CSE211

Computer Organization and Design

Lecture : 3 **Tutorial: 1** **Practical: 0** **Credit: 4**

LOVELY PROFESSIONAL UNIVERSITY

Overview

⍰ **Register Transfer Language**

⍰ **Register Transfer**

⍰ Bus and Memory Transfers

⍰ Logic Micro-operations

⍰ Shift Micro-operations

⍰ Arithmetic Logic Shift Unit

A	shift	register	is defined	as ______							
a)	The	register	capable of	shifting information	to	another	register				
b)	The	register	capable of	shifting information	eit	her to	the right	or	to	the	left
c)	The	register	capable of	shifting information	to	the rig	ht only				
d)	The	register	capable of	shifting information	to	the lef	t only				

Registers capable of shifting in one direction is

____

a) Universal shift register
b) Unidirectional shift register
c) Unipolar shift register
d) Unique shift register

Register Transfer Language

- ⍰In computer science, **register transfer language**(RTL) is a kind of intermediate representation (IR) that is very close to assembly **language**, such as that which is used in a compiler. It is used to describe data flow at the **register-transfer** level of an architecture**.**
- ⍰**Digital Modules 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 <u>micro-operations </u>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)

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.**

Following are some commonly used registers:

1. **Accumulator**:used to store data taken out from memory.
2. **General Purpose Registers**: This is used to store data intermediate results during program execution.
3. **Special Purpose Registers**: Users do not access these registers. These registers are for Computer system,
    - **MAR:** Memory Address Register:- holds the address for memory unit.
    - **MBR:** Memory Buffer Register stores instruction and data received from the memory and sent from the memory.
    - **PC:** Program Counter points to the next instruction to be executed.
    - **IR:** Instruction Register holds the instruction to be executed.

Register Transfer Language

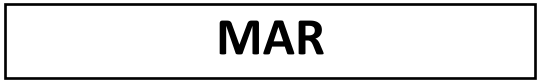
⍰ **Registers are designated by capital letters, sometimes followed by numbers (e.g., A, R13, IR)**

⍰ **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:**

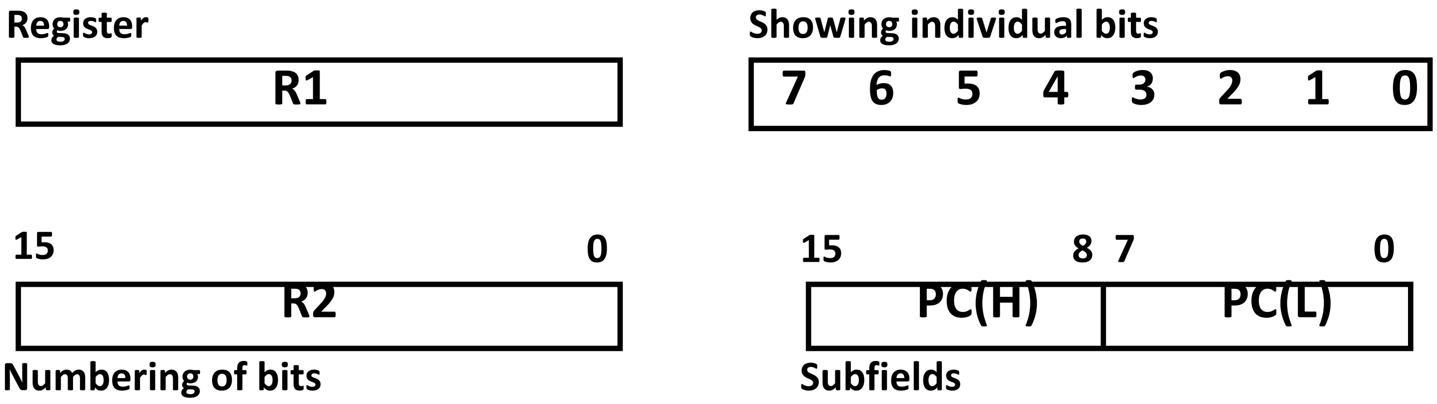


Register Transfer Language

- **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 Language

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

- **A register transfer is indicated as**

R2 ← 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

Register Transfer Language

- **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 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

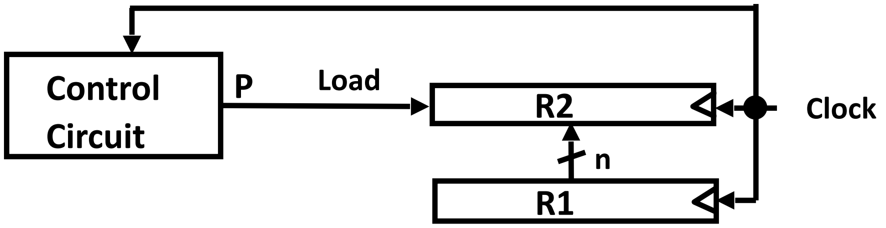
Which means "if P = 1, then load the contents of register R1 into register R2", i.e., if (P = 1) then (R2 ← R1)

Hardware Implementation of Controlled Transfers

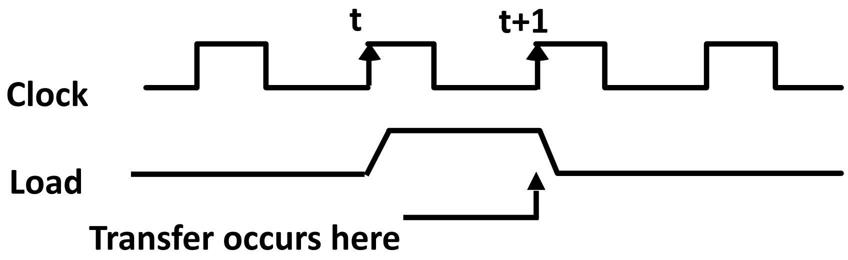
Implementation of controlled transfer

P: R2 ← R1

Block diagram

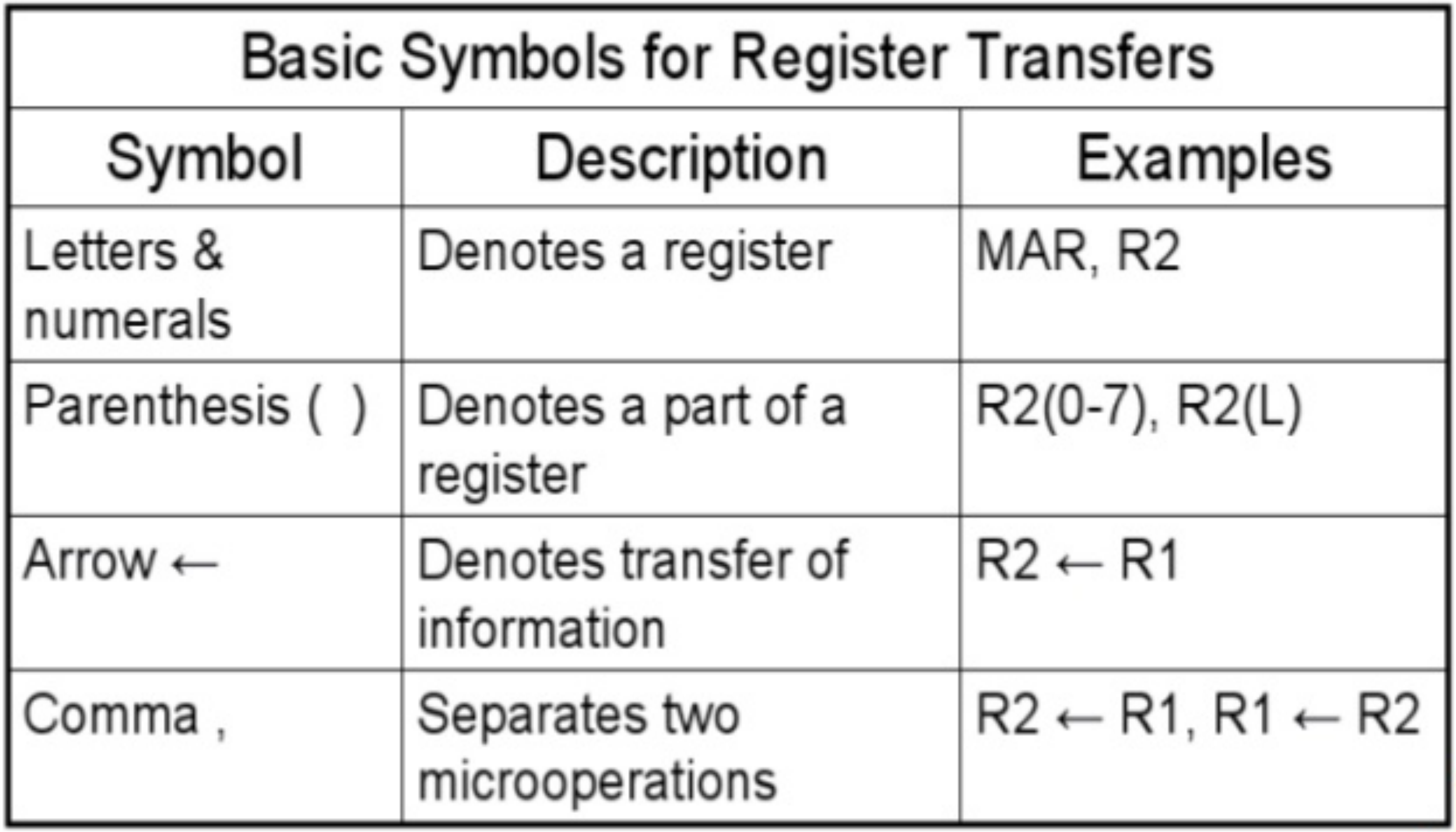


Timing diagram



- **The same clock controls the circuits that generate the control function and the destination register**
- **Registers are assumed to use *positive-edge-triggered* flip-flops**

REGISTER TRANSFER



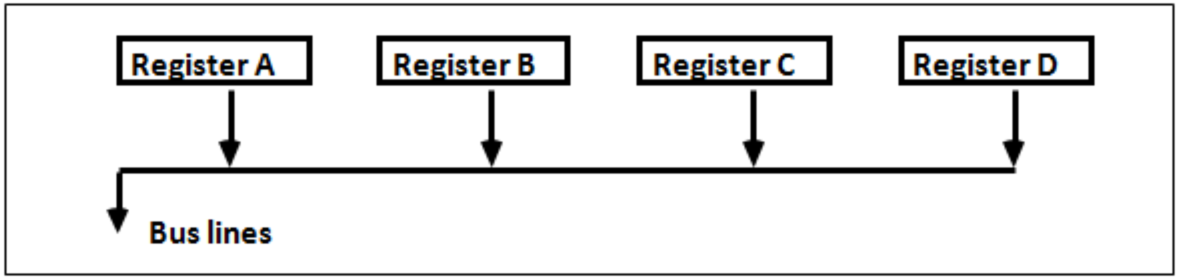
Basic Symbols for Register Transfers		
Symbol	**Description**	**Examples**
Letters & numerals	Denotes a register	MAR, R2
Parenthesis ( )	Denotes a part of a register	R2(0-7), R2(L)
Arrow ←	Denotes transfer of information	R2 ← R1
Comma ,	Separates two microoperations	R2 ← R1, R1 ← R2

Connecting Registers - Bus Transfer

- **In a digital system with many registers, it is impractical to have data and control lines to directly allow each register to be loaded with the contents of every possible other registers**
- **To completely connect n registers → n(n-1) lines**
- **O(n^2) cost**
  - **This is not a realistic approach to use in a large digital system**
- **Instead, take a different approach**
- **Have one centralized set of circuits for data transfer – the bus**
- **BUS STRUCTURE CONSISTS OF SET OF COMMON LINES, ONE FOR EACH BIT OF A REGISTER THROUGH WHICH BINARY INFORMATION IS TRANSFERRED ONE AT A TIME**
- **Have control circuits to select which register is the source, and which is the destination**

Connecting Registers - Bus Transfer

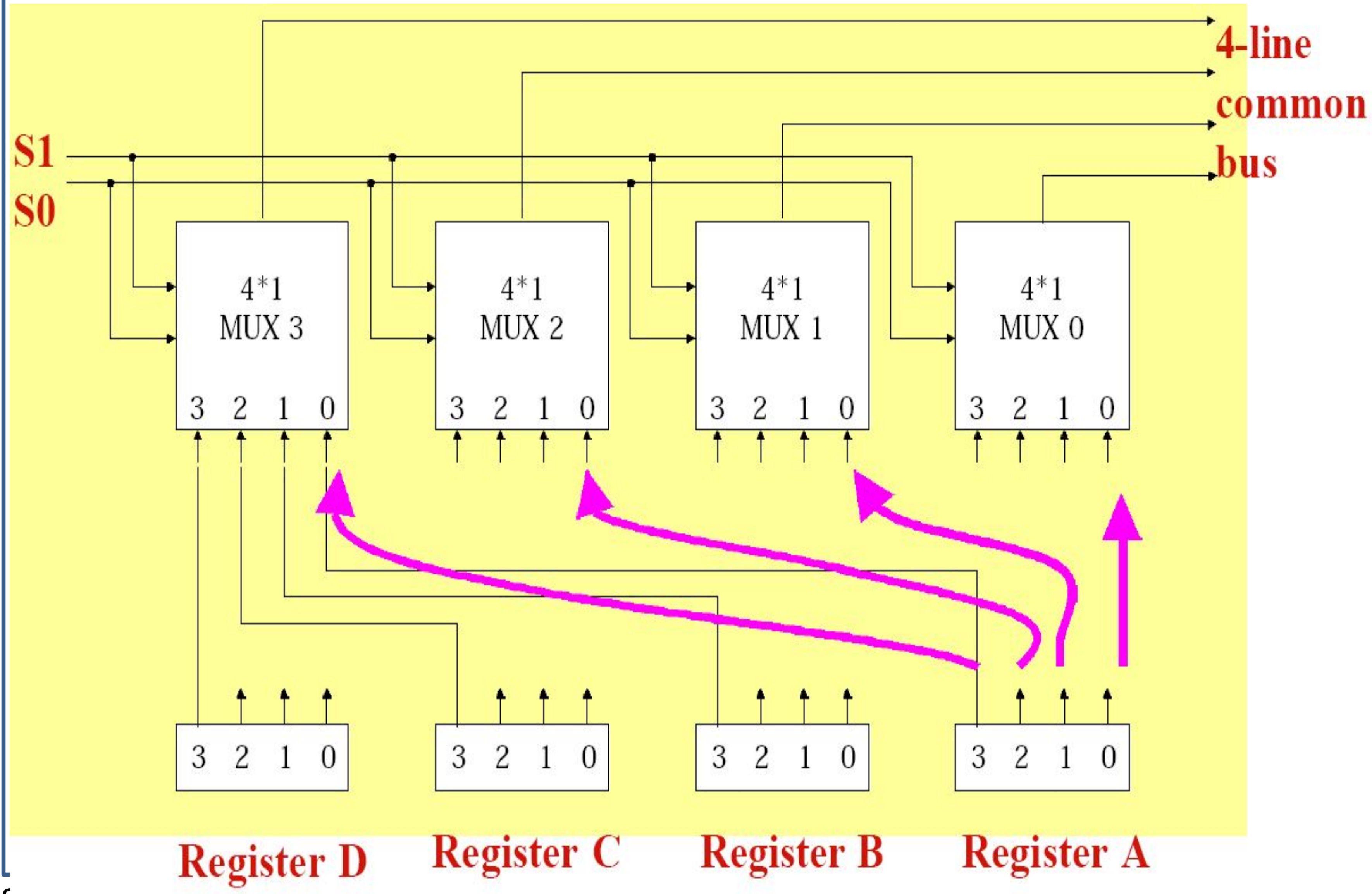
From a register to bus: BUS ← R



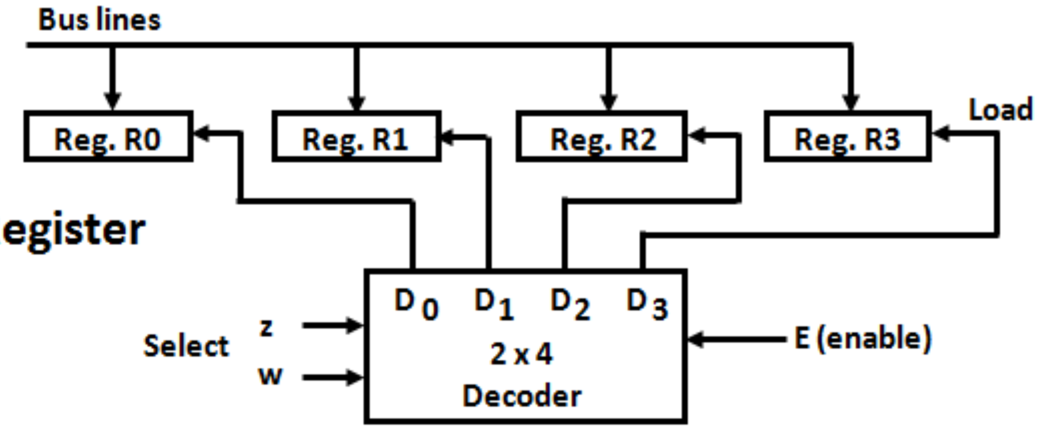
- **One way of constructing common bus system is with multiplexers**
- **Multiplexer selects the source register whose binary information is kept on the bus.**

- **Construction of bus system for 4 register (Next Fig)**
  - **4 bit register X 4**
  - **four 4X1 multiplexer**
  - **Bus selection S0, S1**

Connecting Registers - Bus Transfer



Connecting Registers - Bus Transfer



From a Bus to Register
R ← Bus