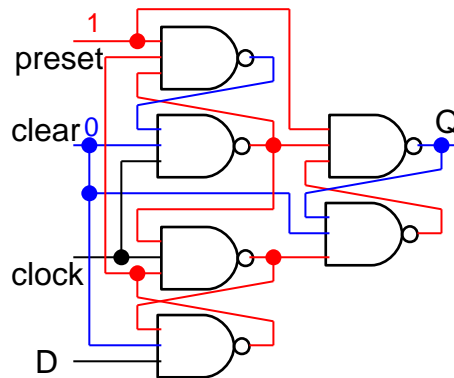
Edge trigger D flip-flop with preset and clear (preset)

- The output forced to 1
- The output becomes 1 even if clock pulse has injected
 - S' side of SR flip-flop is negated if clock pulse has injected



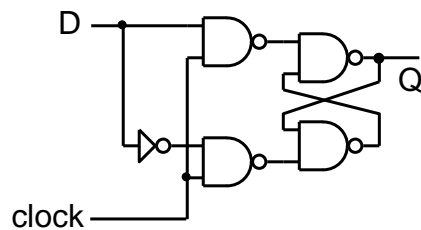
Edge trigger D flip-flop with preset and clear (clear)

- The output forced to 0
- The output becomes 0 even if clock pulse has injected
 - R' side of SR flip-flop is negated if clock pulse has injected



D latch

- A part of D flip-flop
- **Thorough signal** when clock=1
- **Hold value** when clock=0
- In some case, we utilize it in hardware design



clock	D	Q
1	0	0
1	1	1
0	*	Q(previous)



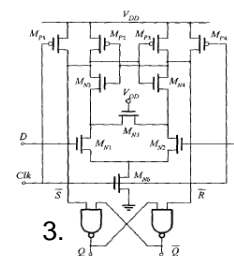
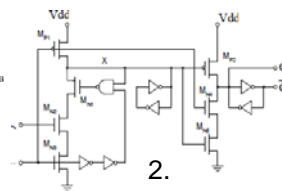
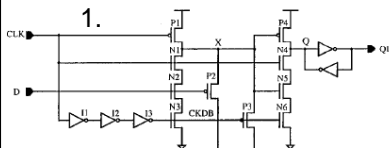
Latch and flip-flop assumption in usual hardware design

- In usual hardware design, we assume following function for latch and flip-flop
- Latch
 - It put through signal if clock signal is enabled
 - It holds last status if clock signal is not enabled
- Flip-flop
 - It updates its status by edge of clock pulse



Explore of faster flip-flops

- Flip-flop is important structure for sequential circuits so that faster one is widely explored
- 1. Hybrid latch flip-flop (AMD K6)
- 2. Semi dynamic flip-flop (UltraSPARC III)
- 3. Sense amplifier based flip-flop (Alpha 21264)



Outline

- Flip-flops and latches (for temporal value)
 - SR flip-flop and its variations
 - D flip-flop and its variations
 - D latch
- Random Access Memory (RAM) and related structure (for storage)
 - Basic organization of RAM
 - Static RAM (SRAM)
 - The other RAMs
 - Content Addressable Memory (CAM)



What's required for storage memory?

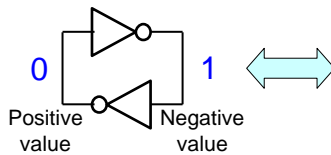
- Data density
 - If we achieve high data density, we can treat large data size
 - Or we can reduce hardware cost in same data size
- Data accessibility
 - We can stuff data to small area if we ignore accessibility, but it is not accepted
 - e.g. tape device has banished because of bad accessibility
 - Usually, we utilize following two types organization
 - Random access memory (RAM) type
 - Content addressable memory (CAM) type



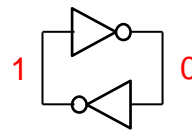
Hold value with inverter loop

- What's a minimized logic which can hold status?
 - > **Inverter (=NOT) loop**
 - Both inverter emphasis signal each other
- How to write data to it?

Represents
"0"



Represents
"1"



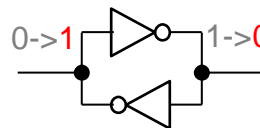
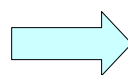
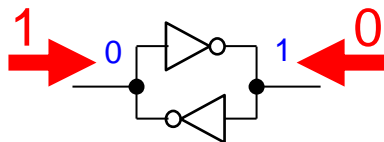
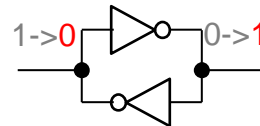
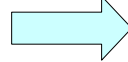
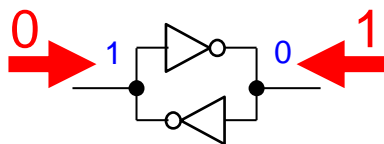
Computing Architecture Lab.
Hajime Shimada

Hardware Design I (Chap. 5)

39

Updating value in inverter loop

- We can overwrite status with **strong signal**
 - Adding signal path which is used for updating
- How to represent strong signal?



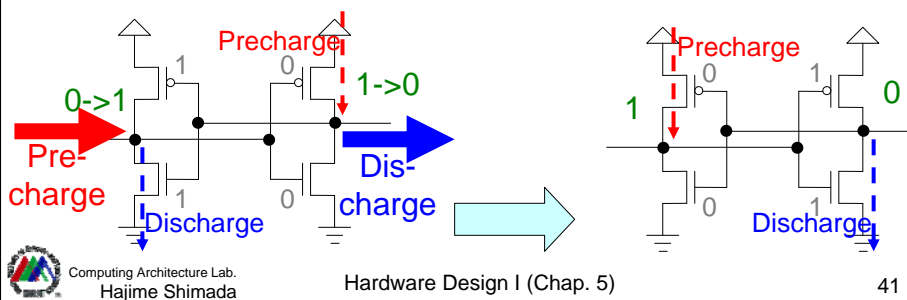
Computing Architecture Lab.
Hajime Shimada

Hardware Design I (Chap. 5)

40

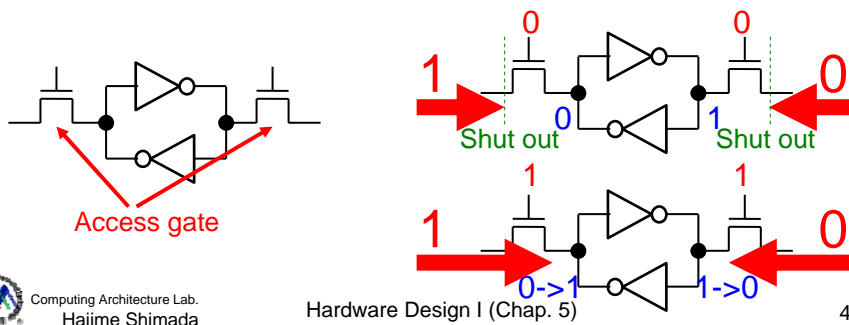
Updating value from electrical viewpoint

- Prepare **powerful current source** to outside
 - If precharge current is larger than discharge current of the inverter, the node becomes 1
 - If discharge current is larger than precharge current of the inverter, the node becomes 0



Access gate (1/2)

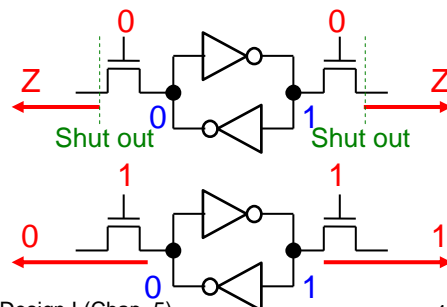
- How to control read/write operation into inverter loop?
 - > Utilize nMOS FET called **access gate**
 - If 0 is applied to access gate, the value does not intrude
 - If 1 is applied to access gate, the value intrudes



Access gate (2/2)

- Also access gate is used for reading internal value
 - If 0 is applied to access gate, the output becomes Z
 - If 1 is applied to access gate, the output becomes a value of inverter loop

c.f.
transmission gate
-> Chap. 4



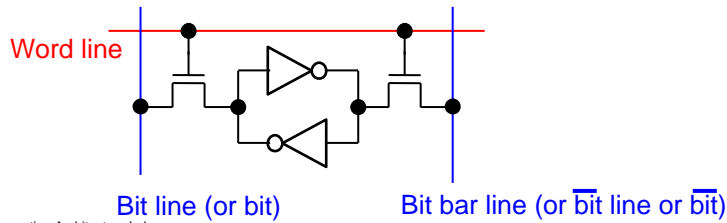
Number of transistor

- The number of transistor becomes 6 in prior organization
 - 2 x INV(2 transistors) and 2 x access gates
- Much less than flip-flops and latches
 - Master-slave D-FF: 36 transistors
 - 8 x NAND2(4 transistors) and 2 x INV
 - Edge trigger D-FF: 24 transistors
 - 6 x NAND2
 - D latch: 17 transistors
 - 4 x NAND2 and 1 x INV



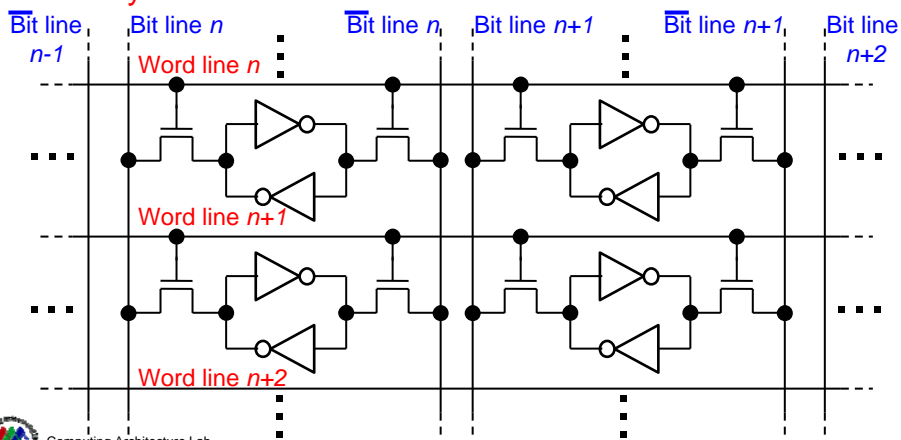
How to connect to outside?

- Input of transmission gate is connected to **word line**
- Outside of transmission gate is connected to **bit line**
 - There's two bit lines which represents positive and negative values
- Usually, we call this organization as a memory cell
- Word line and bit lines are shared between several memory cells



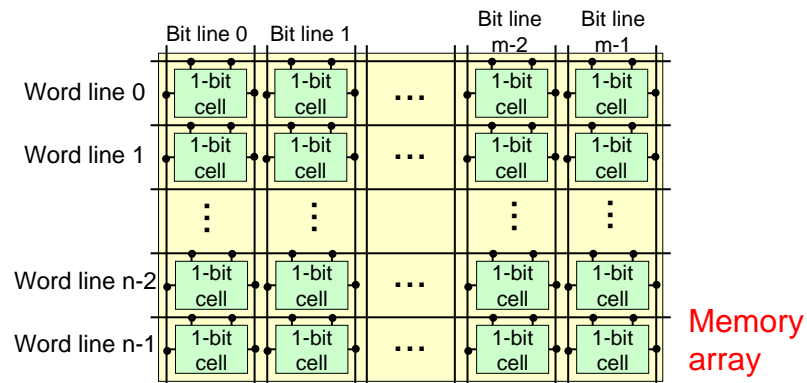
Array of memory cells (1/2)

- By placing prior memory cell, we can create **memory array**



Array of memory cells (2/2)

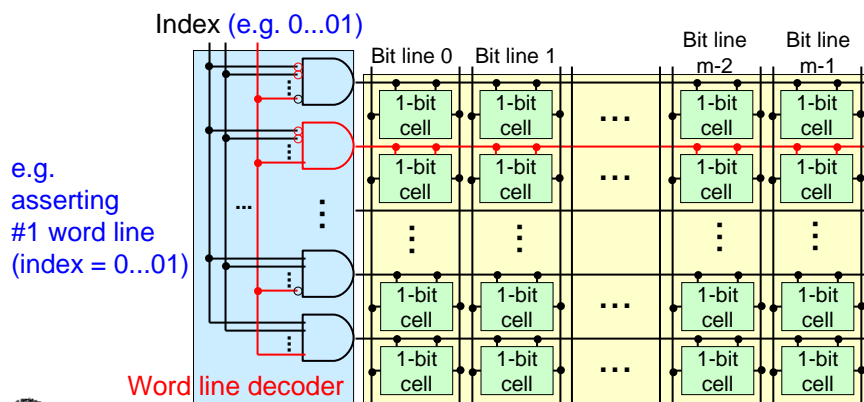
- e.g. A memory array which has n-bit length for vertical and m-bit length for horizontal



How to select one of word lines?

decoder -> Chap. 4

- Prepare **word line decoder** to choose 1 word line
- Length of word line index becomes $\log_2 n$ bits

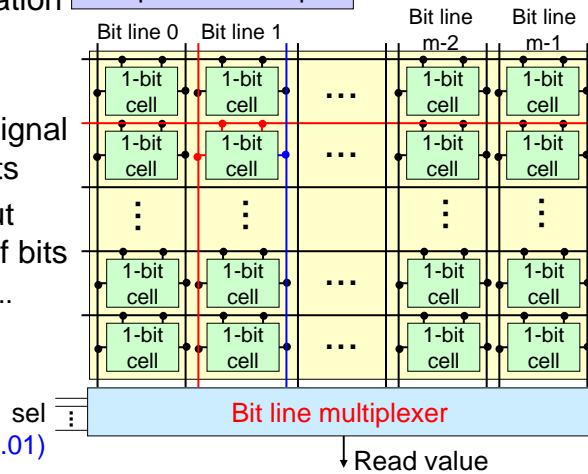


How to select one of bit lines?

- In data read operation **multiplexer -> Chap. 4**
- Prepare **bit line multiplexer**
- Length of select signal becomes $\log_2 n$ bits
- Usually, the output becomes chunk of bits
 - e.g. 8-bit, 32-bit, ...

e.g. selecting #1 bit line (sel = 0...01)

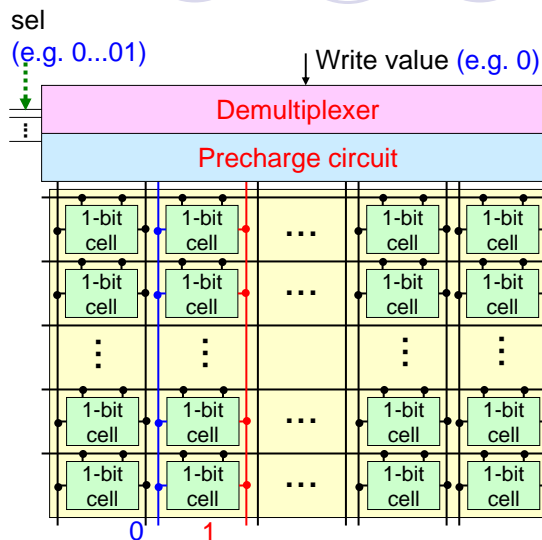
(e.g. 0...01)



How to write data? (1/2)

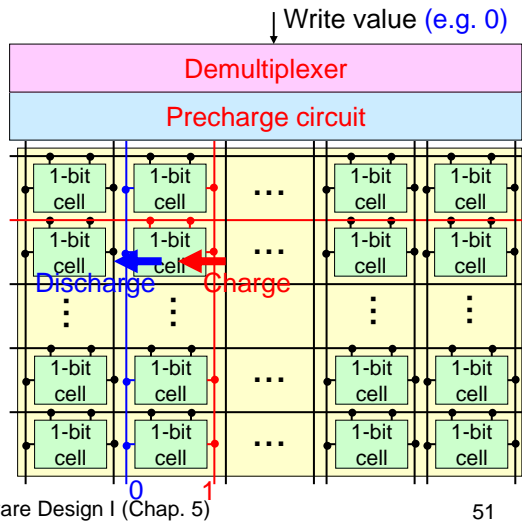
- Prepare **precharge circuit** to write data
 - Precharge bit line and discharge bit bar line if 1 comes
 - Precharge bit bar line and discharge bit line if 0 comes
- Demultiplexer is prepared to deliver value to correct position

-> Chap. 4



How to write data? (2/2)

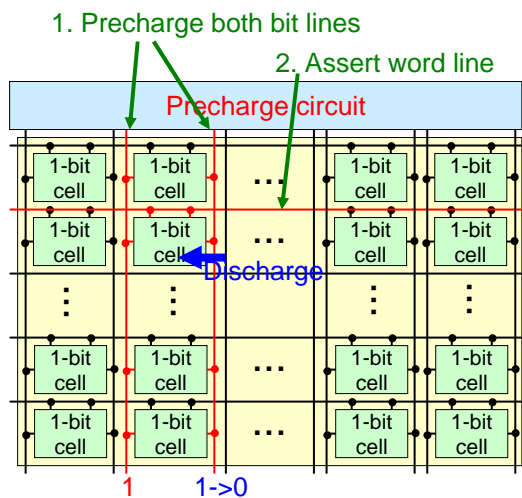
- After asserting word line, the value is written into 1-bit cell
 - Capacitance of bit lines are enough big to overwrite value



How to read value? (strictly)

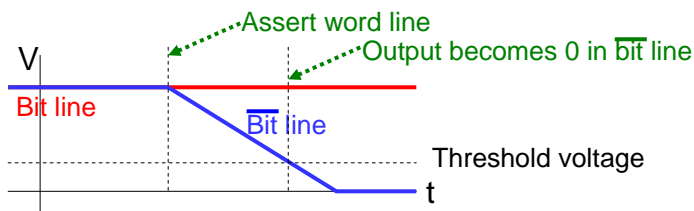
- Strictly speaking, read value operation is done by following operation
 - Precharge both bit lines
 - Assert word line
 - The line connected to "0" side is discharged
- Why?: discharge ability is larger than precharge ability

-> Chap. 1



Sense amplifier (1/2)

- Even if we use discharge, it requires long time to discharge bit line
 - Capacitance of bit line is too large for FET in 1-bit cell
 - To increase data density, we don't want to increase size of FET in 1-bit cell
- > Prepare **sense amplifier** to accelerate output



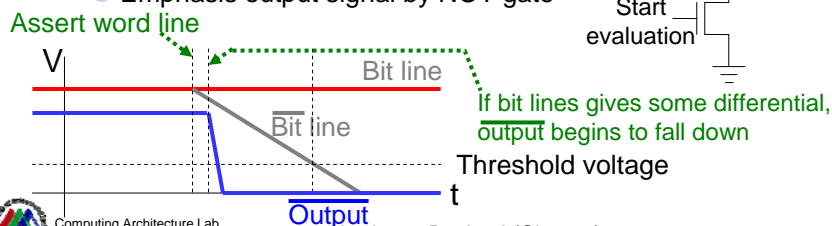
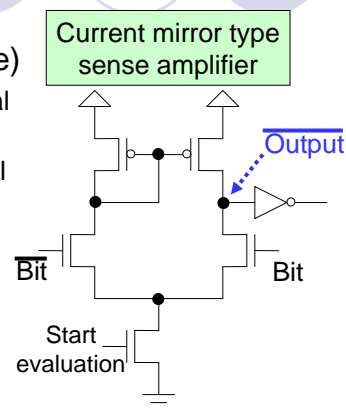
Computing Architecture Lab.
Hajime Shimada

Hardware Design I (Chap. 5)

53

Sense amplifier (2/2)

- Sense amplifier (current mirror type)
 - A circuit which can amplify differential of signals
 - Current flows from V_{dd} to G_{nd} in initial
 - $\overline{\text{Output}}$ becomes intermediate voltage
 - $\overline{\text{Output}}$ becomes 1 if bit begins to fall
 - Output falls to 0 if bit begins to fall
 - Emphasis output signal by NOT gate



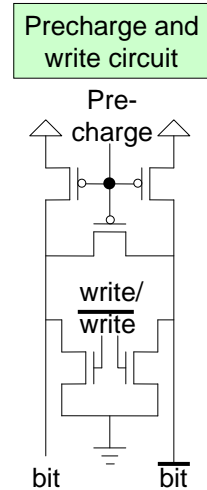
Computing Architecture Lab.
Hajime Shimada

Hardware Design I (Chap. 5)

54

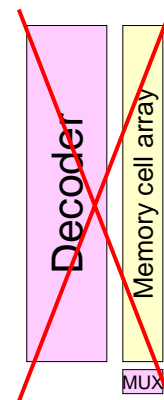
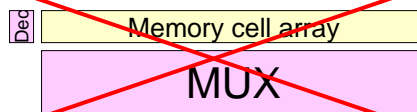
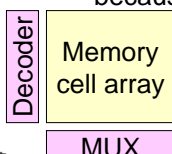
Precharge and write circuit

- Precharge circuit
 - Charge bit lines through pMOS
 - To equalize voltage of bit lines, we prepare pMOS between them
 - If there's slightly voltage difference, sense amplifier amplifies it
- Write circuit
 - Discharge either of bit lines by write value with nMOS
- We have to use **larger transistor** to speedup charge/discharge



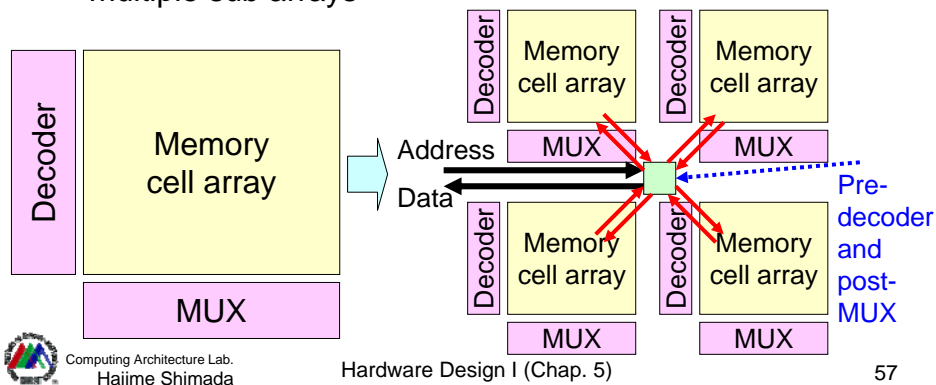
Size of array

- How can we minimize memory array including appending circuits?
- If we extend length of horizontal direction
 - Word line decoder becomes small
 - **But bit line multiplexer and precharge circuit becomes too large**
- Nearly square array is better
 - Strictly speaking, slightly enlarge vertical direction because it only increases decoder



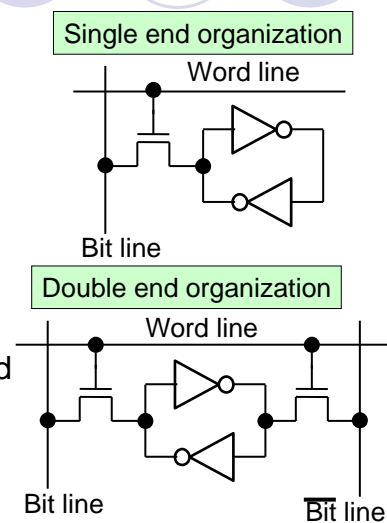
Multiple array organization

- Even if we utilize nearly square array, decoder and MUX becomes too large
- In such case, we can reduce by dividing large array to multiple sub arrays



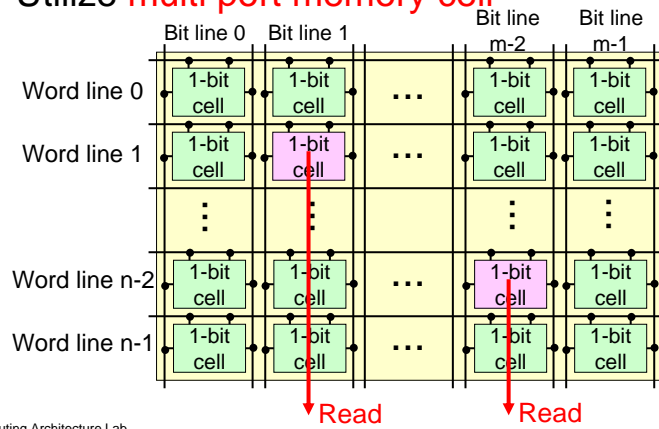
Double end and single end bit lines

- Prior organization is called **double end**
- There's **single end** organization
 - There's only one bit line
 - It can save area
 - But operation speed becomes slower
- Sense amplifier compares voltage between bit line and Vdd



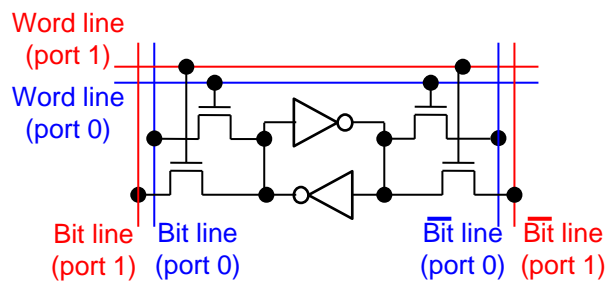
Multi port memory cell (1/2)

- How can I treat multiple read/write request?
 - > Utilize **multi port memory cell**



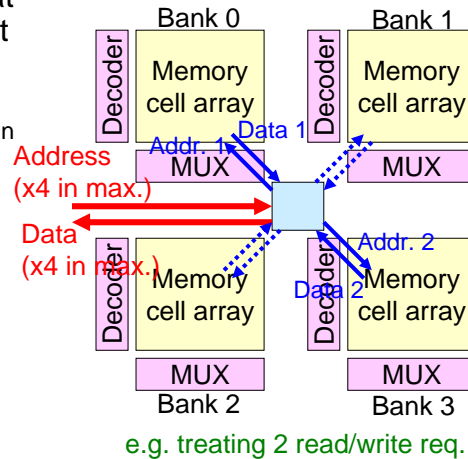
Multi port memory cell (2/2)

- Prepare multiple word and bit lines
 - e.g. 2-port memory cell
 - We can send read/write request either of them
- We have to prepare multiple decoder, MUX, and precharge circuits



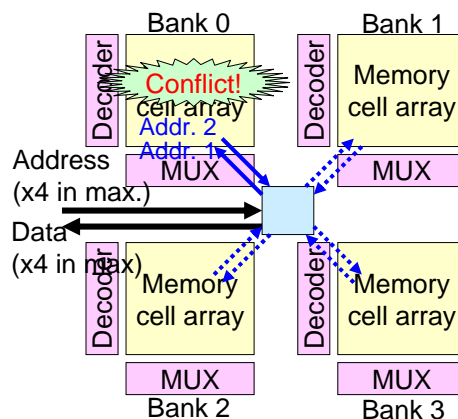
Multi-bank organization (1/2)

- An another method to treat multiple read/write request
 - Allocate data to different bank
 - Usually, consecutive data in memory address are allocated to different bank
 - Allow multiple read/write if data exist in different bank
- Also used for increase memory band width
 - Increase read/write request per unit time
- Also called “interleaving”



Multi bank organization (2/2)

- If read/write requests are concentrated to one bank, we can only allow one of them
 - Called “conflict”
 - Pre-decoder treat arbitration of them
 - Also, hardware which send read/write request must consider data delay caused by conflict



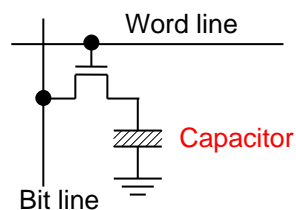
Several RAMs

- The prior organization is called SRAM (Static Random Access Memory)
- There's several type of RAMs
 - Dynamic RAM (DRAM)
 - Flash memory
 - Other advanced RAMs



Dynamic RAM (DRAM)

- Utilize **capacitor** to keep value
 - Bit line discharges slightly if capacitor is not charged on read operation
 - Memory array becomes single end organization
- Area of 1-bit is quite small
 - Used for large storage
 - e.g. **main memory**
- It requires **refresh** operation
 - Because capacitor discharges in proportion to passage of time
 - Read value from memory cell and write it again



Quiz

- How many bits can latest DRAM hold?
 1. 4G bits
 2. 8G bits
 3. 16G bits
 4. 32G bits



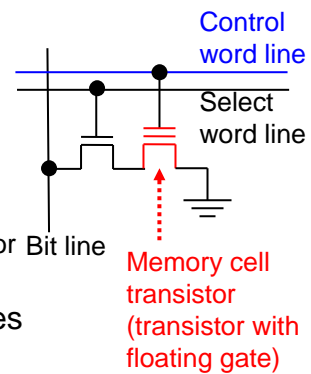
Answer

- 1. **“4G bits”**
 - Comparatively low capacity than following flash memory



Flash memory

- Utilize memory cell transistor (transistor with **floating gate**) to hold value
 - If charge has trapped in floating gate, it represent 1 (high threshold voltage)
 - If charge has trapped, the current flows from bit line to Gnd when it has selected
- The control word line has added
 - Send write signal to memory cell transistor when we update it
- Latest flash memory represents values of multiple bits in one memory cell
 - Utilize 4 voltages when representing 2-bit



Quiz

- How many memory cells can latest flash memory hold?
 1. 4G memory cells
 2. 8G memory cells
 3. 16G memory cells
 4. 32G memory cells



Answer

- 4. “32G memory cells”

- By representing 2-bit values to one memory cell, it can hold 64G bits data

- Further technique

- Stack several silicon die in same package
- Represent 3-bit values with one memory cell



Advanced RAMs (future RAM?)

- MRAM

- Utilize magnetic direction to represent 0 and 1

- PCRAM

- Utilize status of thin membrane (crystal or amorphous)
- The 0 and 1 are detected by difference of resistance

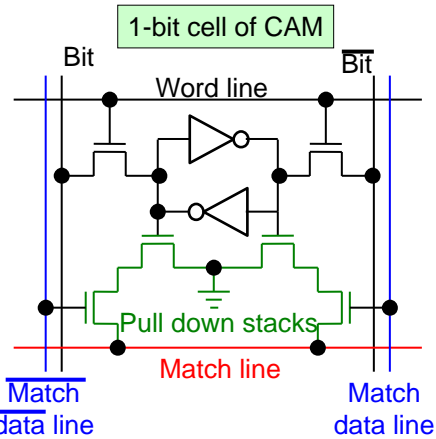
- ReRAM

- Utilize colossal electro-resistance effect
- The 0 and 1 are detected by difference of resistance



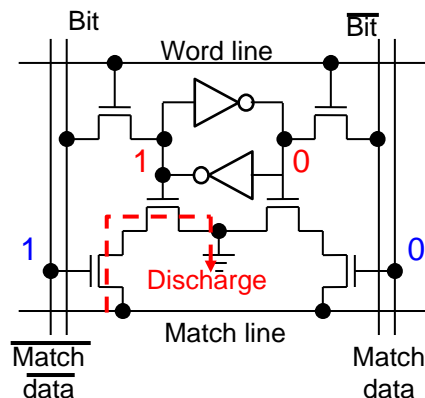
CAM (Content Addressable Memory)

- A circuit which **can compare input value and content of memory**
 - Operate **multiple comparison simultaneously**
 - Usage: packet matching in network router, tag matching, ...
- Achieve by adding some circuits to RAM
 - Match line
 - Match data line and its negation Match data line
 - Pull down stacks



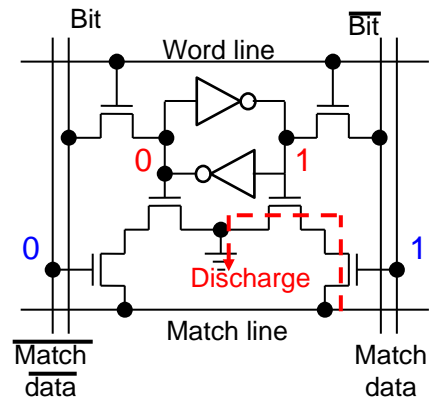
Match operation

- Firstly we charge match line and put match data to match data line
- If it does not match, match line discharged to 0
- Other wise match line keeps 1
- e.g. Content of memory is 1 and match data is 0
 - > Match line is discharged through left pull down stack



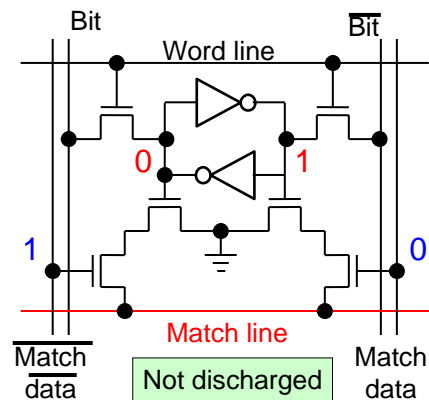
Example of match operation (1/2)

- e.g. Content of memory is 0 and match data is 1
- > Match line is discharged through left pull down stack



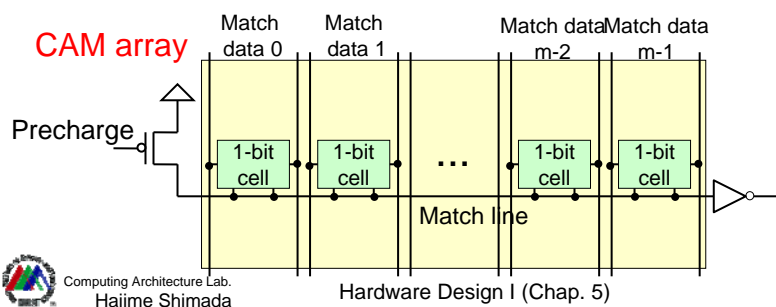
Example of match operation (2/2)

- e.g. Content of memory is 1 and match data is 0
- > Match line is not discharged and keeps 1
 - Either of nMOS FET is conducted in each pull down stack



Multiple bit match operation in CAM array

- By connecting multiple CAM cell to same match line, we can operate multiple bit match operation
- Usually, we add NOT gate to the output of match line
 - To correct negative logic
 - To add current drive ability



Practical circuit utilizing CAM

- e.g. Packet matching of router
 - Put packet information into CAM array
 - Corresponding data (e.g. routing information) is given from RAM array
 - Match line is directly connected to word line of RAM
- CAM allows multiple match so that it sometimes requires priority encoder to choose one of them

