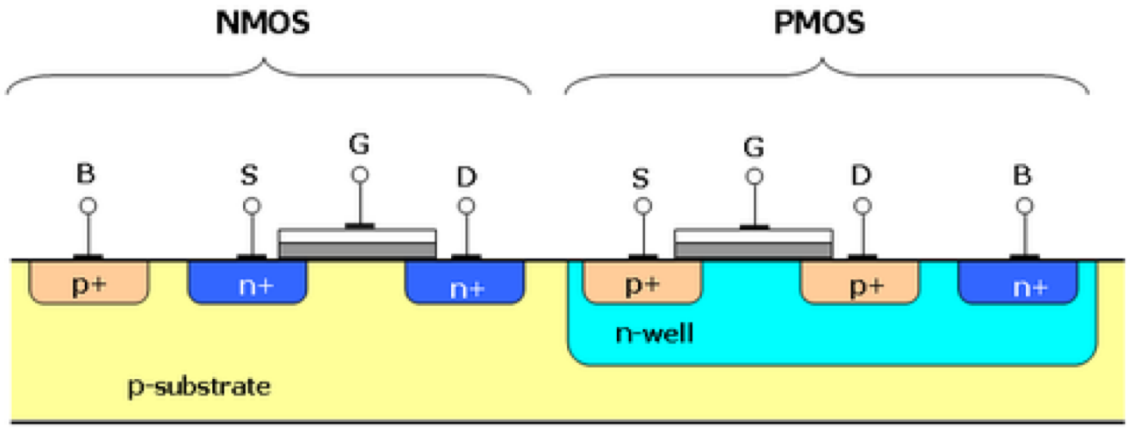
Computer architecture

Logic gates and assembly language

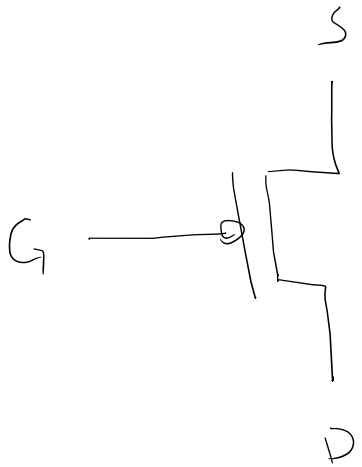
Goal

- Introduce CMOS technology and binary algebra
- Use binary algebra technique to construct combinational circuits
- Introduce sequential circuits
- MIPS assembly language
- Turing machine to CPU architecture (single cycle)

CMOS (2D)



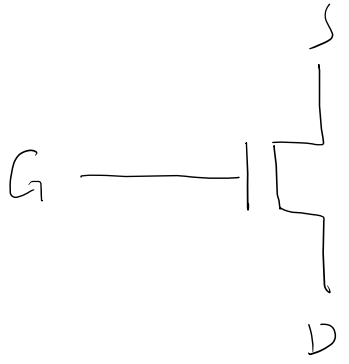
FinFET → 3D structure



PMOS

If $G = 1$, $R_{SD} \to \infty$

If $G = 0$, $R_{SD} = 0$

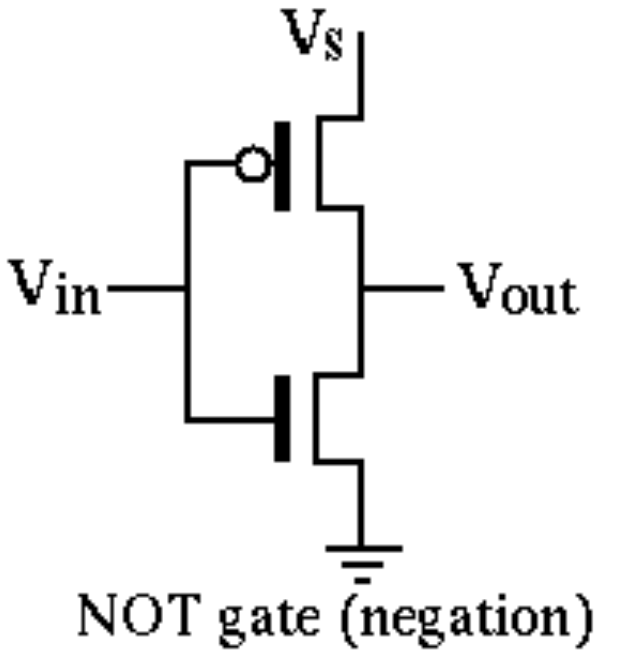


NMOS

If $G = 1$, $R_{SD} = 0$

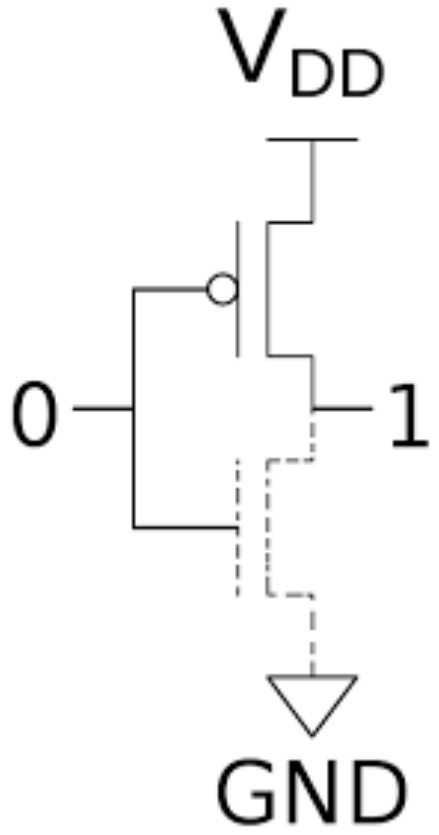
If $G = 0$, $R_{SD} \to \infty$

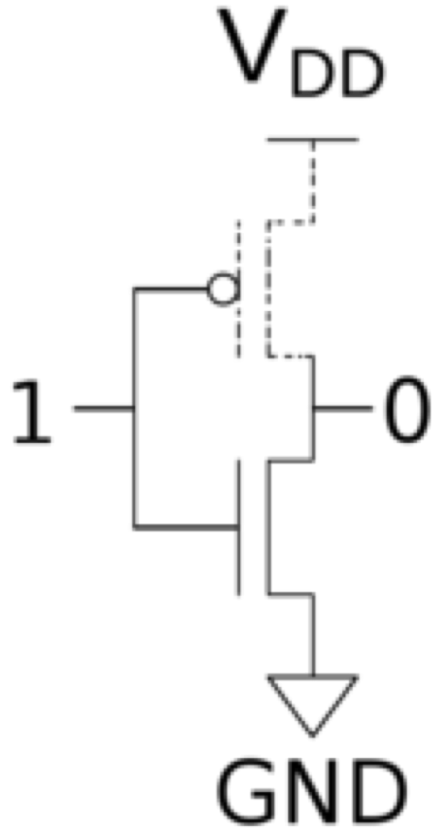
Not gate



NOT gate (negation)

Vin —▷○— Vout



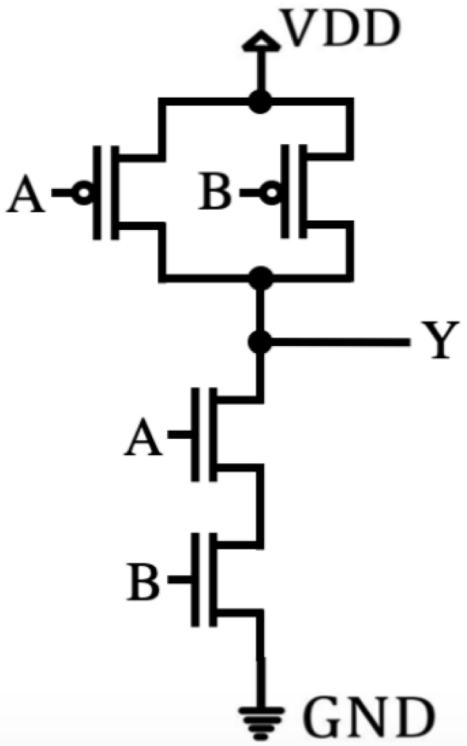


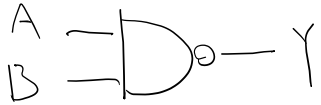
Vin	Vout
0	1
1	0

No current flow in both cases

NAND gate

No current flow in all four cases



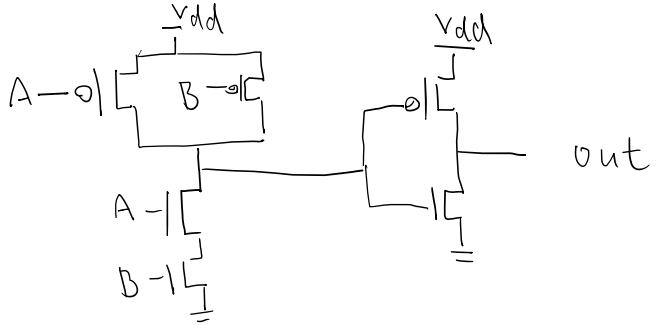


A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

$\Rightarrow Y = \overline{(A \cdot B)}$

= NOT (A AND B)

AND = NOT (NAND)



A	B	Out
0	0	0
0	1	0
1	0	0
1	1	1

Demorgan's law

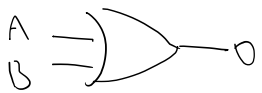
$$\sim (A \wedge B) = \sim A \vee \sim B$$

$$\sim (A \vee B) = \sim A \wedge \sim B$$

AND

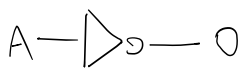
$O = AB$

OR



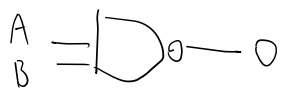
$O = A + B$

NOT



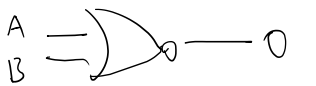
$O = \bar{A}$

NAND



$O = \overline{AB}$

NOR

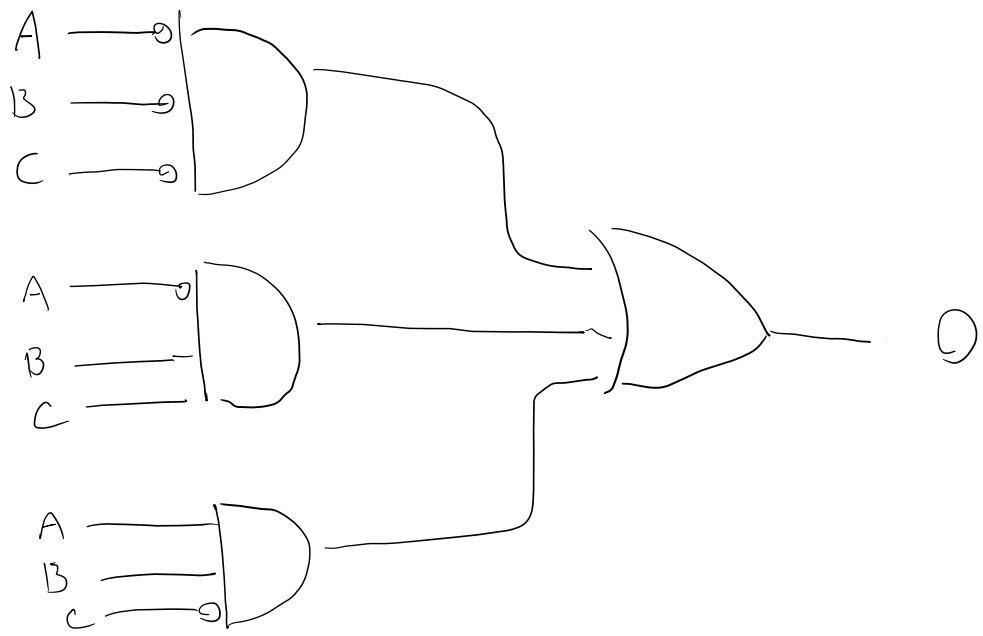


$O = \overline{A + B}$

Combinational logic

A	B	C	O
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	0

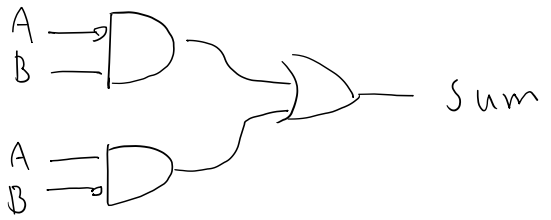
$$O = \bar{A}\,\bar{B}\,\bar{C} + \bar{A}BC + AB\bar{C}$$



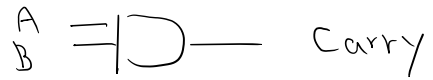
Half adder

A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

$$Sum = \bar{A}B + A\bar{B}$$



$$Carry = AB$$

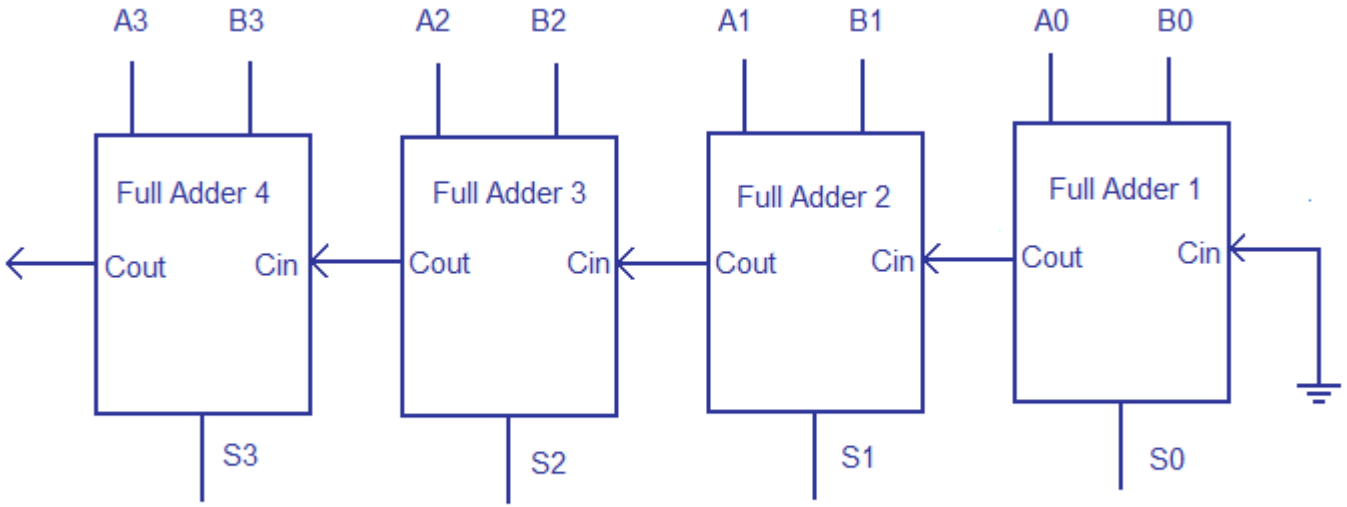


Full adder

A	B	C_{in}	S	C_{out}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

$$S = \bar{A}\bar{B}C_{in} + \bar{A}B\bar{C}_{in} + A\bar{B}\bar{C}_{in} + ABC_{in}$$

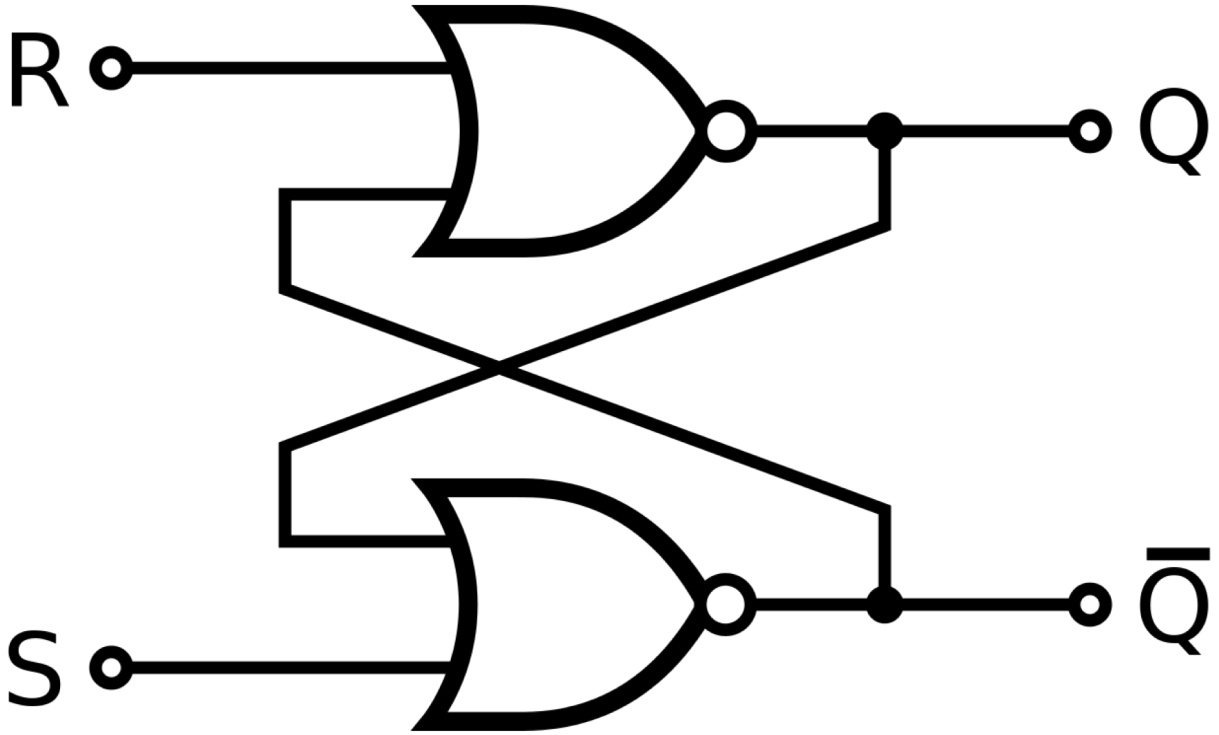
$$C_{out} = \bar{A}(BC_{in}) + A(B + C_{in})$$



4 bit ripple carry adder

www.circuitstoday.com

Feedback and RS flip flop (state storage)



NOR

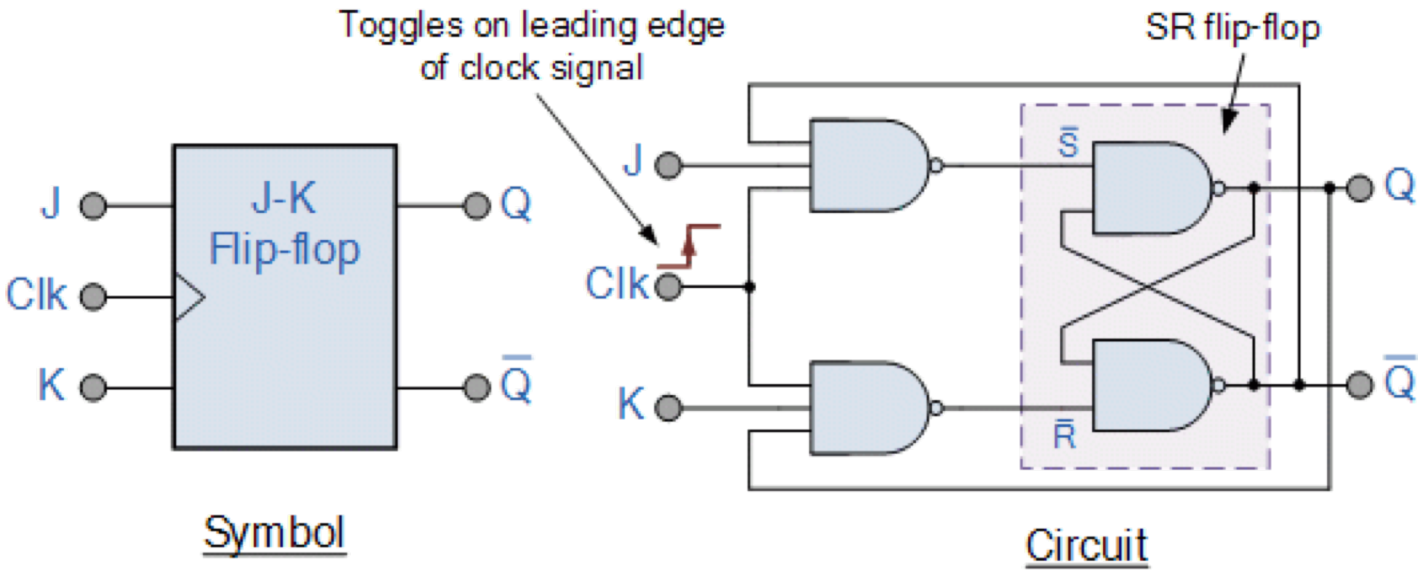
A	B	O
0	0	1
0	1	0
1	0	0
1	1	0

RS Flip Flop

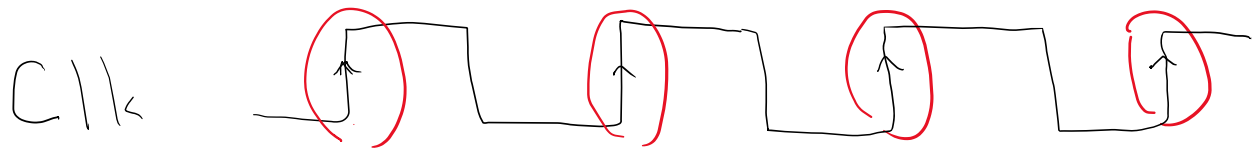
R	S	Q
0	0	Previous Q
0	1	1
1	0	0
1	1	Forbidden

JK flip flop (positive edge)

Clk ↓



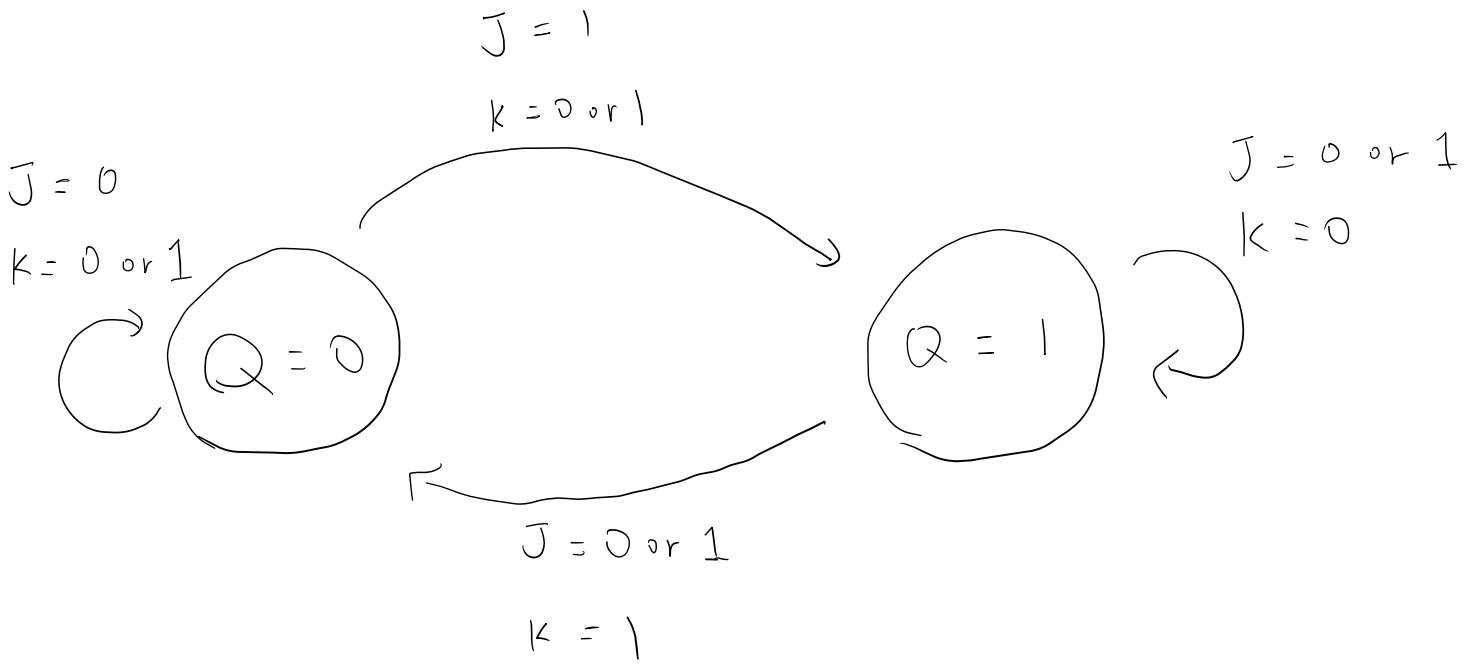
J	K	Q
0	0	prev Q
0	1	0
1	0	1
1	1	prev $\overline{Q}$



JK flip flop finite state machine

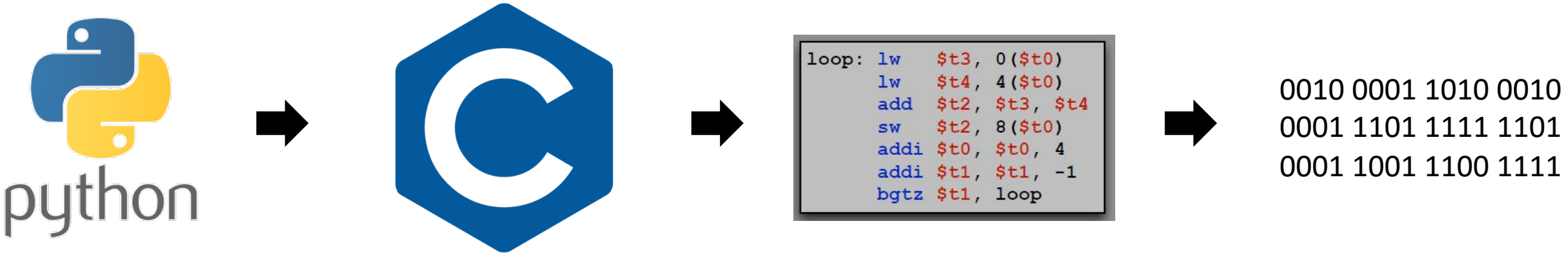
Clk ↓

J	K	Q
0	0	prev Q
0	1	0
1	0	1
1	1	prev $\overline{Q}$



Half time questions

Assembly language



Assembly to binary

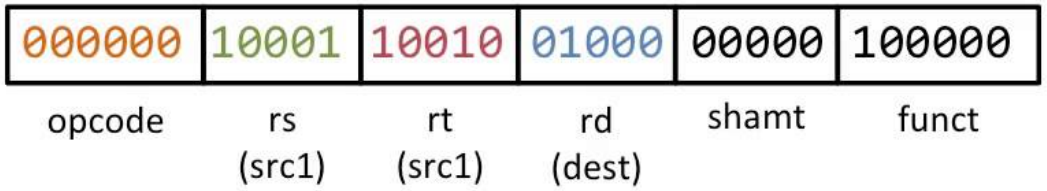
Instruction format (machine language)

- **Machine Language**
  - Computers do not understand “add R8, R17, R18”
  - Instructions are translated to machine language (1s and 0s)

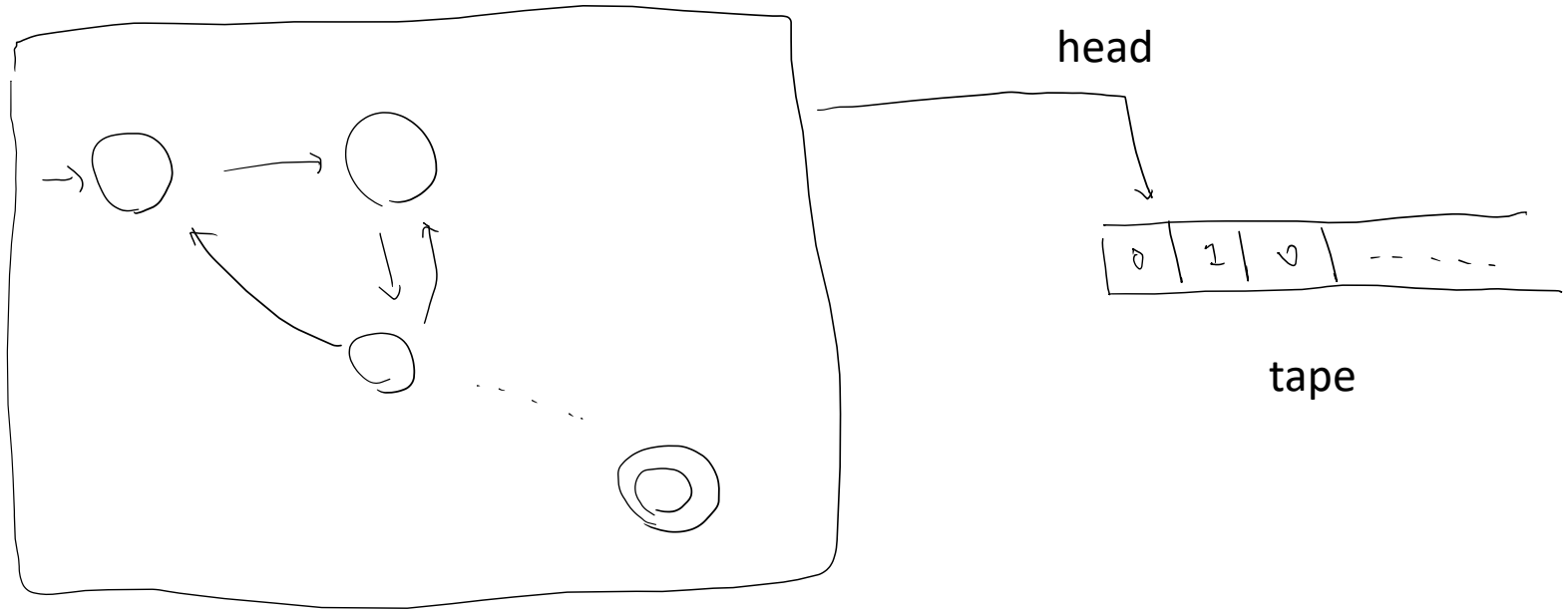
- **Example:**

```
add R8, R17, R18 →
00000010 00110010 01000000 00100000
```

- MIPS instructions have logical fields:



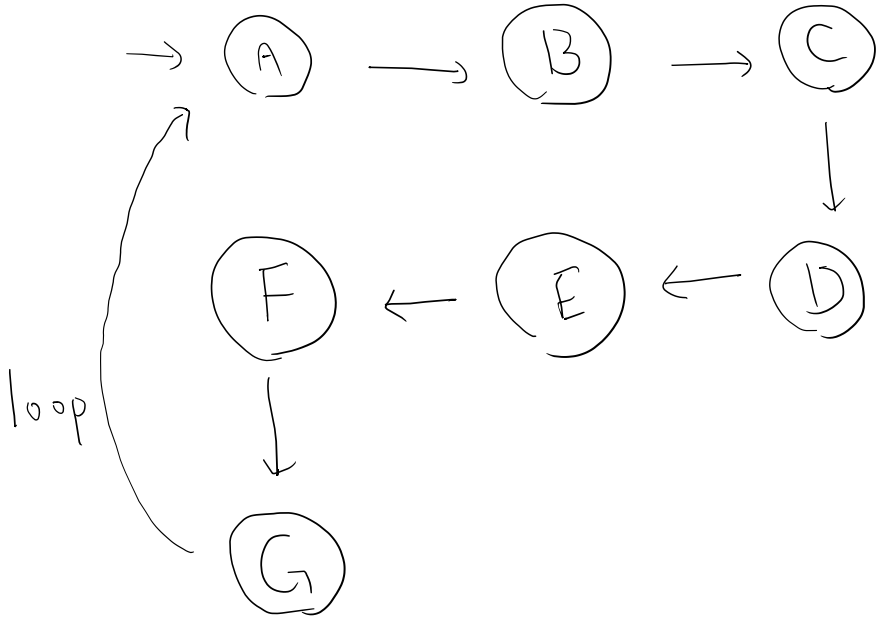
Turing machine



Finite state machine

We design/program finite state machine to perform a task on the tap.

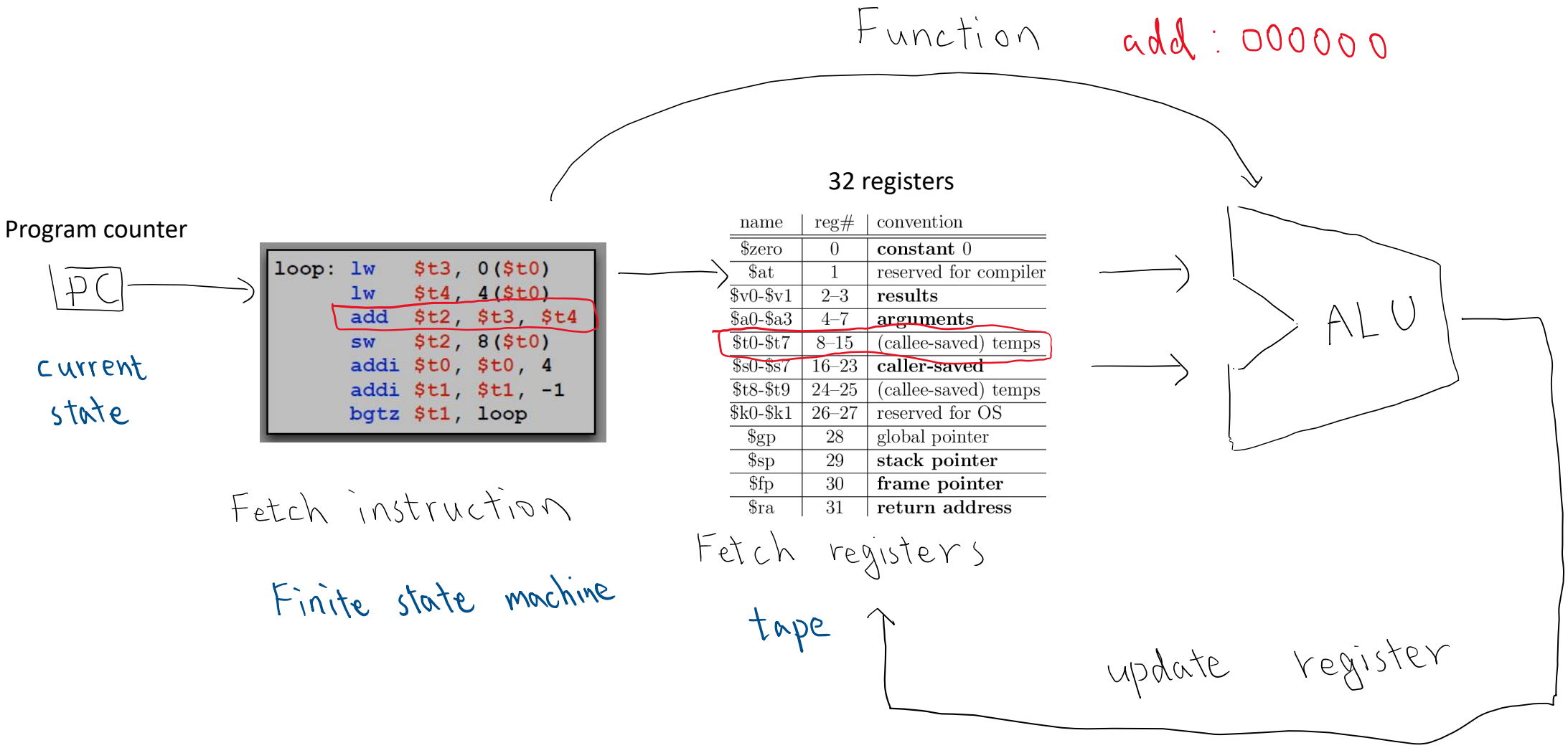
Finite state and Assembly code



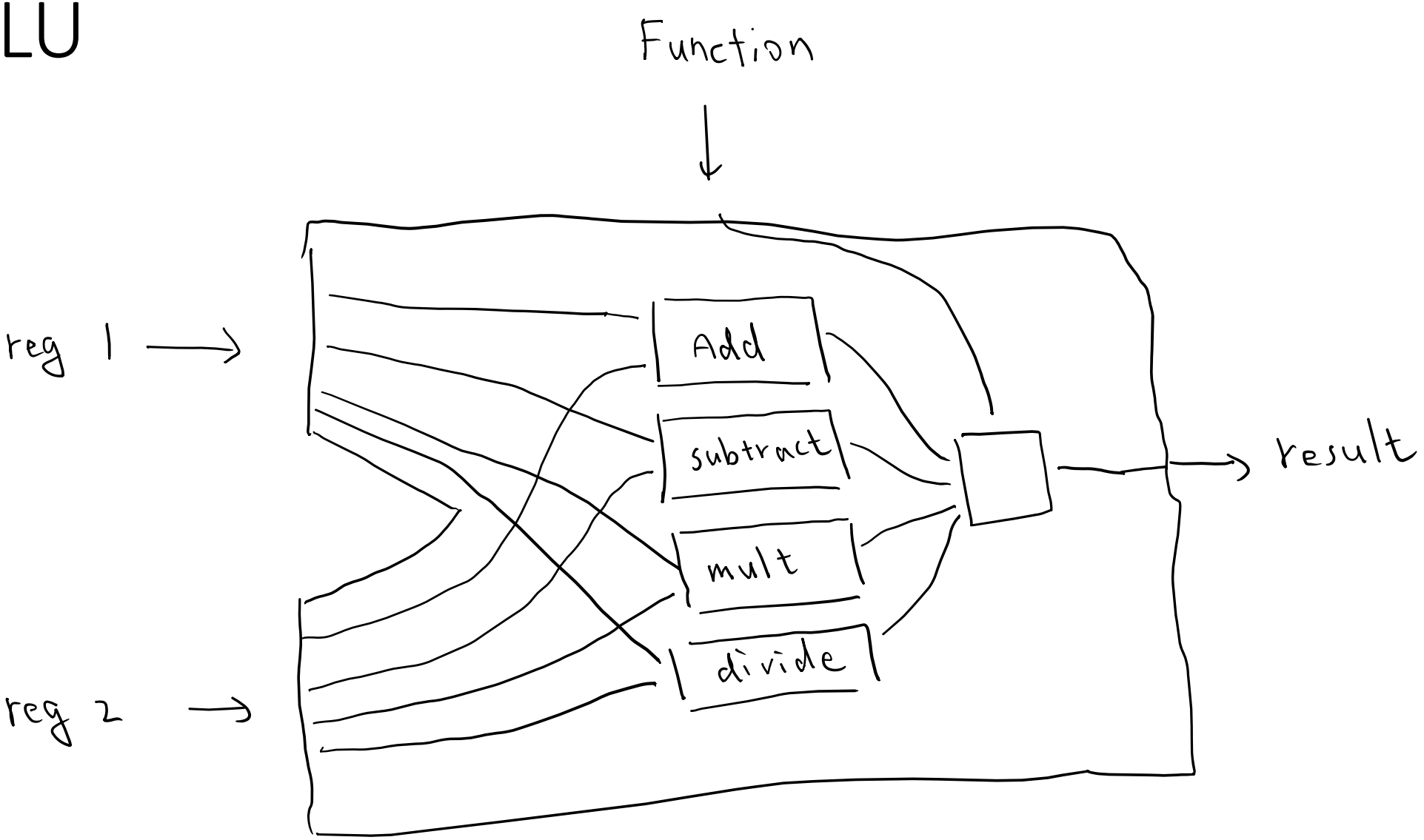

```
loop: lw   $t3, 0($t0)
      lw   $t4, 4($t0)
      add  $t2, $t3, $t4
      sw   $t2, 8($t0)
      addi $t0, $t0, 4
      addi $t1, $t1, -1
      bgtz $t1, loop
```

A
B
C
D
E
F
G

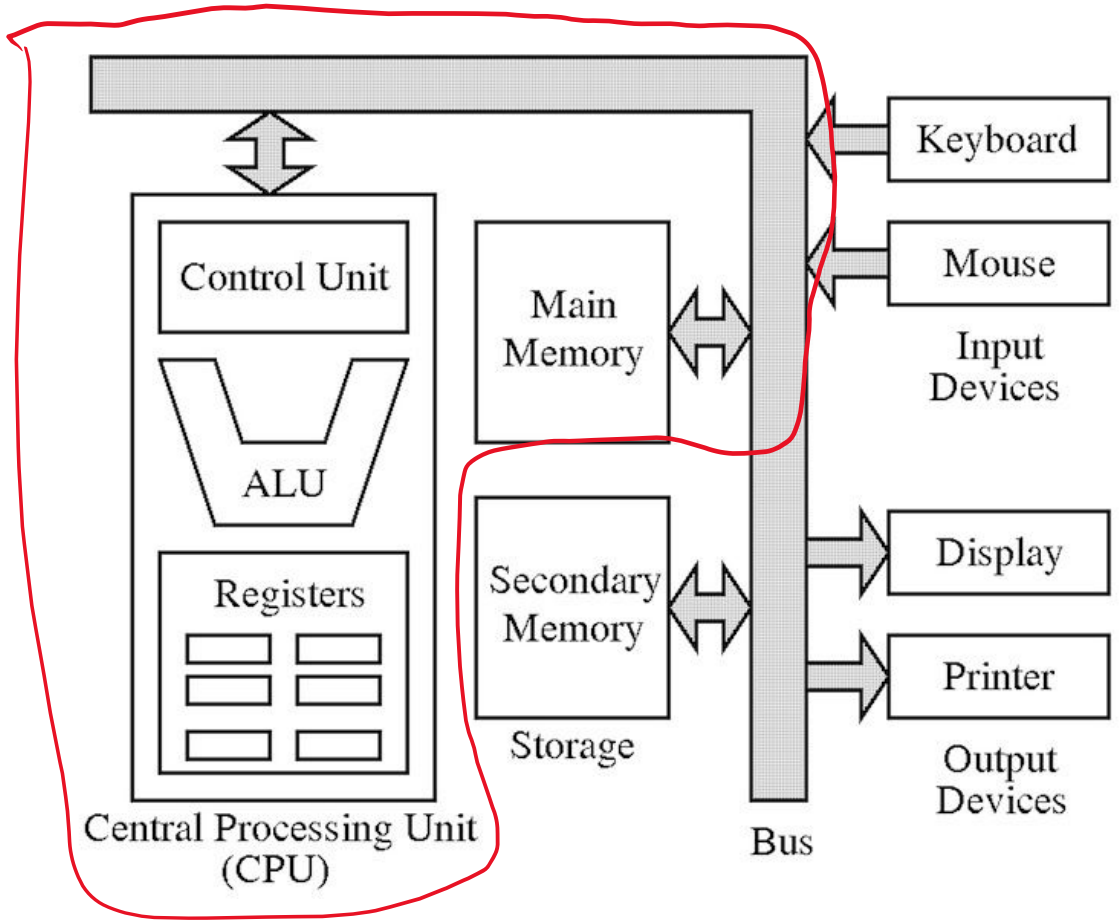
CPU architecture (single cycle)



ALU

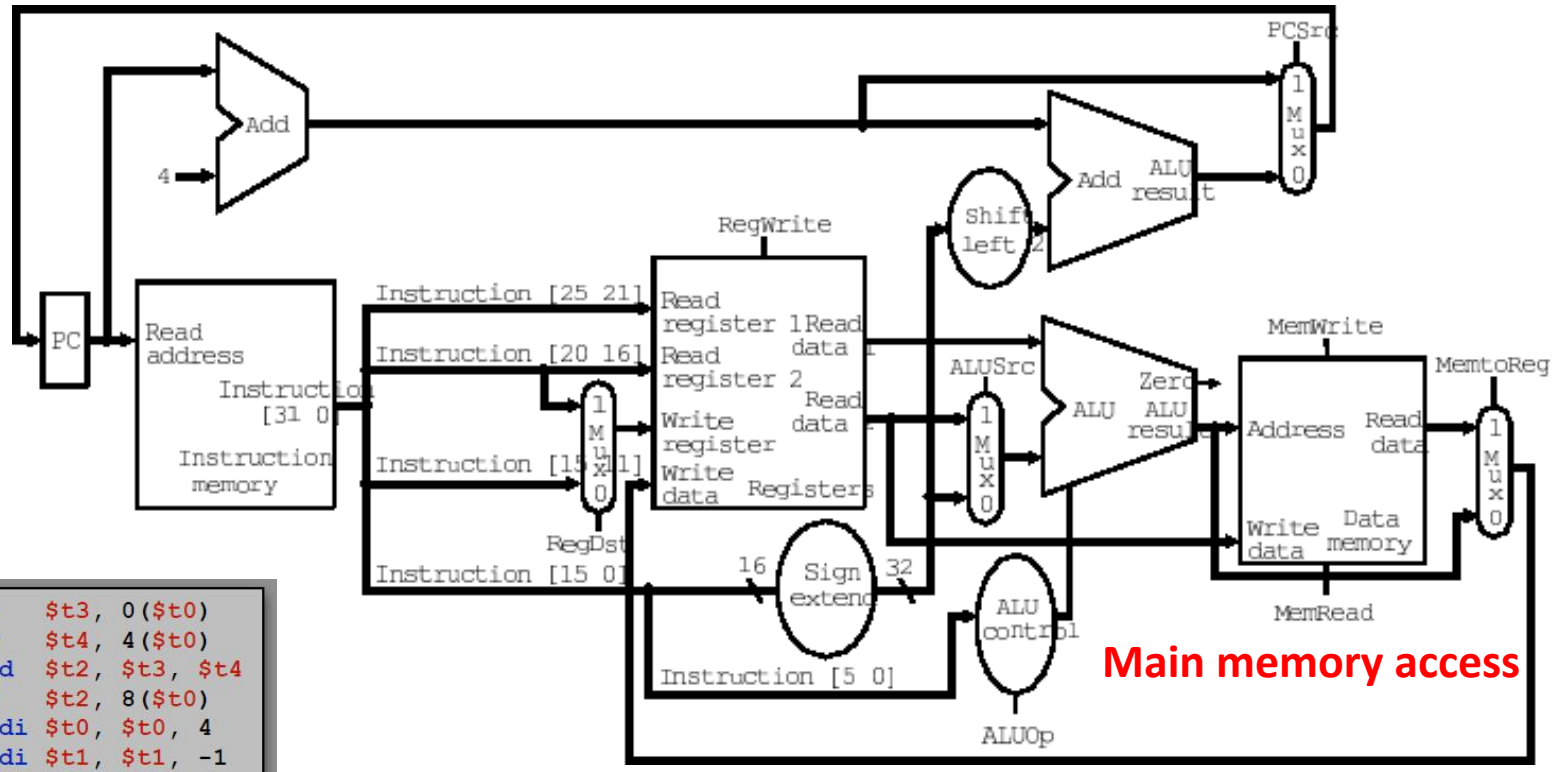


Register and stack memory



Turing machine assume the size of tap is infinite. Memory in registers is limited, so we need main memory (RAM) which is large enough to be considered as infinite space.

Single cycle CPU

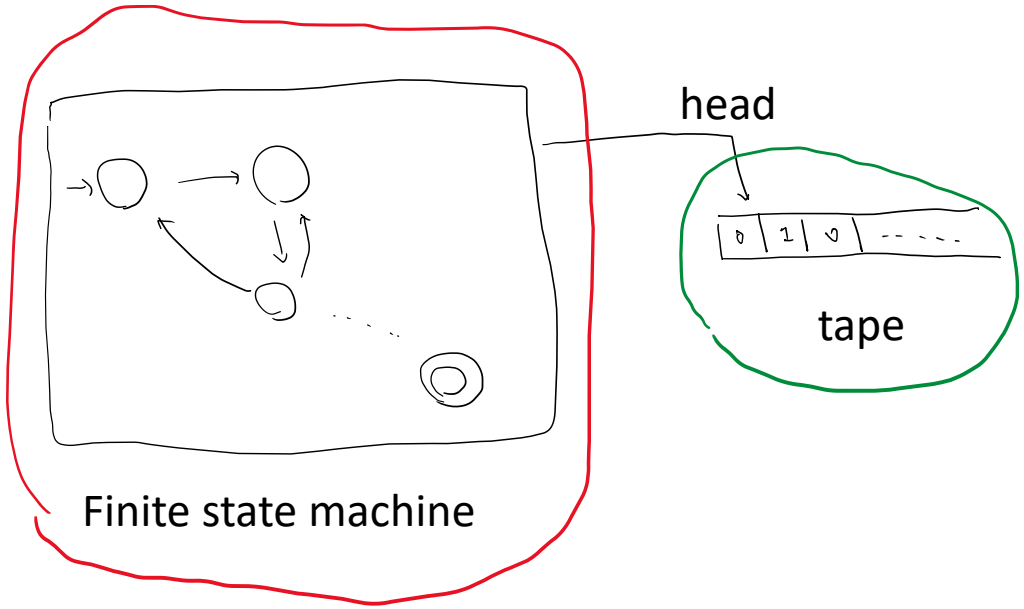


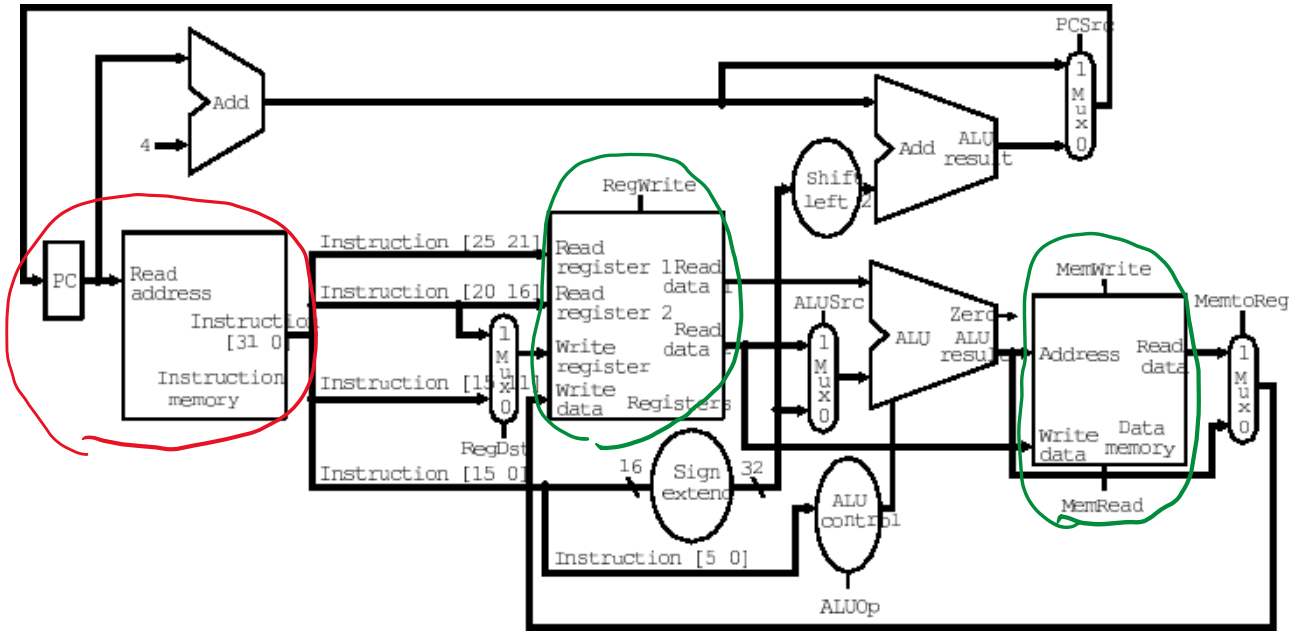

```
loop: lw   $t3, 0($t0)
      lw   $t4, 4($t0)
      add  $t2, $t3, $t4
      sw   $t2, 8($t0)
      addi $t0, $t0, 4
      addi $t1, $t1, -1
      bgtz $t1, loop
```

32 bits instruction

PC ticks on clock, and the period of clock is set to a sufficient time that completes any functional calculation aka. instruction.

Turing machine vs. MIPS CPU

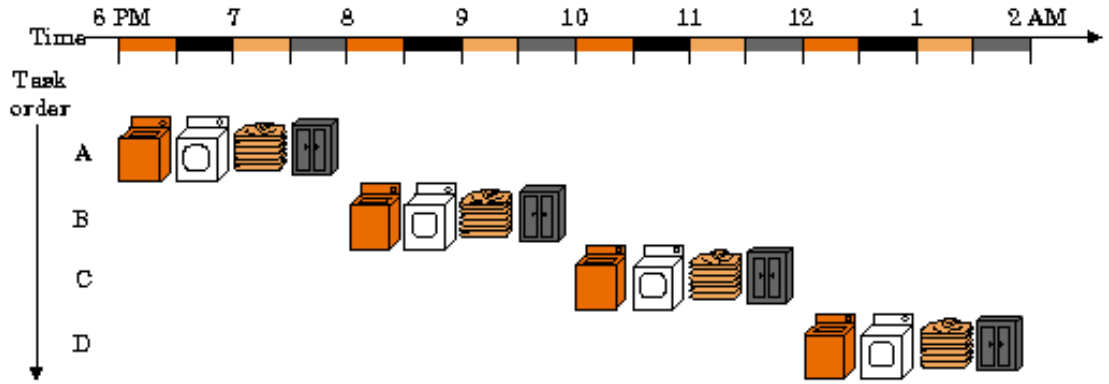


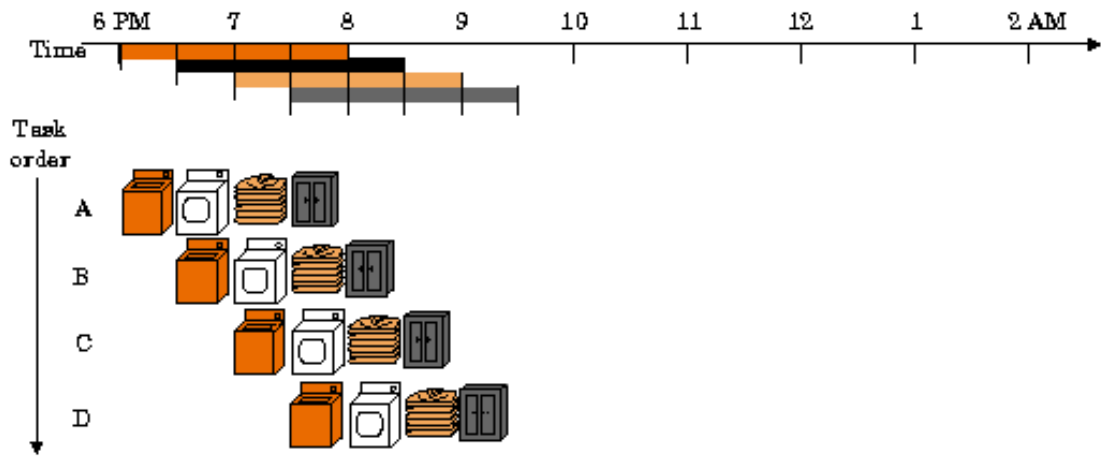


Facts of MIPS CPU

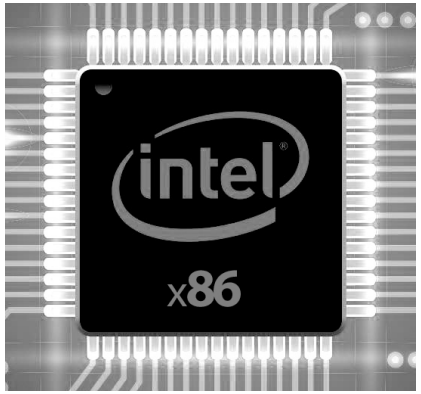
- Can do single-cycle and multi-cycle

- It’s for education. it’s not a real CPU used in the industry.

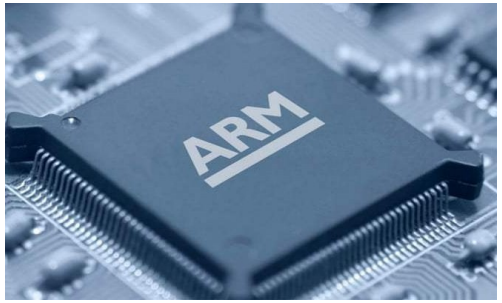




Modern CPU architecture



Private
CISC
Personal computer
and supercomputer



Private
RISC
Smartphone, apple
laptop with M1 chip,
and supercomputer

RISC-V

Public (open source)
RISC

Since Apple proved that ARM is way better than x86 on PC through M1 chip, I don't recommend investing money in intel or AMD.