Chapter 7:
ALU and CPU creation

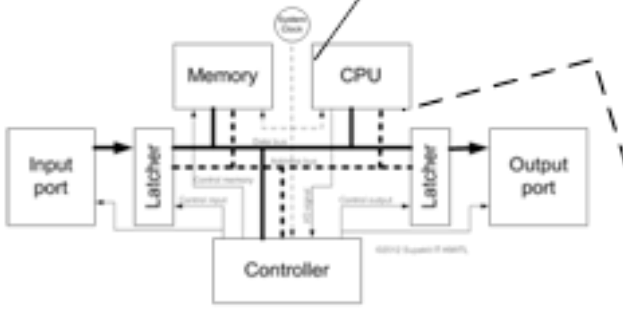
Asst.Prof.Dr.Supakit Nootyaskool

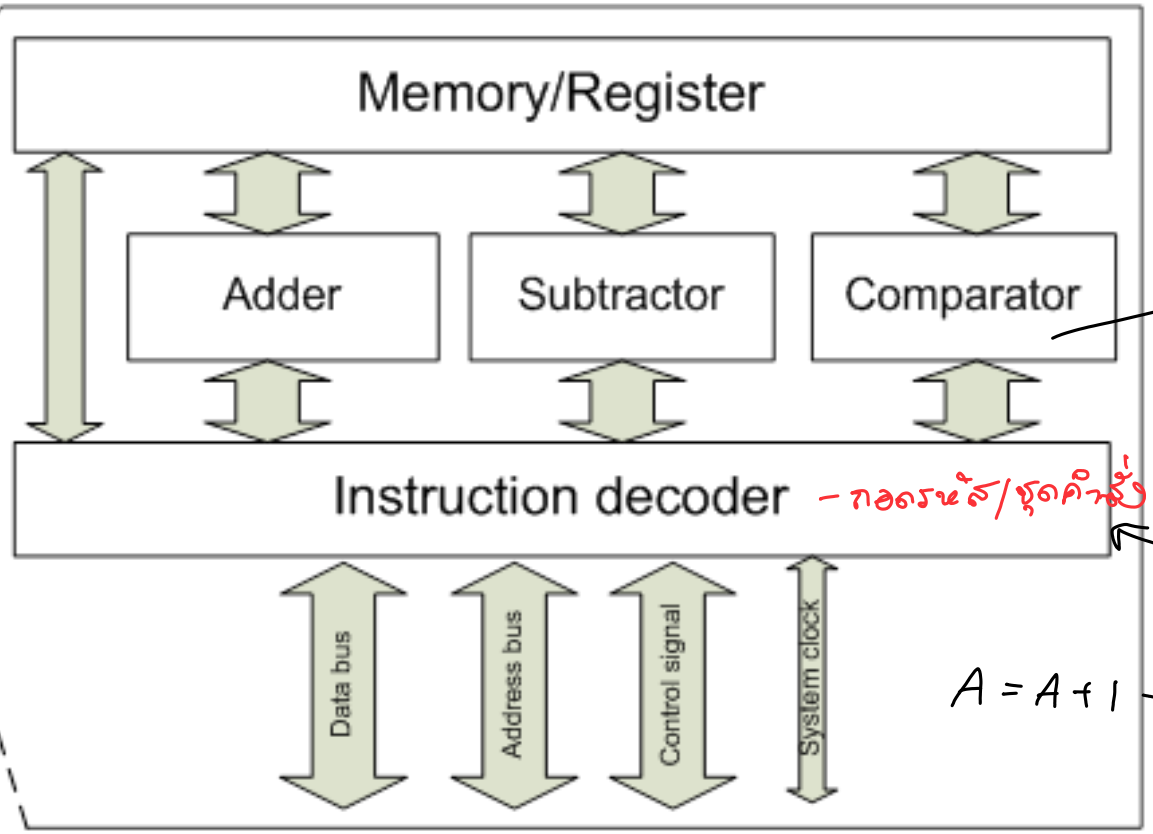
Objective

- To understand the functioning and structure arithmetic logic circuits.
- To recognize the functional concept of the CPU design.
- how cpu was created?

GPUรุ่นใหม่ → สถาปัตยกรรม Vector

CPU structure





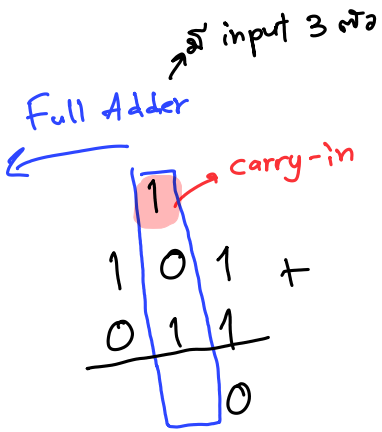
ส่วนคำนวณ

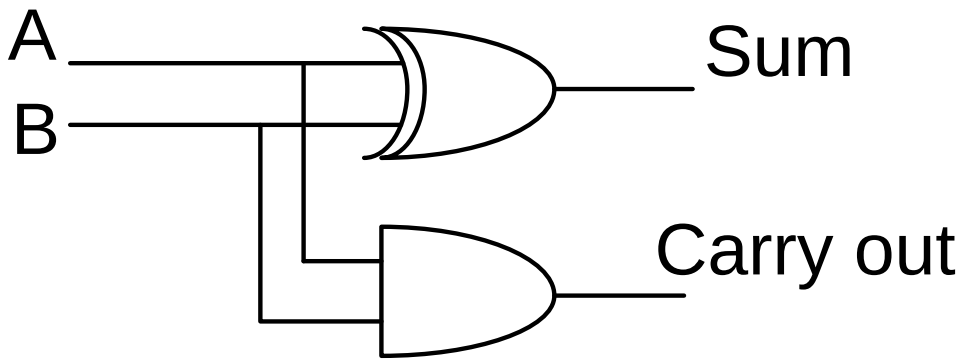
- ถอดรหัส/ชุดคำสั่ง

I O S

A = A + 1 → Assembly

1Bit Half Adder circuit

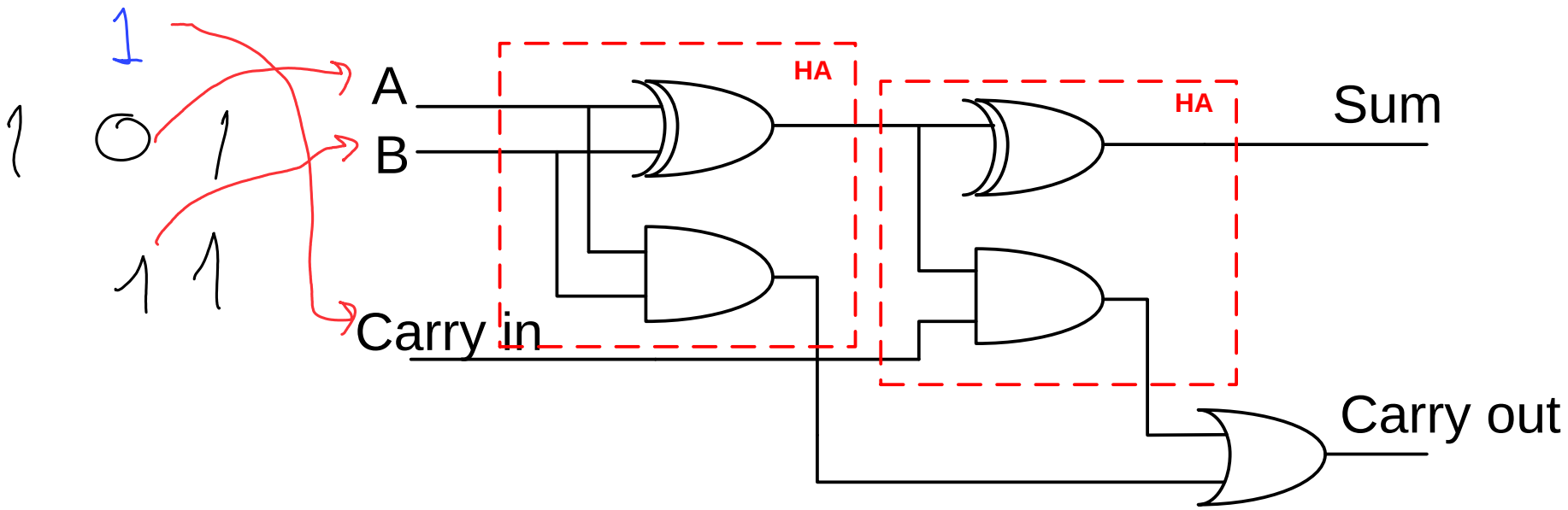




XOR AND

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

1Bit Full adder circuit

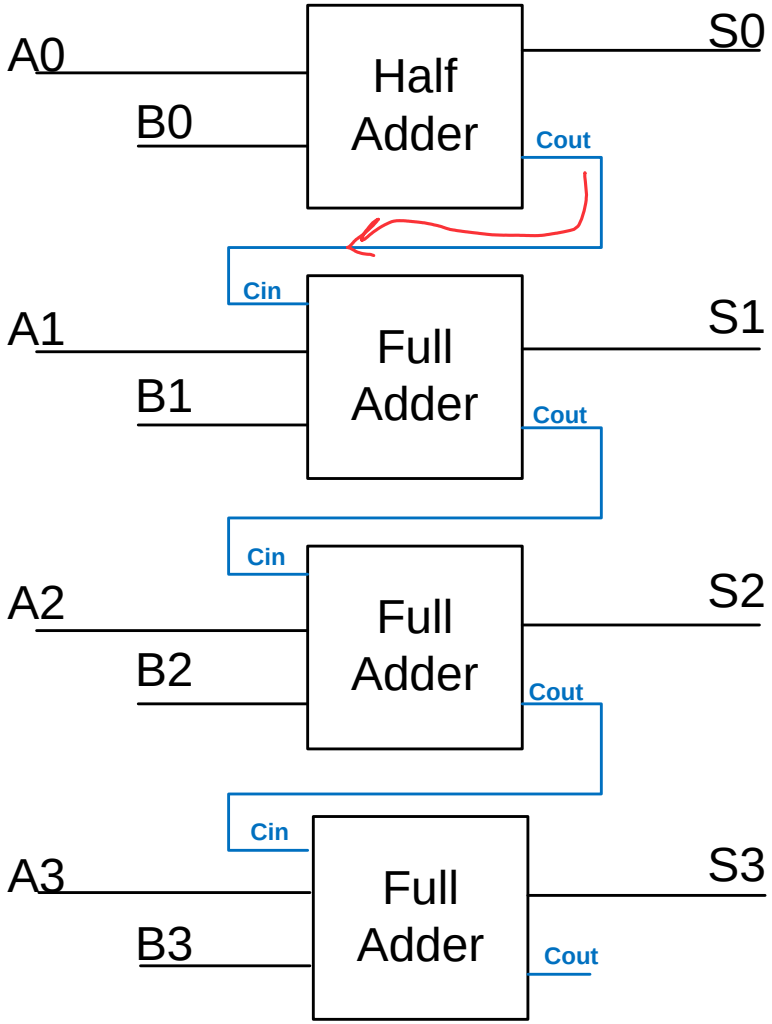


Carry in	A	B	Sum	Carry out
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

4Bit adder circuit

8Bit

↓ ต่อไปเรื่อยๆ

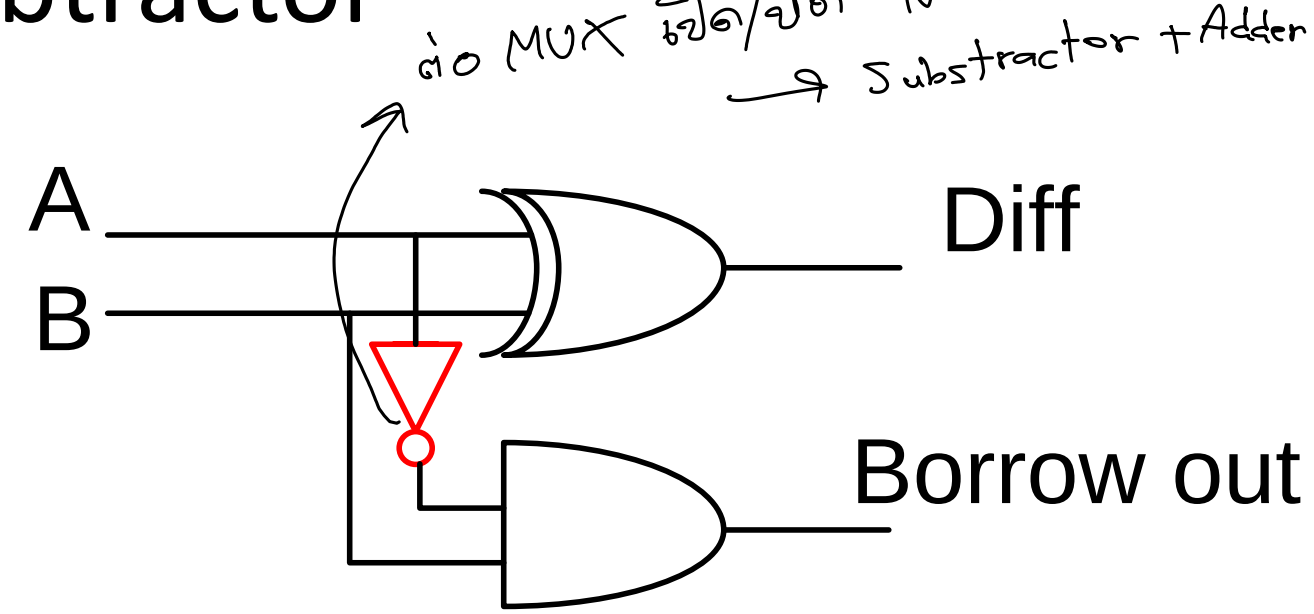


A = 1110

B = 0110

Ci		1	1	0	
A		1	1	1	0 +
B		0	1	1	0
S	0	1	0	0	
Co	1				

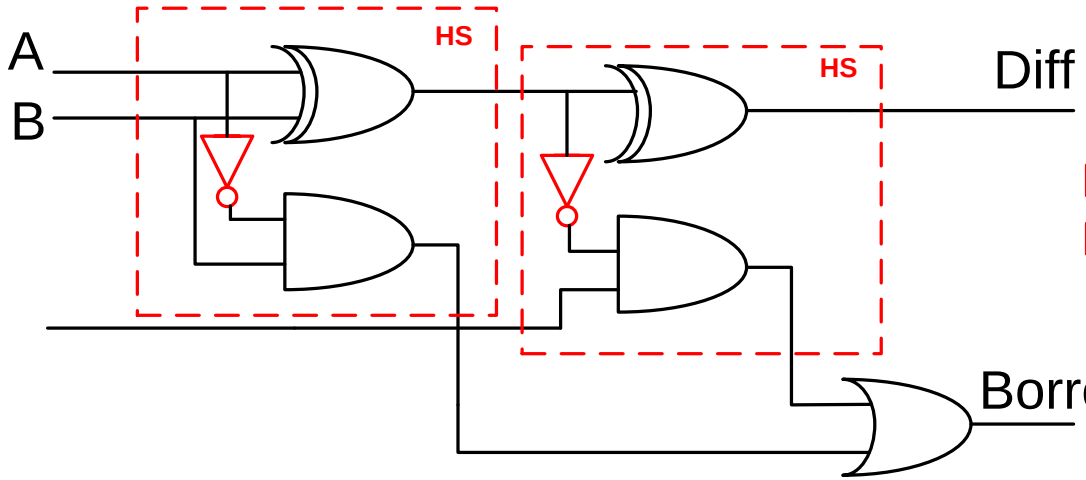
1Bit Half Subtractor



A	B	Diff	Borrow O
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

1Bit Full Subtractor

2hr Exam



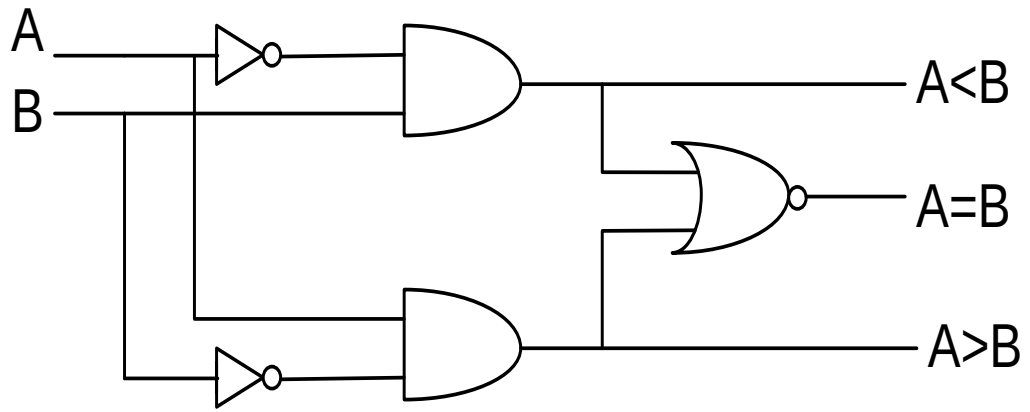
Rules

Diff = |A-B-Borrow|

BorrowO = 1 ; A<(B+Borrow)

A	B	Borrow I	Diff	Borrow O
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

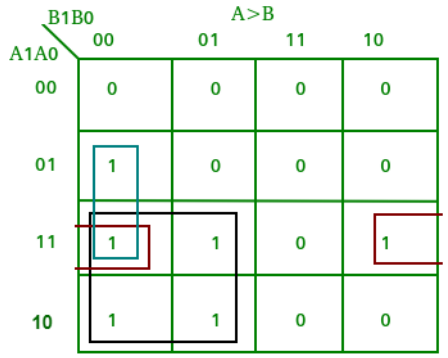
1 Bit Comparator circuit

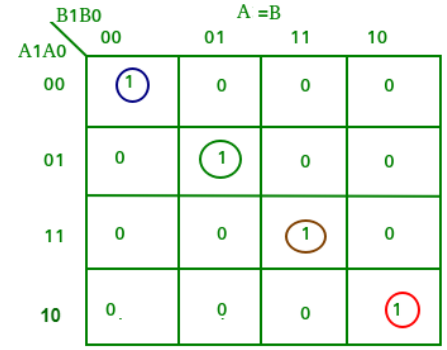


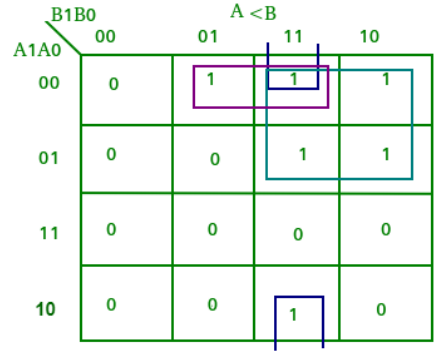
A	B	A<B	A=B	A>B
0	0	0	1	0
0	1	1	0	0
1	0	0	0	1
1	1	0	1	0

2Bit Comparator circuit

INPUT				OUTPUT		
A1	A0	B1	B0	A<B	A=B	A>B
0	0	0	0	0	1	0
0	0	0	1	1	0	0
0	0	1	0	1	0	0
0	0	1	1	1	0	0
0	1	0	0	0	0	1
0	1	0	1	0	1	0
0	1	1	0	1	0	0
0	1	1	1	1	0	0
1	0	0	0	0	0	1
1	0	0	1	0	0	1
1	0	1	0	0	1	0
1	0	1	1	1	0	0
1	1	0	0	0	0	1
1	1	0	1	0	0	1
1	1	1	0	0	0	1
1	1	1	1	0	1	0



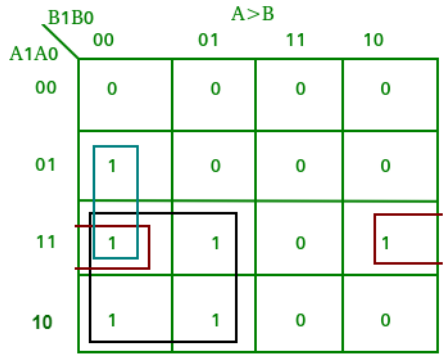


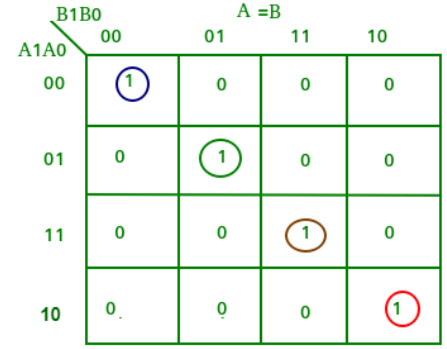


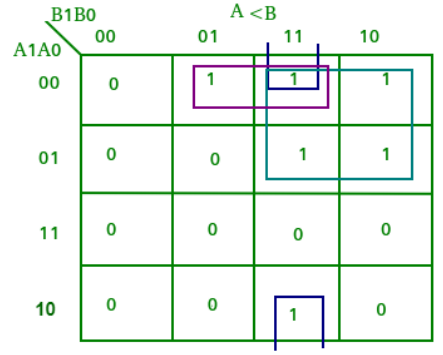
3circuit

2Bit Comparator circuit

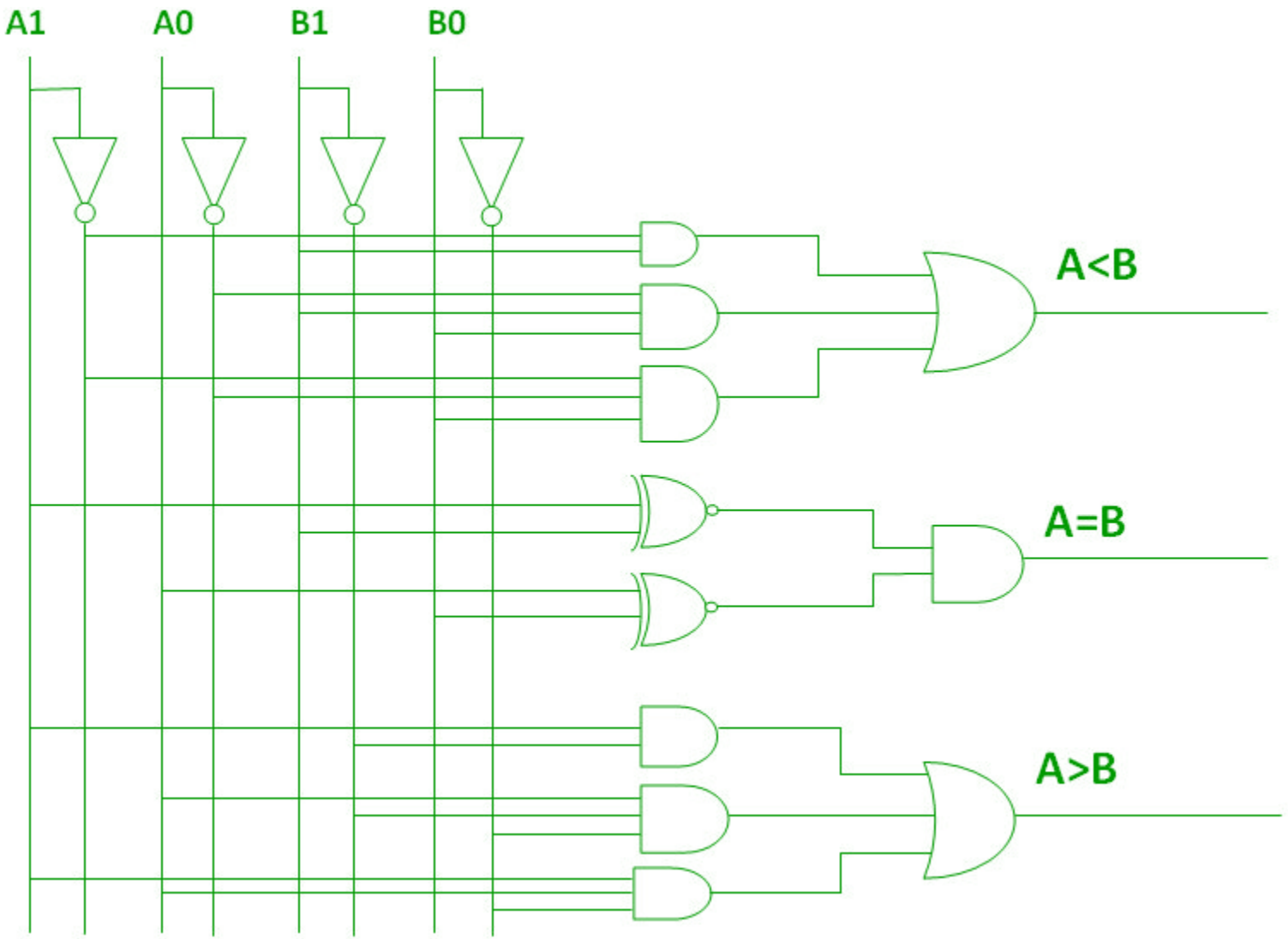
- A>B
  - A1B1’ + A0B1’B0’ + A1A0B0’
- A=B
  - A1’A0’B1’B0’ + A1’A0B1’B0 + A1A0B1B0 + A1A0’B1B0’
  - (A0 Ex-Nor B0) (A1 Ex-Nor B1)
- A<B
  - A1’B1 + A0’B1B0 + A1’A0’B0



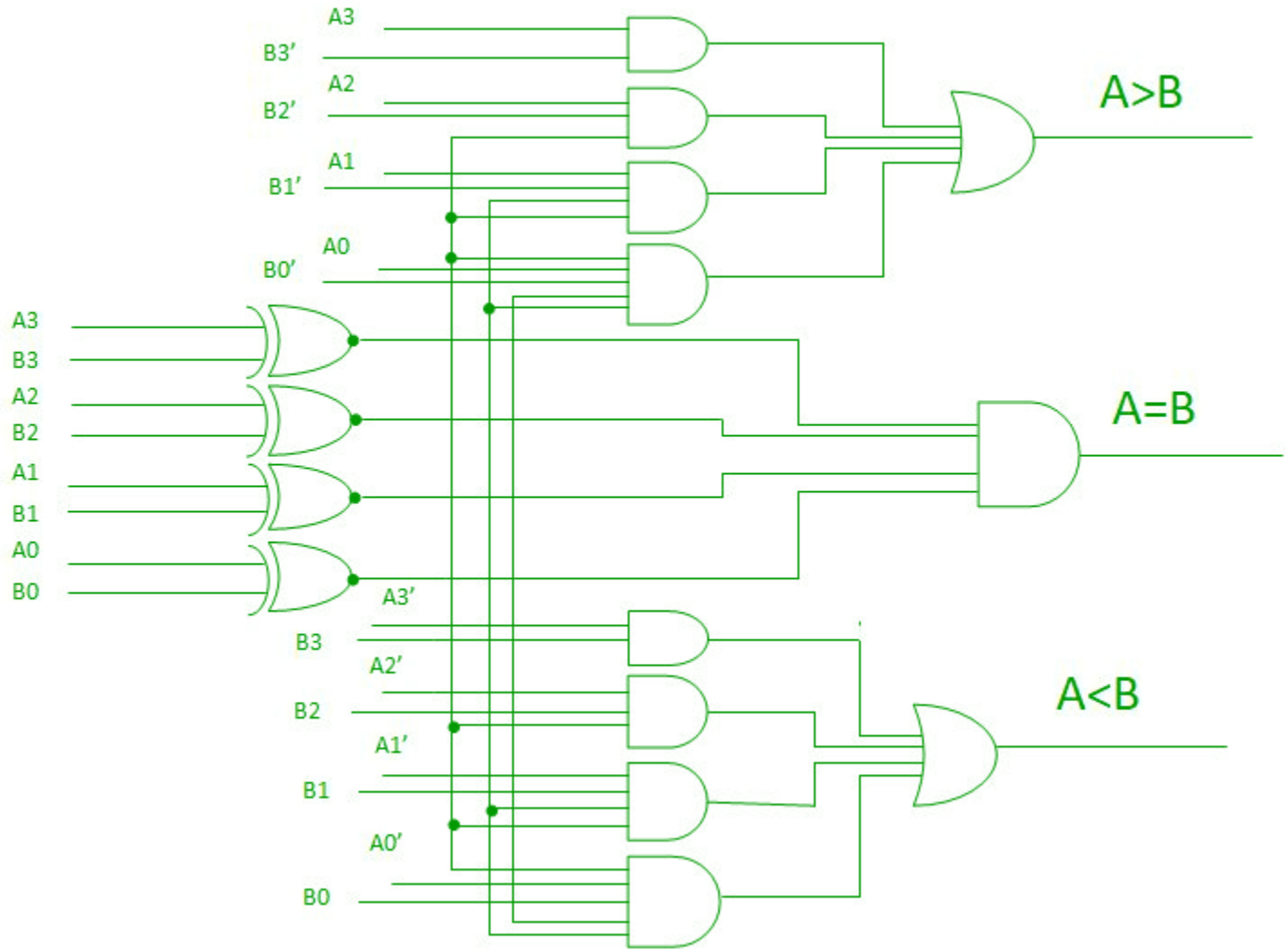




2Bit Comparator circuit

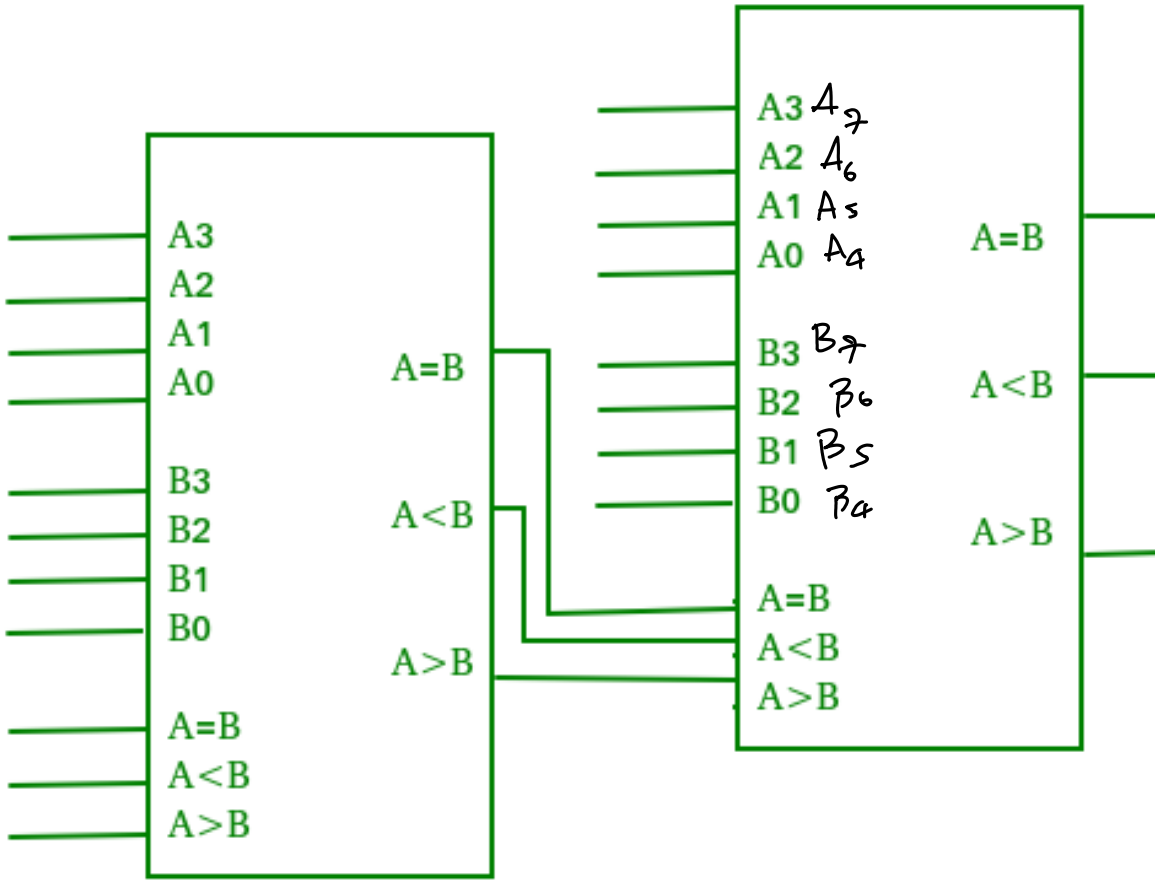


4Bit Comparator circuit



https://www.geeksforgeeks.org/magnitude-comparator-in-digital-logic/

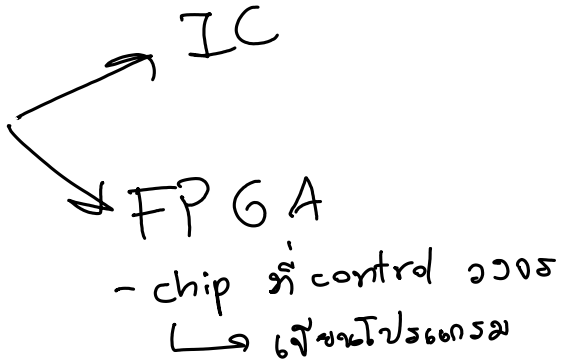
Cascading Comparator circuit



https://www.geeksforgeeks.org/magnitude-comparator-in-digital-logic/

4-BIT CPU CREATION

Create a simple CPU



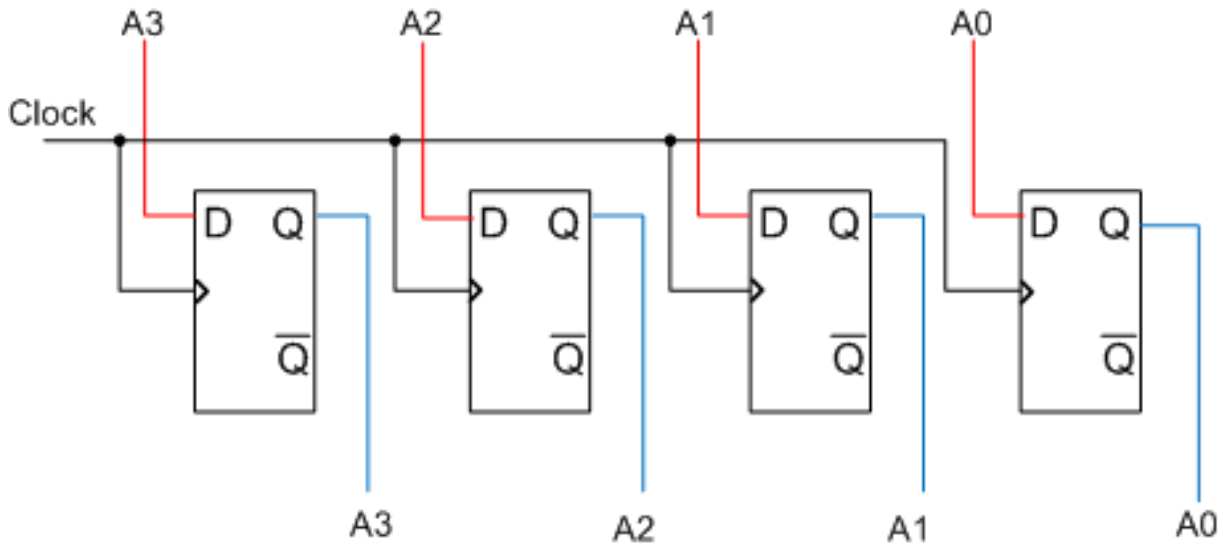
- Size: 4 Bits (≈1970s)
- 2 Register (A and B)

เก็บข้อมูล

Register

(เหมือน layer ของ variable)

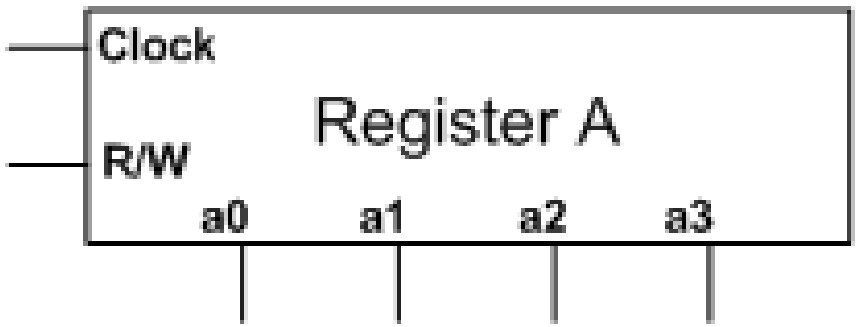
- The register is used collecting data like as variable
- Register create from a memory (D-FF)
- Example Register A

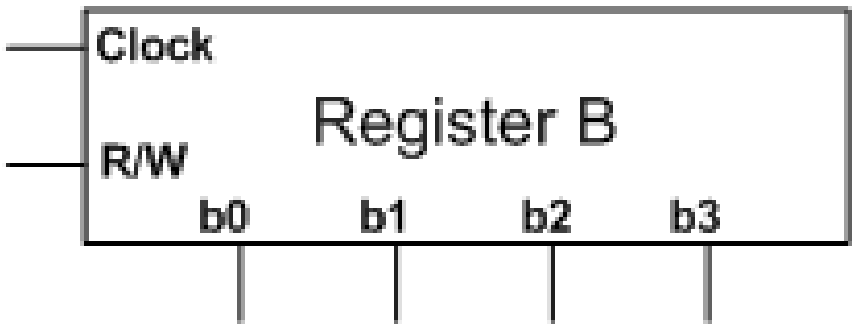


Latch

Register

- Signal
  - Clock
  - Read/Write
  - Input and Output (Bidirectional data)





ใส่กล่อง, เพิ่มขา read/write

มี tristate gate ขา control

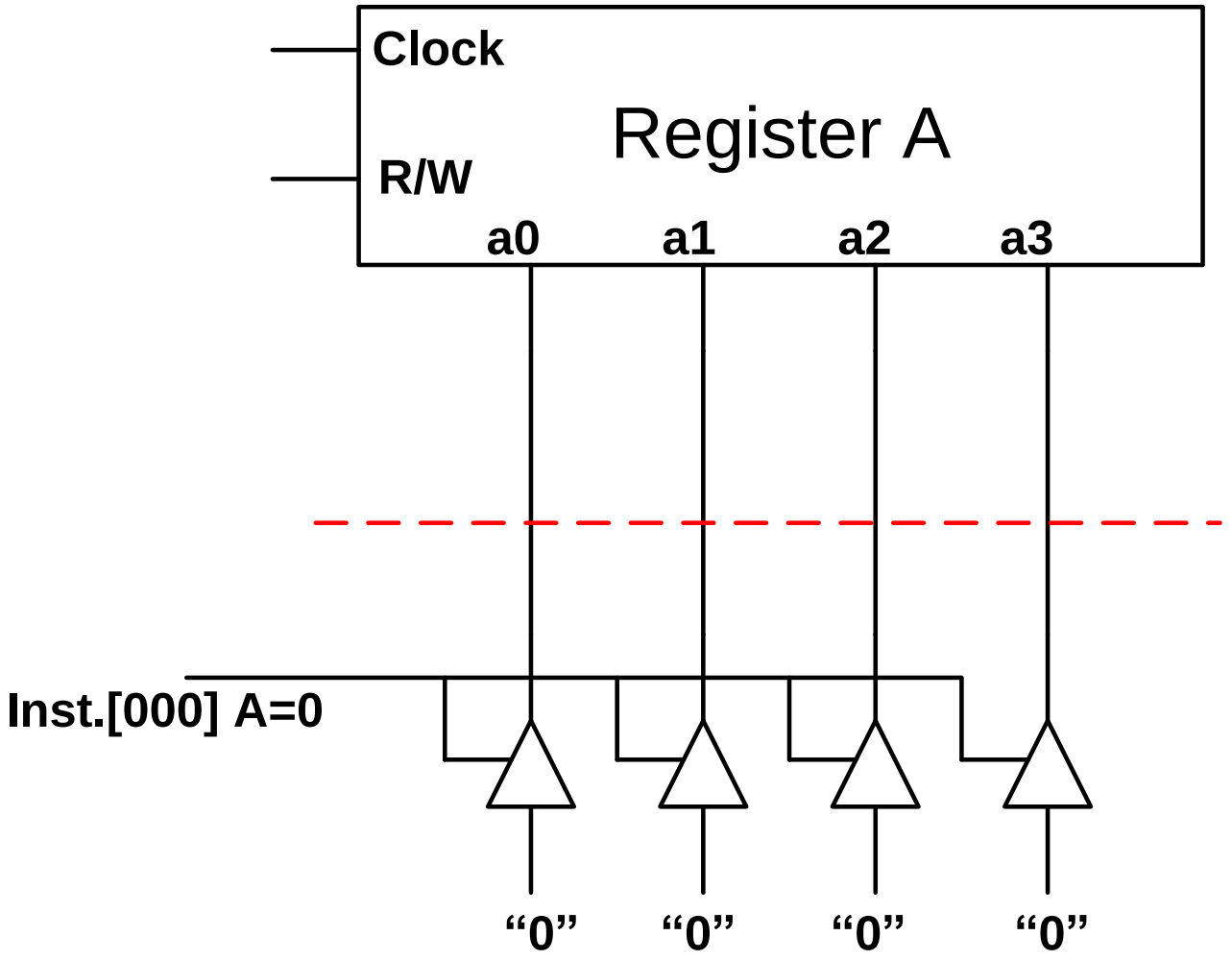
Instruction set

ก้าหนดชุดคำสั่ง

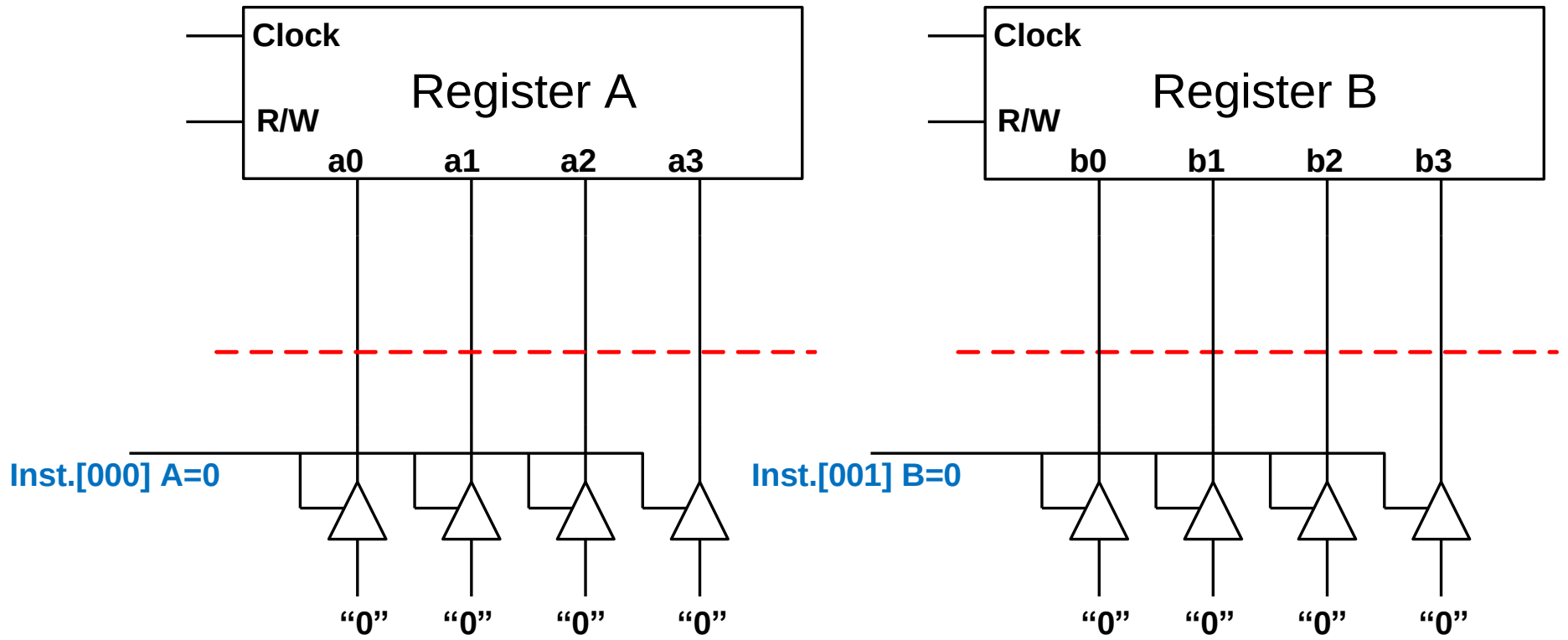
Inst. ID	Command
0 (000)	A = 0
1 (001)	B = 0
2 (010)	A = A +1
3 (011)	B = B + 1
4 (100)	A = A+B
5 (101)	A <B {Compare Bit (CB) = 1 and otherwise CB = 0}
6 (110)	Out A
7 (111)	Out B

→ out display

Inst. [000] A = 0

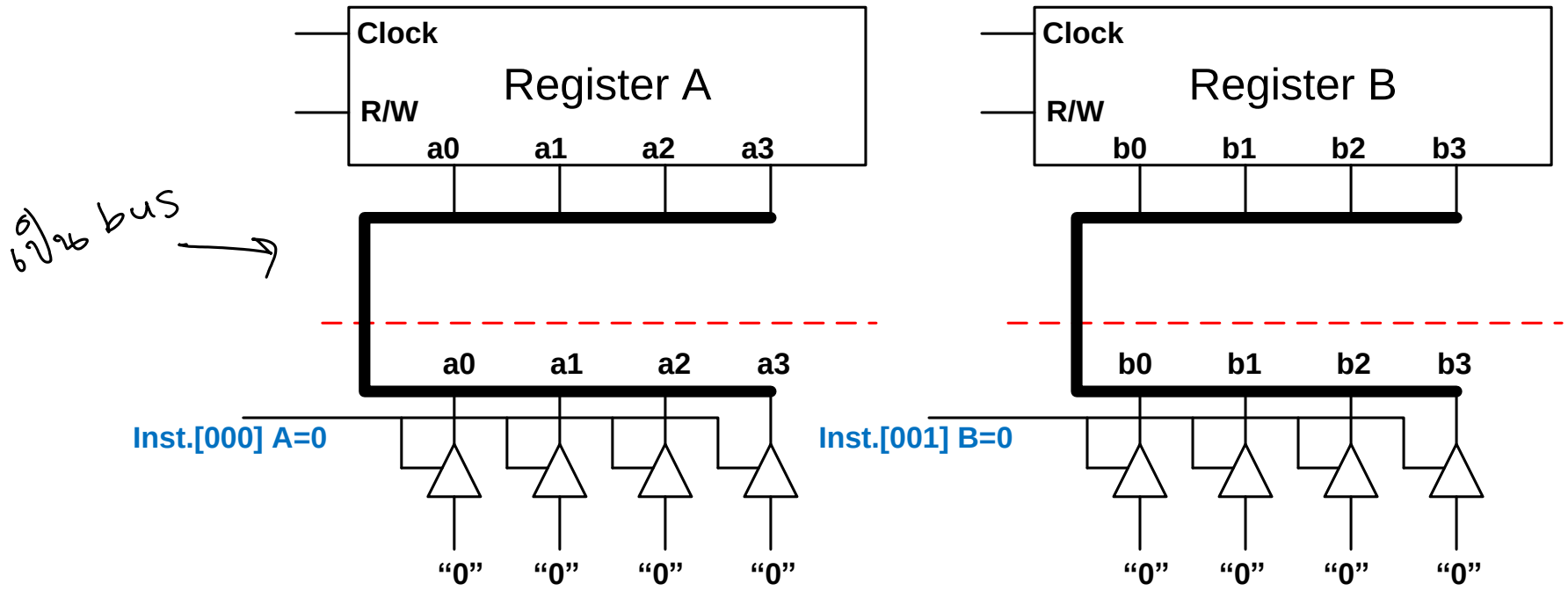


Inst. [001] B = 0


Clock
R/W
Register A
a0
a1
a2
a3
Clock
R/W
Register B
b0
b1
b2
b3
Inst.[000] A=0
Inst.[001] B=0
“0”
“0”
“0”
“0”
“0”
“0”
“0”
“0”

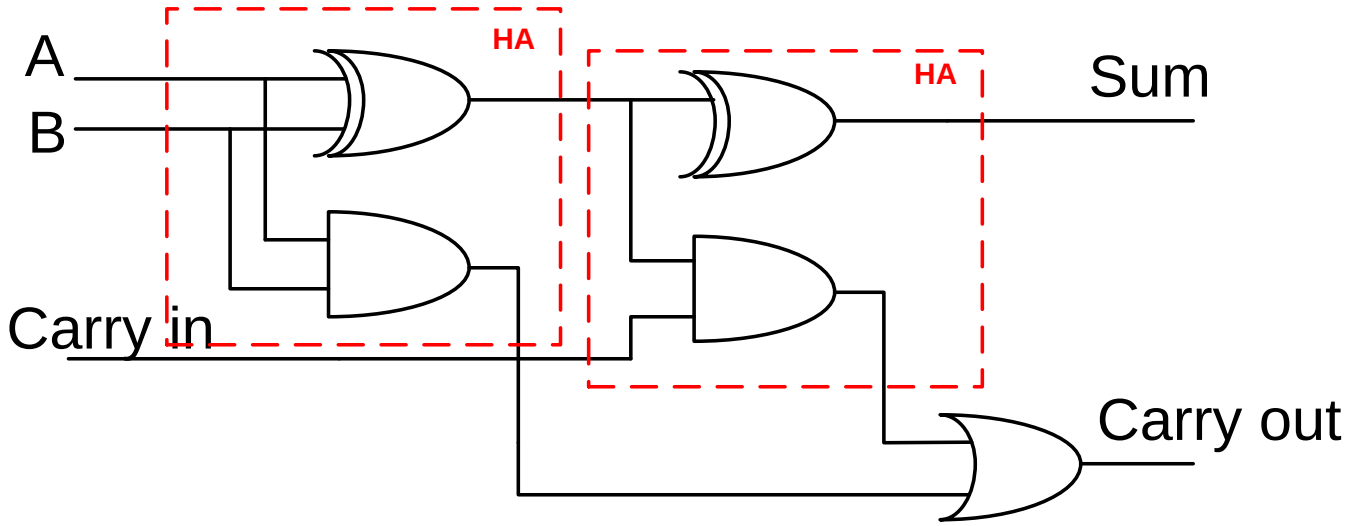
Bus

- A bus is a group of wires that uses in circuit design to reduce the number of the connection.

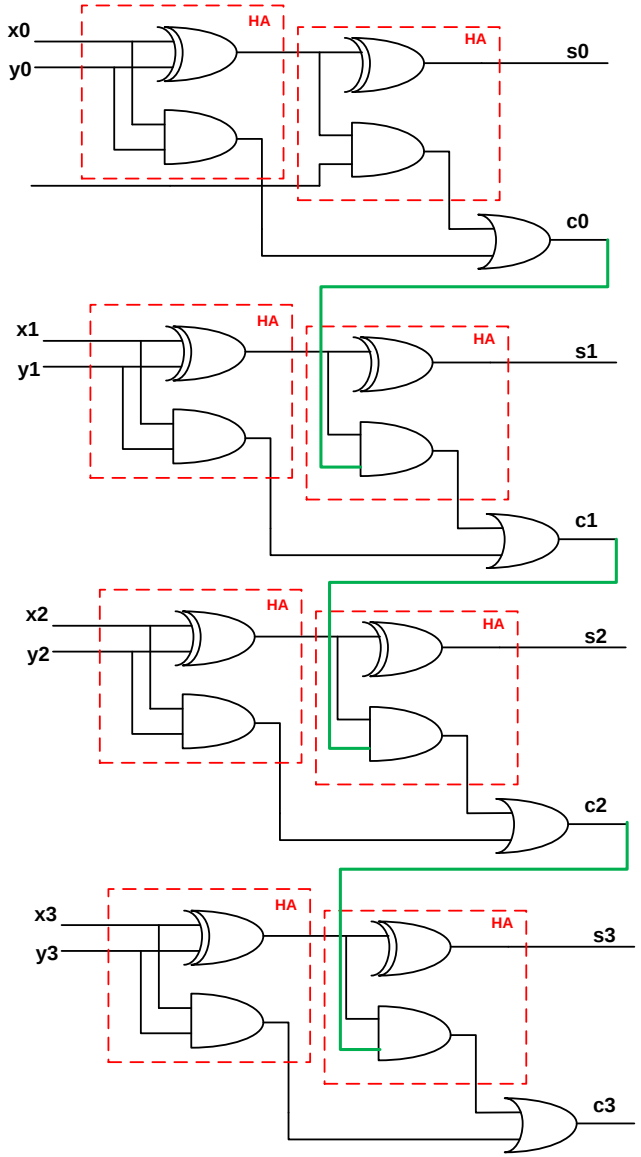


Inst. [010] A=A+1

- 1 Bit full-adder (FA)

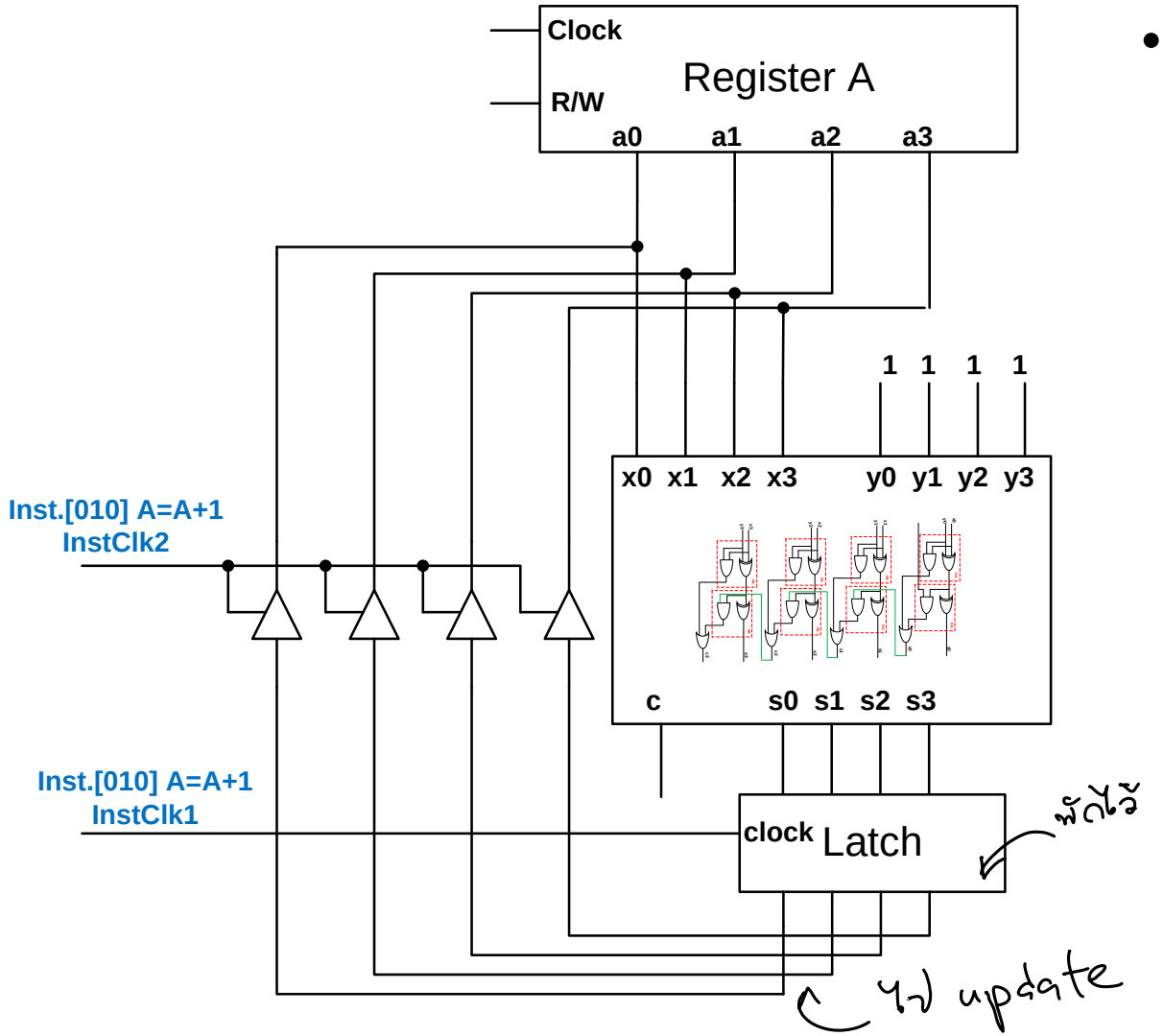


Inst. [010] A=A+1



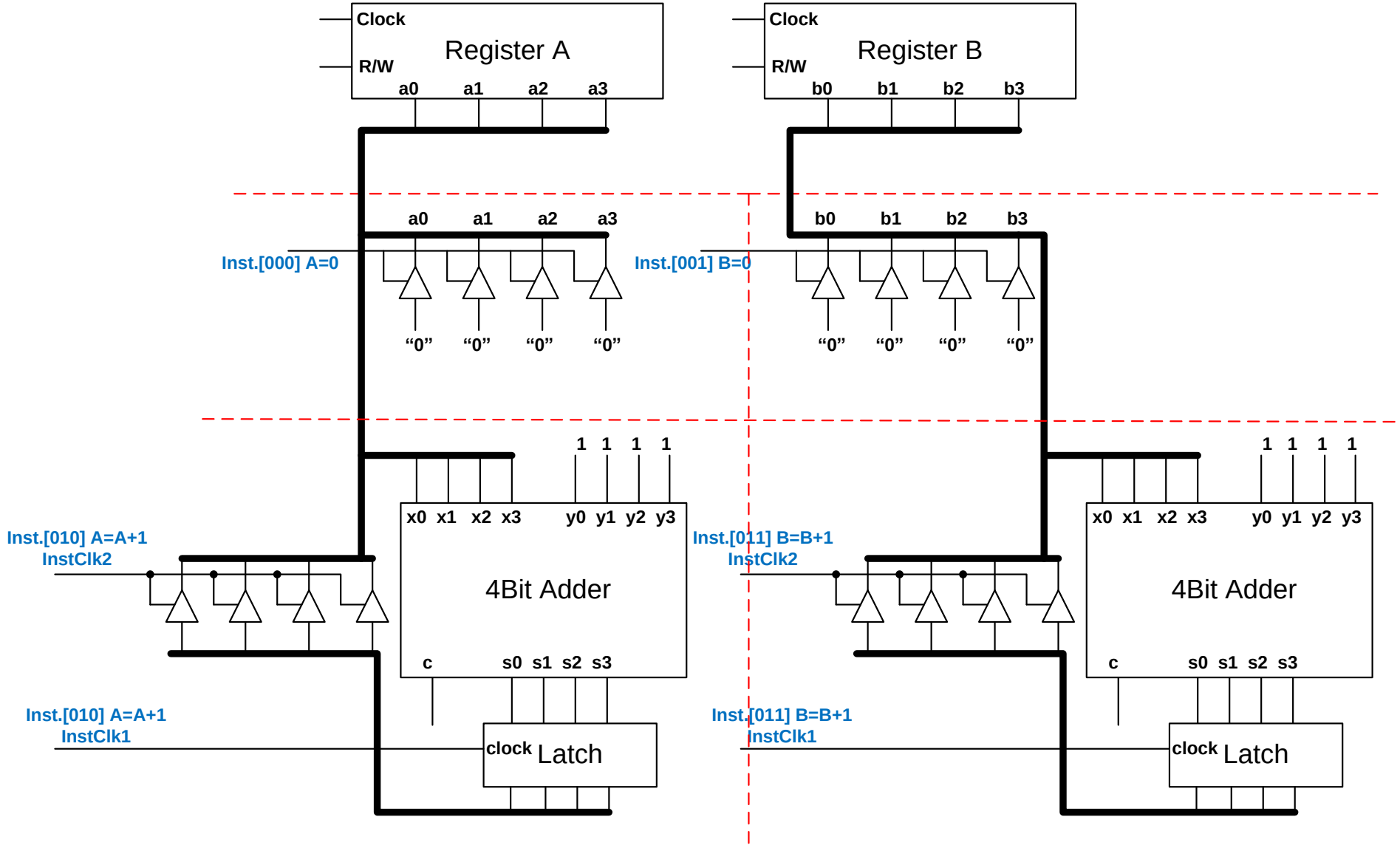
- 4 Bit adder circuit

Inst. [010] A=A+1



- Two clocks command
  - Calculate A +1 and keeping in Latch
  - Load data from latch to Register A

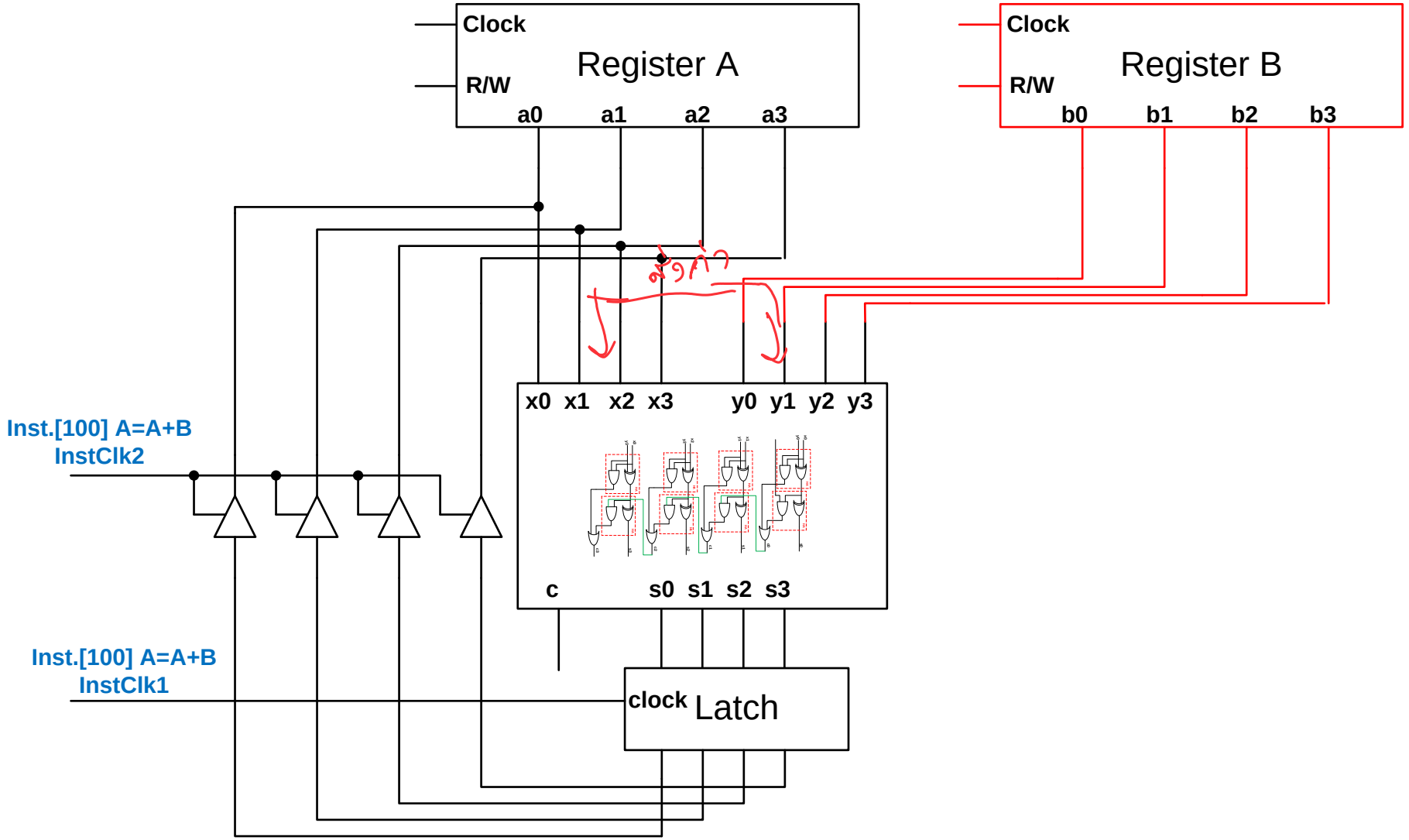
Inst. [011] B=B+1


Clock
R/W
Register A
a0 a1 a2 a3
Clock
R/W
Register B
b0 b1 b2 b3
Inst.[000] A=0
a0 a1 a2 a3
"0" "0" "0" "0"
Inst.[001] B=0
b0 b1 b2 b3
"0" "0" "0" "0"
Inst.[010] A=A+1
InstClk2
1 1 1 1
x0 x1 x2 x3 y0 y1 y2 y3
4Bit Adder
c s0 s1 s2 s3
Inst.[011] B=B+1
InstClk2
1 1 1 1
x0 x1 x2 x3 y0 y1 y2 y3
4Bit Adder
c s0 s1 s2 s3
Inst.[010] A=A+1
InstClk1
clock Latch
Inst.[011] B=B+1
InstClk1
clock Latch

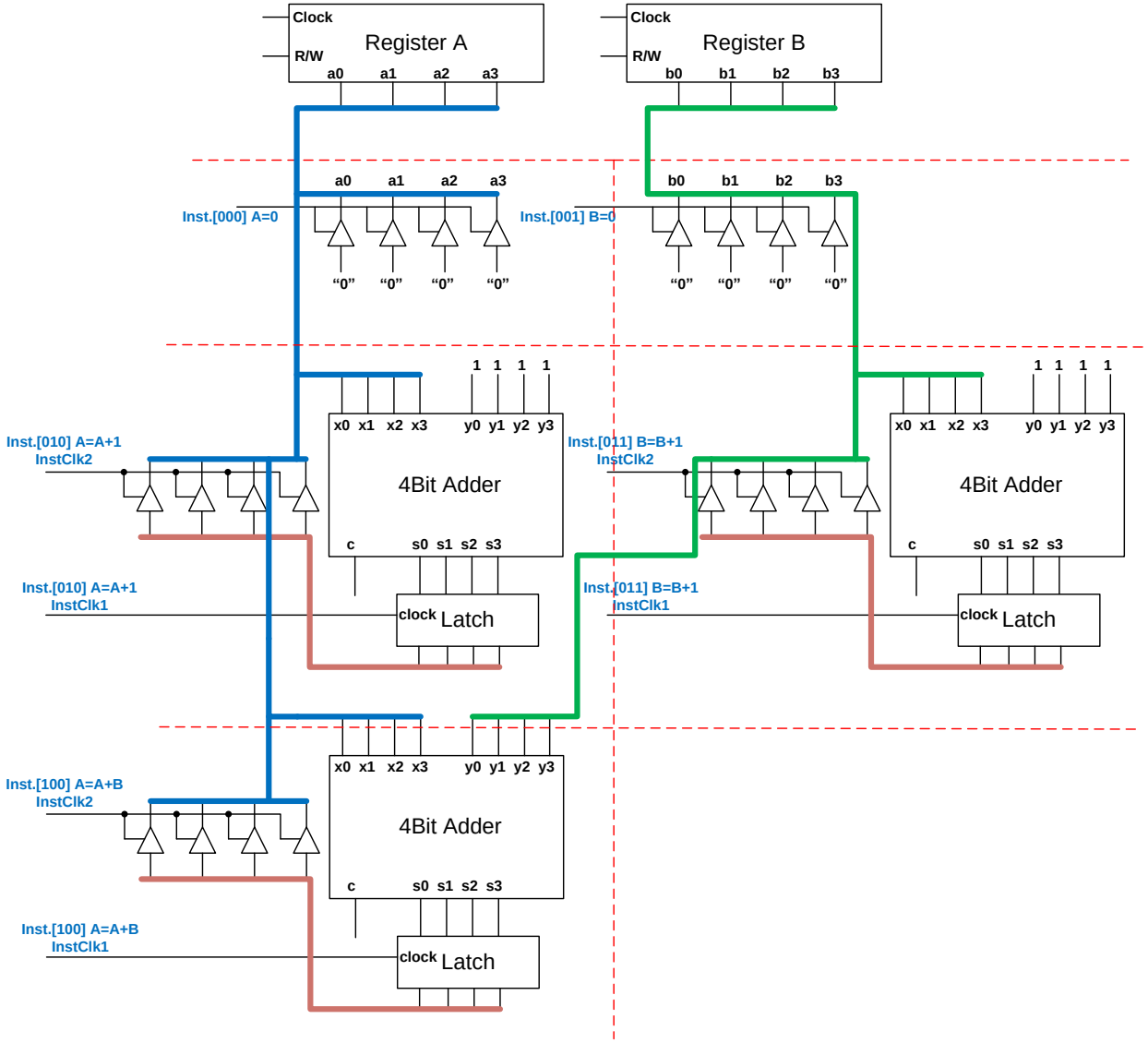
Inst. [100] A = A+B

- How do we build A = A+B circuit?

Inst. [100] A = A+B

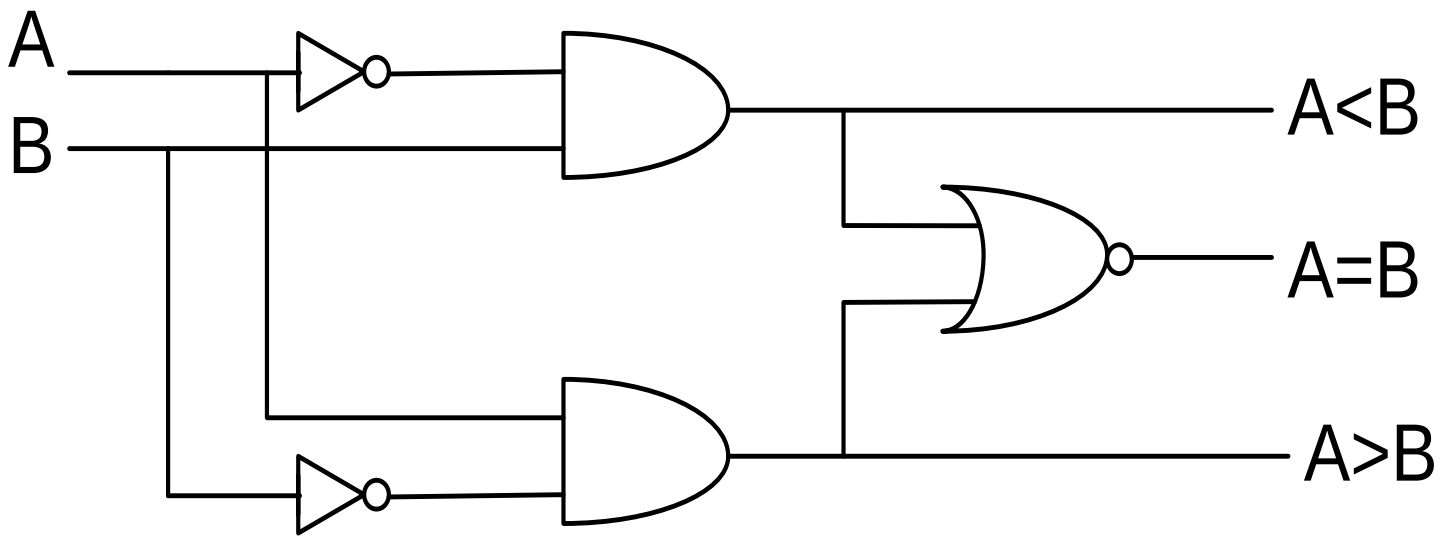

Clock
R/W
Register A
a0
a1
a2
a3
Clock
R/W
Register B
b0
b1
b2
b3
x0
x1
x2
x3
y0
y1
y2
y3
c
s0
s1
s2
s3
Inst.[100] A=A+B
InstClk2
Inst.[100] A=A+B
InstClk1
clock
Latch

Inst. [100] A = A+B


Clock
R/W
Register A
a0 a1 a2 a3
Clock
R/W
Register B
b0 b1 b2 b3
Inst.[000] A=0
Inst.[001] B=0
"0" "0" "0" "0"
1 1 1 1
x0 x1 x2 x3 y0 y1 y2 y3
4Bit Adder
c s0 s1 s2 s3
Inst.[010] A=A+1
InstClk2
Inst.[010] A=A+1
InstClk1
clock Latch
Inst.[011] B=B+1
InstClk2
Inst.[011] B=B+1
InstClk1
Inst.[100] A=A+B
InstClk2
Inst.[100] A=A+B
InstClk1

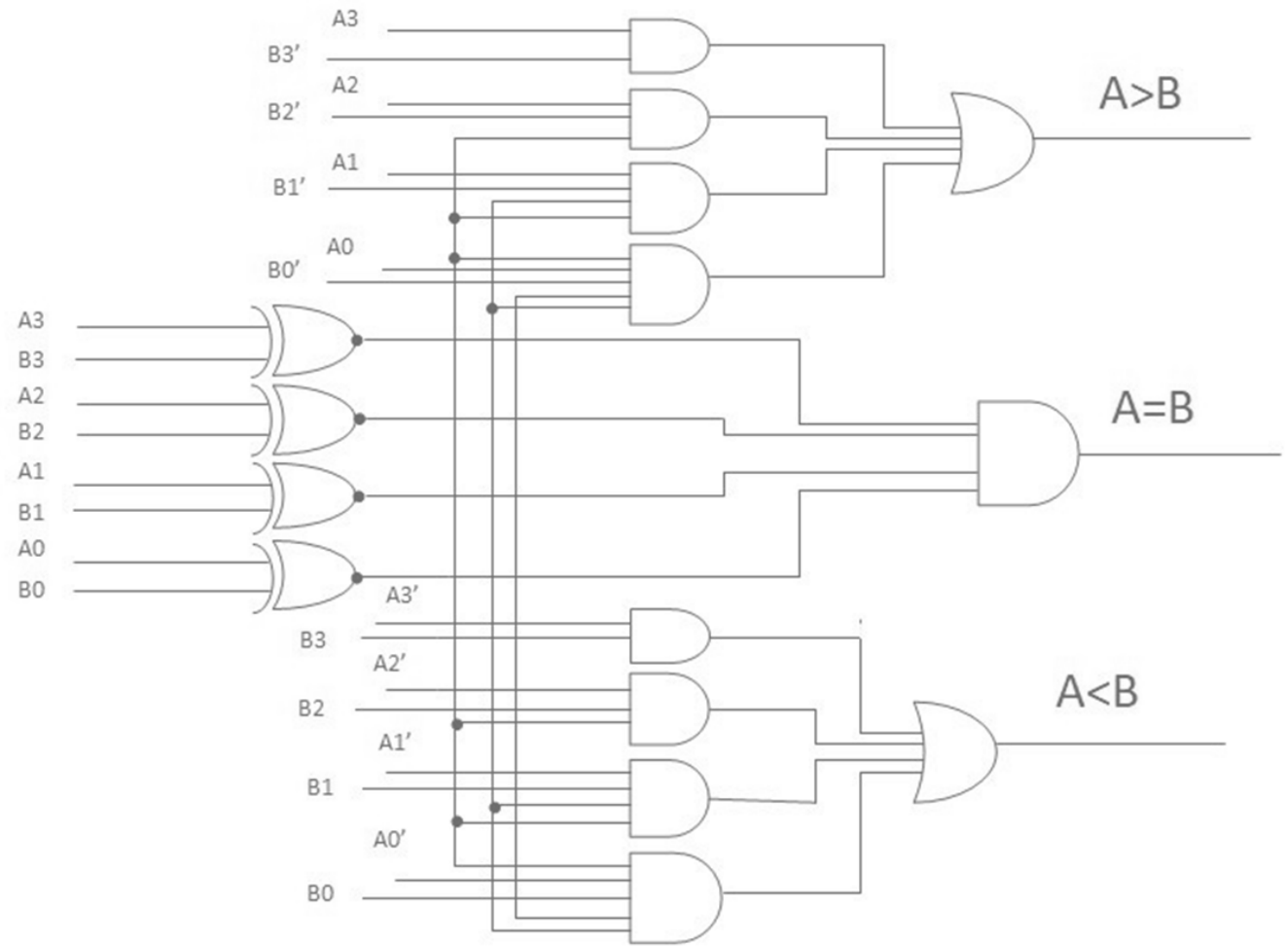
Inst. [101] A<B

- 1 Bit comparator circuit

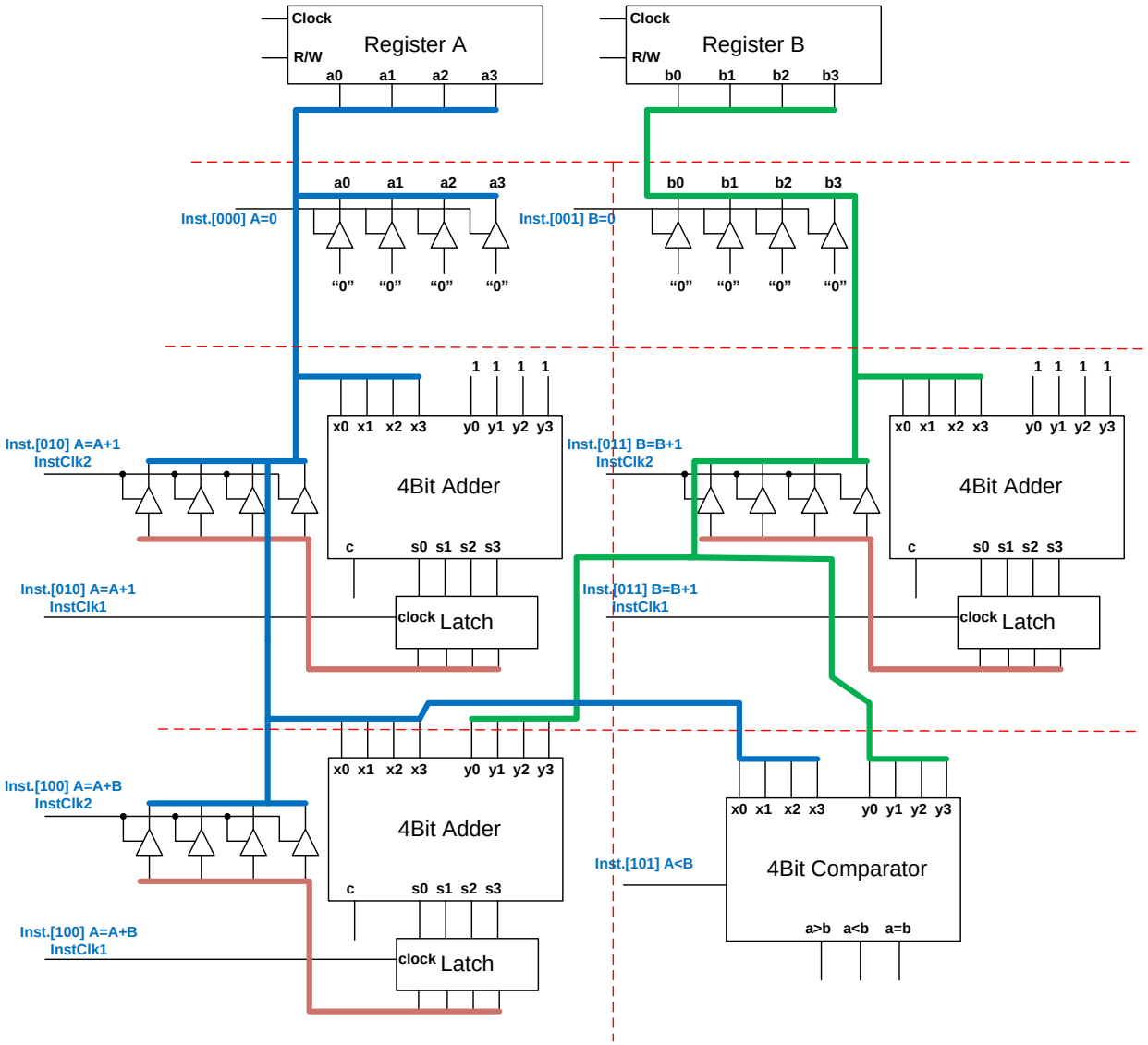


Inst. [101] A<B

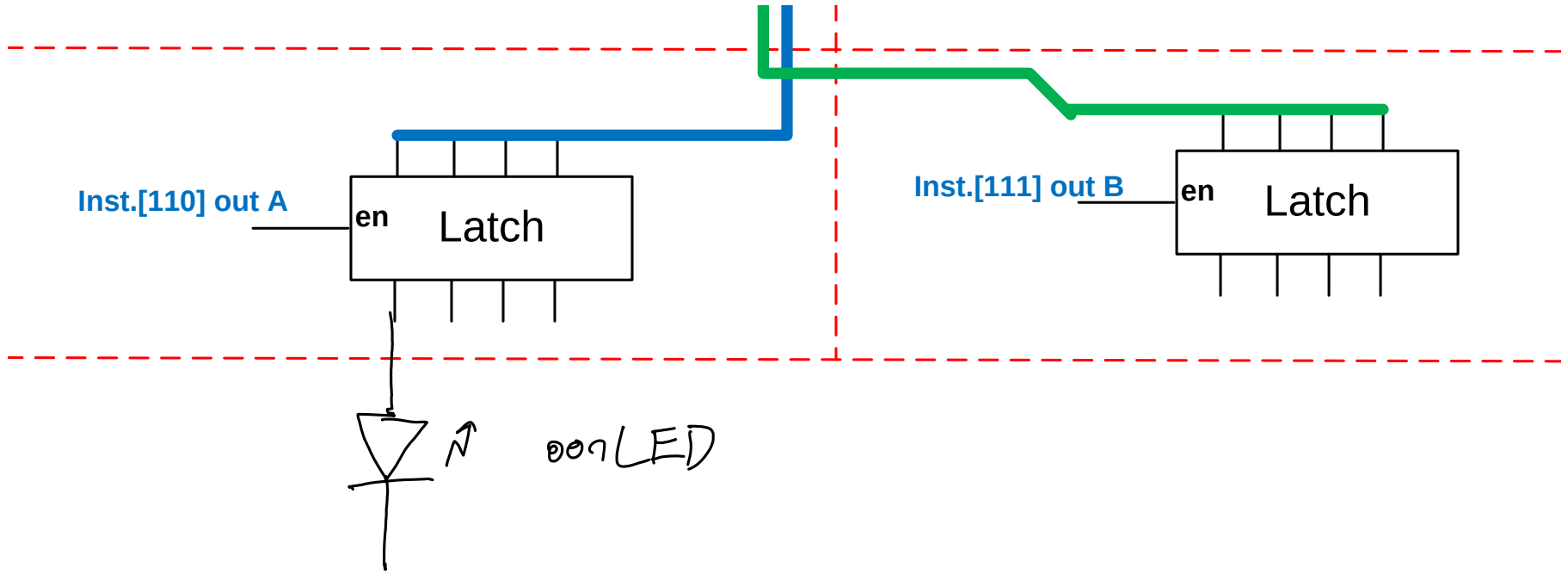
- 4 Bit comparator circuit



Inst. [101] A<B

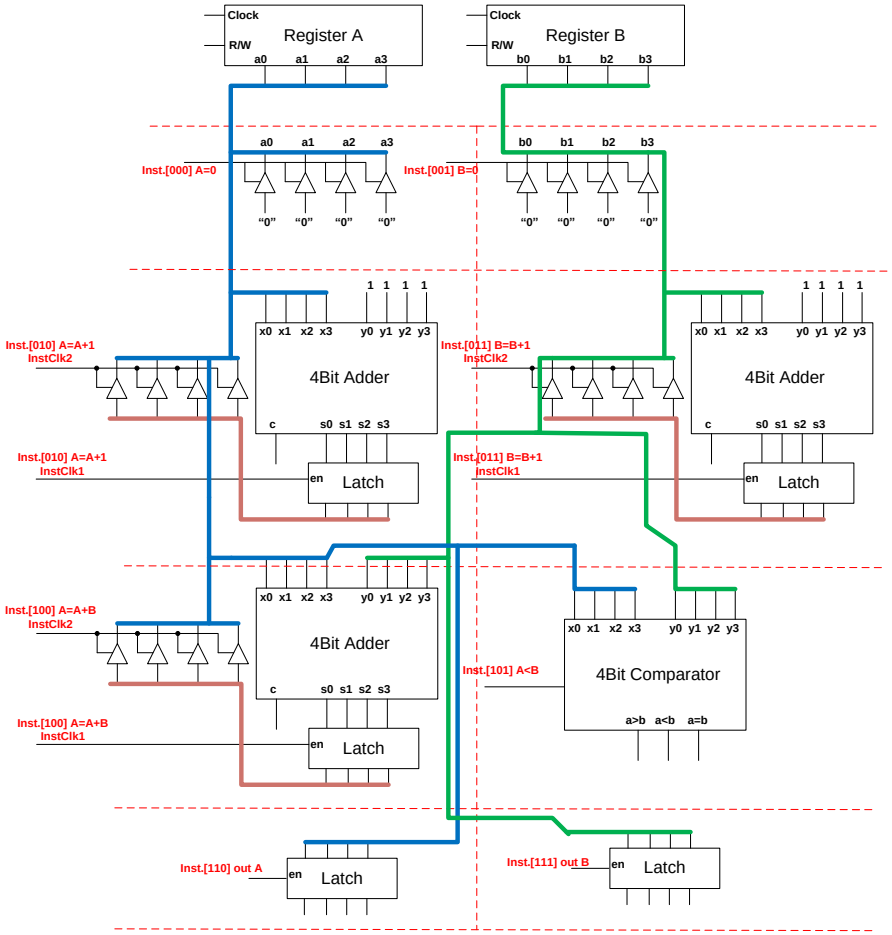

Clock
R/W
Register A
a0 a1 a2 a3
Clock
R/W
Register B
b0 b1 b2 b3
Inst.[000] A=0
Inst.[001] B=0
"0" "0" "0" "0"
1 1 1 1
x0 x1 x2 x3 y0 y1 y2 y3
4Bit Adder
c s0 s1 s2 s3
Inst.[010] A=A+1
InstClk2
Inst.[010] A=A+1
InstClk1
clock Latch
Inst.[011] B=B+1
InstClk2
Inst.[011] B=B+1
InstClk1
Inst.[100] A=A+B
InstClk2
Inst.[100] A=A+B
InstClk1
Inst.[101] A<B
4Bit Comparator
a>b a<b a=b

Inst. [110] Out A
Inst. [111] Out B



Control signals

- 8 Control signals
  - 001
  - 010
  - 011
    - Step1
    - Step2
  - 100
    - Step1
    - Step2
  - 101
    - Step1
    - Step2
  - 110
  - 111



Programmer writes a program

A = 0
B =0
A = A+1
A = A+1
Out A
B = B+1
A = A+B
Out A
Out B

Inst. ID	Command
0 (000)	A = 0
1 (001)	B = 0
2 (010)	A = A +1
3 (011)	B = B + 1
4 (100)	A = A+B
5 (101)	A <B {Compare Bit (CB) = 1 and otherwise CB = 0}
6 (110)	Out A
7 (111)	Out B

Compile

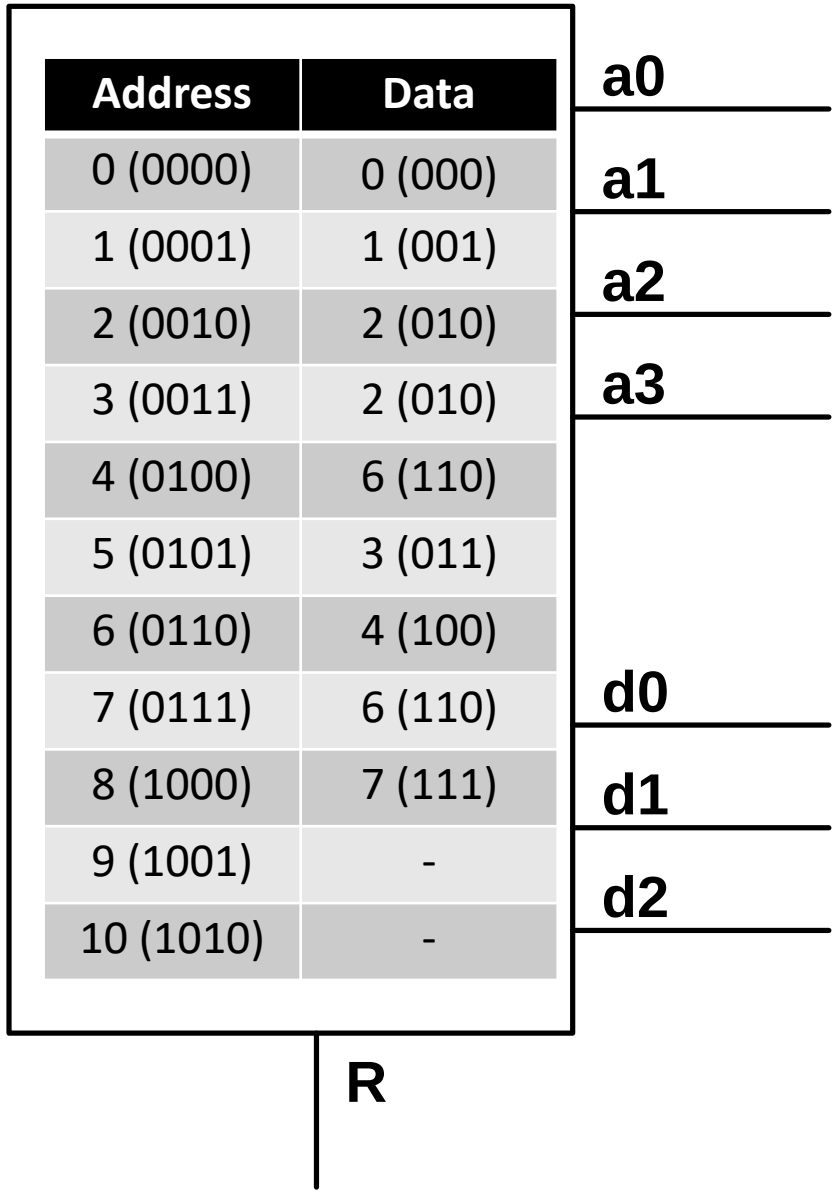
Machine Language

- 0 1 2 2 6 3 4 6 7 (Hex)

- 000
- 001
- 010
- 010
- 110
- 011
- 100
- 110
- 111

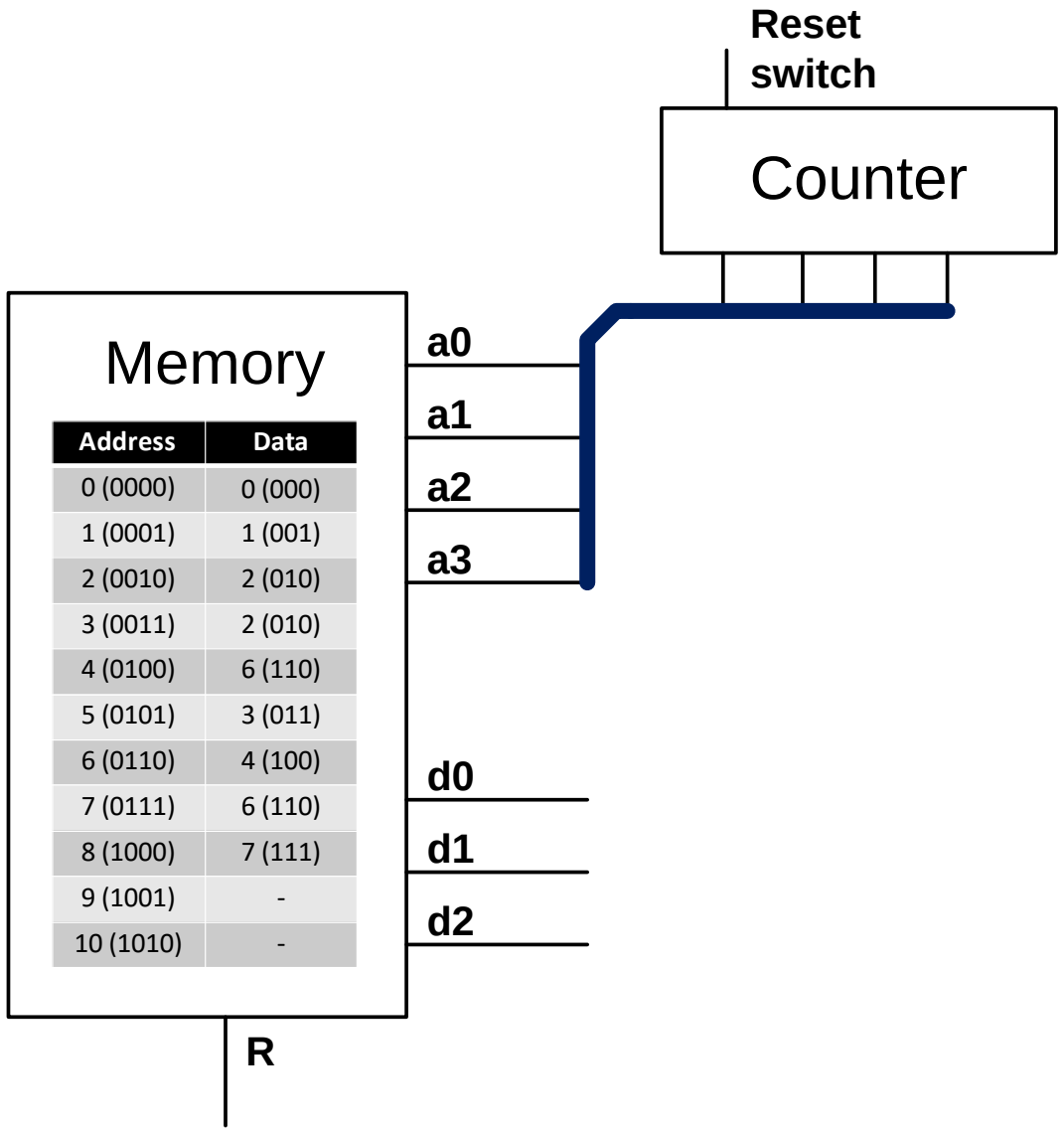
RUN

Keeping program to memory



- Memory
  - Address bus (a0-a3)
  - Data bus (d0-d2)
  - Read signal

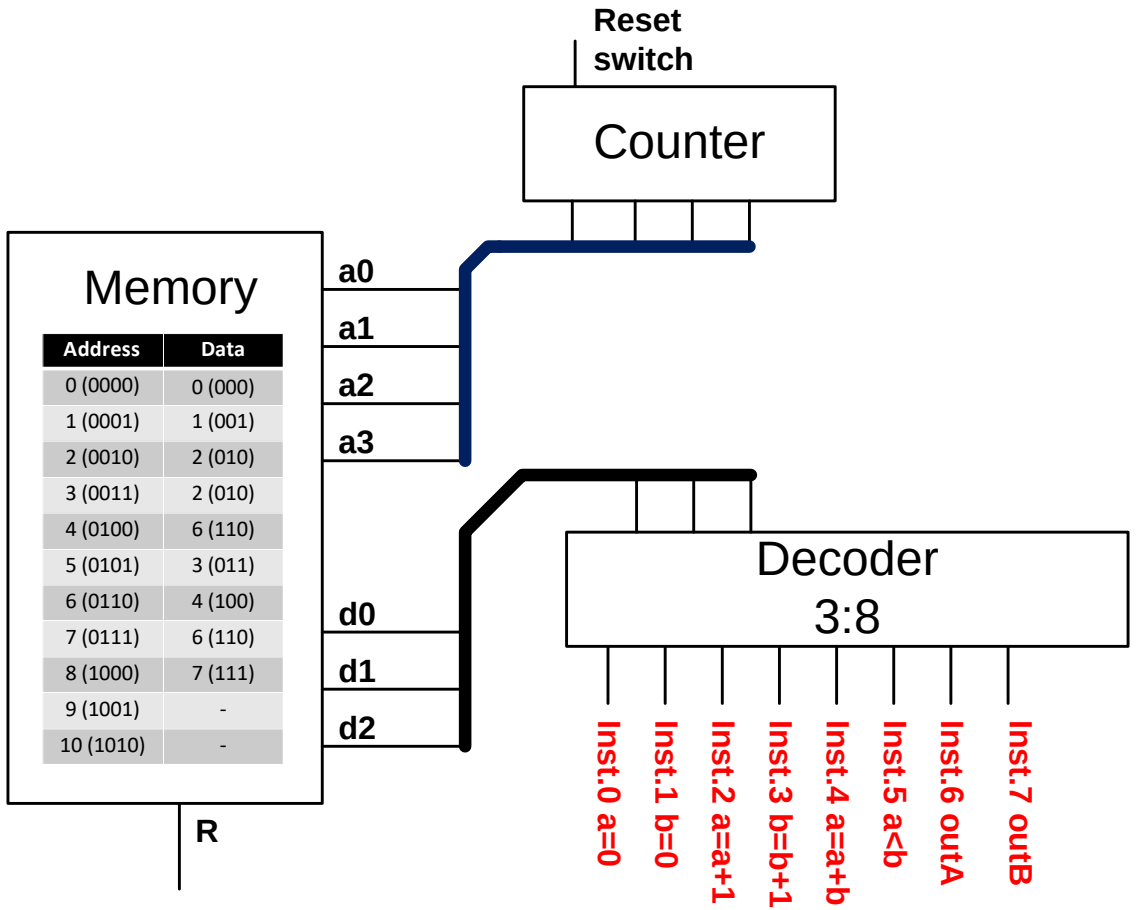
Read memory with a counter



Decode instruction

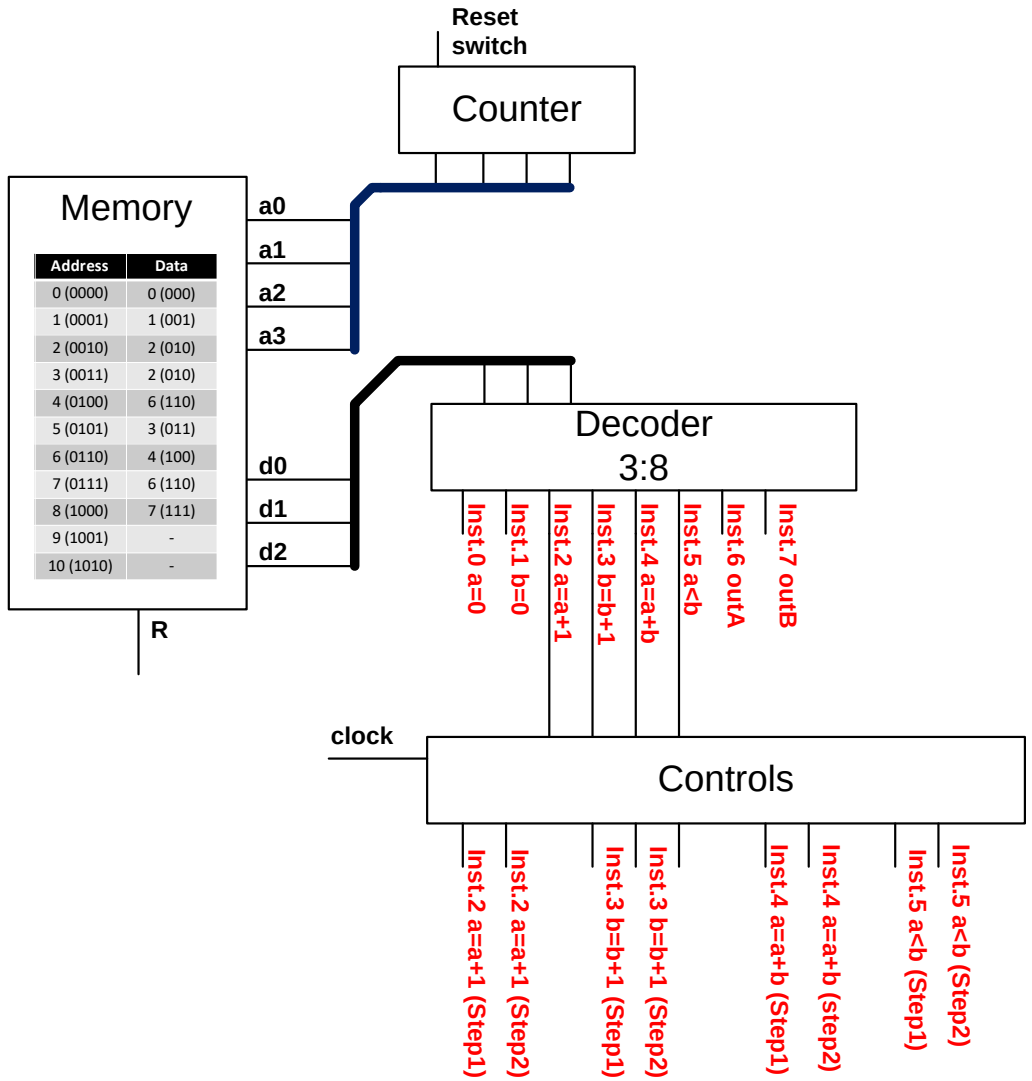
- Decoding instruction uses a decoder circuit
  - 3 to 8

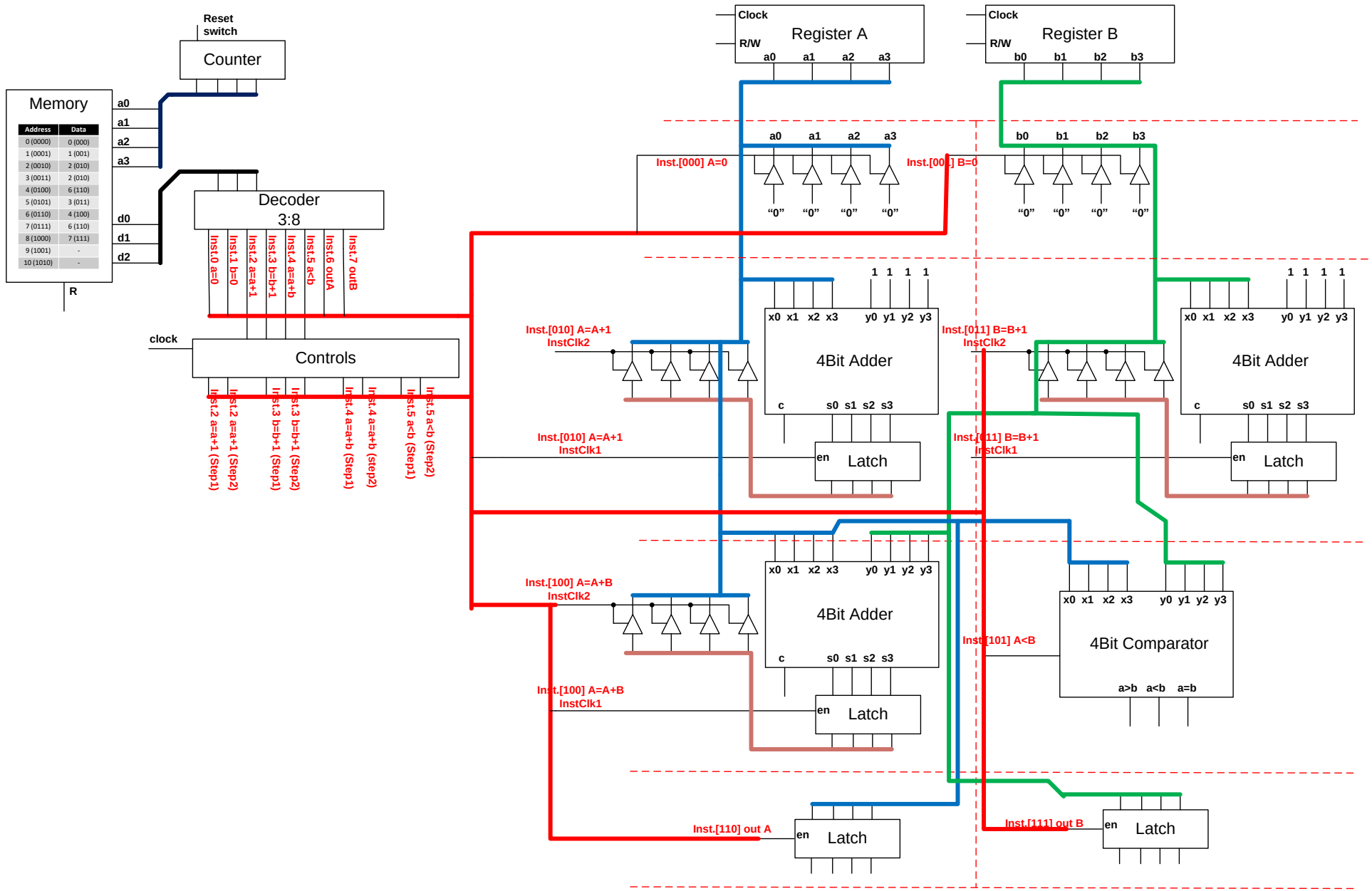
Inst. ID	Command
0 (000)	A = 0
1 (001)	B = 0
2 (010)	A = A +1
3 (011)	B = B + 1
4 (100)	A = A+B
5 (101)	A <B {Compare Bit (CB) = 1 and otherwise CB = 0}
6 (110)	Out A
7 (111)	Out B



Instructions use two clocks

- Control circuit
  - Counter 2 clock
  - Control each step in each instruction.




Reset
switch
Counter
Memory
Address
Data
0 (0000)
0 (000)
1 (0001)
1 (001)
2 (0010)
2 (010)
3 (0011)
2 (010)
4 (0100)
6 (110)
5 (0101)
3 (011)
6 (0110)
4 (100)
7 (0111)
6 (110)
8 (1000)
7 (111)
9 (1001)
-
10 (1010)
-
a0
a1
a2
a3
R
Decoder
3:8
d0
d1
d2
Inst.0 a=0
Inst.1 b=0
Inst.2 a=a+1
Inst.3 b=b+1
Inst.4 a=a+b
Inst.5 a<b
Inst.6 outA
Inst.7 outB
clock
Controls
Inst.2 a=a+1 (Step1)
Inst.2 a=a+1 (Step2)
Inst.3 b=b+1 (Step1)
Inst.3 b=b+1 (Step2)
Inst.4 a=a+b (Step1)
Inst.4 a=a+b (step2)
Inst.5 a<b (Step1)
Inst.5 a<b (Step2)
Clock
R/W
Register A
a0 a1 a2 a3
Clock
R/W
Register B
b0 b1 b2 b3
Inst.[000] A=0
Inst.[000] B=0
"0" "0" "0" "0"
"0" "0" "0" "0"
Inst.[010] A=A+1
InstClk2
Inst.[011] B=B+1
InstClk2
1 1 1 1
x0 x1 x2 x3
y0 y1 y2 y3
4Bit Adder
c
s0 s1 s2 s3
Inst.[010] A=A+1
InstClk1
Inst.[011] B=B+1
InstClk1
en
Latch
Inst.[100] A=A+B
InstClk2
Inst.[101] A<B
4Bit Comparator
a>b a<b a=b
Inst.[100] A=A+B
InstClk1
Inst.[110] out A
Inst.[111] out B