# CSE211

# Computer Organization and Design

- *Register Transfer Language*
- *Register Transfer*

# Overview

- **Register Transfer Language**
- **Register Transfer**
- Bus and Memory Transfers
- Logic Micro-operations
- Shift Micro-operations
- Arithmetic Logic Shift Unit

# Register Transfer Language

- **Combinational and sequential circuits can be used to create simple digital systems.**
- **These are the low-level building blocks of a digital computer.**
- **Simple digital systems are frequently characterized in terms of**
  - **the registers they contain, and**
  - **the operations that are performed on data stored in them**
- **The operations executed on the data in registers are called micro-operations e.g. shift, count, clear and load**

# Register Transfer Language

**Internal hardware organization of a digital computer :**

- **Set of registers and their functions**
- **Sequence of microoperations performed on binary information stored in registers**
- **Control signals that initiate the sequence of micro-operations (to perform the functions)**

- The operations executed on the data in registers are called…….

A) minor-operations

B) micro-operations

C) macro-operations

D) None of the above

# Register Transfer Language

- **Rather than specifying a digital system in words, a specific notation is used, Register Transfer Language**
- **The symbolic notation used to describe the micro operation transfer among register is called a register transfer language**
- **For any function of the computer, the register transfer language can be used to describe the (sequence of) micro-operations**
- **Register transfer language**
  - **A symbolic language**
  - **A convenient tool for describing the internal organization of digital computers in concise/precise manner.**
  - **Can also be used to facilitate the design process of digital systems.**

# Register Transfer

- **Registers are designated by capital letters, sometimes followed by numbers (e.g., A, R13, IR)**
- **Often the names indicate function:**
  - **MAR - memory address register**
  - **PC - program counter**
  - **IR - instruction register**

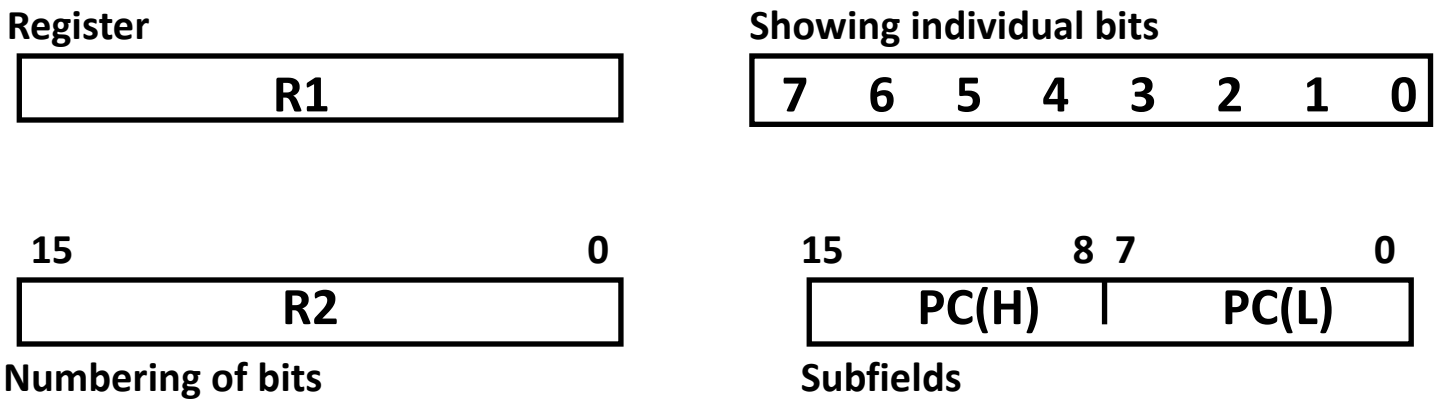
- **Registers and their contents can be viewed and represented in *various ways***
  - **A register can be viewed as a single entity:**

# Register Transfer

- **Designation of a register**

  **- a register**
  **- portion of a register**
  **- a bit of a register**

- **Common ways of drawing the block diagram of a register**



# Register Transfer

- **Copying the contents of one register to another is a register transfer**
- **A register transfer is indicated as**

$R2 \leftarrow R1$

  - **In this case the contents of register R1 are copied (loaded) into register R2**
  - **A simultaneous transfer of all bits from the source R1 to the destination register R2, during one clock pulse**
  - **Note that this is a non-destructive; i.e. the contents of R1 are not altered by copying (loading) them to R2**

- The symbolic notation used to describe the micro operation transfer among register is called…….

A) Register Accessed Language

B) Register Processed Language

C) Register Transfer Language

D) None of the above

# Register Transfer

- **A register transfer such as**

  **R3 ← R5**

  **Implies that the digital system has**

  - **the data lines from the source register (R5) to the destination register (R3)**
  - **Parallel load in the destination register (R3)**
  - **Control lines to perform the action**

# Control Functions

- **Often actions need to only occur if a certain condition is true**
- **This is similar to an "if" statement in a programming language**
- **In digital systems, this is often done via a *control signal*, called a *control function***
  - **If the signal is 1, the action takes place**
- **This is represented as:**

**P: R2 ← R1**

**Which means "if P = 1, then load the contents of register R1 into register R2", i.e., if (P = 1) then (R2 ← R1)**