MULTI-CYCLE PROCESSOR

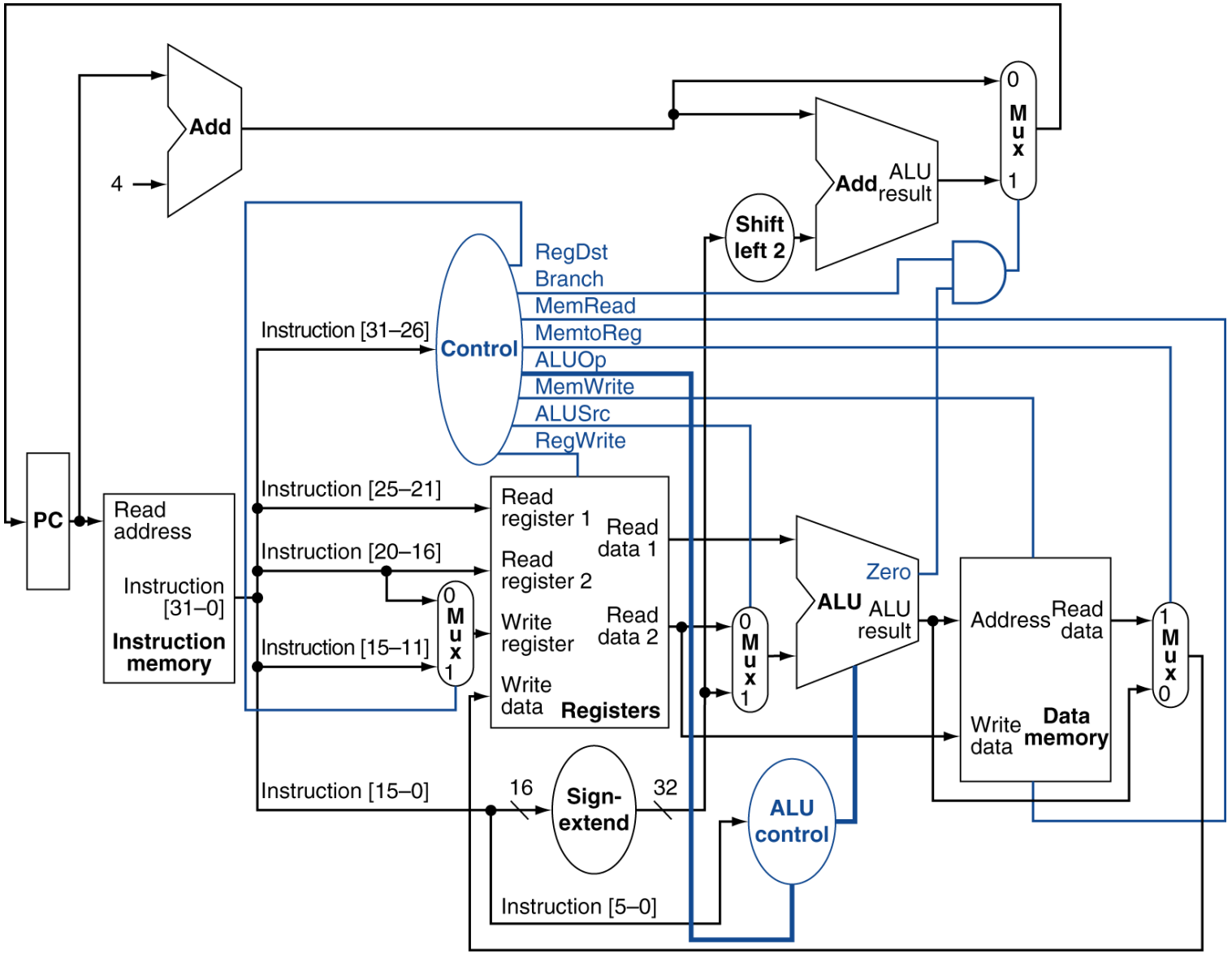
Lecturer: Hui Annie Guo
h.guo@unsw.edu.au
K17-501F

Lecture overview

- Topics
  - Motivation
  - Multi-cycle processor
    - Datapath
    - Control

- Suggested reading
  - H&P B.11
  - Reference
    - H&P textbook (3rd edition) on MC processor
      - Page 318-340

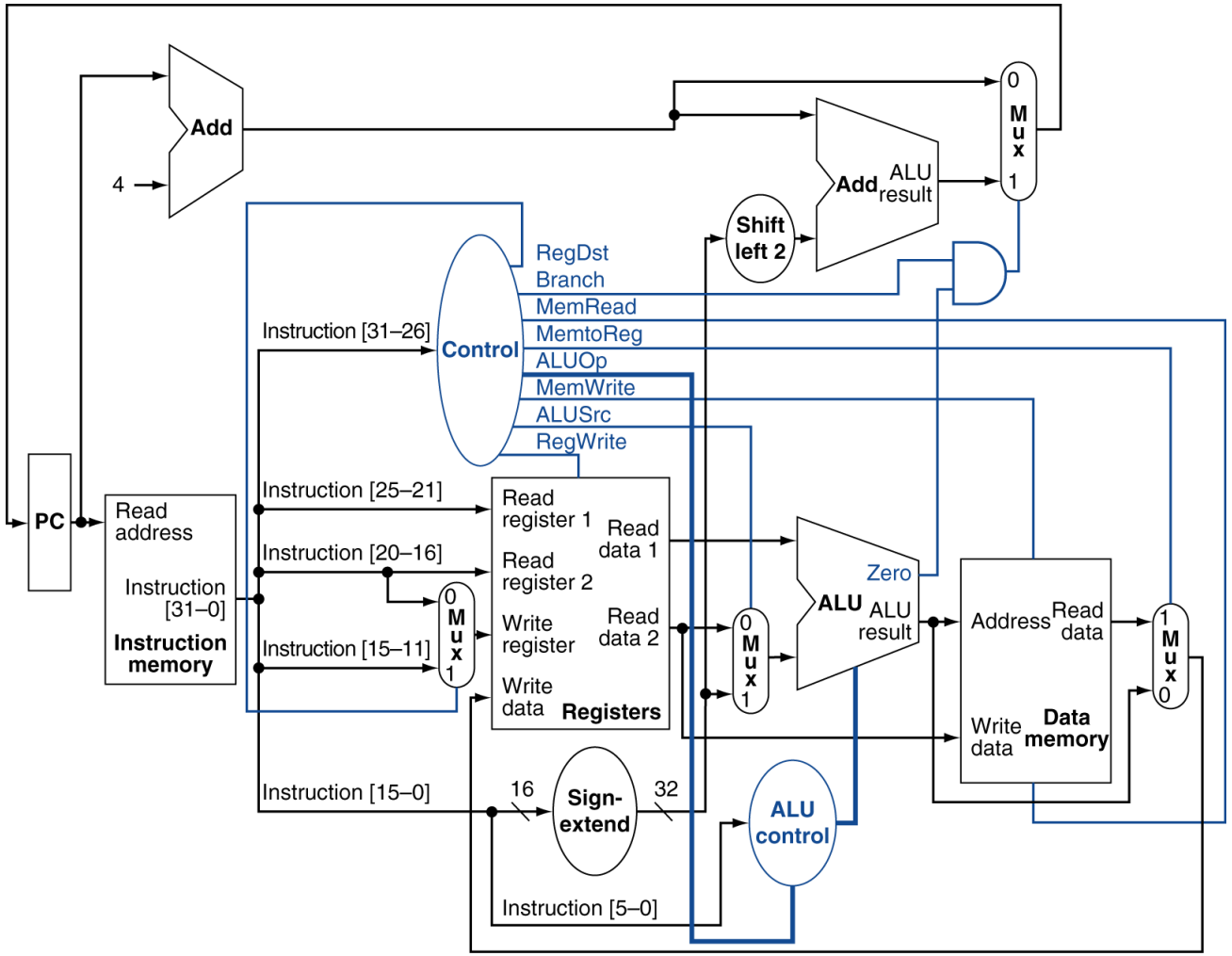
Recall: Single cycle processor



Latency of ADDU and SUBU

PC	IM	RF	MUX	ALU	MUX	setup

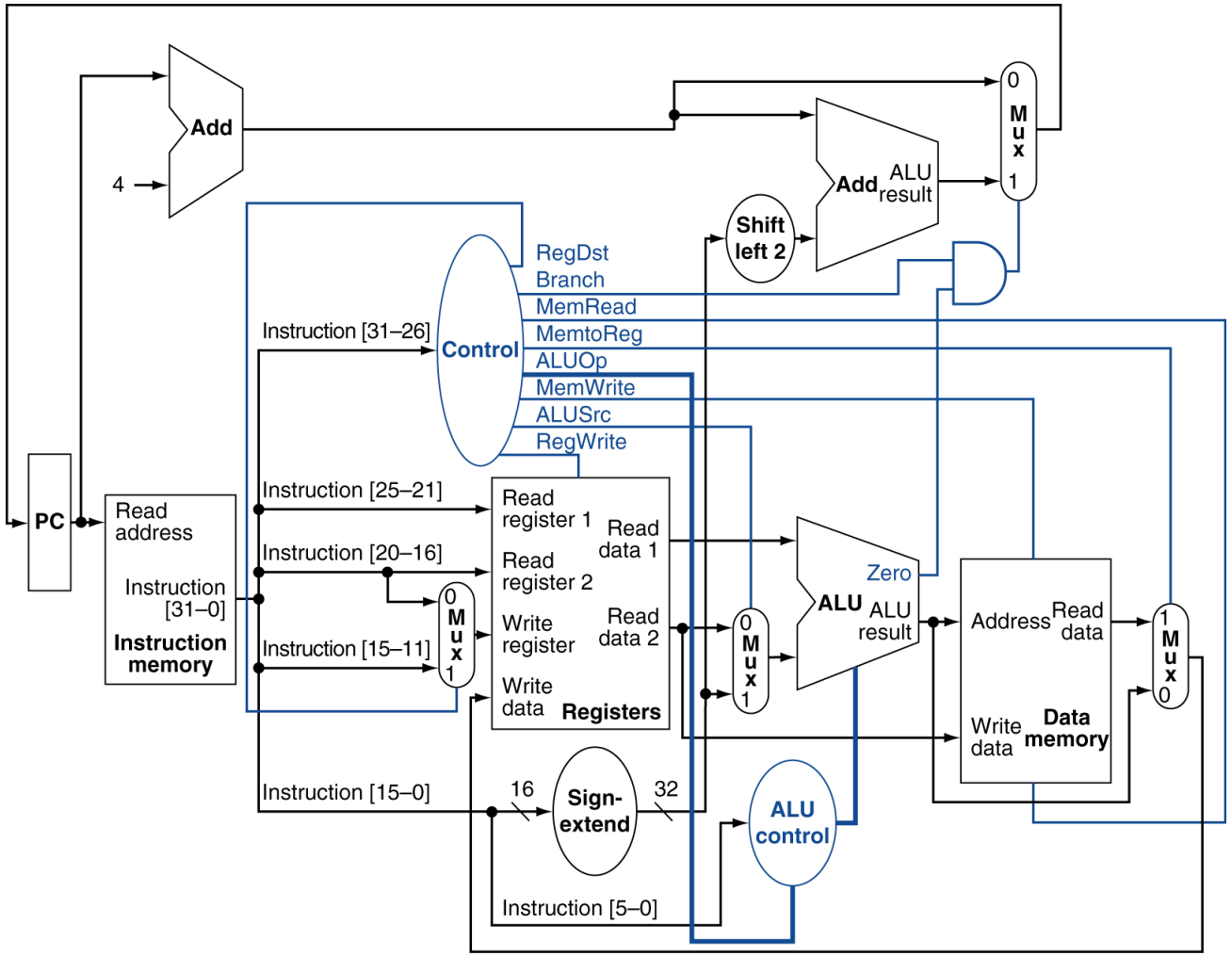
Assume control signals are ready when data flow to a component



Latency of ORI

PC	IM	RF	MUX	ALU	MUX	setup

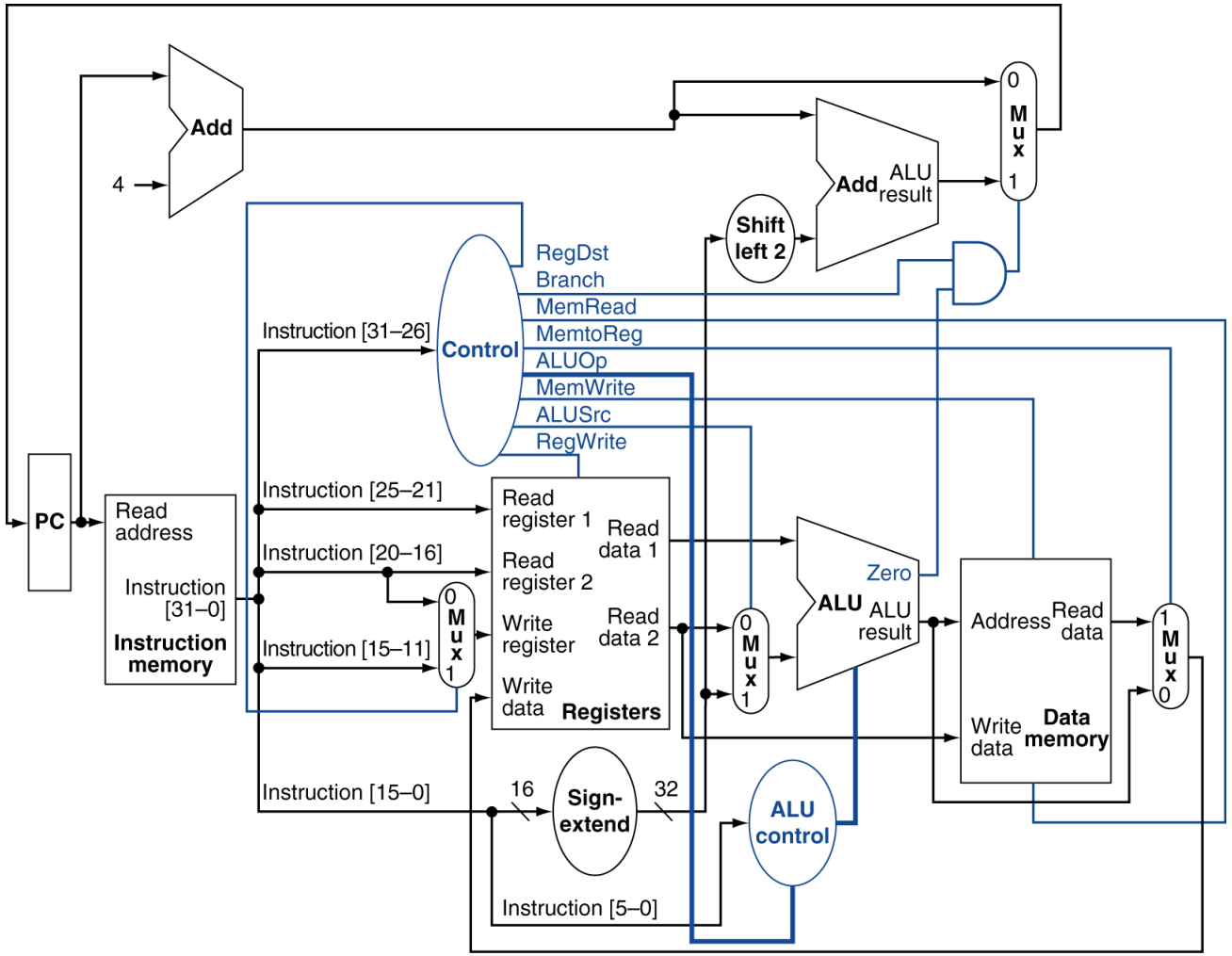
Assume control signals are ready when data flow to a component



Latency of LW

PC	IM	RF	MUX	ALU	DM	MUX	setup

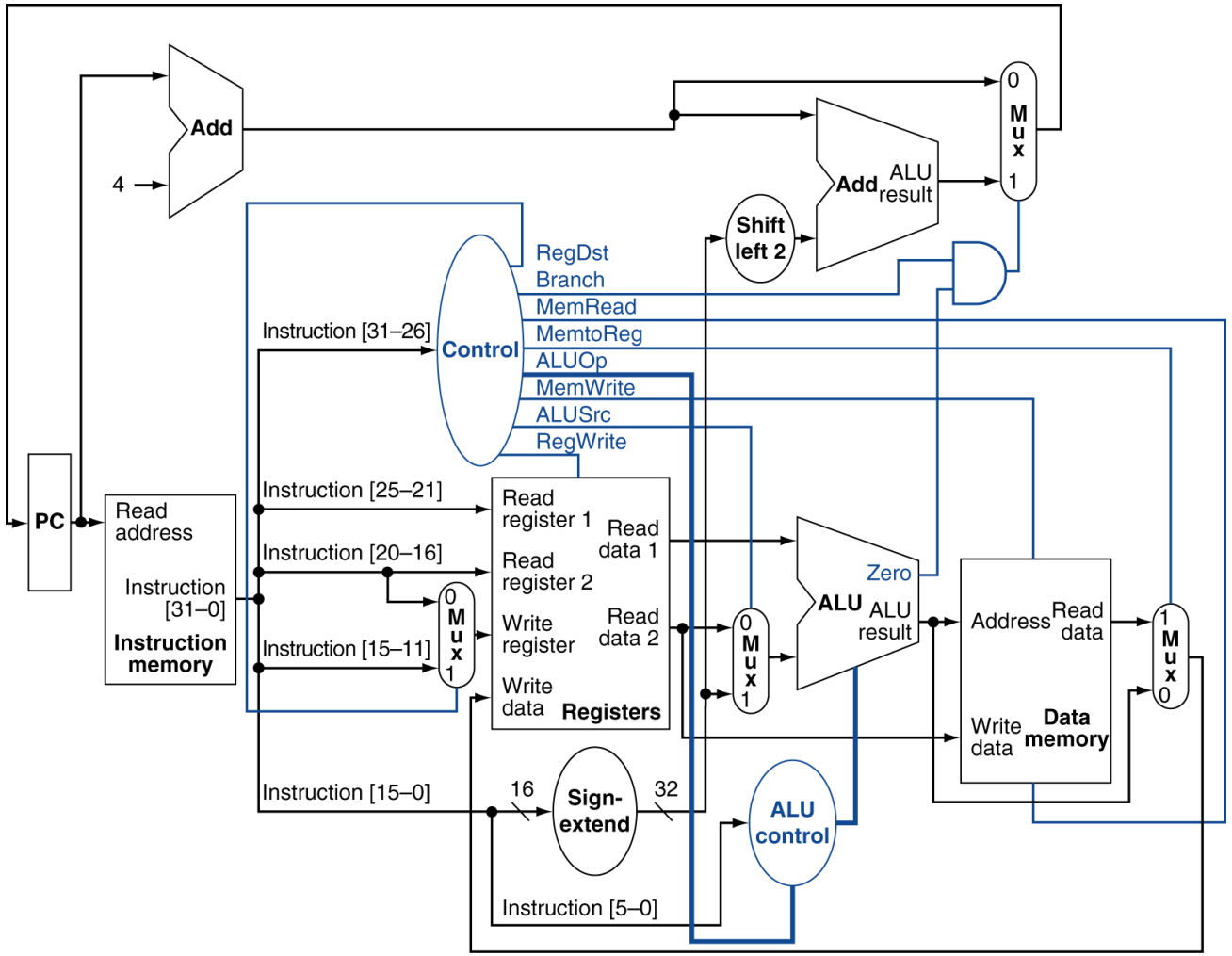
Assume control signals are ready when data flow to a component



Latency of SW

PC	IM	RF	MUX	ALU	DM	MUX	setup

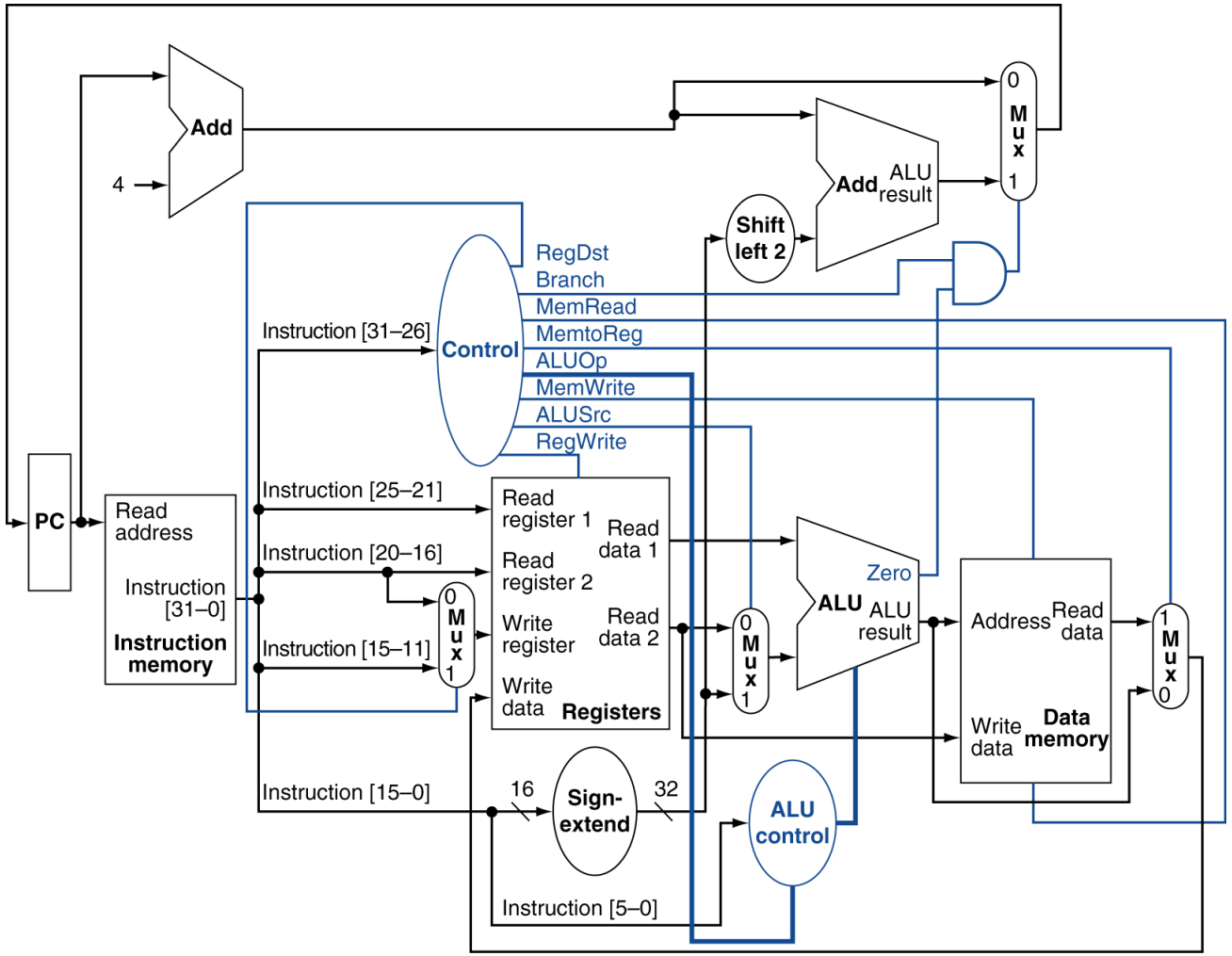
Assume control signals are ready when data flow to a component



Latency of BEQ

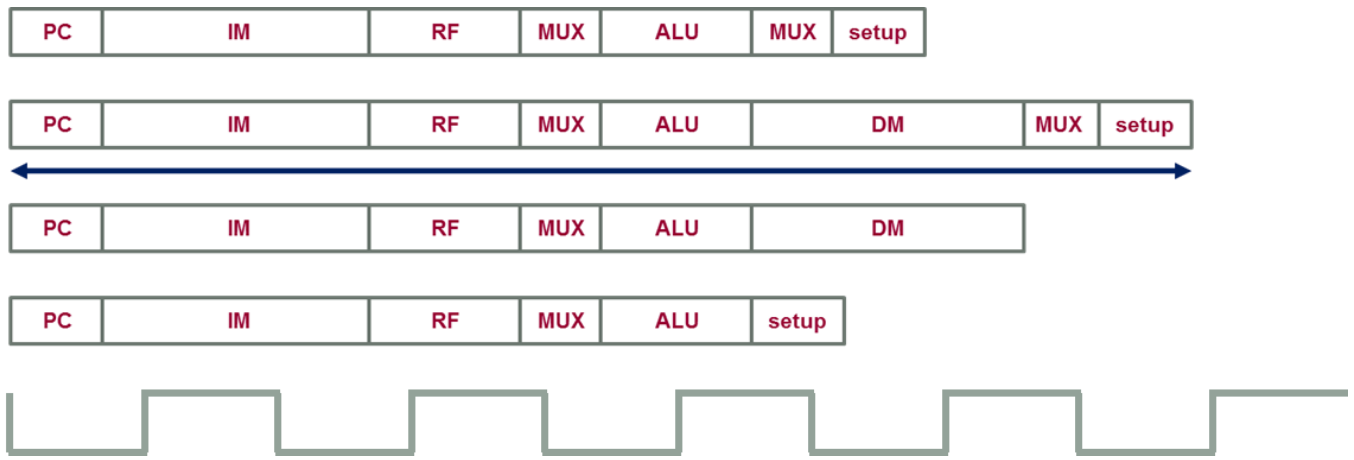
PC	IM	RF	MUX	ALU	setup

Assume control signals are ready when data flow to a component



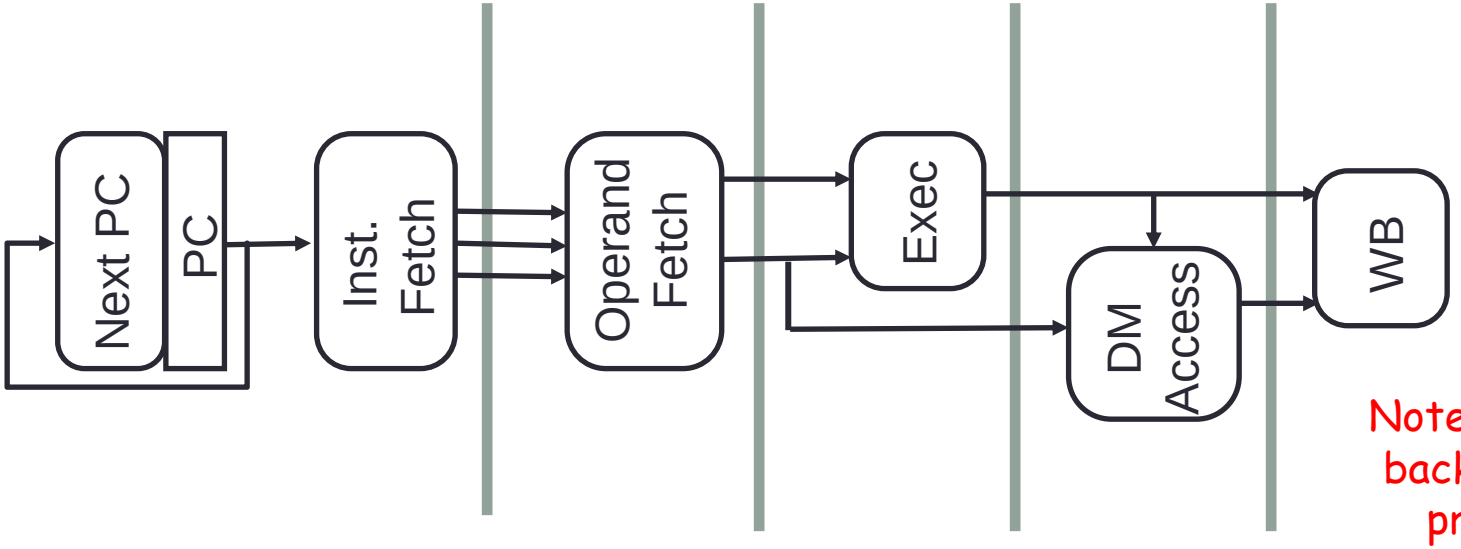
Multi-cycle datapath

- With reduced clock cycle time
- An instruction is completed in multiple clock cycles
  - each instruction uses as a small number of clock cycles as possible



How to implement multi-cycle processor?

- **Partition the single-cycle datapath**
  - **Try to balance time spent in each section**
    - **The clock cycle time is determined by the longest section delay**
- **Insert registers between sections**
  - **Based on the operation of each section**

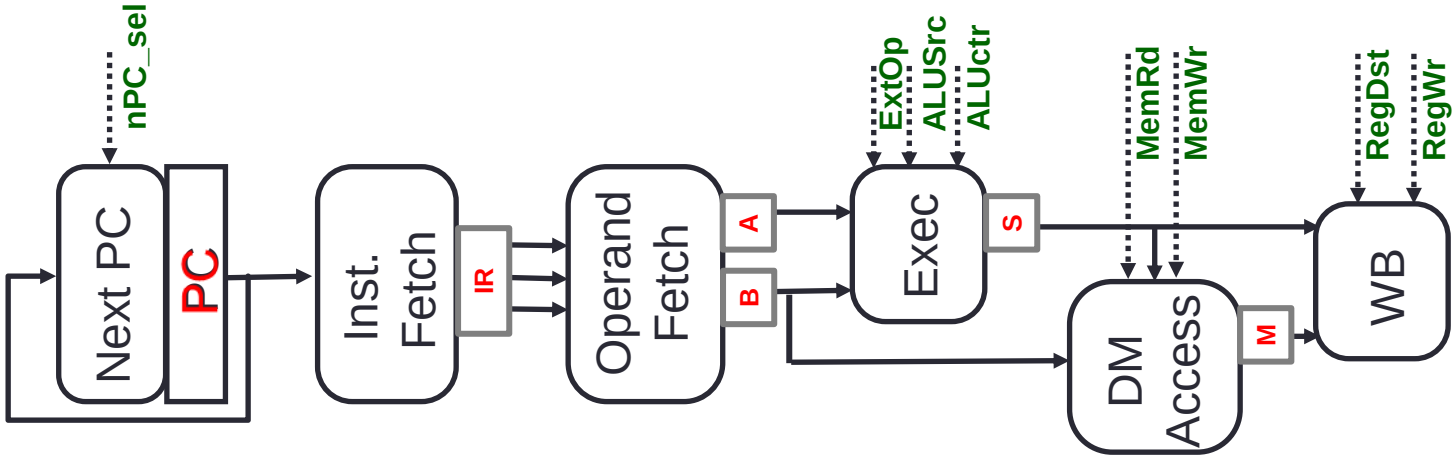


Note: Data flow backwards not presented

Operations of each section

- **Next address logic**
  PC ← branch ? PC + 4+ offset : PC + 4
- **Instruction fetch**
  IR ← Mem[PC]
- **Operand fetch**
  A ← R[rs]
  B ← R[rt]
- **ALU exec**
  S ← A op B

- **Memory read**
  M ← Mem[Addr]
- **Memory write**
  Mem[addr] ← B
- **Write back**
  R[rd|rt] ← R|M

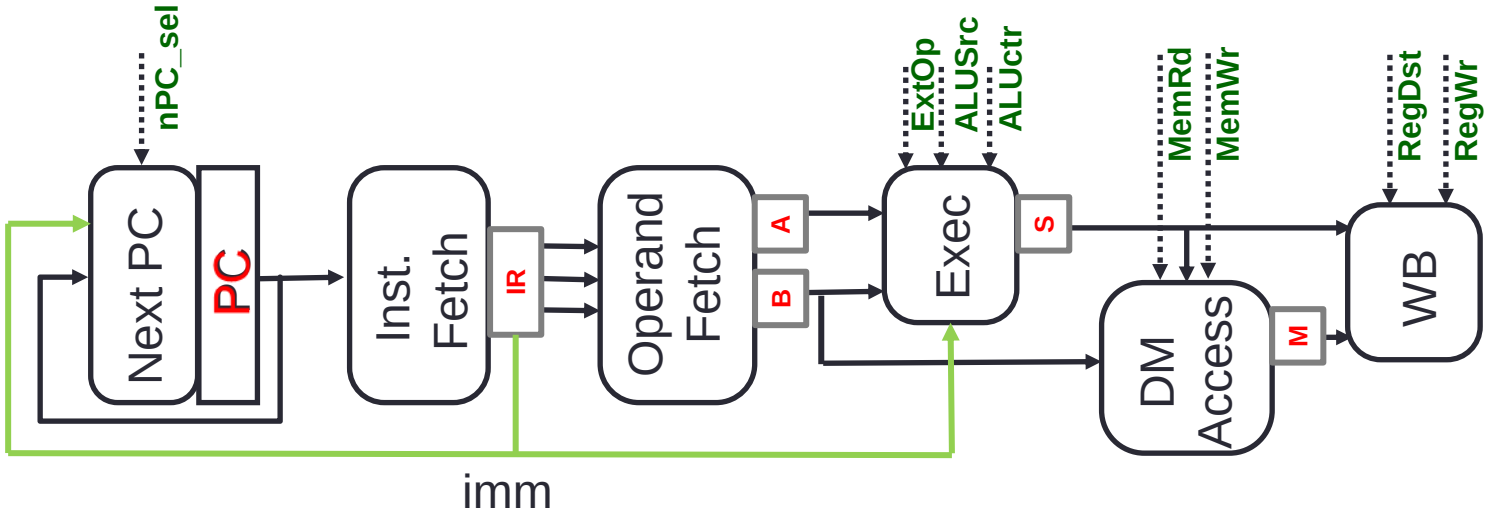


Operations of R-type instructions

Can be done in 4 clocks:

1st clock:	IR ← IM[PC]
2nd clock:	A ← R[rs]; B ← R[rt]
3rd clock:	S ← A op B
4th clock:	R[rd] ← S; PC ← PC+4

Example:
ADDU $1, $2, $3
$1 ← $2+$3, PC ← PC+4

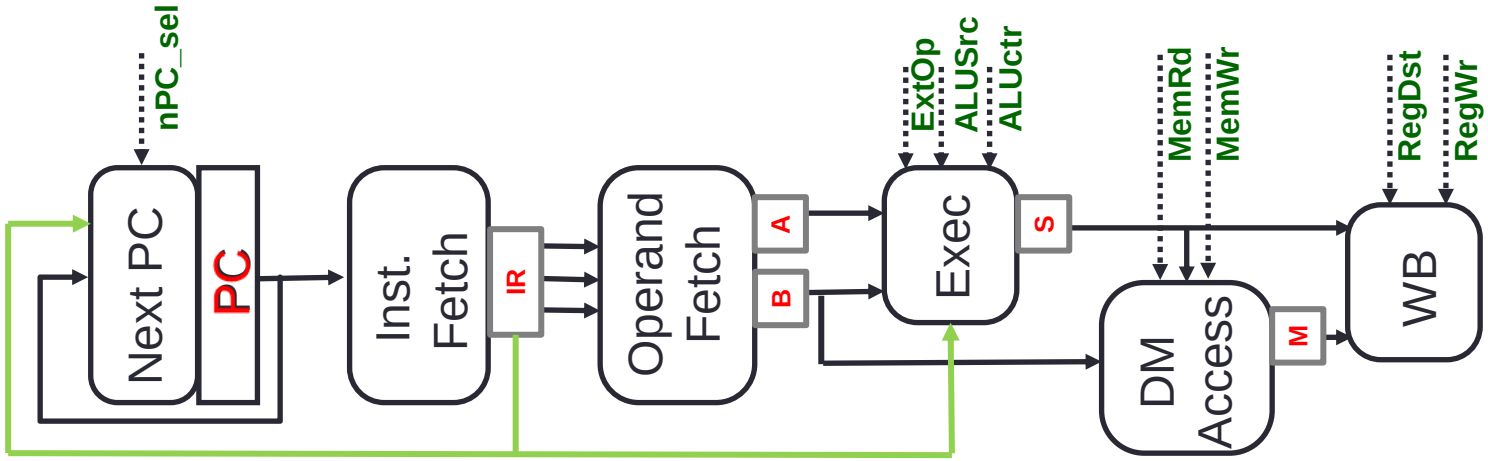


Operations of I-type logic instructions

Can be done in 4 clocks:

Example:
Ori $1, $2, imm
$1 ← $2 or Zero_ext(imm), PC ← PC+4

1st clock:	IR ← IM[PC]
2nd clock:	A ← R[rs]
3rd clock:	S ← A or Zero_ext(imm)
4th clock:	R[rt] ← S; PC ← PC+4

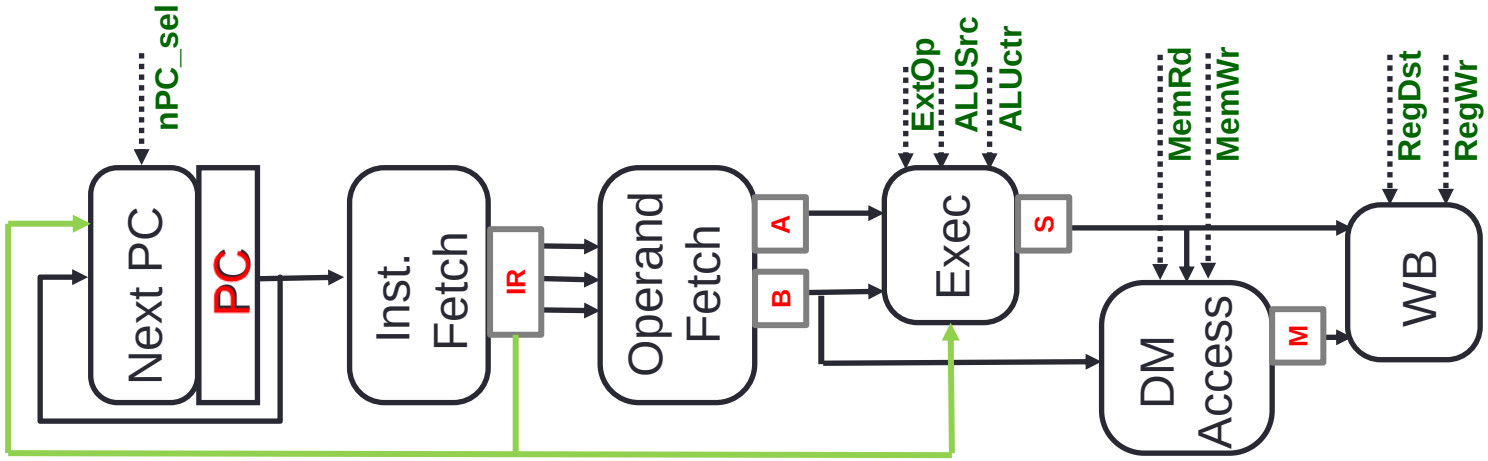


Operations of LW instruction

Example:
LW $1, $2, imm
$1 ← DM[$2+Sign_ext(imm), PC ← PC+4

Can be done in 5 clocks:

1st clock:	IR ← IM[PC]
2nd clock:	A ← R[rs]; B ← R[rt]
3rd clock:	S ← A + Sign_ext(imm)
4th clock:	M ← DM[S]
5th clock:	R[rt] ← S; PC ← PC+4

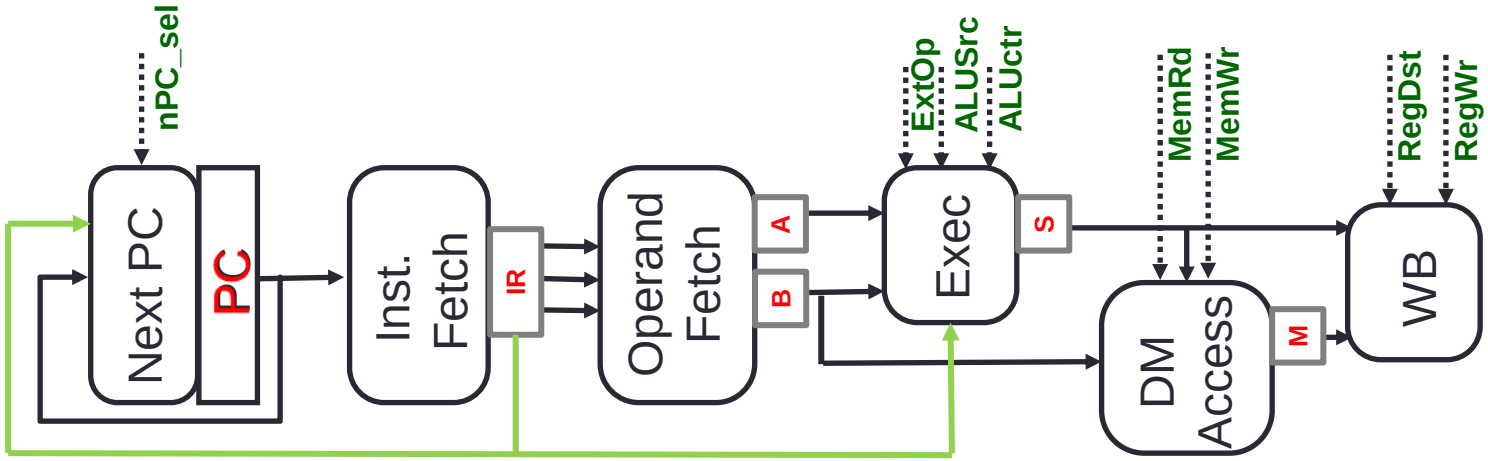


Operations of SW instruction

Can be done in 4 clocks:

Example:
SW $1, $2, imm
DM[$2+Sign_ext(imm)]← $1, PC ← PC+4

1st clock:	IR ← IM[PC]
2nd clock:	A ← R[rs], B ← R([rt]
3rd clock:	S ← A + Sign_ext(imm)
4th clock:	DM[S] ← B

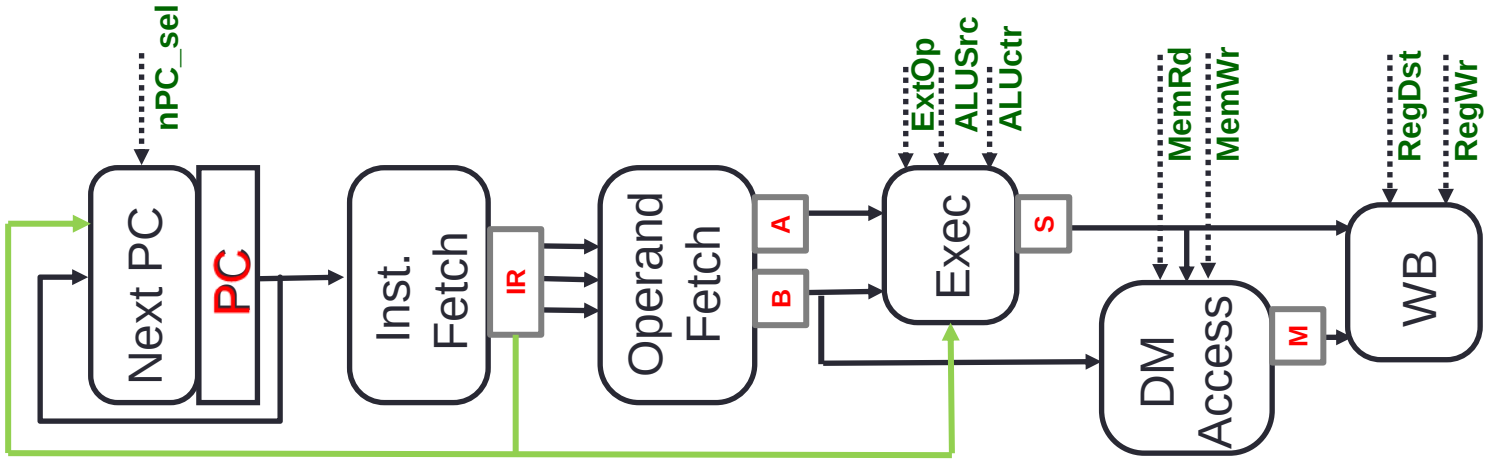


Operations of BEQ instruction

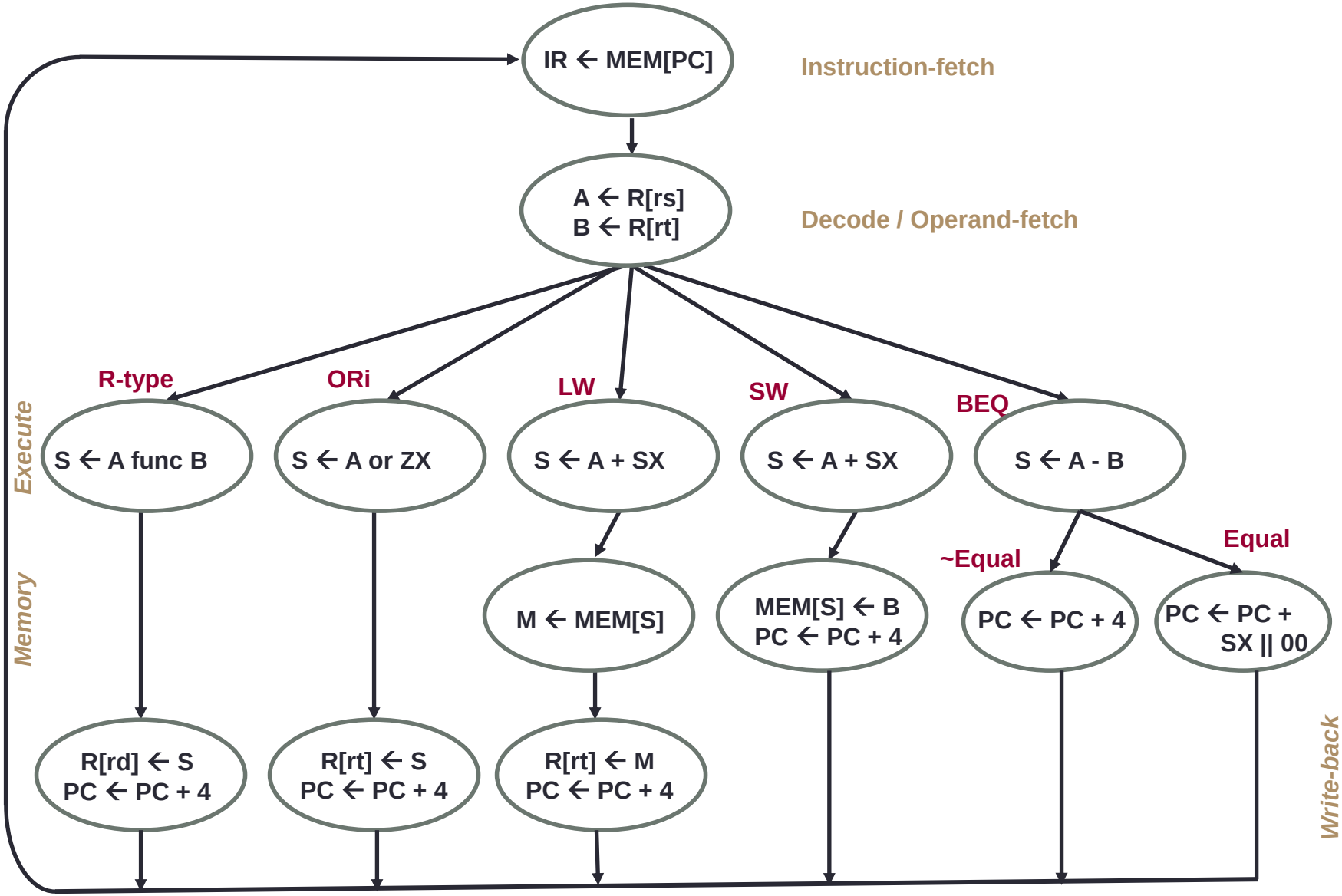
Can be done in 4 clocks:

Example:
BEQ $1, $2, imm
$1==$2, PC ← PC+4: PC+4+Sign_ext(imm)||00

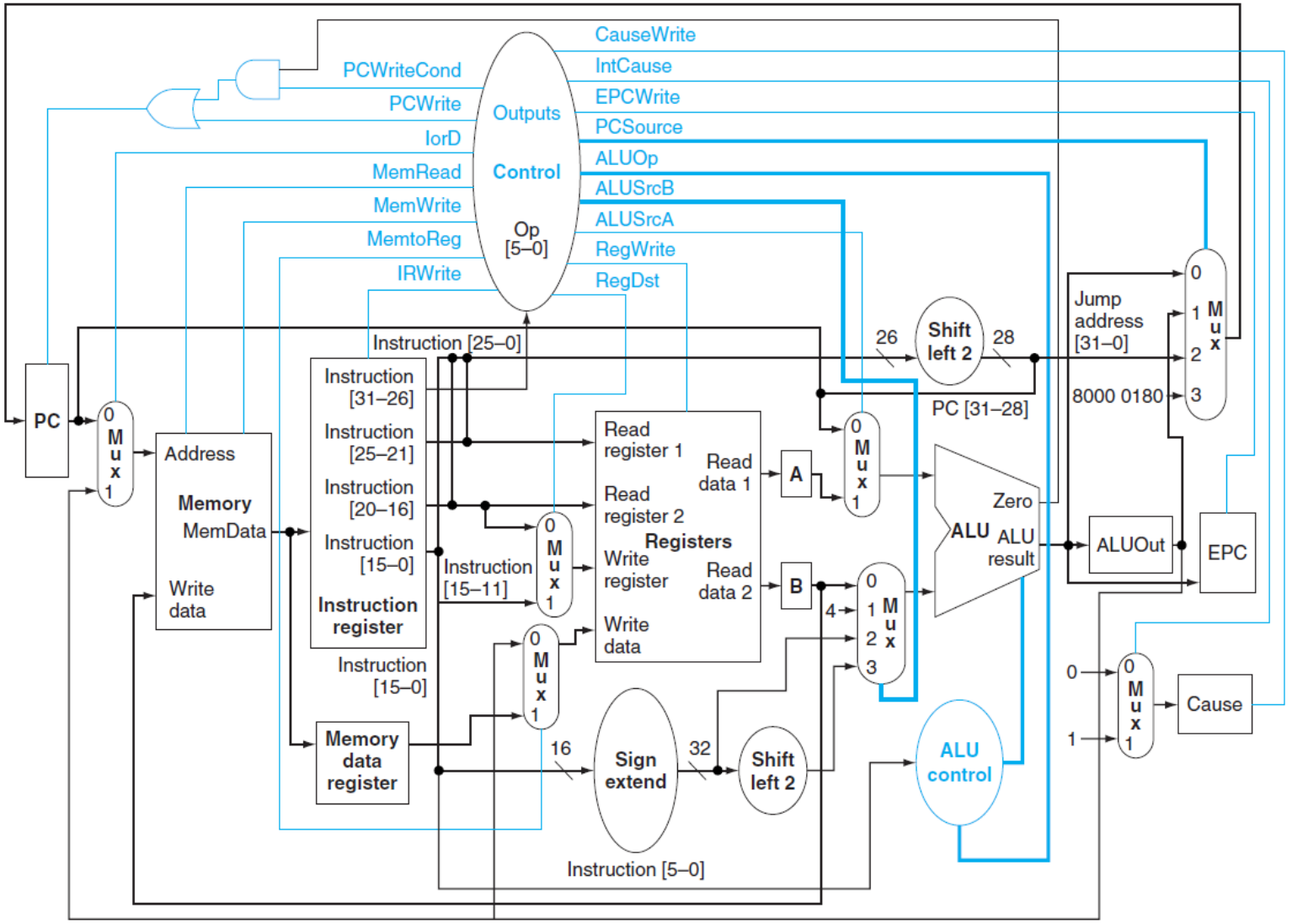
1st clock:	IR ← IM[PC]
2nd clock:	A ← R[rs], B ← R([rt]
3rd clock:	S ← A - B
4th clock:	PC ← (PC+4 or PC+4+Sign_ext(imm)||00)



FSM for control



Example of multi-cycle processor

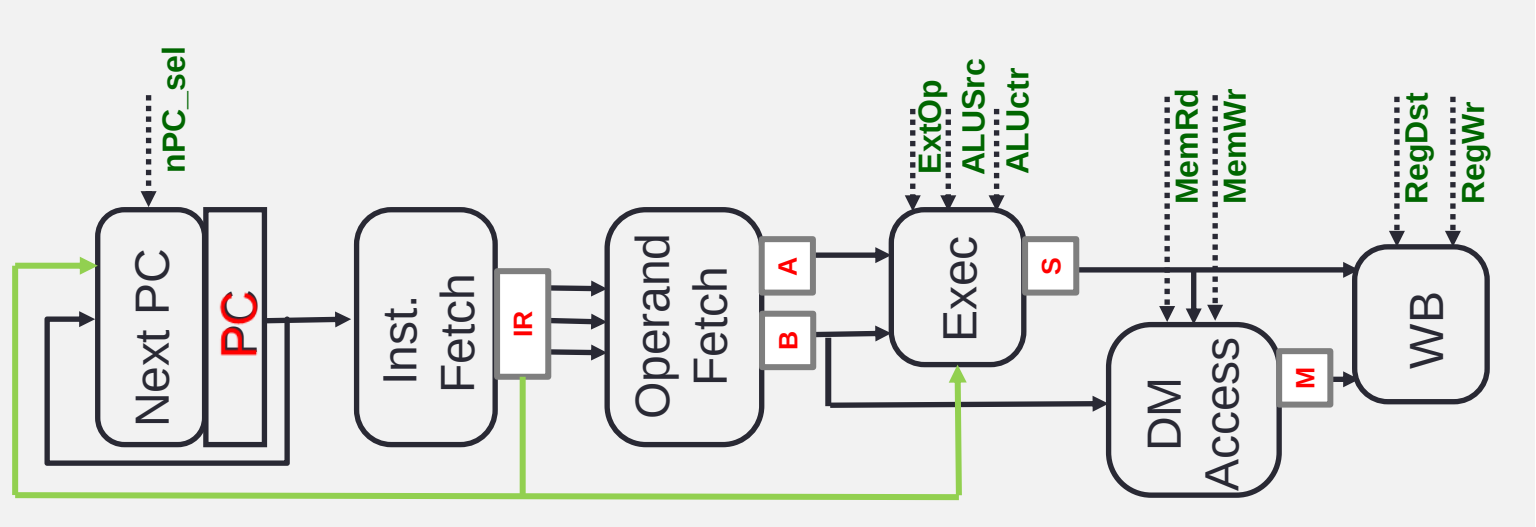


In-class exercise (1)

- **Investigate how resource sharing is applied in the multi-cycle processor design shown in the previous slide.**

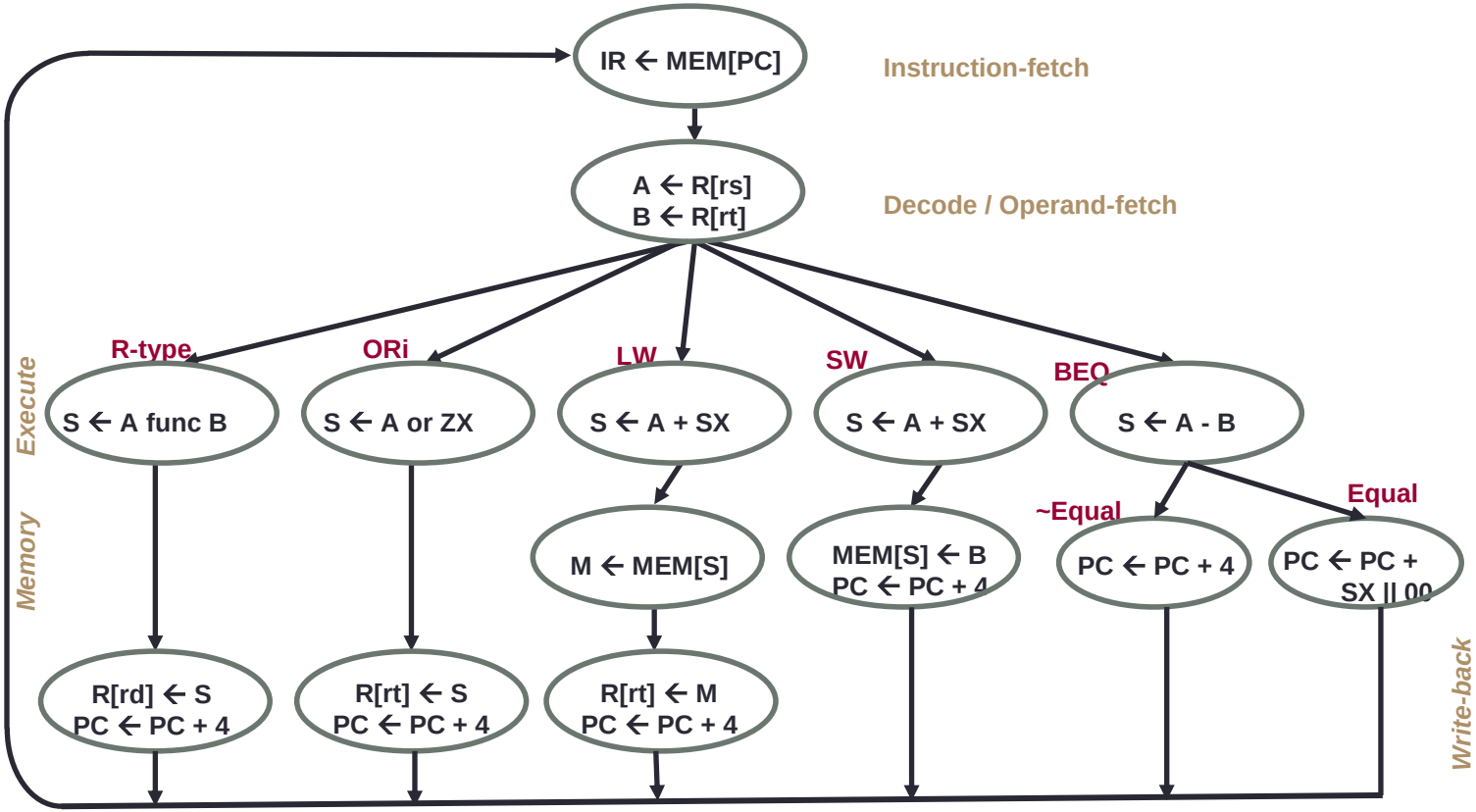
In-class exercise (2)

- **Based on the logic structure of the multi-cycle processor we discussed (as shown below), identify the datapath components in the example MC processor for the following operations:**
  - **(a) instruction fetch**
  - **(b) memory access**



In-class exercise (3)

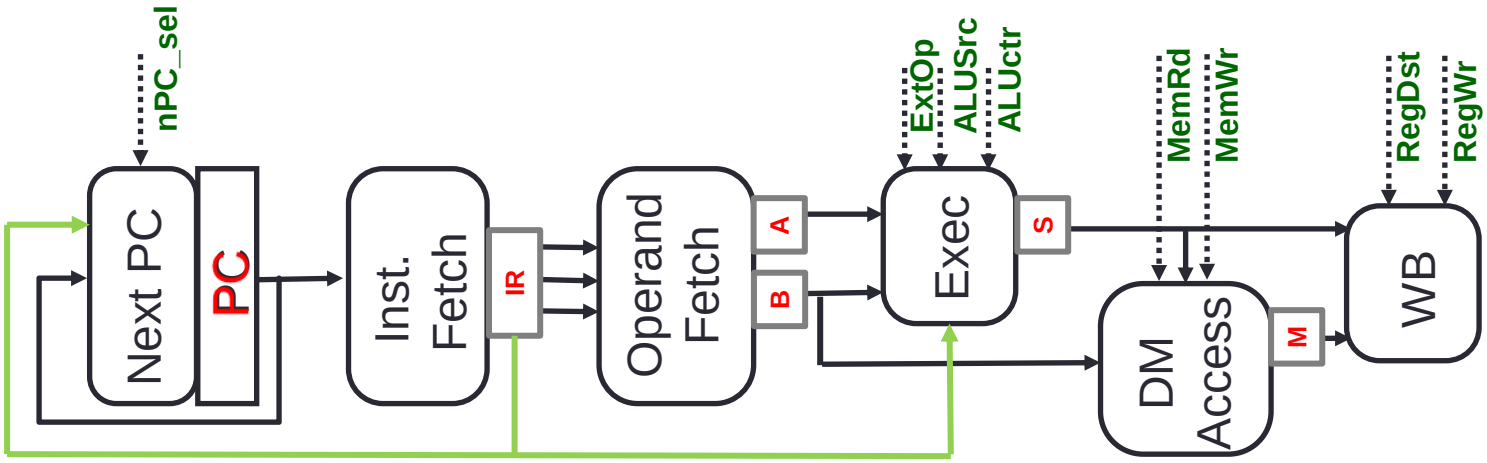
- **To encode the states of the FSM we just discussed, as shown below, how many bits do we need?**



In-class exercise (4)

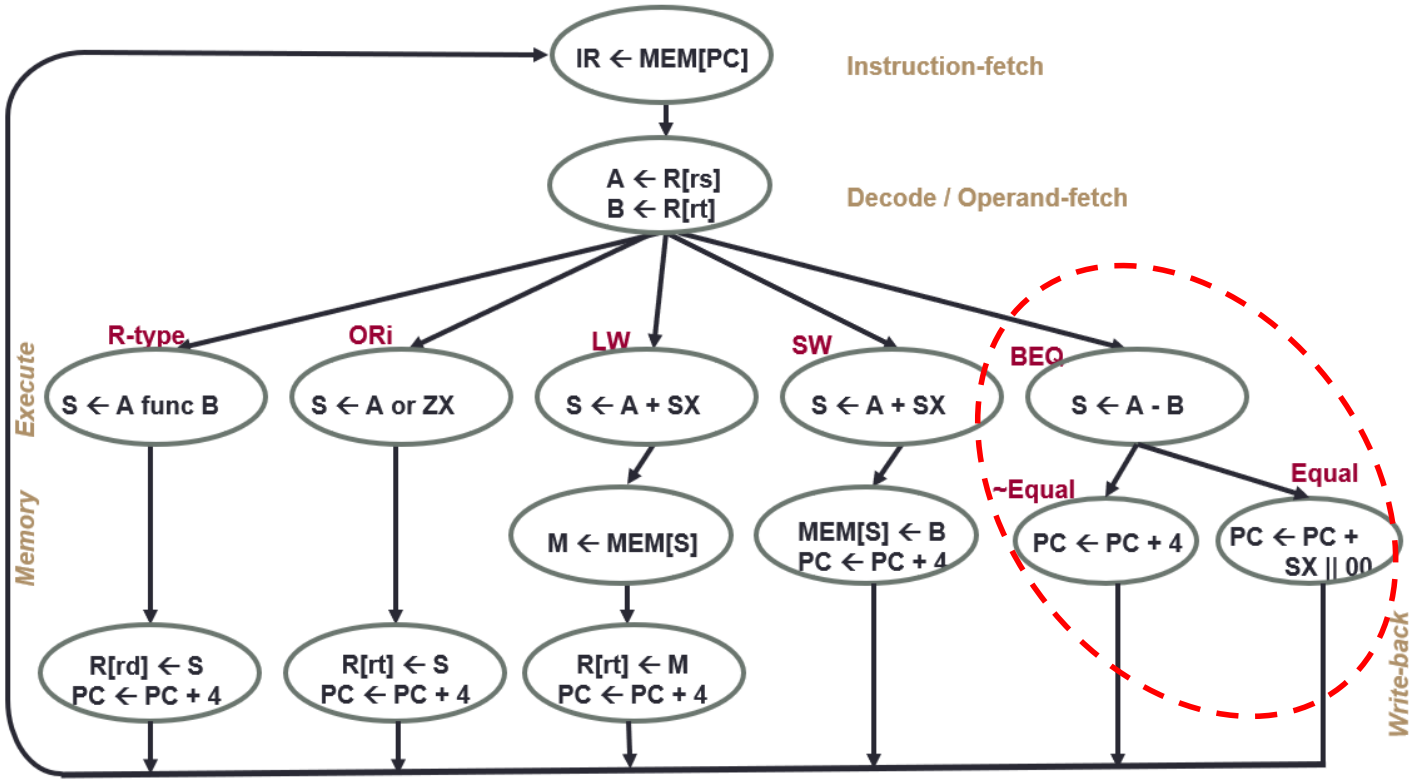
- **For the BEQ instruction, what control signals are involved?**

- **What logic expression needs to be implemented for control signal *nPC_sel*?**



In-class exercise (5)

- **If we merge the three states for BEQ and encode the new state as G3G2G1G0 = 1111, what is the new logic expression for *nPC_sel*?**



In-class exercise (6)

To execute the following code how many clock cycles are needed?

```
lw      $10, 20($1)
subu    $11, $2, $3
ori     $13, $6, $7
addu    $14, $8, $9
```