Computer Architecture and Organization [CSE2003]

(Fall 2019)

1

Anand Motwani
Faculty, SCSE,
VIT- Bhopal University.

Presentation by: Prof. Anand Motwani, Faculty, SCSE, VIT Bhopal University

2

Contents

UNIT – I: Introduction to Computer Architecture

Session – 3:

- Organization of the Von Neumann machine
- Interconnection of components
- Introduction to Registers, RTL and register files

Presentation by: Prof. Anand Motwani, Faculty, SCSE, VIT Bhopal University

3

Session Objectives

After this session students will be able to:

- Understand Organization of the Von Neumann machine.
- Understand Register and RTL

Presentation by: Prof. Anand Motwani, Faculty, SCSE, VIT Bhopal University

4

Entry Ticket

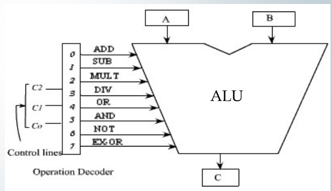
- Convert binary number 110011 to Decimal
- In case of 8-bit numbers, the minimum number that can be stored in computer is ______and maximum number is _____ (if we are working with natural numbers). Formula for the range of nos. = ________
- The basic operations like add subtract are implemented in computer using _____________

Presentation by: Prof. Anand Motwani, Faculty, SCSE, VIT Bhopal University

Solution to Tutorial 1

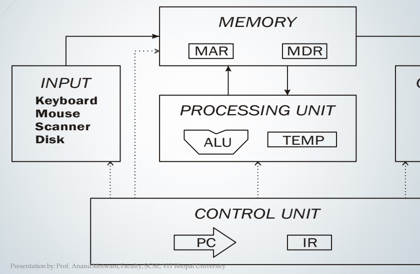
5

C_1	C_0	Arithmetic $C_2 = 0$	Logical $C_2 = 1$
0	0	Addition	OR
0	1	Subtraction	AND
1	0	Multiplication	NOT
1	1	Division	EX-OR

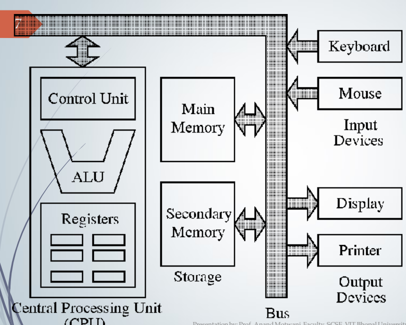


Presentation by: Prof. Anand Motwani, Faculty, SCSE, VIT Bhopal University

Von Neumann Model



The John von Neumann architecture

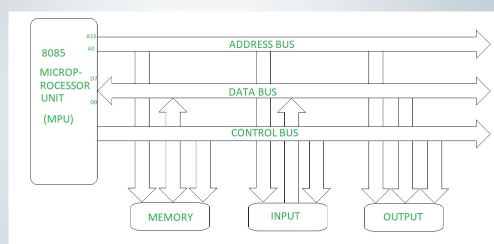


Showing Single Bus structure

Presentation by: Prof. Anand Motwani, Faculty, SCSE, VIT Bhopal University

Common Bus Organization

8



Showing Internal view of Single Bus structure

Bus organization system of 8085 Microprocessor

Presentation by: Prof. Anand Motwani, Faculty, SCSE, VIT Bhopal University

SIMPLE DIGITAL SYSTEMS

- 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 they perform.
- Typically,
  - What operations are performed on the data in the registers
  - What information is passed between registers

Concept behind Registers

A Flip-flop is also called a clock-controlled memory device.

A **Flip-flop** is use to store one bit of information. By connecting several Flip-flops together, they may store data that can represent the state of a sequencer, the value of a counter, an ASCII character in a computer's memory or any other piece of information.

Introduction to Register

- Register is a very fast computer memory, used to store data/instruction in-execution.
- A **Register** is a group of flip-flops with each flip-flop capable of storing **one bit** of information. An *n-bit* register has a group of *n flip-flops* and is capable of storing binary information of *n-bits*.
- A register consists of a group of flip-flops and gates. The flip-flops hold the binary information and gates control when and how new information is transferred into a register.
- Various types of registers are available commercially. The simplest register is one that consists of only flip-flops with no external gates.

Classification of Registers (Common CPU Registers

- These are classified as given below.
- **Accumulator:**
  This is the most frequently used register used to store data taken from memory. It is in different numbers in different microprocessors.
- **Memory Address Registers (MAR):**
  It holds the address of the location to be accessed from memory. MAR and MDR (Memory Data Register) together facilitate the communication of the CPU and the main memory.
- **Memory Data Registers (MDR):**
  It contains data to be written into or to be read out from the addressed location.
- **General Purpose Registers:**
  These are numbered as R0, R1, R2….Rn, and used to store temporary data during any ongoing operation. Its content can be accessed by assembly programming.
- **Program Counter (PC):**
  Program Counter (PC) is used to keep the track of execution of the program. It contains the memory address of the next instruction to be fetched. PC points to the address of the next instruction to be fetched from the main memory when the previous instruction has been successfully completed. Program Counter (PC) also functions to count the number of instructions.

- **Instruction Register (IR):**
  It is the register which holds the instruction which is currently been executed.

REGISTER TRANSFER AND MICROOPERATIONS

13

•Register Transfer:
The term **register transfer** means 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.

• Micro-operations

MICROOPERATIONS (1)

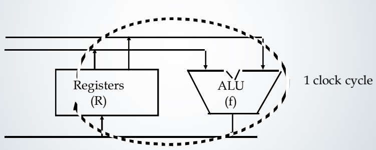
14

- The operations on the data in registers are called micro-operations.
- The functions built into registers are examples of micro-operations
  - Shift
  - Load: The transfer of new information into a register is referred to as loading the register.
  - Clear
  - Increment
  - …

MICROOPERATION (2)

15

An elementary operation performed (during one clock pulse), on the information stored in one or more registers



$R \leftarrow f(R, R)$

f: shift, load, clear, increment, add, subtract, complement, and, or, xor, …

ORGANIZATION OF A DIGITAL SYSTEM

16

Definition of the (internal) organization of a computer

- Set of registers and their functions

- Micro-operations set

  Set of allowable micro-operations provided by the organization of the computer

- Control signals that initiate the sequence of micro-operations (to perform the functions)

REGISTER TRANSFER LEVEL

17

- Viewing a computer, or any digital system, in this way is called the register transfer level
- This is because we're focusing on
  - The system's registers
  - The data transformations in them, and
  - The data transfers between them.

REGISTER TRANSFER LANGUAGE

18

- Rather than specifying a digital system in words, a specific notation is used, *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 (RTL)
  - The symbolic notation used to describe the micro-operation transfers amongst registers is called **Register transfer language**.
- OR
- Register Transfer Language
  - A symbolic language
  - A convenient tool for describing the internal organization of digital computers
  - Can also be used to facilitate the design process of digital systems.

DESIGNATION OF REGISTERS

19

- 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:

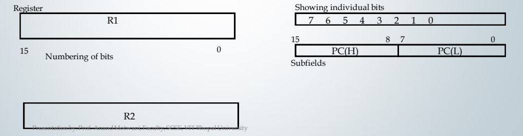
MAR

  - Registers may also be represented showing the bits of data they contain

DESIGNATION OF REGISTERS

20

- 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 & μ-operations *Register Transfer*

REGISTER TRANSFER

21

- 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

Presentation by: Prof. Anand Motwani, Faculty, SCSE, VIT Bhopal University

Register Transfer & μ-operations *Register Transfer*

REGISTER TRANSFER

22

- 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

Presentation by: Prof. Anand Motwani, Faculty, SCSE, VIT Bhopal University

Register Transfer & μ-operations *Register Transfer*

CONTROL FUNCTIONS

23

- 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)

Presentation by: Prof. Anand Motwani, Faculty, SCSE, VIT Bhopal University

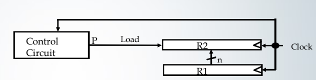
Register Transfer & μ-operations *Register Transfer*

HARDWARE IMPLEMENTATION OF CONTROLLED TRANSFERS

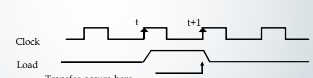
24

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

Presentation by: Prof. Anand Motwani, Faculty, SCSE, VIT Bhopal University

SIMULTANEOUS OPERATIONS

25

- If two or more operations are to occur simultaneously, they are separated with commas

P: R3 ← R5, MAR ← IR

- Here, if the control function P = 1, load the contents of R5 into R3, and at the same time (clock), load the contents of register IR into register MAR

BASIC SYMBOLS FOR REGISTER TRANSFERS

26

Symbols	Description	Examples
Capital letters & numerals	Denotes a register	MAR, R2
Parentheses ()	Denotes a part of a register	R2(0-7), R2(L)
Arrow ←	Denotes transfer of information	R2 ← R1
Colon :	Denotes termination of control function	P:
Comma ,	Separates two micro-operations	A ← B, B ← A

27

- Summary of Typical Arithmetic Micro-Operations:

R3 ← R1 + R2	Contents of R1 plus R2 transferred to R3
R3 ← R1 - R2	Contents of R1 minus R2 transferred to R3
R2 ← R2'	Complement the contents of R2
R2 ← R2'+ 1	2's complement the contents of R2 (negate)
R3 ← R1 + R2'+ 1	subtraction
R1 ← R1 + 1	Increment
R1 ← R1 – 1	Decrement

CONNECTING REGISTRS

28

- **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**
- **Have control circuits to select which register is the source, and which is the destination**

INTRODUCTION

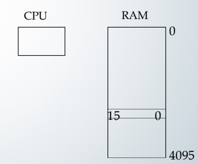
29

- Every different processor type has its own design (different registers, buses, micro operations, machine instructions, etc)
- Modern processor is a very complex device
- It contains
  - Many registers
  - Multiple arithmetic units, for both integer and floating point calculations
  - The ability to pipeline several consecutive instructions to speed execution
  - Etc.
- However, to understand how processors work, we will start with a simplified processor model
- This is similar to what real processors were like ~25 years ago
- M. Morris Mano introduces a simple processor model he calls the ***Basic Computer***
- We will use this to introduce processor organization and the relationship of the RTL model to the higher level computer processor

THE BASIC COMPUTER

30

- The Basic Computer has two components, a processor and memory
- The memory has 4096 words in it
  - $4096 = 2^{12}$, so it takes 12 bits to select a word in memory
- Each word is 16 bits long



INSTRUCTIONS

31

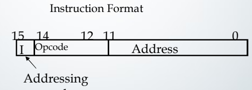
- Program
  - A sequence of (machine) instructions
- (Machine) Instruction
  - A group of bits that tell the computer to *perform a specific operation* (a sequence of micro-operation)
- The instructions of a program, along with any needed data are stored in memory
- The CPU reads the next instruction from memory
- It is placed in an *Instruction Register* (IR)
- Control circuitry in control unit then translates the instruction into the sequence of microoperations necessary to implement it

INSTRUCTION FORMAT

32

- A computer instruction is often divided into two parts
  - An *opcode* (Operation Code) that specifies the operation for that instruction
  - An *address* that specifies the registers and/or locations in memory to use for that operation
- In the Basic Computer, since the memory contains 4096 (= 2^{12}) words, we needs 12 bit to specify which memory address this instruction will use
- In the Basic Computer, bit 15 of the instruction specifies the *addressing mode* (0: direct addressing, 1: indirect addressing)
- Since the memory words, and hence the instructions, are 16 bits long, that leaves 3 bits for the instruction's opcode

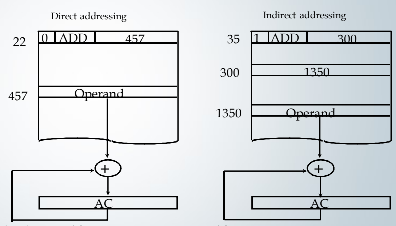
Instruction Format



ADDRESSING MODES

33

- The address field of an instruction can represent either
  - Direct address: the address in memory of the data to use (the address of the operand), or
  - Indirect address: the address in memory of the address in memory of the data to use



- Effective Address (EA)
  - The address, that can be directly used without modification to access an operand for a computation-type instruction, or as the target address for a branch-type instruction

PROCESSOR REGISTERS

34

- A processor has many registers to hold instructions, addresses, data, etc
- The processor has a register, the *Program Counter* (PC) that holds the memory address of the next instruction to get
  - Since the memory in the Basic Computer only has 4096 locations, the PC only needs 12 bits
- In a direct or indirect addressing, the processor needs to keep track of what locations in memory it is addressing: The *Address Register* (AR) is used for this
  - The AR is a 12 bit register in the Basic Computer
- When an operand is found, using either direct or indirect addressing, it is placed in the *Data Register* (DR). The processor then uses this value as data for its operation
- The Basic Computer has a single *general purpose register* – the *Accumulator* (AC)

PROCESSOR REGISTERS

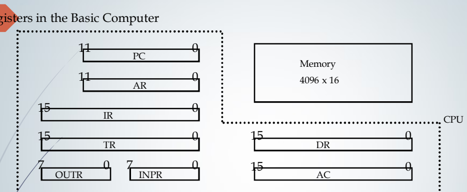
35

- The significance of a general purpose register is that it can be referred to in instructions
  - e.g. load AC with the contents of a specific memory location; store the contents of AC into a specified memory location
- Often a processor will need a scratch register to store intermediate results or other temporary data; in the Basic Computer this is the *Temporary Register* (TR)
- The Basic Computer uses a very simple model of input/output (I/O) operations
  - Input devices are considered to send 8 bits of character data to the processor
  - The processor can send 8 bits of character data to output devices
- The *Input Register* (INPR) holds an 8 bit character gotten from an input device
- The *Output Register* (OUTR) holds an 8 bit character to be send to an output device

BASIC COMPUTER REGISTERS

36

Registers in the Basic Computer



List of BC Registers

DR	16	Data Register	Holds memory operand
AR	12	Address Register	Holds address for memory
AC	16	Accumulator	Processor register: Results of operations go to this register
IR	16	Instruction Register	Holds instruction code
PC	12	Program Counter	Holds address of instruction
TR	16	Temporary Register	Holds temporary data
INPR	8	Input Register	Holds input character
OUTR	8	Output Register	Holds output character

COMMON BUS SYSTEM

37

- The registers in the Basic Computer are connected using a bus
- This gives a savings in circuitry over complete connections between registers

38

Exit Ticket

- The maximum size of main memory that can be used in any computer is determined by the addressing scheme.
- A computer that generates 16-bit address is capable of addressing upto 2^{16} which is equal to 64K memory location.
- Similarly, for 32 bit addresses, the total capacity will be ________ which is equal to _______ memory location.

39

Questions

40

Computer Architecture

- Computer Architecture concerns Machine Organization, interfaces, application, technology, measurement & simulation that Includes:
- Instruction set
- Data formats
- Principle of Operation (formal description of every operation)
- Features (organization of programmable storage, registers used, interrupts mechanism, etc.)
- In short, it is the combination of Instruction Set Architecture, Machine Organization and the related hardware.

41

References

Presentation by: Prof. Anand Motwani, Faculty, SCSE, VIT Bhopal University