Data Manipulation

National Chiao Tung University

Chun-Jen Tsai

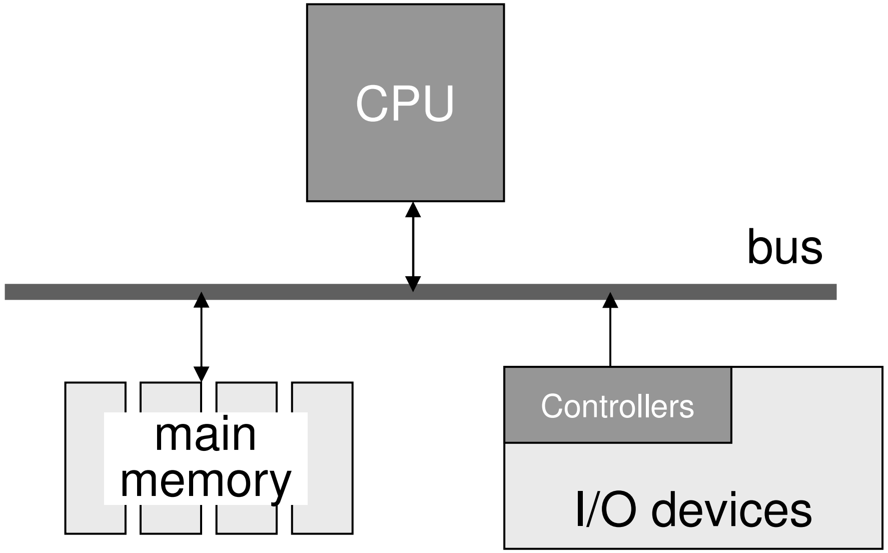
03/09/2012

Computer Architecture

- Central Processing Unit (CPU) contains
  - Arithmetic/Logic Unit (ALU)
  - Control Unit
  - Registers
  - Cache Memory

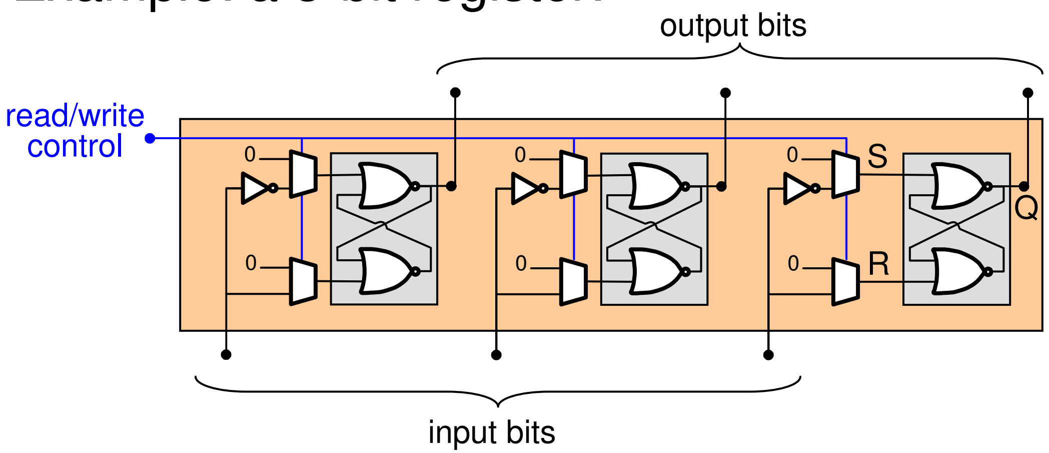
processor core

- Bus
- Main Memory
- I/O devices



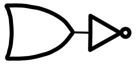
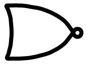
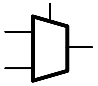
Register

- ❑ A register is a special memory cell inside the CPU that can store an n-bit data
  - For modern CPUs, n is usually 8, 16, 32, or 64
  - Registers are used to store intermediate computation results
- ❑ An n-bit register is composed of a group of n flip-flops
  - Example: a 3-bit register:



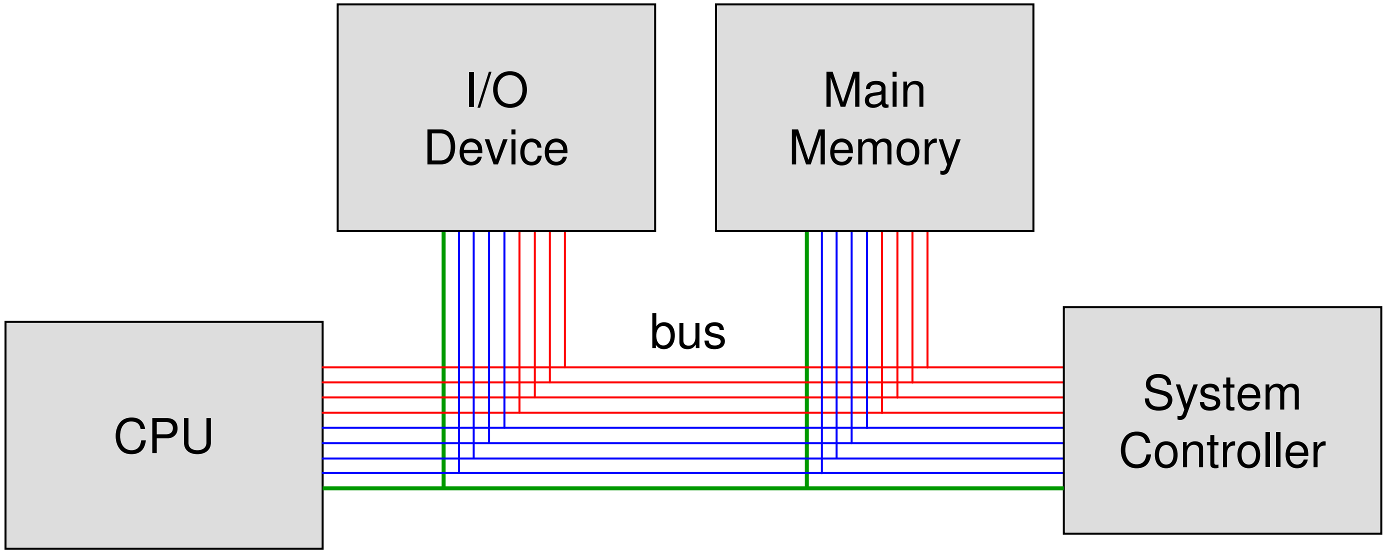
The truth table of the flip-flop is

S	R	Q
0	0	Previous value
1	0	0
0	1	1
1	1	Don’t care

Note:  is denoted as , it is called a NOR gate.  is a 2-to-1 multiplexor.

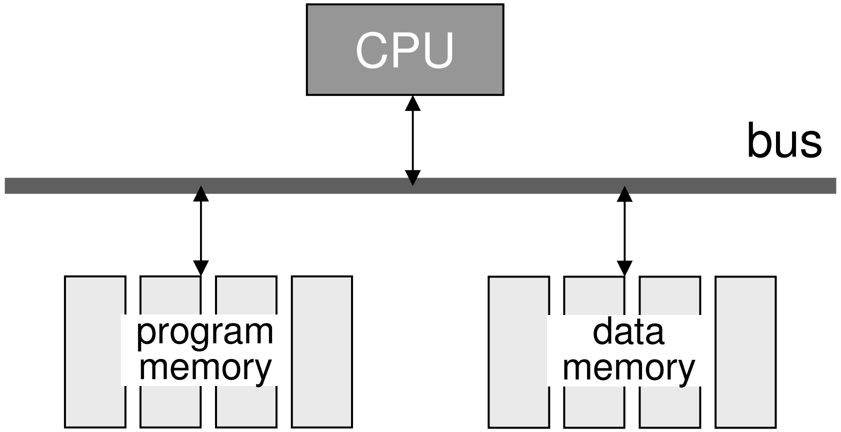
Bus

- ❑ A bus is a collection of wires to connect a computer's CPU, memory, and I/O devices
  - ■ Information (0's and 1's) are transmitted among the CPU, memory, and devices via the bus following some handshaking rules



Stored Program Concept

- ❑ A program is a sequence of ***instructions*** that implements an algorithm
- ❑ A program is just a special type of data and can be stored in main memory
  - ■ Program memory and data memory can be shared or separate; e.g. separate memory architecture:



What Do Instructions Look Like

- When you ask other people to do something, you use human languages (Chinese, English, etc.)
- However, human languages are too cumbersome and too ambiguous for computers to ***decode*** and ***execute***!
- Machine languages must be
  - Concise – easy to decode
  - Precise – each instruction can be executed in only one way

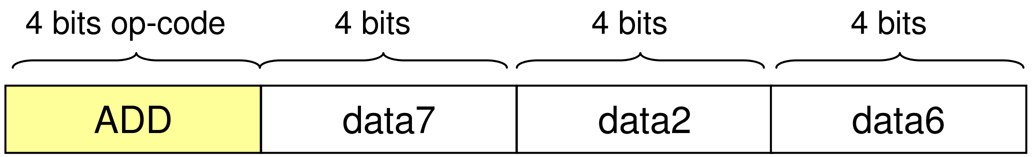
Note: There are computers that can follow instructions given in simple human languages, but it may not be worth doing it this way

Machine Language

- A machine instruction is an instruction coded as a bit pattern directly recognizable by the CPU
- A machine language is the set of all instructions recognizable by a CPU
  - Each CPU has its own machine language, called the Instruction Set Architecture (ISA) of the CPU
- The bit-pattern of a machine instruction can be divided into two parts: op-code field and operand field

Parts of a Machine Instruction

- ❑ Op-code field
  - Specifies which machine operation to execute
  - One per instruction
- ❑ Operand field (a.k.a. addressing mode field)
  - Data and addresses related to this operation
  - Number of operands varies depending on op-code
- ❑ Example: the instruction that asks CPU to add data2 and data6 and store the result in data7 can be coded in a 16-bit number as follows:



4 bits op-code	4 bits	4 bits	4 bits
ADD	data7	data2	data6

Instruction Lengths

- A machine language can use fixed-length coding or variable-length coding of all its instructions
  - Fixed-length coding: easier for the CPU to decode, usually allows fewer instructions in the language
  - Variable-length coding: harder for the CPU to decode, but allows a richer set of instructions

Machine Language and Architecture

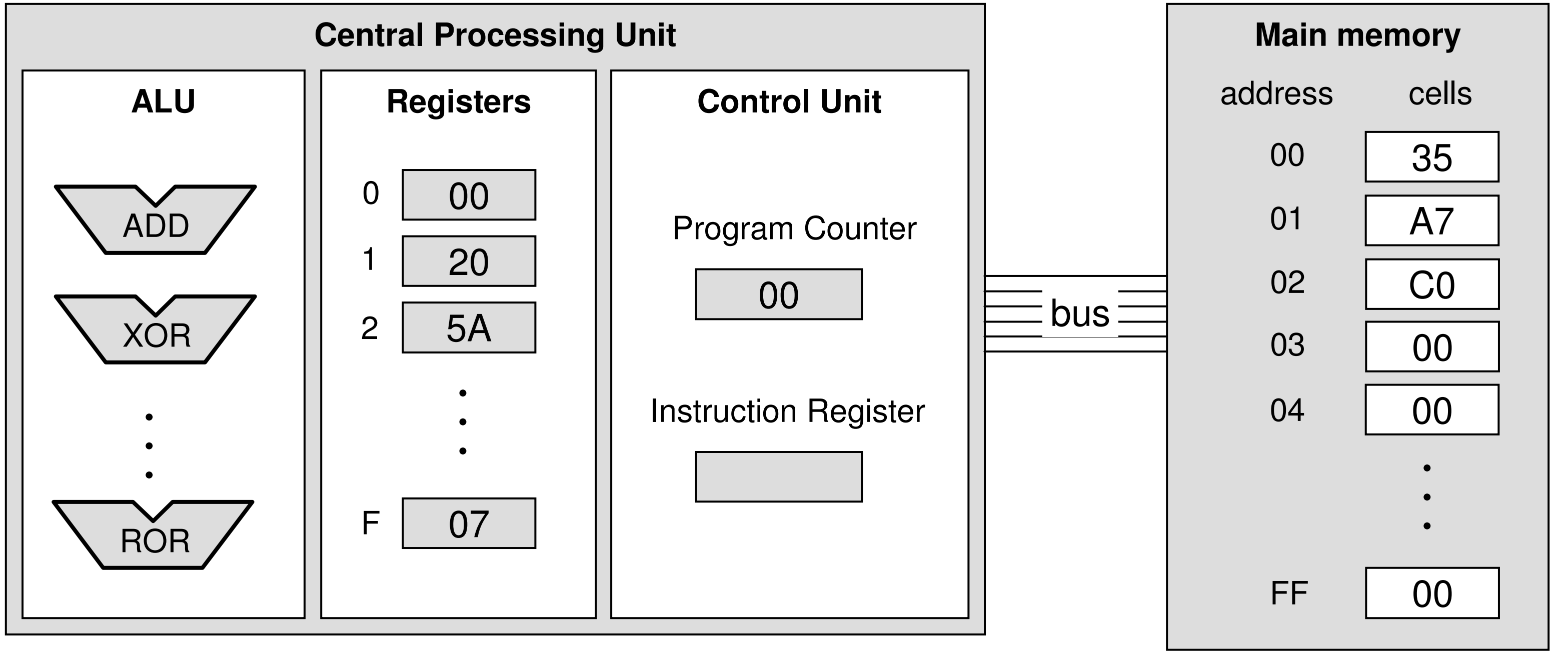
- Machine language reflects the architecture of the CPU, there are two major types of design philosophy: RISC and CISC
- Reduced Instruction Set Computer (RISC)
  - CPU only executes a small set of simple instructions
  - However, CPU executes these simple instructions really fast
  - Usually use fixed-length coding of instructions
- Complex Instruction Set Computer (CISC)
  - CPU executes many complex and powerful instructions
  - Complex instructions takes longer to execute, but an algorithm implemented in CISC machine language requires less instructions than that of RISC's
  - Usually use variable-length coding of instructions

Machine Instruction Types

- The instructions of a machine language can be classified into three groups:
  - Data Transfer: copy data between CPU registers and/or main memory cells
  - Arithmetic/Logic: use existing data values (stored in registers or main memory cells) to compute a new data value
  - Control: direct the execution flow of the program

Example: A Simple CPU

❑ The textbook describes a simple CPU architecture



Central Processing Unit

ALU

ADD

XOR

⋮

ROR

Registers

0	00
1	20
2	5A
⋮	
F	07

Control Unit

Program Counter

00

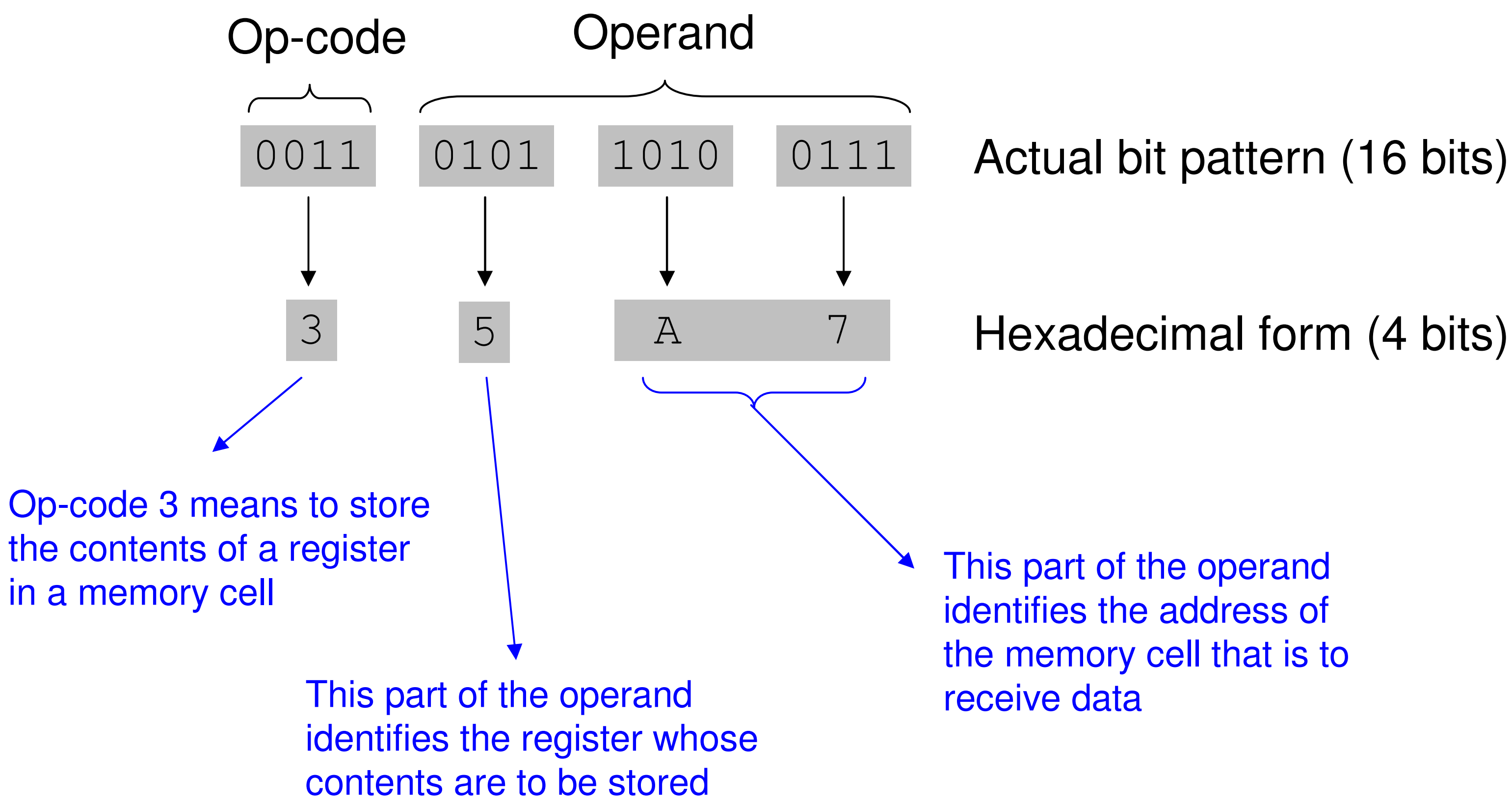
Instruction Register

bus

Main memory

address	cells
00	35
01	A7
02	C0
03	00
04	00
⋮	
FF	00

Example: An Instruction



Op-code

Operand

0011 0101 1010 0111

Actual bit pattern (16 bits)

3 5 A 7

Hexadecimal form (4 bits)

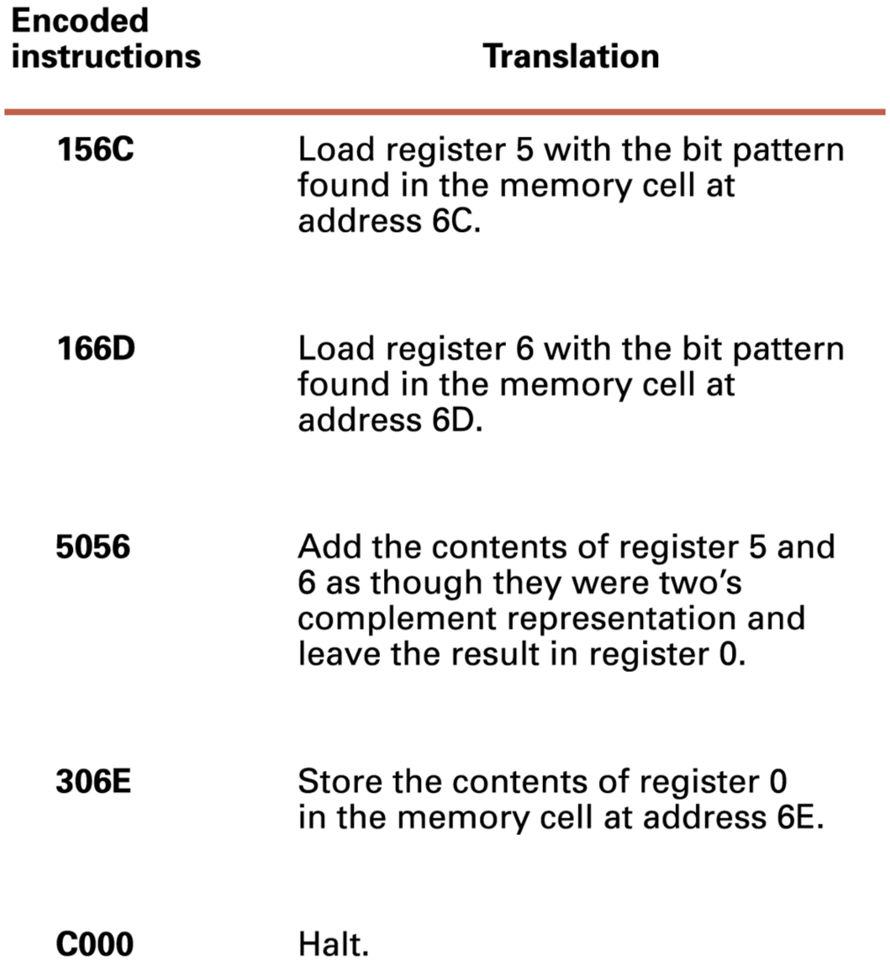
Op-code 3 means to store the contents of a register in a memory cell

This part of the operand identifies the register whose contents are to be stored

This part of the operand identifies the address of the memory cell that is to receive data

Note: The complete instruction set is listed in Appendix C of the textbook

Example: Program in Machine Codes



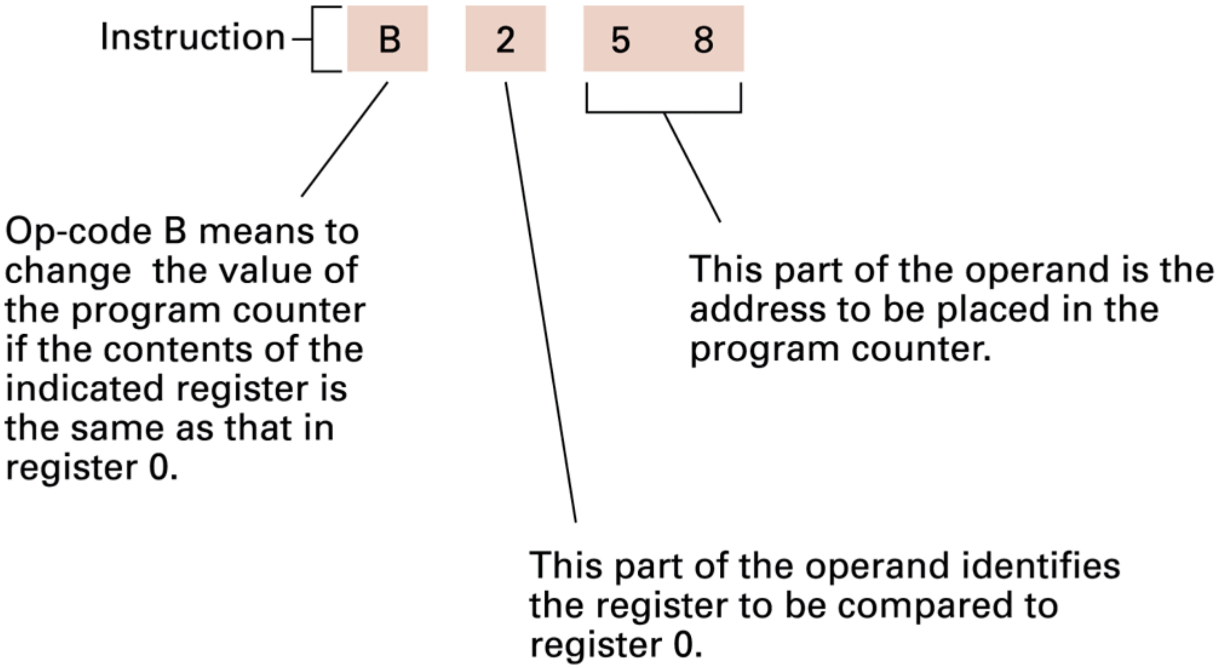
Encoded instructions	Translation
156C	Load register 5 with the bit pattern found in the memory cell at address 6C.
166D	Load register 6 with the bit pattern found in the memory cell at address 6D.
5056	Add the contents of register 5 and 6 as though they were two's complement representation and leave the result in register 0.
306E	Store the contents of register 0 in the memory cell at address 6E.
C000	Halt.

Program Execution

- Controlled by two special-purpose registers
  - Program counter: address of next instruction
  - Instruction register: current instruction
- Each machine cycle of program execution is composed of three steps:
  - *Fetch* – copies memory cells addressed by the program counter to the instruction register and increment the program counter to the next instruction in main memory
  - *Decode* – decodes the bit pattern in the instruction register to determine the operation required and the related operands
  - *Execute* – performs the operation specified by the instruction

Program Flow Control

- Some instructions change the next instruction to be fetched based on some condition
  - Example: conditional jump instruction

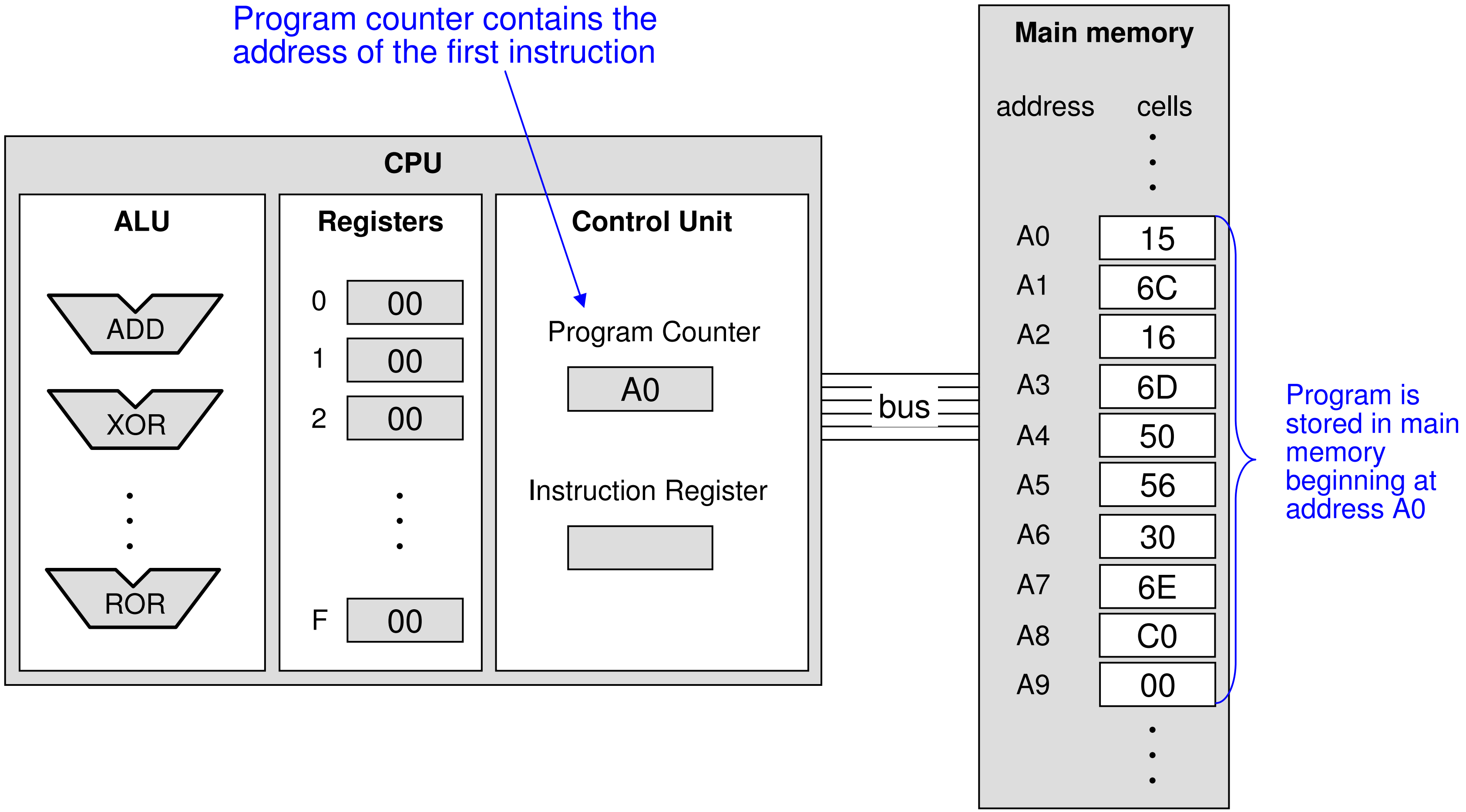


Op-code B means to change the value of the program counter if the contents of the indicated register is the same as that in register 0.

This part of the operand is the address to be placed in the program counter.

This part of the operand identifies the register to be compared to register 0.

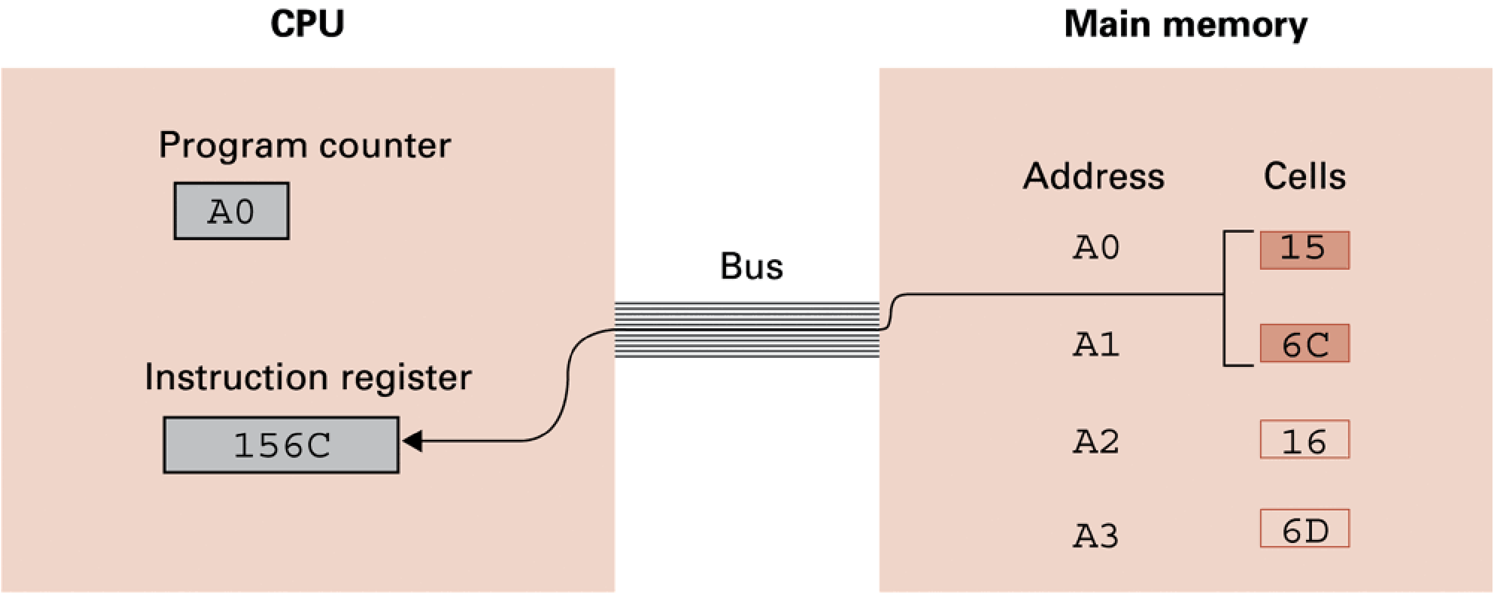
Running a Program in Main Memory



Program counter contains the address of the first instruction

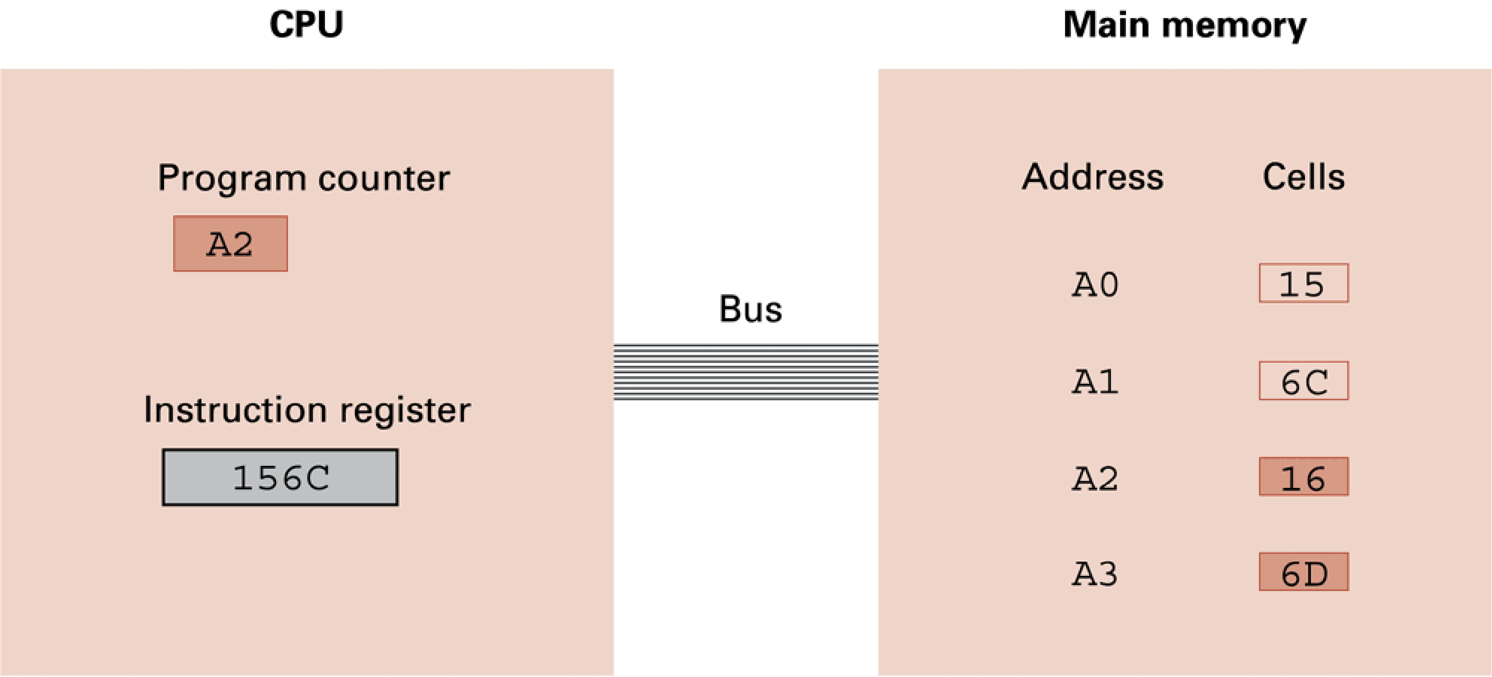
Program is stored in main memory beginning at address A0

Fetch Step (1/2)



a. At the beginning of the fetch step the instruction starting at address A0 is retrieved from memory and placed in the instruction register.

Fetch Step (2/2)



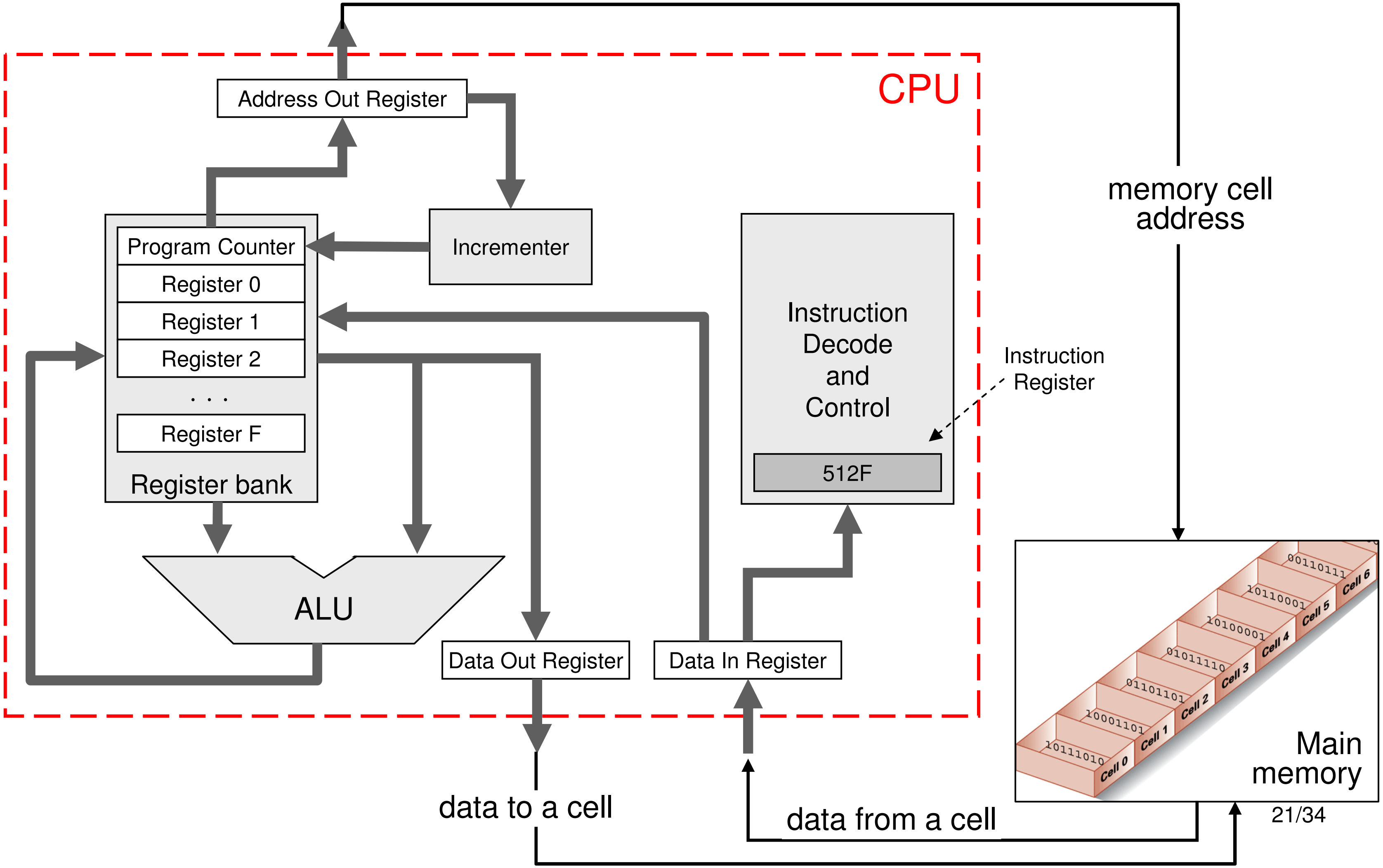
b. Then the program counter is incremented so that it points to the next instruction.

Decode and Execution

- To understand how execution and decode is done inside a CPU, we need a more detail architecture diagram of a CPU than the one illustrated in the textbook

  (Note: next three slides are beyond the scope of this course, but is good for your understanding of CPU)

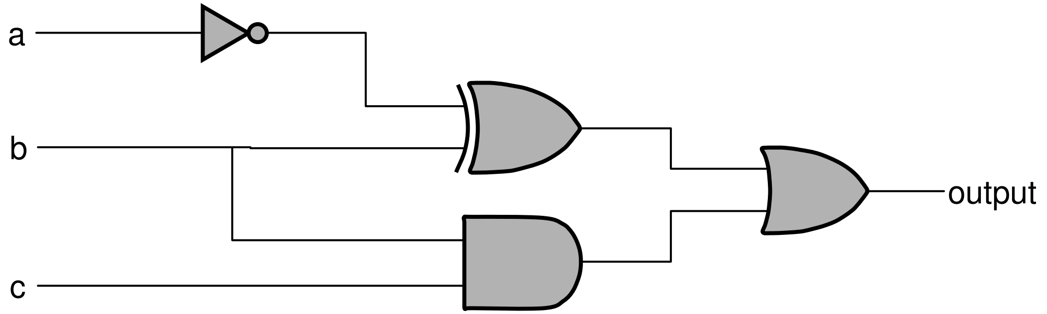
Internal Structure of a Realistic CPU



Concept of "Data Path"

- ❑ In previous chapter, we mentioned that a function can be implemented using a "circuit of gates:"

Circuit

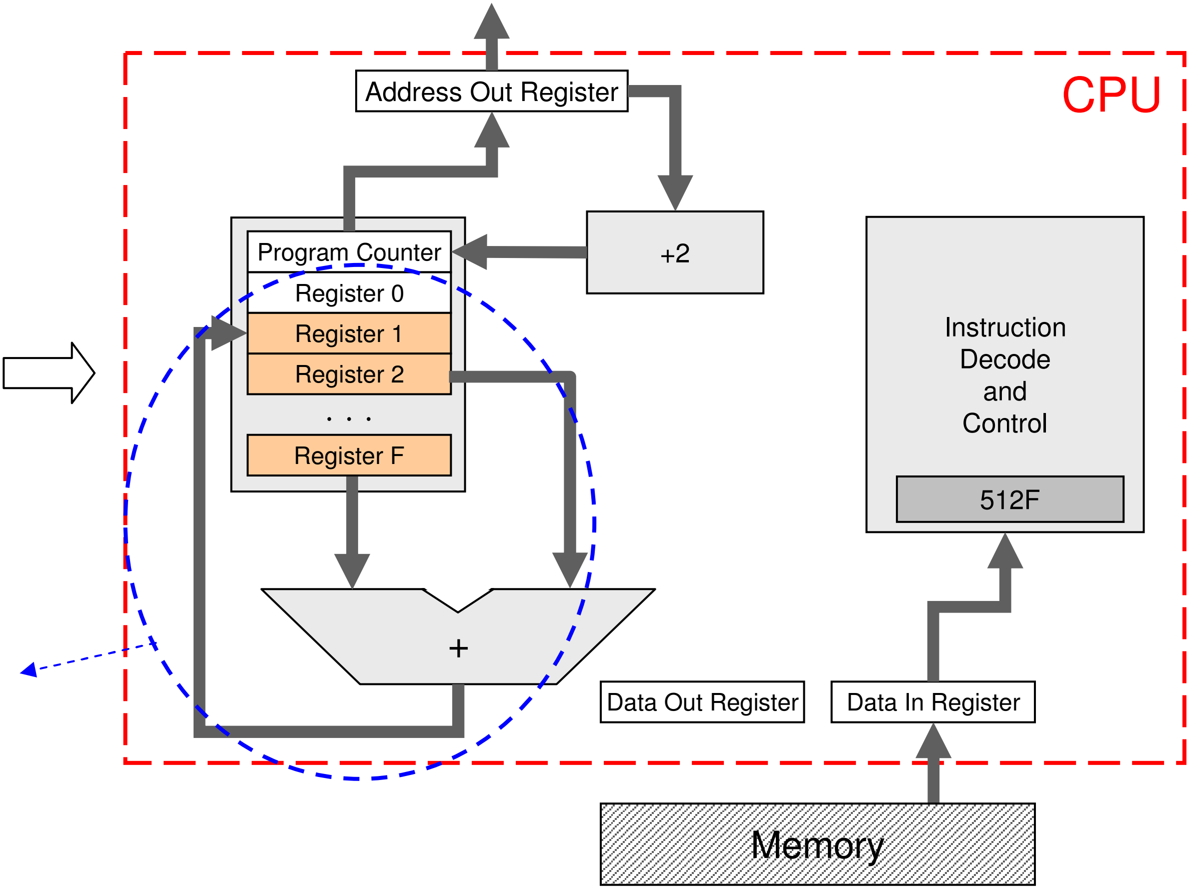


- ❑ Each function (or data processing circuit) can also be referred to as a "data path"
- ❑ A general purpose computer can control its data path based on the instructions it receives

Execution of an Instruction

- After decoding of an “add” instruction, the *data path* of a CPU may become as follows:

Computation of
R1 = R2 + RF

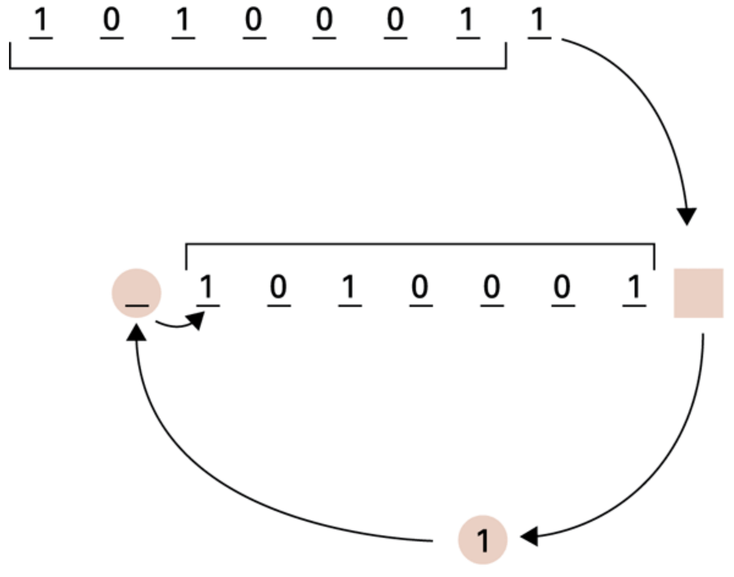


datapath

Arithmetic/Logic Operations

- The Arithmetic/Logic Unit (ALU) of a CPU is the muscle that performs data manipulation
- Three types of operations are supported by ALU:
  - Logic: AND, OR, XOR, NOT
  - Rotate and Shift: rotate (a.k.a. circular shift), logical shift, arithmetic shift
  - Arithmetic: add, subtract, multiply, divide

Rotation (Circular Shift)



The bit pattern represented by A3 (hexadecimal)

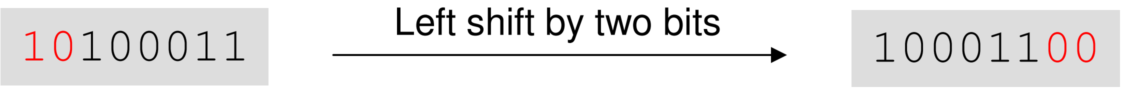
The bits move one position to the right. The rightmost bit "falls off" and is placed in the hole at the other end.

1 1 0 1 0 0 0 1

The final bit pattern, which is represented by D1 (hexadecimal)

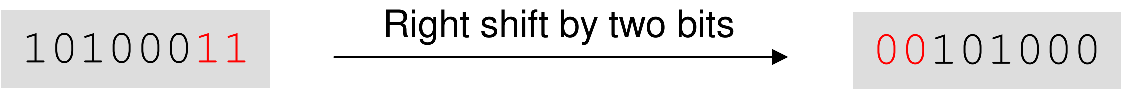
Logical Shift and Arithmetic Shift

- There is only one way to do an n-bit left shift



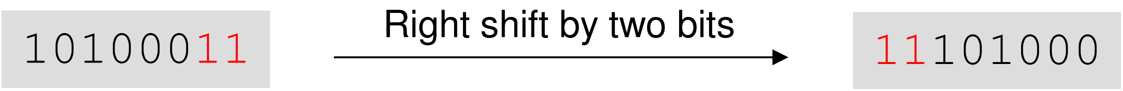
10100011 $\xrightarrow{\text{Left shift by two bits}}$ 10001100

- There are two ways to do n-bit right shift
  - Logical right shift

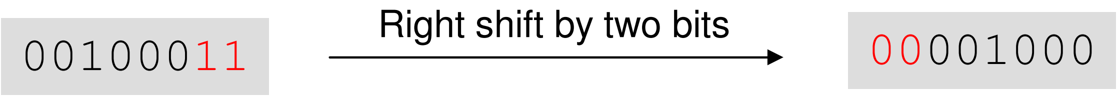


10100011 $\xrightarrow{\text{Right shift by two bits}}$ 00101000

  - Arithmetic right shift

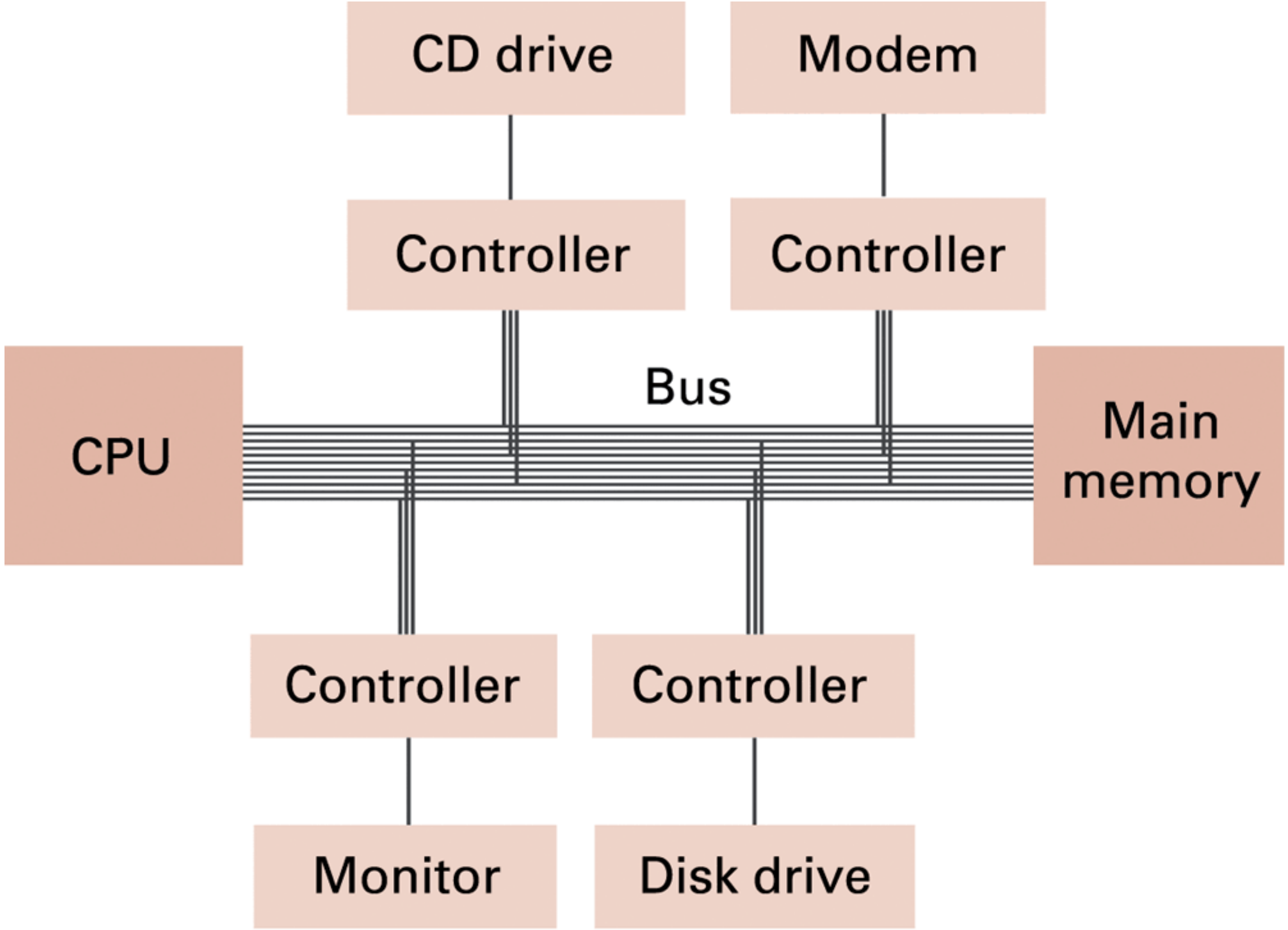


10100011 $\xrightarrow{\text{Right shift by two bits}}$ 11101000



00100011 $\xrightarrow{\text{Right shift by two bits}}$ 00001000

I/O Subsystem of a Computer



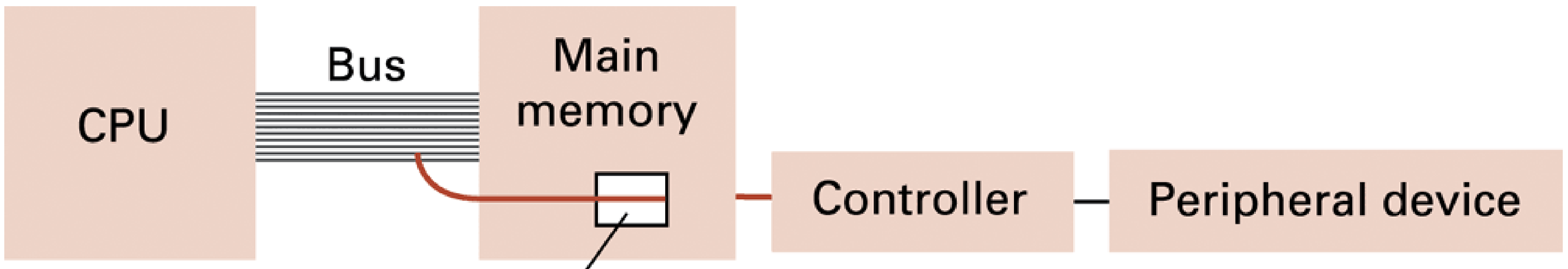
Communicating with Devices (1/2)

- Controller is an intermediary device that handles communication between the computer and a device
- CPU transfers data to/from the device using addresses
  - Dedicated instruction I/O: CPU uses dedicated I/O addresses to communicate with device controllers; often these addresses are called “I/O ports”
  - Memory-mapped I/O: CPU uses main memory addresses to communicate with device controllers

Communicating with Devices (2/2)

- ❑ Direct memory access (DMA)
  - ■ A controller can access main memory directly when CPU is not using the bus; this capability is called DMA
  - ■ Sometimes, we design a special circuit just to move data around inside the system, we also call this circuit a DMA
- ❑ Von Neumann Bottleneck
  - ■ If the CPU fetches both the instructions and data through a single bus connected to a memory device, the performance of the CPU would be limited by the performance of the BUS and the memory device
- ❑ Handshaking
  - ■ The process of controlling the transfer of data between components connected via a bus

Memory Mapped I/O Example



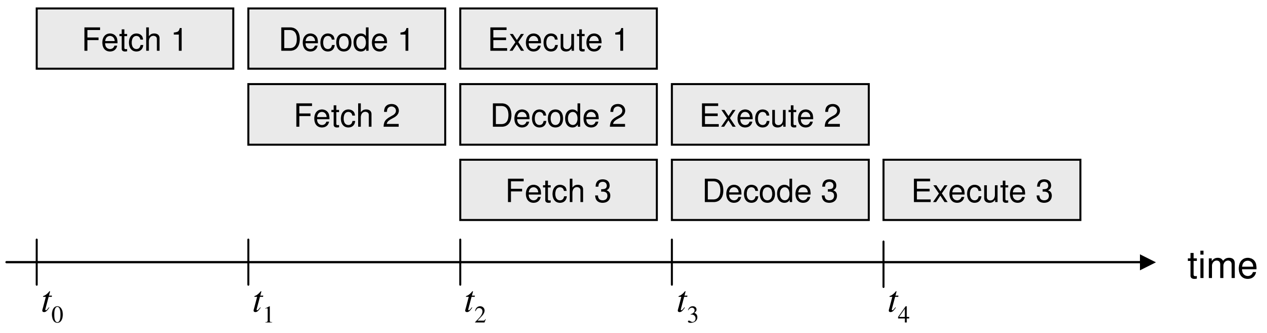
I/O address
(an empty memory cell
that does not really exist
in main memory device)

Data Communication Terminologies

- ❑ Modem (modulation-demodulation):
  - ■ In communication systems, modulation is a process that “pack” data (analog or digital) to a carrier signal (like packing goods in a truck) for transmission of data in the real world
  - ■ In computer terminology, “modem” is a device that perform this operation when the carrier is a telephone line
- ❑ Serial communication: transfers one bit at a time
- ❑ Parallel communication: transfers multiple bits simultaneously
- ❑ Multiplexing: interleaving of data so that different data can be transmitted over a single communication path

Improving CPU Architecture

- Pipelining: overlap steps of the CPU operation cycles



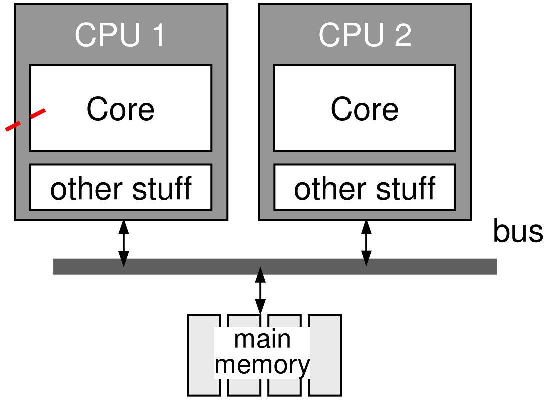
- Parallel processing: execute multiple operations simultaneously
- Parallel processing can be performed within a CPU or across CPUs

Multiprocessor Systems

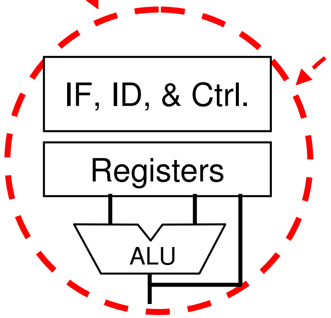
- ❑ A single processing unit execute one instruction a time, which is called Single-Instruction stream Single-Data stream (SISD) architecture
- ❑ If multiple processing units (multi-CPU, multi-core, or multi-ALU) are connected to the main memory, we have parallel processing architecture:
  - ■ Multiple-Instructions stream Multiple-Data stream (MIMD): different instructions are issued at the same time to operate on different data
  - ■ Single-Instruction stream Multiple-Data stream (SIMD): one instruction is issued to operate on different data

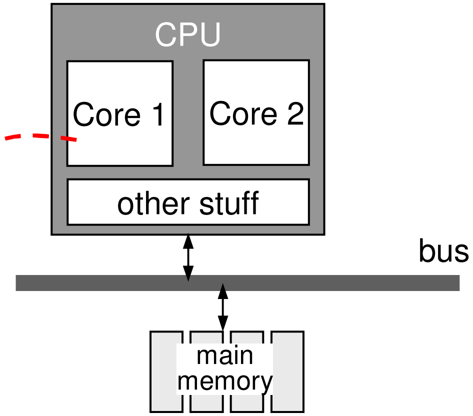
Multi-CPU, Multi-Core, Multi-ALU

- ❑ Multi-CPU Architecture:

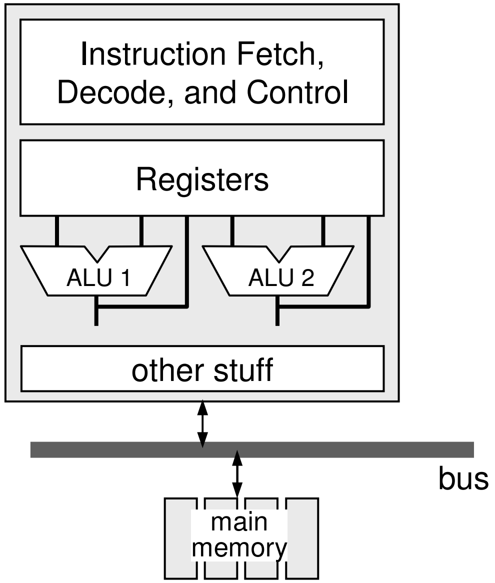


- ❑ Multi-Core Architecture:





- ❑ Multi-ALU Architecture:



Note: “other stuff” could be interface logic, cache, MMU, timer, … etc.