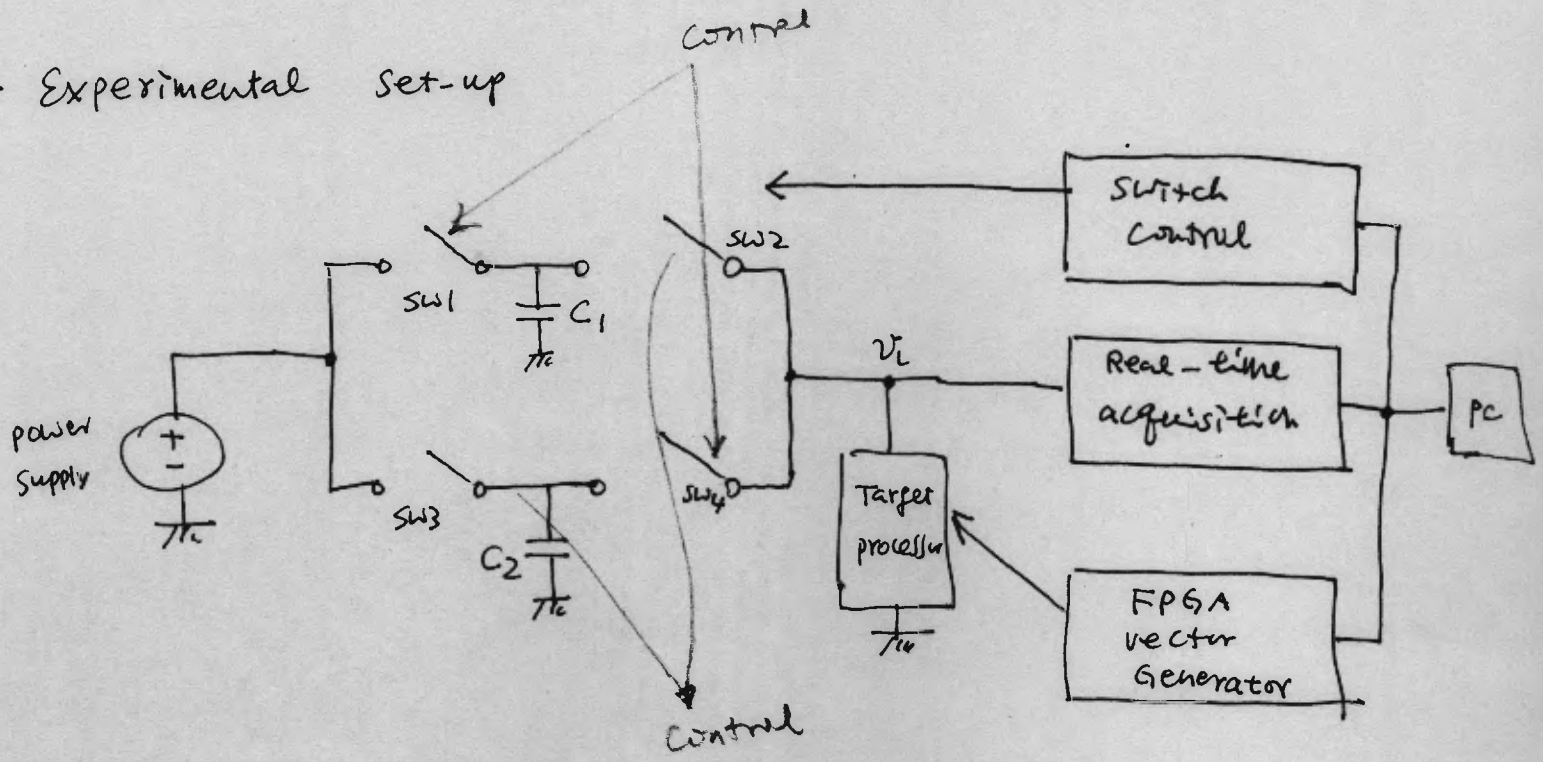
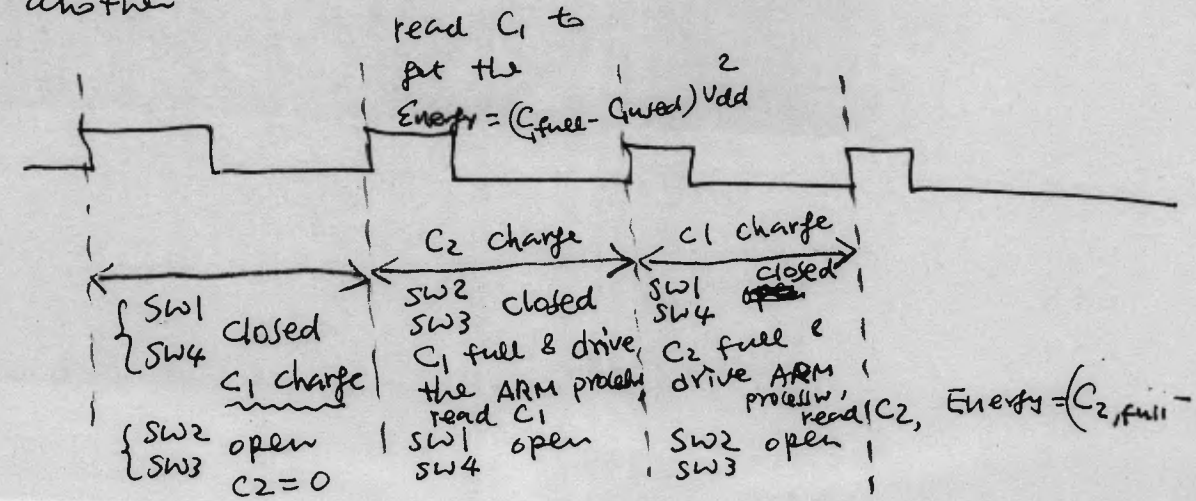
Cycle-Accurate Energy Consumption Measurement & Analysis.

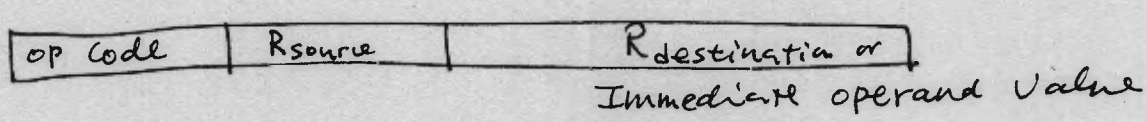
- Based on ARM processor (ARM7TDMI)
- Based on a paper published in ISLPED 2000. pp. 185-190
- Experimental set-up



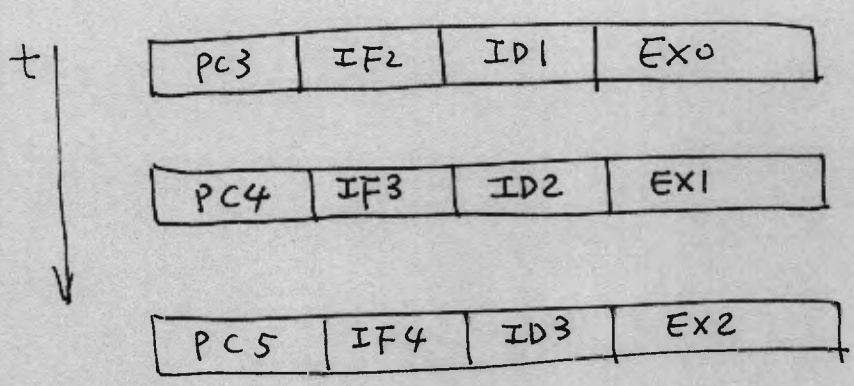
- The switch pairs (SW1 & SW4, SW2 & SW3) alternatively repeat an on/off action
- Each capacitor charges for one clock and discharges for another



- FPGA ~~board~~ is used to generate address, data for the processor
- ARM is a RISC machine with simple instruction formats.

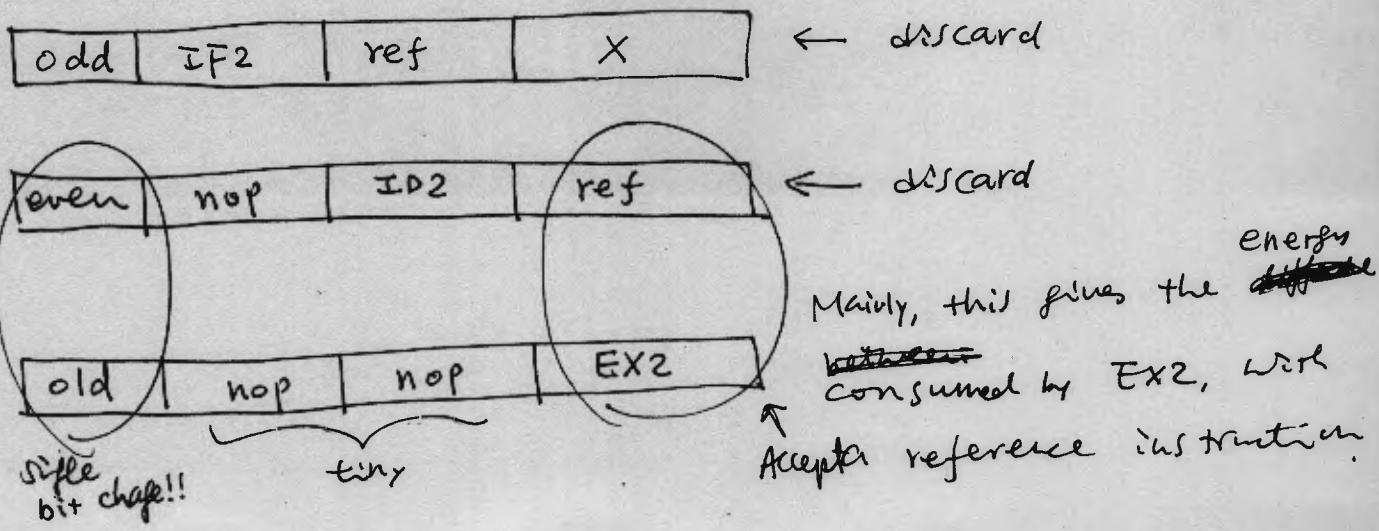


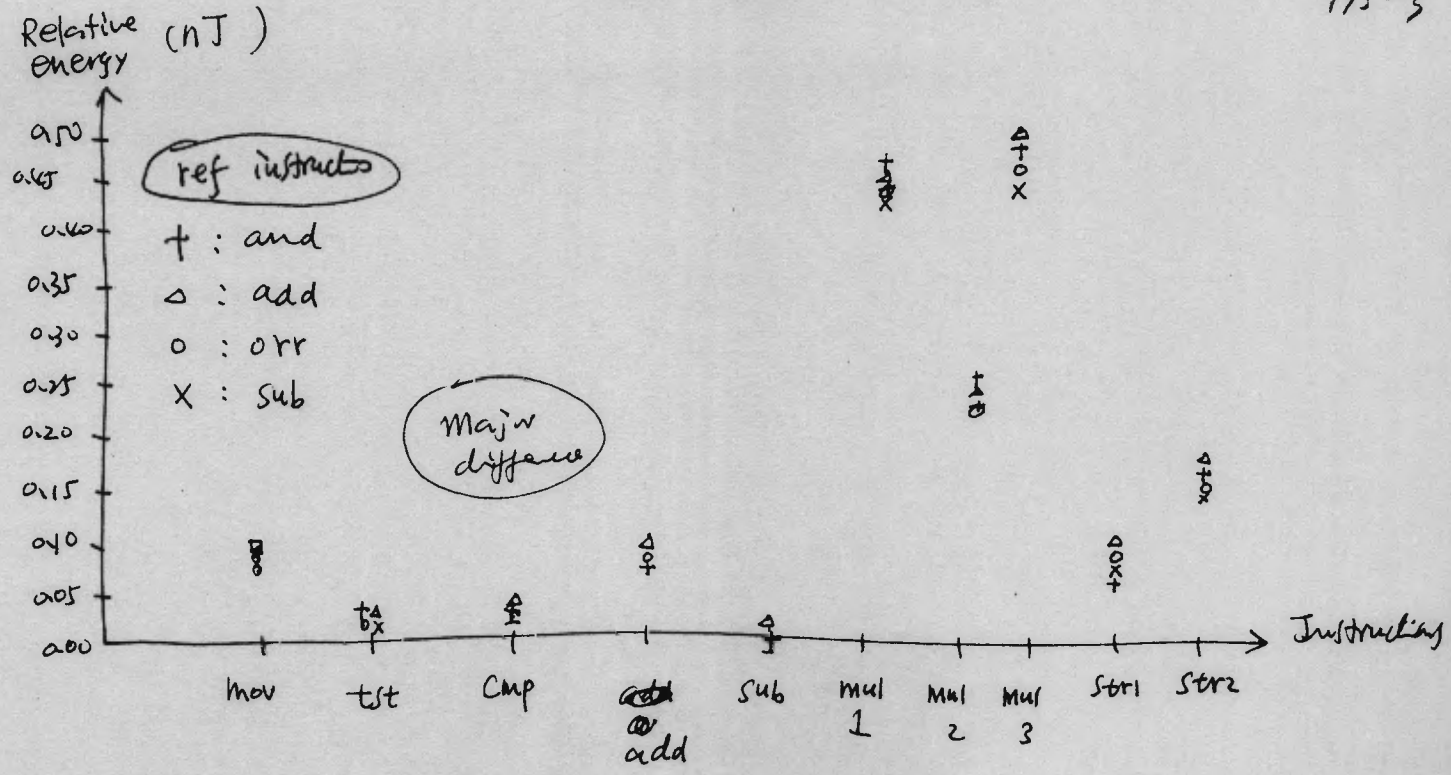
- pipeline of RISC ARM.



- pipeline setup for measuring each pipeline stage.

Execution: (1) Relationship between E(ref instruction) & E(current instruction)
Stage



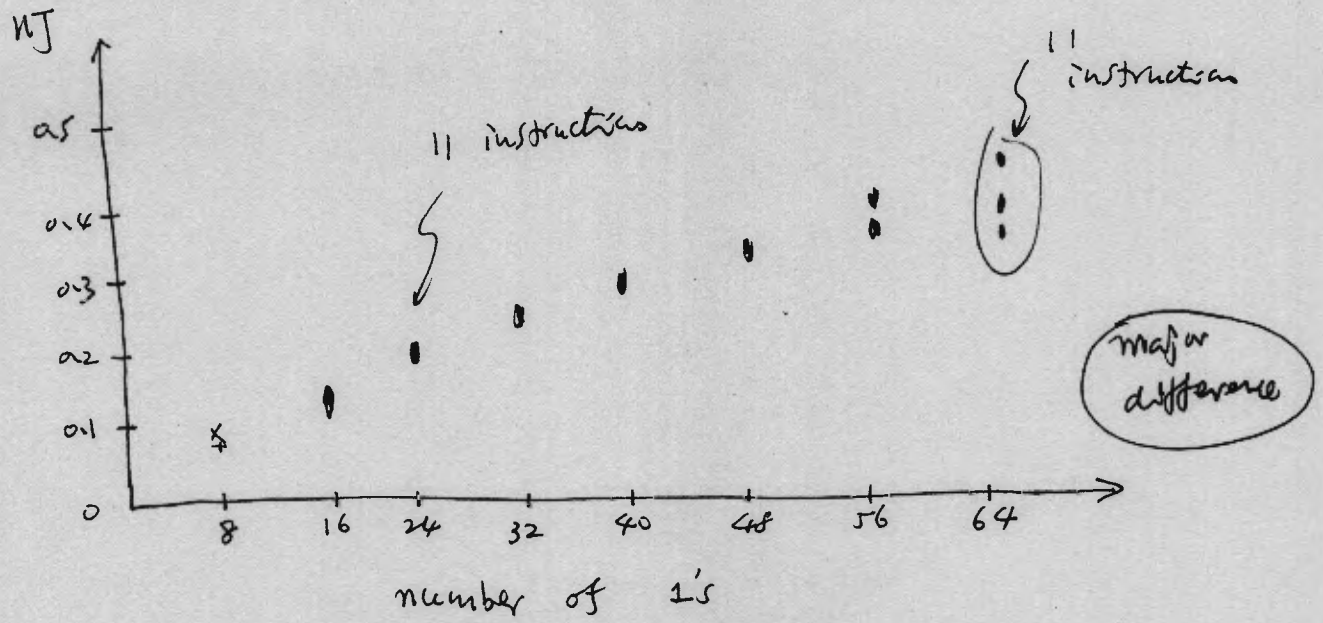


- keep operands fixed, change operators with different (4) reference instructions,
- Conclusion: unique base energy cost is associated with each instruction, regardless of the reference instructions.

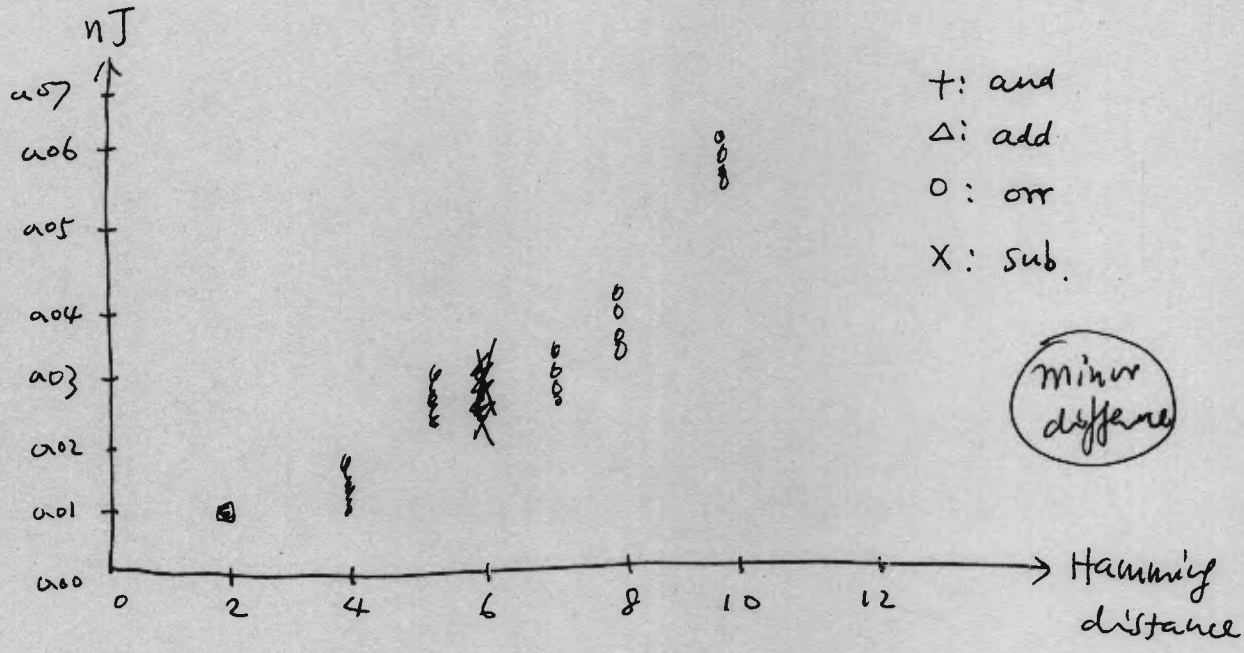
  key: reference instructions are just minor!!

(2) Energy variation due to register values over 11 instructions.

Significant due to dynamic CMOS configuration.

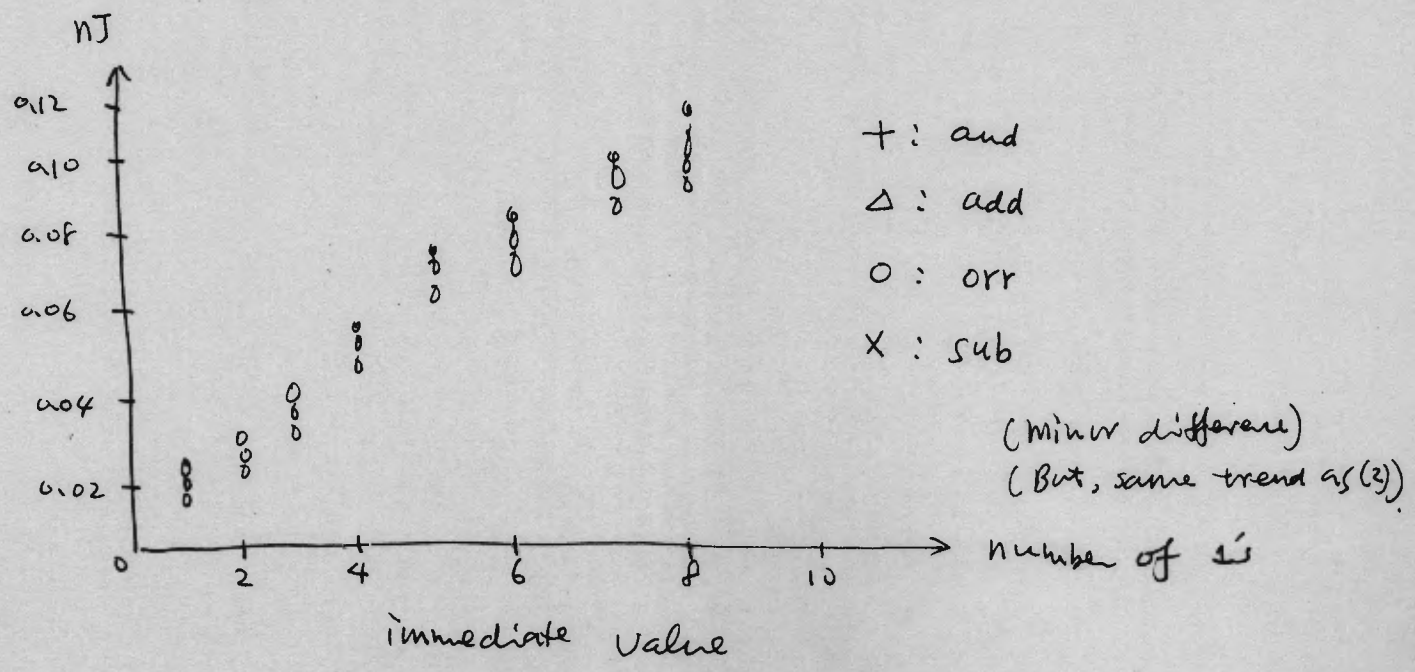


(3) Energy variation by register numbers



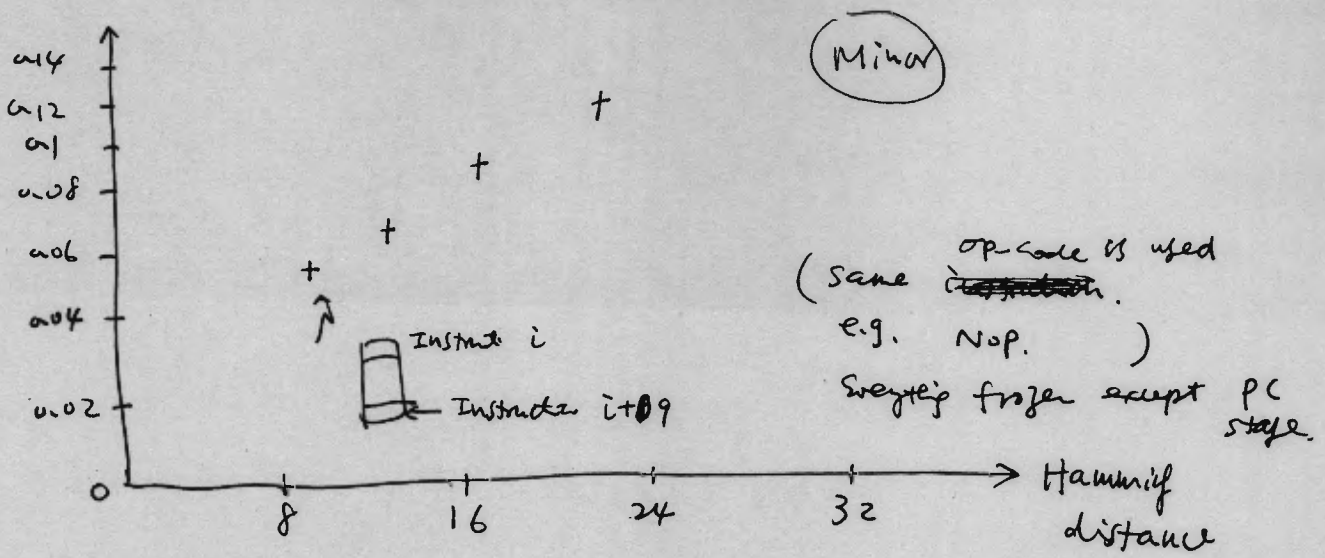
- Ex stage energy is proportional to the Hamming distance between the register ~~name~~ numbers in previous and current instructions

(4) Energy variation due to immediate operand value



PC Stage

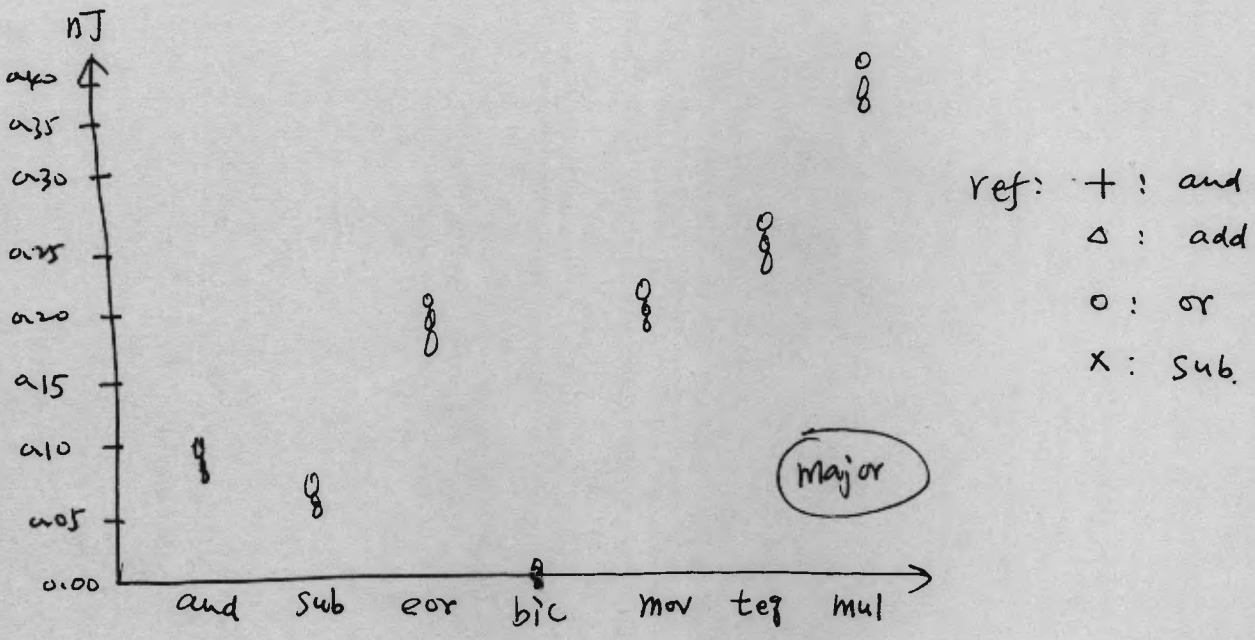
Hamming distance between previous and current instruction fetch addresses is a major concern for PC energy consumption.



ID Stage

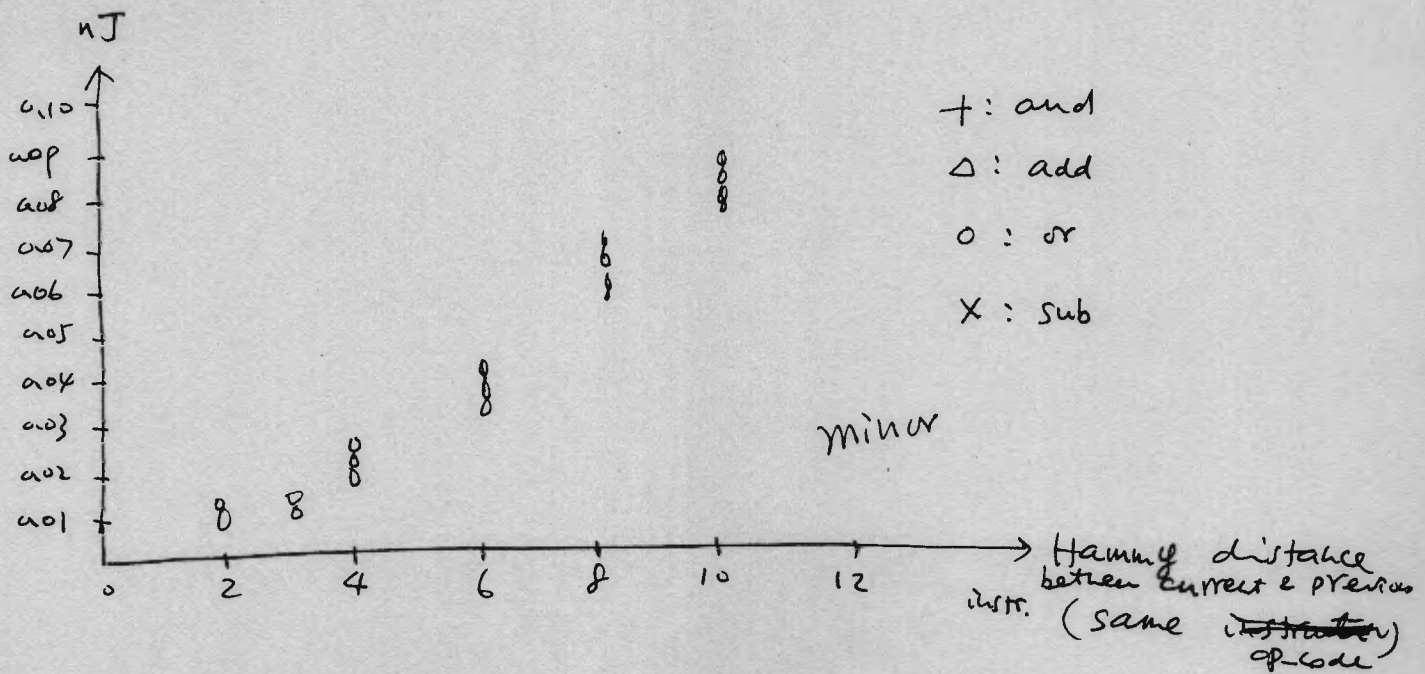
odd	ref	X	X	← discard
even	IF3	ref	X	← discard
odd	nop	ID3	ref	← accept

(1) Base cost of ID stage energy with 4 reference instructions.

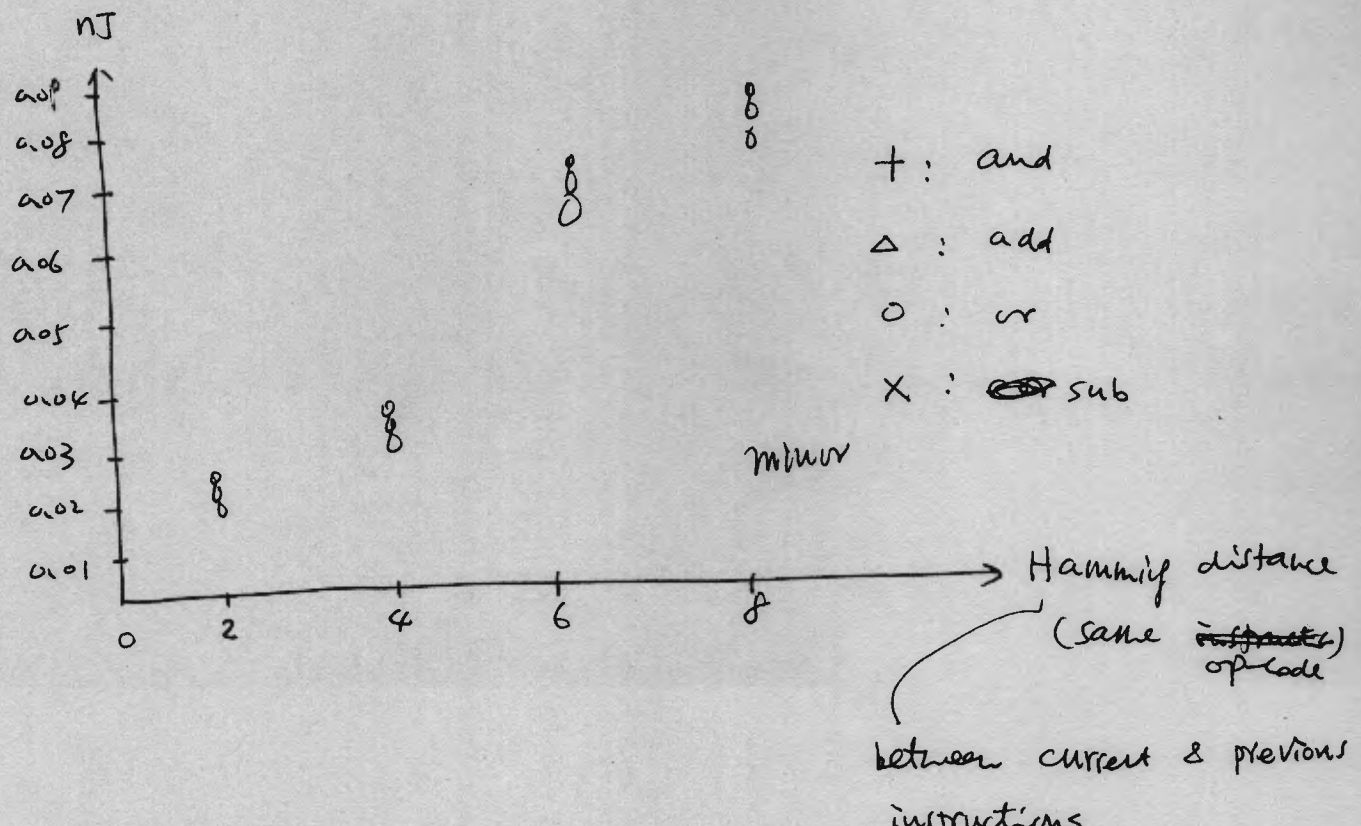


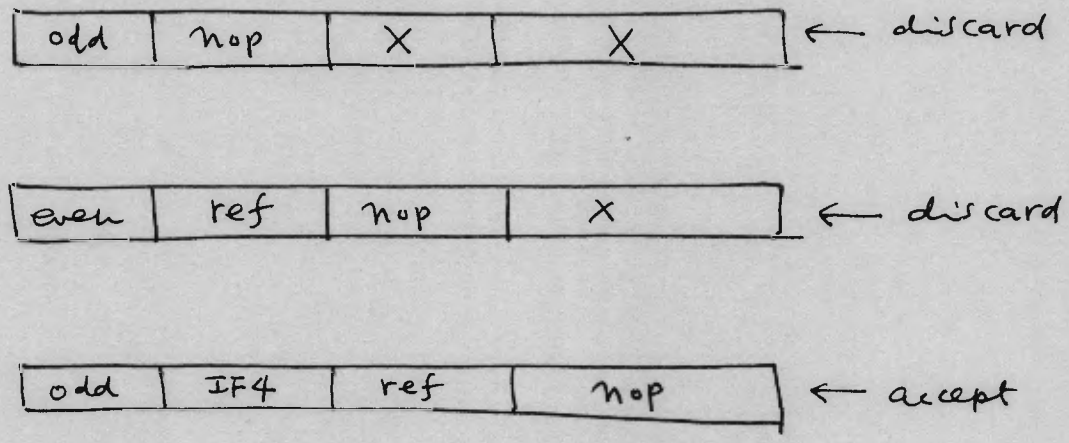
- We have unique base cost of ID stage energy by the op-Codes.

(2) Energy variation in ID due to register number

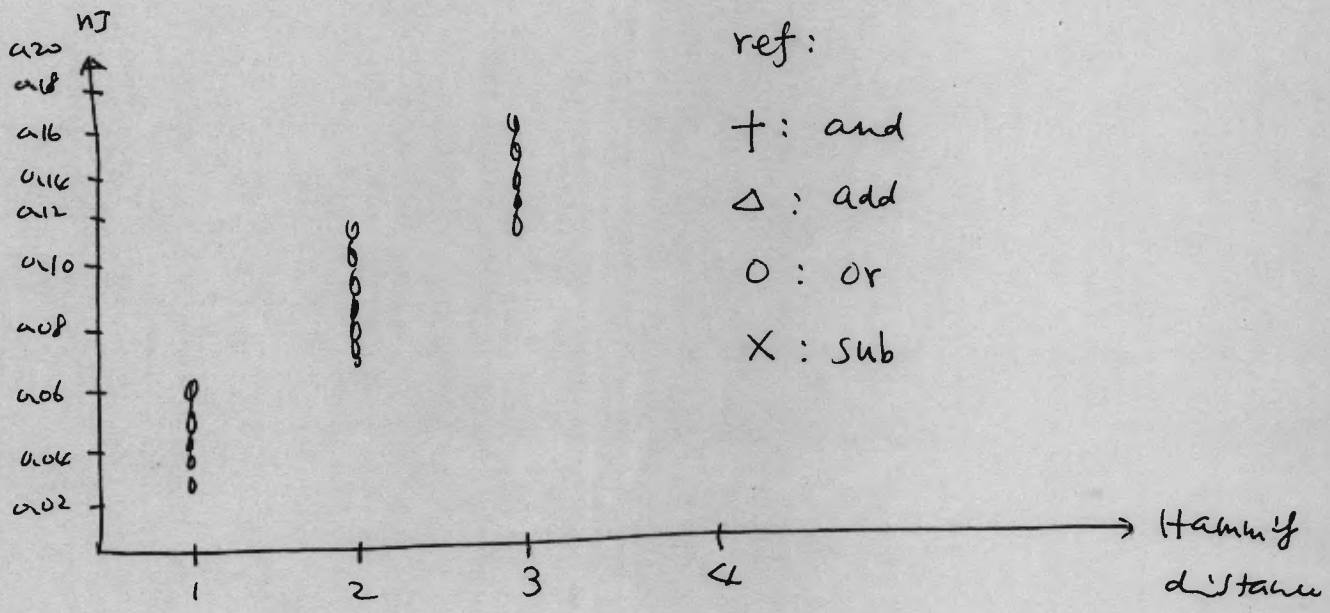


(3) Energy variation due to immediate operand value.



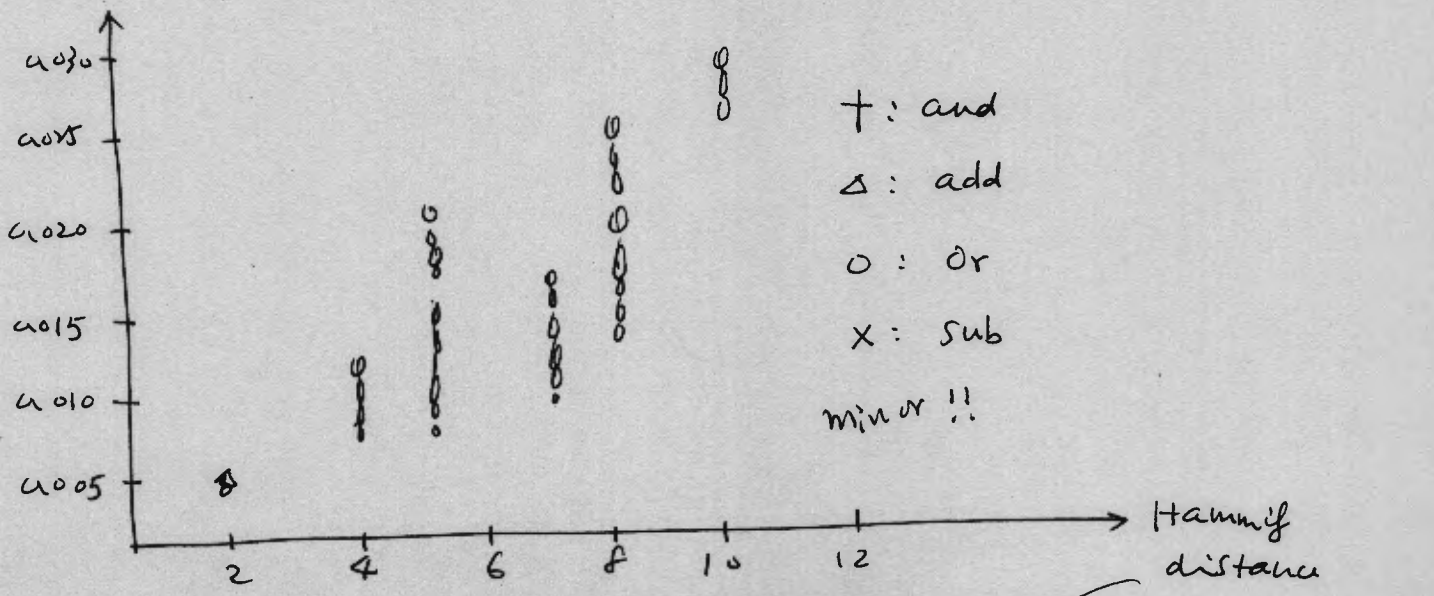


(1) IF energy variation due to hamming distance between op-codes (same operands)



Medium effort

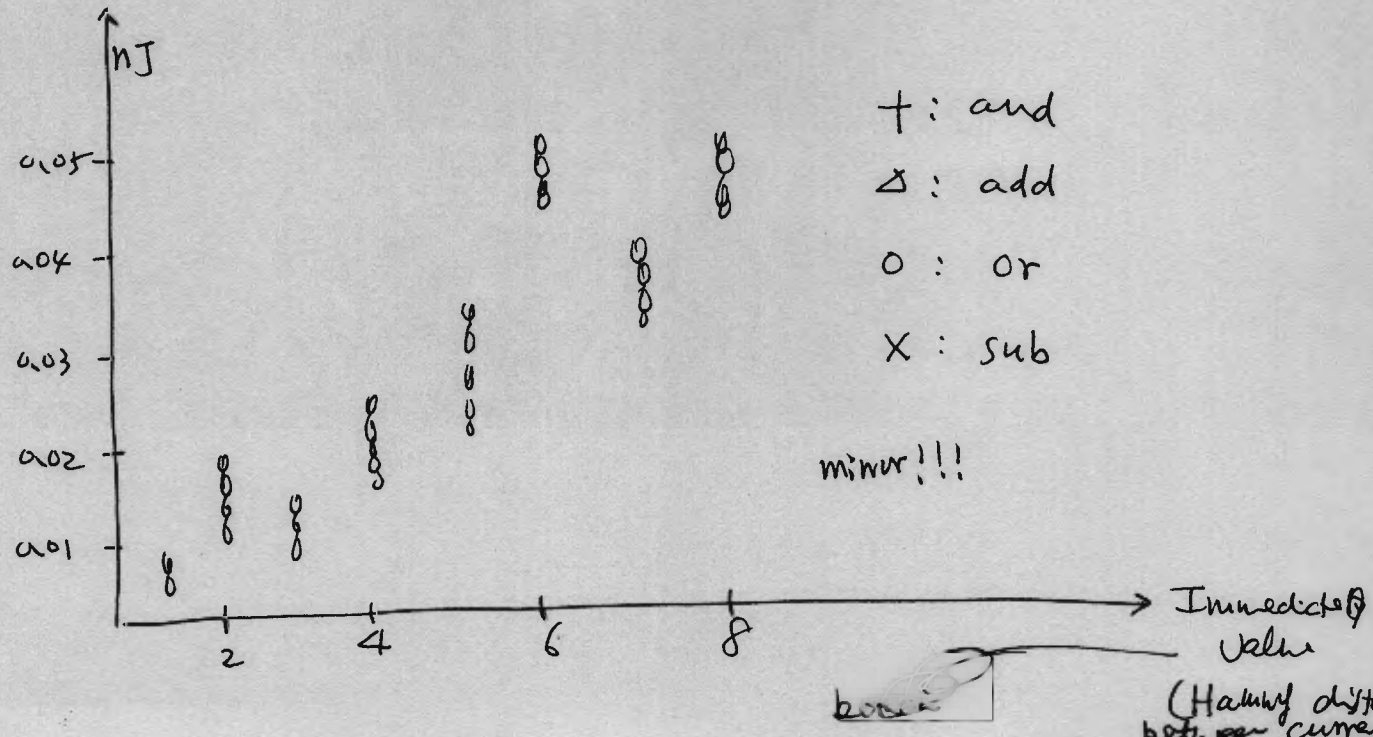
(2) Energy variation due to register number
(Same op-code)



between current & previous instruction

- Not significant amount in energy consumption

(3) Immediate operand value (same op code)

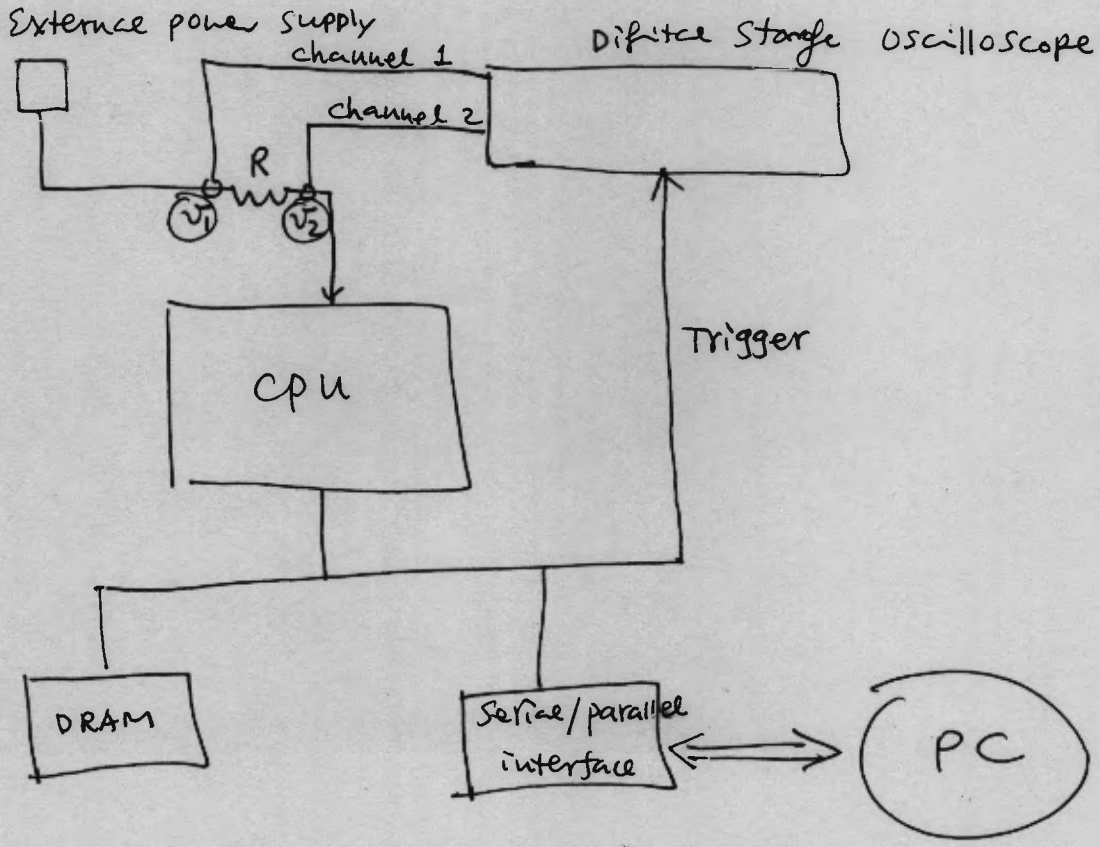


- Summary

① Energy consumption depends on the Hamming distance between values in current and previous cycles or the # of 1's in the current value.

② Each instruction has a based energy cost in the ID & EX stages, not sensitive to the previous pipline status.

How to measure instantaneous power?



$$P(t) = I(t)\,V(t)$$

$$= \left(\frac{V_1(t) - V_2(t)}{R}\right) V_2(t)$$

Software power optimizations

page 176-0

- selection of the least expensive instructions
- minimize the freq (or cost) of memory accesses
- exploit power minimization features of hardware

By Instruction selection & ordering

Cache performance

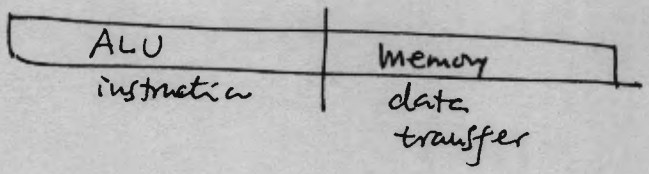
Large code and small cache ⟶ frequent cache misses

⟶ high power penalty

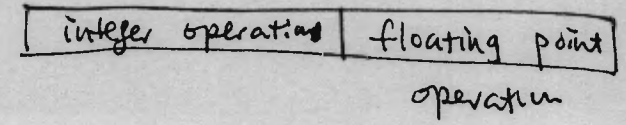
∴ should maximize code density for embedded processors.

Instruction packing

Example: Fujitsu DSP

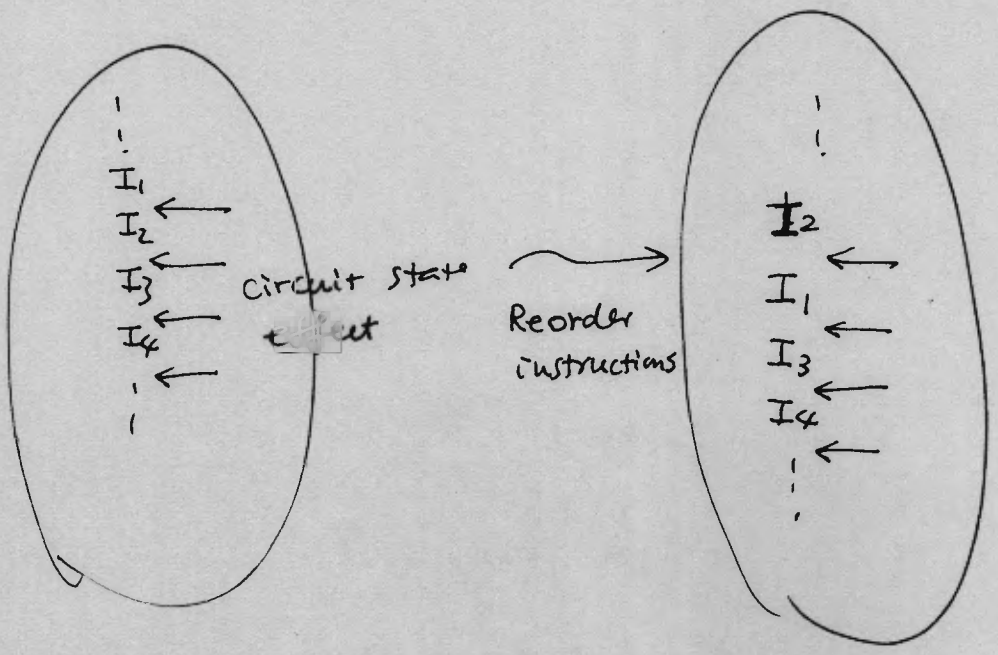


* 50% of energy reduced if instruction fetch (each) overhead is not duplicated



- This method is also used in VLIW & Superscalar architectures.

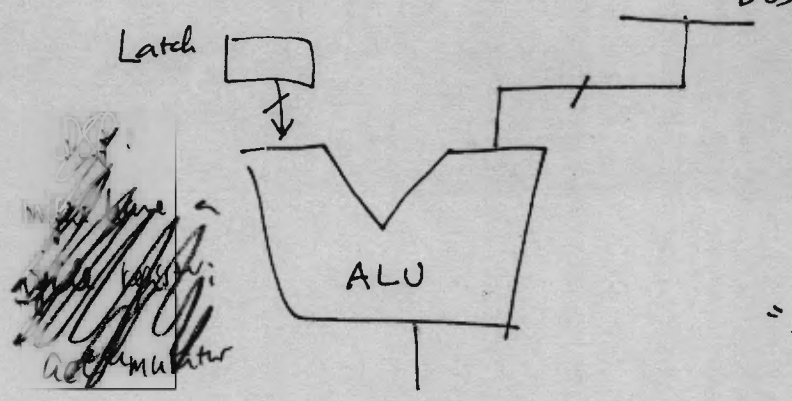
Instruction ordering



to minimize the circuit state power

* This technique is more significant for DSP's than general-purpose architectures.

- Operand swapping



Two consecutive additions

x+7

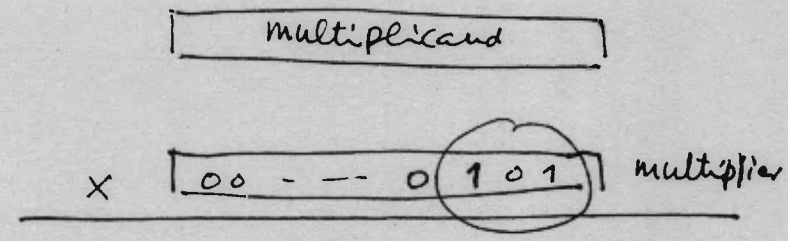
y+7

"7" should be placed in the latch.

otherwise

Latch ← x (BUS)
Bus ← 7 (add)
Latch ← y (BUS)
Out ← (add)

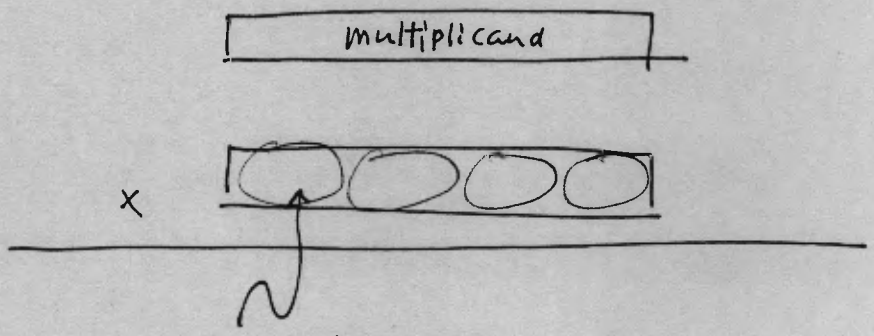
Another example: ~~Booth~~ Shift multiplier



only very limited # of add & shift operations are required.

⇒ Reorder small weight # to the second operand (multiplier)

Booth multiplier



* Different bit pattern determines the # of additions & subtractions
* The # of additions & subtractions ~~for each bit position~~ is called recoding weight
* place the value with low recoding weight to the 2nd position. (multiplier)

Minimizing Memory Access cost

Memory is both a power and performance bottleneck.

∴ It is slow & power hungry.

Try to

- Minimize the # of memory accesses required by an algorithm.
- Make memory accesses as close to the processor as possible.

  register ⟶ cache ⟶ RAM
- Minimize the total memory required by an algorithm
- Make the most efficient use of available memory bandwidth.

  e.g., try to use multiple word, parallel loads.

Example:

```
For i=1 to N do
    B[i] = f(A[i]);

for i=1 to N do
    C[i] = g(B[i]);
```

↑ If B array is large,
register ⟷ memory
transfer
waste power!!
2N memory transfers

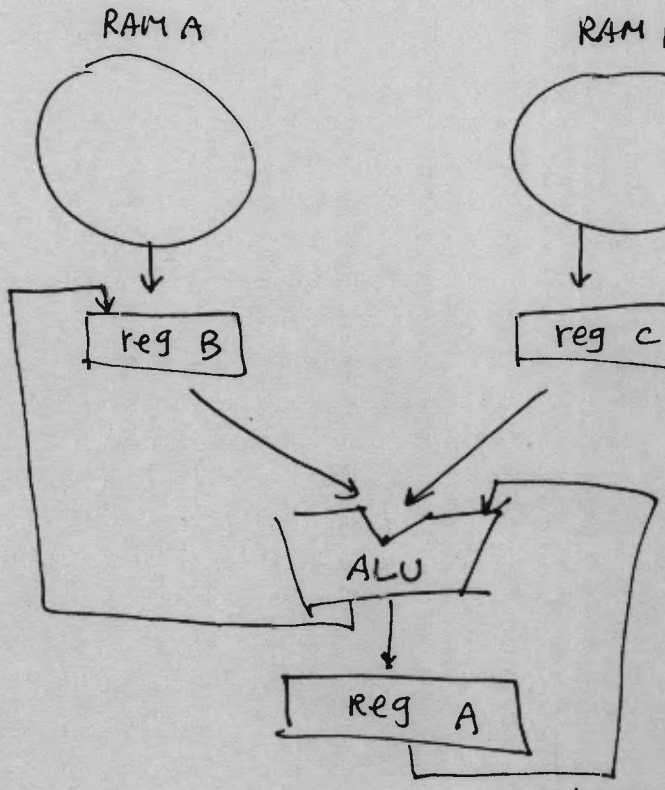
⇒

```
For i=1 to N do
begin
    B[i] = f(A[i]);
    C[i] = g(B[i]);
end
```

↑ compiler can use a register to store B[i].

N memory transfers only.

- Try to maximize parallel loads of multiple words from memory (e.g. dual load)
- Reduce the # of instruction fetch, execute, → good for both Energy & performance
- Example:



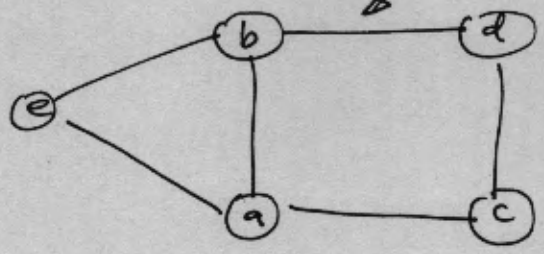
perform (x*y)+z

	(Method a)	(Method B)	(Method c)
RAM A	x, y, z	x, y	x, z
RAM B		z	y
	Load B←x;	DLoad B←x, A←z;	DLoad B←x, C←y;
	Load C←y;	Load C←y;	Load A←z; Mult B←B.C;
	Load A←z; Mult B←B.C;	Mult B←B.C;	ADD A←A.B;
	ADD A←A.B;	ADD A←A.B;	
Energy [pJ]	10.57	9.32	8.85
power [uW]	105.7	93.2	118.0

- Dual loads are beneficial for energy reduction, but not necessarily for power reduction.
  - ∴ Instantaneous power dissipation of dual loads > Single loads.

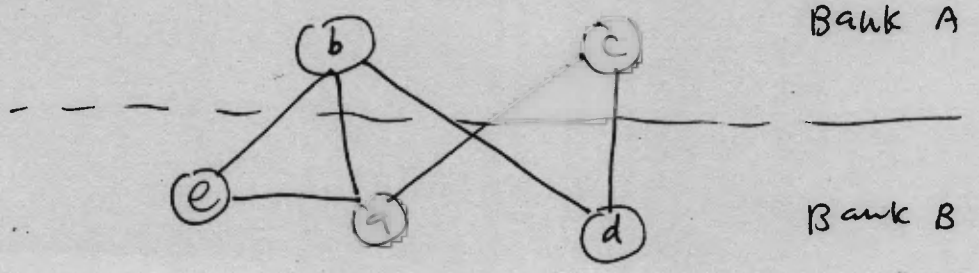
- Algorithm for memory bank assignment

```
ADD   R1 ← a, e
ADD   R2 ← e, b
ADD   R3 ← a, b
ADD   R4 ← b, d
ADD   R5 ← a, c
ADD   R6 ← c, d
```



there exists an instruction with b and d as arguments.

Graph partioning



- The cost of a partition is the # of execution cycles to do all memory transfers.

- Edge in the same partition ⟶ 2 memory transfers

  edge cross the partition ⟶ 1 memory access

- Cost in the above partition:

  T < 2 for ⓐ—ⓔ
  T < 5 for all others

Exploiting Low power Features of Hardware.

- Software Control over power management (instead of gated clock).

* Software designer can use instructions to power-down Some components

* Example:

Standby mode: CPU Core is stopped, but other operations are maintained.

Sleep mode: all operations except the real time clock stops.

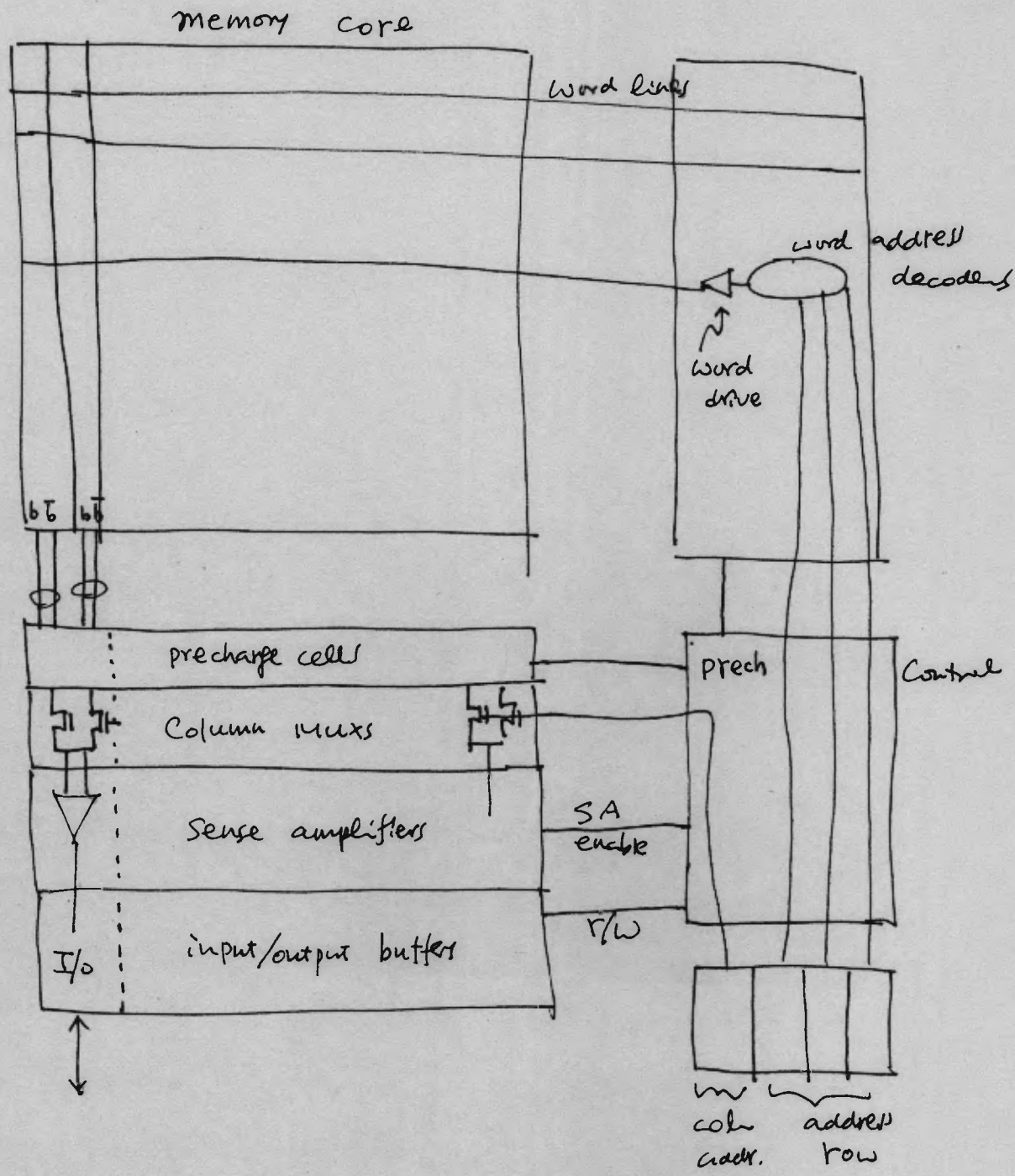
- Advantage:

  - Software designer and compiler can better determine when to remove or slow down a clock.
  - pure hardware power down may make incorrect decision (based on processor activity) ⟶ hurt performance & power ∵ system restoring cost.

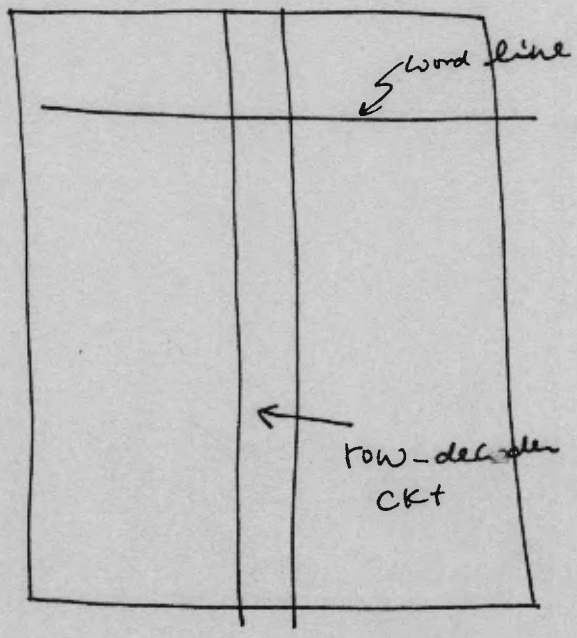
- Disadvantage:

Needs program execution cycles, more costly than hardware-controled power down.

Organization of a RAM.

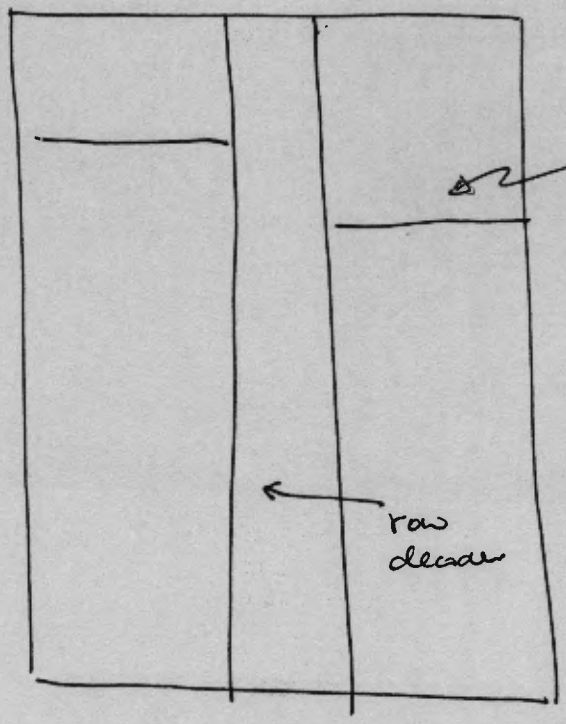


Memory Array Architecture



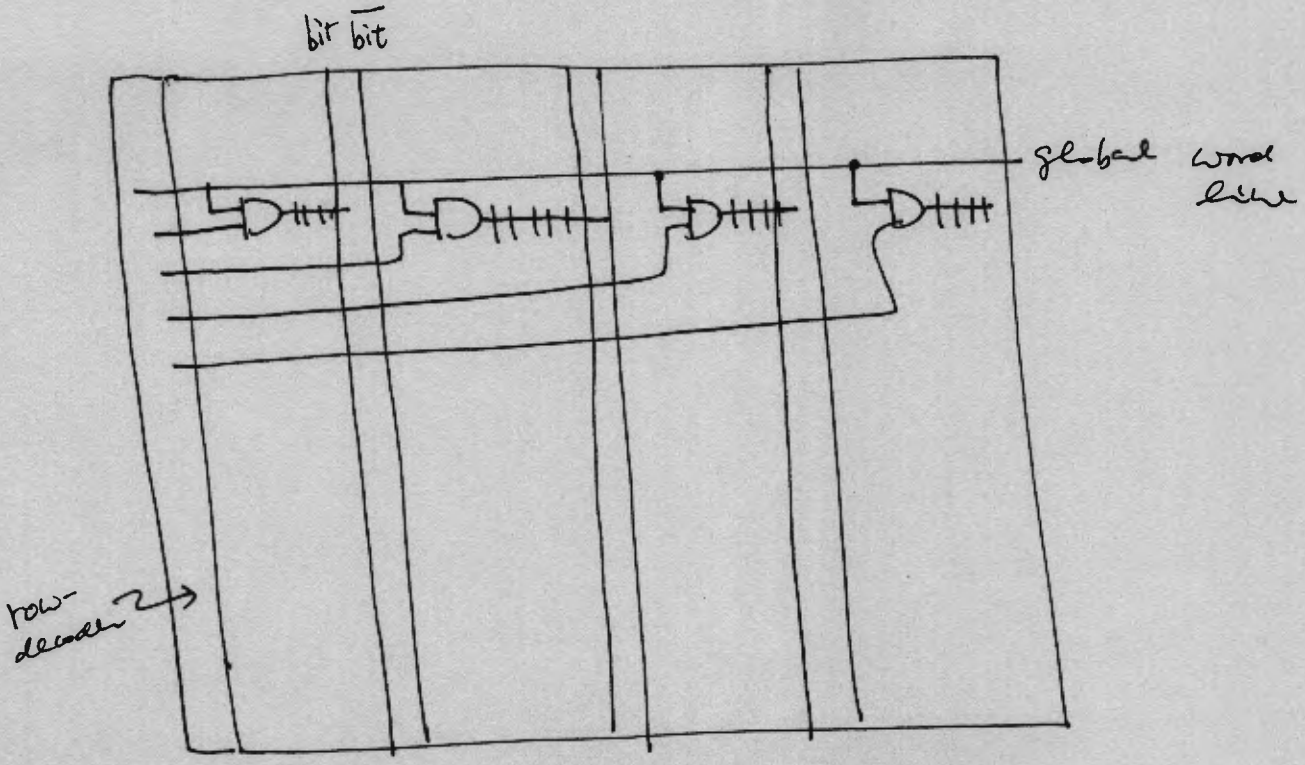
Full plane

- Each row address accesses both sides of the array
- Good for large capacity array.

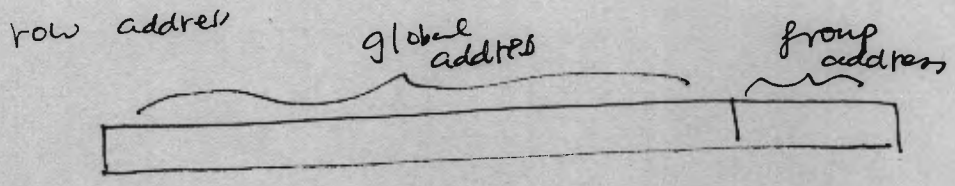


Half plane

- Each row address accesses one side of the array
- Lower power consumption.

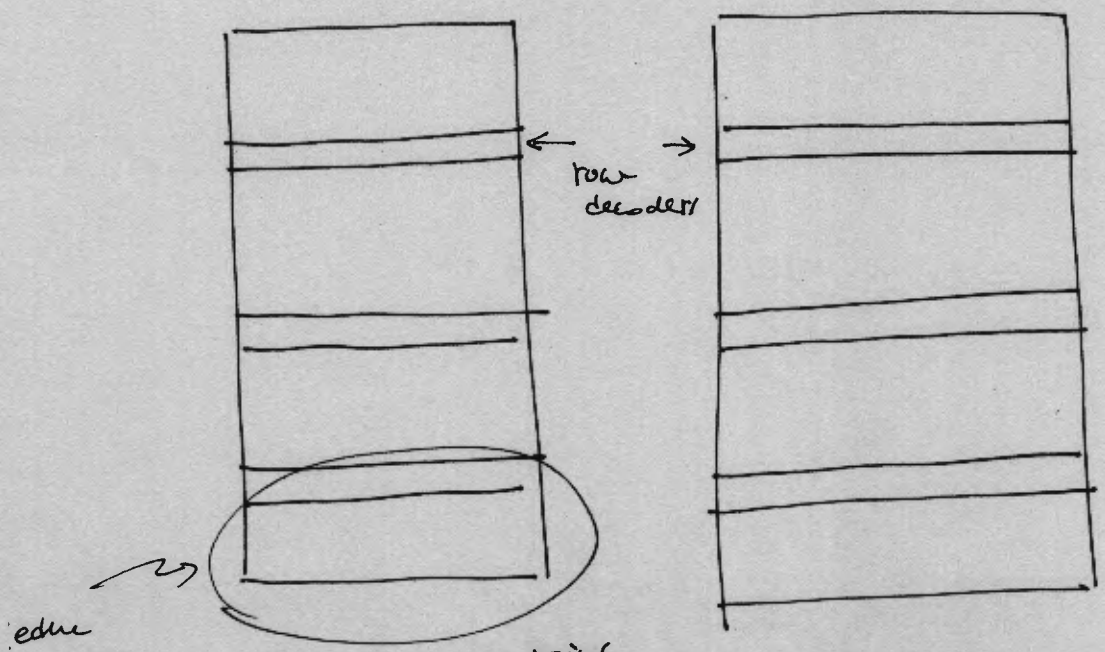


Divided word line



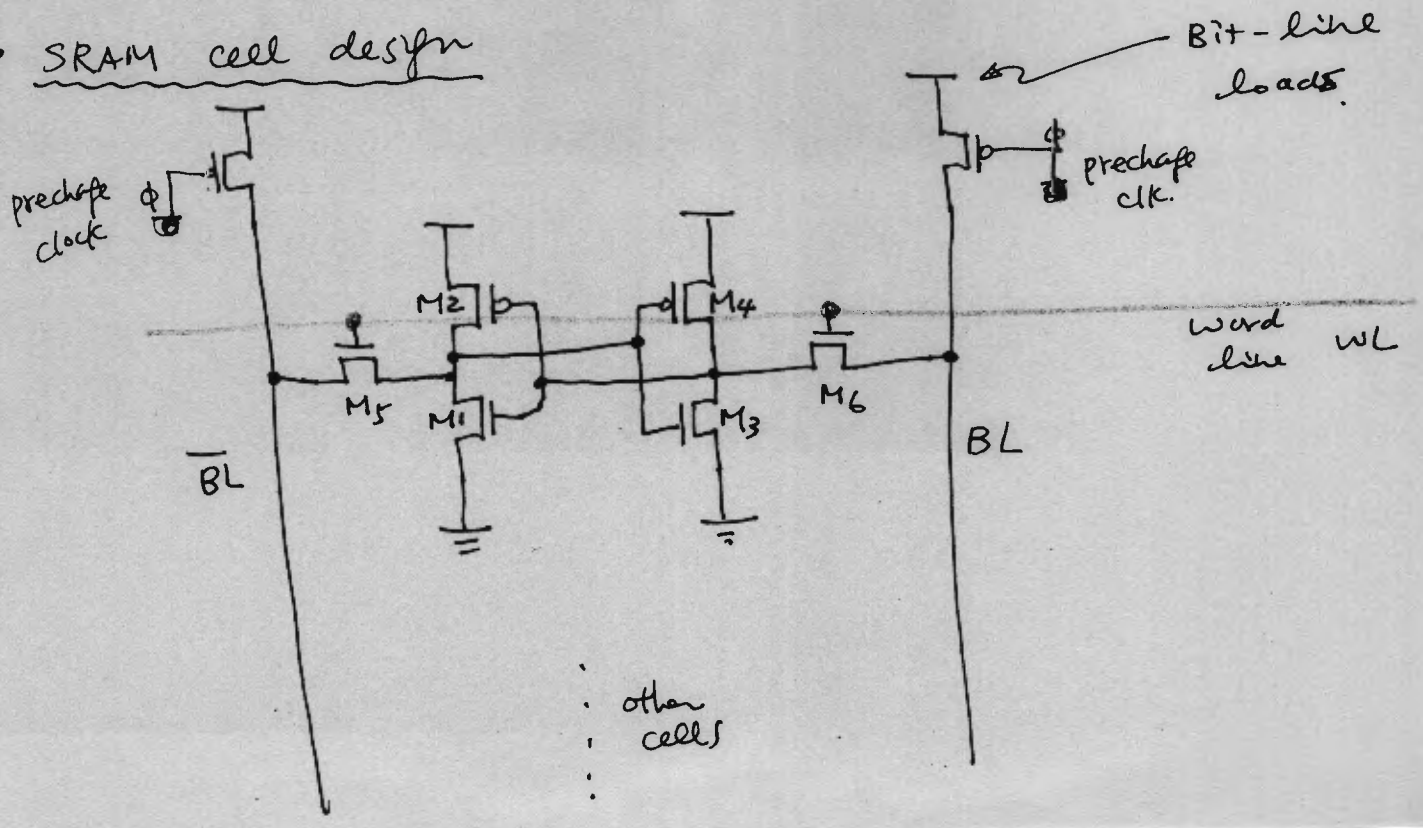
- Needs at least 2 metal lines
- local word line: polysilicon
- global word lines: metal 1
- bit lines: metal 2.

- Block-oriented



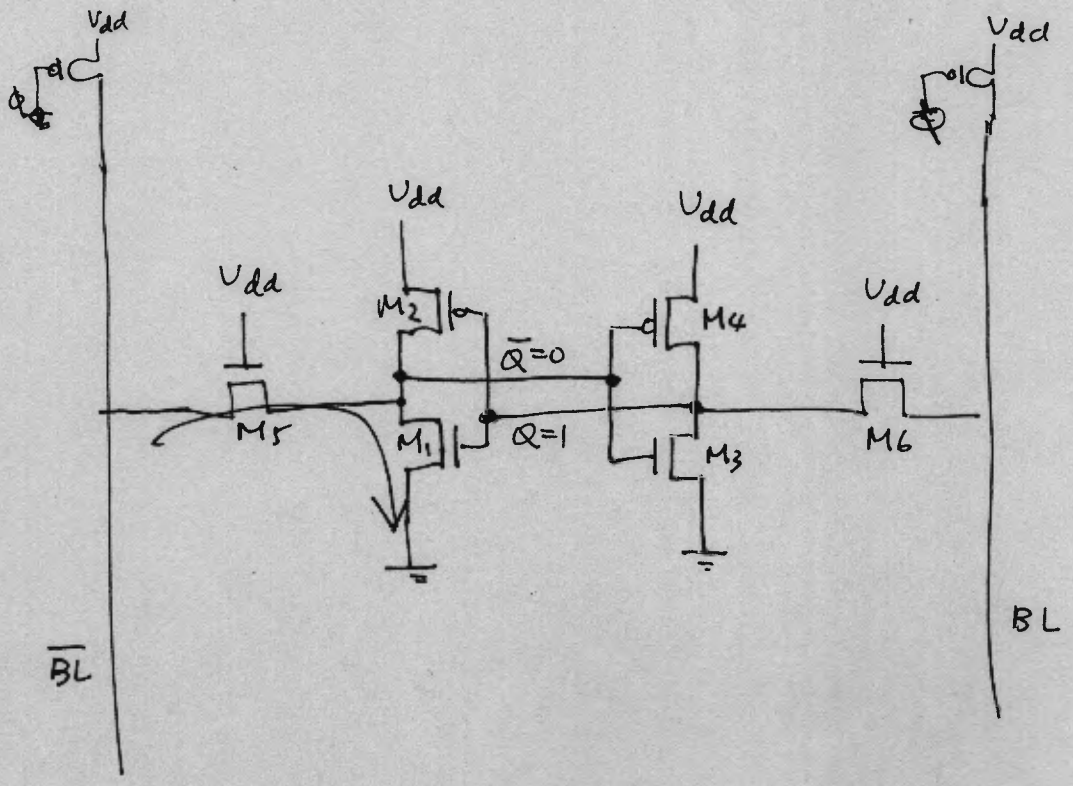
Each block contains its own Block selection (to select a specific block) CKt, x-decoder and (select one of the rows) sense amplifier.

- SRAM cell design



- Transistor sizing is very important.

- Exapl for Read



- Assume the cell stores logic 1

  ∴ $Q=1$, $\bar{Q}=0$

- Before read, both bit lines are precharged to 1.

- We expect $BL=1$, but $\overline{BL}$ will be pulled-down

- Capacitances of $\overline{BL}$ and BL are much larger (PF) than transistor capacitances, so the swing of $\overline{BL}$ will not be too large

  ∴ needs a sensor amplifier to amplify the signal.

- $\bar{Q}=0$ cannot be ~~dest~~ destroyed.

  i.e., the voltage of $\bar{Q}$ can not be pulled up too much.

$$V_{\bar{Q}} = \left(\frac{V_{dd}}{R_{M5} + R_{M1}}\right) R_{M1} \approx 0.3\, V_{dd}$$

$$3.5 \overline{)1.00} \quad 0.03 \quad .05$$

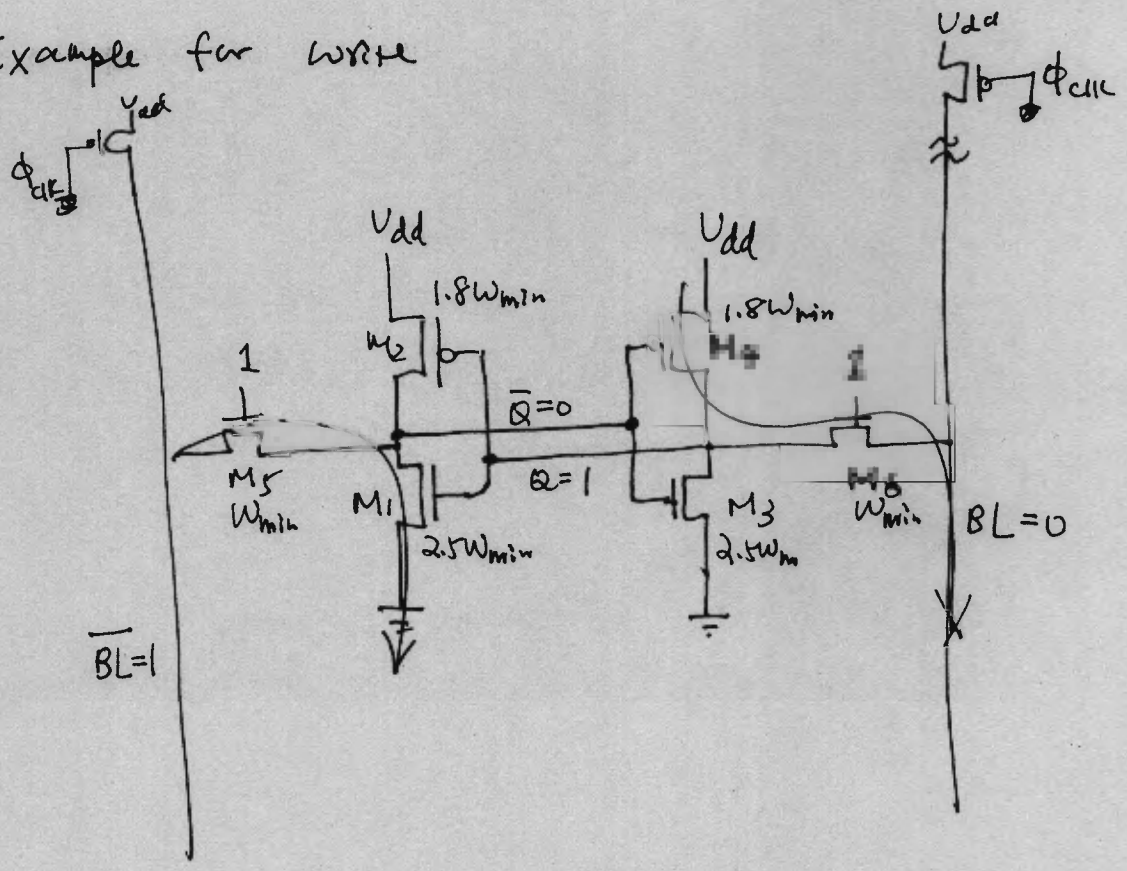
- A reasonable sizing is

  $W_5 = W_{min}$, $W_1 = 2.5\, W_{min}$.

  Similary, $W_6 = W_{min}$, $W_3 = 2.5\, W_{min}$

⊗ P on if $V_{gs} < V_{TP}$

- Example for write



- If write 1, no chage!

- So, let us try to write 0.

- $\overline{BL}$ cannot change the state of $\bar{Q}$ as we discussed.

- $Q=1$ must be charged first by discharging through $M6 \rightsquigarrow BL$ line.

- $Q=1$ must drop below $V_{dd}+V_{TP0}$ such that M_2 can be on. So, there is a sizing problem between M_4 and M_6.

$\frac{5}{0.7}$ / $\frac{}{0.3}$ ③

- Once Q drops to below $V_{dd}+V_{TP0}$, M_2 begins to be on and charge $\bar{Q}$ to 1 and change the state of the cell. (Get Stablized)!!

- M_4 must be ~~large enough~~ small, such that Q can drop to below $V_{dd}+V_{TP0}$.

small (in size)
i.e., its R must be large!!

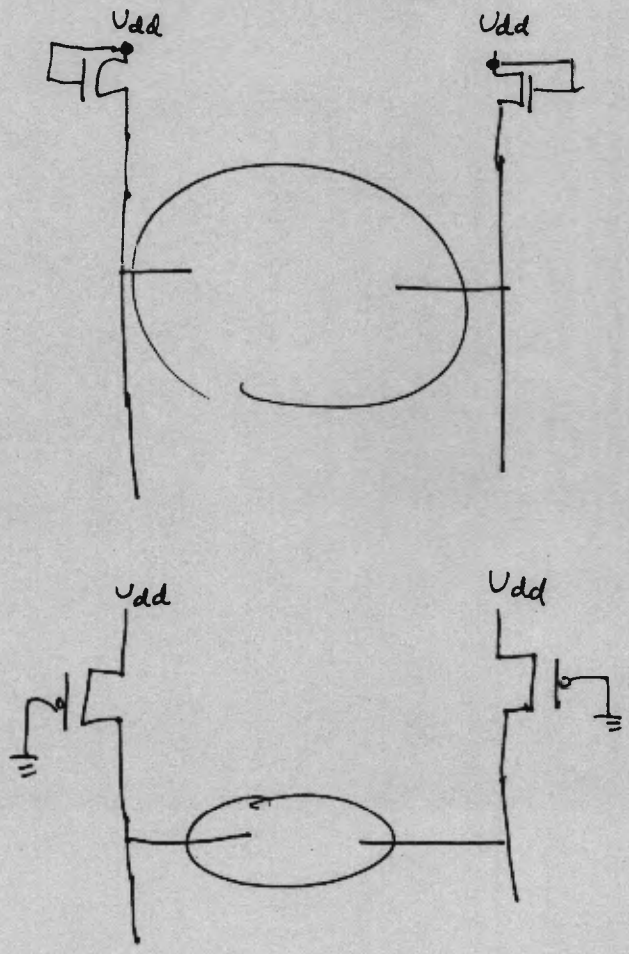
$$\left(\frac{W}{L}\right)_4 \leq 1.5\left(\frac{W}{L}\right)_6$$

This makes $V_{Q=1} < 0.5 V_{dd}$

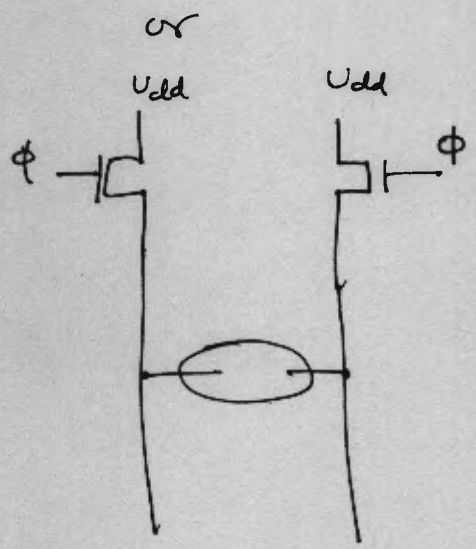
$$\therefore W_4 = W_2 = 1.5 W_{min}$$

- Bit line Load.
  - In fact, the bit lines can be charged to lower than V_{dd} to save power
  - However, if BL (in P.184) is too low, Q=1 cannot be maintained, and the state may be changed.

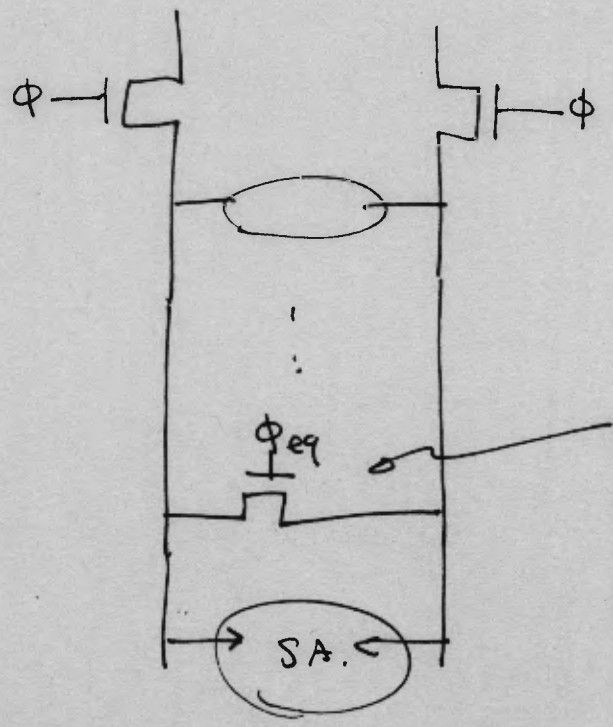
    ∴ Bit line voltage ↑ ⇒ more stable memory state.
  - can have many different designs.



or

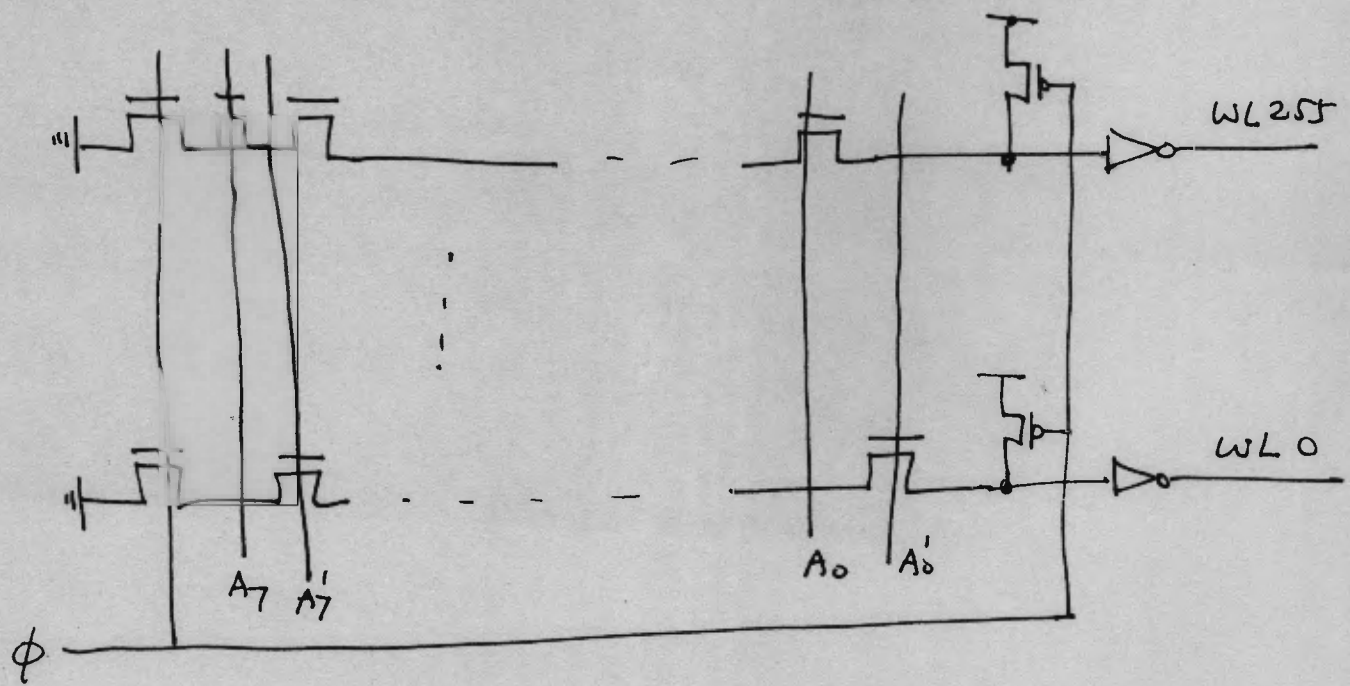


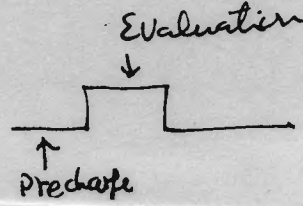
No matter what kind of pull-up ckt, equalizing device must be added.



After each r/w operation, this must be activated for a while to equalize the voltage.

Row Decoder 8 → 256 mapping (Exaple) — NAND-type



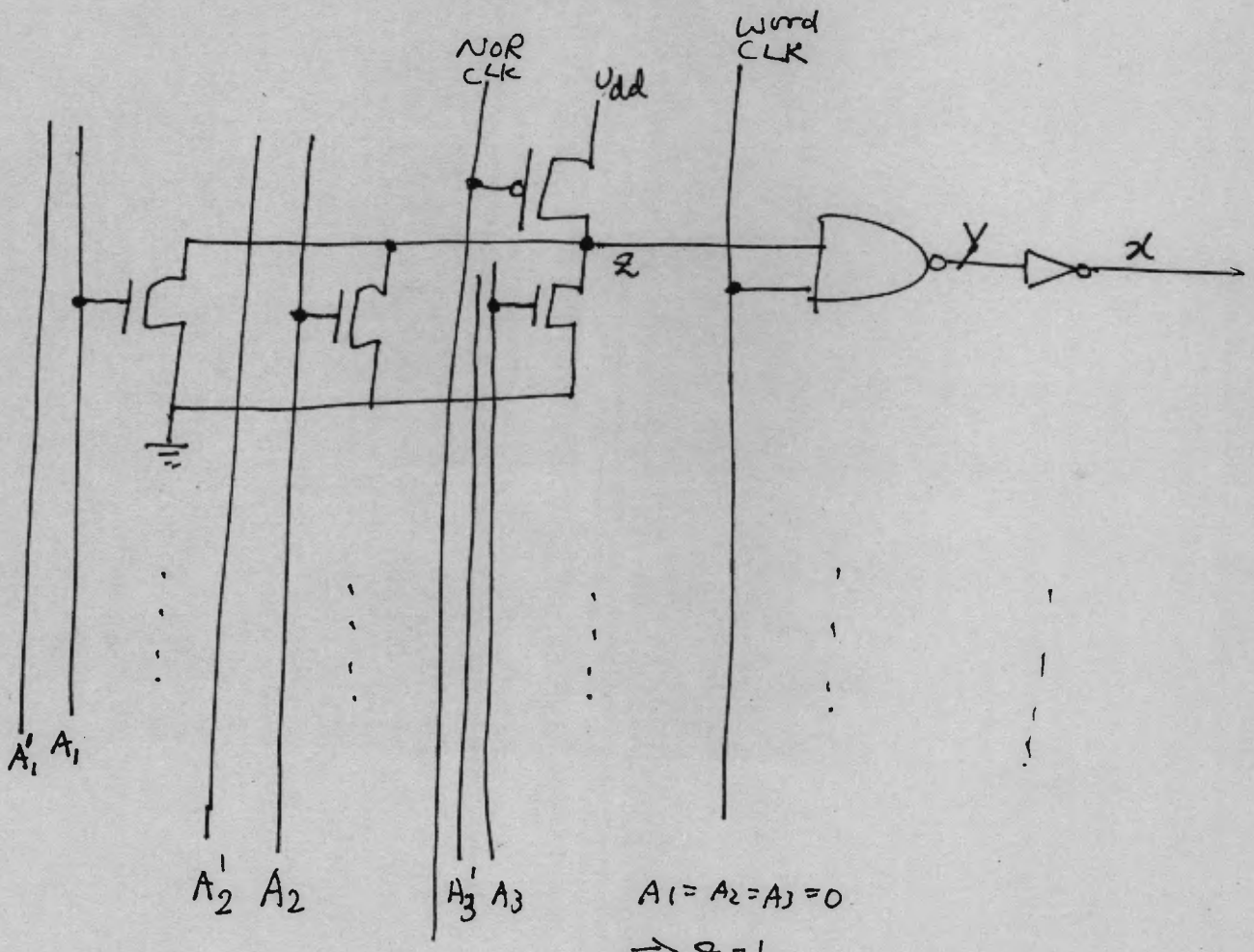


- Dynamic ckt
- $\phi = 0$, all outputs are zero.

  $\phi = 1$, only one word line would be 1
- very power-saving, slower (∵ larger height of evaluation NMOS stack)
  (only one line discharge)
- must add an inverter buffer, since there is no power source to drive the high capacitive load.

- NOR-type

3 ⟶ 8 mapping



$A_1 = A_2 = A_3 = 0$

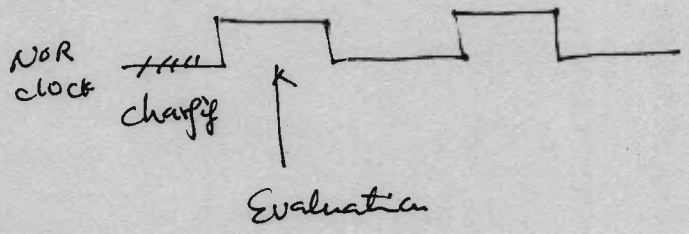
$\Rightarrow z = 1,$

$\Rightarrow y = 0$

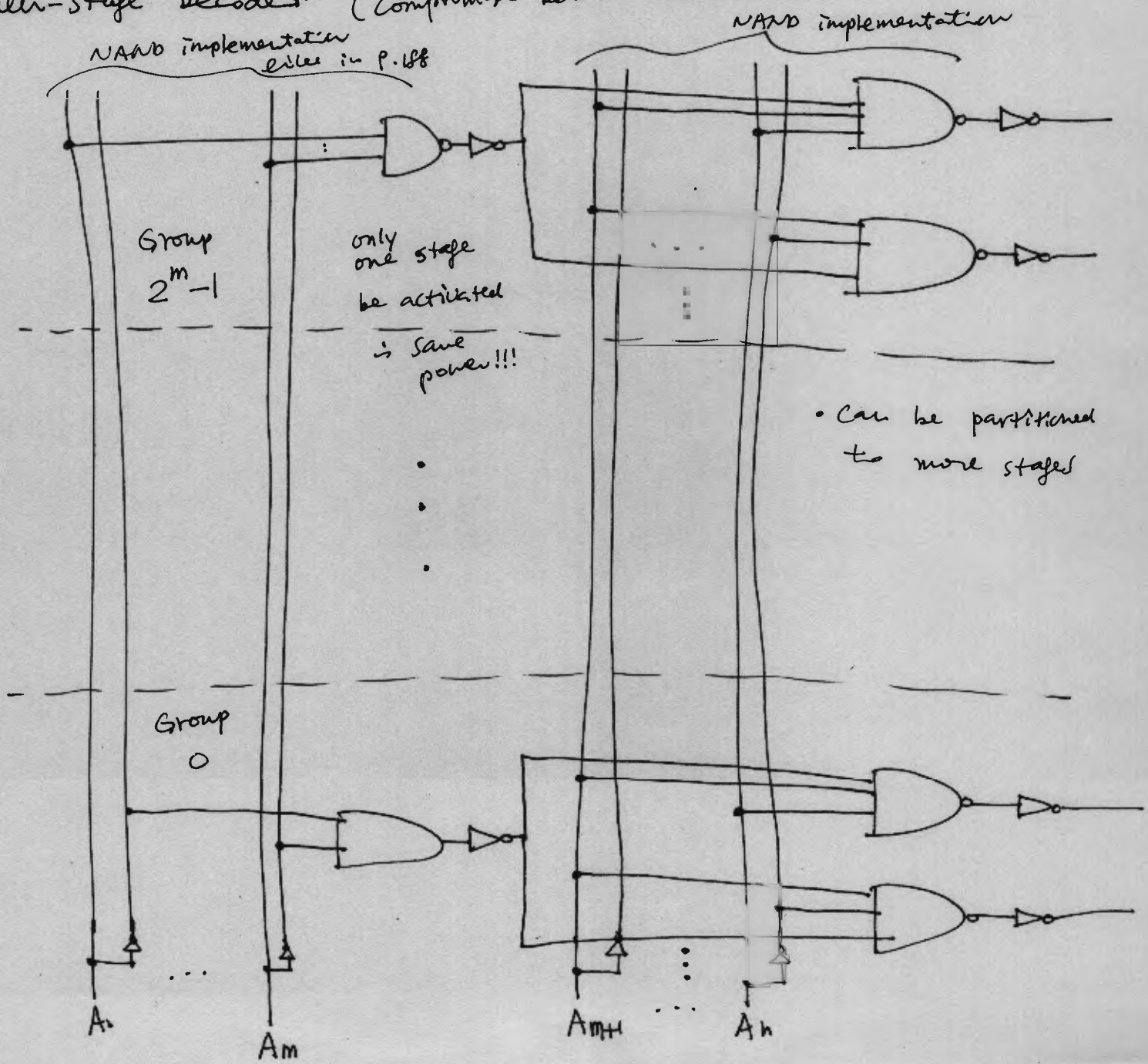
$x = 1$

- faster, but consumes more power

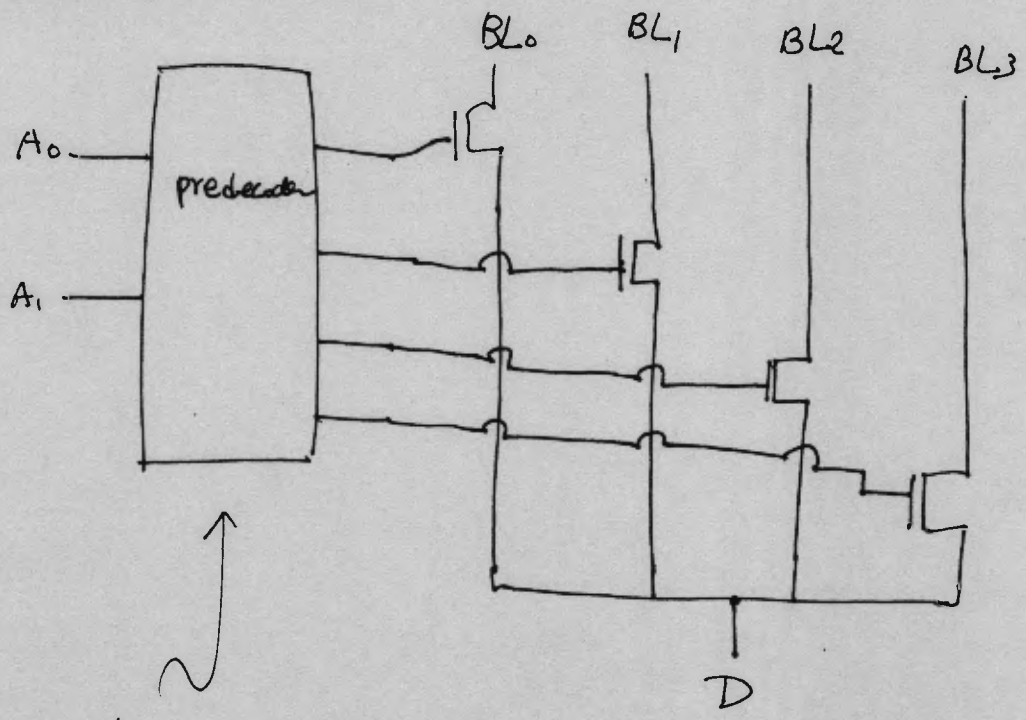
  ∵ N-1 rows are discharging.



Multi-stage decoder (Compromise between NAND-type & NOR type)



~~Row~~ Column Decoder : show by 2→4 mapping

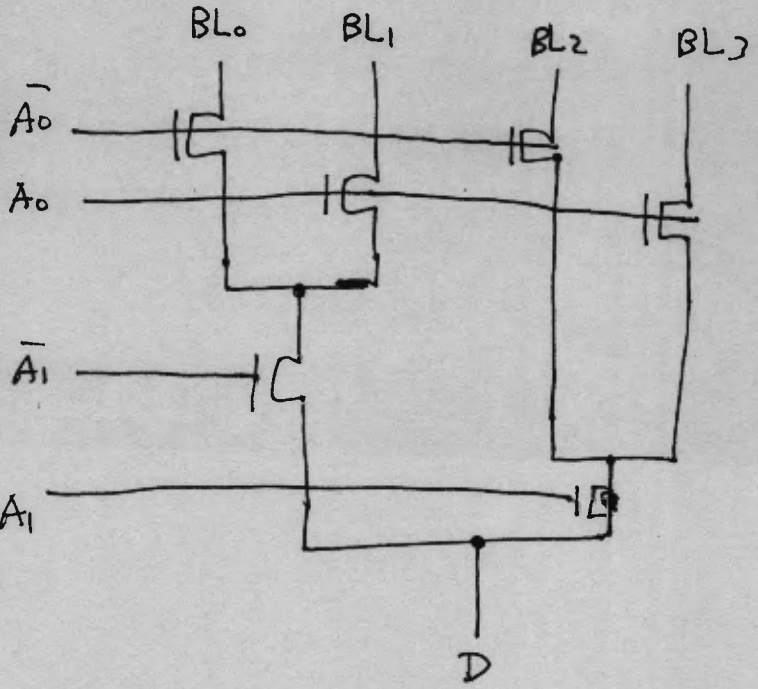


predecoder + pass transistor

Don't need to be fast!

~~this is near the end of read operation~~

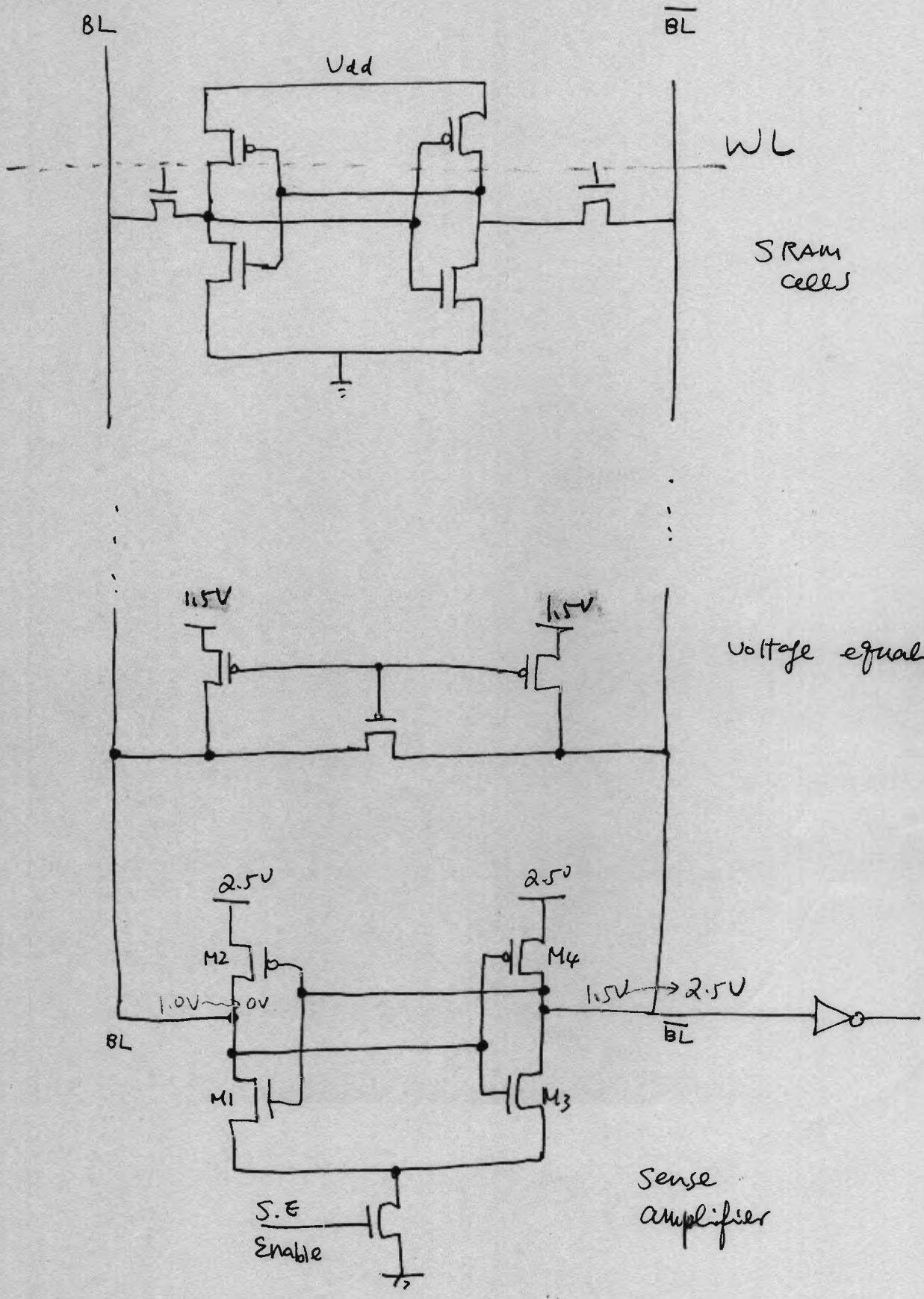
Tree decoder



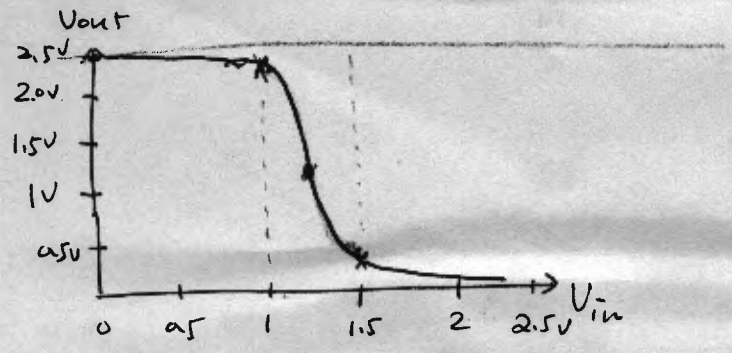
slow!

may need to pass many transistors

· Sense Amplifier



- Inverter is a high-gain device



- Charge in input 0.5V ⟹ output charge about 2.0V
- ∴ 4 times of gain.

- Initially, both BL & $\overline{BL}$ are precharged to 1.5V.

- If BL is discharged to 1.0V, $\overline{BL}$ remains at 1.5V. → increase → 2.5V

  $\overline{BL}$ remains at 1.5V → $\overline{BL}$ dropped to 0.5V → eventually → 0V.

  If $\overline{BL}$ is discharged to 1.0V. → BL remains at 1.5V → eventually → 2.5V

  BL remains at 1.5V → $\overline{BL}$ dropped to 0.5V → eventually → 0V.

- This is most commonly used in real designs and consume much less power than current mirror based S.A.

- The sense amplifier enable signal is generated using a self-timed approach to reduce the power consumption.

  That is BL will not charged to 2.5V
  $\overline{BL}$ will not be discharged to 0V } or vice versa

- I is constant, $I_1 \downarrow$ $\therefore I_2 \uparrow$

$$I_2 = \frac{V_{dd}}{r_4 + r_2}$$

$r_4 \uparrow$ $\therefore$ r_2 must be much smaller than original

$$\therefore V_2 = \left(\frac{V_{dd}}{r_4 + r_2}\right) r_2$$

(r_4: ↑ larger; r_2: ↑ much smaller; r_2: much smaller)

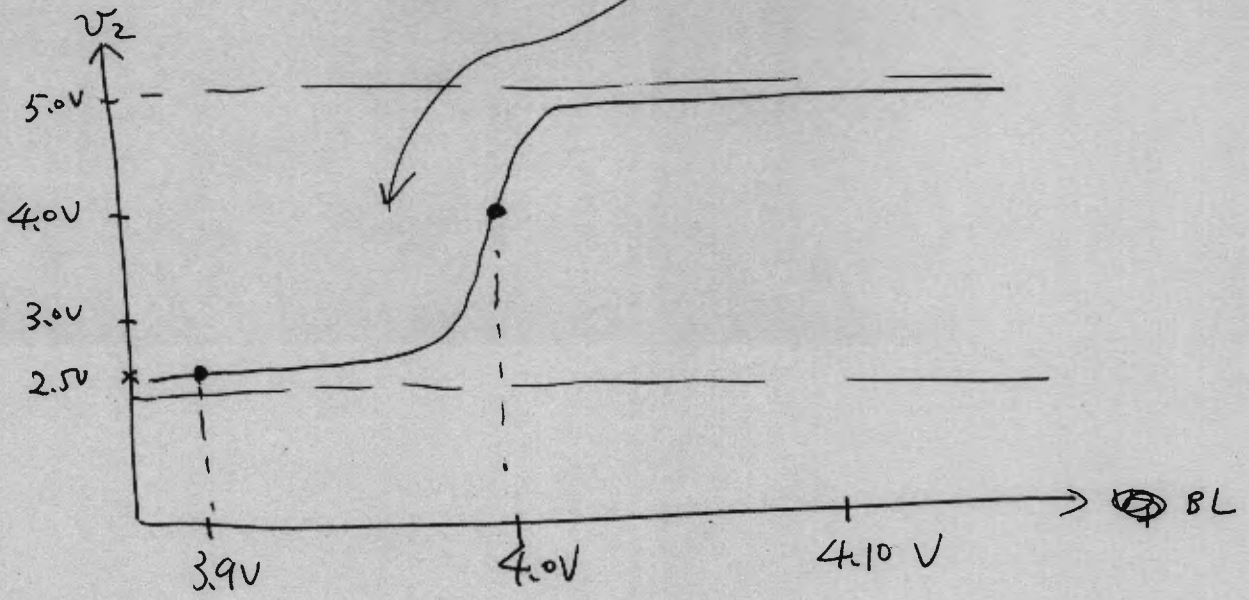
$\therefore V_2$ drop to a small value.

- Spice simulation

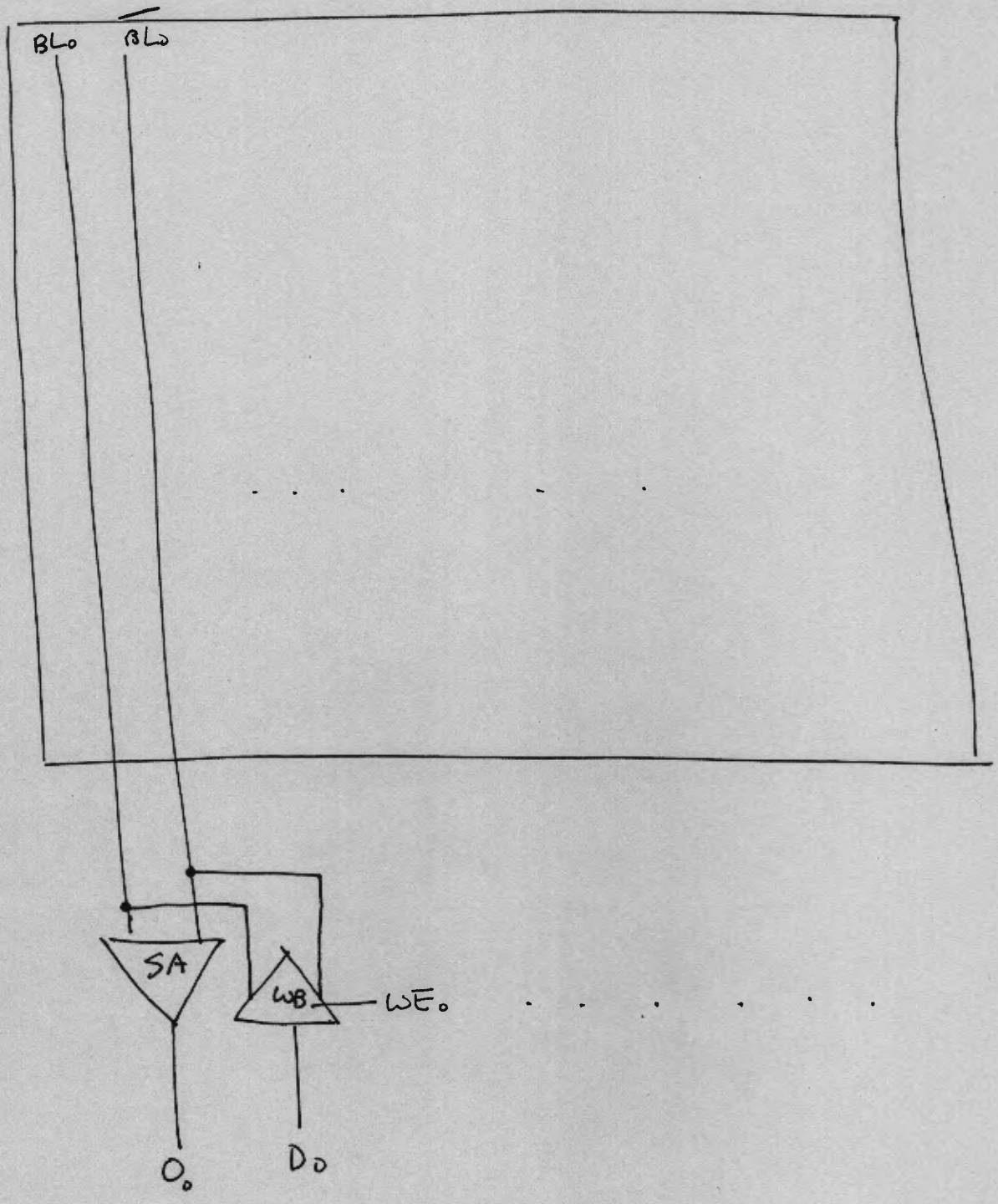
BL fixed to 4V

charge BL

output V_2 very sensitive to BL charge!!



- Read / write operation

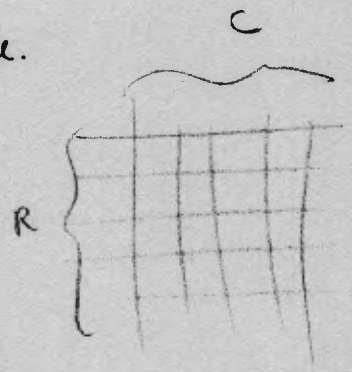


- Low-power memory design

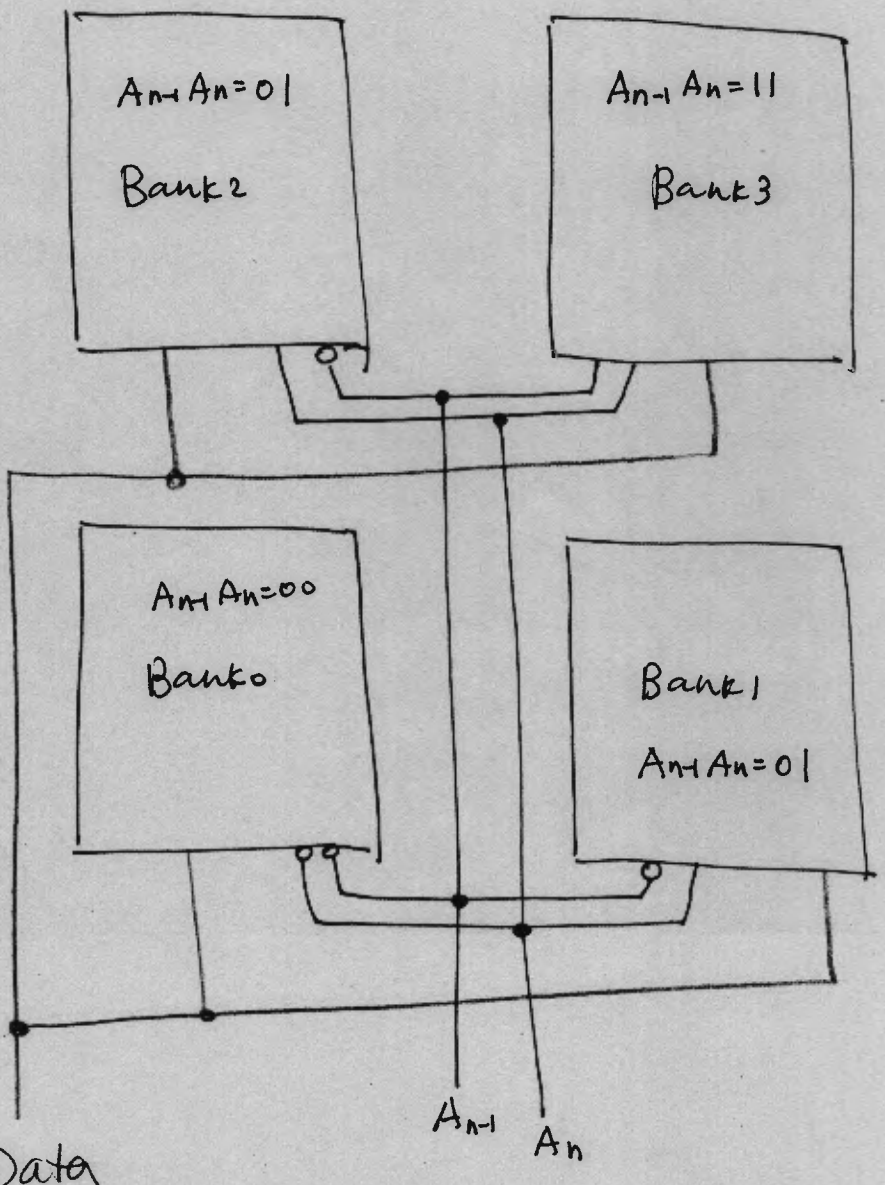
① Memory sub-banking

- R rows, C columns, C_{cell}: C (bit line switch/cell)

is totally. $R \cdot C \cdot C_{cell}$ of cap. is switched.



- Try to partition the memory

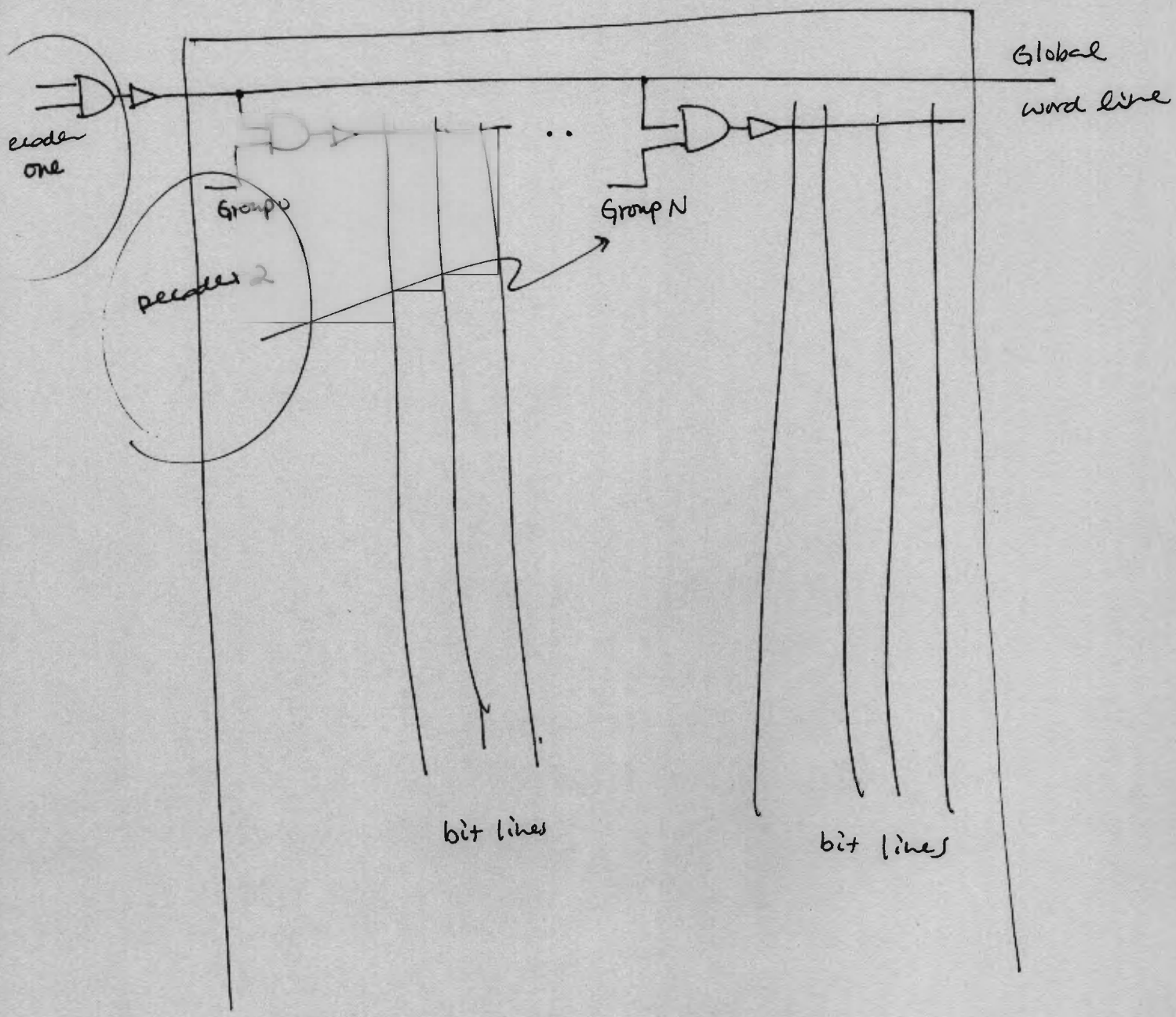


Make it B banks

$$\Rightarrow \frac{R \cdot C \cdot C_{cell}}{B}$$

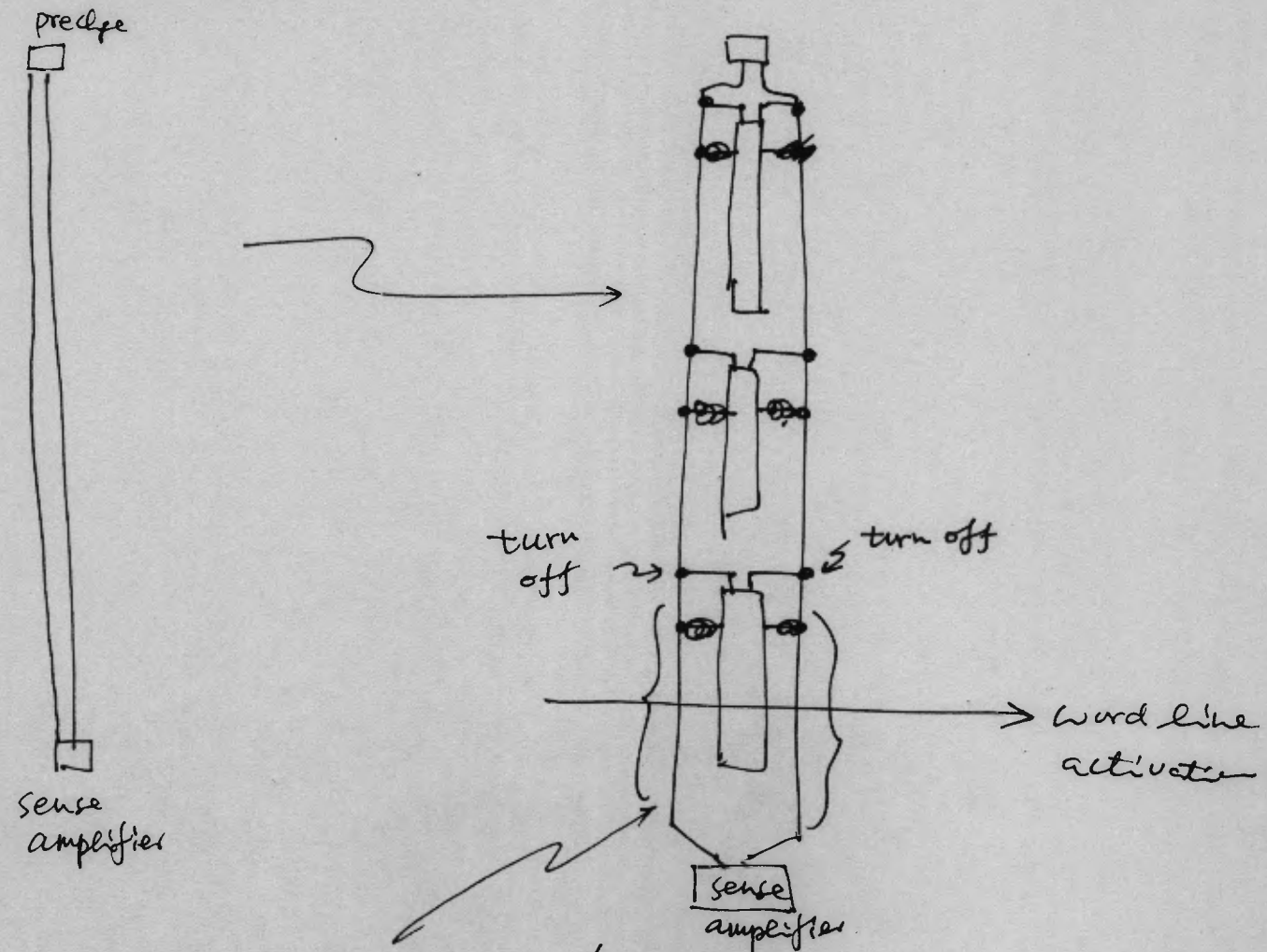
- if interconnect is not considered.
- Can find an optimal solution.

Divided Word Line Architecture



- Reduce power consumption in Bit Lines.

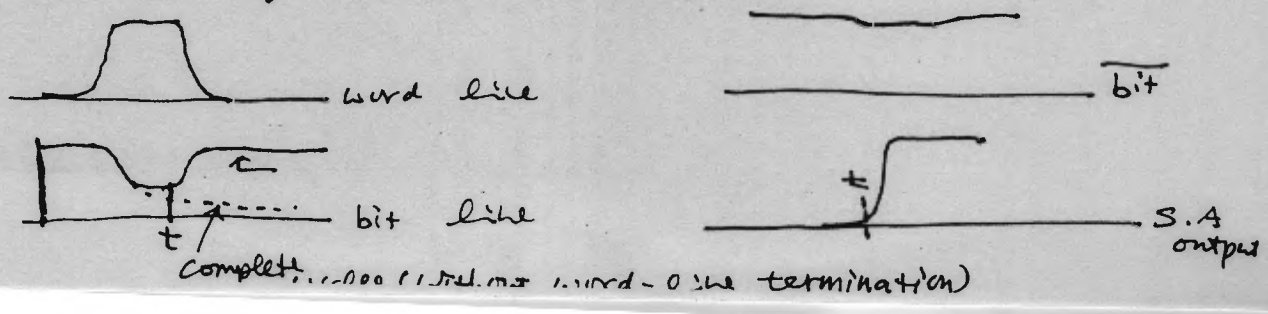
- Bit line segmentation



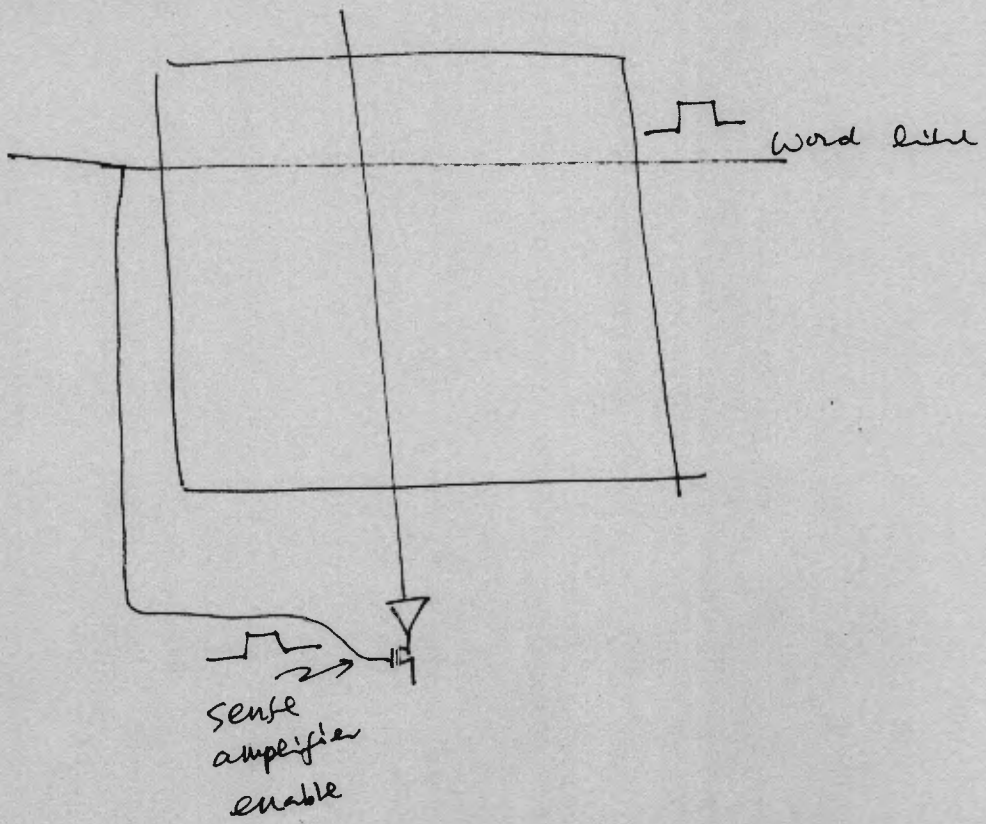
- Reduce voltage swing on Bit Lines

  * Not quite feasible

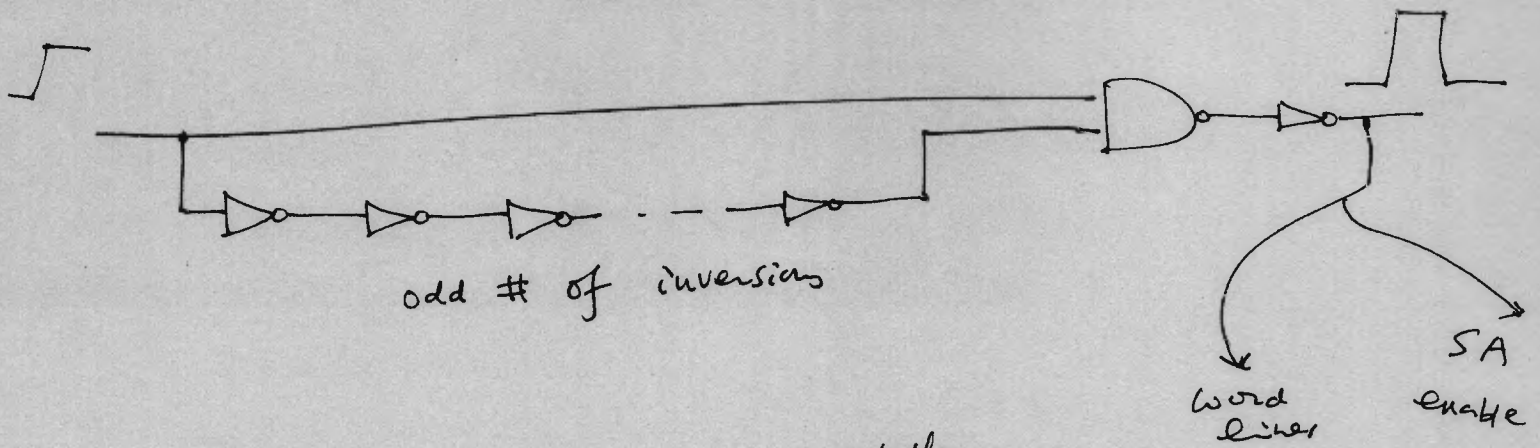
    ∵ affect noise sensitivity, complexity of sense amplifier, performance of the RAM core.



Pulsed word Lines

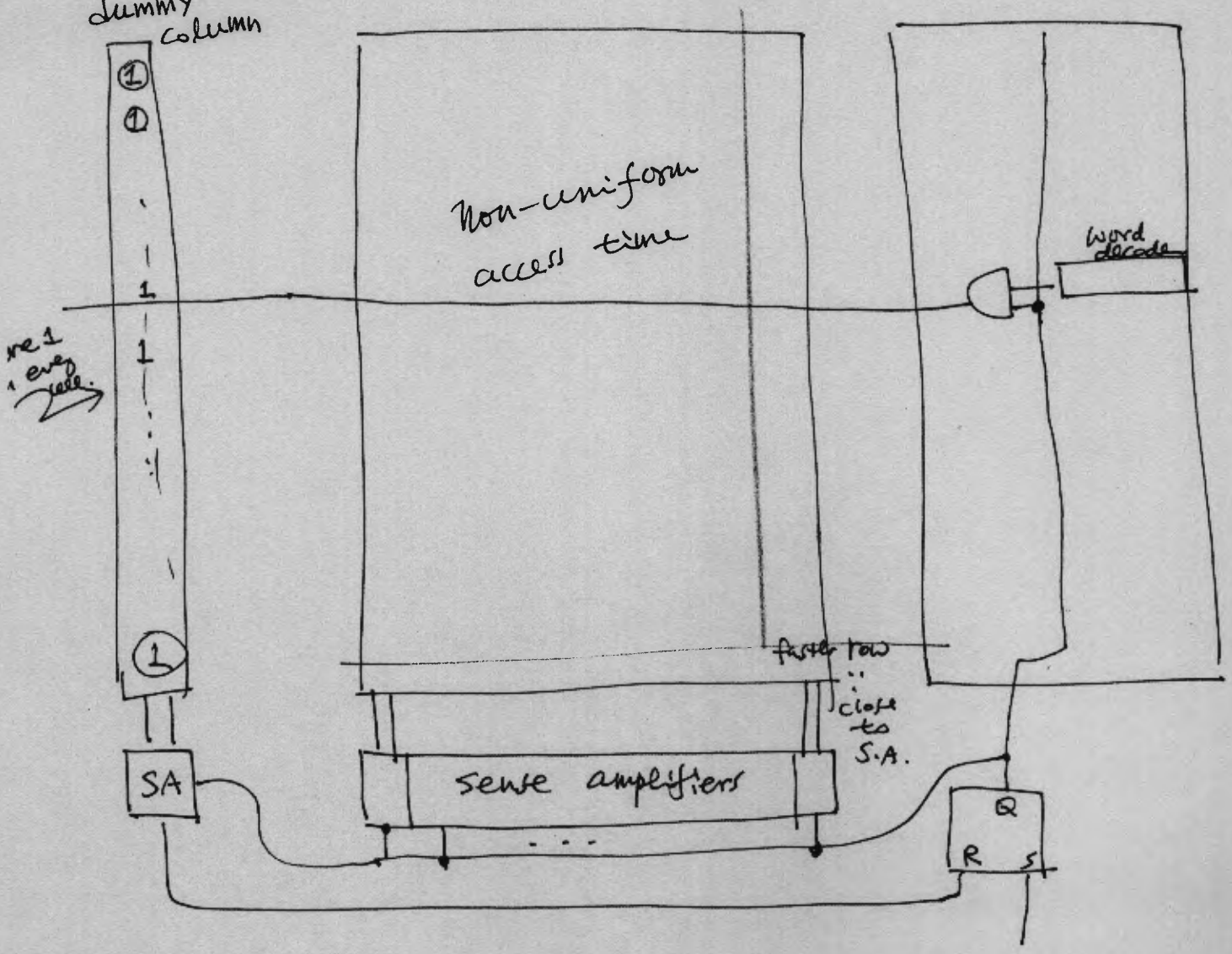


- Try to limit the bit line voltage discharge.
- Try to enable the word liner for precisely the time needed to develop the bit cell voltage discharge.



- It is hard to control the width
- Try to use self-timed word lines.

Self-timed word lines.



① SR flip-flop is set, and the word line is triggered. (SA)

② When the dummy SA (slowest) generate 1, the rest of the columns would have been sensed.

③ The high signal reset the flip-flop and turn off the word line. (SA)

Summary: power optimization for SRAM

Reduce total capacitance switched

① Banked organization

② Word-line division

③ Multi-stage decoder

④ Bit line segmentation

Reduce the voltage swing across switched capacitances

① Self-timing RAM

turn off word line & SA enable, s.t. the voltage swing in the bit lines can be minimized.

② Reduce bit line voltage swing.

- pn junction reverse-biased leakage current

- subthreshold leakage

$$I_{sub} \approx \frac{W_{eff}}{L_{eff}} I_{D0} e^{q(V_{GS} - V_{TN})/nKT}$$

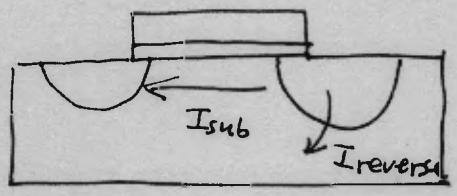
where I_{D0}: constant

q: 1.6×10^{-19} Columb

K: Boltzmann constant

T: absolute temperature

n: Subtreshold constant



- Tunneling leakage

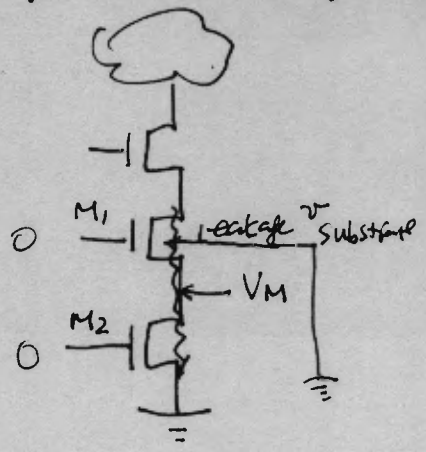
- Gate induced drain leakage

⋮ ⋮

- Mainly deal with subthreshold leakage in this discussion.

- circuit techniques for leakage reduction

➢ ① Standby leakage control using transistor stacks.



Stacking effect: I_{sub} flowing through a ~~series~~ Stack of series-connected transistors reduced when more than one transistor in the stack is turned off.

- when M_1 and M_2 are off, V_M is positive due to small drain current.

Reasons for stack effect:

ⓐ V_{gs1} becomes negative, thus, I_{sub} reduces substantially.

ⓑ $$V_T = V_{T0} + \gamma\left(\sqrt{2\phi_F + |V_{BS}|} - \sqrt{2\phi_F}\right)$$

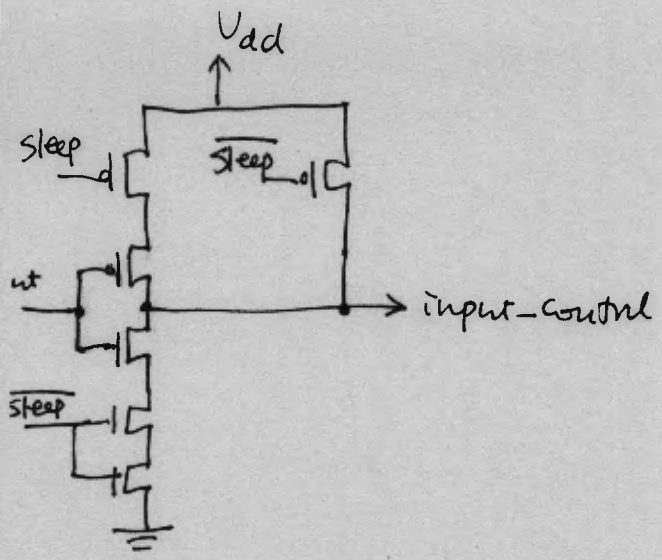
γ: body effect coefficient

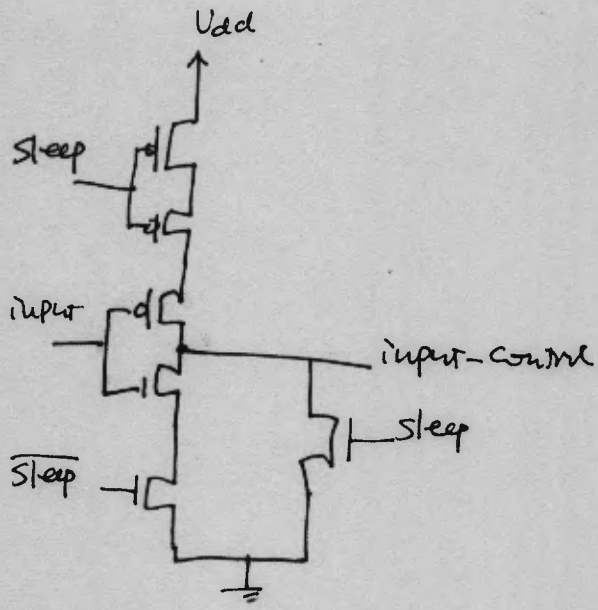
$|V_{BS}| = V_{substrate} - V_{source}$

$\therefore V_{BS1}$ is a negative value, and $V_T \uparrow$

- For significant body effect, V_T may increase from V_{T0} by 1.5 ~ 2 times
- If $V_T \uparrow \Rightarrow I_{sub} \downarrow$
- the leakage of a two-transistor stack can be an order of magnitude less than the leakage in a single transistor

- Due to the stack effect, I_{sub} of a logic ckt dependents on its input vector.
- How to find the best input vector is a very hot research topic
- Circuit support is very simple!!





Control to 1 logic

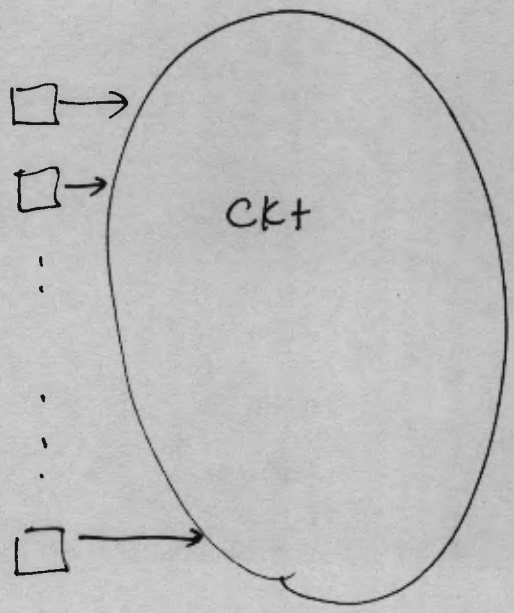
Sleep = 1, → input-control = 1

Sleep = 0 → input-control = $\overline{\text{input}}$

Control to 0 logic

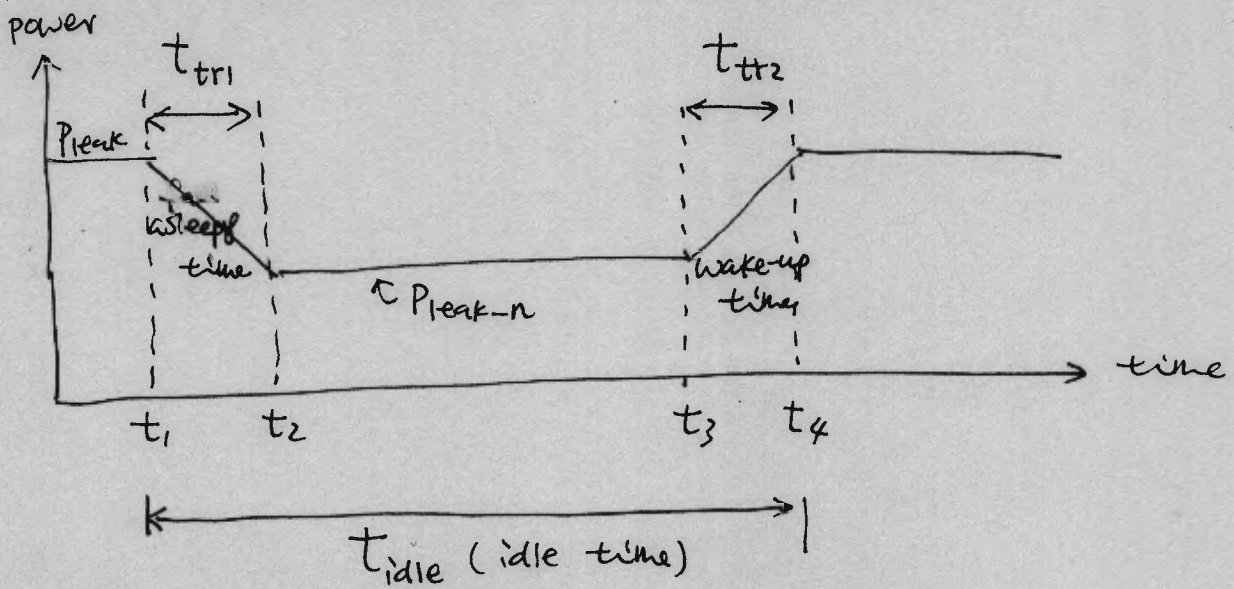
sleep = 1, → input-control = 1

sleep = 0 → input-control = $\overline{\text{input}}$



Depending on the input control vector, Control-to-1 logic and Control-to-0 logic are used.

- The minimum time required to save energy.



$$E_{w/o\text{-}scheme} = P_{leak} \cdot t_{idle}$$

$$E_{w\text{-}scheme} = P_{leak\text{-}n} \cdot (t_{idle} - t_{tr1} - t_{tr2}) + E_{tr1} + E_{tr2}$$

$$\approx P_{leak\text{-}n} (t_{idle} - 2t_{tr}) + E_{tr}$$

$$E_{w\text{-}scheme} < E_{w/o\text{-}scheme}$$

$$t_{idle} > \frac{E_{tr1} + E_{tr2} - P_{leak\text{-}n}(t_{tr1} + t_{tr2})}{(P_{leak} - P_{leak\text{-}n})}$$

$$\approx \frac{E_{tr} - P_{leak\text{-}n}(2t_{tr})}{(P_{leak} - P_{leak\text{-}n})}$$

- Generally, t_{idle} is very small (in the level of ns)
  ∵ small transition energy penalty
- t_{idle} decreases with technology scaling.

② Multiple V_{th} Design

Use high V_{th} transistors to suppress I_{sub}.

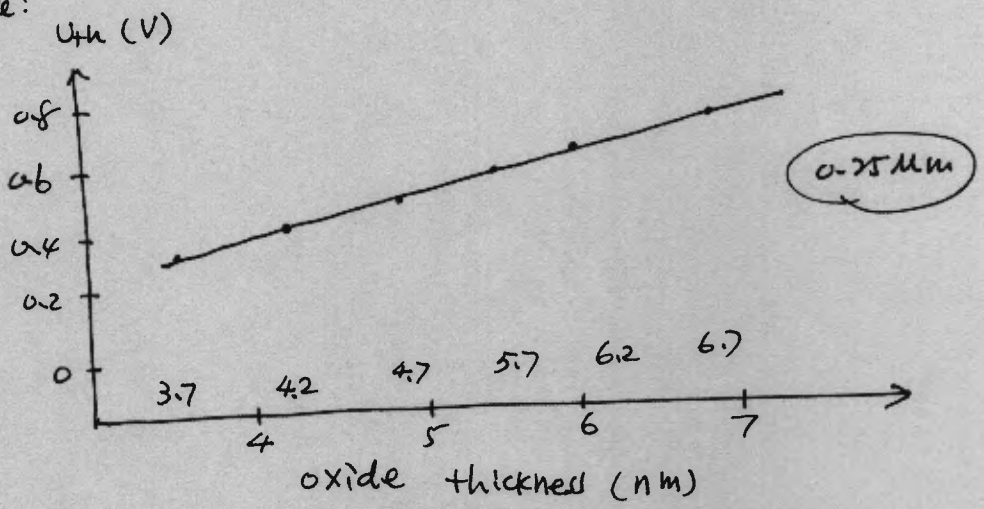
Use low V_{th} transistors to achieve high performance.

How to implement V_{th} transistors ?

- multiple channel doping
- multiple oxide CMOS
- multiple channel length
- multiple body bias.

• V_{th} is fixed for each gate!!

Example:



Higher oxide thickness not only reduces I_{sub}, but also reduces:

(a) gate oxide tunneling

(b) gate cap ⇒ thus dynamic power consumption.

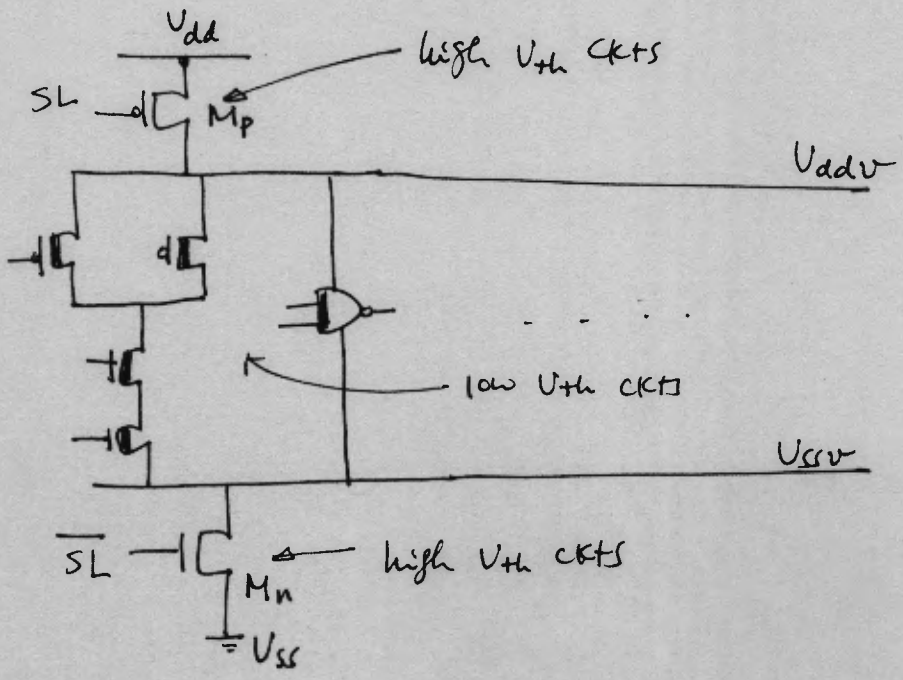
Side effect:

t_{ox} ↑ ⇒ short-channel effect ↑

⇒ channel length must be increased.

Circuit Techniques

(a) Multiple-threshold voltage CMOS. (MTCMOS)



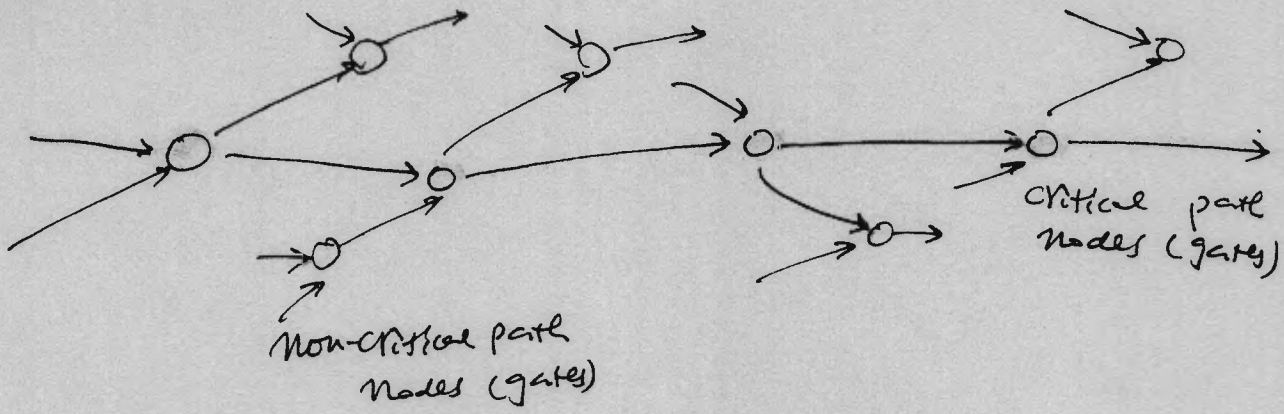
- active mode: $SL=0$, M_p & M_n are on, • standby mode: $SL=1$

  ∵ their on-transistances are small

  ∴ the virtual supply voltages (V_{ddv}, V_{ssv}) functions as real power lines

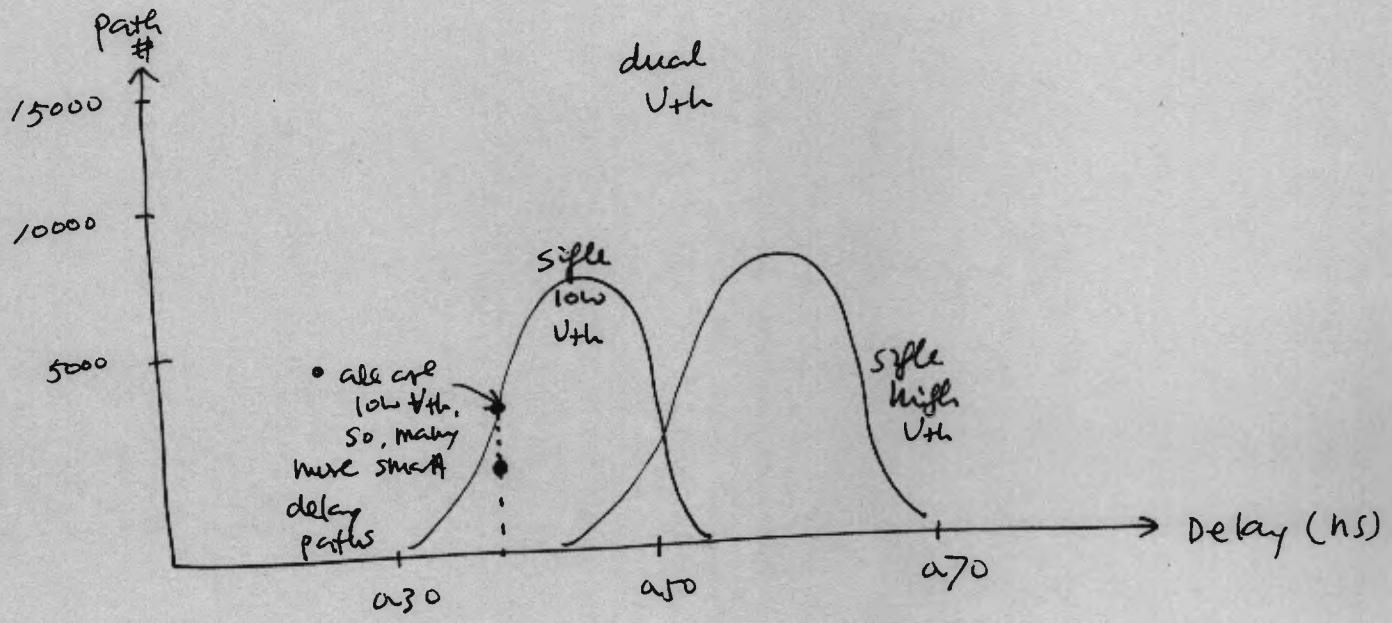
- In fact, only one type of high V_{th} transistor is enough for leakage control
- NMOS insertion scheme is preferable, since NMOS on-resistance is smaller.
- Increase area & delay

(b) Dual-threshold CMOS



Assign low Vth ~~gates~~ to critical path nodes

— high ——— non-critical ———.

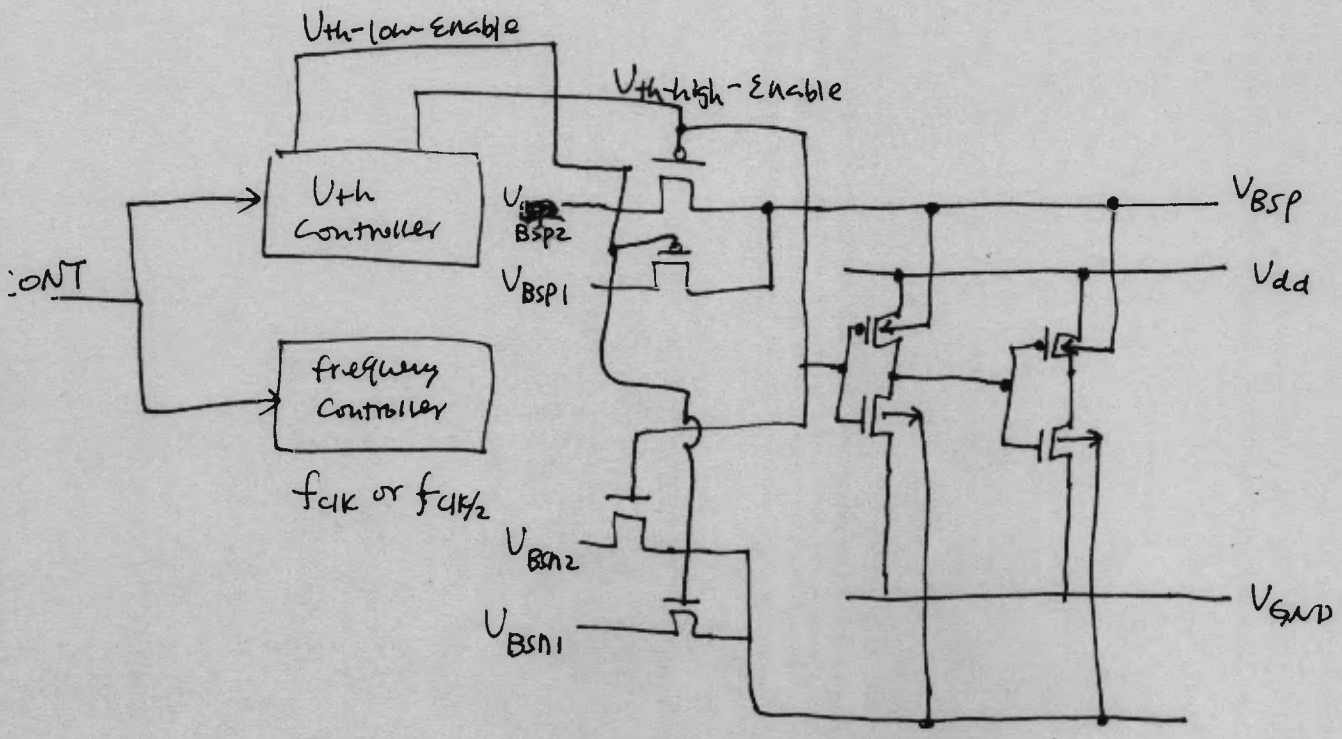


· Dual threshold technique is good for leakage power reduction during both standby & active modes without delay & area overhead.

(There is no mode control signal e.g., SL)

) Dynamic V_{th} Design (e.g., Body bias control (BBC))

key: When workload decreases, less power is consumed by increasing V_{th}.



CONT: obtained from software

CONT: controls both Substrat bias voltage + operating frequency

performance overhead:

$$t_{delay} = \frac{\Delta V_{sub} * C_{sub}}{W_{driving\text{-}device} * I_{on}}$$

ΔV_{sub}: voltage difference (of substrate) to be charged.

C_{sub}: Substrate capacitance

$W_{driving\text{-}device}$: Width of the driving devices

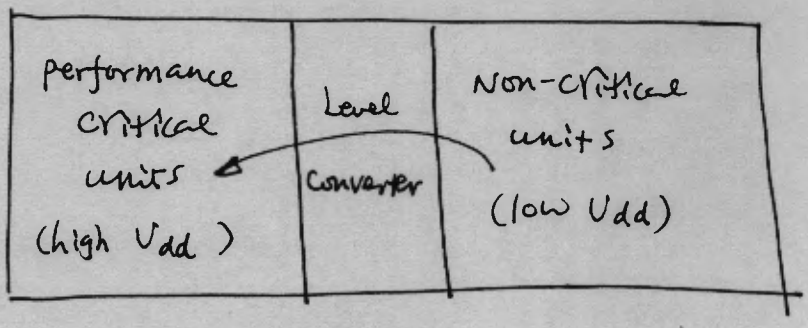
I_{on}: transistor saturation current.

	a1s	a18	a07
$t_{delay} \approx$	30ns	21ns	7.2ns

- Minimum idle time: in the level of ms.

:) Supply voltage scaling

- Originally, developed for switching power reduction.
- For 1.2V, 0.13μm technology, supply voltage scaling has significant impacts on I_{sub} & I_{gate} (gate leakage) in the order of V^3 and V^4 respectively.

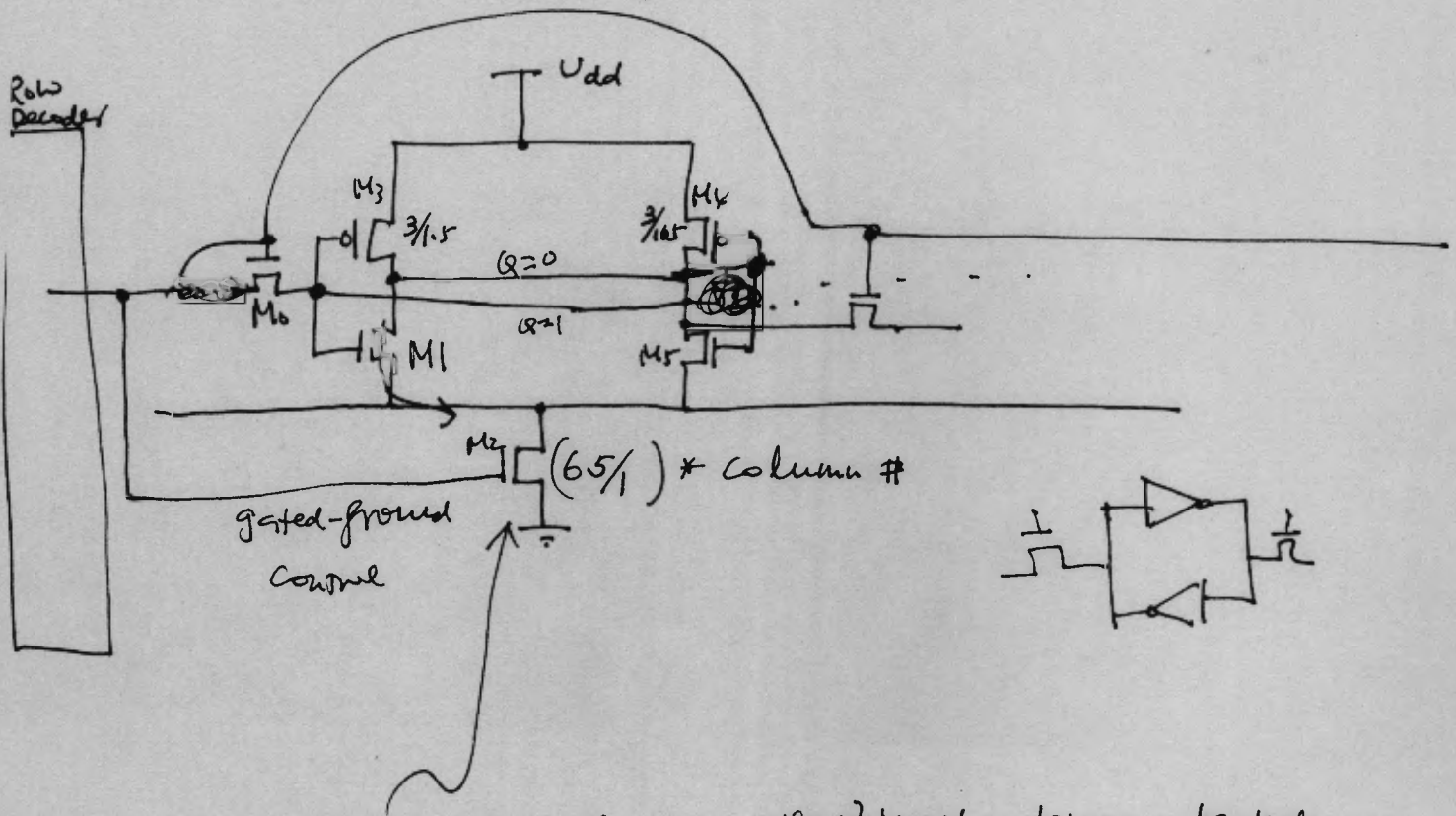


- Whenever an output from a low Vdd unit drives an input of a high Vdd unit, a level conversion is needed at the interface.

Leakage Reduction for Cache memory.

- 30% of Alpha and 6% of Strong Arm processors are devoted to cache
- For 0.13μ technology, leakage energy accounts for 30% of L1 cache and 50% of L2 cache energy.

(a) Data retention gated-ground cache.



- Gated ground achieves significantly lower leakage
  ∵ two off transistors connected in series, reduce the leakage current by orders of magnitude.
- The gated-ground transistor plays a major role in data retention capability and stability of the cell.
  ∴ must be carefully sized.

- The ground-gated transistor must be large enough to ① sink current during Read/write operation, and (active mode)
  ② enhance the data retention capability of the cache (standby mode)

- But, a large gated-ground transistor may reduce the stack effect, thereby diminishing the energy savings.
  ※ Too small resistance to generate enough body effect to increase V_{th}.

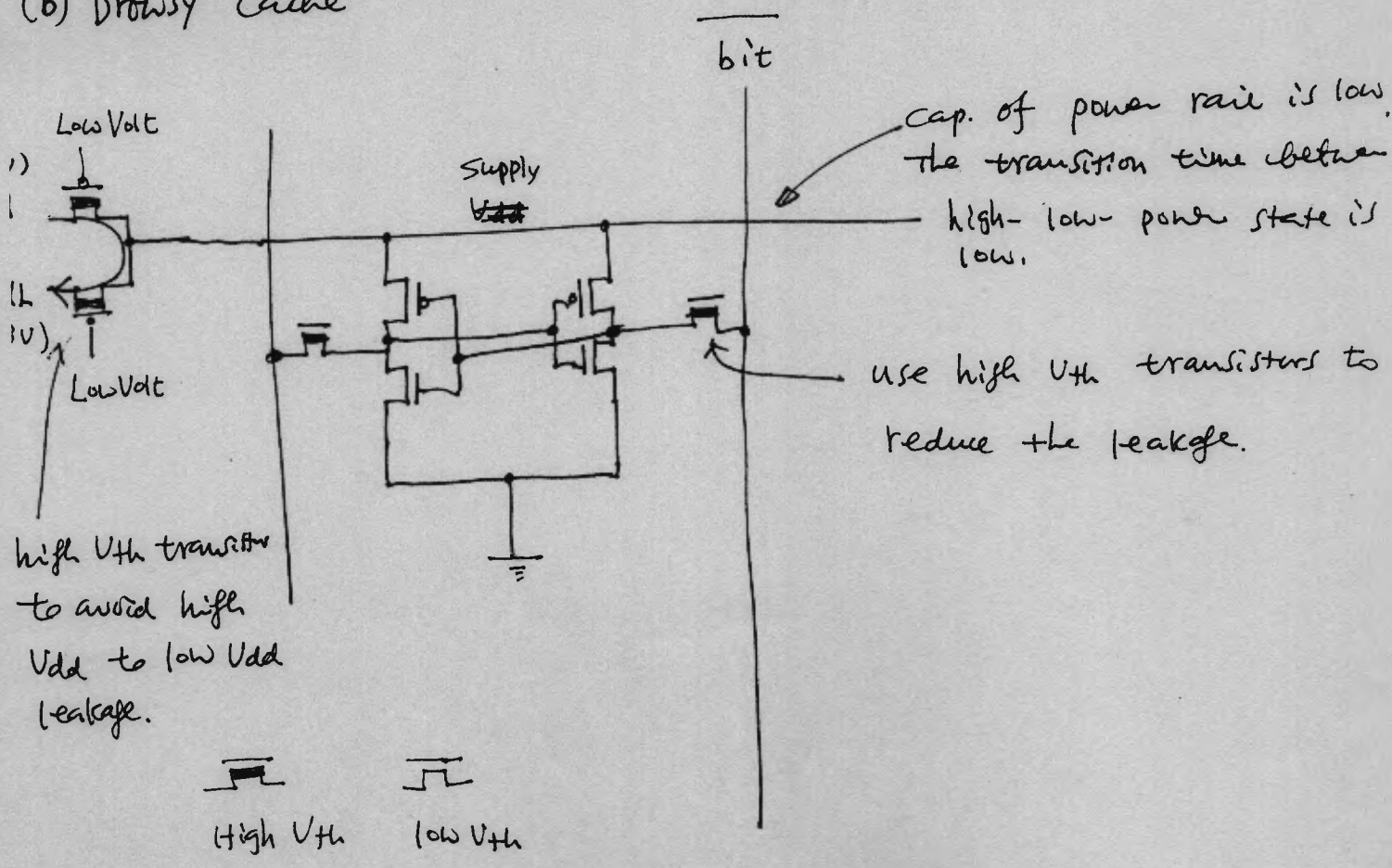
- To maintain the performance, proper sizing of the decoder is required, because the decoder drives the gated-ground transistor.

- Turning off the gated-ground transistor cuts off the leakage path to the ground, but it also cut off the opportunity to firmly strap nodes (e.g., M1), which are at zero, (接地) to the ground.

  - Easier to write logic 1 to that node by a noise.
  - Turn on the gated-ground transistor restores the zero data.
  - Simulation shows that data is not lost even if the gated ground is turned off for indifinite time.

transistor

(b) Drowsy Cache

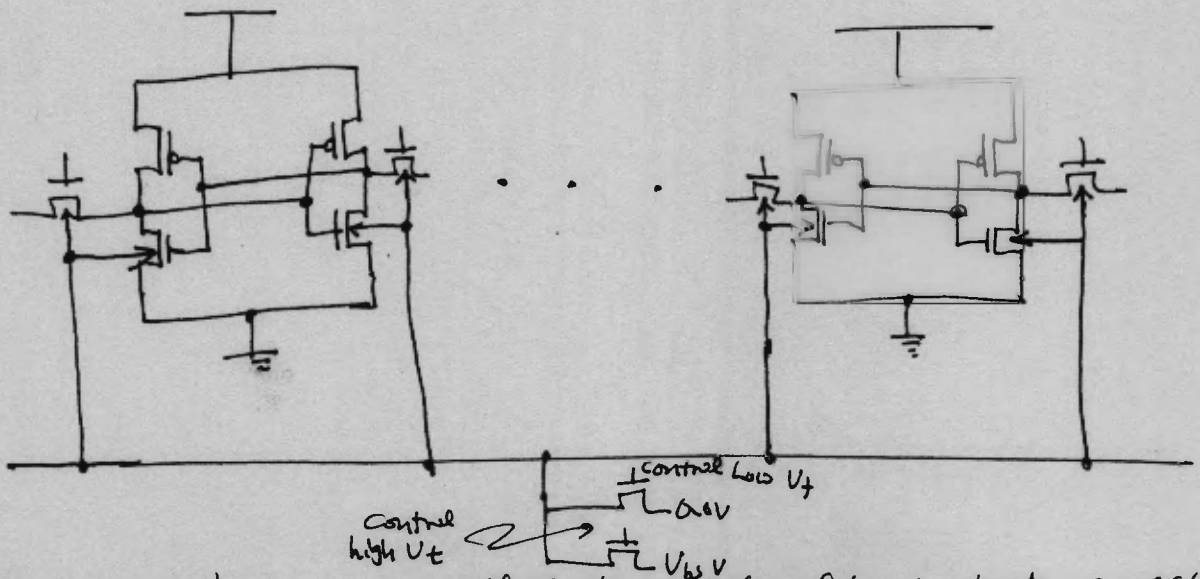


- Normal operation : supply = Vdd (1.0V)

  drowsy mode : supply = 0.3V (The state of the cell can be maintained)
  (70nm technology)

- The combined effect of $I_{Leakage}$ ↓ & Supply Voltage ↓ gives a large reduction in the leakage power.

(C) Dynamic threshold Voltage Vth SRAM.



- High V_{th} (V_{bs} V) is assigned to cache lines not accessed for a certain period (30 ms ⟶ 100 ms)
- Low V_{th} (0V) is assigned to cache lines in frequent use to maintain high performance
- Instead of turning a cache line to high Vth right after its access, leaves the cache line into low Vth for a certain period (30–100 ms).

(temporal & spatial locality) of program reference

Comparison between Input vector control & Body bias control

	Leakge Reduction % 0.18u, 0.07u		performance penalty 0.18u, 0.07u	Area overhead % 0.18u, 0.07u	Minimum Idle time 0.18 0.07	Dynamic power overhead 0.18 0.07	Targted ckt
VC	47.6	47.2	1 Clk cycle	3.84	4.6MS 5.7n	very low	D
3C	70.7	53.3	21ns, 7.2ns	1.44, 2%	18.4u 4.03u	Low	D&M.
ted Vdd ith ata eserv xge (calc 10)	80.3	94.1	50ns 20ns	1 1.5	172.3 ns 3.0 ns	very low	D&M

(∵ The sleep transistor provides Stacking effect to all transistors of the same type in the ckt)

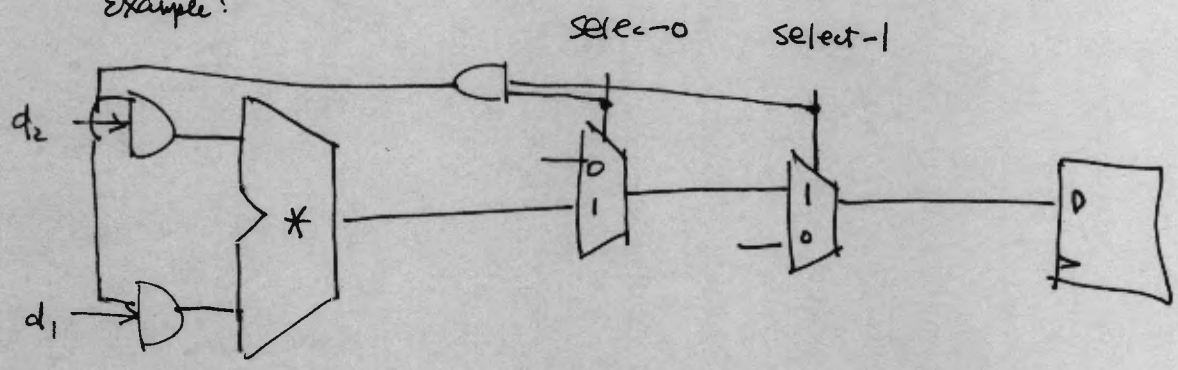
Technology scaling

	Leakge Reduction Effectiveness	Relative overhead	Minimum idle time
IVC	increase	fixed	Decrease (X 0.001)
BBC	Decrease	increase	Decrease (X 0.5f)
Gated Vdd w/ data preserving	increase	increase	Decrease (X 0.05)

Synopsys Low-power design flow.

- Clock gating
  - Module-level clock gating
  - Register-level clock gating
  - Cell-level clock gating
- Automatic clock gating at the register level
  - including testability concerns
- Operand isolation

Example:



- Logic minimization
  - Sizing & buffering
  - Technology mapping
  - phase assignment
  - Algebraic transformation
- Leakage control
  - Multiple-Threshold design
  - Variable threshold biasing

- Voltage scaling

  E.g., create separate voltage islands

- Modeling & Estimation

  Switching power

  Internal power

  leakage power

  Modeling

  Scalable polynomial power models (SPPMs)

  Modeling activity