Computer Organization and Architecture

Introduction

Books

- Computer Organization and Architecture – Designing for Performance
  - William Stallings
- Computer Organization
  - Hamacher, Vranesic & Zaky

Architecture & Organization 1

- Architecture is those attributes visible to the programmer
  - Instruction set, number of bits used for data representation, I/O mechanisms, addressing techniques.
  - e.g. Is there a multiply instruction?
- Organization is how features are implemented
  - Control signals, interfaces, memory technology.
  - e.g. Is there a hardware multiply unit or is it done by repeated addition?

Architecture & Organization 2

- All Intel x86 family share the same basic architecture
- The IBM System/370 family share the same basic architecture

- This gives code compatibility
    - At least backwards
- Organization differs between different versions

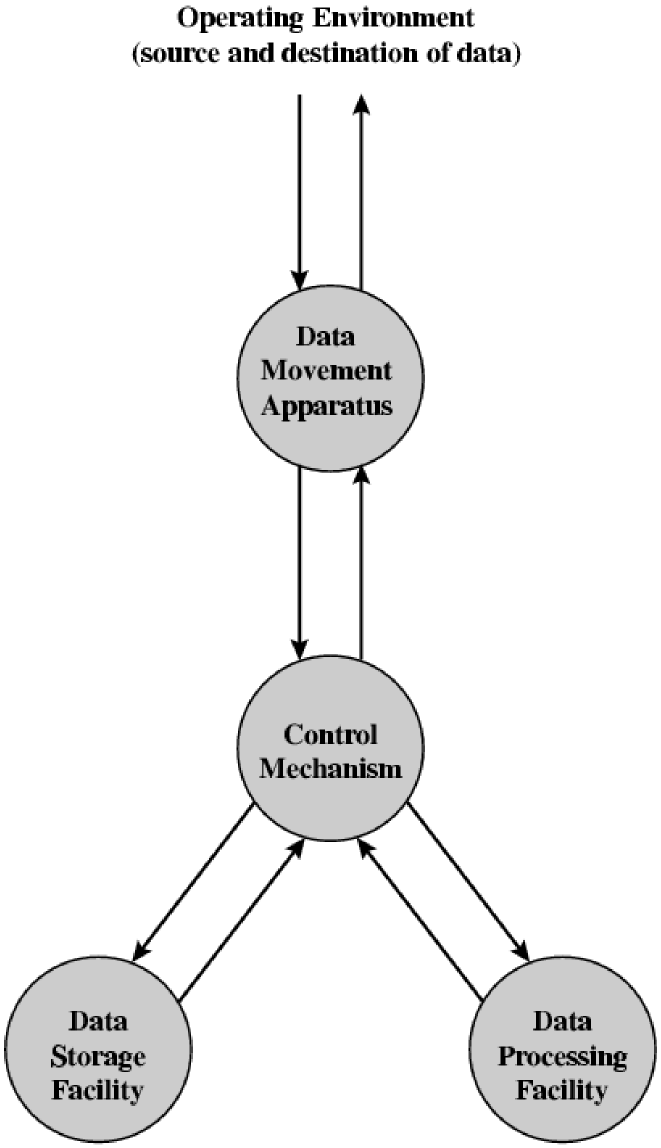
Structure & Function

- Structure is the way in which components relate to each other
- Function is the operation of individual components as part of the structure

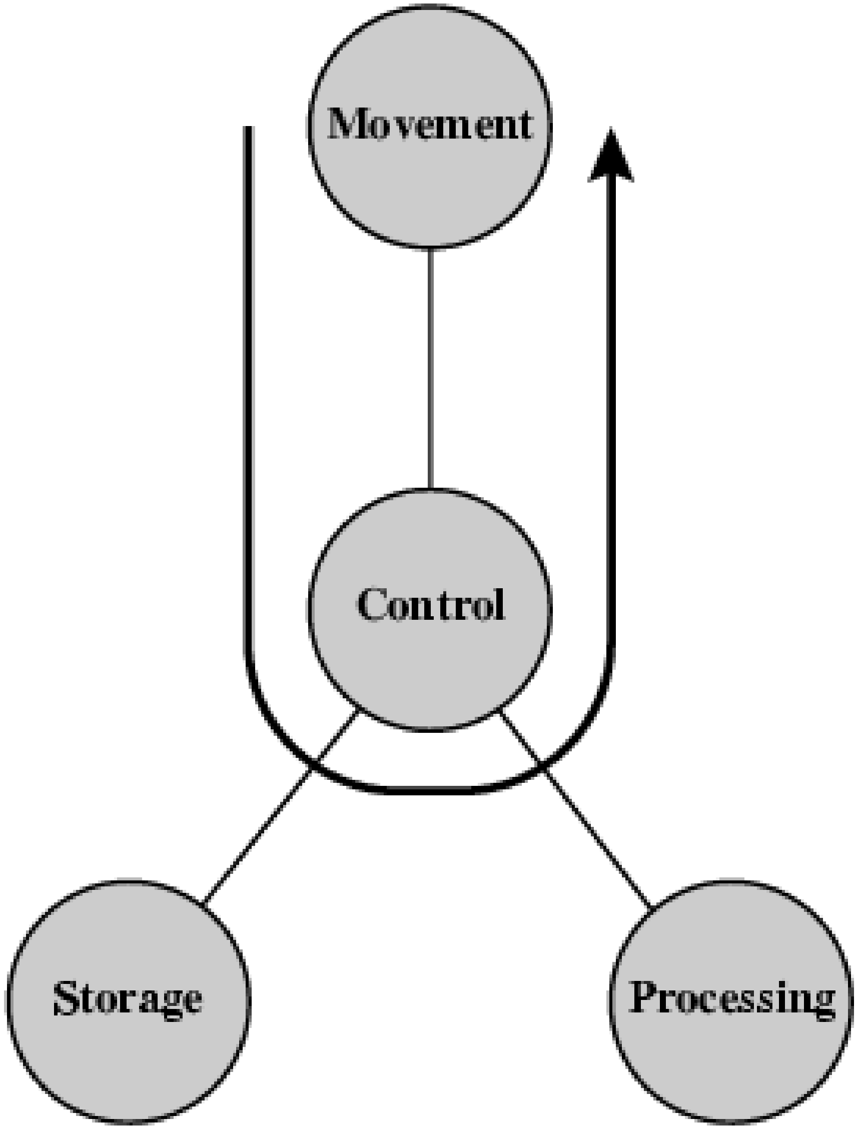
Function

- All computer functions are:
  - Data processing
  - Data storage
  - Data movement
  - Control

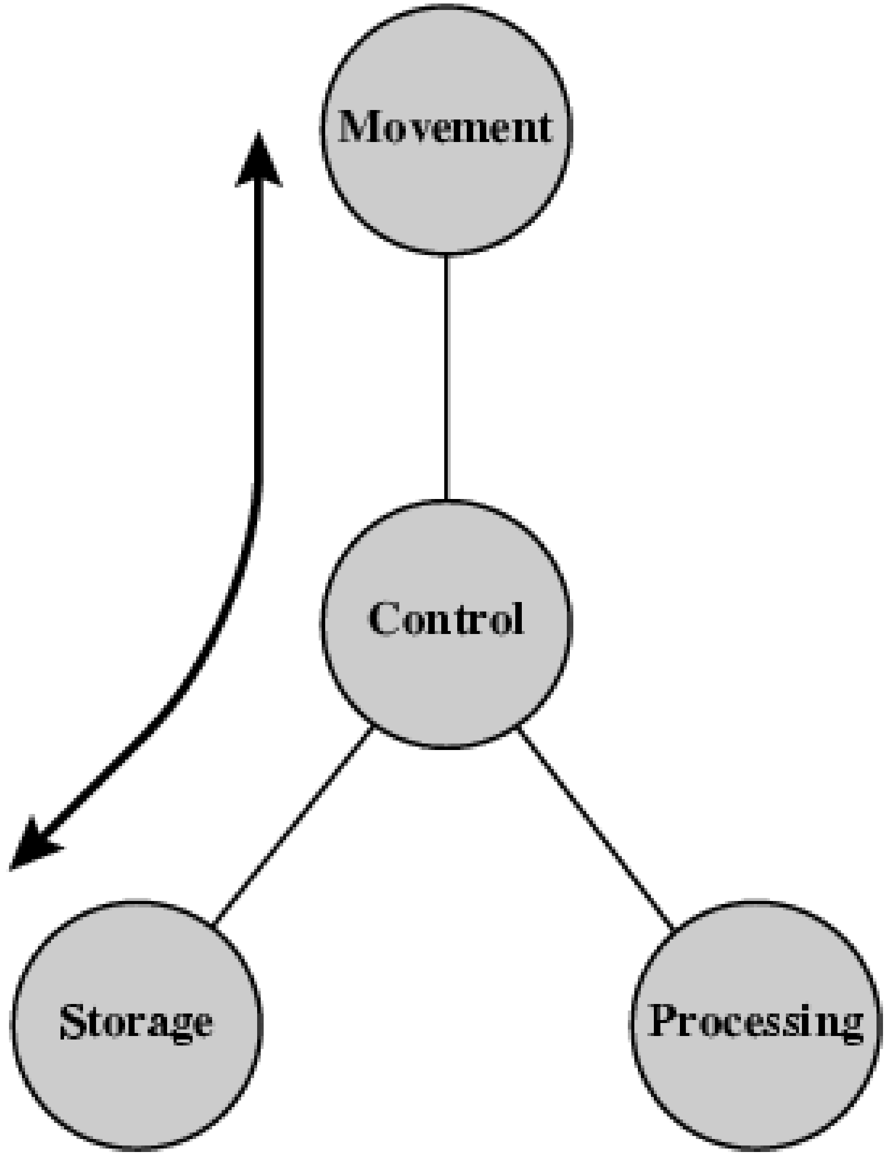
Functional View



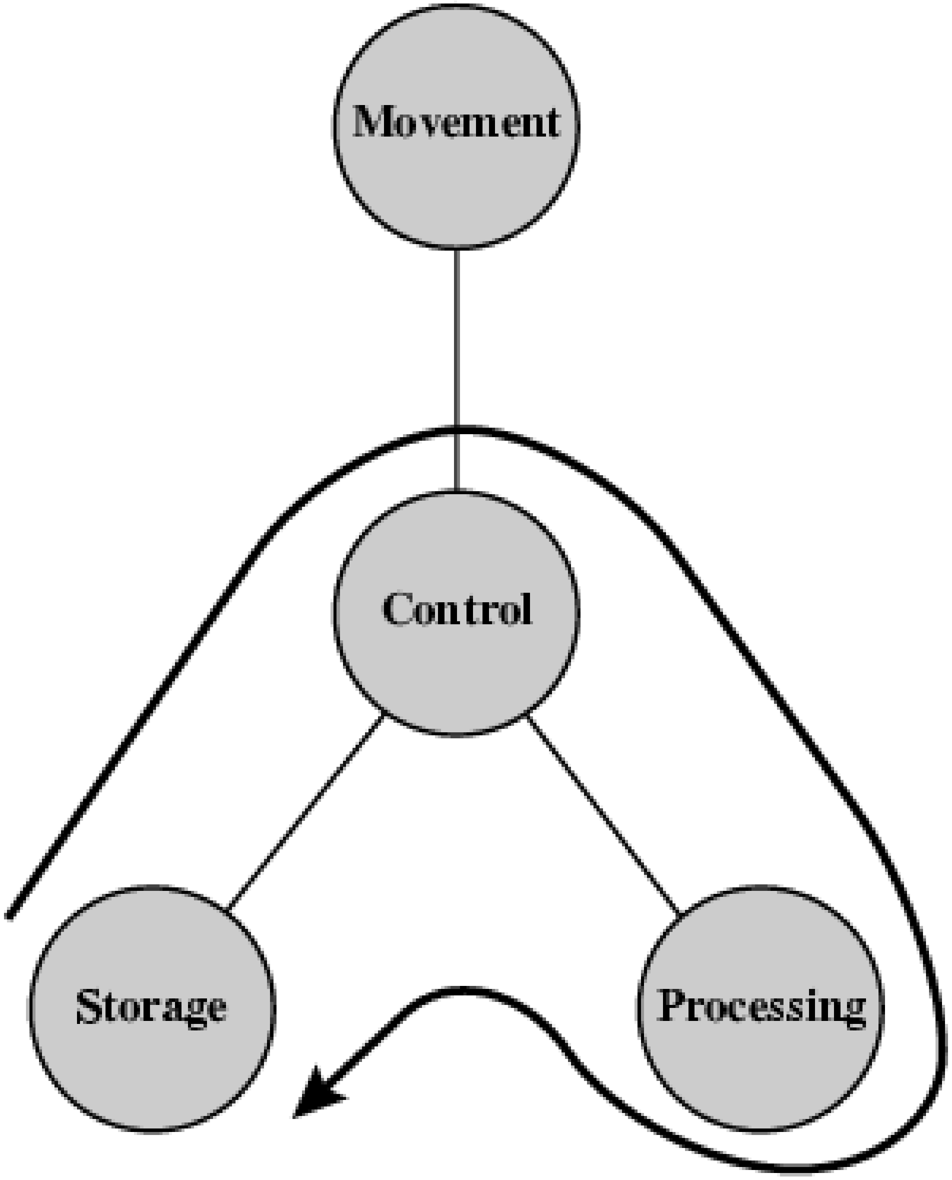
Operations (a) Data movement



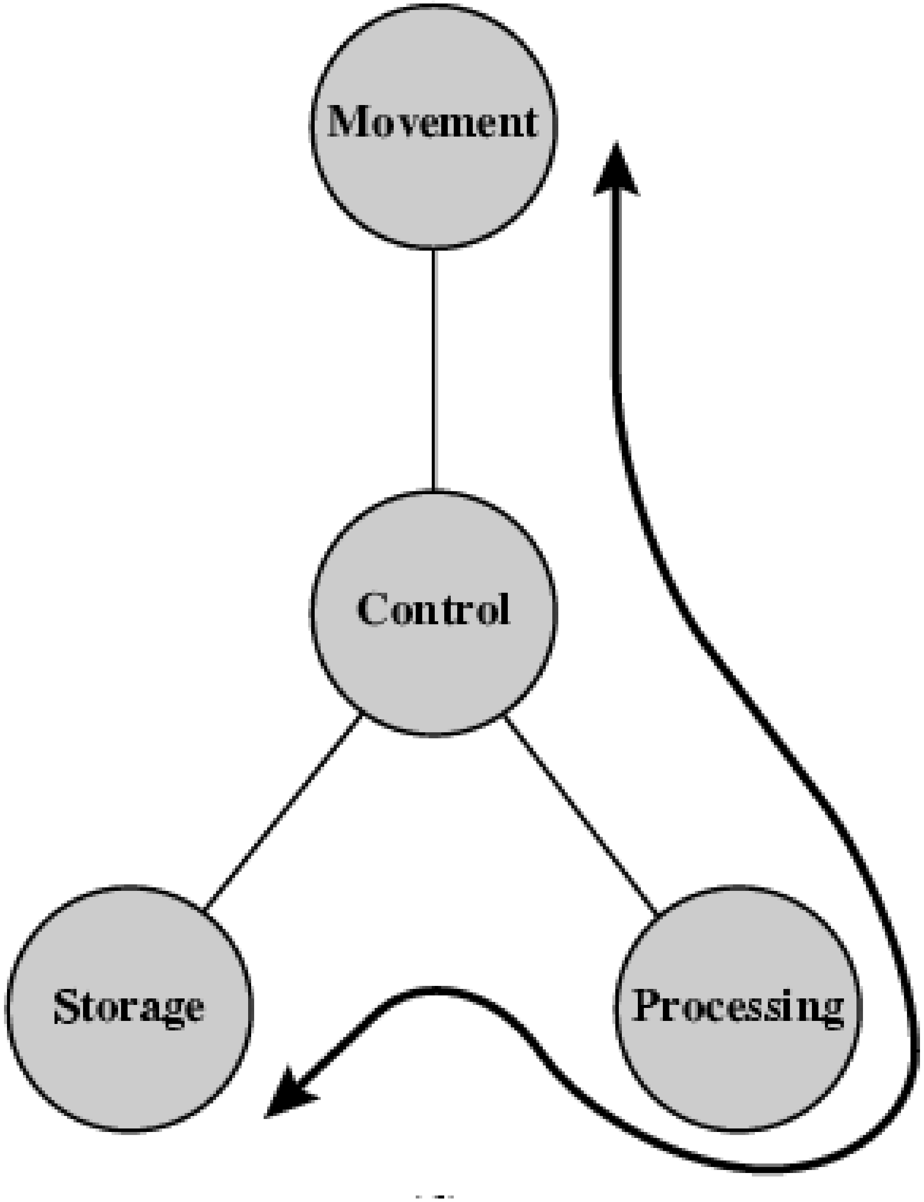
Operations (b) Storage



Operation (c) Processing from/to storage



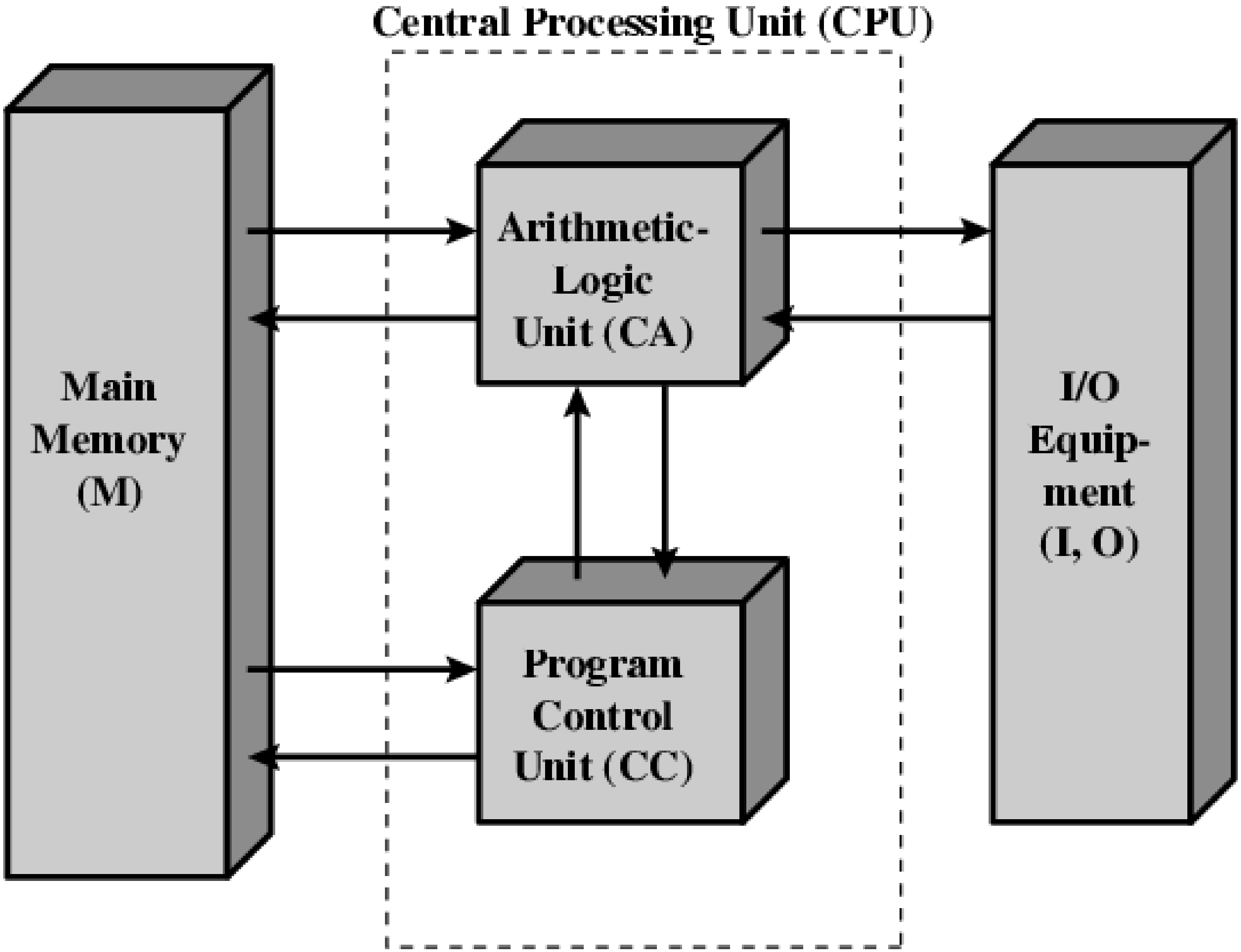
Operation (d)
Processing from storage to I/O



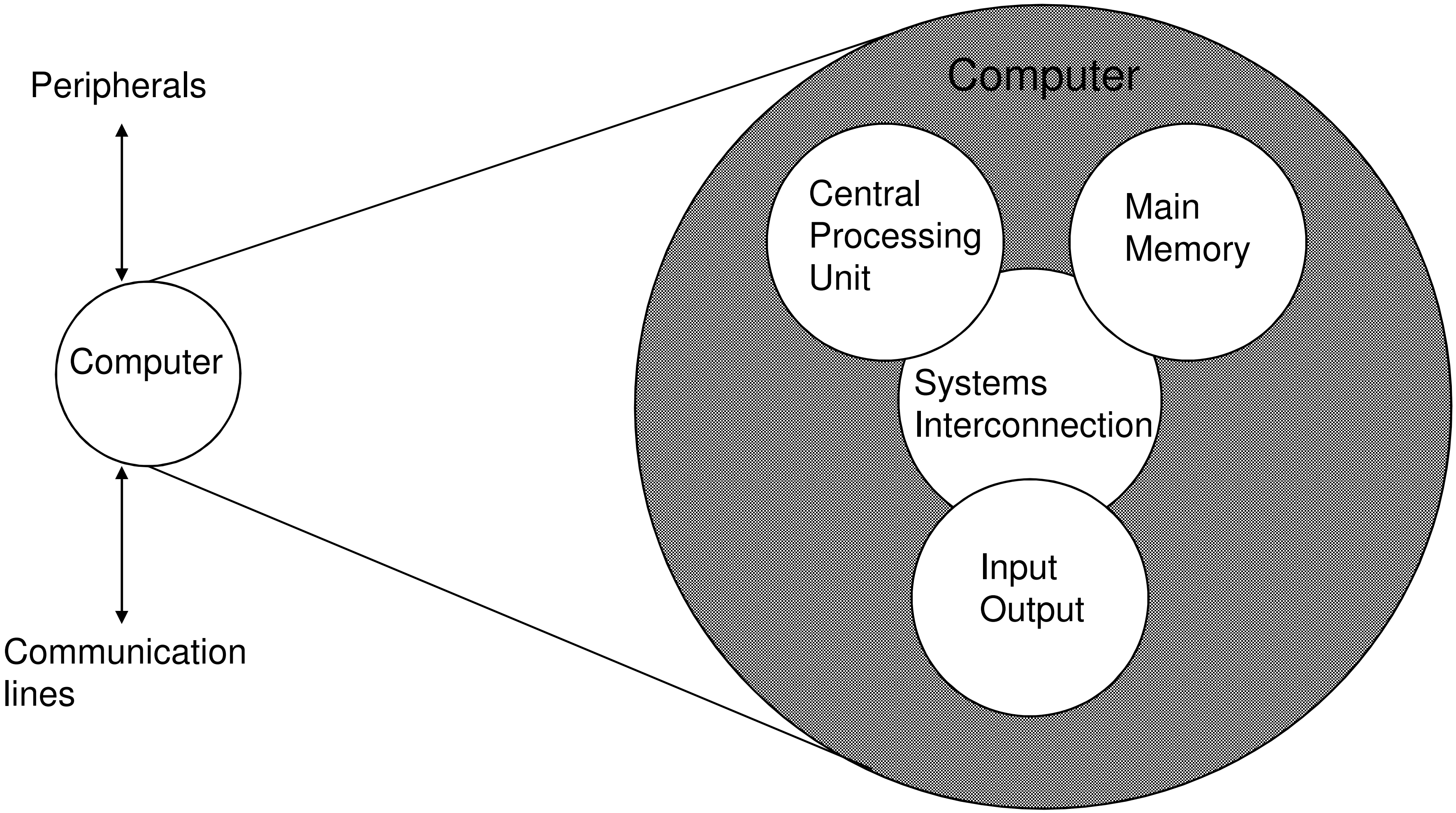
Von Neumann Stored Program Principle

- Stored Program concept
- Main memory storing programs and data
- ALU operating on binary data
- Control unit interpreting instructions from memory and executing
- Input and output equipment operated by control unit
- Princeton Institute for Advanced Studies
    - IAS
- Completed 1952

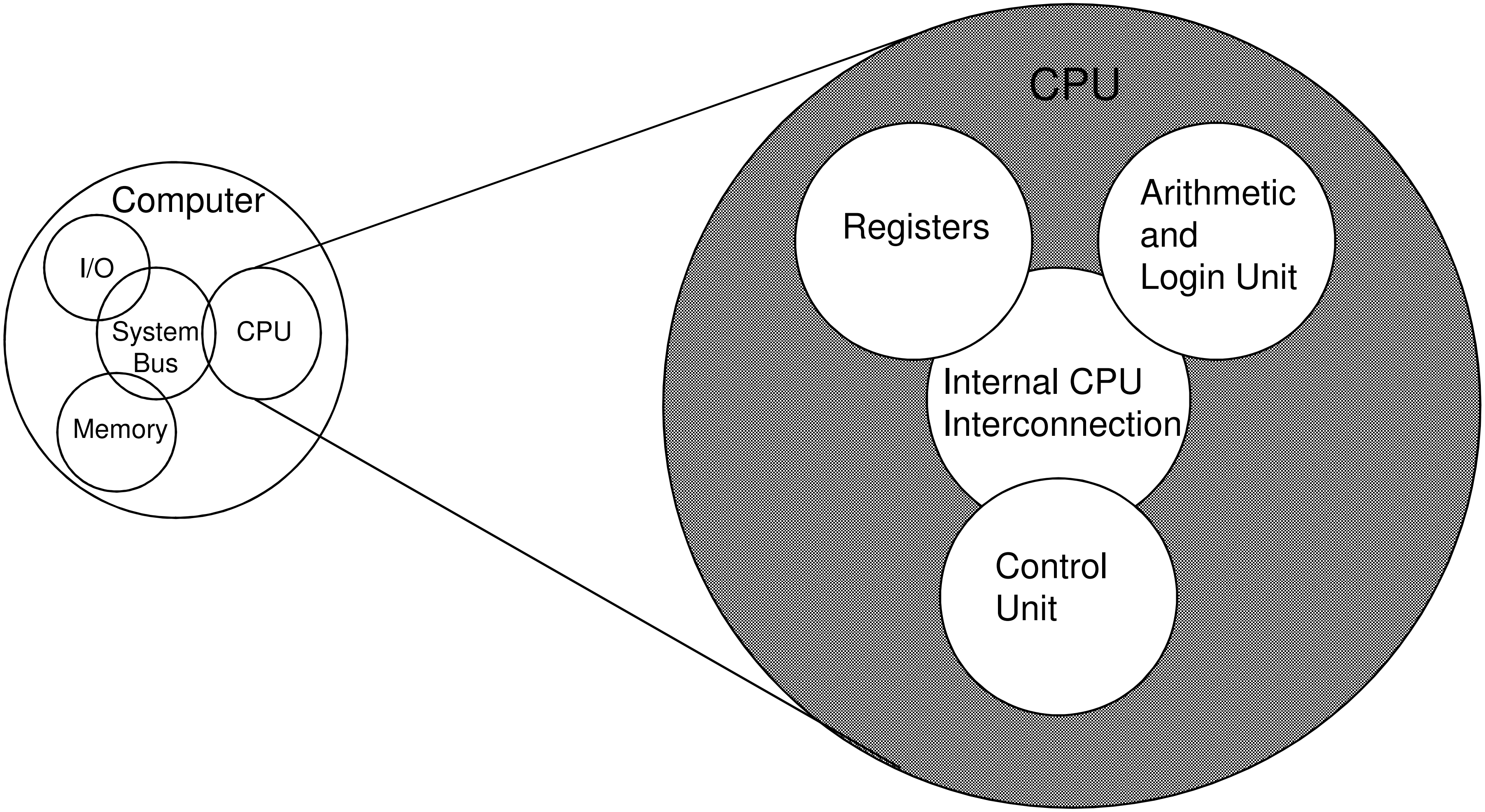
Structure of von Neumann machine


Central Processing Unit (CPU)
Main Memory (M)
Arithmetic-Logic Unit (CA)
Program Control Unit (CC)
I/O Equip-ment (I, O)

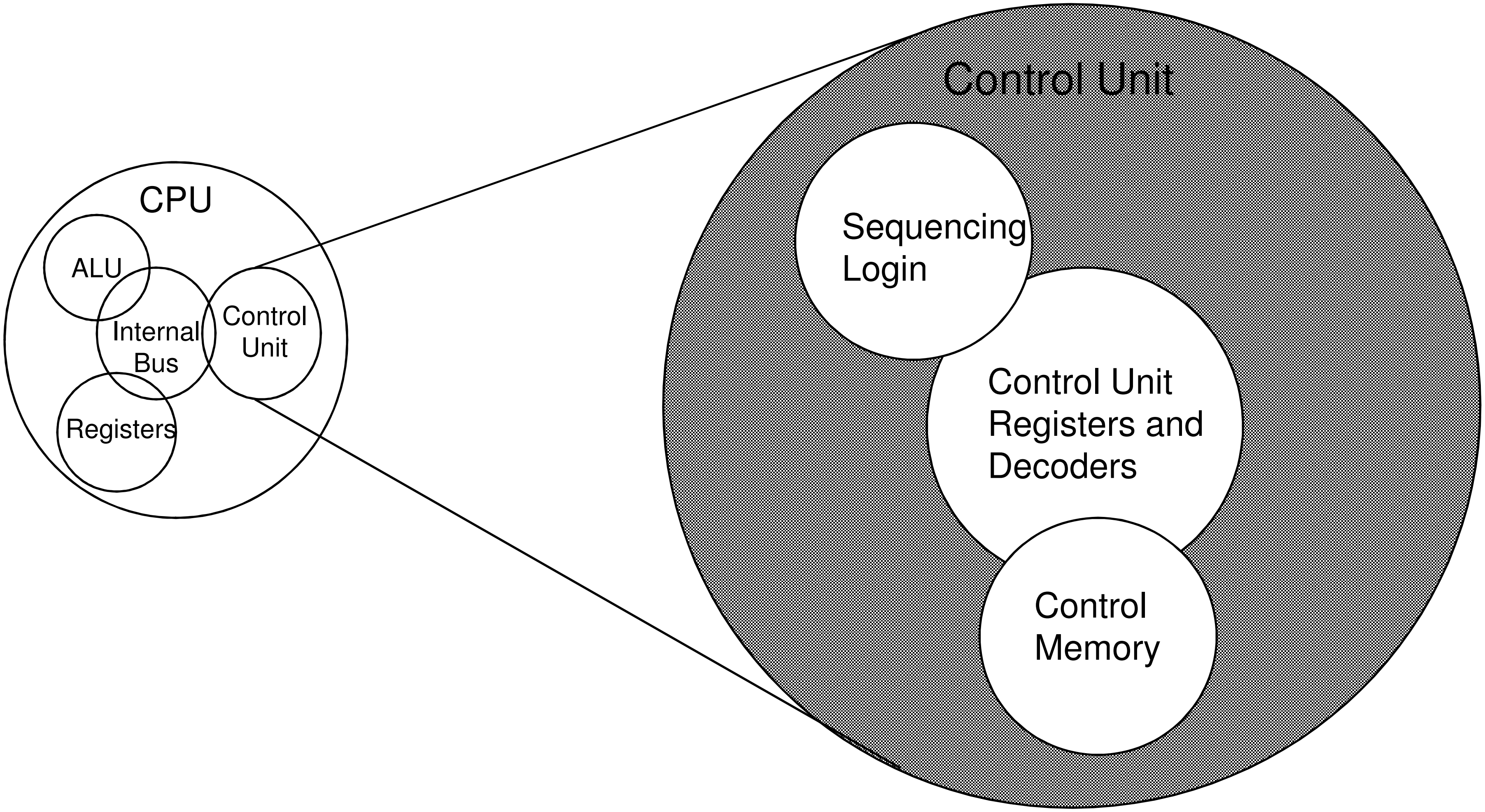
Structure - Top Level



Structure - The CPU



Structure - The Control Unit



Program Concept

- Hardwired systems are inflexible
- General purpose hardware can do different tasks, given correct control signals
- Instead of re-wiring, supply a new set of control signals

What is a program?

- A sequence of steps
- For each step, an arithmetic or logical operation is done
- For each operation, a different set of control signals is needed

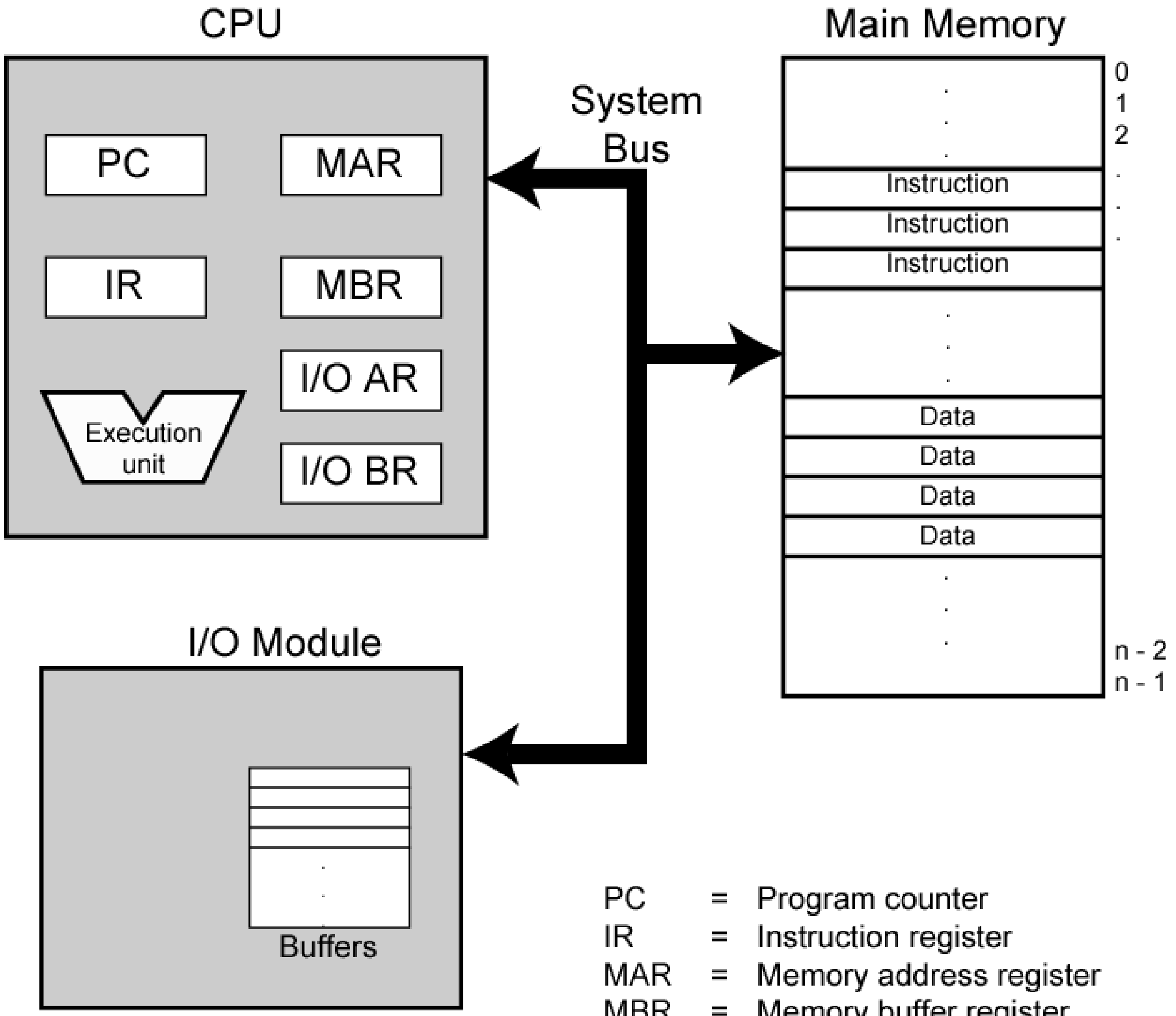
Function of Control Unit

- For each operation a unique code is provided
  - e.g. ADD, MOVE
- A hardware segment accepts the code and issues the control signals

Components

- The Control Unit and the Arithmetic and Logic Unit constitute the Central Processing Unit
- Data and instructions need to get into the system and results out
  - Input/output
- Temporary storage of code and results is needed
  - Main memory

Computer Components: Top Level View



PC = Program counter
IR = Instruction register
MAR = Memory address register
MBR = Memory buffer register
I/O AR = Input/output address register
I/O BR = Input/output buffer register

Connecting

- All the units must be connected
- Different type of connection for different type of unit
  - Memory
  - Input/Output
  - CPU

Memory Connection

- Receives and sends data
- Receives addresses (of locations)
- Receives control signals
  - Read
  - Write
  - Timing

Input/Output Connection(1)

- Similar to memory from computer’s viewpoint
- Output
  - Receive data from computer
  - Send data to peripheral
- Input
  - Receive data from peripheral
  - Send data to computer

Buses

- There are a number of possible interconnection systems
- Single and multiple BUS structures are most common
- e.g. Control/Address/Data bus (PC)
- e.g. Unibus (DEC-PDP)

What is a Bus?

- A communication pathway connecting two or more devices
- Usually broadcast
- Often grouped
    - A number of channels in one bus
    - e.g. 32 bit data bus is 32 separate single bit channels

Data Bus

- Carries data
  - Remember that there is no difference between “data” and “instruction” at this level
- Width is a key determinant of performance
  - 8, 16, 32, 64 bit

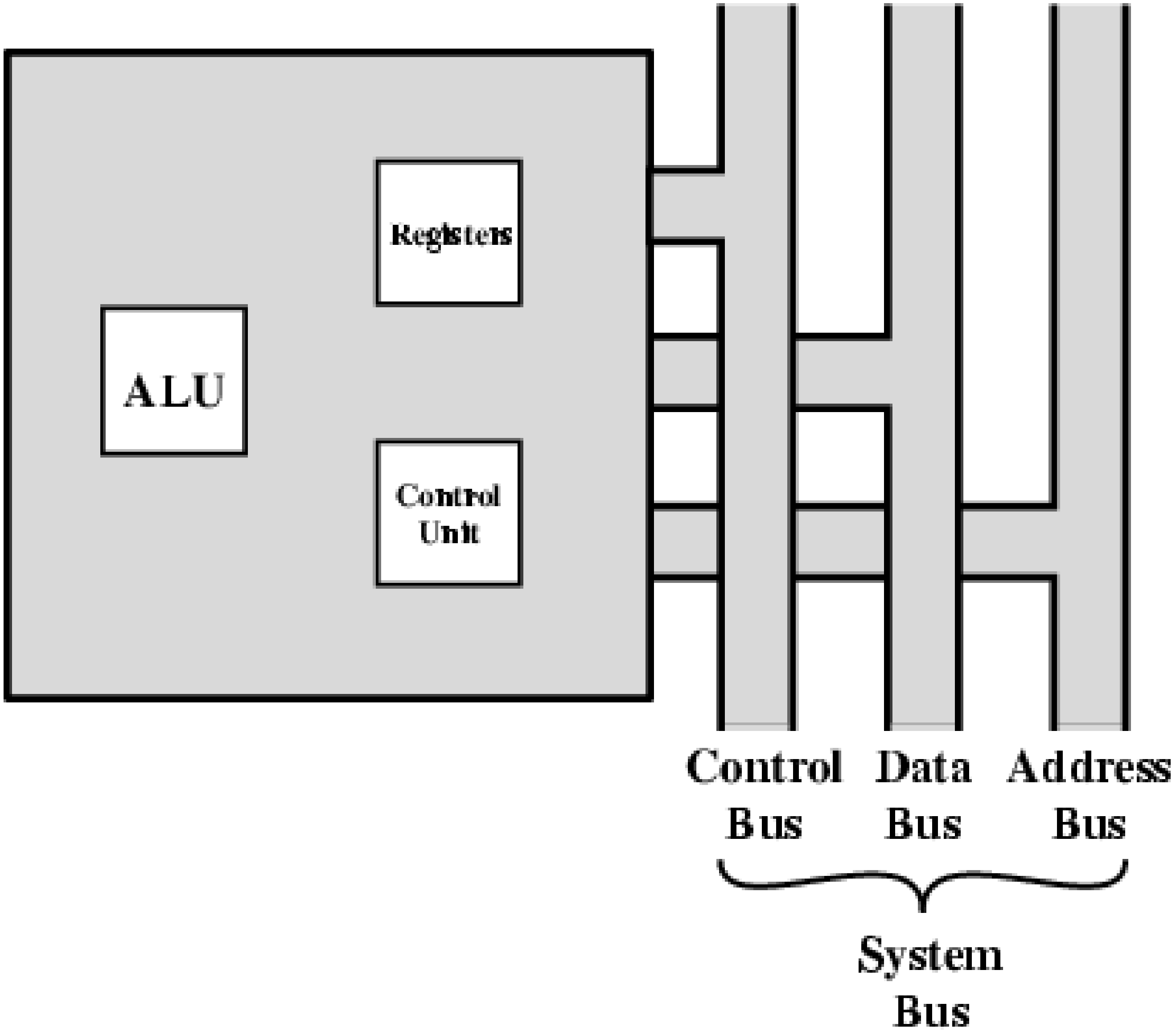
Address bus

- Identify the source or destination of data
- e.g. CPU needs to read an instruction (data) from a given location in memory
- Bus width determines maximum memory capacity of system
  - e.g. 8080 has 16 bit address bus giving 64k address space

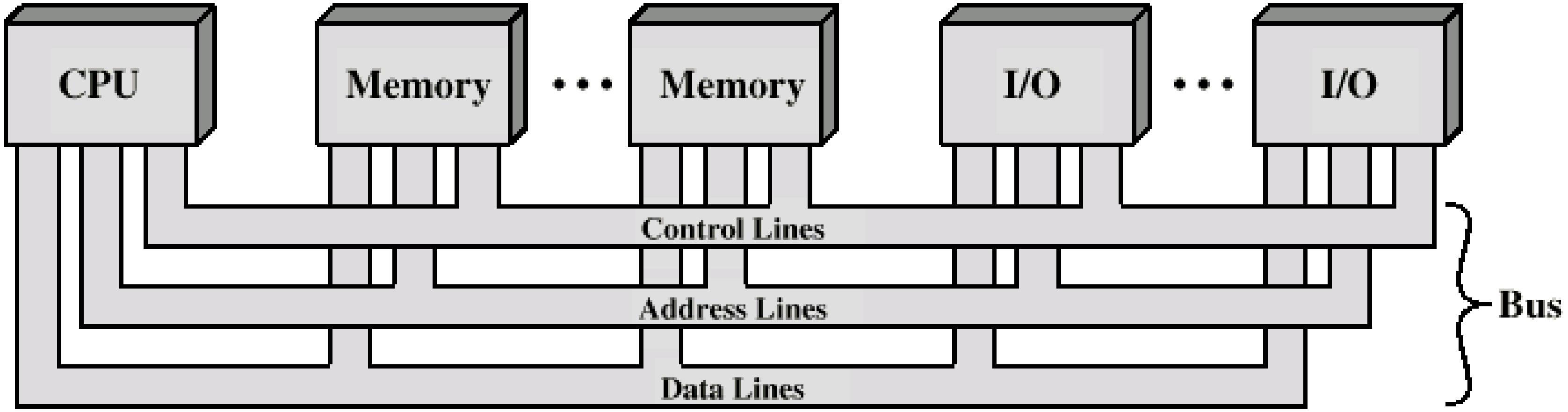
Control Bus

- Control and timing information
  - Memory read/write signal
  - Interrupt request
  - Clock signals

CPU With Systems Bus

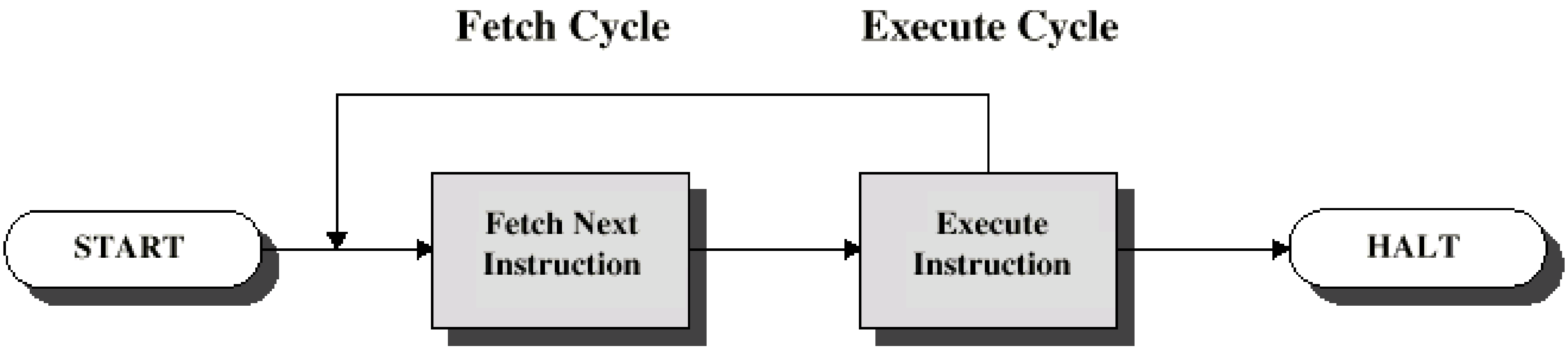


Bus Interconnection Scheme

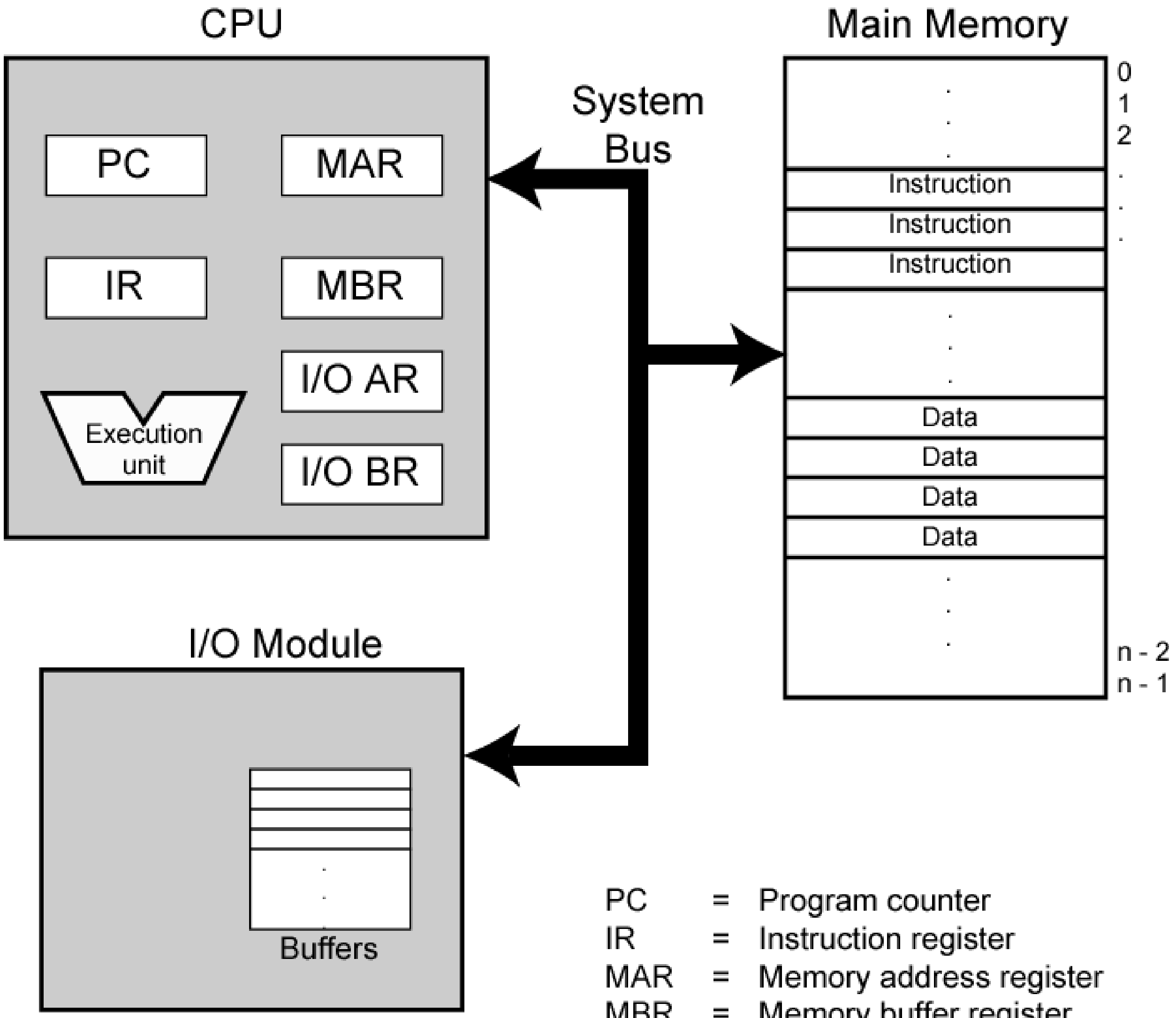


Instruction Cycle

- Two steps:
  - —Fetch
  - —Execute



Computer Components: Top Level View



PC	=	Program counter
IR	=	Instruction register
MAR	=	Memory address register
MBR	=	Memory buffer register
I/O AR	=	Input/output address register
I/O BR	=	Input/output buffer register

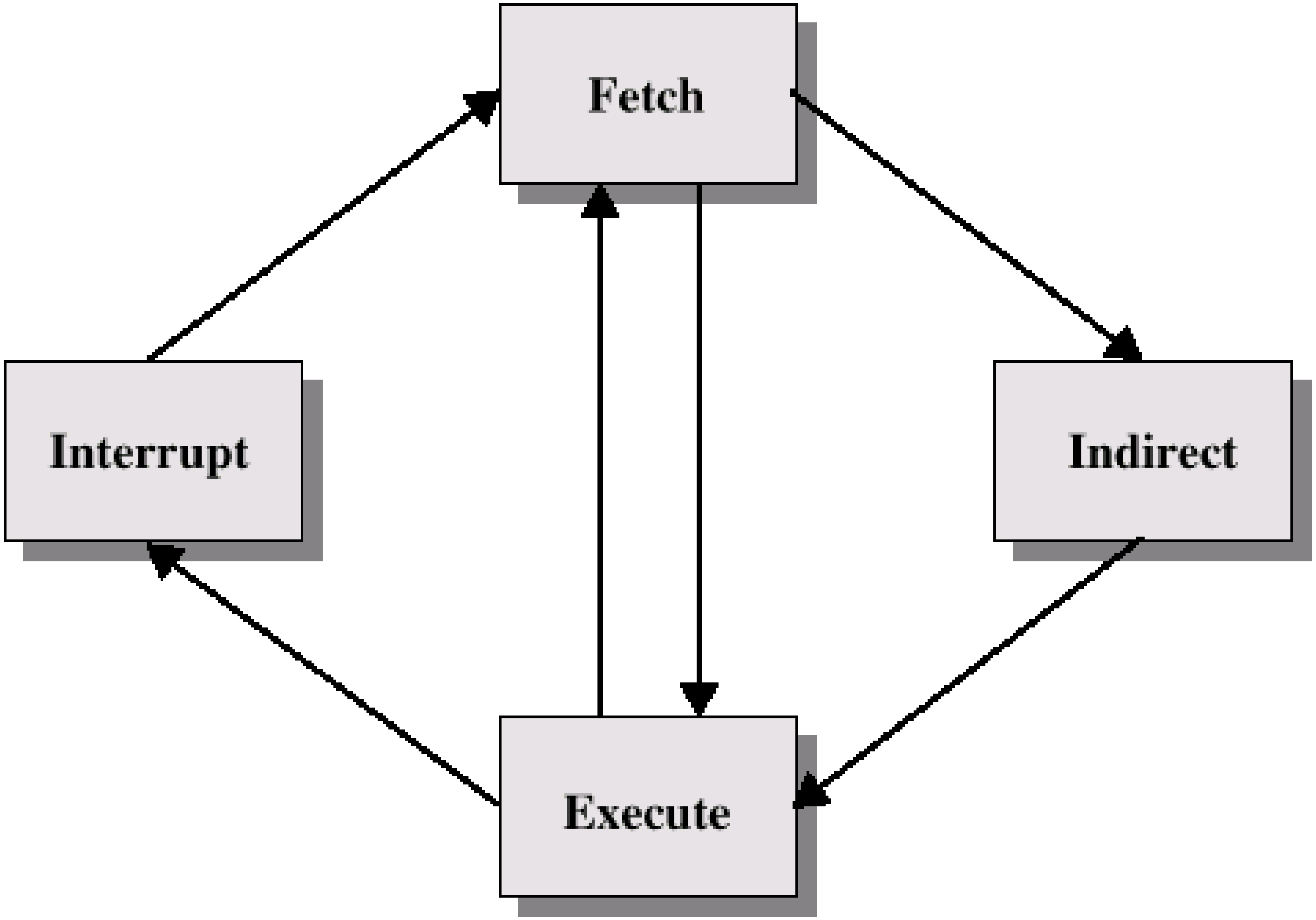
Fetch Cycle

- Program Counter (PC) holds address of next instruction to fetch
- Processor fetches instruction from memory location pointed to by PC
- Increment PC
  - Unless told otherwise
- Instruction loaded into Instruction Register (IR)
- Processor interprets instruction and performs required actions

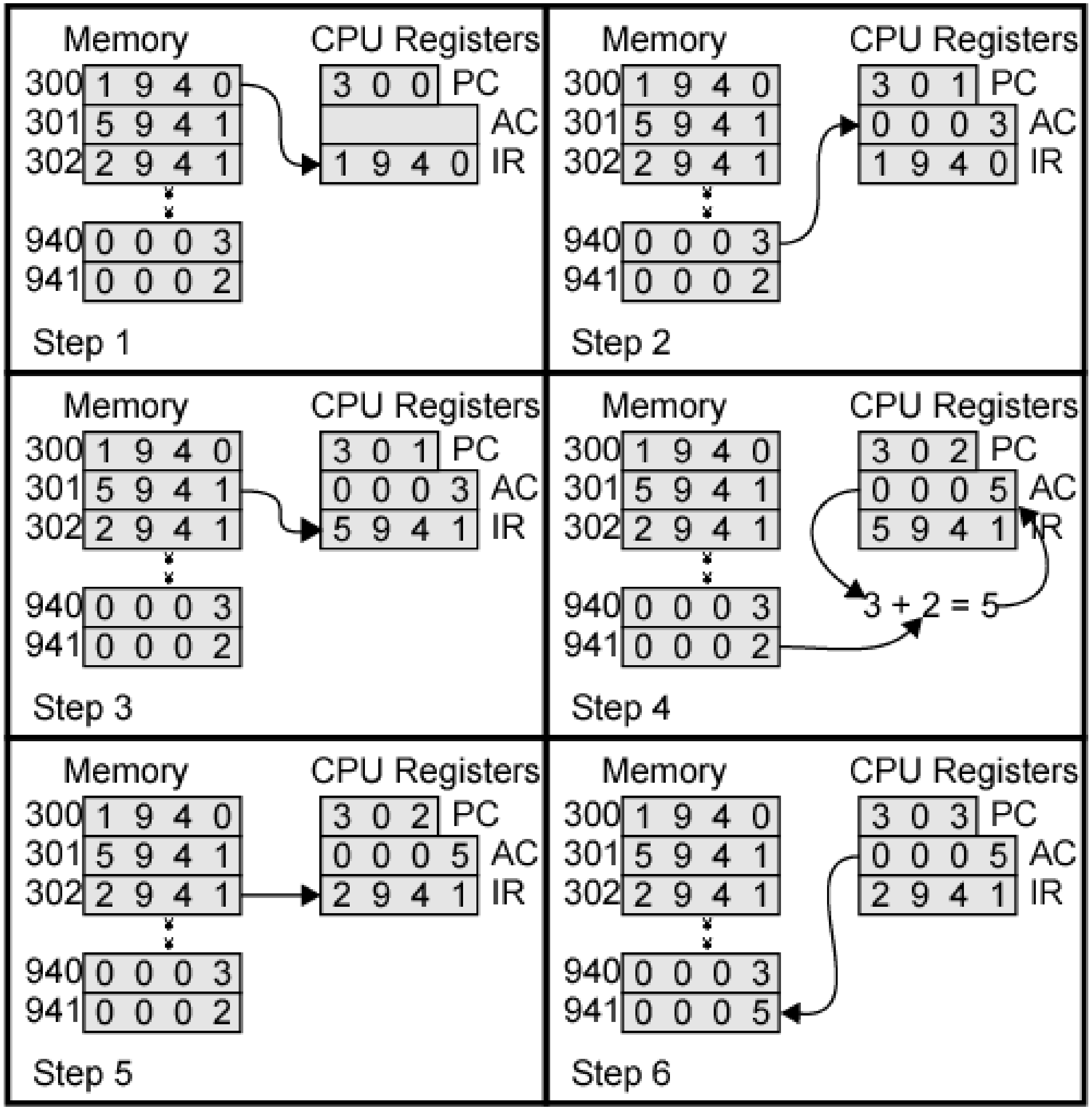
Execute Cycle

- Processor-memory
  - data transfer between CPU and main memory
- Processor I/O
  - Data transfer between CPU and I/O module
- Data processing
  - Some arithmetic or logical operation on data
- Control
  - Alteration of sequence of operations
  - e.g. jump
- Combination of above

Instruction Cycle with Indirect



Example of Program Execution



Memory

300 1 9 4 0
301 5 9 4 1
302 2 9 4 1
940 0 0 0 3
941 0 0 0 2

CPU Registers

3 0 0 PC
AC
1 9 4 0 IR

Step 1

Memory

300 1 9 4 0
301 5 9 4 1
302 2 9 4 1
940 0 0 0 3
941 0 0 0 2

CPU Registers

3 0 1 PC
0 0 0 3 AC
1 9 4 0 IR

Step 2

Memory

300 1 9 4 0
301 5 9 4 1
302 2 9 4 1
940 0 0 0 3
941 0 0 0 2

CPU Registers

3 0 1 PC
0 0 0 3 AC
5 9 4 1 IR

Step 3

Memory

300 1 9 4 0
301 5 9 4 1
302 2 9 4 1
940 0 0 0 3
941 0 0 0 2

CPU Registers

3 0 2 PC
0 0 0 5 AC
5 9 4 1 IR

3 + 2 = 5

Step 4

Memory

300 1 9 4 0
301 5 9 4 1
302 2 9 4 1
940 0 0 0 3
941 0 0 0 2

CPU Registers

3 0 2 PC
0 0 0 5 AC
2 9 4 1 IR

Step 5

Memory

300 1 9 4 0
301 5 9 4 1
302 2 9 4 1
940 0 0 0 3
941 0 0 0 5

CPU Registers

3 0 3 PC
0 0 0 5 AC
2 9 4 1 IR

Step 6

Data Bus and Address Bus

- Size of Address Bus:

SIZE	BINARY	DEC	HEXA	
8	0000 0000	0	00	
8	1111 1111	255	FF	
8	0101 0111	87	57	
8	0000 0110	6	06	
10	11 1111 1111	1023	3FF	
12	1111 1111 1111	4095	FFF	
16	1111 1111 1111 1111	2^{16} -1	FFFF	
20	1111 1111 1111 1111 1111	2^{20} -1	FFFFF	
30	11 ……………………………. 1111	2^{30} -1	3FFFFFFF	
32	1111 ………………………….. 1111	2^{32} -1	FFFFFFFF	

Data Bus and Address Bus

- Size of Address Bus and Memory Capacity:

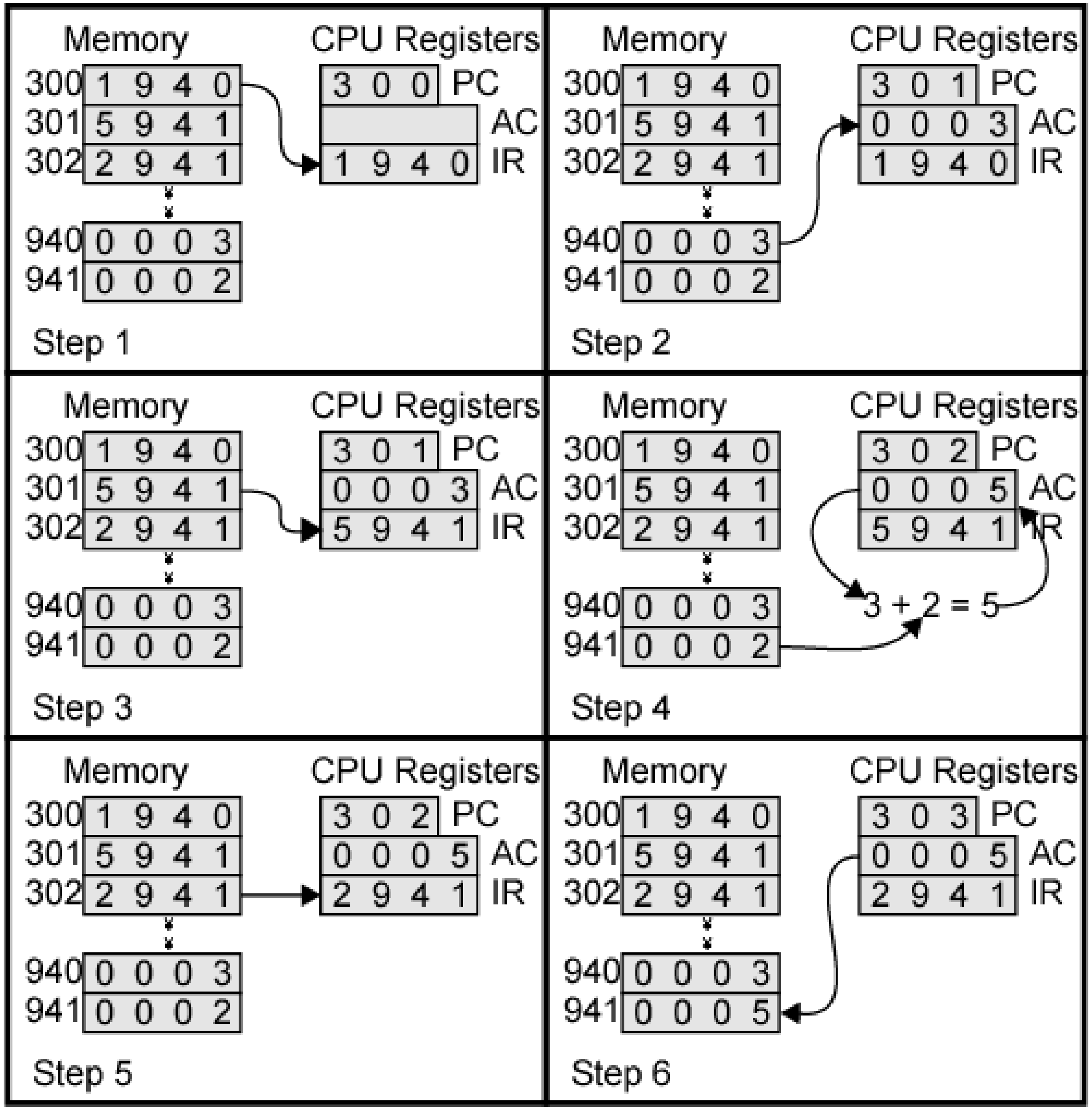
SIZE	BINARY	DEC	HEXA	
8	0000 0000	0	00	
8	1111 1111	255	FF	256
10	11 1111 1111	1023	3FF	1K
12	1111 1111 1111	4095	FFF	4K
16	1111 1111 1111 1111	2^{16} -1	FFFF	64K
20	1111 1111 1111 1111 1111	2^{20} -1	FFFFF	1M
30	11 1111	2^{30} -1	3FFFFFFF	1G
32	1111 1111	2^{32} -1	FFFFFFFF	4G

Data Bus and Address Bus

- Size of Data Bus/Memory Location:

SIZE	BINARY	DEC	HEXA
8	1111 1111 0111 1111	-127 +127	00 - FF
12	1111 1111 1111 0111 1111 1111	-2047 +2047	000 - FFF
16	1111 1111 1111 1111 0111 1111 1111 1111	$-(2^{15} – 1)$ $+(2^{15} – 1)$	0000- FFFF
20	1111 1111 1111 1111 1111 0111 1111 1111 1111 1111	$-(2^{19} -1)$ $+(2^{19} -1)$	00000 - FFFFF
32	1111 …………………………….1111 0111 …………………………….1111	$-(2^{31} -1)$ $+(2^{31} -1)$	00000000 – FFFFFFFF

Example of Program Execution



Memory

300 1 9 4 0
301 5 9 4 1
302 2 9 4 1
940 0 0 0 3
941 0 0 0 2

CPU Registers

3 0 0 PC
AC
1 9 4 0 IR

Step 1

Memory

300 1 9 4 0
301 5 9 4 1
302 2 9 4 1
940 0 0 0 3
941 0 0 0 2

CPU Registers

3 0 1 PC
0 0 0 3 AC
1 9 4 0 IR

Step 2

Memory

300 1 9 4 0
301 5 9 4 1
302 2 9 4 1
940 0 0 0 3
941 0 0 0 2

CPU Registers

3 0 1 PC
0 0 0 3 AC
5 9 4 1 IR

Step 3

Memory

300 1 9 4 0
301 5 9 4 1
302 2 9 4 1
940 0 0 0 3
941 0 0 0 2

CPU Registers

3 0 2 PC
0 0 0 5 AC
5 9 4 1 IR

3 + 2 = 5

Step 4

Memory

300 1 9 4 0
301 5 9 4 1
302 2 9 4 1
940 0 0 0 3
941 0 0 0 2

CPU Registers

3 0 2 PC
0 0 0 5 AC
2 9 4 1 IR

Step 5

Memory

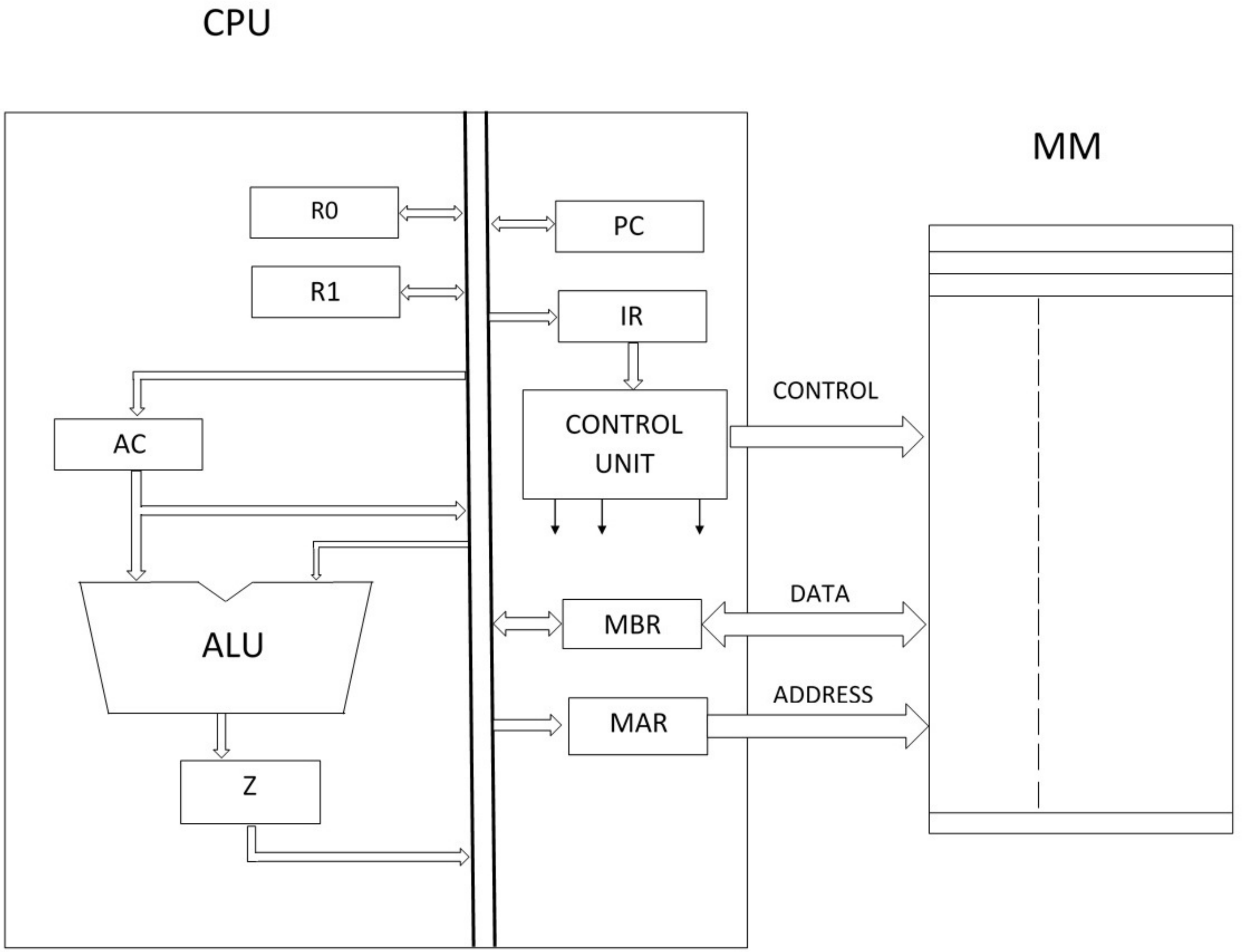
300 1 9 4 0
301 5 9 4 1
302 2 9 4 1
940 0 0 0 3
941 0 0 0 5

CPU Registers

3 0 3 PC
0 0 0 5 AC
2 9 4 1 IR

Step 6

CPU Organization



Machine Instruction

Machine	Instruction Format				Assembly
Instruction	Operation	Address			Code
1940	0001	1001	0100	0000	LDA M
5941	0101	1001	0100	0001	ADD M
2941	0010	1001	0100	0001	STA M

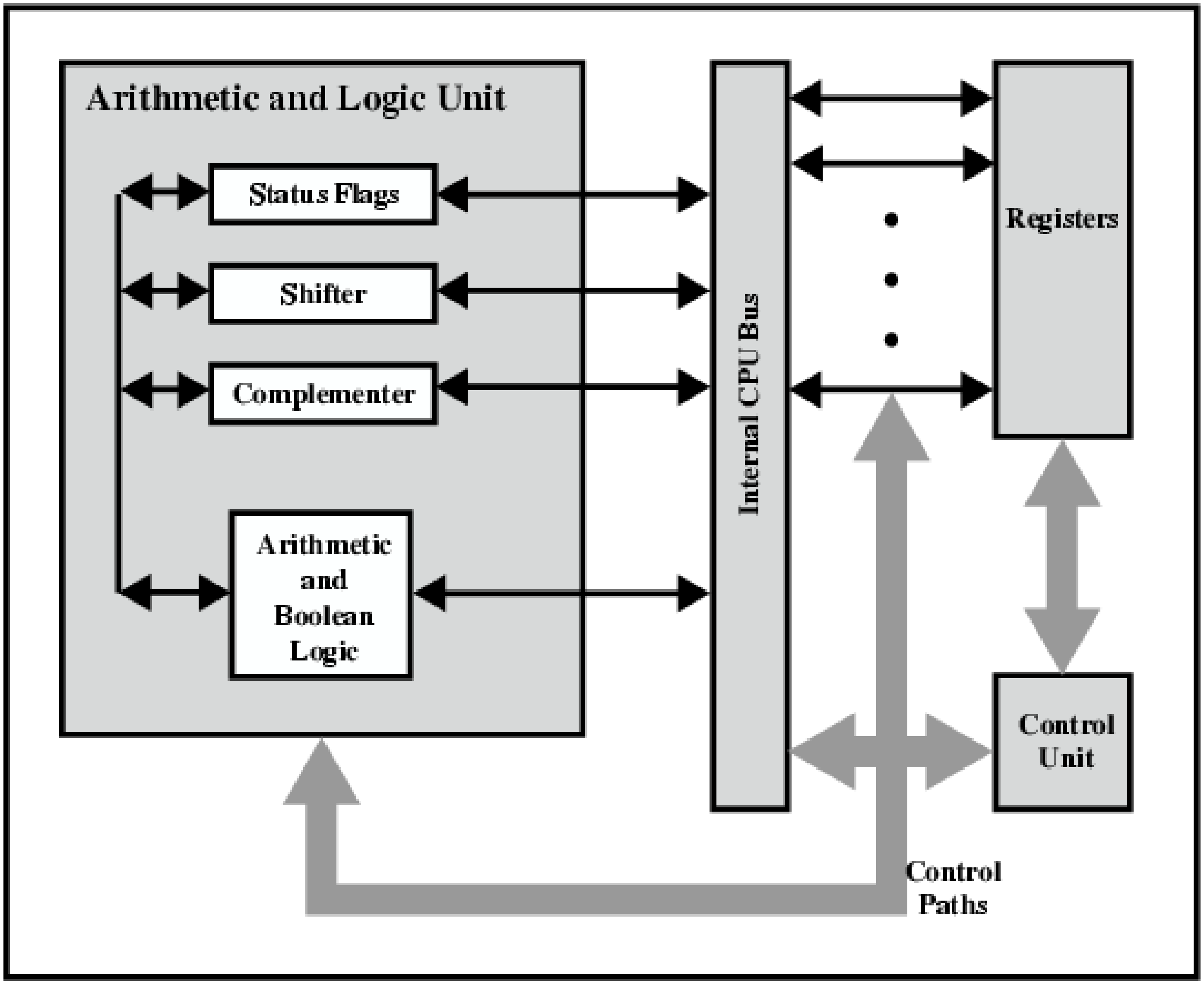
(LDA M) LOAD AC: Load the accumulator by the contents of memory location specified in the instruction

(ADD M) ADD AC: Add the contents of memory location specified in the instruction to accumulator and store the result in accumulator

(STA M) STORE AC: Store the contents of accumulator the memory location specified in the instruction

High Level Code	Assembly Code	Machine Code (HEX)
Y = X + Y	LDA X ADD Y STA Y	1940 5941 2941

CPU Internal Structure



Registers

- CPU must have some working space (temporary storage)
- Called registers
- Number and function vary between processor designs
- One of the major design decisions
- Top level of memory hierarchy

User Visible Registers

- General Purpose
- Data
- Address
- Condition Codes

How Many GP Registers?

- Between 8 - 32
- Fewer = more memory references
- More does not reduce memory references and takes up processor space
- Large enough to hold full address
- Large enough to hold full word
- Often possible to combine two data registers
  - C programming
  - double int a;
  - long int a;

Control & Status Registers

- Program Counter
- Instruction Decoding Register
- Memory Address Register
- Memory Buffer Register

Condition Code Registers

- Sets of individual bits
  - e.g. result of last operation was zero
- Can be read (implicitly) by programs
  - e.g. Jump if zero
- Can not (usually) be set by programs
- Needs for conditional instructions

Program Status Word

- A set of bits
- Includes Condition Codes
- Sign of last result
- Zero
- Carry
- Equal
- Overflow
- Interrupt enable/disable
- Supervisor

Function of Control Unit

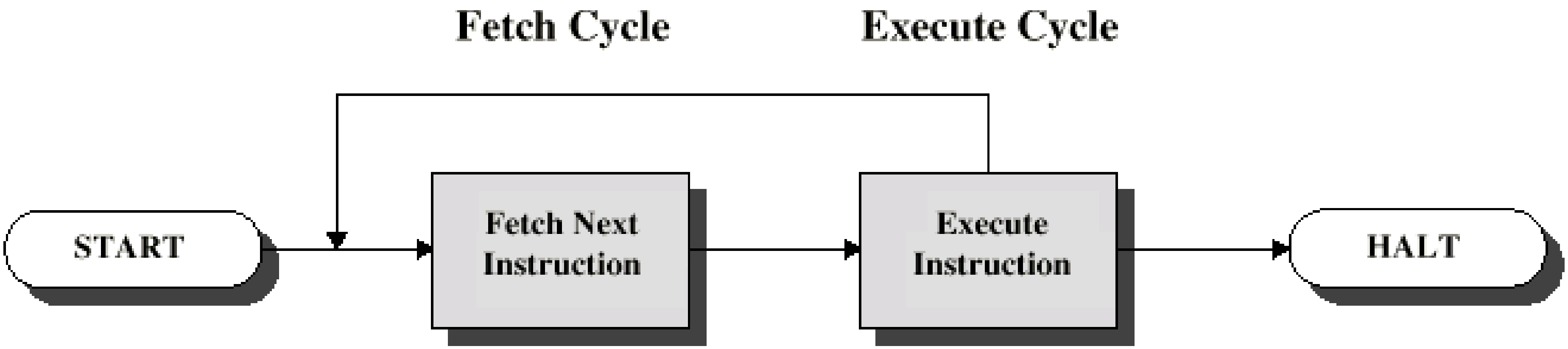
- For each operation a unique code is provided
    - e.g. ADD, MOVE
- A hardware segment accepts the code and issues the control signals

Components

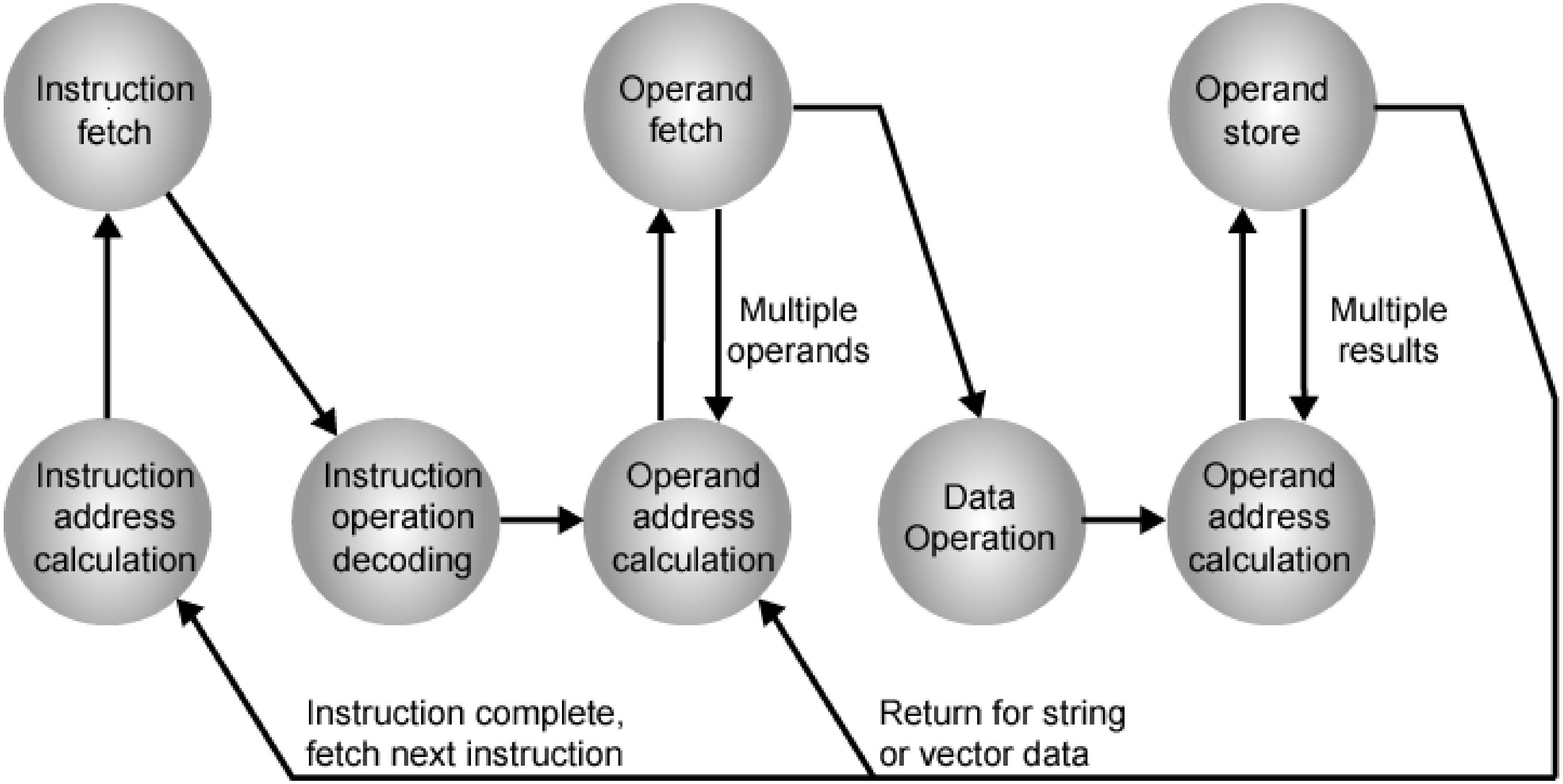
- The Control Unit and the Arithmetic and Logic Unit constitute the Central Processing Unit
- Data and instructions need to get into the system and results out
    - Input/output
- Temporary storage of code and results is needed
    - Main memory

Instruction Cycle

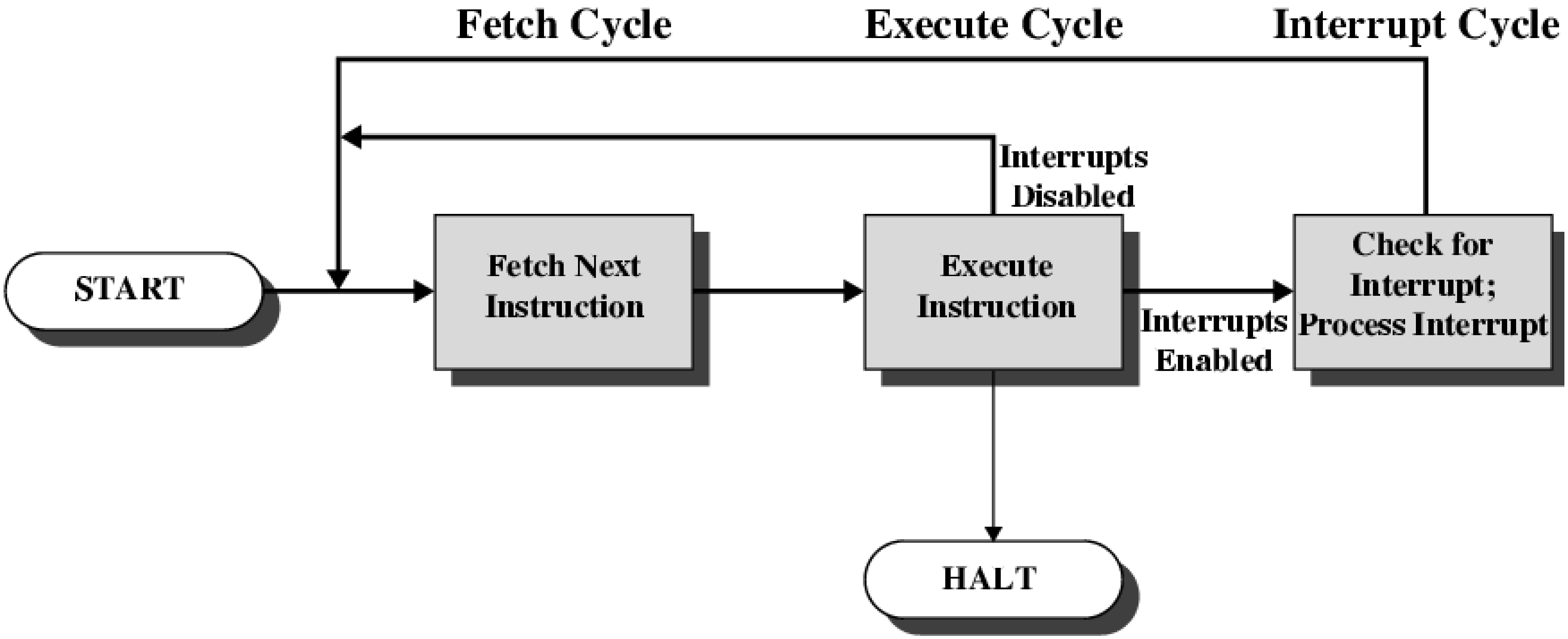
- Two steps:
  - —Fetch
  - —Execute



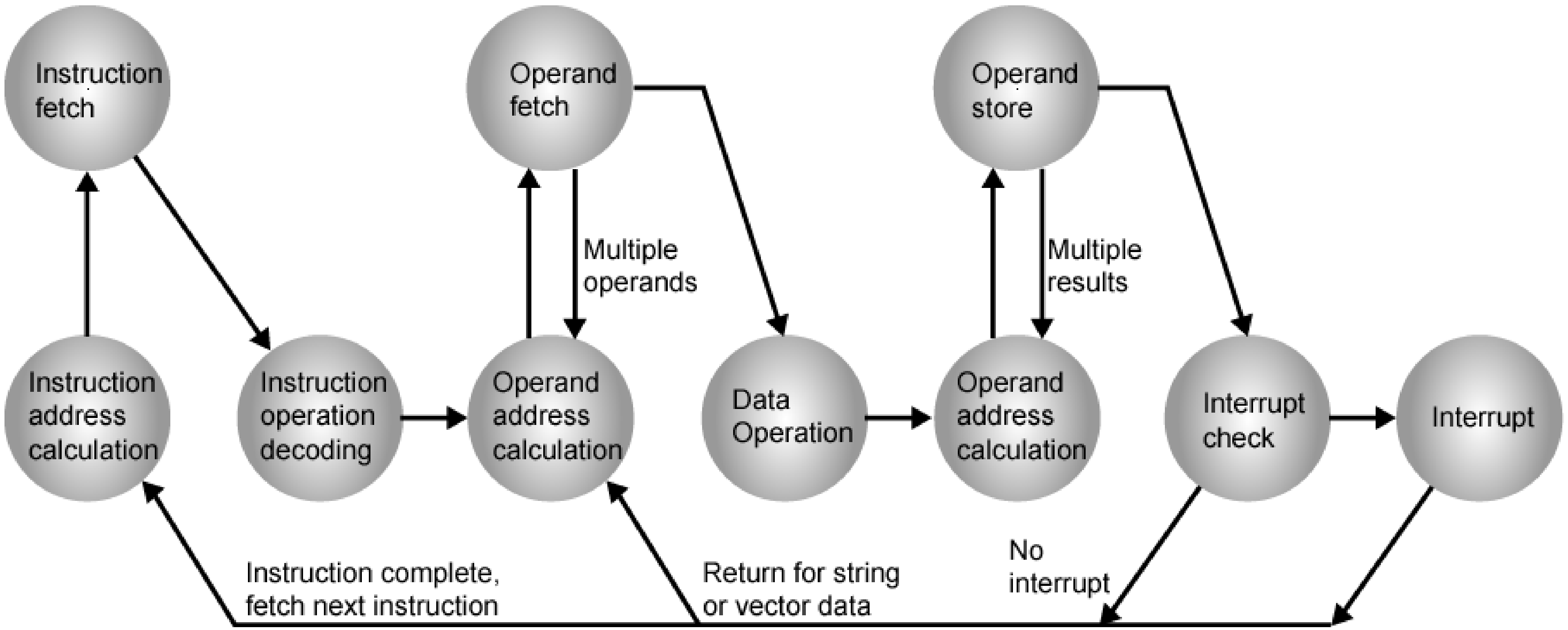
Instruction Cycle State Diagram


Instruction fetch
Operand fetch
Operand store
Multiple operands
Multiple results
Instruction address calculation
Instruction operation decoding
Operand address calculation
Data Operation
Operand address calculation
Instruction complete, fetch next instruction
Return for string or vector data

Instruction Cycle with Interrupts



Instruction Cycle (with Interrupts) - State Diagram


Instruction fetch
Operand fetch
Operand store
Multiple operands
Multiple results
Instruction address calculation
Instruction operation decoding
Operand address calculation
Data Operation
Operand address calculation
Interrupt check
Interrupt
Instruction complete, fetch next instruction
Return for string or vector data
No interrupt