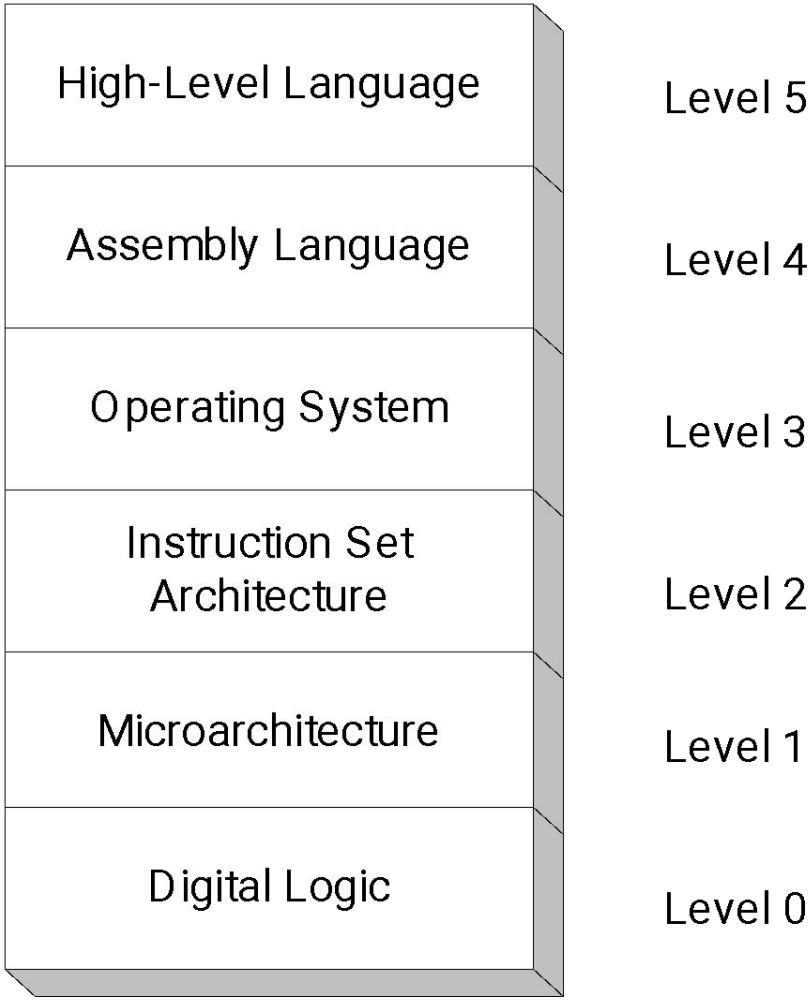
IA-32 Architecture

Computer Organization and Assembly Languages

with slides by Kip Irvine and Keith Van Rhein

Virtual machines

Abstractions for computers

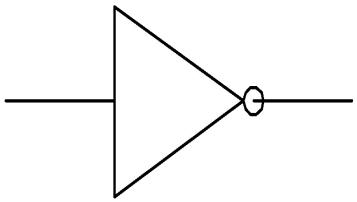


NOT

- Inverts (reverses) a boolean value
- Truth table for Boolean NOT operator:

X	¬X
F	T
T	F

Digital gate diagram for NOT:



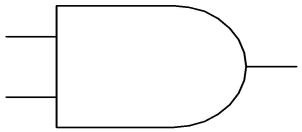
NOT

AND

- Truth if both are true
- Truth table for Boolean AND operator:

X	Y	X ∧ Y
F	F	F
F	T	F
T	F	F
T	T	T

Digital gate diagram for AND:



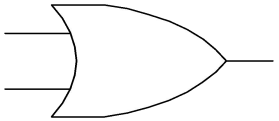
AND

OR

- True if either is true
- Truth table for Boolean OR operator:

X	Y	X ∨ Y
F	F	F
F	T	T
T	F	T
T	T	T

Digital gate diagram for OR:



OR

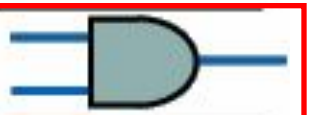
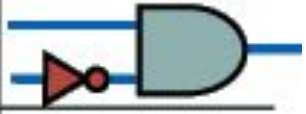
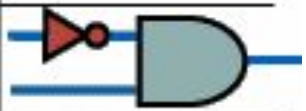
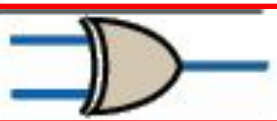
Truth tables

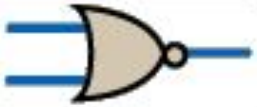
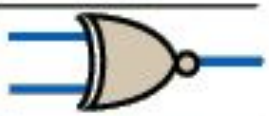
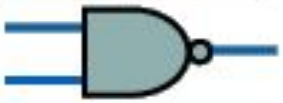
- A Boolean function has one or more Boolean inputs, and returns a single Boolean output.
- A truth table shows all the inputs and outputs of a Boolean function

Example: $\neg X \vee Y$

X	¬X	Y	¬X ∨ Y
F	T	F	T
F	T	T	T
T	F	F	F
T	F	T	T

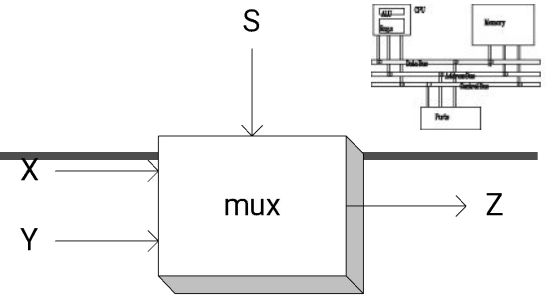
All possible 2-input Boolean functions

0 0 0 0	0	
0 0 0 1	AND	
0 0 1 0	xy'	
0 0 1 1	x	
0 1 0 0	x'y	
0 1 0 1	y	
0 1 1 0	XOR	
0 1 1 1	OR	

1 0 0 0	NOR	
1 0 0 1	XNOR	
1 0 1 0	y'	
1 0 1 1	x + y'	
1 1 0 0	x'	
1 1 0 1	x'+y	
1 1 1 0	NAND	
1 1 1 1	1	

Truth tables

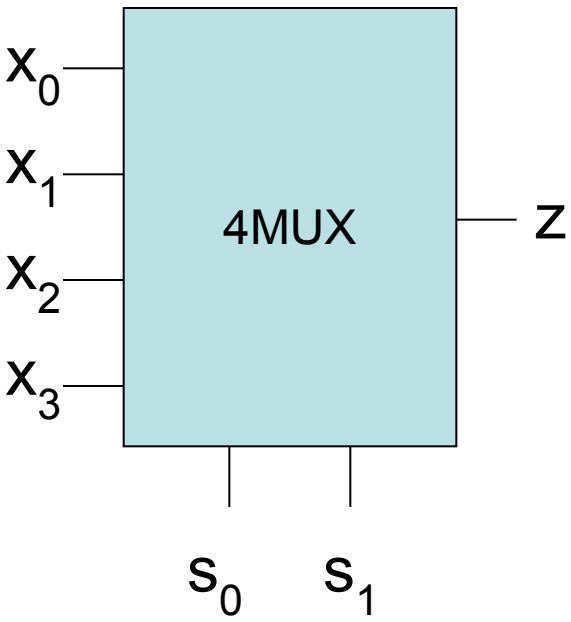
- Example: $(Y \wedge S) \vee (X \wedge \neg S)$

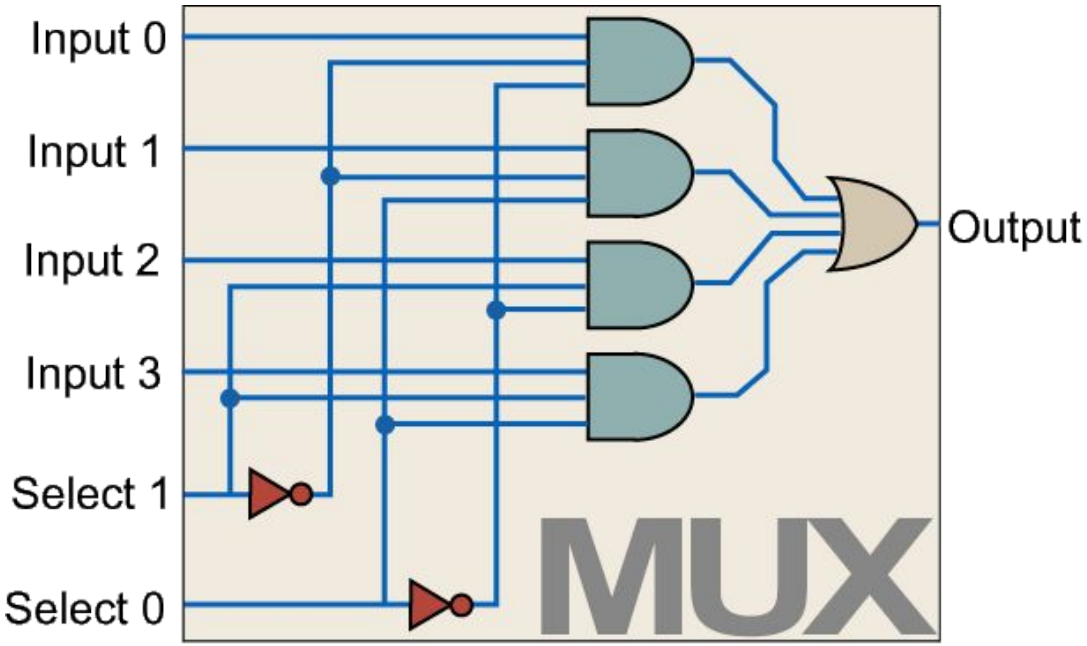


Two-input multiplexer

X	Y	S	$Y \wedge S$	$\neg S$	$X \wedge \neg S$	$(Y \wedge S) \vee (X \wedge \neg S)$
F	F	F	F	T	F	F
F	T	F	F	T	F	F
T	F	F	F	T	T	T
T	T	F	F	T	T	T
F	F	T	F	F	F	F
F	T	T	T	F	F	T
T	F	T	F	F	F	F
T	T	T	T	F	F	T

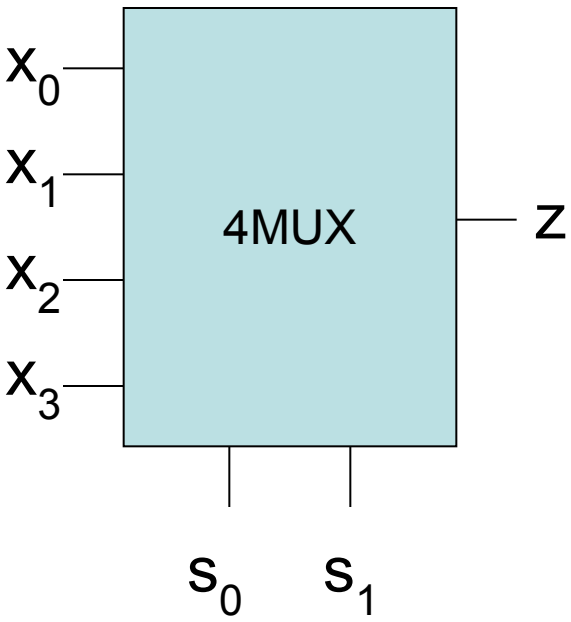
4-multiplexer

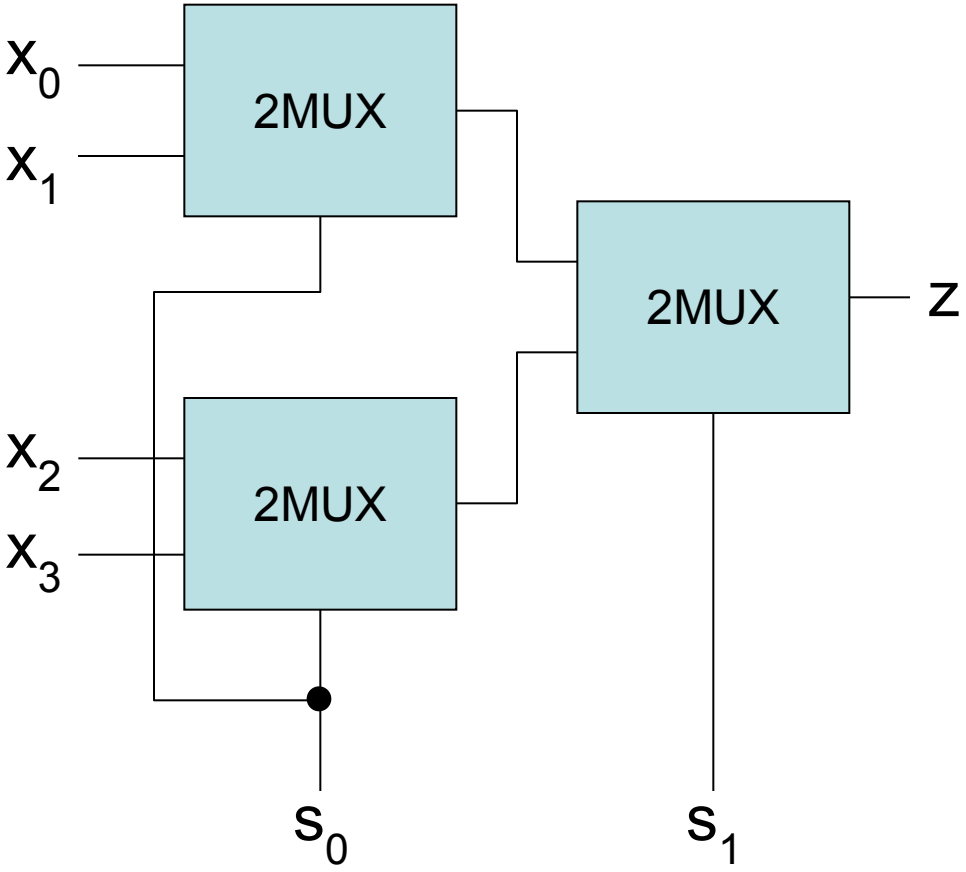




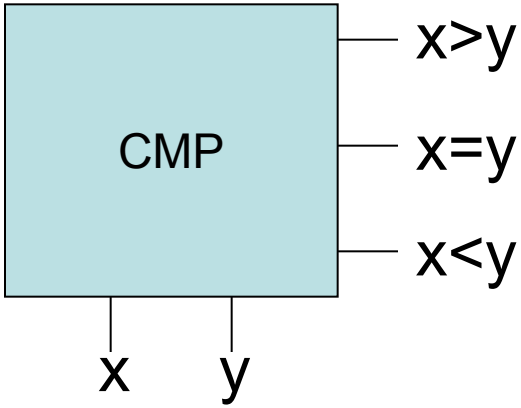
This boolean function describes a multiplexer, a digital component that uses a selector bit (S) to select one of two outputs (X or Y). If S = false, the function output (Z) is the same as X. If S = true, the function output is the same as Y

4-multiplexer



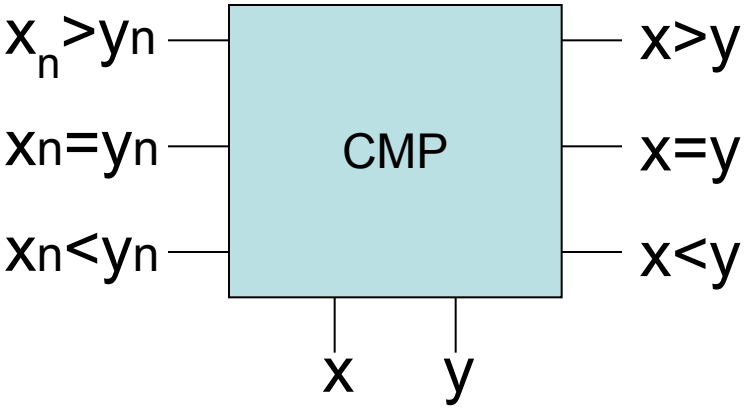


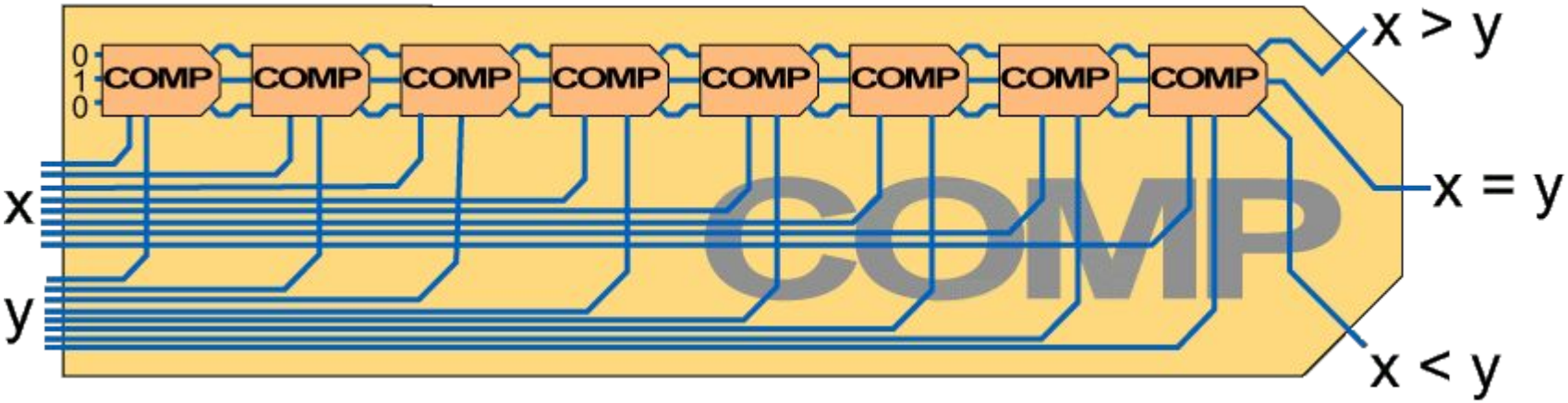
Comparator



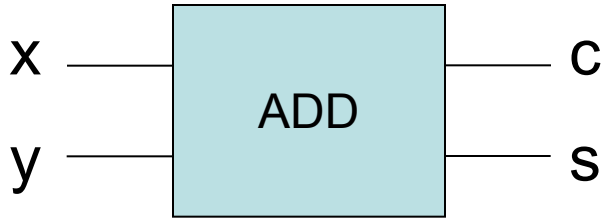
x	y	x>y	x=y	x<y

8-bit comparator

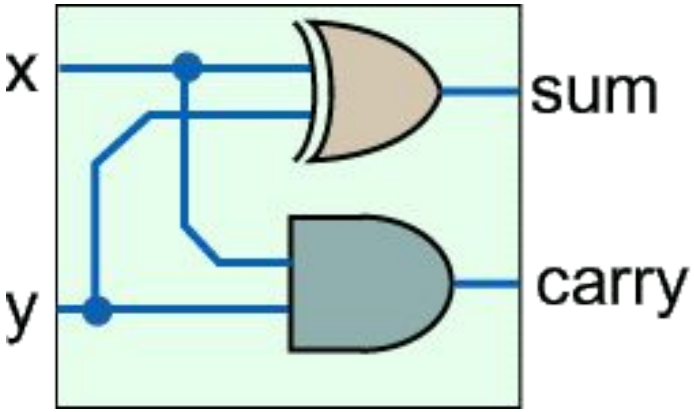




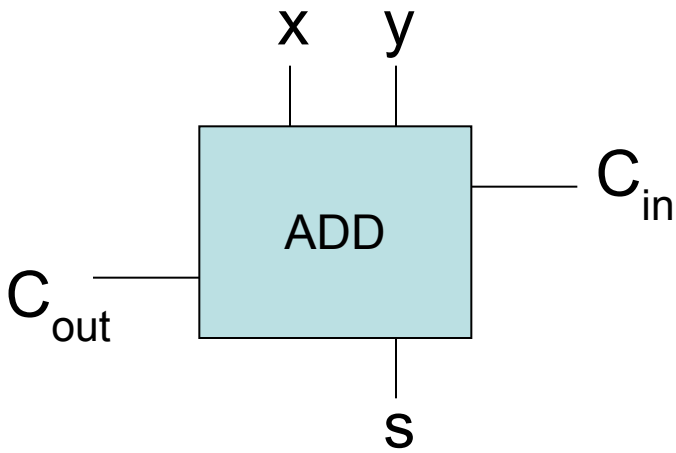
1-bit half adder

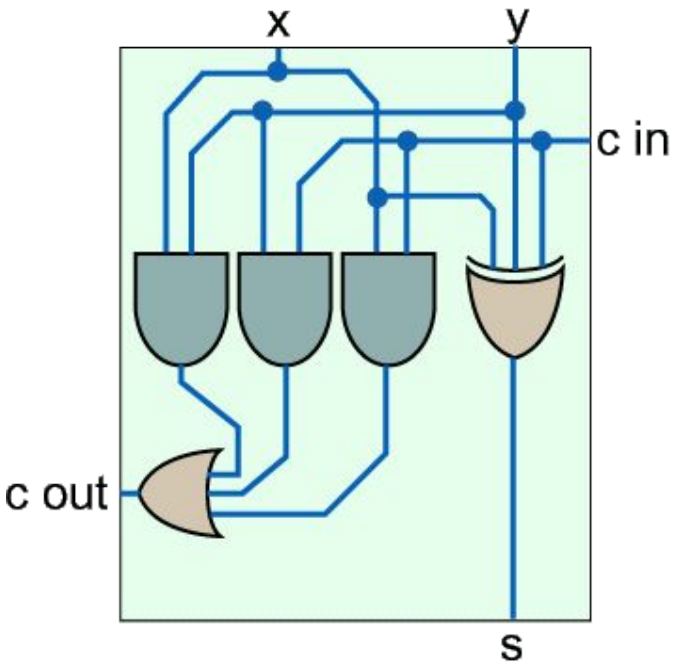


x	y	s	c



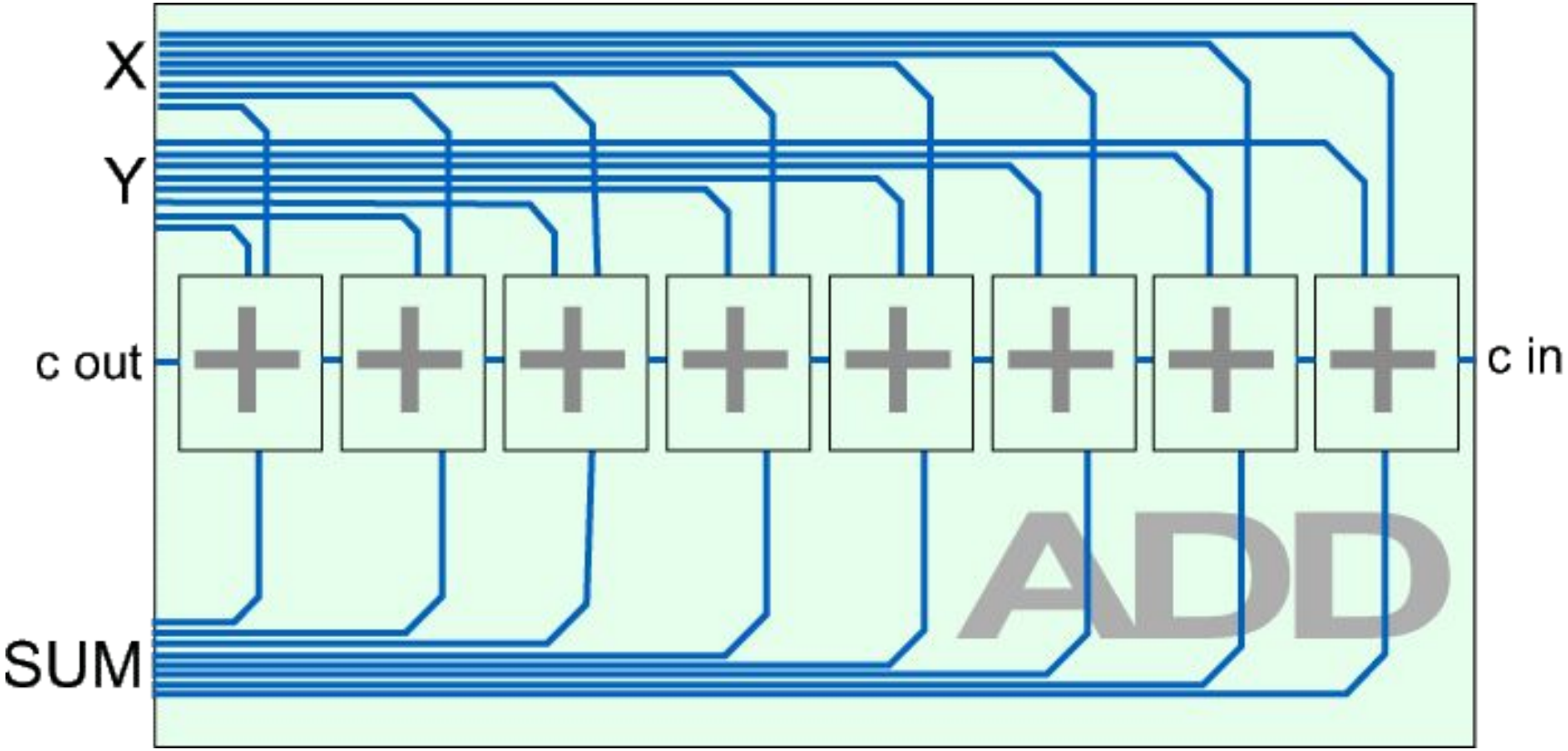
1-bit full adder



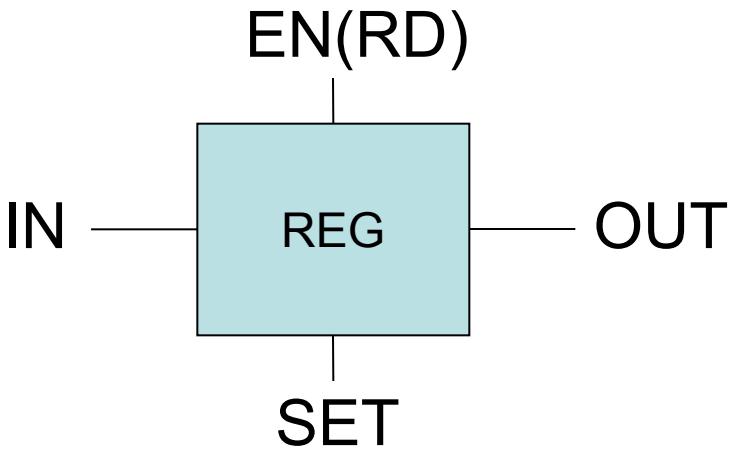


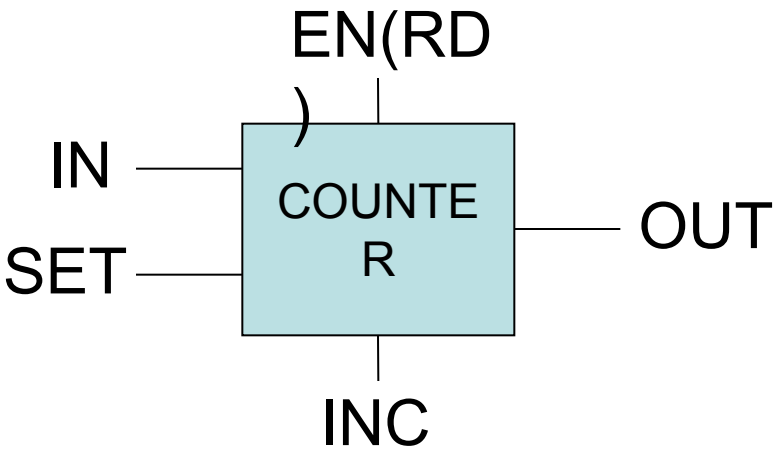
x	y	C_{in}	C_{out}	s

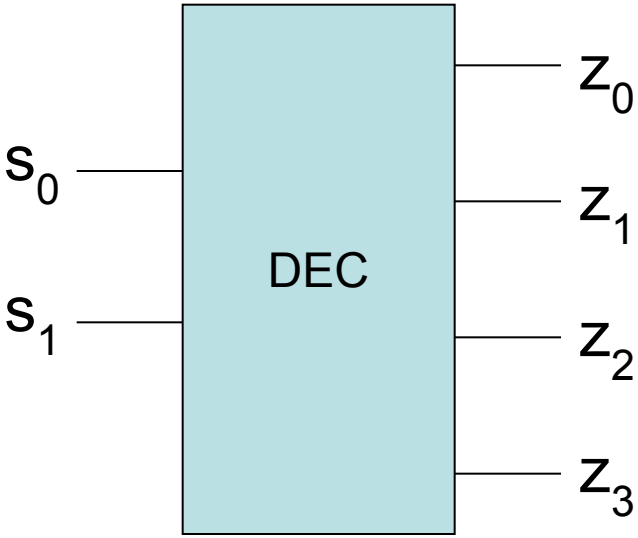
8-bit adder



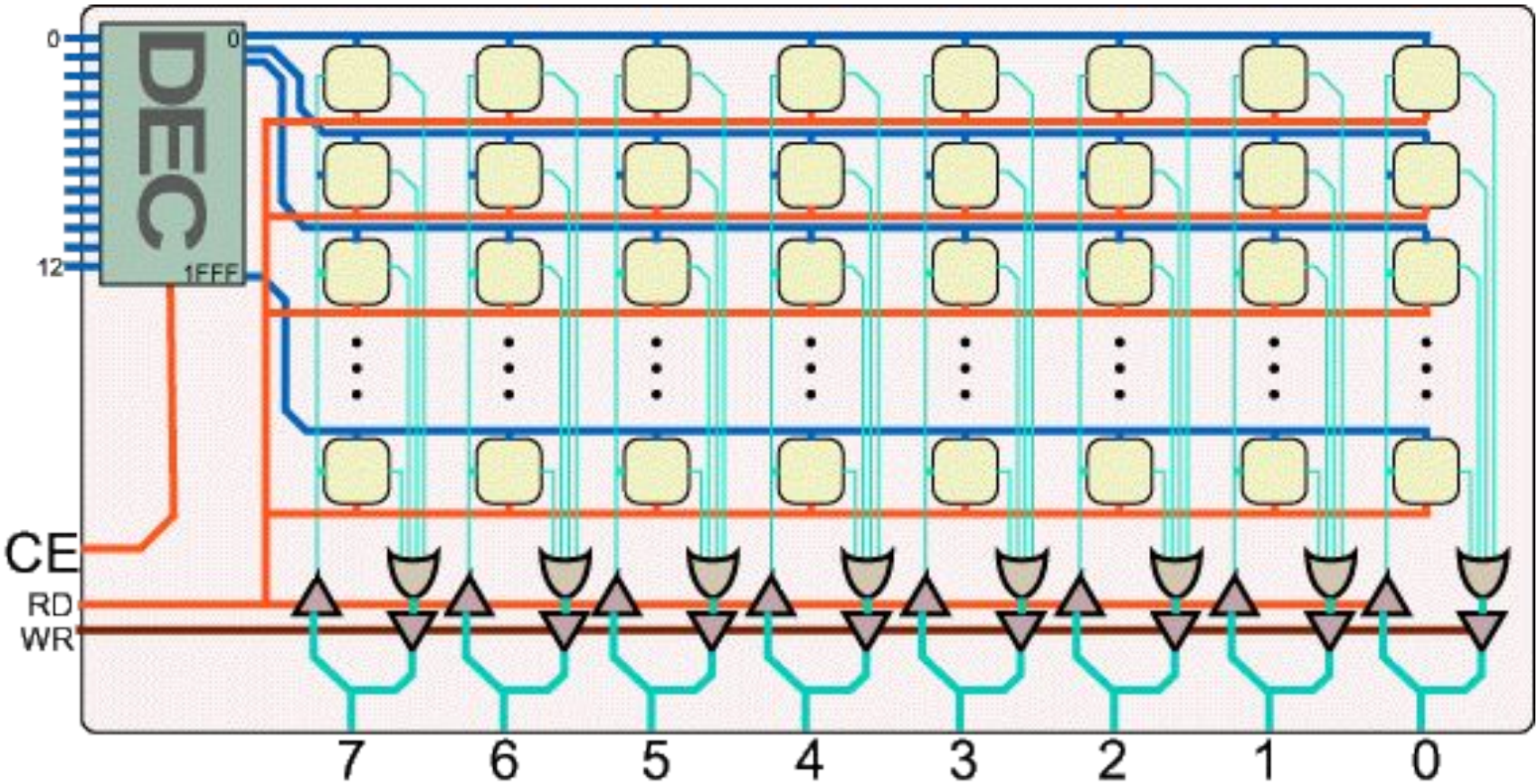
Registers and counters







Memory

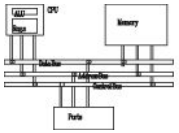


8K 8-bit memory

Microcomputer concept

A computer system usually contains four bus types: data, I/O, control, and address.
The data bus transfers instructions and data between the CPU and memory.
The I/O bus transfers data between the CPU and the system input/output devices.
The control bus uses binary signals to synchronize actions of all devices attached to the system bus.
The address bus holds the addresses of instructions and data when the currently executing instruction transfers data between the CPU and memory.

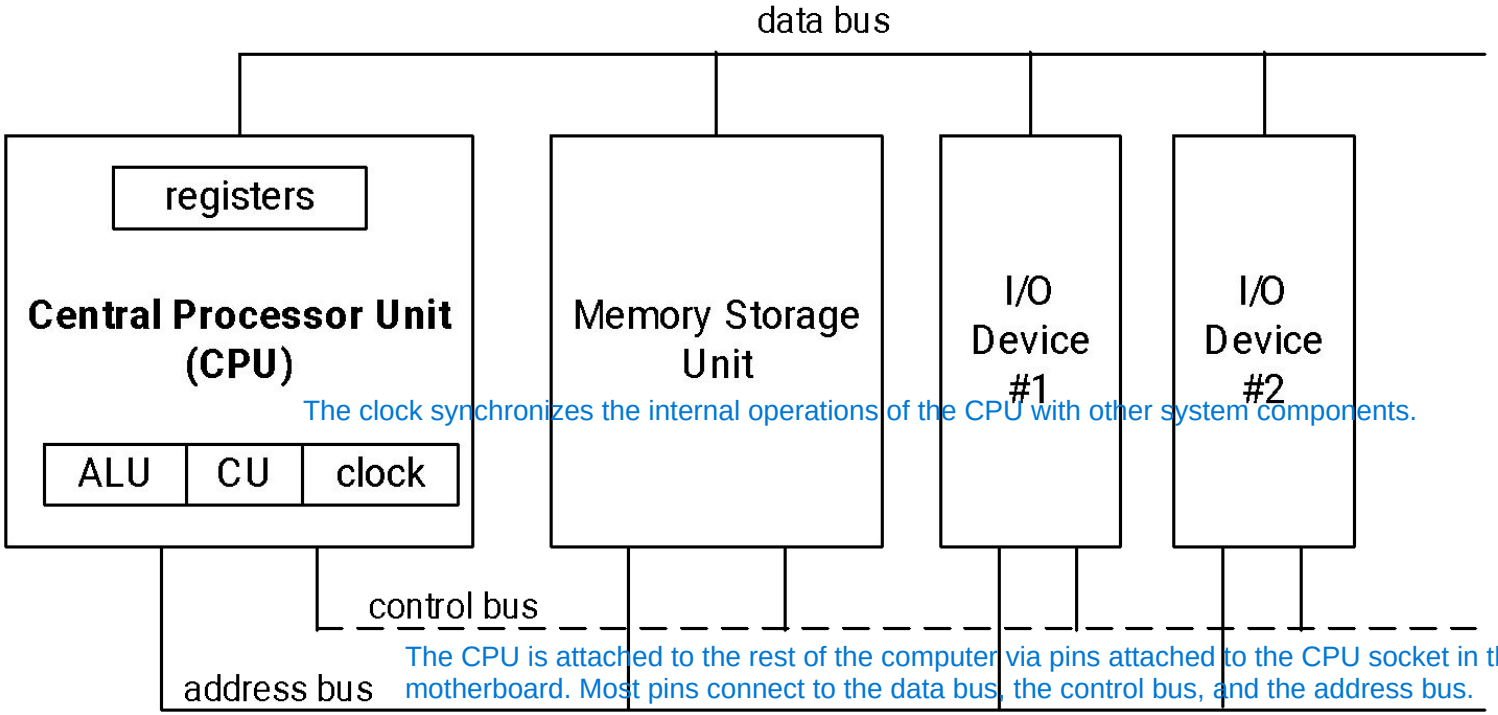
Basic microcomputer design



- clock synchronizes CPU operations
- control unit (CU) coordinates sequence of execution steps

The central processor unit (CPU), where calculations and logic operations take place, contains a limited number of storage locations named registers, a high-frequency clock, a control unit, and an arithmetic logic unit.

- ALU performs arithmetic and logic operations

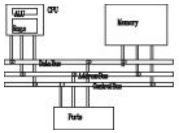


The clock synchronizes the internal operations of the CPU with other system components.

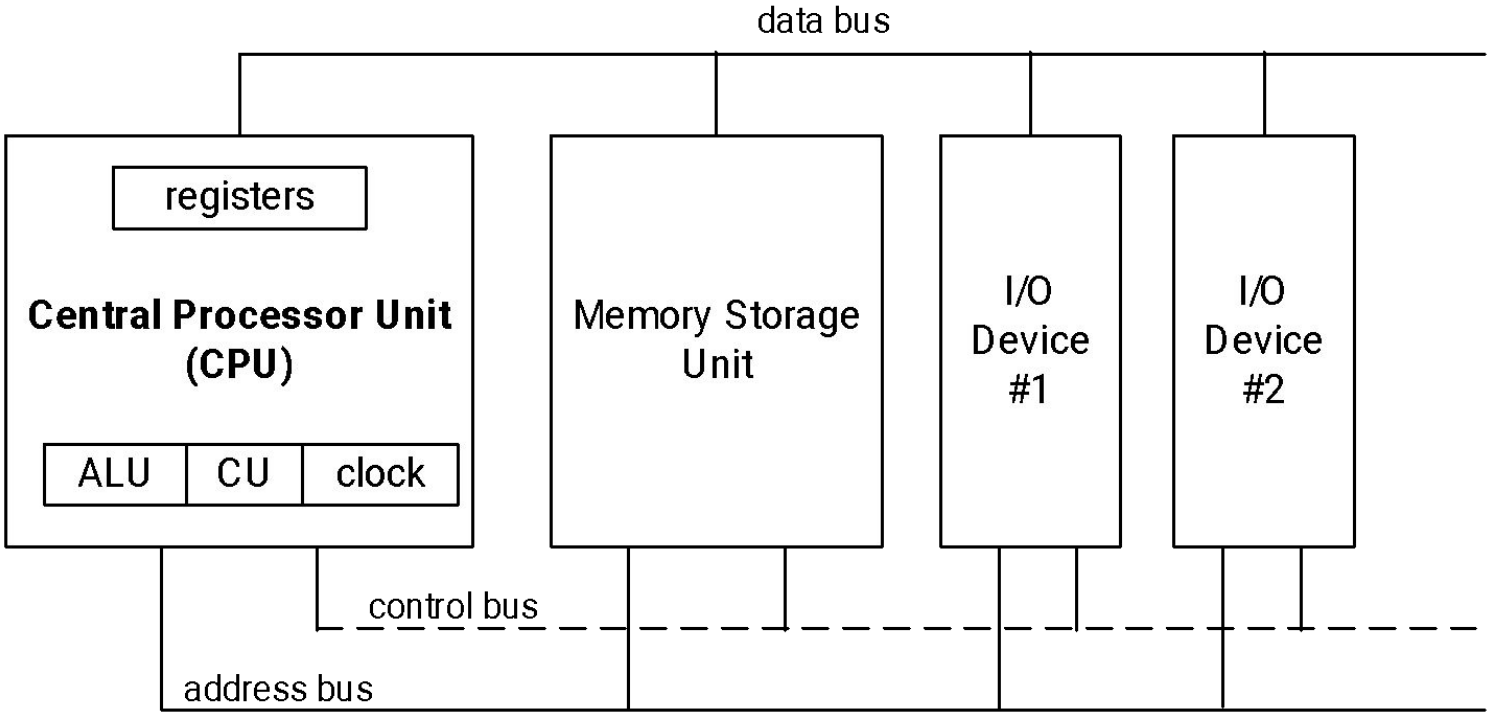
The CPU is attached to the rest of the computer via pins attached to the CPU socket in the computer's motherboard. Most pins connect to the data bus, the control bus, and the address bus.

The control unit (CU) coordinates the sequencing of steps involved in executing machine instructions.

Basic microcomputer design



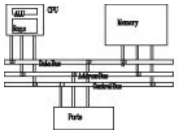
- The memory storage unit holds instructions and data for a running program
- A bus is a group of wires that transfer data from one part to another (data, address, control)



The memory storage unit is where instructions and data are held while a computer program is running. The storage unit receives requests for data from the CPU,

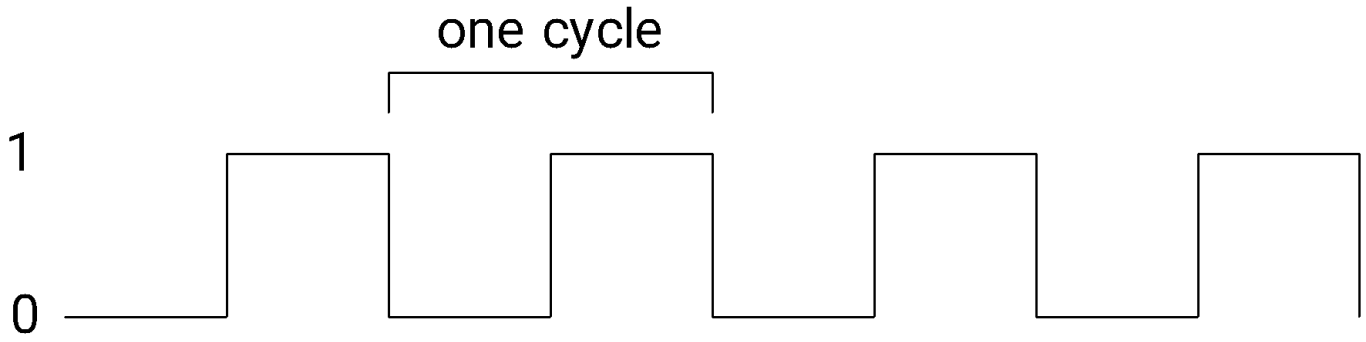
transfers data from random access memory (RAM) to the CPU,
and transfers data from the CPU into memory.

Clock



The clock synchronizes the internal operations of the CPU with other system components.

- synchronizes all CPU and BUS operations
- machine (clock) cycle measures time of a single operation Clock cycle: The basic unit of time for machine instructions is a clock cycle.
- clock is used to trigger events

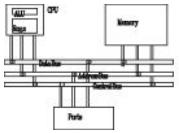


- Basic unit of time, 1GHz→clock cycle=1ns
- A instruction could take multiple cycles to complete, e.g. multiply in 8088 takes 50 cycles

A machine instruction requires at least one clock cycle to execute, and a few require in excess of 50 clocks (the multiply instruction on the 8088 processor).

Execute: The ALU executes the instruction using the named registers and internal registers as operands and sends the output to named registers and/or memory. The ALU updates status flags providing information about the processor state.

Instruction execution cycle

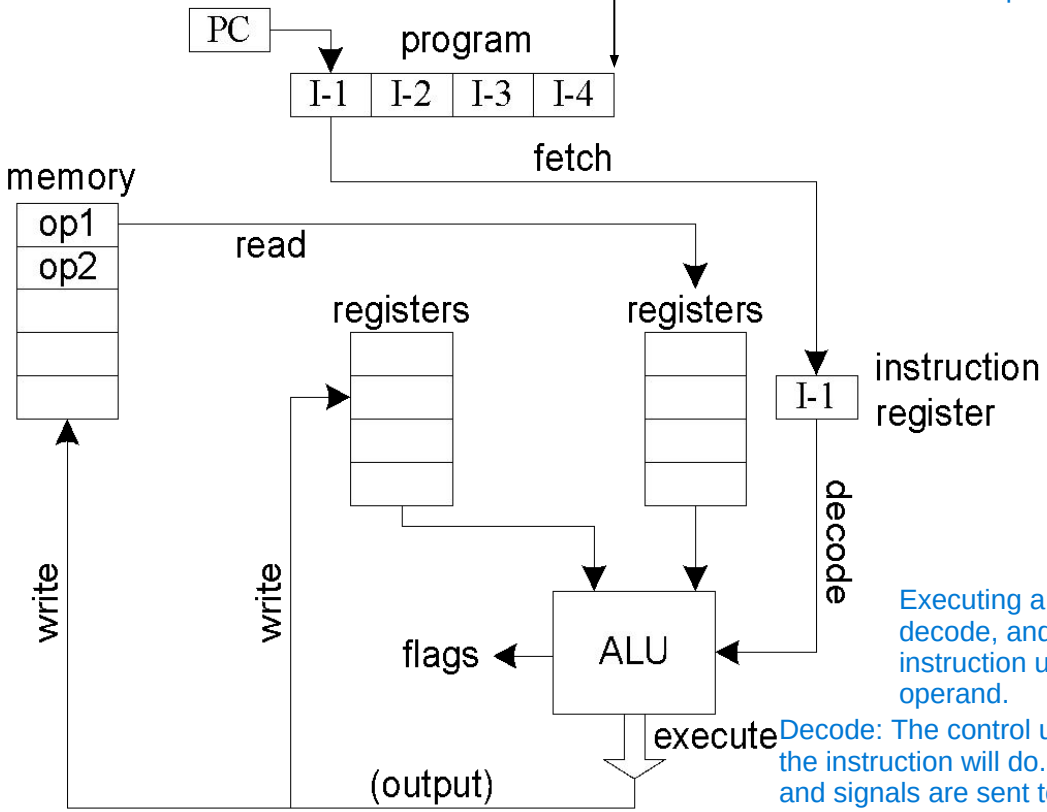


Store output operand: If the output operand is in memory, the control unit uses a write operation to store the data.

program counter The instruction pointer(Program counter) contains the address of the next instruction.

instruction queue The instruction queue holds a group of instructions about to be executed.

Fetch: The control unit fetches the next instruction from the instruction queue and increments the instruction pointer (IP). The IP is also known as the program counter.



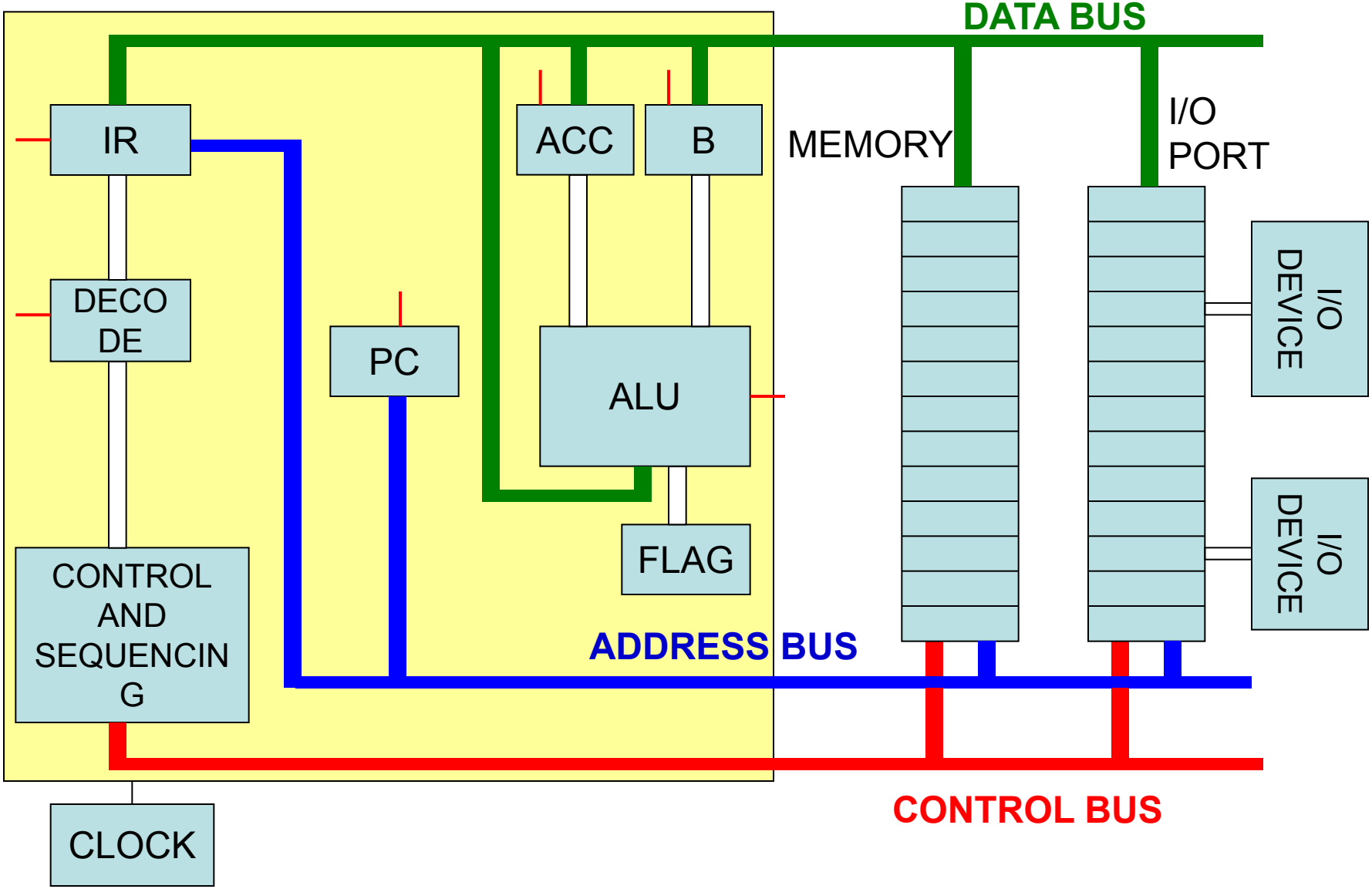
- **Fetch**
- **Decode**
- Fetch operands
- **Execute**
- Store output

Executing a machine instruction requires three basic steps: fetch, decode, and execute. Two more steps are required when the instruction uses a memory operand: fetch operand and store output operand.

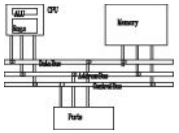
Decode: The control unit decodes the instruction's function to determine what the instruction will do. The instruction's input operands are passed to the ALU, and signals are sent to the ALU indicating the operation to be performed.

Fetch operands: If the instruction uses an input operand located in memory, the control unit uses a read operation to retrieve the operand and copy it into internal registers. Internal registers are not visible to user programs.

A simple microcomputer



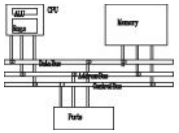
Instruction set



OPCODE	MNEMONIC	OPCODE	MNEMONIC
0	NOP	A	CMP
1	LDA	B	JG
2	STA	C	JE
3	ADD	D	JL
4	SUB		
5	IN		
6	OUT		
7	JMP		
8	JN		
9	HLT		

OPCODE	OPERAND
4	12

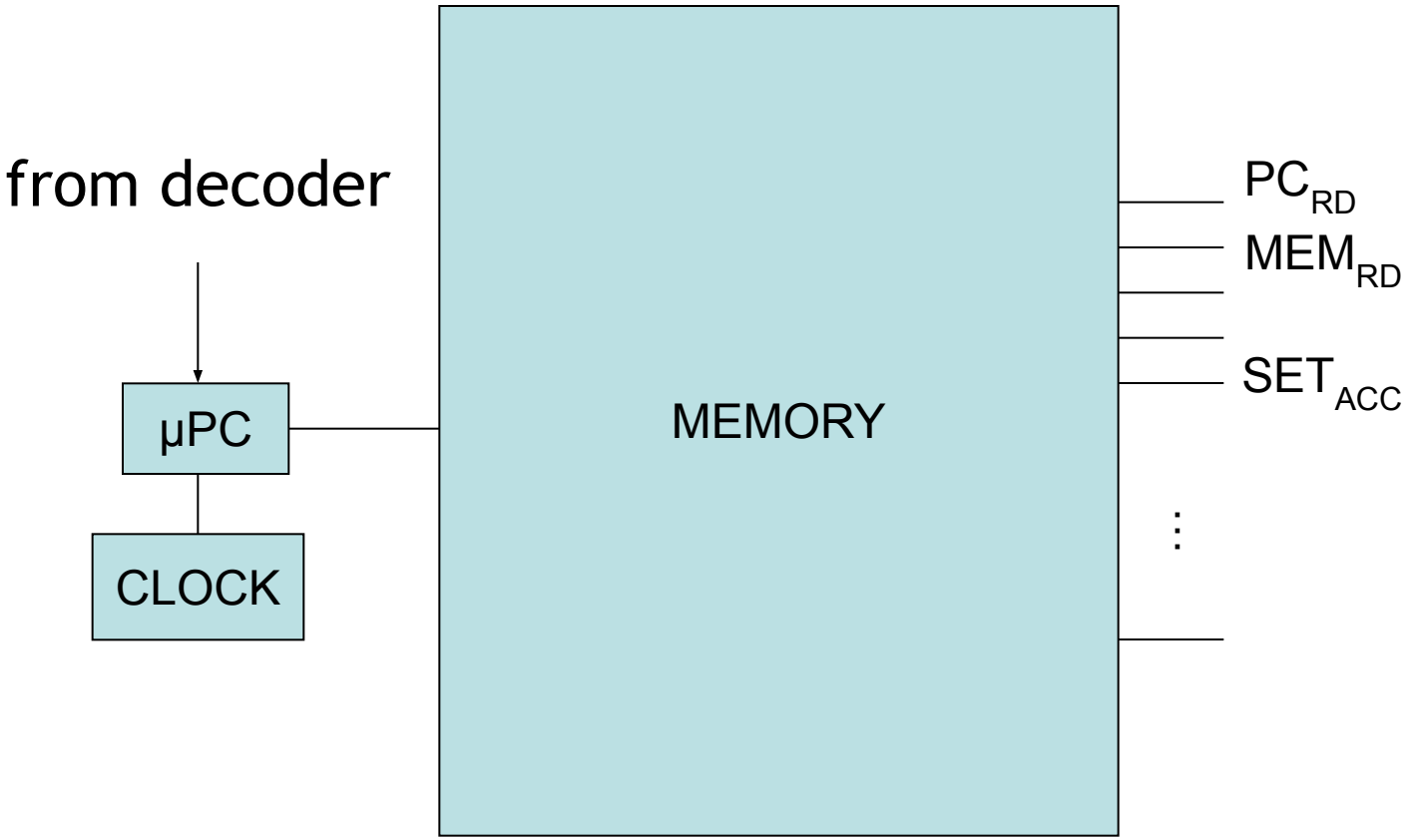
Control bus



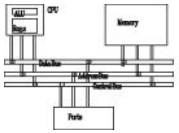
- A series of control signals to control all components such as registers and ALU
- Control signal for load ACC:

  SET_{ACC}=1, others=0

Control and sequencing unit



Control and sequencing unit



		PC_{RD}	MEM_{RD}	MEM_{WT}	IR_{SET}	….
fetch	0000	1	0	0	0	0….
	0001	0	1	0	0	
	0002	0	0	0	1	
decode	0003	4-bit IR RD				
	0004	DECODER RD, μPC SET				
exec	0005					
fetch						
decode						
	000B					

Decoder