Computer Organization & Architecture

Lecture 2: Register Transfer and Micro-operations

By; Tanmoy Biswas,

Head, Department of Computer Science,

Syamaprasad College

Digital Modules and Micro-operation

- A **digital system** is an interconnection of **digital hardware modules** that accomplish a specific **information processing task.**
- The modules are constructed from such digital components;
  1. **Registers.**
  2. **Decoders.**
  3. **Arithmetic elements,**
  4. **Control logic.**
- The various modules are interconnected with **common data and control paths** to form a digital computer system.

Digital Modules and Micro-operation

- The operations executed on data stored in registers are called **micro-operations.**
- A micro-operation is an elementary operation performed on the information stored in one or more registers.
- The result of the operation may replace the previous binary information of a register or may be transferred to another register.
- Examples of **micro-operations are shift, count, clear, and load.**

Internal Hardware Organization

The internal hardware organization of a digital computer is best defined by specifying:

- The set of registers it contains and their function.
- The sequence of micro-operations performed on the binary information stored in the registers.
- The control that initiates the sequence of micro-operations.

Register Transfer Language

- Every operation involving sequence of micro-operations in a computer can be explained in words, but it is lengthy descriptive explanation.

- **Suitable symbology is used** to describe the sequence of transfers between registers and the various arithmetic and logic micro-operations associated with the transfers.

- The use of symbols instead of a narrative explanation provides an organized and concise manner for listing the micro-operation sequences in registers and the control functions that initiate them.

Register Transfer Language

- The symbolic notation used to describe the micro-operation transfers among registers is called a **register transfer language.**

- The term **"register transfer"** implies the availability of hardware logic circuits that can perform a **stated micro-operation and transfer the result of the operation to the same or another register.**

- The word "language" is borrowed from programmers, who apply this term to programming languages.

REGISTER TRANSFER AND MICROOPERATIONS

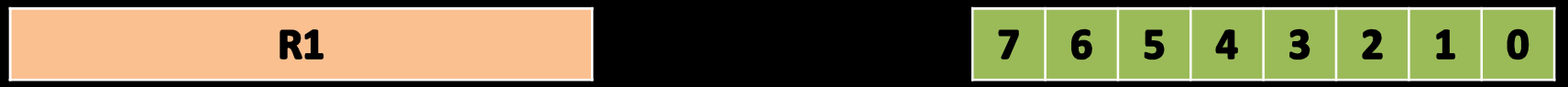
- Register Transfer Language
- Register Transfer
- Bus and Memory Transfers
- Arithmetic Micro-operations
- Logic Micro-operations
- Shift Micro-operations
- Arithmetic Logic Shift Unit

Lets understand Register Transfer Logic with example

- Computer registers are designated by **capital letters** (sometimes followed by numerals) to denote the function of the register.
- The register that holds an **address for the memory unit** is usually called a memory address register and is designated by the name MAR.
- Other designations for registers are PC (for program counter), IR (for instruction register) and R1 (for processor register).

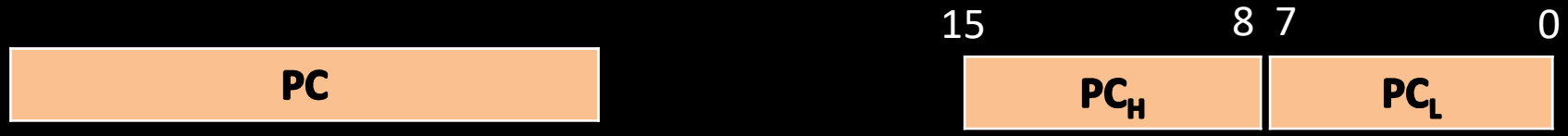
Lets understand Register Transfer Logic with example

- The individual flip-flops in an n-bit register are numbered in sequence from 0 through n - 1, starting from 0 in the rightmost position and increasing the numbers toward the left.



- A 16-bit register is partitioned into two parts. Bits 0 through 7 are assigned the symbol L (for low byte) and bits 8 through 15 are assigned the symbol H (for high byte).

- The name of the 16-bit register is PC. The symbol PC(0-7) or PC(L) refers to the low-order byte and PC(8-15) or PC(H) to the high-order byte.



Understanding Register Transfer Logic with example

- Information transfer from one register to another is designated in symbolic form by means of a **replacement operator.**
- The statement;

R2 <--R1

Before execution

R2							
D7	D6	D5	D4	D3	D2	D1	D0
X	X	X	X	X	X	X	X

R1							
D7	D6	D5	D4	D3	D2	D1	D0
0	1	1	1	0	0	1	1

After execution

Note: the content of register R1 remains un-altered

R2							
D7	D6	D5	D4	D3	D2	D1	D0
0	1	1	1	0	0	1	1

R1							
D7	D6	D5	D4	D3	D2	D1	D0
0	1	1	1	0	0	1	1

Understanding Register Transfer Logic with example

- Normally, we want the transfer to occur only under a predetermined control condition. This can be shown by means of an if-then statement.

If (P = 1) then (R2 <--R1)

- Where P is a control signal generated in the control section.

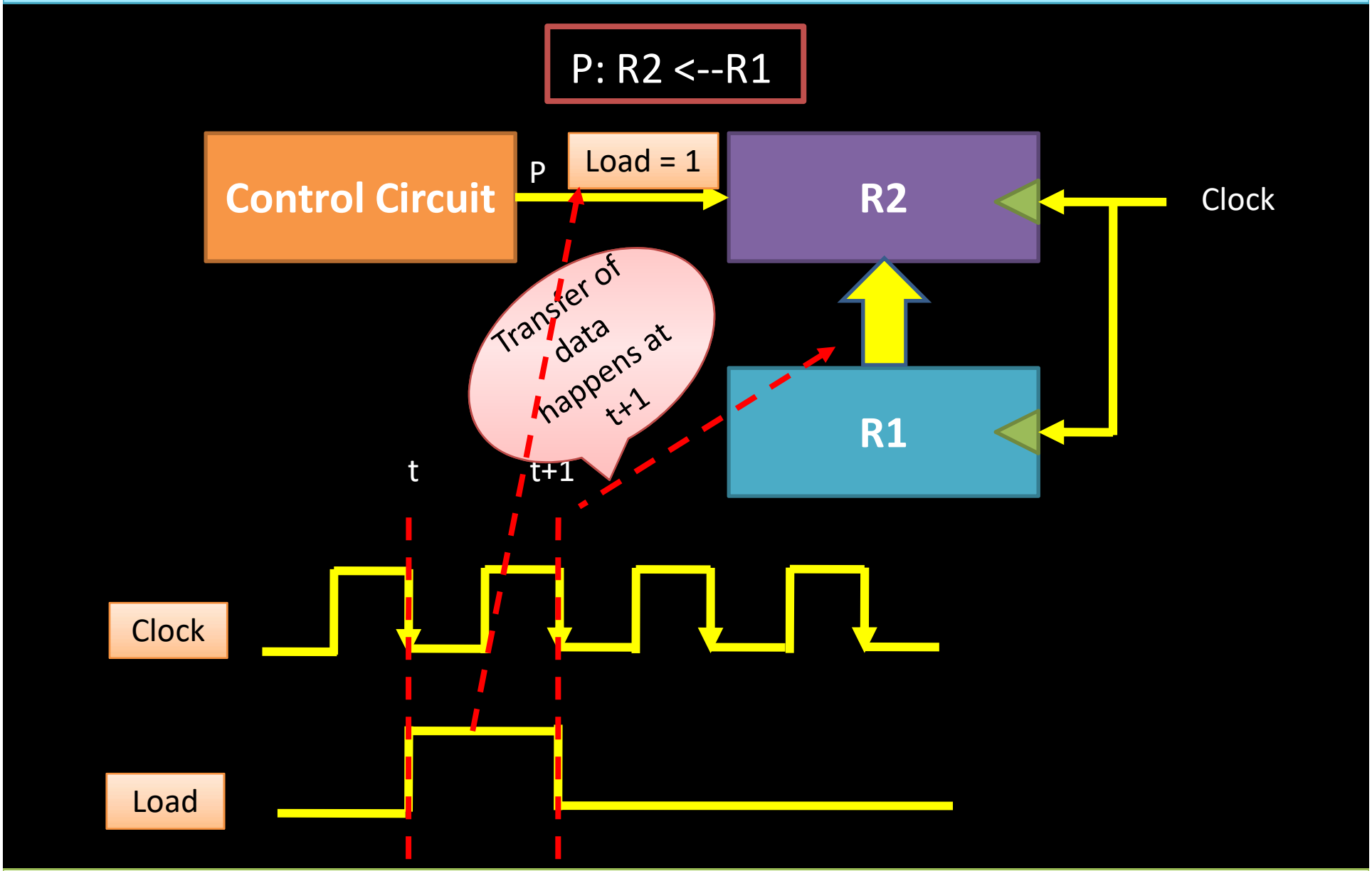
- It is sometimes convenient to separate the control variables from the register transfer operation by specifying a control function.

- A control function is a Boolean variable that is equal to 1 or 0. The control function is included in the statement as follows:

P: R2 <--R1

- The control condition is terminated with a colon. It symbolizes the requirement that the transfer operation be executed by the hardware only if P = 1 .

Understanding Register Transfer Logic with example



Note that the clock is not included as a variable in the register transfer statements. It is assumed that all transfers occur during a clock edge transition.

Understanding Register Transfer Logic with example

Basic symbols used for Register Transfers

Symbol	Description	Example
Letter and numerals combination	Denotes a Register	MAR, R1, R3, R3 etc
Parenthesis ( )	Denotes a part of register	R2 (0-7) or R2(L), R2(8-15) or R2(H)
Arrow ←	Denotes transfer of information	R2 ← R1
Comma ,	Separates two micro-operations	R2 ← R1 , R4 ← R3

The End