**Fundamentals of Computer Organization**

* Readings: 1.1 – 1.7
* Categories of computers:
  + General-purpose computers – suitable for processing a wide range of applications (e.g. PC)
  + Application-specific computers – suitable for processing a single type of application
* Types of computer systems:
  + Personal computers – used by individuals in home, school, office
  + Workstations – powerful desktops designed for industrial use by an individual; e.g. for engineering applications
  + Enterprise systems/servers/mainframes – large-scale business data processing/storage; usually scalable
  + Supercomputers – specifically designed to perform large-scale numerical calculations; e.g. weather forecasting, fluid simulations, cryptanalysis
  + Embedded computers – designed to control other systems; e.g. cars, cameras
  + Cloud computers – geographically distributed storage & computing resources connected via the Internet w/ lightweight terminals
* **Programmable computer**
  + Can store a sequence of instructions
  + Can execute the stored sequence of instructions
  + Can conditionally select a path of execution (control transfer)
* History …
* **Digital computer** uses electronic circuits to implement a programmable computer
  + Accepts digitalized input, processes it according to internally stored instructions, and produces the resulting output
  + 5 types of functional units:
    - Input unit + output unit → input/output systems
    - Memory unit → memory system
    - Arithmetic & logic unit + control unit → CPU
* **I/O systems**
  + Input – environment → computer
  + Output – computer → environment
* Memory device interface
  + Control signals (clock, memory enable, read/write), address [9 downto 0] → RAM → data [7 downto 0]
* **Memory hierarchy**
  + Secondary memory – large & slow
    - E.g. disks, flash memory
  + Primary memory – medium-sized, fast
    - E.g. RAM
  + Caches – smaller, faster; on-chip and off-chip
  + Registers – on-chip
  + Disk ←(software managed)→ main memory ←(hardware managed)→ L2$ ←(hardware managed)→ I$, D$ ←(compiler managed)→ processor + registers
* **Central processing unit** – consists of a central control unit & ALU
  + Contains registers to store immediate results and system state
  + Controlled using a sequence of simple instructions
  + **Arithmetic logic unit**
    - Performs arithmetic operations (e.g. add/subtract) & logic operations (e.g. bitwise AND/OR)
  + **Control unit**
    - Fetches instructions from memory & orchestrates instruction execution
    - Can be centralized or distributed
* Interconnection network
  + - **Bus** – set of signal wires whose use may be shared by multiple functional units over time
    - Implemented using tri-state drivers
      * Only one functional unit may write signal to the bus at one time (bus driving)
      * Bus conflict occurs if two functional units drive the bus at the same time
    - Bus structures:
      * Single system bus for all systems
      * Independent buses for each system
        + Some can be slower/faster depending on needs
      * Multiple units can write signal at the same time
* Connecting CPU to memory device
  + Control unit & ALU – implements instruction execution cycle
    - Sends control signals to memory
  + Program counter (PC) – holds memory address of next instruction
  + Instruction register (IR) – holds current instruction
  + General-purpose registers – hold data & addresses
  + Memory data register (MDR) – temporary storage for data
    - Before a store/write, set to the data value to be written
    - After a fetch/read, set to the data value read
  + Memory address register (MAR) – temporary storage for address
    - Before (any type of) memory transaction, set to the address value
  + Movement of instructions is determined by PC
  + Movement of data is determined by instructions
  + Status register (SR) – stores flags about the status of the processor
  + Control register (CR) – controls the behaviour of the processor
  + Only PC, SR, CR can be modified by instructions
  + IR, MDR, MAR are managed by the control unit
  + Process-memory interface:
    - Read – sends address of read location & Read command to memory, wait for retrieval of word, places it in processor register
    - Rite – sends address of write location & word to be written & Write command to memory
* Connecting CPU to I/O device
  + Memory-mapped I/O – treat I/O units as memory
* **Executing an instruction**:
  + Get address of instruction to execute from PC → send address to memory and issue Read command
  + Get requested instruction word from memory → store in IR
  + Increment PC
  + Execute instruction (e.g. with ALU, memory Read/Write commands, etc.)