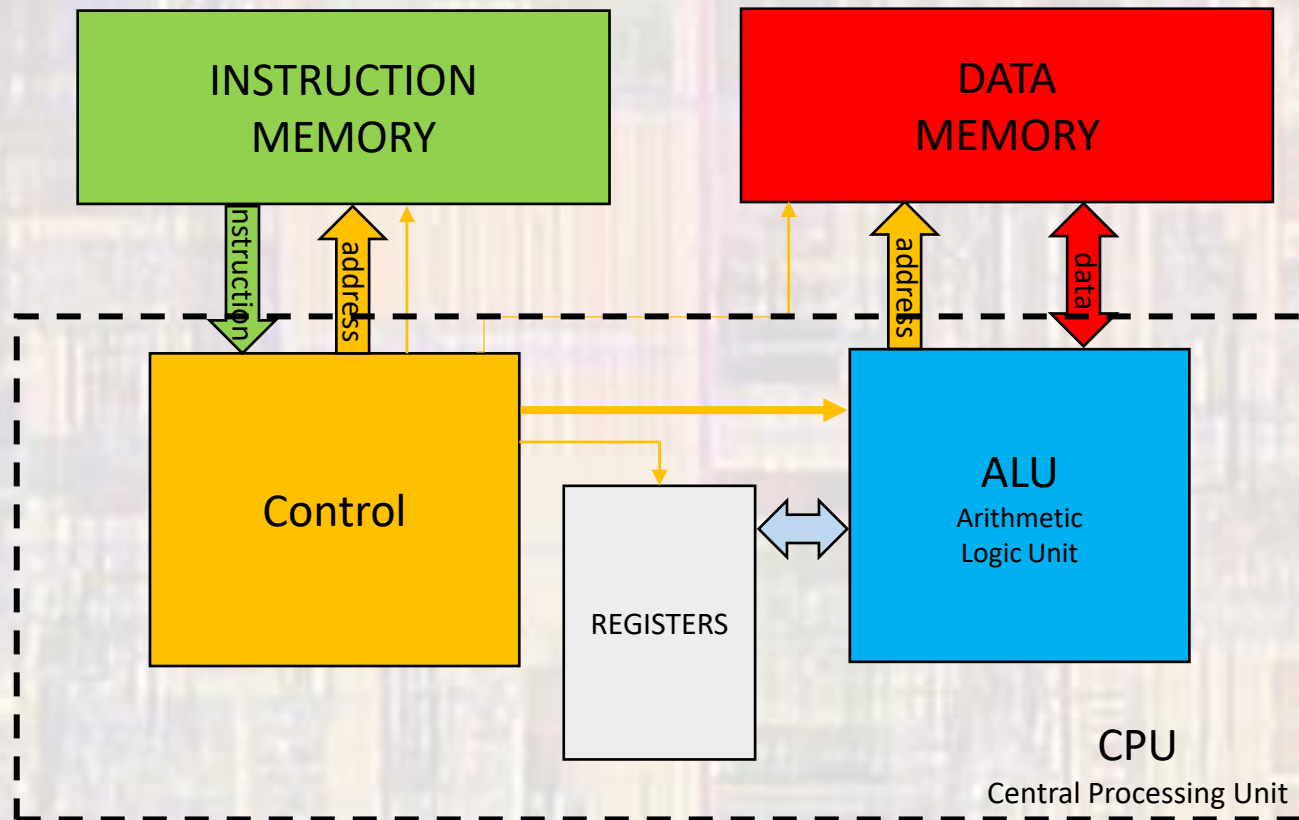


Program Execution

Last updated 8/15/22

Program Execution

- Processor Architecture
 - Harvard – separate Instruction and Data memory paths



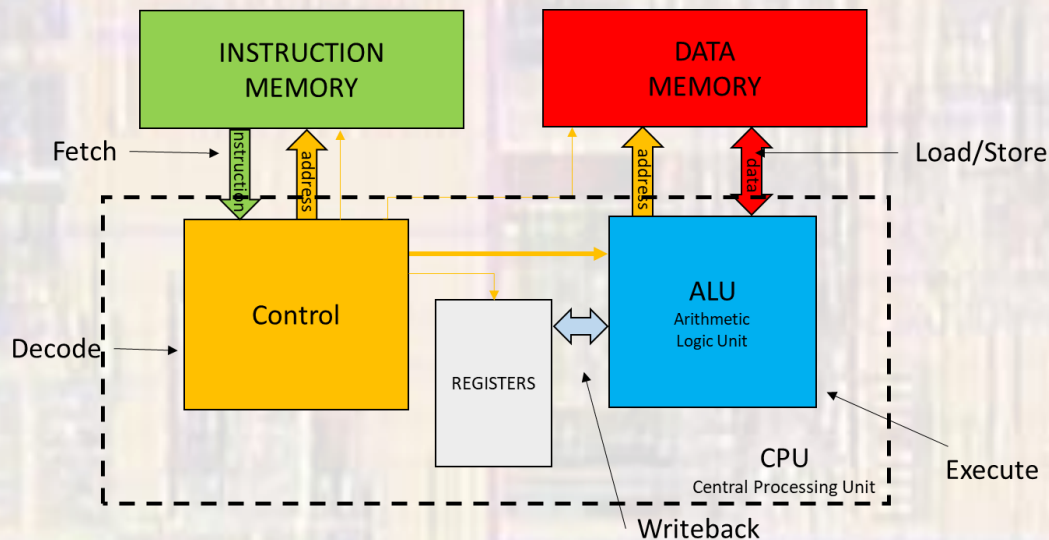
Program Execution

- RISC Instruction set
 - 2 basic types of instructions
 - Register based instructions
 - Memory instructions
 - Register Instructions
 - Only require access to the internal registers
 - Arithmetic
 - Logical
 - Control
 - Memory Operations
 - Read or write to memory/registers

Program Execution

- Instruction Execution

	Fetch	get next instruction from instruction memory
	Decode	determine what the instruction is
depend on instruction	Execute	if necessary – do what the instruction requires
	Load	if necessary – get value from data memory
	Store	if necessary – place value in data memory
	Write Back	if necessary – store result in register



Program Execution

- Instruction Sequencing
 - Program Counter (PC)
 - Register that holds the NEXT instruction memory location to be fetched
 - Provides the address for the instruction memory read
 - Typically the register is incremented each clock cycle
 - Incremented by the size of an instruction
 - e.g. for a 16 bit instruction word the PC would be incremented by 2

Program Execution

- Instruction Sequencing
 - Program control
 - Linear flow – increment PC normally
 - Function call
 - Store the “planned” next instruction address somewhere
 - Place the first instruction address for the function in the PC
 - Execute linearly in the function until done
 - Restore the “planned” next instruction address

Program Execution

- 1 line of code - complete

$a = b + c;$

The compiler turned the single line into 7 instructions

The compiler has assigned

b to memory location 4000

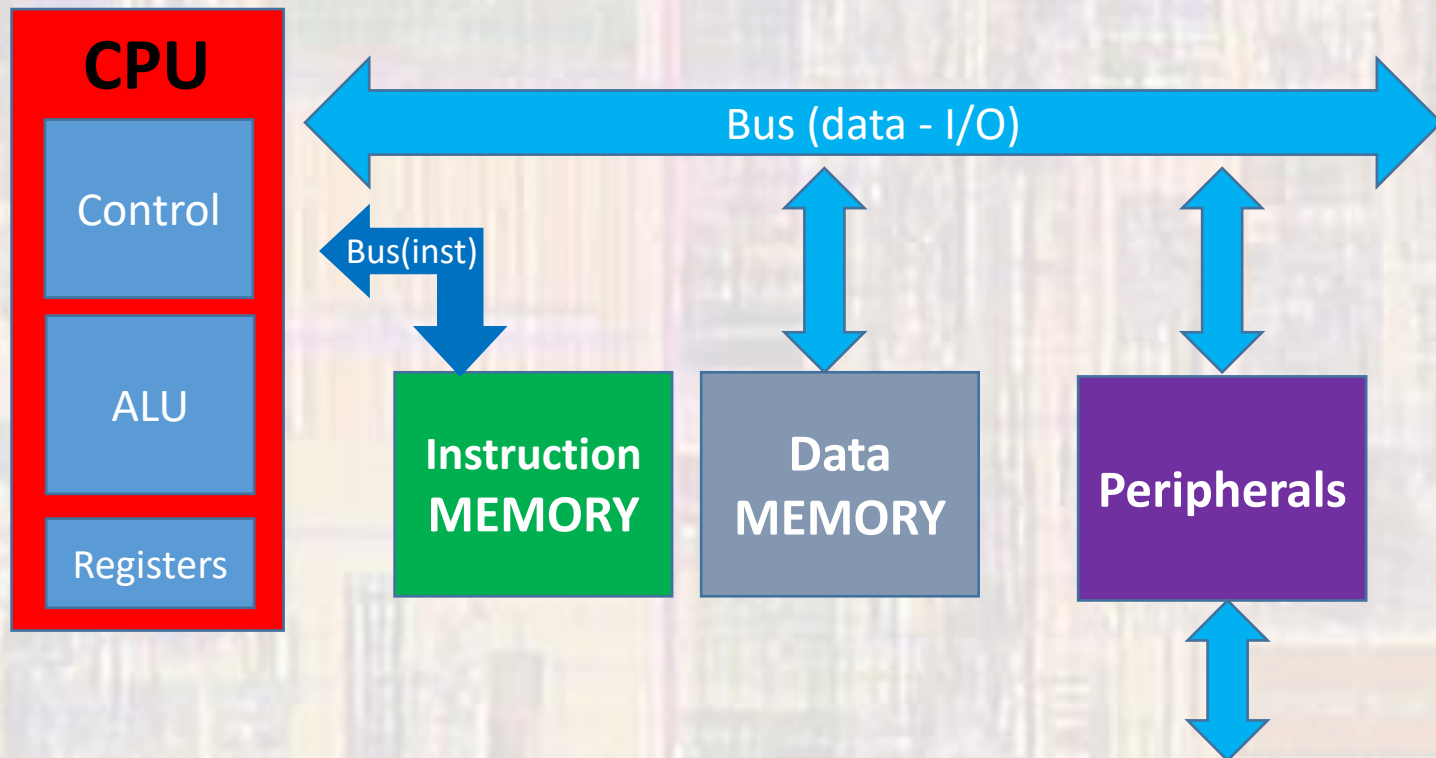
c to memory location 4004

a to memory location 4008

Mem loc	Instruction	Encoding	action
1000	ldi R1, 4000	90	Load loc for B into R1
1002	ld R2, mem(R1)	11	Put value at loc for B in R2
1004	ldi R1, 4004	92	Load loc for C into R1
1006	ld R3, mem(R1)	12	Put value at loc for C in R3
1008	add R2, R3, R4	27	$R4 \leftarrow R2 + R3$
100A	ldi R1, 4008	84	Load loc for A into R1
100C	st mem(R1), R4	21	Put value of R4 into loc for A

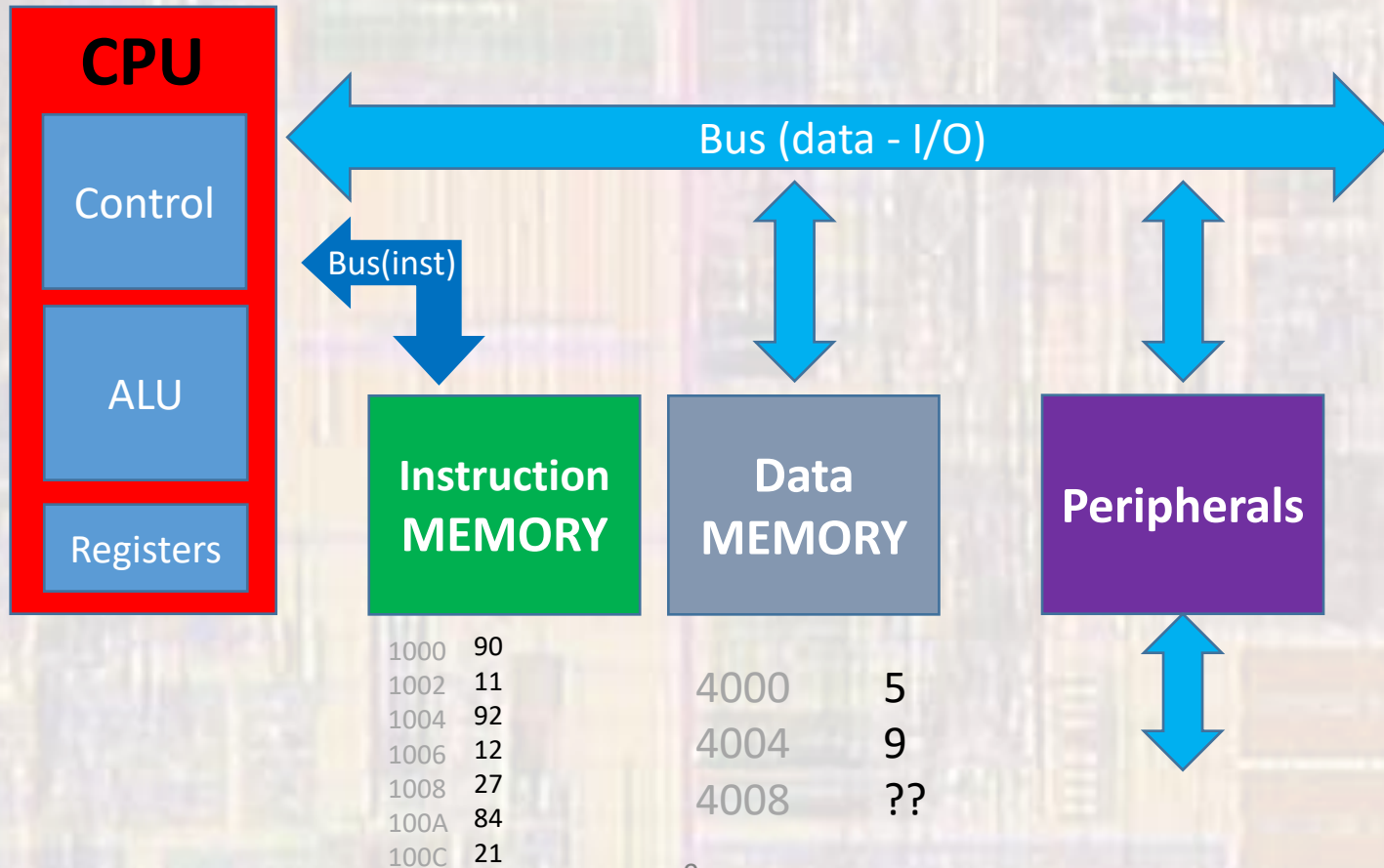
Program Execution

- Simplified Block Diagram



Program Execution

- Status
 - Data locations filled by previous commands
 - PC currently pointing to Instruction memory location 1000



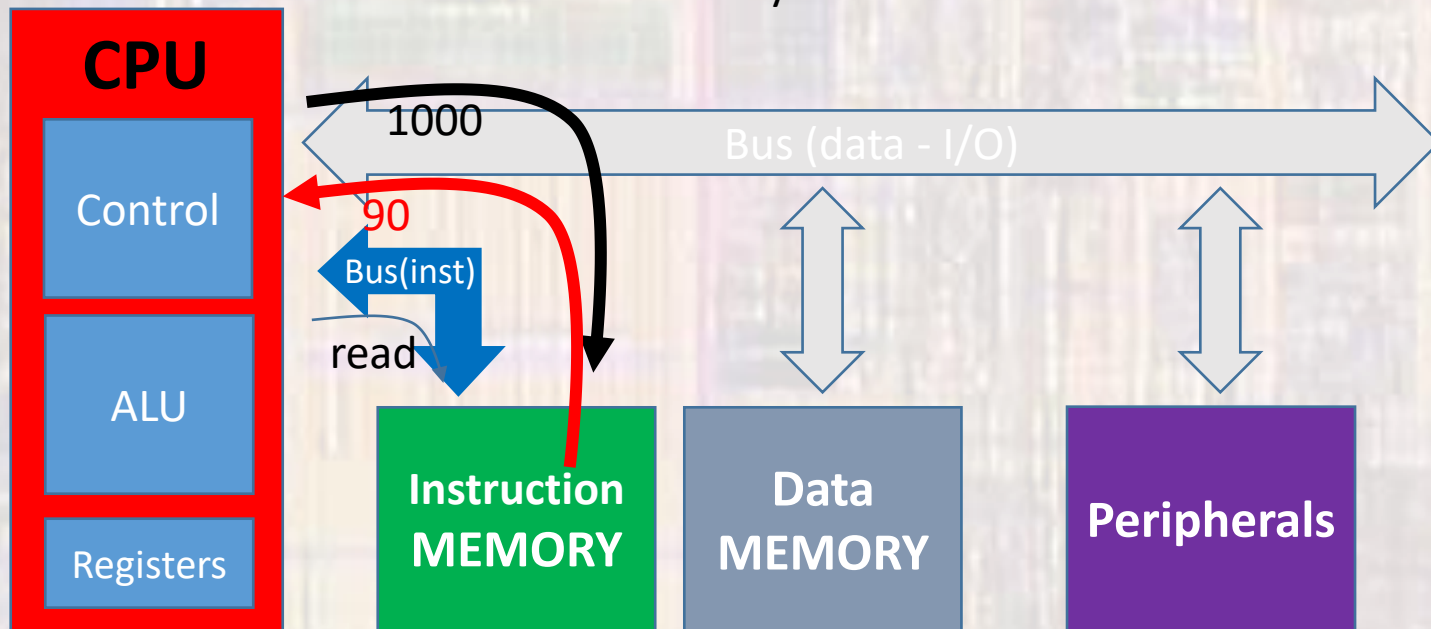
R1 ??
R2 ??
R3 ??
R4 ??

Program Execution

- Request Instruction (fetch)

Control puts a memory location (1000) on the address bus along with a read signal

Instruction memory returns the value at that location (90)



R1 ??
R2 ??
R3 ??
R4 ??

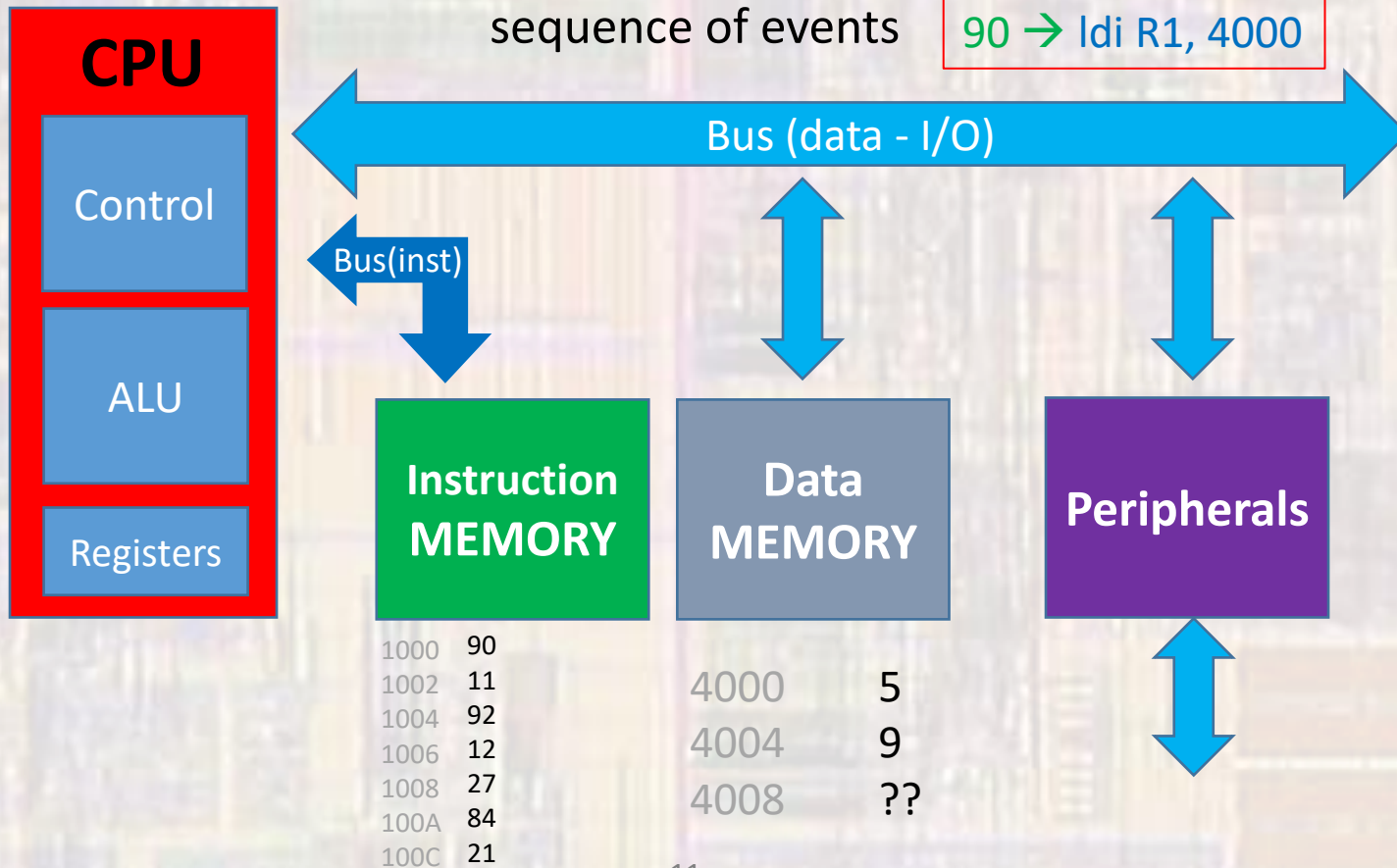
1000	90
1002	11
1004	92
1006	12
1008	27
100A	84
100C	21

4000	5
4004	9
4008	??

Program Execution

- Request Instruction (decode)

Control decodes the word returned by the memory and prepares to execute a pre-defined sequence of events 90 → Idi R1, 4000

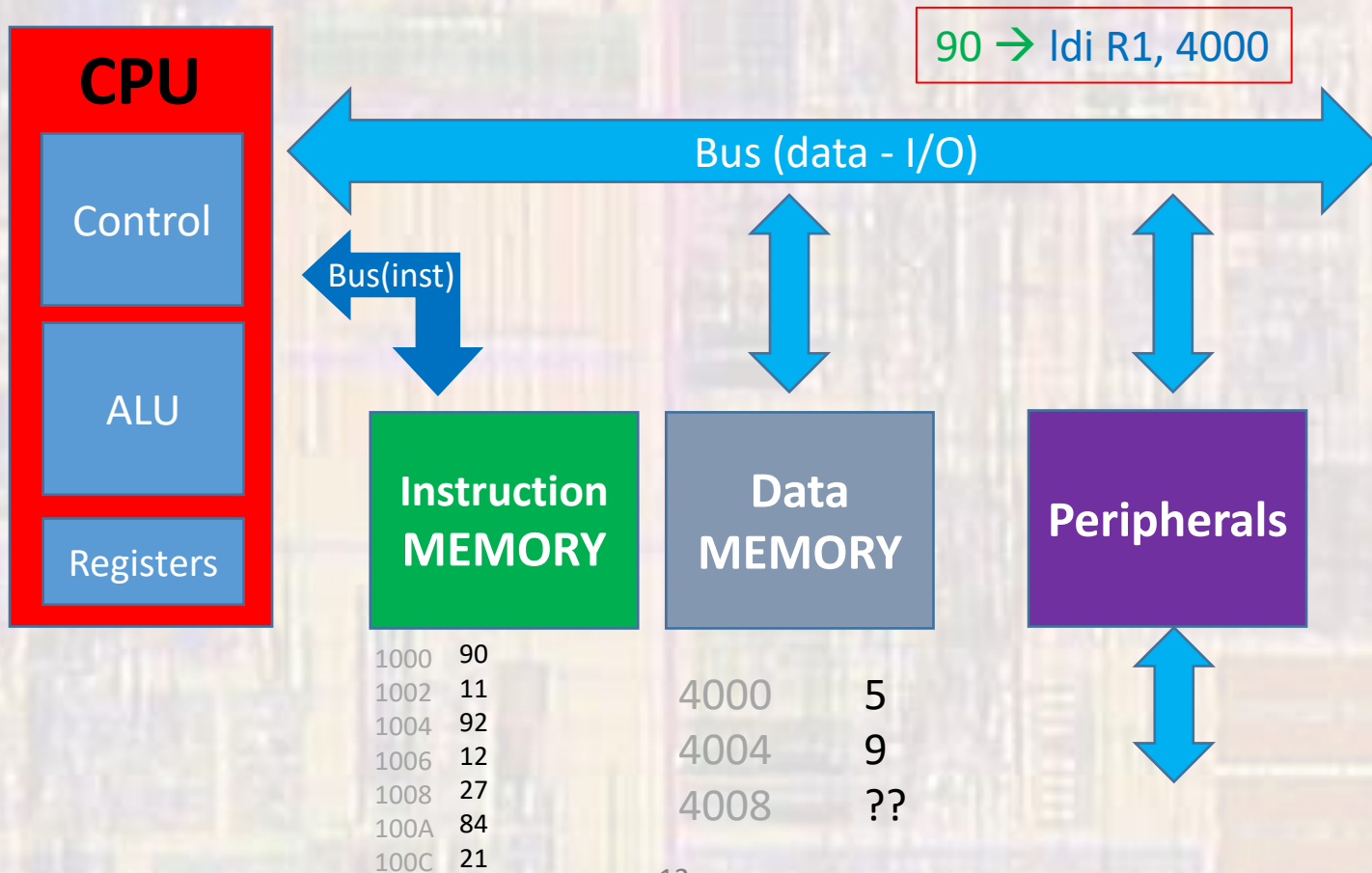


R1 ??
R2 ??
R3 ??
R4 ??

Program Execution

- Request Instruction (execute)

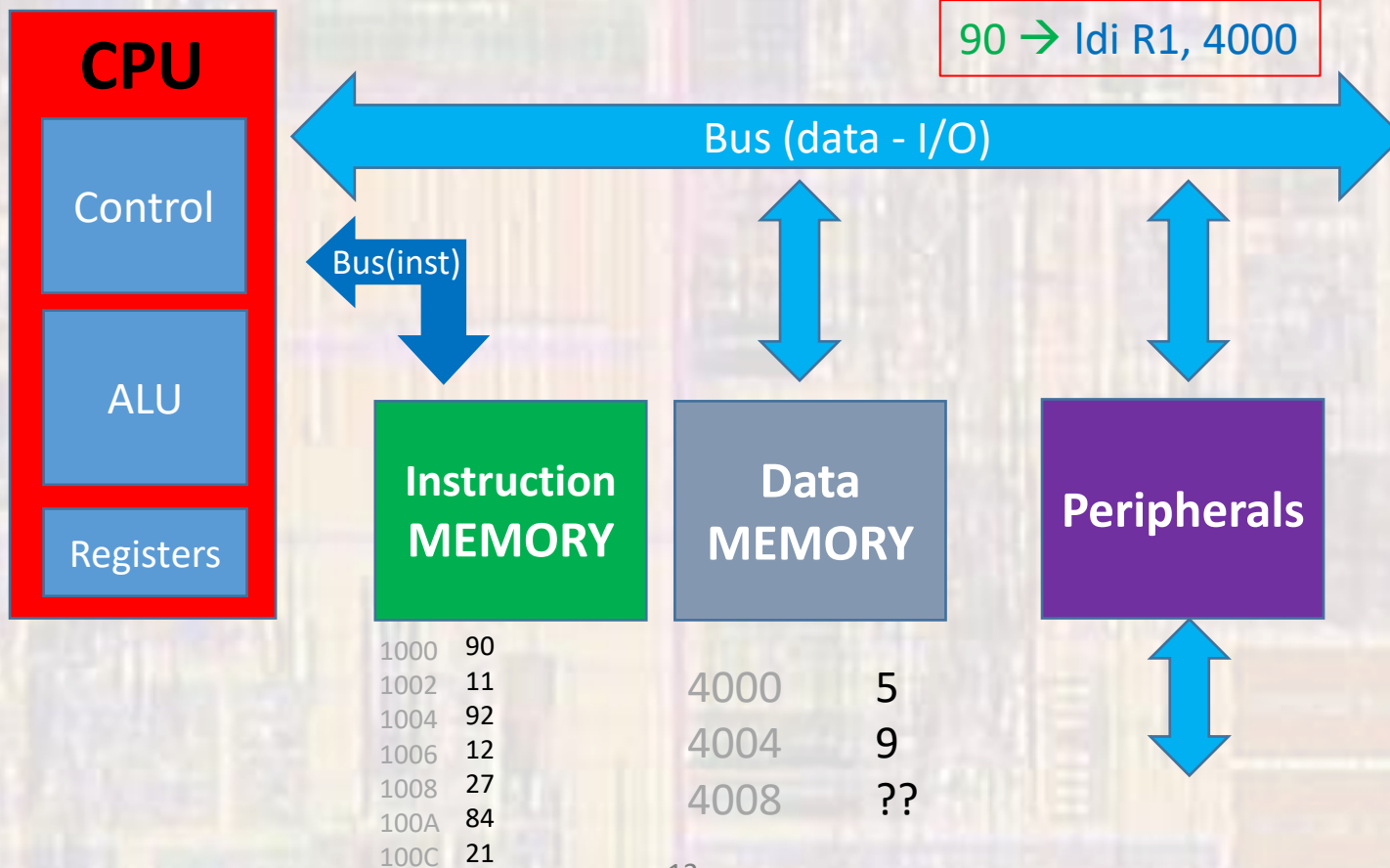
Does nothing for this instruction



Program Execution

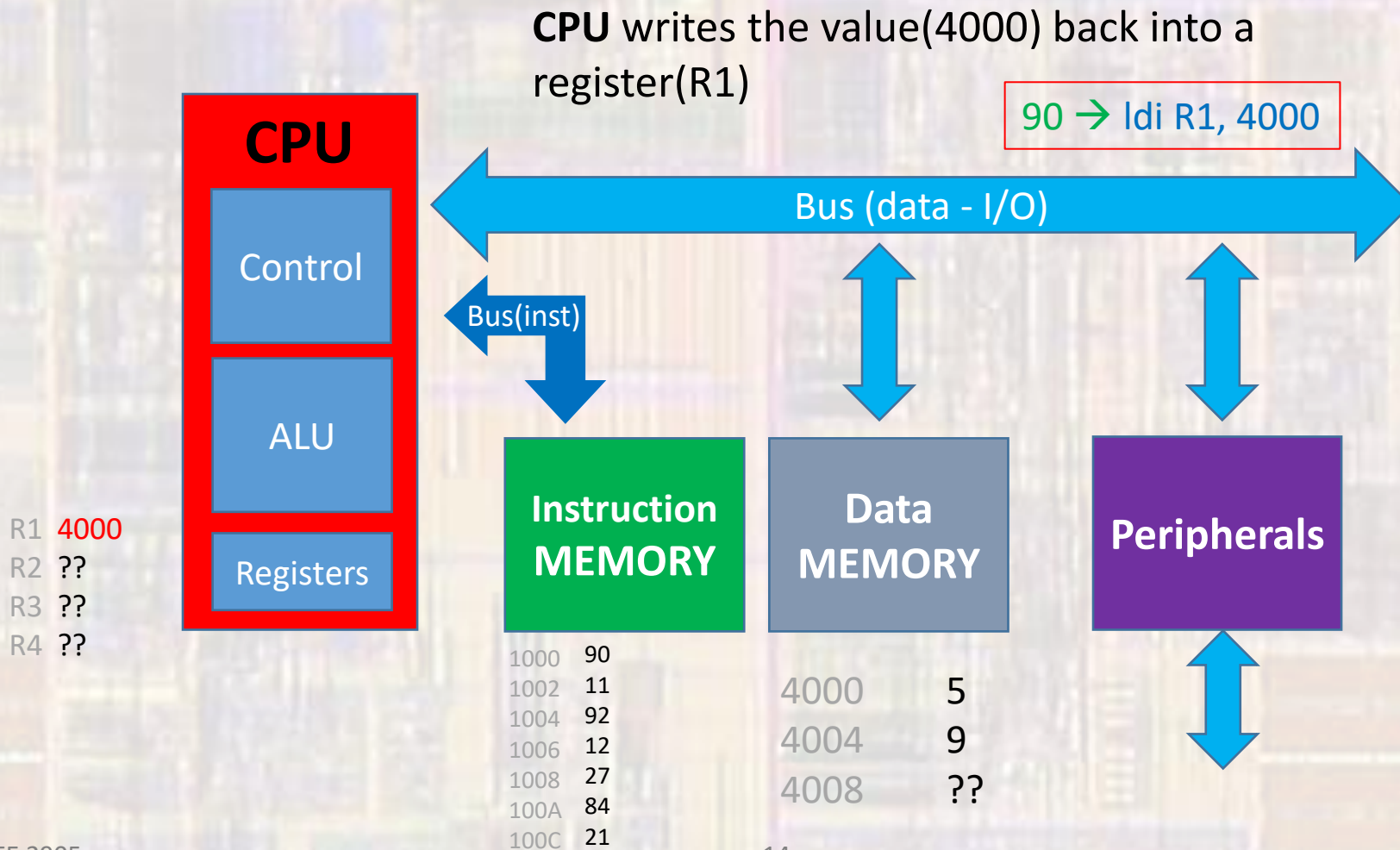
- Request Instruction (mem)

Does nothing for this instruction



Program Execution

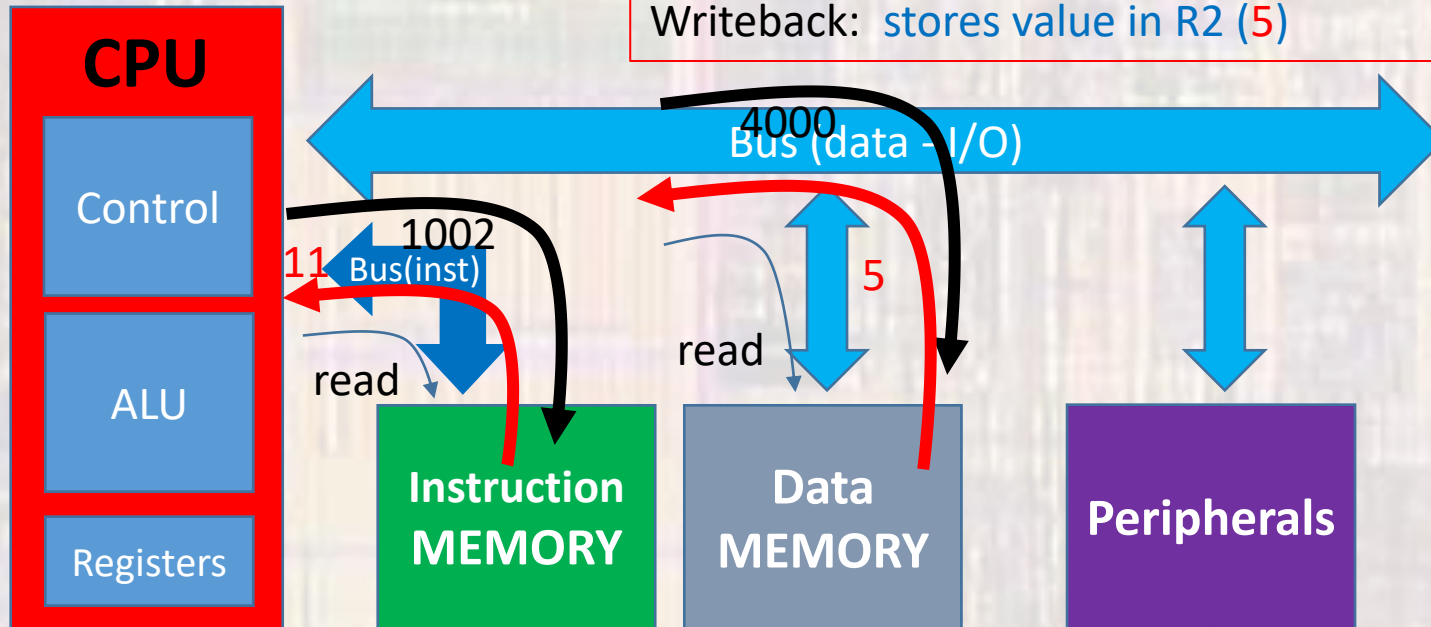
- Request Instruction (write back)



Program Execution

- PC advances (1002)

Fetch: → 1002 ← 11
 Decode: 11 → ld R2, mem(R1)
 Execute: idle
 MEM: value at location in R1(4000) ← (5)
 Writeback: stores value in R2 (5)



R1 4000
 R2 5
 R3 ??
 R4 ??

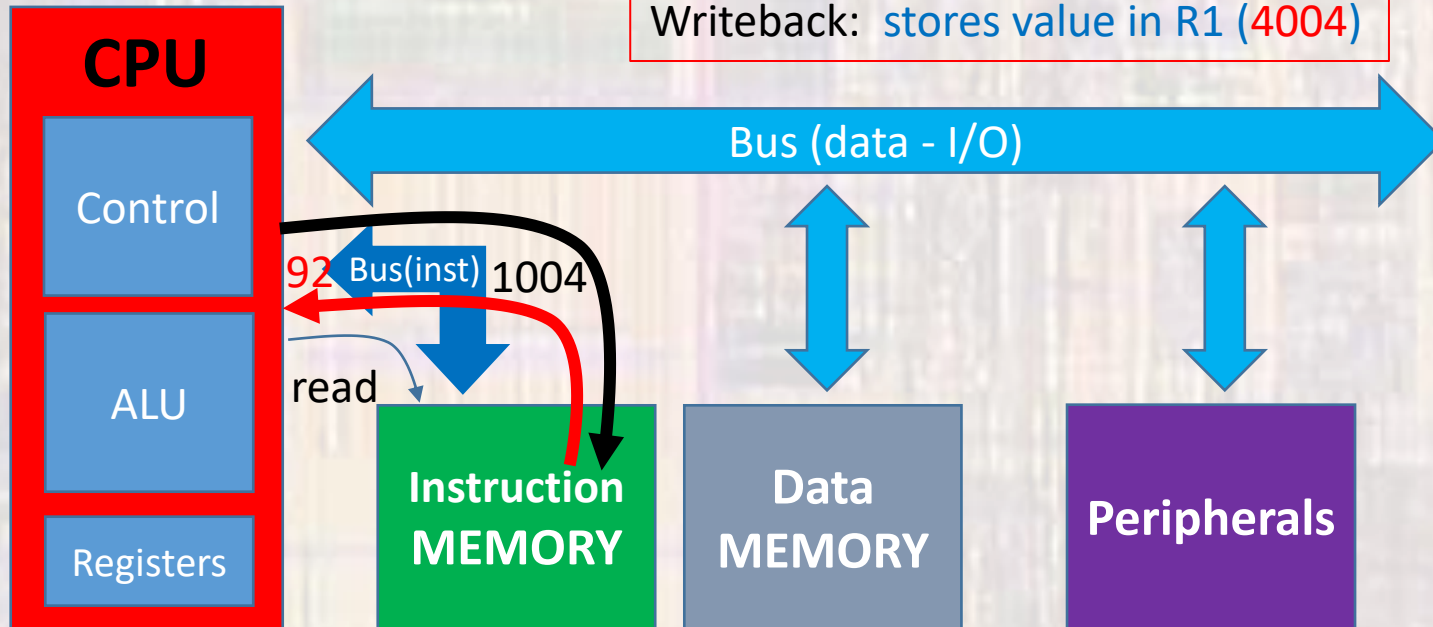
1000	90
1002	11
1004	92
1006	12
1008	27
100A	84
100C	21

4000	5
4004	9
4008	??

Program Execution

- PC advances (1004)

Fetch: → 1004 ← 92
 Decode: 92 → Idi R1, 4004
 Execute: idle
 MEM: idle
 Writeback: stores value in R1 (4004)



R1 4004
 R2 5
 R3 ??
 R4 ??

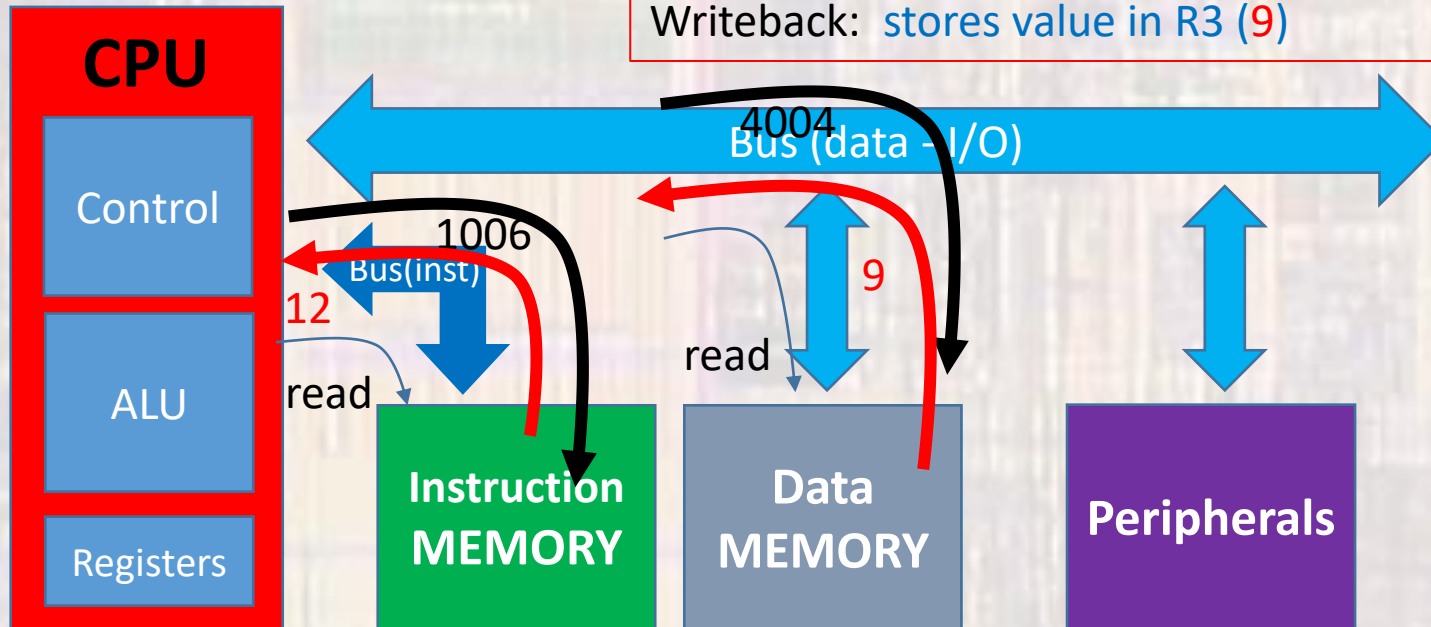
1000 90
 1002 11
 1004 92
 1006 12
 1008 27
 100A 84
 100C 21

4000 5
 4004 9
 4008 ??

Program Execution

- PC advances (1006)

Fetch: → 1006 ← 12
 Decode: 12 → ld R3, mem(R1)
 Execute: idle
 MEM: value at location in R1(4004) ← (9)
 Writeback: stores value in R3 (9)



R1 4004
 R2 5
 R3 9
 R4 ??

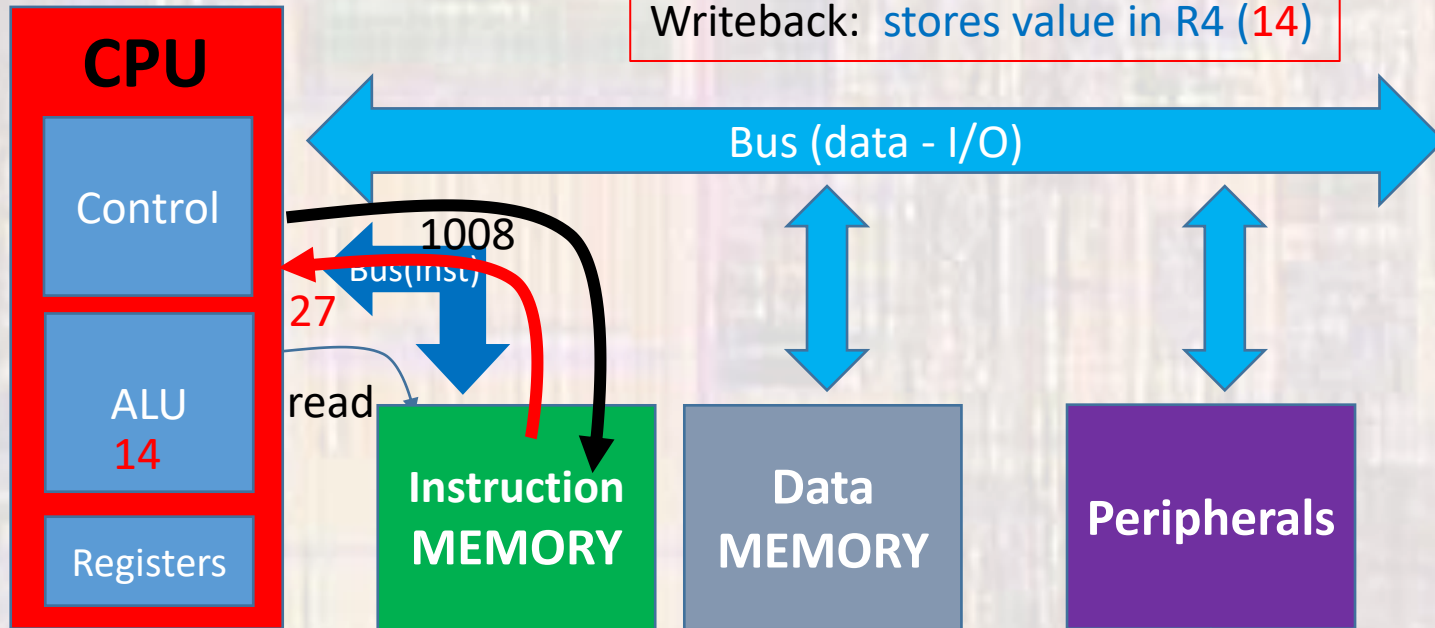
1000 90
 1002 11
 1004 92
 1006 12
 1008 27
 100A 84
 100C 21

4000 5
 4004 9
 4008 ??

Program Execution

- PC advances (1008)

Fetch: $\rightarrow 1008 \leftarrow 27$
 Decode: $27 \rightarrow$ add R2, R3, R4
 Execute: adds R2 + R3 $\rightarrow 14$
 MEM: idle
 Writeback: stores value in R4 (14)



R1 4004
 R2 5
 R3 9
 R4 14

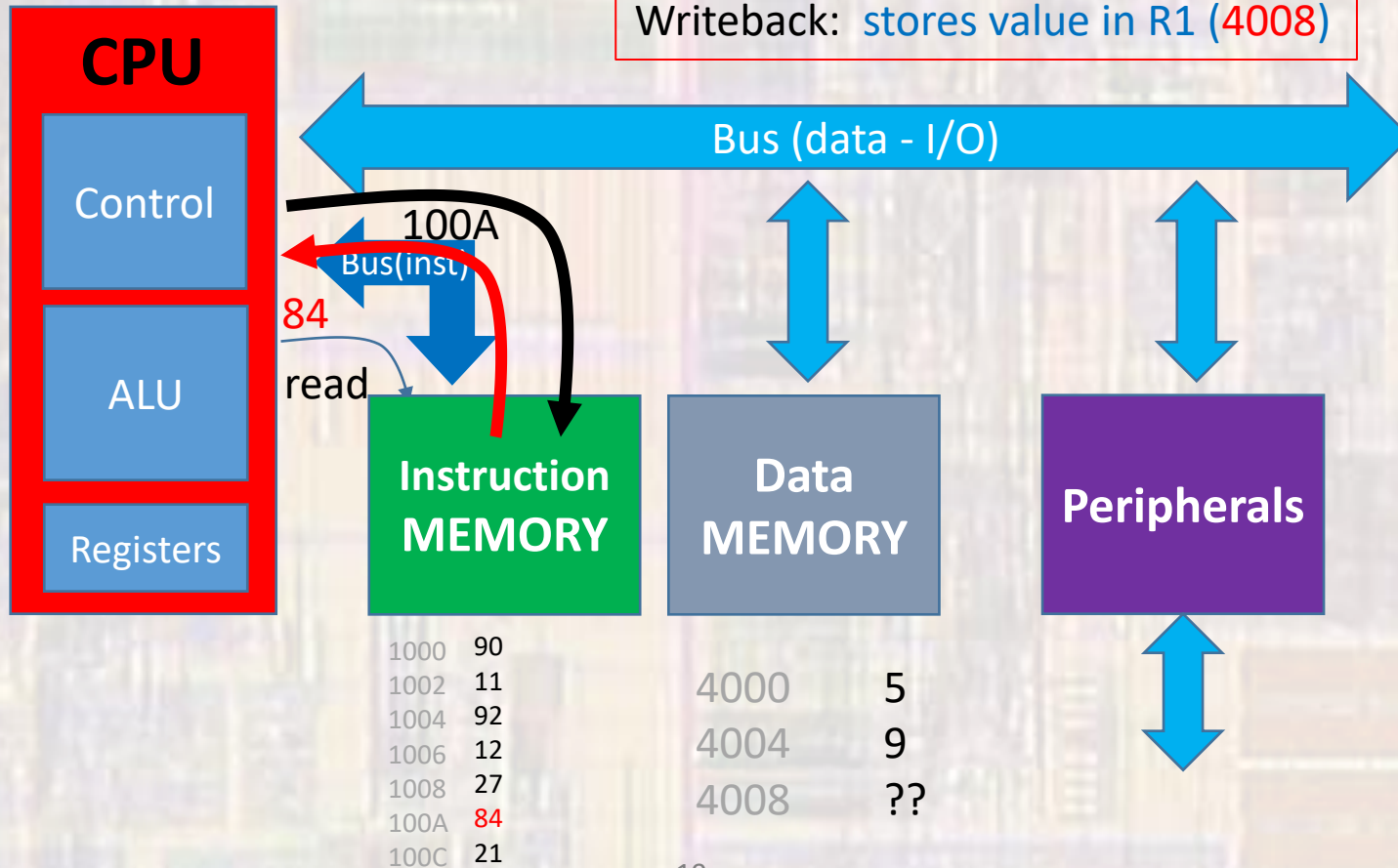
1000 90
 1002 11
 1004 92
 1006 12
 1008 27
 100A 84
 100C 21

4000 5
 4004 9
 4008 ??

Program Execution

- PC advances (100A)

Fetch: → 100A ← 84
 Decode: 84 → Idi R1, 4008
 Execute: idle
 MEM: idle
 Writeback: stores value in R1 (4008)



R1 4008
 R2 5
 R3 9
 R4 14

Program Execution

- PC advances (100C)

Fetch: → 100C ← 21
 Decode: 21 → st mem(R1), R4
 Execute: idle
 MEM: R4(14) → location in R1(4008)
 Writeback: idle

