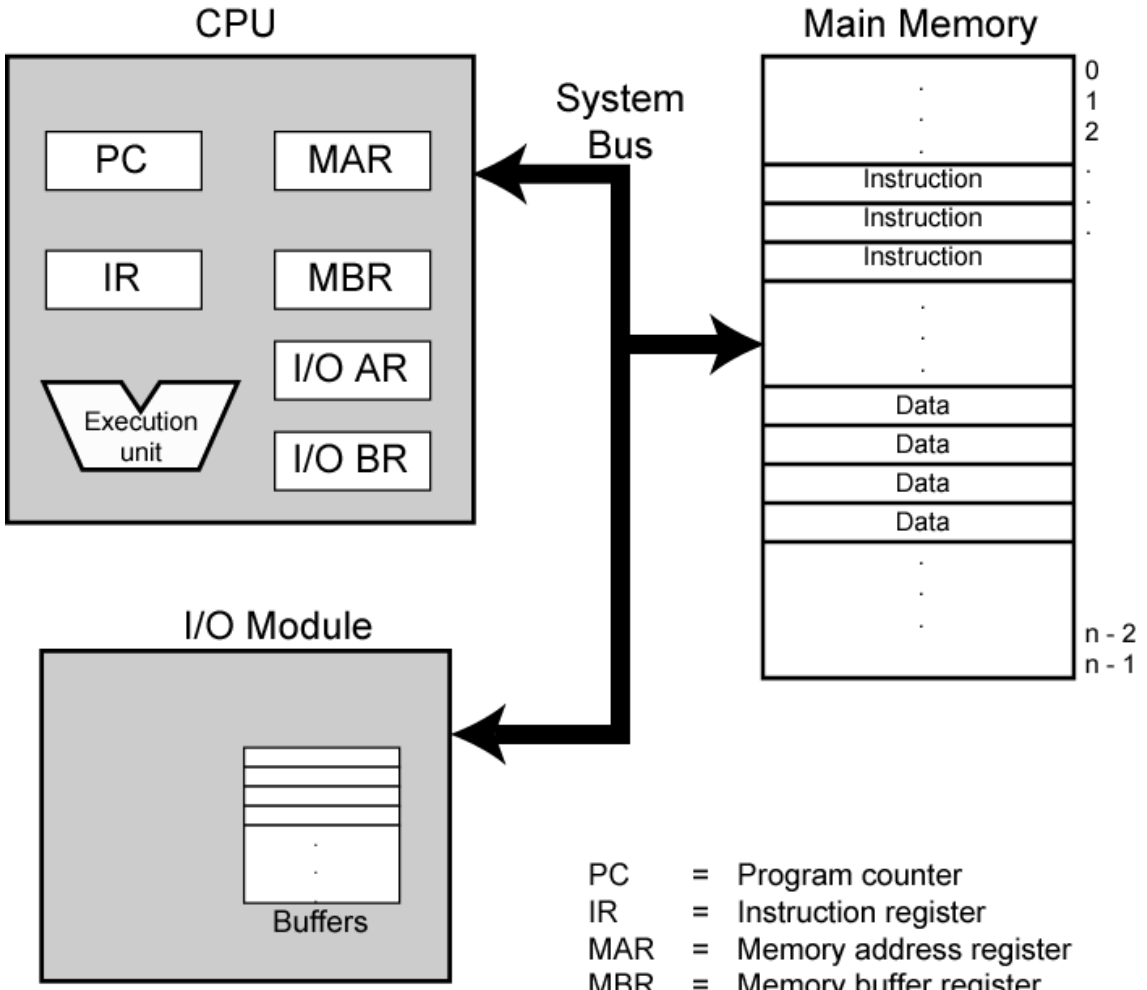
CPU Functions

ICT 2203 Computer Architecture

Partially based on Computer Organization and Architecture 8th Edition by William Stallings

Computer Components: Top Level View



Components

- The Control Unit and the Arithmetic and Logic Unit
  - 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

Central Processing Unit (CPU)

- Carries out the program's instructions!
  - Operates on data it finds in the computer's memory.
- Includes all binary circuits that carry out arithmetic & logic operations, reduced to a single IC.
  - Also called Microprocessor
- CPU has four key parts that we will examine:
  - Control Unit
  - Arithmetic & Logic Unit
  - Registers
  - Clock
  - And, of course, wires that connect everything together.

Central Processing Unit (CPU)

- Control Unit (CU):
  - circuitry for coordinating machine's activities. Controls sequence of operations.
- Arithmetic & Logic Unit (ALU):
  - circuitry to perform data manipulation (arithmetic & logic).
- Registers:
  - Temporary storage areas. Holds information applicable to the current operation.
- Clock:
  - Triggers start and stop of all CPU operations. (heartbeat)

Program Concept

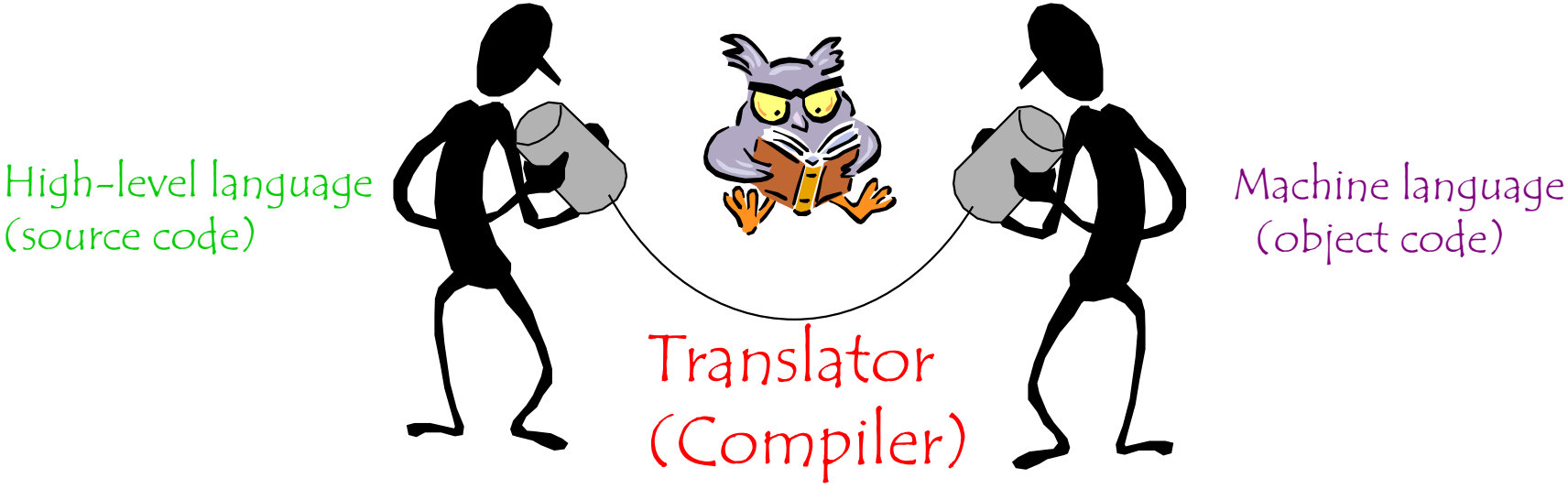
- Hardwired systems are inflexible
- General purpose hardware can do different tasks, given correct control signals
- Instead of re-wiring, supply a new set of control signals
- For each operation a unique code is provided
  - e.g. ADD, MOVE
- A common hardware segment accepts the code and issues the control signals
- That is a programmable computer!

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

Programming for a CPU

- What kind of code do programmers use?
- What kind of code does a CPU understand?
- So, what has to occur *before* the CPU can *execute* a source-code program?



Instruction Set

- CPUs support a set of very simple instructions that typically fall into the following categories:
  - Data movement (load, store, copy…)
  - Arithmetic/logical (add, subtract, compare..)
  - Program control (branch, halt…)
- Very primitive commands (operations) executed by the CPU
  - logical structure
- These commands are implemented as electronic binary circuits which can transform the 0s and 1s.
  - physical structure

Instruction Set

- Instructions are given to the processor in the form of a program, so it knows what circuits to use, in what order; and from where the data should be read or to where it should be stored.

Instruction	Meaning
STO	Store data in a particular memory location
ADD	Add two numbers together
SUB	Subtract one number from another
MUL	Multiply two numbers
DIV	Divide two numbers
INC	Increment a number by adding 1
CMP	Compare two numbers to see if they are equal
JMP	Jump to a specific position in the instruction code

Machine language

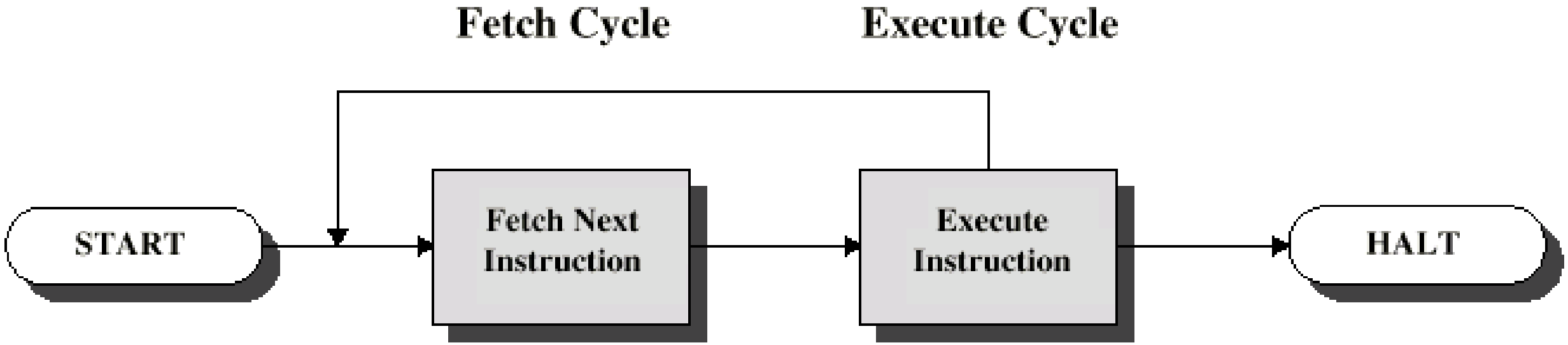
- Instructions are stored and processed in machine language, also called microcode or machine code.
- A program consists of a sequence of instructions.
- Each instruction contains a fixed-length instruction code that:
  - Identifies the operation to perform: op code
  - Tells the CPU how to determine the operands
- For example:
  - assume a 4-bit op-code + two 6-bit operands = one 16 bit instruction

Machine language

- Machine language consists solely of bit patterns.
- Machine language bit patterns are based directly on CPU's instruction set (on its binary circuits).
- CPU chip: designed to recognize certain bit patterns as representing certain instructions, which correspond directly to certain available binary circuits.
- Example: An ADD instruction in a 16-bit machine language:
  - 0101 110011 111100
  - Op-code: Operands: *(RAM or Register addresses)*
  - 0101 110011 111100

Instruction Cycle

- Two steps:
  - Fetch
  - Execute



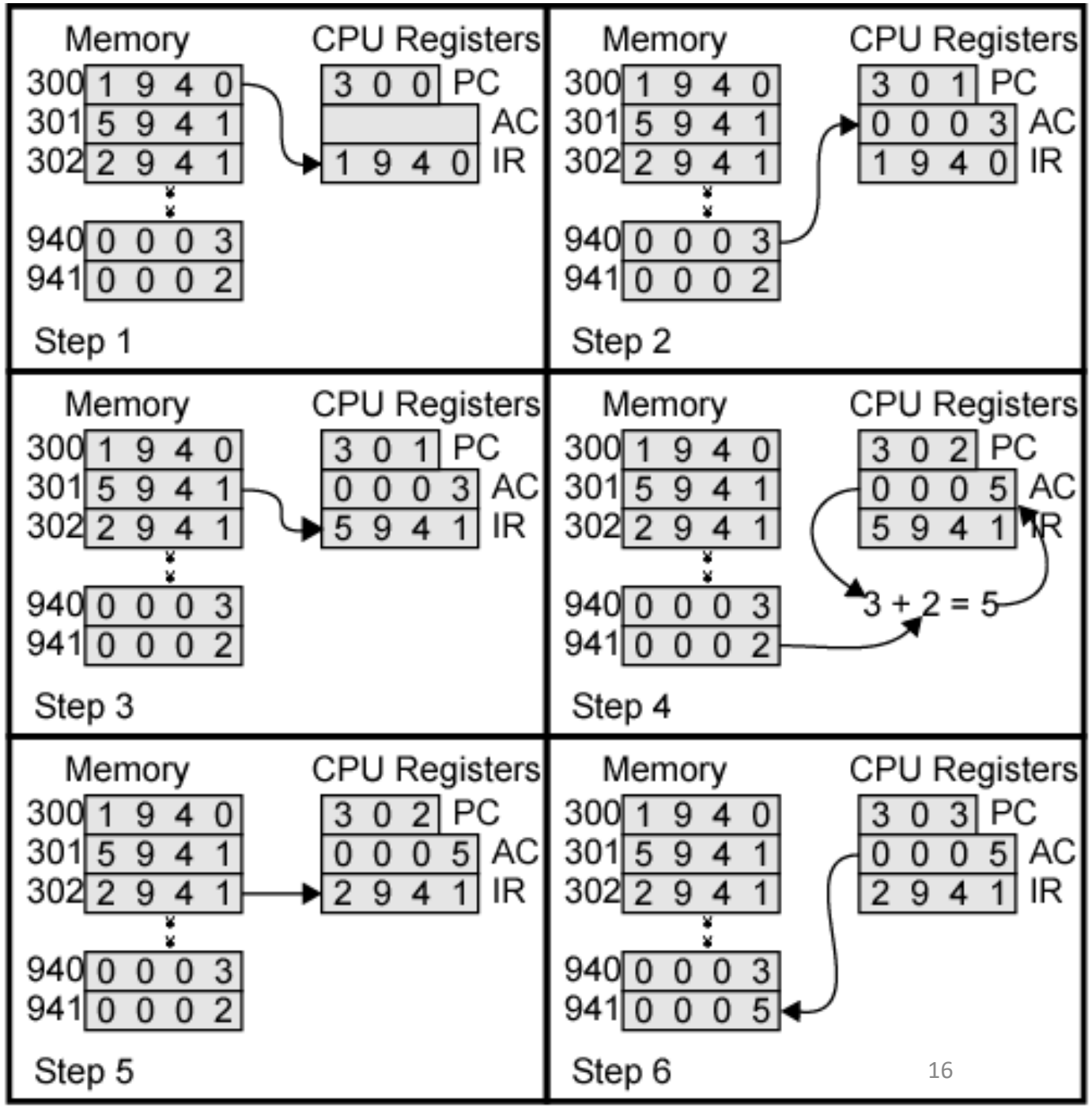
Fetch Cycle

- 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

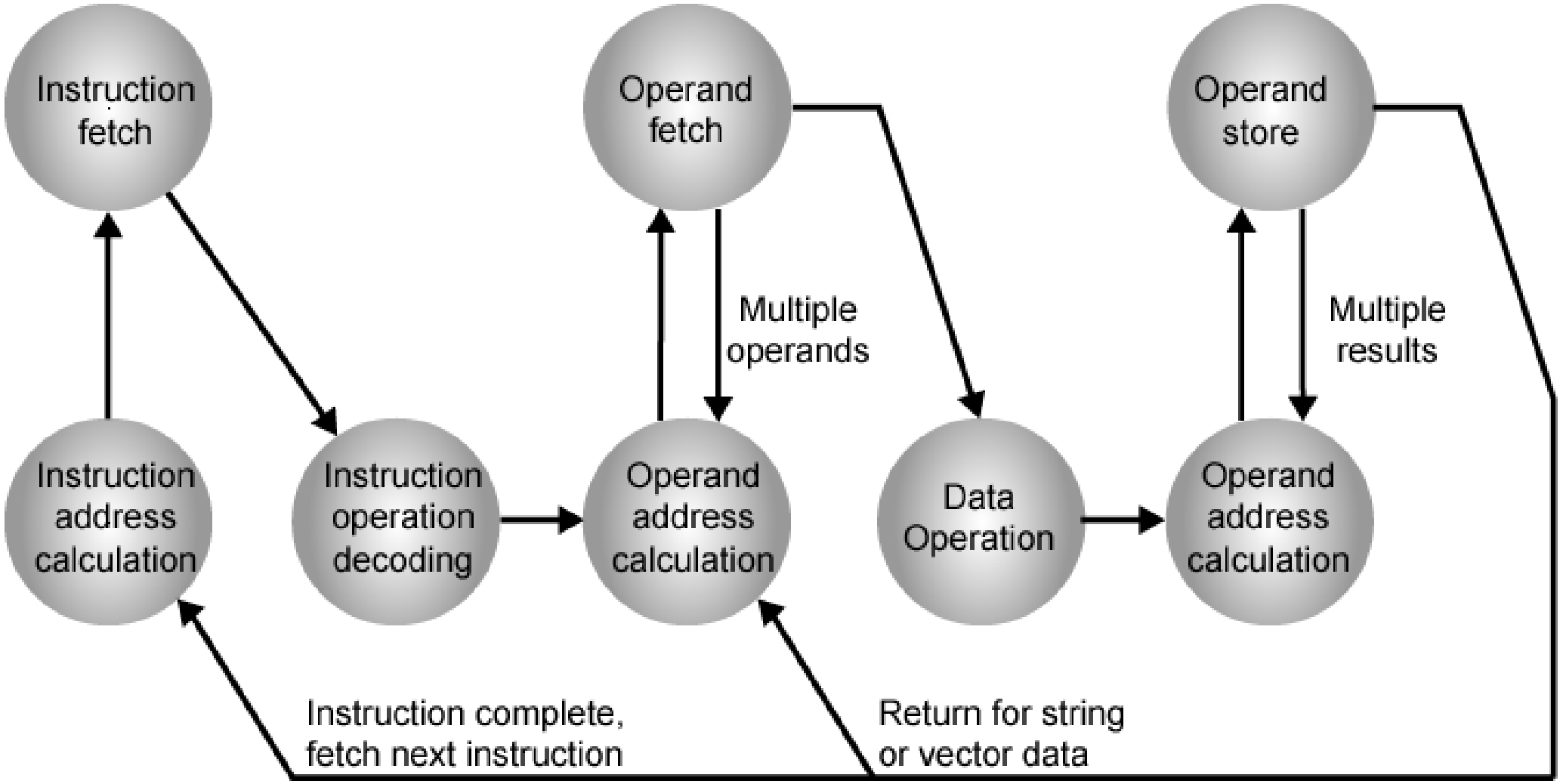
Execute Cycle

- 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

Example of Program Execution



Instruction Cycle State Diagram



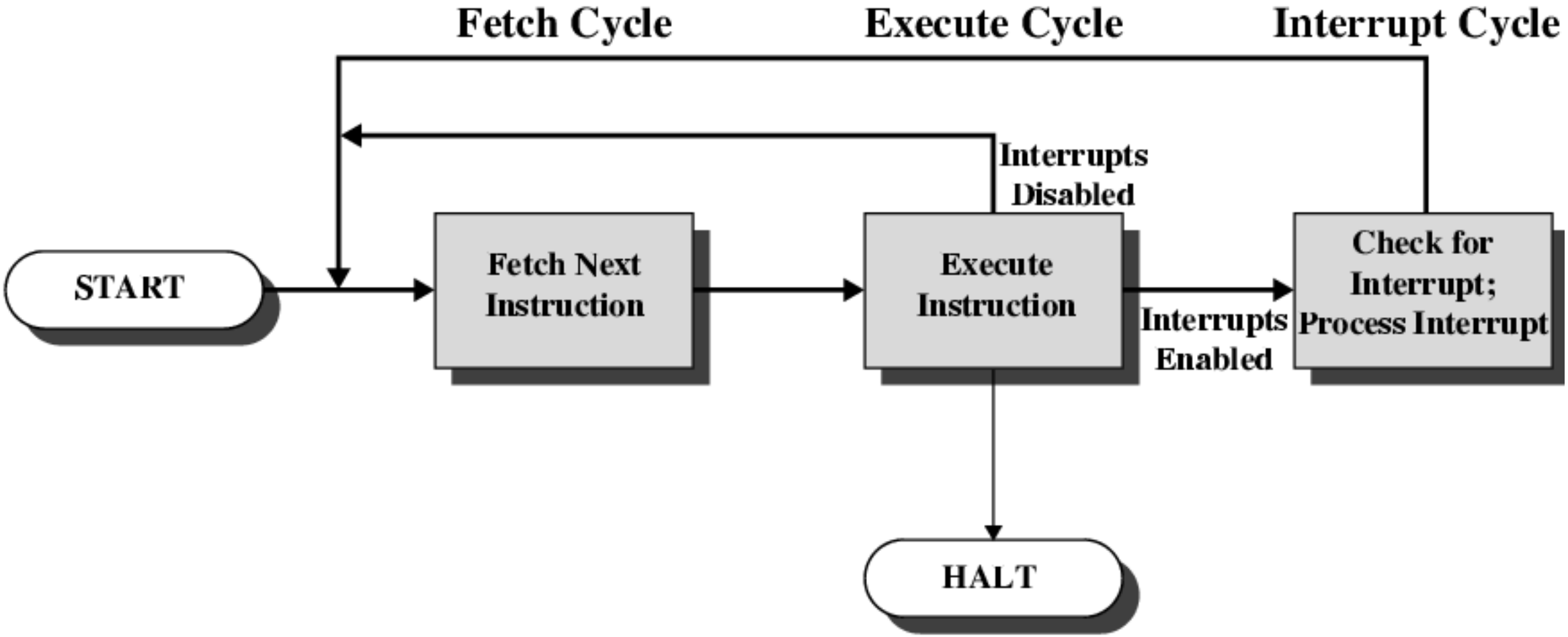
Interrupts

- Mechanism by which other modules (e.g. I/O) may interrupt normal sequence of processing
- Program
  - e.g. overflow, division by zero
- Timer
  - Generated by internal processor timer
  - Used in pre-emptive multi-tasking
- I/O
  - from I/O controller
- Hardware failure
  - e.g. memory parity error

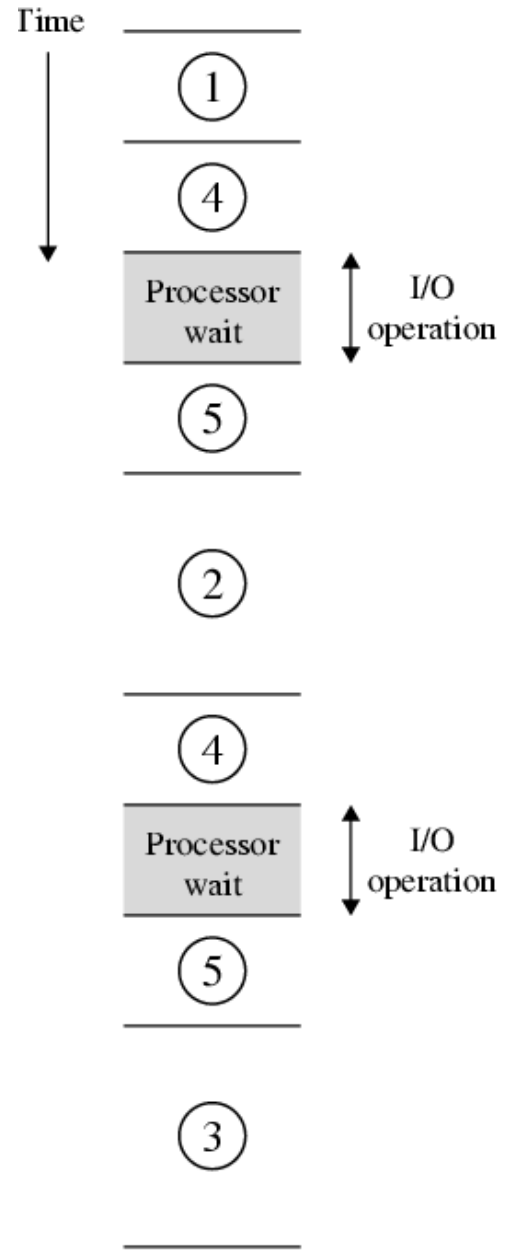
Interrupt Cycle

- Added to instruction cycle
- Processor checks for interrupt
  - Indicated by an interrupt signal
- If no interrupt, fetch next instruction
- If interrupt pending:
  - Suspend execution of current program
  - Save context
  - Set PC to start address of interrupt handler routine
  - Process interrupt
  - Restore context and continue interrupted program

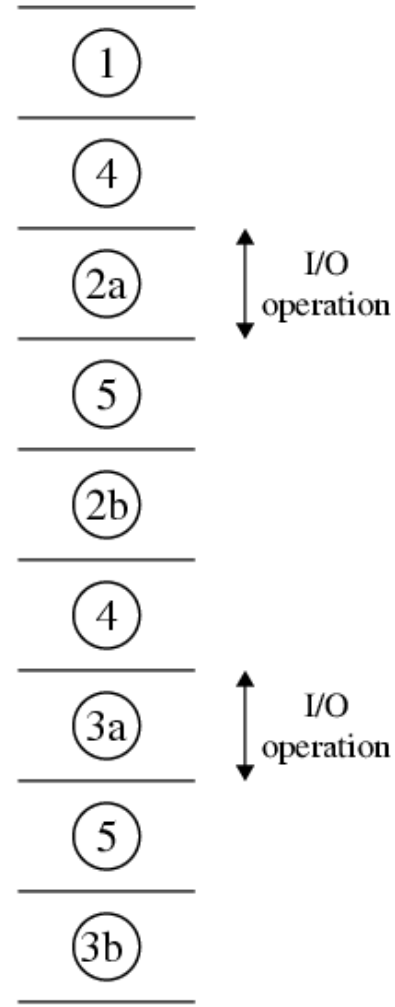
Instruction Cycle with Interrupts



Program Timing Short I/O Wait

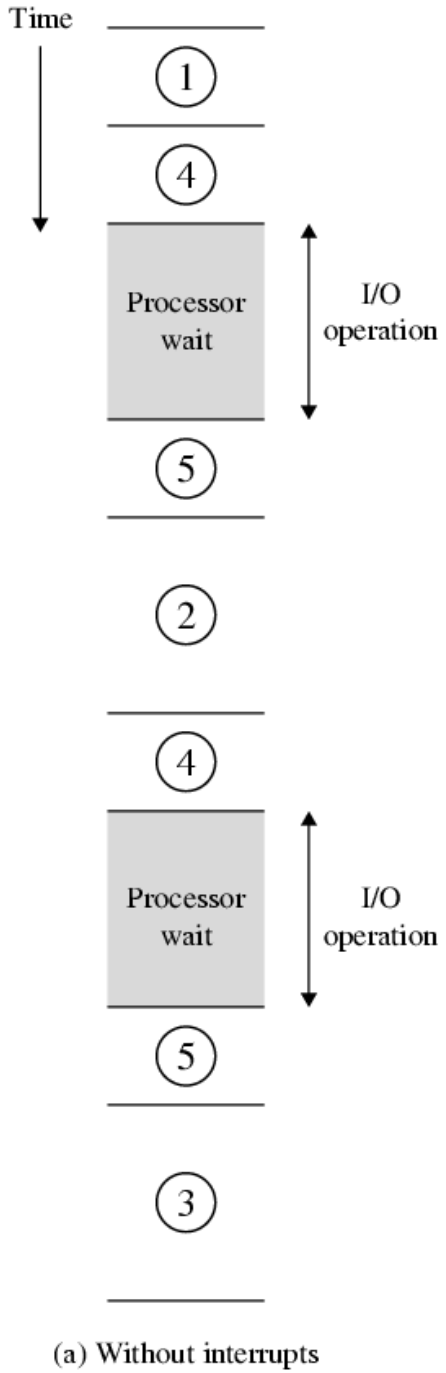


(a) Without interrupts

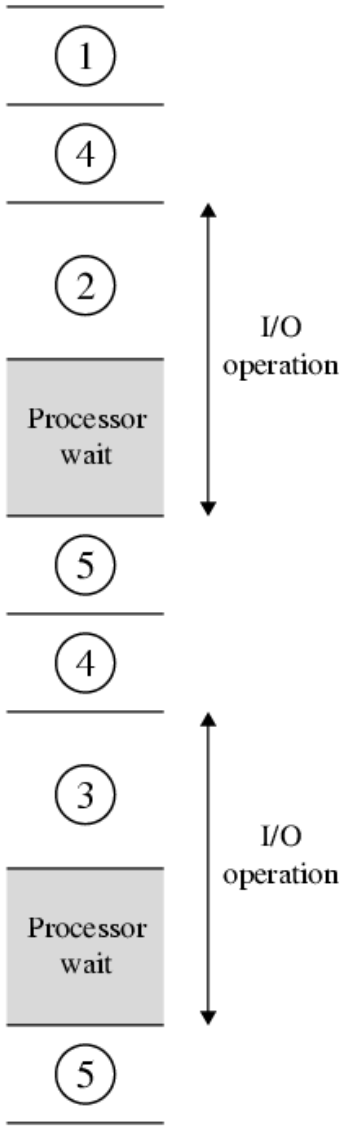


(b) With interrupts

Program Timing
Long I/O Wait

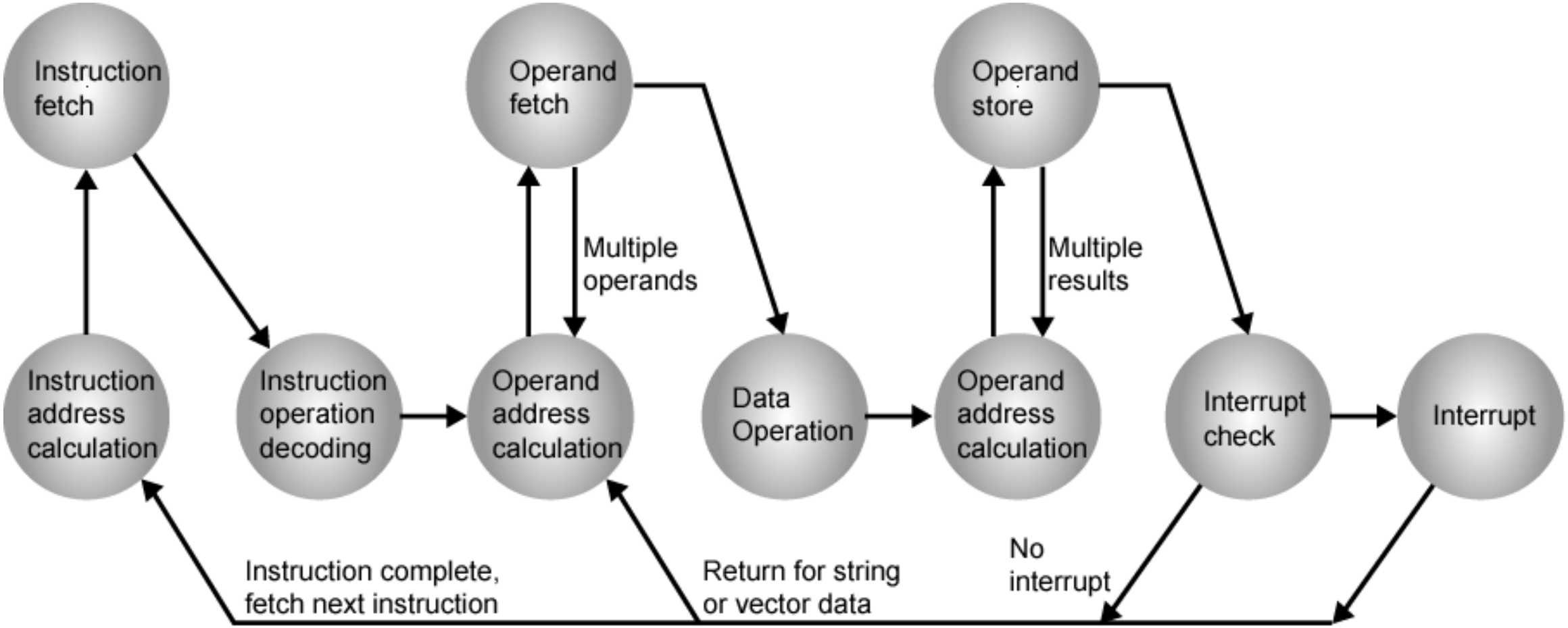


(a) Without interrupts



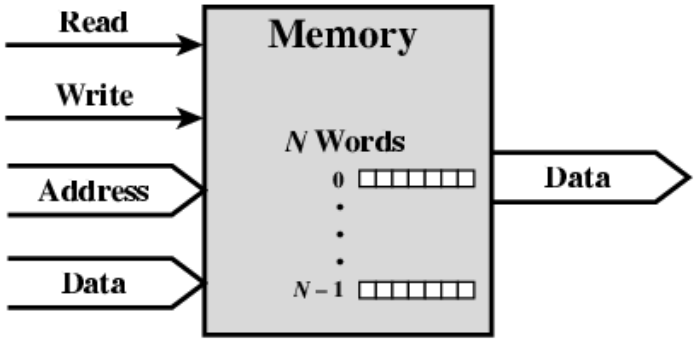
(b) With interrupts

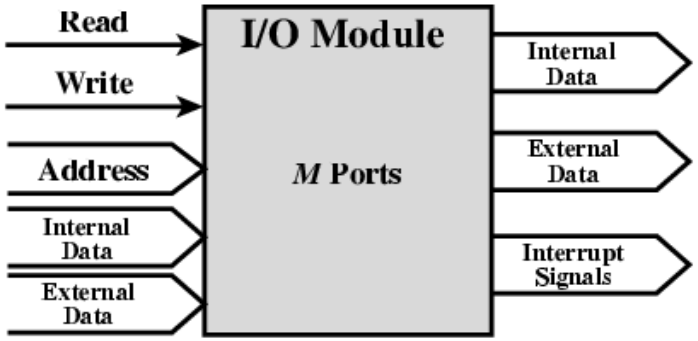
Instruction Cycle (with Interrupts)

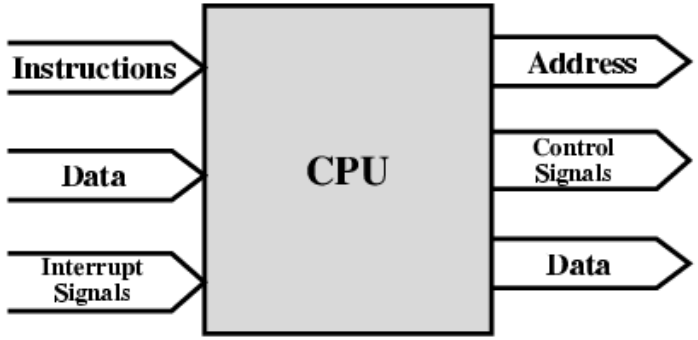


Connecting

- 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

- 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
- Receive control signals from computer
- Send control signals to peripherals
  - e.g. spin disk
- Receive addresses from computer
  - e.g. port number to identify peripheral
- Send interrupt signals (control)

CPU Connection

- Reads instruction and data
- Writes out data (after processing)
- Sends control signals to other units
- Receives (& acts on) interrupts

Buses

- There are a number of possible interconnection systems
- 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
- Power lines may not be shown
- Single and multiple BUS structures are most common
  - Control/Address/Data bus (PC)

Buses

- Data Bus:
  - Carries data
    - Both “data” and “instruction”
  - Width is a key determinant of performance
    - 8, 16, 32, 64 bit
- Control Bus:
  - Control and timing information
    - Memory read/write signal
    - Interrupt request
    - Clock signals
- 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

Next:

Memory