Control Logic I

CSCI 2050U - Computer Architecture

Randy J. Fortier
@randy_fortier

OntarioTech UNIVERSITY

Outline

- Instruction cycle
- Additional digital circuit components
- Data path/bus
- Fetch

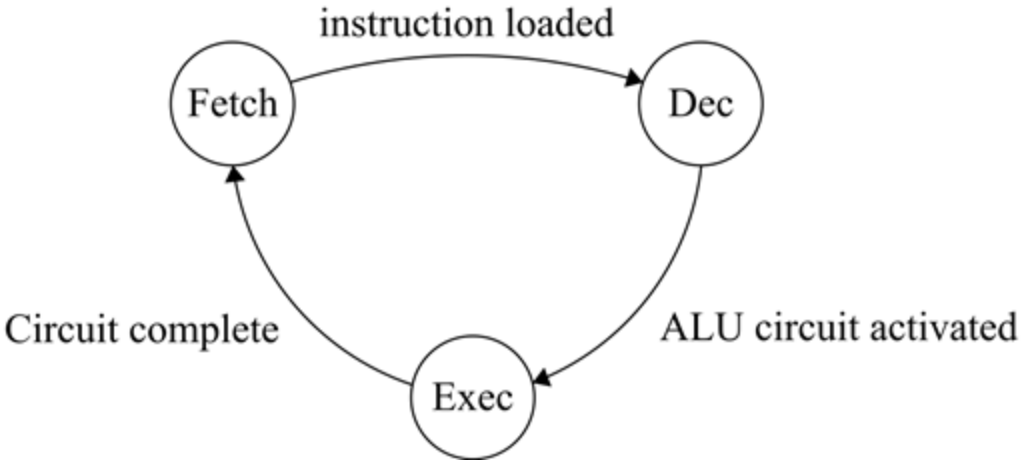
Instruction Cycle

CSCI 2050U - Computer Architecture

OntarioTech UNIVERSITY

The Instruction Cycle

- A computer executes a program one instruction at a time, according to the instruction cycle:
  - Fetch
  - Decode
  - Execute



The Instruction Cycle

- Fetch:
  - Instruction fetch: Load the instruction from memory
  - Operand fetch: Load the operand(s) from memory (if any)

The Instruction Cycle

- Decode:
  - The control unit handles decoding
  - Circuit activation
    - Activate the circuit (e.g. in the ALU) to perform the requested operation
    - De-activate all other circuitry
  - Register activation
    - Activate the registers to be used for input operands
    - Activate the registers to be used for the result of the operation
    - De-activate all other registers

The Instruction Cycle

- Execute:
  - Allow the data to pass:
    - From the input registers
    - Through the activated ALU circuit
    - Into the output register

A Simple Computer System

- Hypothetical Academic Computer System (HAX)
  - Simple, RISC-style, instruction set
    - Each instruction has a 4-bit opcode, 4-bits of padding, and 8-bits of operand
  - 256 word memory (8-bit words), total of 256 bytes of memory
  - 8-bit data path
  - Seven 8-bit general-purpose registers (A, B, C, D, E, F, and G)
  - Special-purpose registers:
    - PC: Program counter (address of the next instruction)
    - IR: Instruction register (stores the instruction opcode)
    - MAR: Memory address register (the address in memory for read/write)
    - MBR: Memory buffer register (the data to be written, the data read)
    - FLAGS: Zero, Greater Than, Carry, Overflow

More Digital Circuit Components

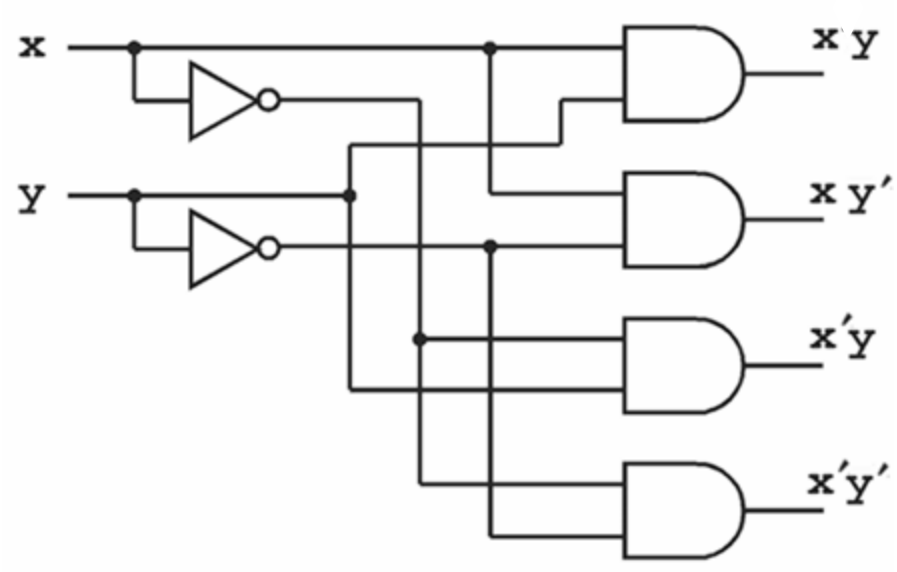
CSCI 2050U - Computer Architecture

Randy J. Fortier
@randy_fortier

OntarioTech UNIVERSITY

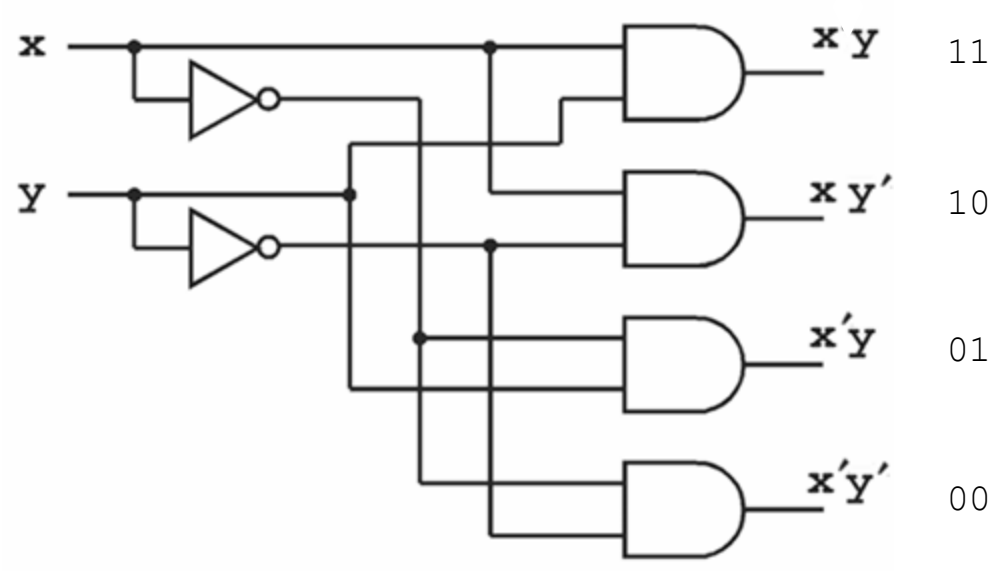
Decoders

- A decoder (DEC) is a component which activates (i.e. voltage high, e.g. 5v) one of its output lines for each unique input combination
  - Inputs: n
  - Outputs: 2^n



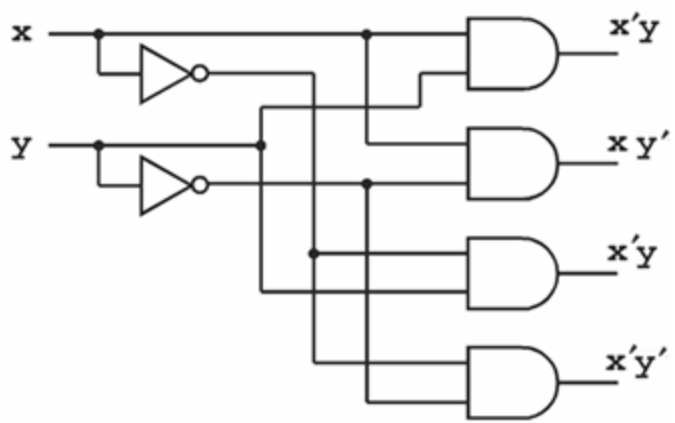
Decoders

- A decoder is a component which activates (i.e. voltage high, e.g. 5v) one of its output lines for each unique input combination
  - This is a 2-to-4 line decoder



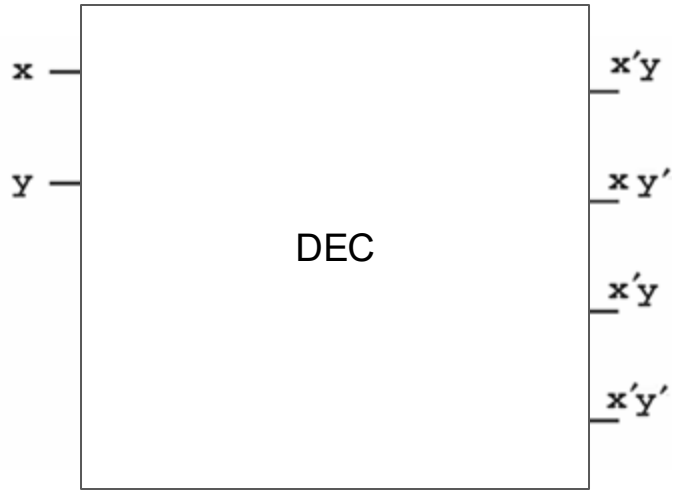
Decoders

- A decoder is often used for two important purposes:
  - Decoding an instruction, enabling the correct ALU circuit
  - Decoding a memory address, enabling the inputs and/or outputs of the correct memory cell



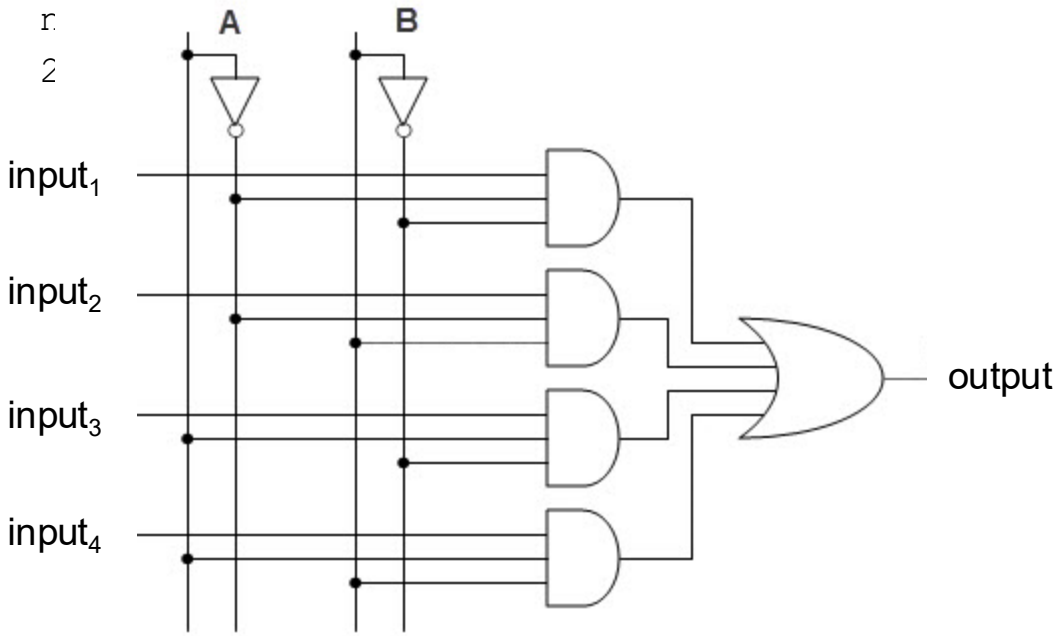
Decoders

- Decoders and other components can be drawn in block notation
  - This abstraction simplifies our job as we move to more complex circuits



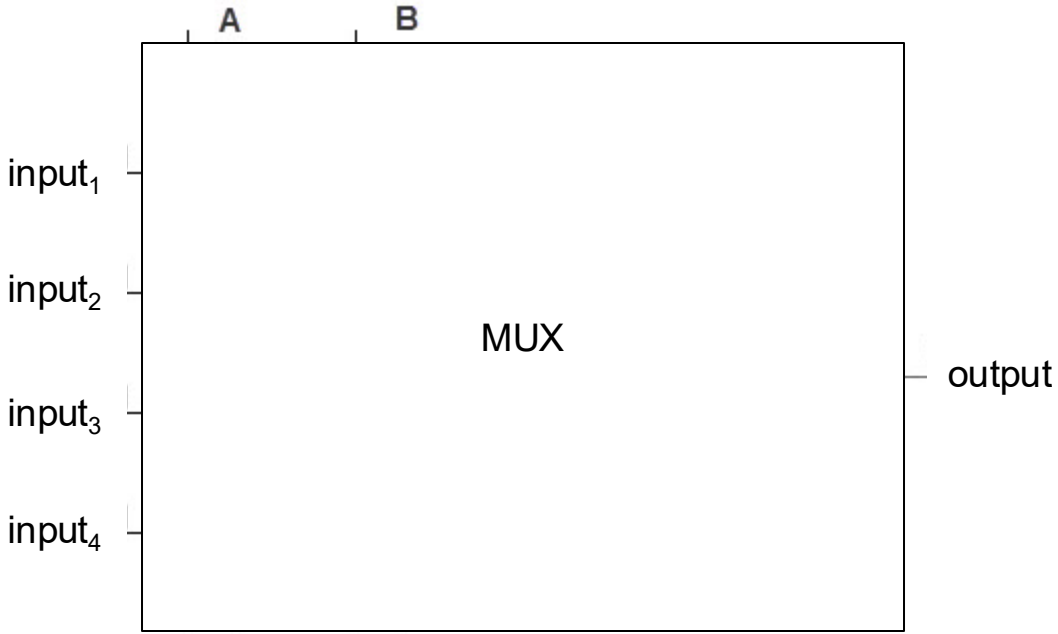
Multiplexer

- A multiplexer (MUX) is a component with many inputs and only one output
  - This is a 4-to-1 multiplexer
  - Selector bits (A and B in the diagram, below) decide which input gets mapped to the output
  - Selector bits: n
  - Input bits: 2



Multiplexer

- The multiplexer in block notation:



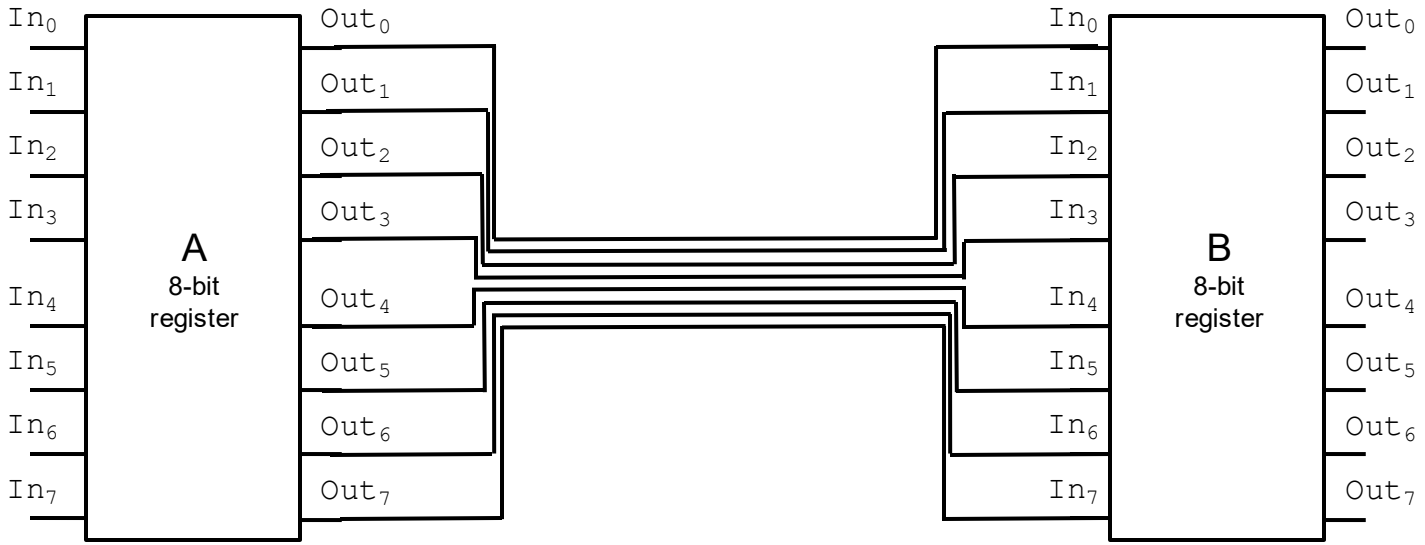
Buses and the Data Path

CSCI 2050U - Computer Architecture

OntarioTech UNIVERSITY

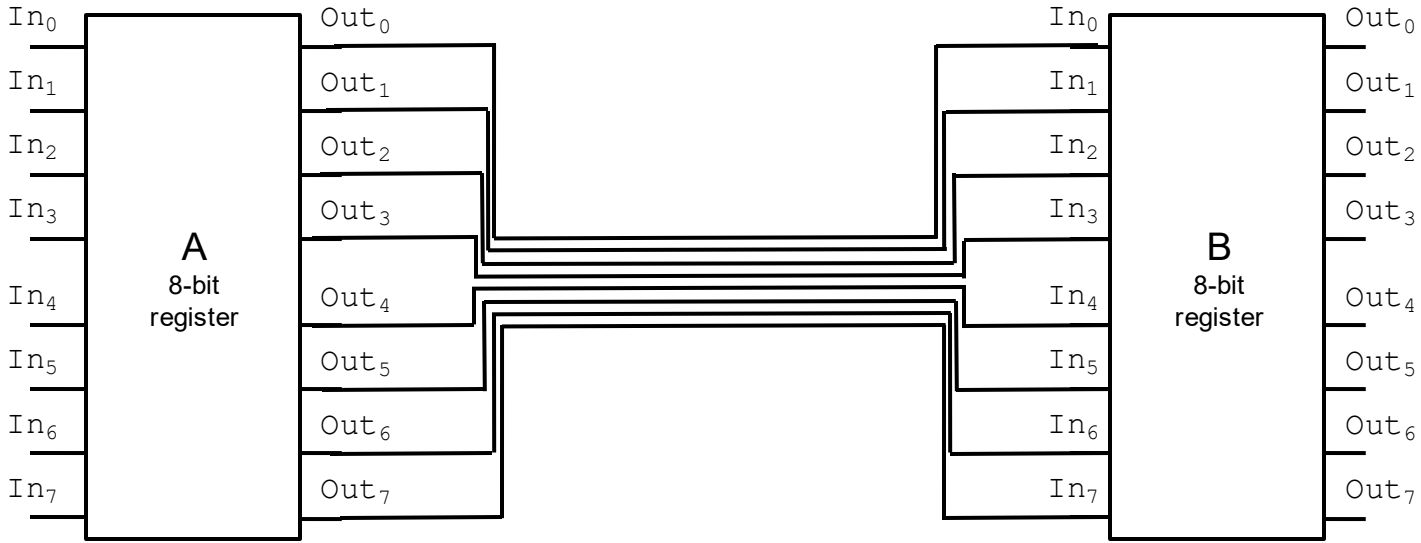
Buses

- A bus is similar to a wire or a connection
  - The key difference is that a bus has several parallel paths
  - e.g. Imagine an 8-bit bus between register A and register B

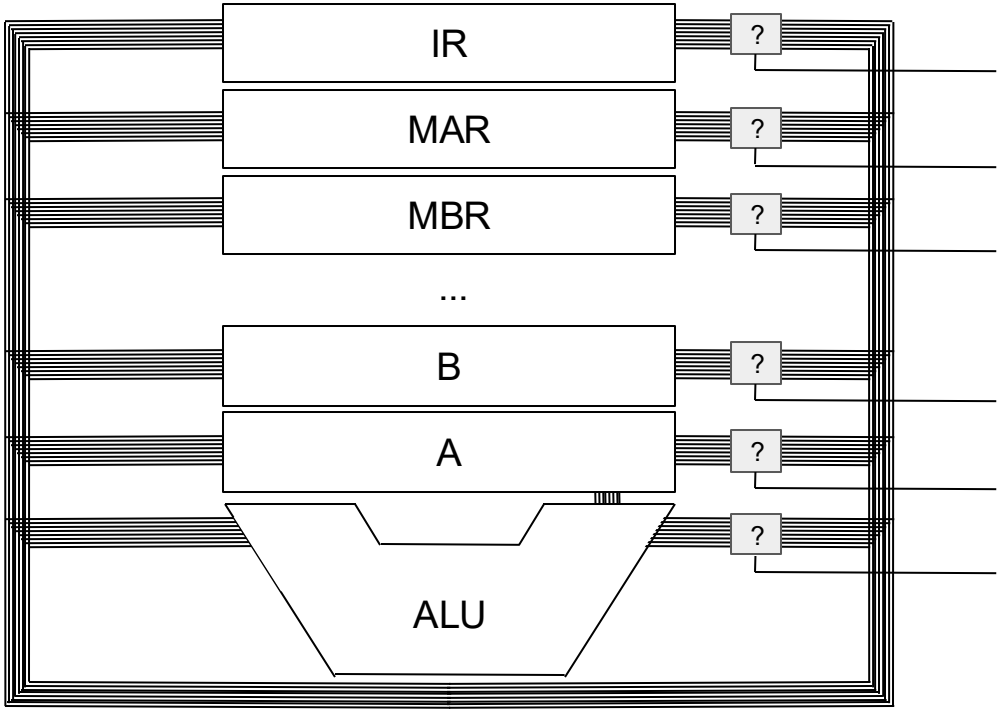


Buses

- A bus is very rarely point to point
  - Usually, there are more than 2 components connected to the bus
    - The bus is a shared medium
  - We need some mechanism to control what goes onto the bus

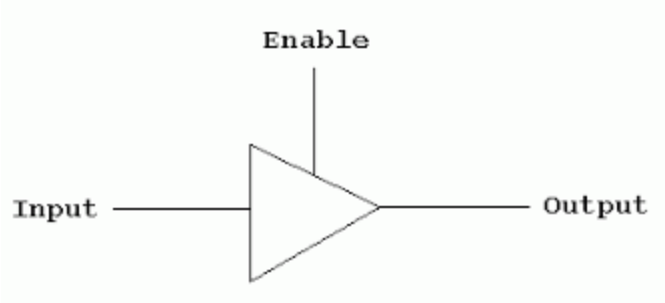


The Data Path: A Shared Bus



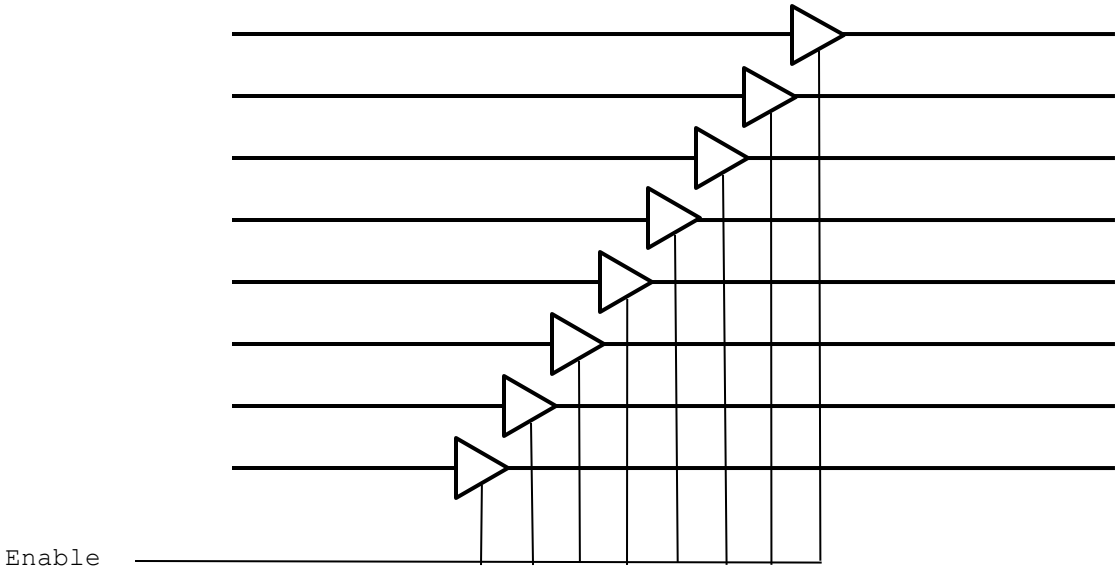
The Data Path: A Shared Bus

- It isn't really a problem if more than one component reads from the bus simultaneously
- It is only a problem when more than one component writes to the bus simultaneously
  - The signals collide, producing a distorted (unrecognizable) signal
- We block data going onto the bus using a tri-state buffer:



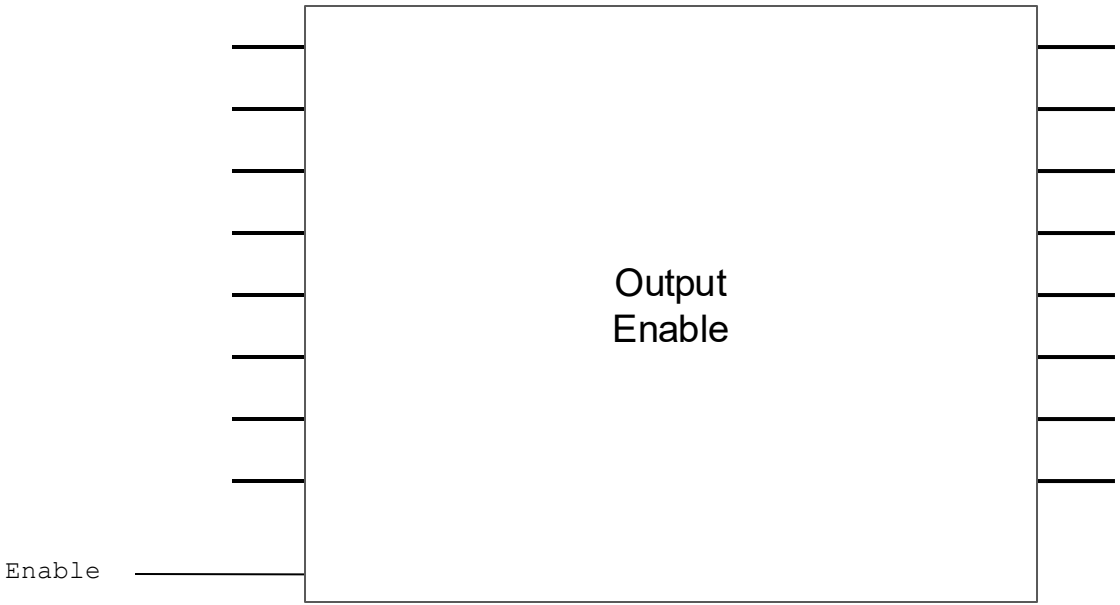
The Data Path: A Shared Bus

- Here are our tri-state buffer components

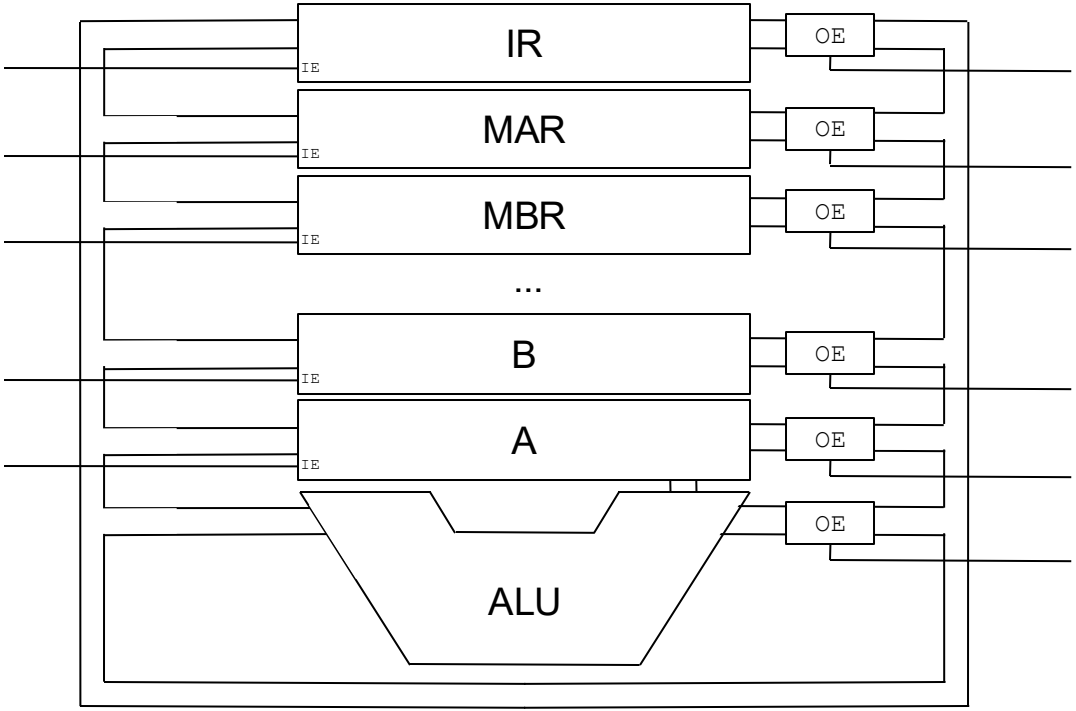


The Data Path: A Shared Bus

- Here are our tri-state buffer components

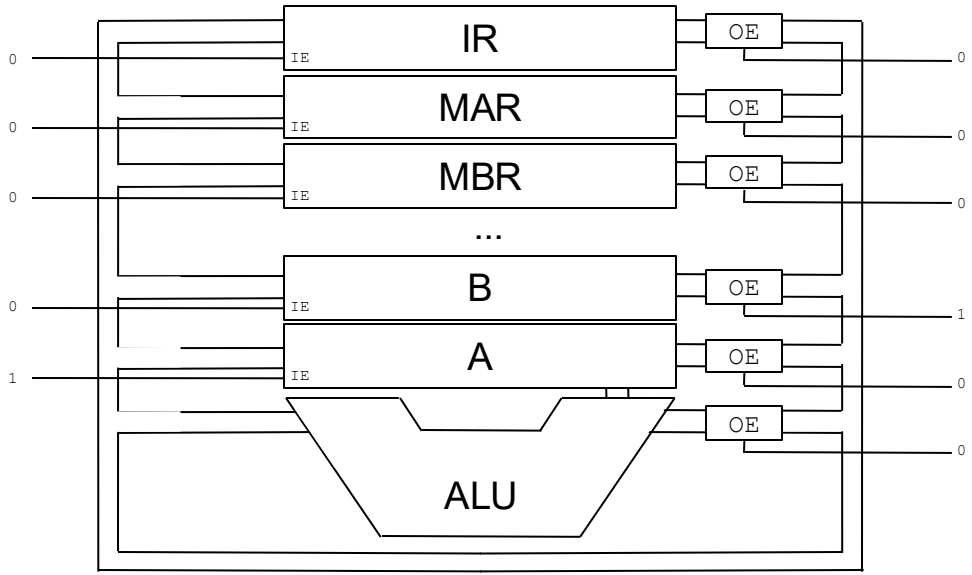


The Data Path: A Shared Bus



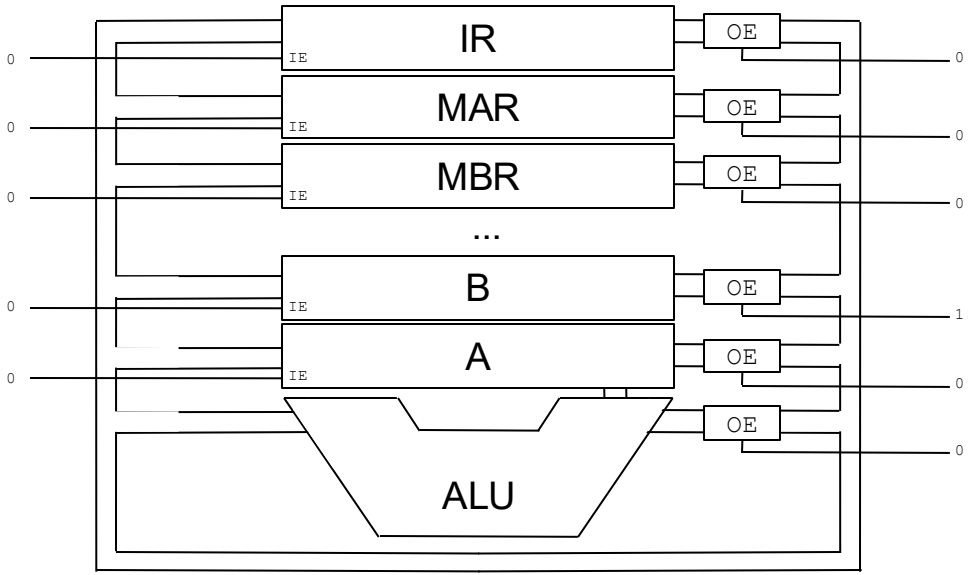
The Data Path: The Simplest Instruction (MOV)

- Let's consider the simplest instruction (a register transfer):
  - `MOV A, B`
    - Move the value from the register B into the register A



The Data Path: An Arithmetic Instruction (`ADD`)

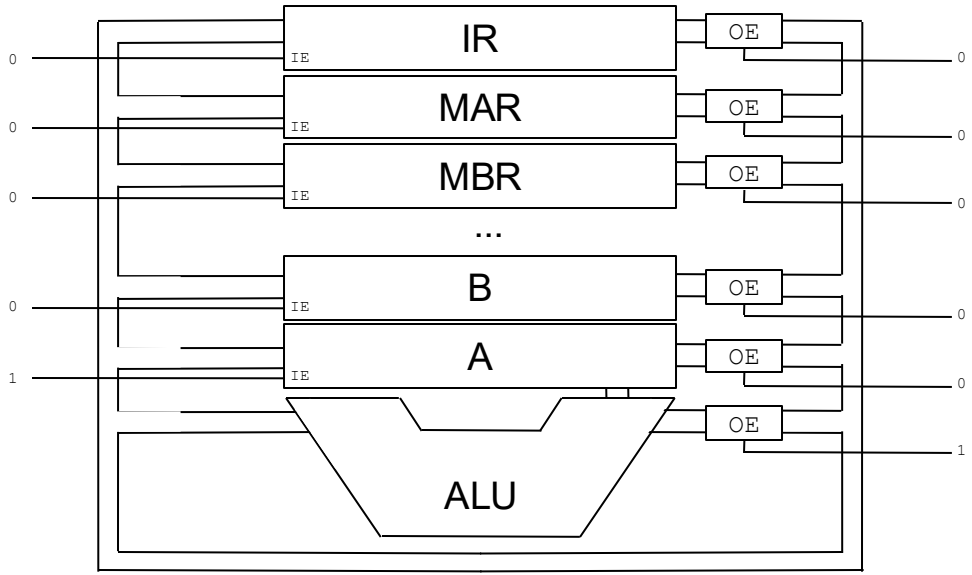
- Let's consider the simplest instruction:
  - `ADD A, B`
    - Add the value from the register B to the register A, storing the result in A



Cycle 1

The Data Path: An Arithmetic Instruction (`ADD`)

- Let's consider the simplest instruction:
  - `ADD A, B`
    - Add the value from the register B to the register A, storing the result in A



Cycle 2

Register Transfer Language (RTL)

- Many instructions can be written in RTL, which helps explain what the instructions do
  - These RTL instructions are also called microoperations (or microcode)
  - It isn't necessarily the case that the hardware implements these RTL instructions directly, but most processors do support them
  - Example (`MOV A, B`)
    - `A ← B`
    - `PC ← PC + 2`
    - `IR ← M[PC]`

Fetch

CSCI 2050U - Computer Architecture

Randy J. Fortier
@randy_fortier

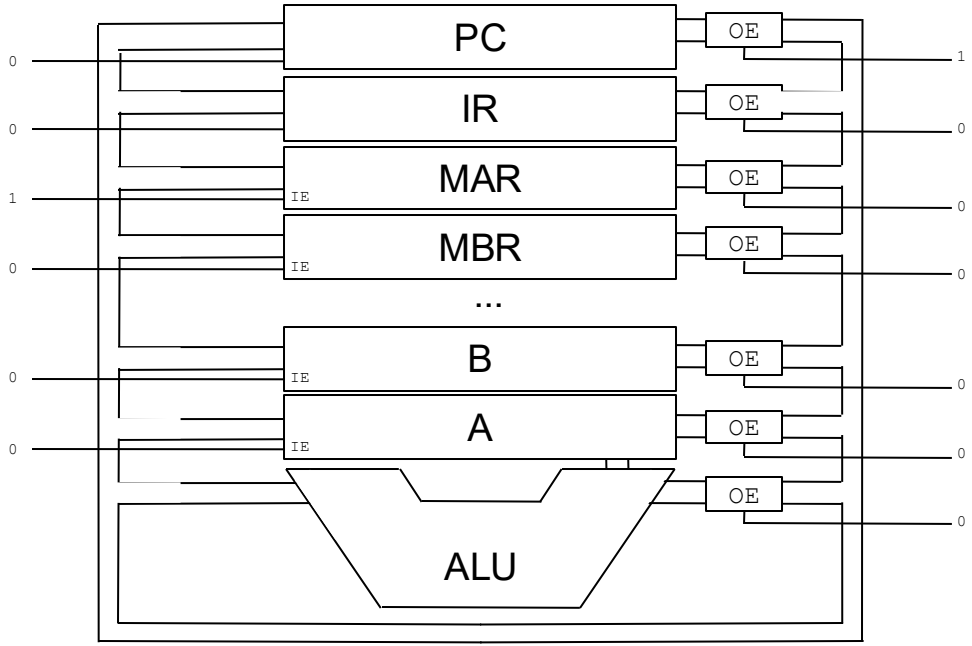
OntarioTech UNIVERSITY

Executing Instructions

- Every instruction needs additional housekeeping:
  - Before: The instruction needs to be fetched from memory
  - After: The program counter needs to be updated

Instruction Fetch

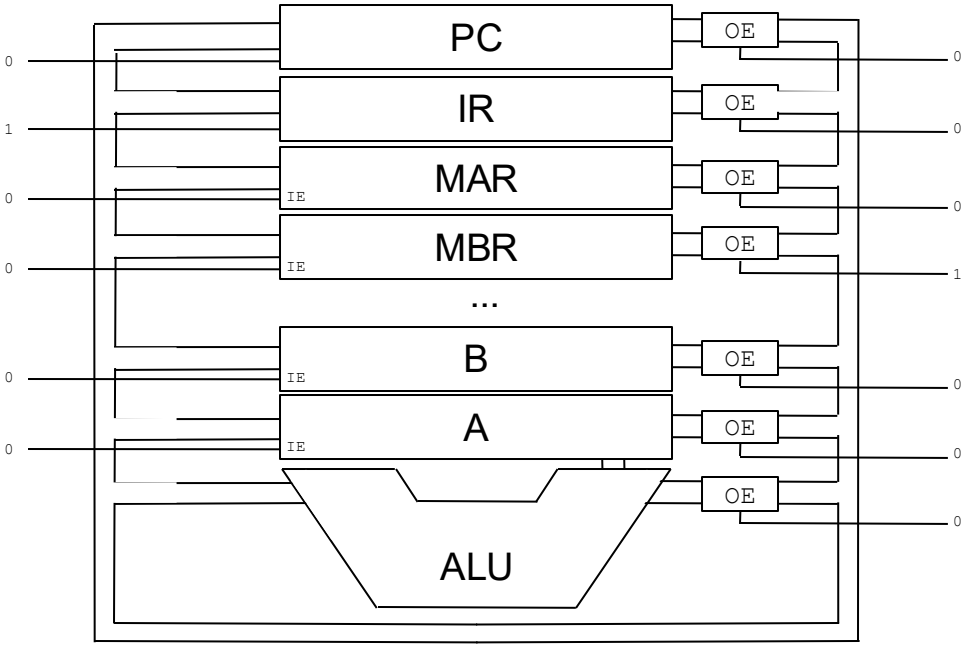
- Instruction fetch:



Cycle 1

Instruction Fetch

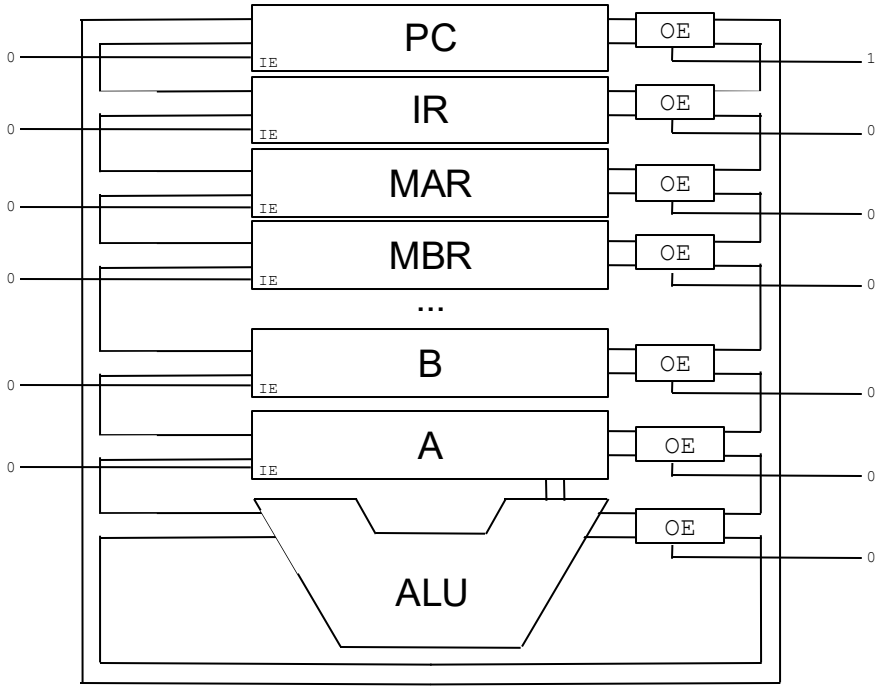
- Instruction fetch:



Cycle 2

Increment Program Counter

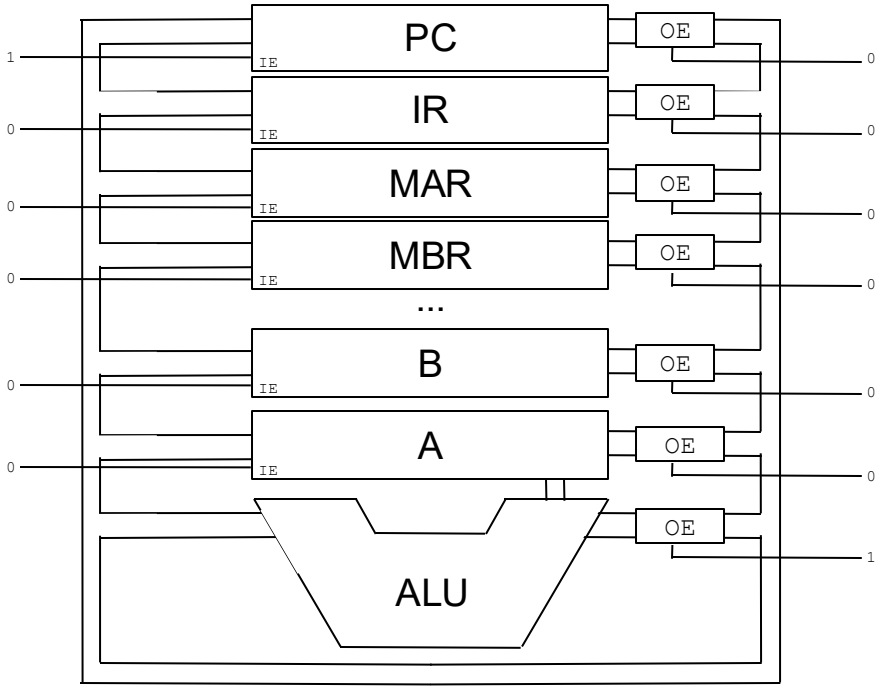
- Increment program counter:



Cycle 1

Increment Program Counter

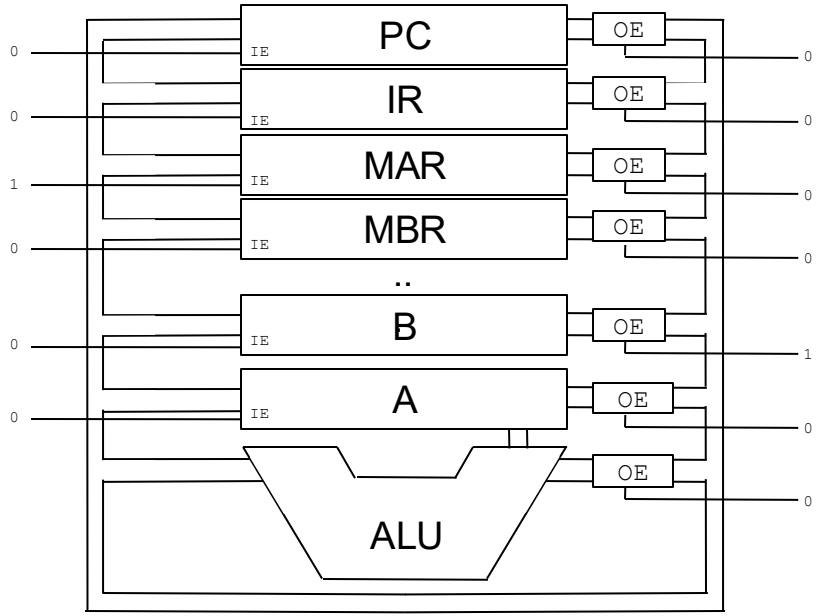
- Increment program counter:



Cycle 2

Memory Load

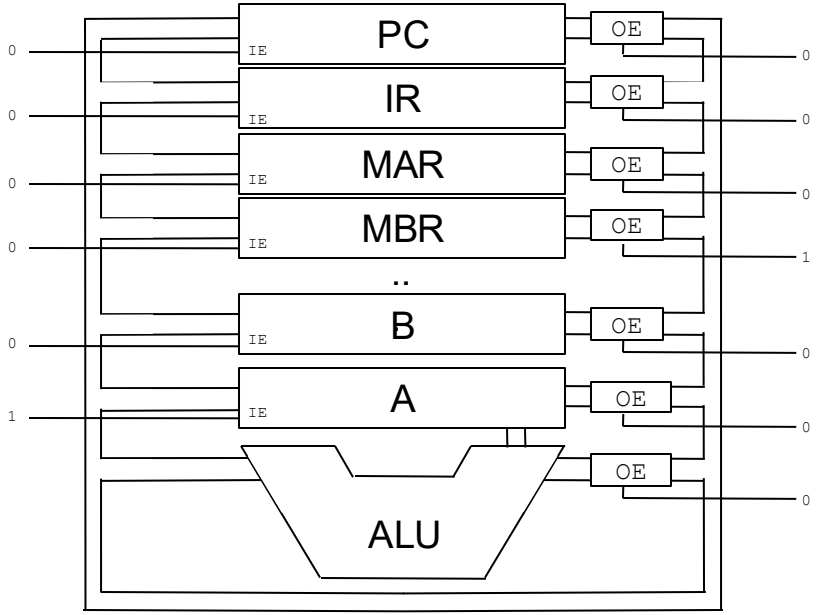
- In a RISC processor, memory loads are usually limited, e.g.:
  - `LOAD B`
    - Load the value into A from the memory address found in B



Cycle 1

Memory Load

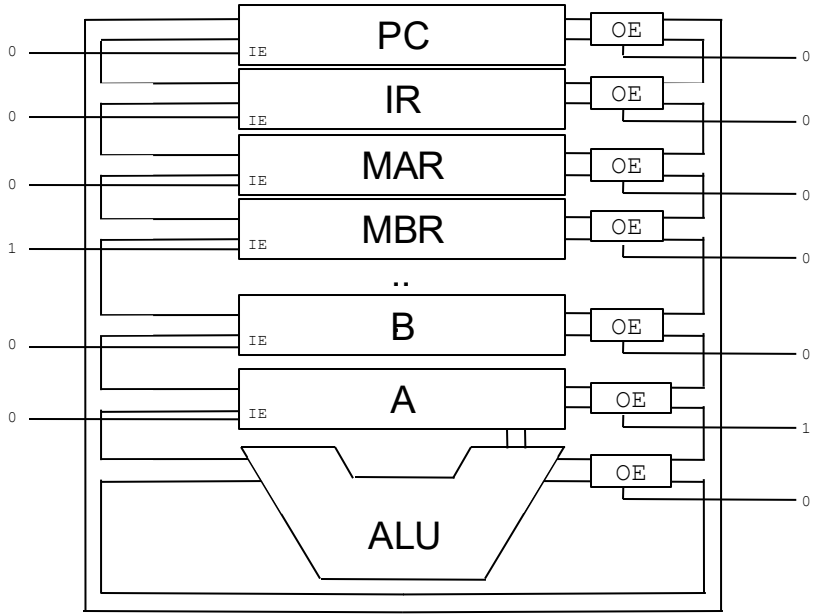
- In a RISC processor, memory loads are usually limited, e.g.:
  - `LOAD B`
    - Load the value into A from the memory address found in B



Cycle 2

Memory Store

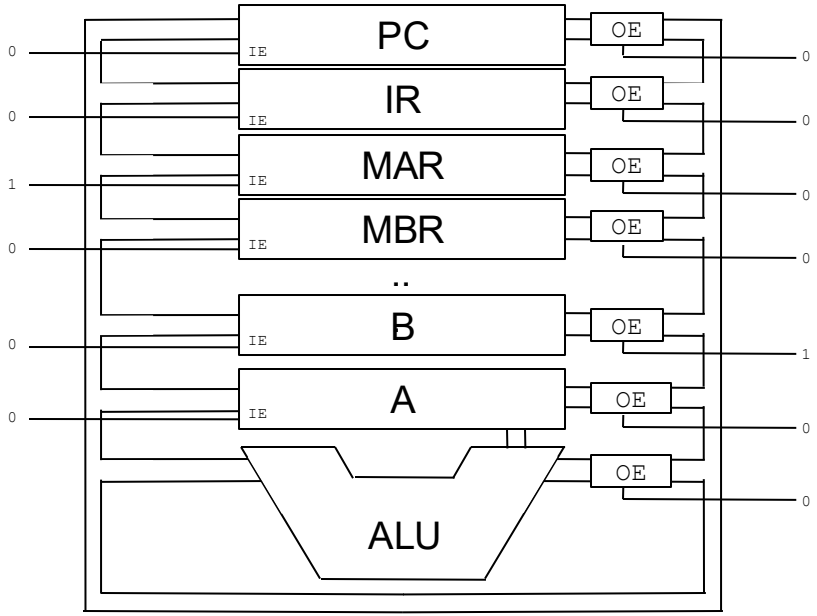
- In a RISC processor, memory stores are also usually limited, e.g.:
  - `STORE B`
    - Store the value in A into the memory address found in B



Cycle 1

Memory Store

- In a RISC processor, memory stores are also usually limited, e.g.:
  - `STORE B`
    - Store the value in A into the memory address found in B



Cycle 2

Wrap-Up

- Instruction cycle
- Additional digital circuit components
- Data path/bus
- Fetch

What is next?

- Decode
- Execute
  - Register transfer language
  - Example program execution