CSC 222: Computer Organization & Assembly Language

5 – Processor Basics

Instructor: Ms. Nausheen Majeed

Outline

- Basic Operational Concepts
  - Processor Clock
  - Instruction Representation
  - Instruction Cycle

References

- ***Chapter 1,*** Ytha Yu and Charles Marut, "Assembly Language Programming and Organization of IBM PC"
- ***Chapter 3***, William Stallings, "Computer Organization & Architecture"
- ***Chapter 2***, Subrata Ghoshal, "Computer Organization & Architecture"

Basic Operational Concepts

Machine Instruction Elements

- Each instruction must have elements that contain the information required by the CPU for execution.
- These elements can be:
  - **Operation code**: Specifies the operation to be performed (e.g.. ADD, I/O). The operation is specified by a binary code, known as the operation code, or opcode.
  - **Source operand reference**: The operation may involve one or more source operands, that is, operands that are inputs for the operation.
  - **Result operand reference**: The operation may produce a result. Also called destination operand.
  - **Next instruction reference**: This tells the CPU where to fetch the next instruction.

Reference: http://cnx.org/content/m29425/latest/

Instruction Representation

- Within the computer, each instruction is represented by a sequence of bits.
- 16 bits instruction
  - 4 bit opcode, 6 bit operand 1, 6 bit operand 2
  - 4 bit opcode, 12 bit operand
- 32 bits instruction
- 64 bits instruction

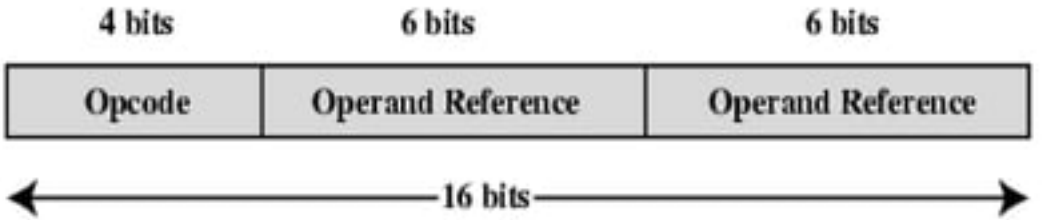
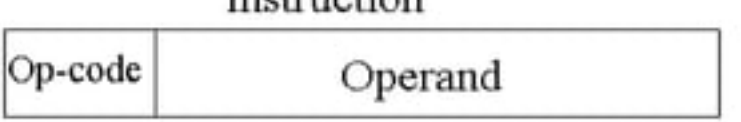


Figure: A Simple Instruction Format



Contd..

- Binary representations of machine instructions is difficult to remember.
- Use a symbolic representation of machine instructions.
- Opcodes are represented by abbreviations, called mnemonics, that indicate the operation. Common examples include:

ADD	Add
SUB	Subtract
MPY	Multiply
DIV	Divide
LOAD	Load data from memory
STOR	Store data to memory

Instruction Types

- **Data processing**: Arithmetic and logic instructions
- **Data storage**: Memory instructions
- **Data movement**: I/O instructions
- **Transfer of Control**: Test and branch instructions

No. of Addresses in an Instruction

- Three addresses
  - Operand 1, operand 2, result
- Two addresses
  - Source
  - Destination
- One addresses
  - Source or Destination
- Zero address
  - Zero-address instructions are applicable to a special memory organization, called a Stack. A stack is a last-in-first-out set of locations.

Types of Operands

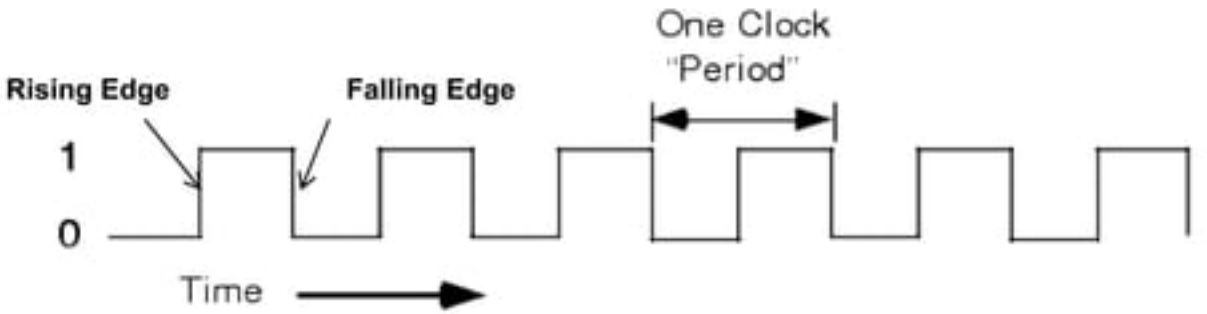
- Machine instructions operate on data.
- The most important general categories of data are:
  - Addresses
  - Numbers
  - Characters
  - Logical data

Basic Operations – Processor

- Execute the software by fetching instruction from memory
- Look for any external signal and react accordingly
  - Input signals from keyboard or mouse etc.

Processor Clock

- Heart of any processor
- Simple digital signals at equal time intervals
  - Alternate On Off states
- All activity within the CPU is synchronized with the edges (rising or falling) of this clock signal.



Reference: http://www.plantation-productions.com/Webster/www.artofasm.com/Linux/HTML/SystemOrganizationa4.html

Program Counter (a.k.a. Binary Counter)

- With every falling edge or rising edge (depending upon processor) of clock signal, the counter is incremented by one.
- Width varies from processor to processor
- The contents of PC are used as target address for the memory area

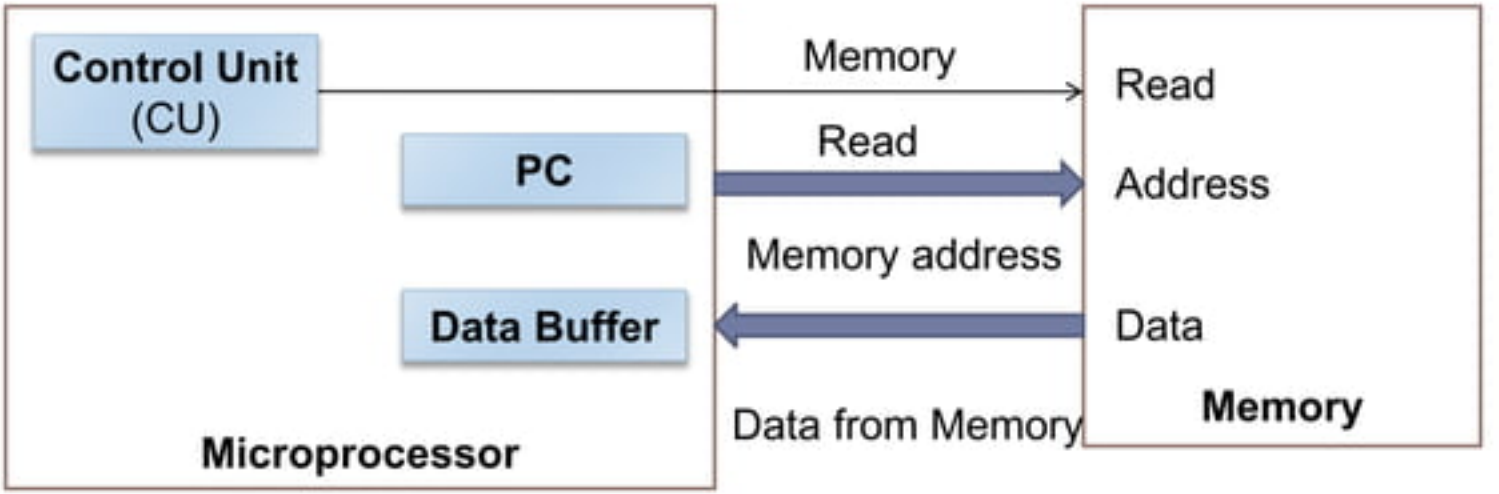
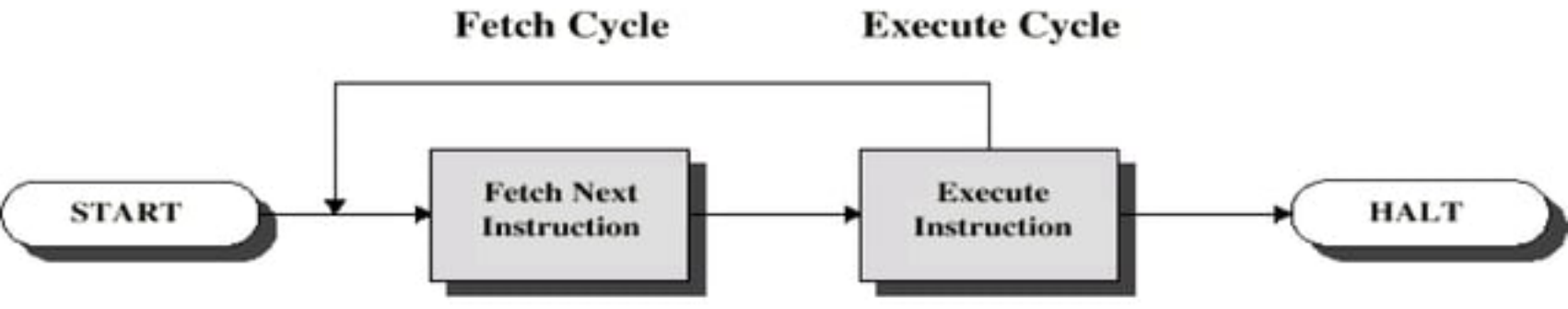


Figure: Reading from memory

Basic Instruction Cycle

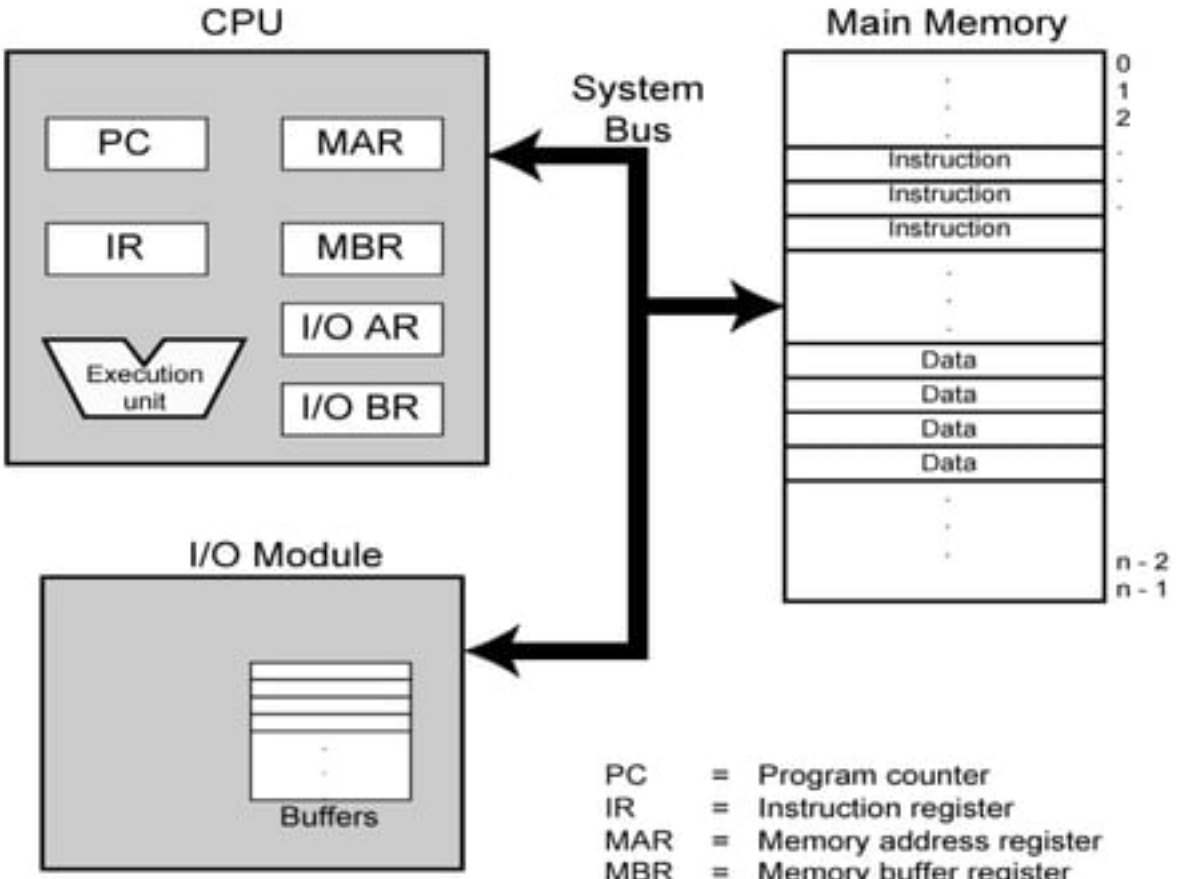
- Fetch → Decode → Execute
- Fetch
  1. Fetch an instruction from memory
  2. Decode the instruction to determine the operation
  3. Fetch data from memory if necessary
- Execute
  4. Perform the operation on the data
  5. Store the result in memory if needed



Contd..

- Internal CPU Registers used in instruction cycle:
  - Program Counter (PC) = Address of instruction
  - Instruction Register (IR) = Instruction being executed
  - Accumulator (AC) = Temporary Storage

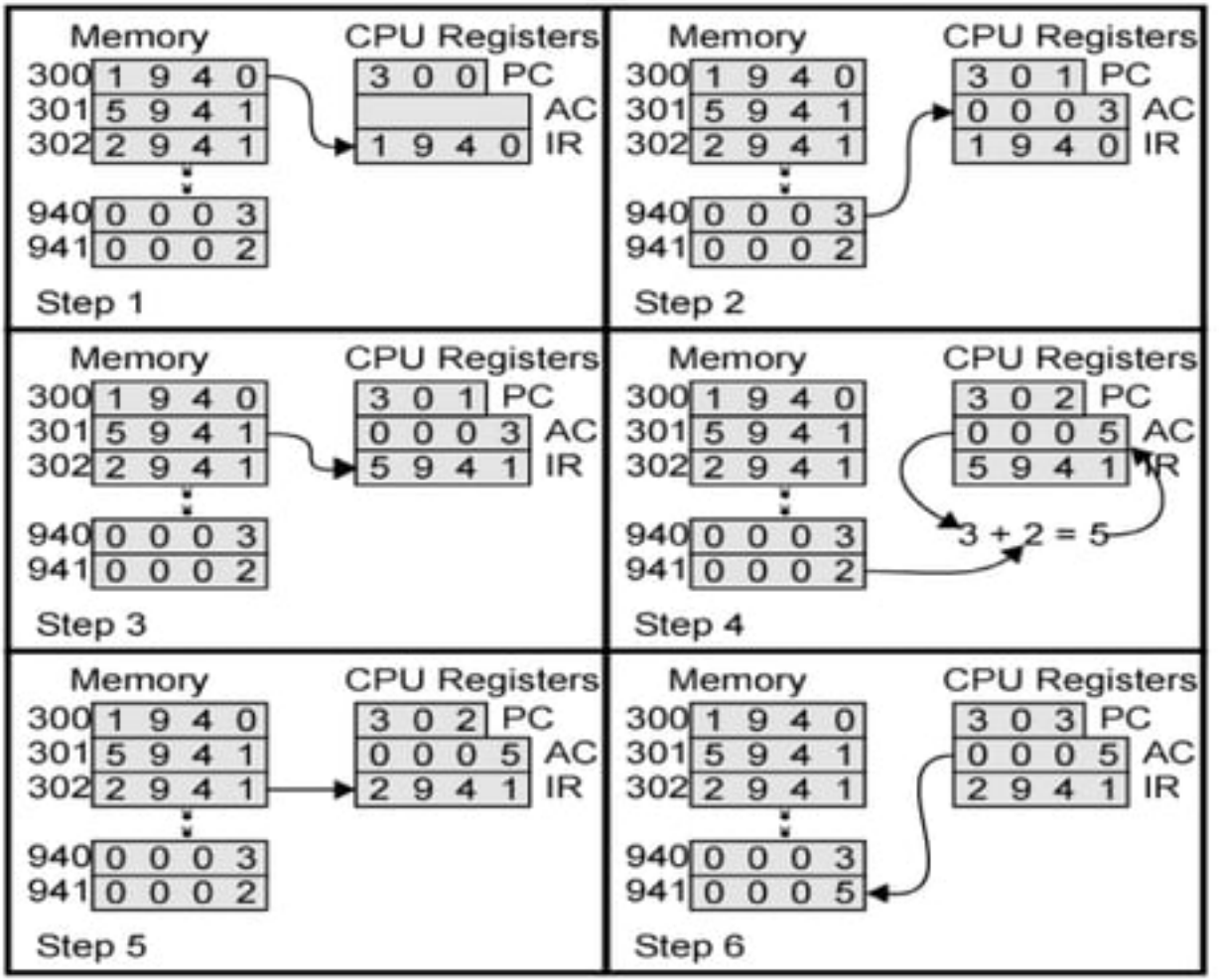
Computer Components: Top Level View Contd..



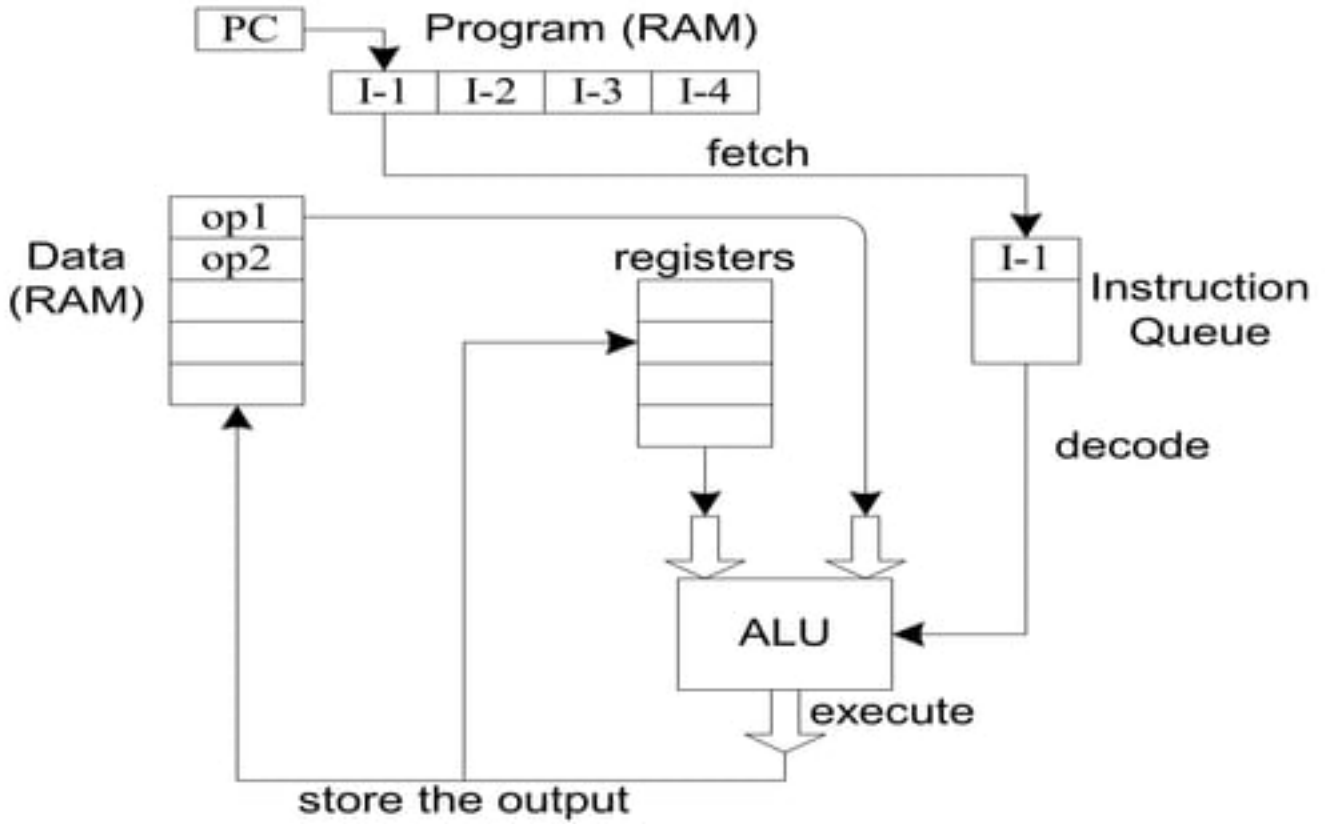
Detailed Steps

- Address in the Program Counter register
  - Program Counter (PC) holds address of next instruction to fetch
- Fetch the instruction from the memory
- Increment the Program Counter
  - Unless told otherwise
- Instruction loaded into Instruction Register (IR)
- Decode the type of instruction
- Fetch the operands
- Execute the instruction
- Store the results

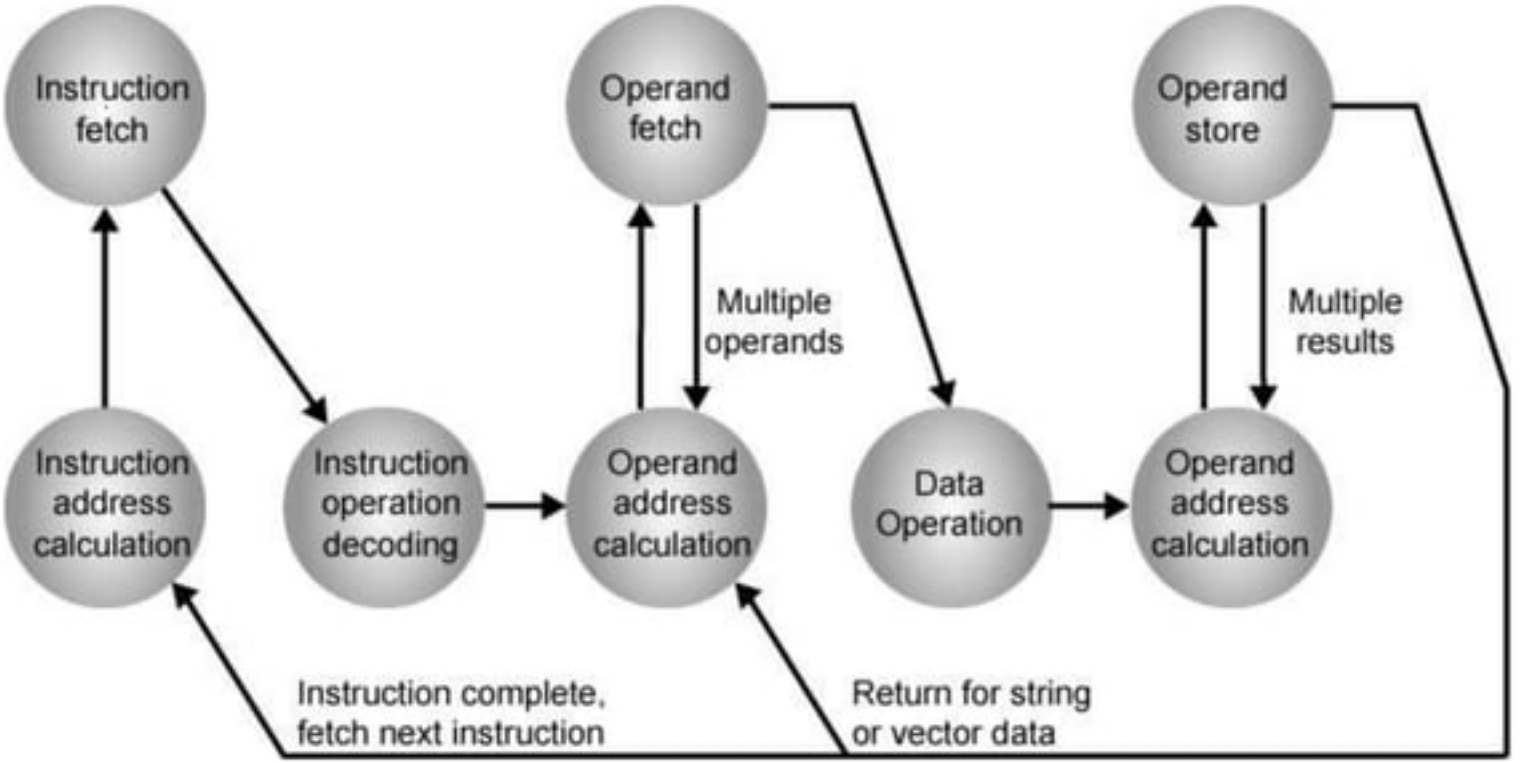
Example Program Execution



Instruction Execution Cycle



Instruction Cycle State Diagram



Contd..

- Instruction Fetch
  - Read instruction from memory into processor
- Instruction Operation Decoding
  - Determine the type of operation to be performed and operand(s) to be used.
- Operand Address Calculation
  - If operation involves reference to an operand in memory or I/O, then determine the address of operand.
- Operand Fetch
  - Fetch from memory or read from I/O
- Data Operation
  - Perform the operation
- Operand Store
  - Write into memory or out to I/O if required