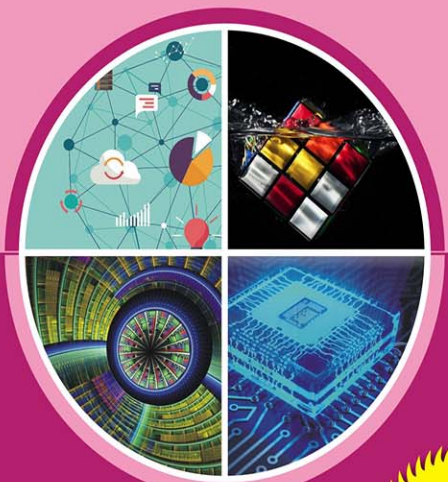
Quantum Series®
ENGINEERING

QUANTUM *Series*

Semester - 3 CS & IT

Computer Organization & Architecture



- Topic-wise coverage of entire syllabus in Question-Answer form.
- Short Questions (2 Marks)

Session 2019-20 Odd Semester

Includes solution of following AKTU Question Papers
2014-15 • 2015-16 • 2016-17 • 2017-18 • 2018-19

www.askbooks.net

A.S.K.

Always.Seek.Knowledge

All AKTU QUANTUMS are available

- An initiative to provide free ebooks to students.
- Hub of educational books.

1. All the ebooks, study materials, notes available on this website are submitted by readers you can also donate ebooks/study materials.
2. We don't intend to infringe any copyrighted material.
3. If you have any issues with any material on this website you can kindly report us, we will remove it asap.
4. All the logos, trademarks belong to their respective owners.

QUANTUM SERIES

For

B.Tech Students of Second Year
of All Engineering Colleges Affiliated to
Dr. A.P.J. Abdul Kalam Technical University,
Uttar Pradesh, Lucknow
(Formerly Uttar Pradesh Technical University)

Computer Organization & Architecture

By

Aditya Kumar

QUANTUM PAGE PVT. LTD.
Ghaziabad ■ New Delhi

PUBLISHED BY : **Apram Singh**
Quantum Page Pvt. Ltd.
Plot No. 59/2/7, Site - 4, Industrial Area,
Sahibabad, Ghaziabad-201 010

Phone : 0120 - 4160479

Email : pagequantum@gmail.com **Website: www.quantumpage.co.in**

Delhi Office : 1/6590, East Rohtas Nagar, Sahadara, Delhi-110032

© ALL RIGHTS RESERVED
No part of this publication may be reproduced or transmitted, in any form or by any means, without permission.

Information contained in this work is derived from sources believed to be reliable. Every effort has been made to ensure accuracy, however neither the publisher nor the authors guarantee the accuracy or completeness of any information published herein, and neither the publisher nor the authors shall be responsible for any errors, omissions, or damages arising out of use of this information.

Computer Organization & Architecture (CS/IT : Sem-3)

1st Edition : *2010-11*
2nd Edition : *2011-12*
3rd Edition : *2012-13*
4th Edition : *2013-14*
5th Edition : *2014-15*
6th Edition : *2015-16*
7th Edition : *2016-17*
8th Edition : *2017-18*
9th Edition : *2018-19*
10th Edition : *2019-20* (*Thoroughly Revised Edition*)
11th Edition : *2020-21*

Price: Rs. 80/- only

Printed Version **:** e-Book.

CONTENTS

KCS 302 : Computer Organization & Architecture

1
UNIT

Introduction

CONTENTS

PART-1

Introduction : Functional Units of Digital System and their Interconnection.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 1.1. **Draw a block diagram of a computer's CPU showing all the basic building blocks such as program counter, accumulator, address and data registers, instruction register, control unit etc., and describe how such an arrangement can work as a computer, if connected properly to memory, input/output etc.**

AKTU 2016-17, Marks 7.5

Answer

Block diagram of computer's CPU :

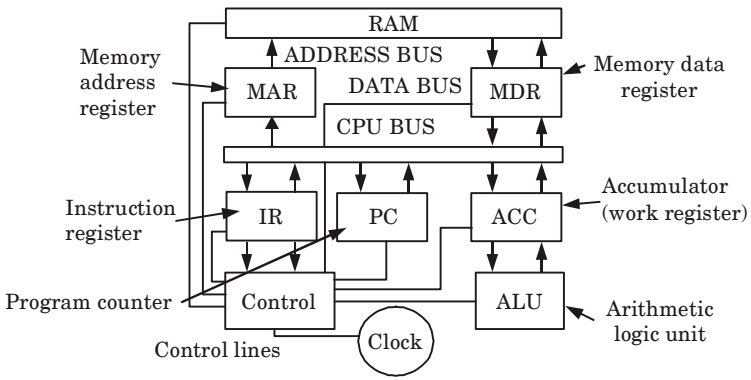


Fig. 1.1.1.

A computer performs five major operations. These are :

1. It accepts data or instructions as input.
2. It stores data and instruction.
3. It processes data as per the instructions.
4. It controls all operations inside a computer.
5. It gives results in the form of output.

Arrangement of CPU, memory, input/output to work as a computer :

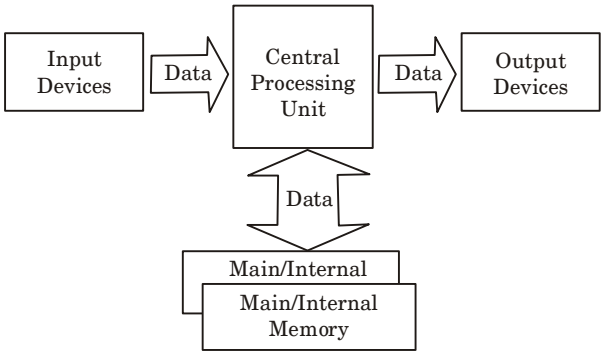


Fig. 1.1.2.

a. Input unit : This unit is used for entering data and programs into the computer system by the user for processing.

b. Storage unit : The storage unit is used for storing data and instructions before and after processing.

c. Output unit : The output unit is used for storing the result as output produced by the computer after processing.

d. Processing unit : The task of performing operations like arithmetic and logical operations is called processing.

The Central Processing Unit (CPU) takes data and instructions from the storage unit and makes all sorts of calculations based on the instructions given and the type of data provided. It is then sent back to the storage unit.

Que 1.2. Explain the functional units of digital system and their interconnections.

Answer

The main functional units of a digital computer are shown in Fig. 1.2.1.

1. Central Processing Unit (CPU) :

a. The CPU is the brain of a computer system.

b. This unit takes the data from the input devices and processes it according to the set of instructions called program.

c. The output of processing of the data is directed to the output devices for use in the outside world.

d. CPU has two major parts called ALU and Control Unit.

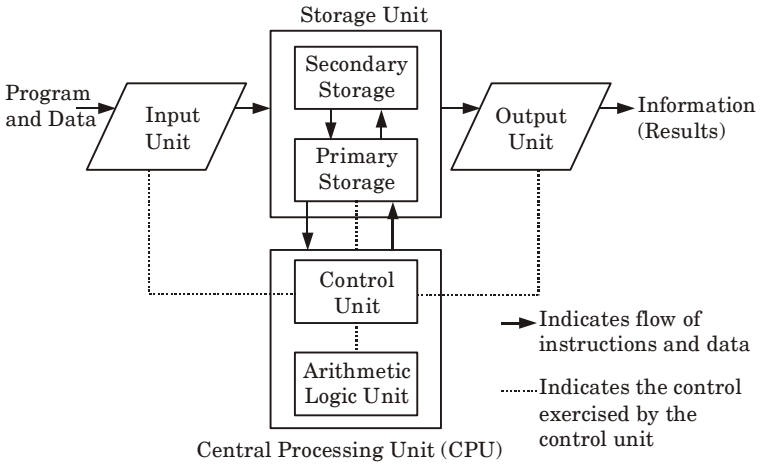


Fig. 1.2.1. Functional unit of digital computer.

i. Arithmetic Logic Unit (ALU) :

a. ALU is responsible for carrying out following operations :

1. Arithmetic operations on data by adding, subtracting, multiplying and dividing one set with another.
2. Logical operations by using AND, OR, NOT and exclusive-OR operation which is done by analyzing and evaluating data.

b. ALU of a computer system is the place where actual execution of instructions takes place during processing operation.

ii. Control Unit (CU) :

a. This unit is mainly used for generating the electronic control signals for the synchronization of various operations.

b. All the related functions for program execution such as memory read, memory write, I/O read, I/O write, execution of instruction, are synchronized through the control signal generated by the control unit.

c. It manages and controls all the operations of the computer.

2. Input unit : An input unit performs following functions :

a. It accepts (or reads) instructions and data from outside world.

b. It converts these instructions and data in computer acceptable form.

c. It supplies the converted instructions and data to computer system for further processing.

3. **Output unit :** An output unit performs following functions :
   a. It accepts the results produced by a computer, which are in coded form.
   b. It converts these coded results to human acceptable (readable) form.
   c. It supplies the converted results to outside world.
4. **Storage unit :** A storage unit holds (stores) :
   a. Data and instructions required for processing (received from input devices).
   b. Intermediate results of processing.
   c. Results for output, before they are released to an output device.

PART-2

Bus, Bus Architecture, Types of Buses.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 1.3. **What is a bus in digital system ? Also explain its types.**

Answer

1. A bus is a group of wires connecting two or more devices and providing a path to perform communication.
2. A bus that connects major computer components/modules (CPU, Memory, I/O) is called a system bus.
3. These system buses are separated into three functional groups :
   i. Data bus :
      a. The data bus lines are bidirectional.
      b. The data bus consists of 8, 16, 32 or more parallel lines.
   ii. Address bus :
      a. It is a unidirectional bus.
      b. The address bus consists of 16, 20, 24 or more parallel lines. The CPU sends out the address of the memory location or I/O port that is to be written or read by using address bus.
   iii. Control bus :
      a. Control lines regulate the activity on the bus.

b. The CPU sends signals on the control bus to enable the outputs of addressed memory device or port device.

Que 1.4. Describe the architecture of bus.

Answer

1. The computer bus consists of two parts, the address bus and a data bus. The data bus transfers actual data, whereas the address bus transfers address or memory location of where the data should go.
2. The bus provides physical links and the means of controlling the communication exchange of signals over the bus. Fig. 1.4.1 depicts the organization of a single shared bus.

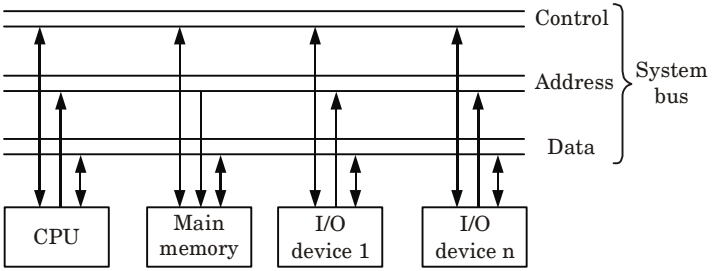


Fig. 1.4.1. Architecture of a single shared bus.

3. The principle use of the system bus is high-speed data transfer between the CPU and memory.
4. Most I/O devices are slower than the CPU or the memory. The I/O devices are attached to the system bus through external interfaces.
5. The Input/Output (I/O) ports are used to connect various devices to the computer and hence, enable communication between the device and the computer.

PART-3

Bus Arbitration.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 1.5. **Discuss the bus arbitration.**

OR

Write a short note on bus arbitration.

AKTU 2014-15, Marks 05

Answer

1. Bus arbitration is a mechanism which decides the selection of current master to access bus.
2. Among several masters and slave units that are connected to a shared bus, it may happen that more than one master or slave units will request access to the bus at the same time.
3. In such situation, bus access is given to the master having highest priority.
4. Three different mechanisms are commonly used for this :
   - **i. Daisy chaining :**
     a. Daisy chaining method is cheaper and simple method.
     b. All master make use of the same line for bus request.
     c. The bus grant signal serially propagates through each master until it encounters the first one that is requesting.
   - **ii. Parallel arbitration :** The parallel arbitration consists of priority encoder and a decoder. In this mechanism, each bus arbiter has a bus request output line and input line.
   - **iii. Independent priority :** In this each master has separate pair of bus request and bus grant lines and each pair has a priority assigned to it.

Que 1.6. **Discuss the advantages and disadvantages of polling and daisy chaining bus arbitration schemes.**

AKTU 2015-16, Marks 10

OR

Explain daisy changing method. Write its advantages and disadvantages.

Answer

Daisy chaining :

1. In this, all masters make use of the same line for bus request.
2. The bus grant signal serially propagates through each master until it encounters the first one that is requesting access to the bus.
3. This master blocks the propagation of the bus grant signal, activates the busy line and gains control of the bus.
4. Therefore any other requesting module will not receive the grant signal and hence cannot get the bus access.

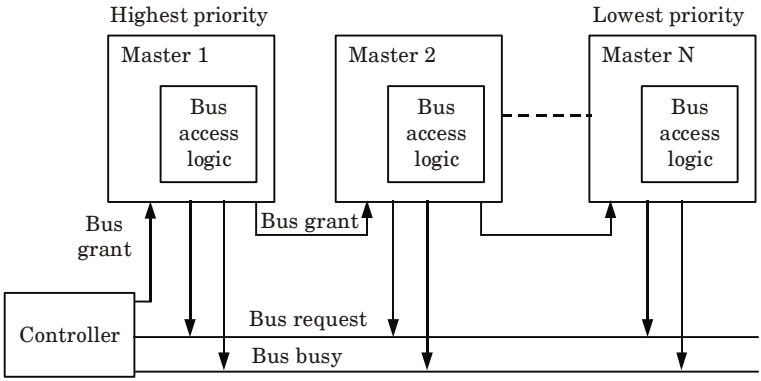


Fig. 1.6.1. Daisy chaining method.

Advantages of daisy chaining :

1. It is a simple and cheaper method.
2. It requires the least number of lines and this number is independent of the number of masters in the system.

Disadvantages of daisy chaining :

1. The propagation time delay of bus grant signal is proportional to the number of masters in the system. This makes arbitration time slow and hence limits the number of master in the system.
2. The priority of the master is fixed by the physical location of master.
3. Failure of any one master causes the whole system to fail.

Advantages of polling bus arbitration :

1. If the one module fails entire system does not fail.
2. The priority can be changed by altering the polling sequence stored in the controller.

Disadvantages of polling bus arbitration :

1. It requires more bus request and grant signals ($2 \times n$ signals for n modules).
2. Polling overhead can consume a lot of CPU time.

PART-4

Register, Bus and Memory Transfer.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 1.7. What is memory transfer ? What are different registers associated for memory transfer ? Discuss.

Answer

1. Memory transfer means to fetch (Read) or store (Write) data.
2. The read operation transfers a copy of the content from memory locations to CPU.
3. The store operation transfers the word information from the CPU to a specific memory location, destroying previous content of that location.
4. The memory word is symbolized by the letter *M*.

Registers associated for memory transfer :

1. There are two registers associated for memory transfer : address register and data register.
2. The required information is selected from the memory location by address. It is stored in the Address Register (AR).
3. The data is transferred to another register called Data Register (DR).
4. Consider a simple read operation,

   Read : DR ← M[AR]

   Here, the information is transferred into Data Register (DR) from the memory word *M* selected by the addresses in Address Register [AR].
   The write operation is denoted as,
   Write : M[AR] ← B

Que 1.8. Explain the operation of three state bus buffers and show its use in design of common bus. AKTU 2016-17, Marks 15

Answer

1. A three state gate is a digital circuit that exhibits three states.
2. Two of the states are signals equivalent to logic 1 and 0.
3. The third state is a high-impedance state.
4. The high-impedance state behaves like an open circuit, which means that the output is disconnected and does not have logic significance.
5. Three state gates may perform any conventional logic, such as AND or NAND.
6. However, the one most commonly used in the design of a bus system is the buffer gate.
7. The graphic symbol of a three state buffer gate is shown in Fig. 1.8.1.

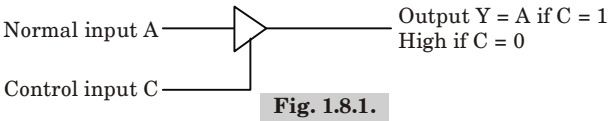


Fig. 1.8.1.

8. The control input determines the output state.
9. When the control input is equal to 1, the output is enabled and the gate behaves like any conventional buffer, with the output equal to the normal input.
10. When the control input is 0, the output is disabled and the gate goes to a high state, regardless of the value in the normal input.
11. A large number of three state gate outputs can be connected with wires to form a common bus line without endangering loading effects.

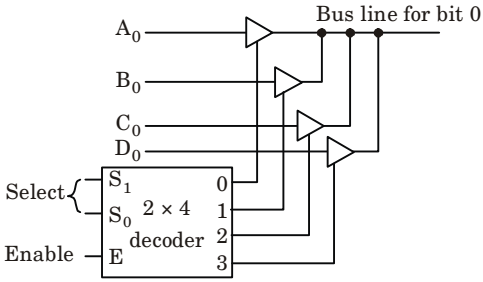


Fig. 1.8.2.

12. The outputs of four buffers are connected together to form a single bus line.
13. The control inputs to the buffers determine which of the four normal inputs will communicate with the bus line.
14. Not more than one buffer may be in the active state at any given time.
15. To construct a common bus for four registers of n bits each using three state buffers, we need n circuits with four buffers in each.
16. Each group of four buffers receives one significant bit from the four registers.
17. Only one decoder is necessary to select between the four registers.

Que 1.9. Explain why the single shared bus is so widely used as an interconnection medium in both sequential and parallel computers. What are its main disadvantages ?

Answer

Single shared bus is so widely used as an interconnection medium in both sequential and parallel computer because of following reasons :

1. The shared bus is the simplest and least expensive way of connecting several processors to a set of memory modules.

2. It allows compatibility and provides ease of operation and high bandwidth.

Disadvantages of single shared bus are :

1. The main disadvantage of the shared bus is that its throughput limits the performance boundary for the entire multiprocessor system. This is because at any time only one memory access is granted, likely causing some processors to remain idle. To increase performance by reducing the memory access traffic, a cache memory is often assigned to each processor.
2. Another disadvantage of the shared bus design is that, if the bus fails, catastrophic failure results. The entire system will stop functioning since no processor will be able to access memory.

Que 1.10. What is the benefit of using multiple bus architecture compared to a single bus architecture ?

Answer

Following are the benefits of using multiple bus architecture compared to single bus architecture :

1. Single bus have long control sequence because only the data item can be transferred over the bus in a clock cycle.
2. To reduce the number of steps needed, most commercial processor provides multiple internal paths using multiple buses.
3. The introduction of incrementer unit eliminates the need to add port to the PC using the main ALU.
4. By providing more paths for data transfer, a significant reduction in the number of clock cycles needed to execute an instruction is achieved.

PART-5

Processor Organization, General Register Organization.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 1.11. Explain general-purpose register based organization.

Answer

1. In this organization, the registers communicate with each other not only for direct data transfers, but also while performing various micro-operations.

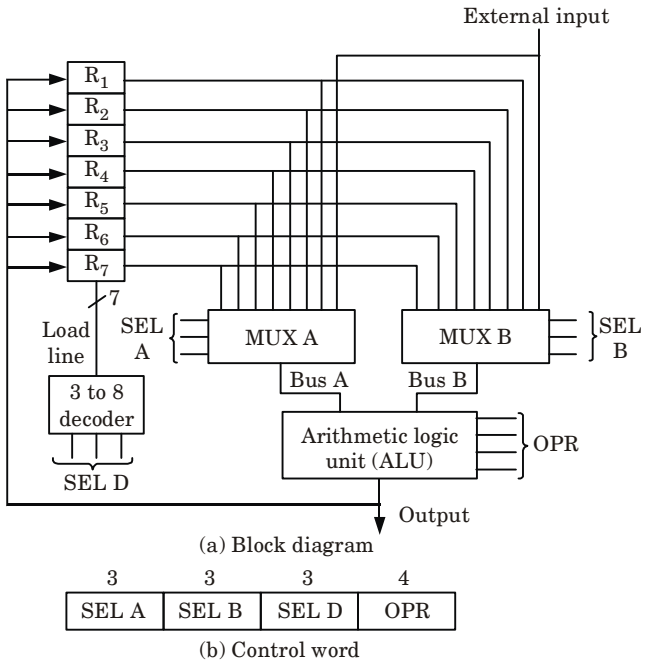


Fig. 1.11.1. General-purpose register based organization.

2. Seven registers are used for general purpose, the output of each register is connected to two multiplexer (MUXs) inputs.
3. Three select lines are used to select any one of the seven registers and the contents of selected registers are supplied to the inputs of ALU.
4. The buses *A* and *B* are used to form the inputs to the common arithmetic logic unit (ALU).
5. The operation to be performed is selected in the ALU and is determined by the arithmetic or logic micro-operation by using function select lines (OPR).
6. The result of the micro-operation is available as output data and also goes into the inputs of all the registers.
7. Any one of the destination register receives the information from the output bus which is selected by a decoder.

PART-6

Stack Organization.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 1.12. What is stack ? Give the organization of register stack with all necessary elements and explain the working of push and pop operations. **AKTU 2016-17, Marks 15**

Answer

1. A stack is an ordered set of elements in which only one element can be accessed at a time.
2. The point of access is called the top of the stack.
3. The number of elements in the stack or length of the stack is variable.
4. Items may only be added or deleted from the top of the stack.
5. A stack is also known as a pushdown list or a Last-In-First-Out (LIFO) list.

Organization of register stack :

Consider the organization of a 64-word register stack as illustrated in Fig. 1.12.1.

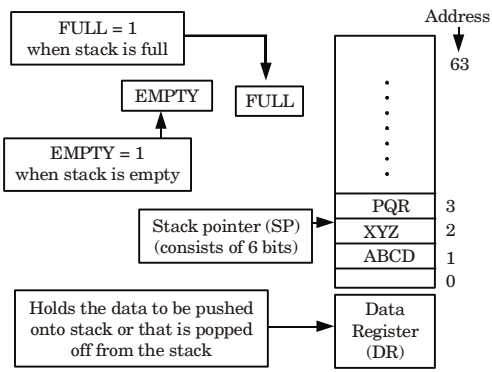


Fig. 1.12.1. Block diagram of 64-word stack.

The four separate registers used in the organization are :

1. **Stack Pointer register (SP) :** It contains a value in binary each of 6 bits, which is the address of the top of the stack. Here, the stack pointer SP contains 6 bits and SP cannot contain a value greater than 111111 *i.e.*, value 63.
2. **FULL register :** It can store 1 bit information. It is set to 1 when the stack is full.
3. **EMPTY register :** It can store 1 bit information. It is set to 1 when stack is empty.
4. **Data Register (DR) :** It holds the data to be written into or to be read from the stack.

Working of POP and PUSH :

POP (Performed if stack is not empty *i.e.*, if EMPTY = 0) :

DR ← M[SP]	Read item from the top of stack
SP ← SP – 1	Decrement stack pointer
If (SP = 0) then (EMPTY ← 1)	Check if stack is empty
FULL ← 0	Mark the stack not full

PUSH (Performed if stack is not full *i.e.*, if FULL = 0) :

SP ← SP + 1	Increment stack pointer
M[SP] ← DR	Write item on top of the stack
If (SP = 0) then (FULL ← 1)	Check if stack is full
EMPTY ← 0	Mark the stack not empty

Que 1.13. What is a memory stack ? Explain its role in managing subroutines with the help of neat diagrams.

AKTU 2016-17, Marks 15

Answer

Memory stack : Memory stack is a series of memory spaces that is used in the processes that is done by processor and is temporarily stored in registers.

Role in managing subroutines :

1. The stack supports program execution by maintaining automatic process-state data.
2. If the main routine of a program, for example, invokes function *a* ( ), which in turn invokes function *b* ( ), function *b* ( ) will eventually return control to function *a* ( ), which in turn will return control to the main ( ) function as shown in Fig. 1.13.1.
3. To return control to the proper location, the sequence of return addresses must be stored.
4. A stack is well suited for maintaining this information because it is a dynamic data structure that can support any level of nesting within memory constraints.

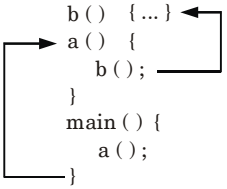


Fig. 1.13.1. Stack management.

5. When a subroutine is called, the address of the next instruction to execute in the calling routine is pushed onto the stack.
6. When the subroutine returns, this return address is popped from the stack, and program execution jumps to the specified location as shown in Fig. 1.13.2.

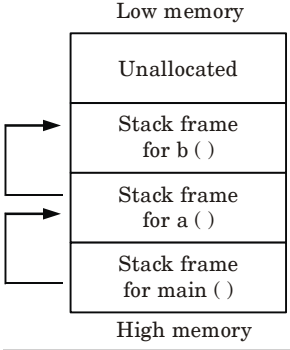


Fig. 1.13.2. Calling a subroutine.

7. The information maintained in the stack reflects the execution state of the process at any given instant.
8. In addition to the return address, the stack is used to store the arguments to the subroutine as well as local (or automatic) variables.
9. Information pushed onto the stack as a result of a function call is called a frame. The address of the current frame is stored in the frame or base pointer register.
10. When a subroutine is called, the frame pointer for the calling routine is also pushed onto the stack so that it can be restored when the subroutine exits.

Que 1.14. What is the stack organization ? Compare register stack and memory stack.

Answer

Stack organization : Refer Q. 1.12, Page 1–13B, Unit-1.

Comparison between register stack and memory stack :

S. No.	Register stack	Memory stack
1.	Register stacks are momentary spaces for internal processes that are being done by the processor.	Memory stacks are a series of memory spaces that is used in the processes that is done by processor and are temporarily stored in registers.
2.	Register stack is generally on (CPU).	Memory stack is on RAM.
3.	Access to register stack is faster.	Access to memory stack is slower.
4.	It is limited in size.	It is large in size.

Que 1.15. Explain an accumulator based central processing unit organization with block diagram.

Answer

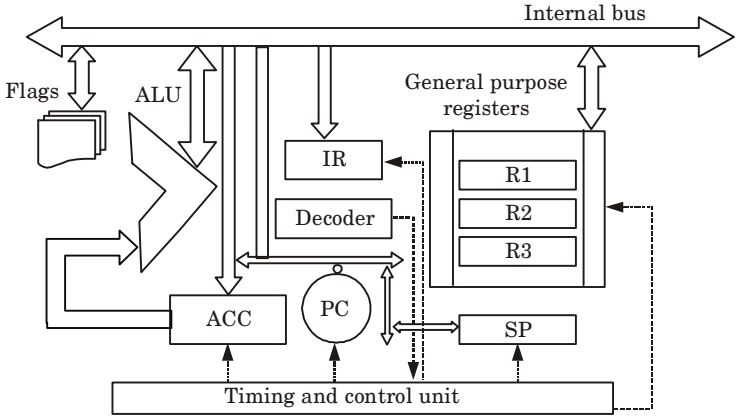


Fig. 1.15.1. Block diagram of accumulator based CPU organization.

1. **ALU :** A most generic computer system is composed of a unit to do arithmetic, shift and logical micro-operations commonly known as ALU of CPU.
2. **Program Counter (PC) :** This keeps track of the instruction address in memory from where the next instruction needs to be fetched. The instructions are stored in memory in an order decided by programmer.
3. **General purpose registers (R1, R2, R3) :** It suggests that the registers are involved in operations like load inputs, store intermediate results of arithmetic, logical and shift micro-operations. The initial inputs are loaded into registers from memory and final results are later moved into memory.

4. **Accumulator (ACC) :** This block acts as the default temporary storage register location for all mathematical operations in ALU.
5. **Instruction Register IR and Decoder :** After instruction is fetched from the memory its stored in Instruction Register. The instruction is then decode by the decoder.
6. **Stack pointer (SP) :** Stack pointer is involved in managing the stack transfers during and program execution.
7. **Timing and control unit :** This block manages the sequencing of events on a timeline between various components of a CPU. All the blocks are controlled in a manner to optimize the computational power of the unit by minimizing the failures.
8. **Flags :** Flags are also registers or bits inside registers which are set or cleared for a particular condition on an arithmetic operation. Some of the most common flags are :
   i. **Sign :** Is used to identify the set/reset of most significant bit of the result.
   ii. **Carry :** Is used to identify, a carry during addition, or borrow during subtraction/comparison.
   iii. **Parity :** Set if the parity is even. Refer parity from here.
   iv. **Zero :** To identify when the result is equal to zero.
9. **Bus sub-system :** All the data transfers in-between memory and CPU registers including instruction fetches are carried over bus.

Que 1.16. **Explain various types of processor organization.**

AKTU 2015-16, Marks 10

Answer

Types of processor organizations :

1. **General-purpose register based :** Refer Q. 1.11, Page 1–11B, Unit-1.
2. **Stack based :** Refer Q. 1.12, Page 1–13B, Unit-1.
3. **Accumulator based :** Refer Q. 1.15, Page 1–16B, Unit-1.

PART-7

Addressing Modes.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 1.17. **Write short note on relative addressing mode and indirect addressing mode.**

OR

Explain the following addressing modes with the help of an example each :

i. Direct **ii. Register indirect**
iii. Implied **iv. Immediate**
v. Indexed

AKTU 2014-15, Marks 10

Answer

i. Direct :

1. A very simple form of addressing is direct addressing, in which the address field contain the effective address of the operand : $EA = A$
   where, EA = Actual (effective) address of the location containing the referenced operand.
   A = Contents of the address field in the instruction.

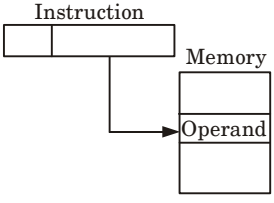


Fig. 1.17.1. Direct.

2. A direct address in instruction needs two reference to memory :
   a. Read instruction b. Read operand

ii. Displacement addressing :

1. A very powerful mode of addressing combines the capabilities of direct addressing and register indirect addressing.
2. It is known by a variety of names depending upon the content of its use but the basic mechanism is the same.
3. Displacement addressing requires that the instruction have two address fields, at least one of which is explicit.
4. The value contained in one address field (value = A) is used directly.
5. The other address field or an implicit reference based on opcode, refers to a register whose contents are added to A to produce the effective address.

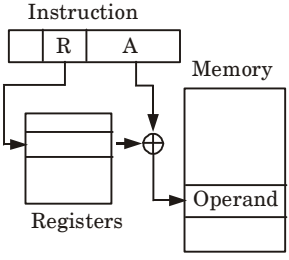


Fig. 1.17.2. Displacement addressing.

iii. Relative addressing :

1. Relative addressing means that the next instruction to be carried out is an offset number of locations away, relative to the address of the current instruction.
2. Consider this bit of pseudo-code
   Jump + 3 if accumulator = = 2
   Code executed if accumulator is NOT = 2
   Jump + 5 (unconditional relative jump to avoid the next line of code)
   acc : (code executed if accumulator is = 2)
3. In the code, the first line of code is checking to see if the accumulator has the value of 2 then the next instruction is 3 lines away.
4. This is called a conditional jump and it is making use of relative addressing.

iv. Register indirect mode :

1. Register indirect mode is similar to indirect addressing.
2. The only difference is whether the address field refers to a memory location or a register.
3. Thus, for register indirect address, $EA = (R)$

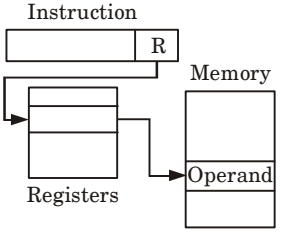


Fig. 1.17.3. Register indirect.

v. Implied mode :

1. In this mode, the operands are specified implicitly in the definition of the instruction.
2. All register reference instructions that use an accumulator are implied mode instructions.
3. Zero address instructions in a stack-organized computer are implied mode instruction since the operands are implied to be on top of the stack. It is also known as stack addressing mode.

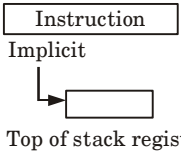


Fig. 1.17.4. Implied mode.

vi. Immediate mode :

1. In this mode, the operand is specified in the instruction itself.
2. The operand field contains the actual operand to be used in conjunction with the operation specified in the instruction.

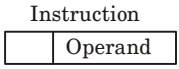


Fig. 1.17.5. Immediate mode.

vii. Indexed :

1. The effective address of the operand is generated by adding a constant value to the contents of a register.
2. The register used may be either a special register for this purpose or more commonly, it may be any one of a set of general purpose registers in the CPU.
3. It is referred to as an index register. We indicate the index mode symbolically as, $X(R)$
   where X denotes a constant and R is the name of register involved.
4. The effective address of the operand is given by, $EA = X + [R]$
5. In the process of generating the effective address, the contents of the index register are not changed.

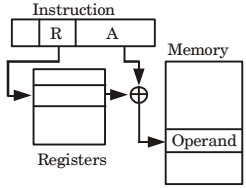


Fig. 1.17.6. Indexed.

Example :

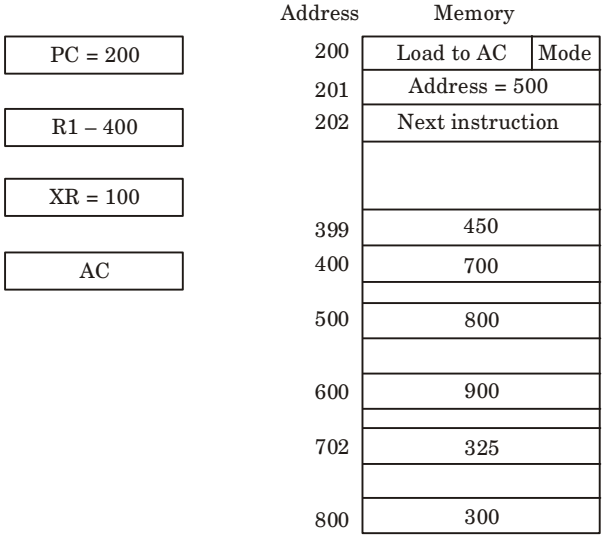


Fig. 1.17.7.

Addressing Mode	Effective Address	Content of AC
Direct address	500	800
Immediate operand	201	500
Indirect address	800	300
Indexed address	600	900
Implied	–	400
Register indirect	400	700

Que 1.18. **What is difference between implied and immediate addressing modes ? Explain with an example.**

Answer

S. No.	Implied addressing mode	Immediate addressing mode
1.	No operand is specified in the instruction.	Operand is specified in the instruction itself.
2.	The operands are specified implicit in the definition of instruction.	The operands are contained in an operands field rather than an address field.
3.	This mode is used in all register reference instructions.	This mode is very useful for initializing the registers to a constant value.
4.	**Example :** The instruction "Complement Accumulator" written as : CMA.	**Example :** The instruction : MVI 06 ADD 05

Que 1.19. **Describe auto increment and auto decrement addressing modes with proper example ?**

Answer

Auto increment mode :

1. In this mode the Effective Address (EA) of the operand is the content of a register specified in the instruction.
2. After accessing the operands, the contents of this register are incremented to point to the next item in the list.

Example : Add (R2) +, R0

Here the contents of R2 are first used as an EA then these are incremented.

Auto decrement mode :

1. In this mode the contents of a register specified in the instruction are decremented.
2. These contents are then used as the effective address of the operand.

Example : Add – (R2), R0

Here the contents of R2 are first decremented and then used as an EA for the operand which is added to the content of R0.

Que 1.20. How addressing mode is significant for referring memory ? List and explain different types of addressing modes.

AKTU 2016-17, Marks 15

Answer

1. The addressing mode is significant for referring memory as it is a code that tells the control unit how to obtain the Effective Address (EA) from the displacement.
2. Addressing mode is a rule of calculation, or a function that use displacement as its main argument and other hardware component as (such as PC, registers and memory locations) as secondary arguments and produce the EA as a result.

Types of addressing modes : Refer Q. 1.17, Page 1–17B, Unit-1.

VERY IMPORTANT QUESTIONS

Following questions are very important. These questions may be asked in your SESSIONALS as well as UNIVERSITY EXAMINATION.

Q. 1. Draw a block diagram of a computer's CPU showing all the basic building blocks such as program counter, accumulator, address and data registers, instruction register, control unit etc., and describe how such an arrangement can work as a computer, if connected properly to memory, input/output etc.

Ans. Refer Q. 1.1.

Q. 2. Write a short note on bus arbitration.

Ans. Refer Q. 1.5.

Q. 3. Discuss the advantages and disadvantages of polling and daisy chaining bus arbitration schemes.

Ans. Refer Q. 1.6.

Q. 4. Explain the operation of three state bus buffers and show its use in design of common bus.

Ans. Refer Q. 1.8.

Q. 5. What is stack ? Give the organization of register stack with all necessary elements and explain the working of push and pop operations.

Ans. Refer Q. 1.12.

Q. 6. What is a memory stack ? Explain its role in managing subroutines with the help of neat diagrams.

Ans. Refer Q. 1.13.

Q. 7. Explain various types of processor organization.

Ans. Refer Q. 1.16.

Q. 8. Explain the following addressing modes with the help of an example each :

i. Direct **ii. Register indirect**
iii. Implied **iv. Immediate**
v. Indexed

Ans. Refer Q. 1.17.

Q. 9. How addressing mode is significant for referring memory ? List and explain different types of addressing modes.

Ans. Refer Q. 1.20.

2
UNIT

Arithmetic and Logic Unit

CONTENTS

PART-1

Arithmetic and Logic Unit : Look Ahead Carries Adders.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 2.1. Describe sequential Arithmetic and Logic Unit (ALU) using proper diagram. **AKTU 2017-18, Marks 07**

Answer

1. By combining arithmetic and logic circuits with the help of multiplexer, we can get the arithmetic and logic units.

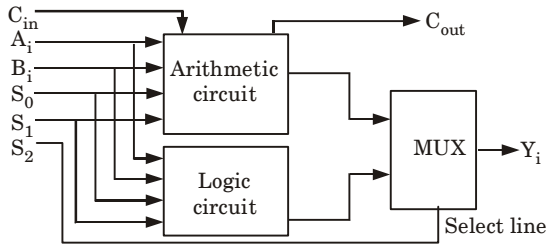


Fig. 2.1.1. Block diagram of ALU.

2. When the mode select line $S_2 = 0$, this ALU acts as an arithmetic circuit, so the output of arithmetic circuit is transferred as final output.
3. Otherwise ($S_2 = 1$) the output of the logic circuit is transferred as final output.
4. Based on the mode select S_2 and input carry C_{in}, we increase or decrease the number of arithmetic and logic operations.
5. When $S_2 = 0$, the ALU performs arithmetic operation and when $S_2 = 1$, with $C_{in} = 0$, the ALU performs logic operations.
6. We know that the carry input is not required in the logic circuits.
7. When logic operation is selected ($S_2 = 1$), the carry input must be zero.
8. This given us the output sum in full adder circuit as

$$Y_i = A_i \oplus B_i \oplus C_i \qquad (\because \ C_i = 0)$$
$$Y_i = A_i \oplus B_i$$

Que 2.2. Write short note on look ahead carry adders.

Answer

1. A Carry Look Ahead adder (CLA) or fast adder is a type of adder used in digital logic.

2. A carry look ahead adder improves speed by reducing the amount of time required to determine carry bits.
3. The carry look ahead adder calculates one or more carry bits before the sum, which reduces the waiting time to calculate the result of the larger-value bits of the adder.
4. Carry look ahead depends on two things :
   a. Calculating for each digit position whether that position is going to propagate a carry if one comes in from the right.
   b. Combining these calculated values to be able to deduce quickly whether, for each group of digits, that group is going to propagate a carry that comes in from the right.
5. Carry look ahead logic uses the concepts of generating and propagating carries.
6. The addition of two binary numbers in parallel implies that all the bits of the augend and addend are available for computation at the same time.
7. The carry propagation time is an important attribute of the adder because it limits the speed with which two numbers are added.
8. A solution is to increase the complexity of the equipment in such a way that the carry delay time is reduced.

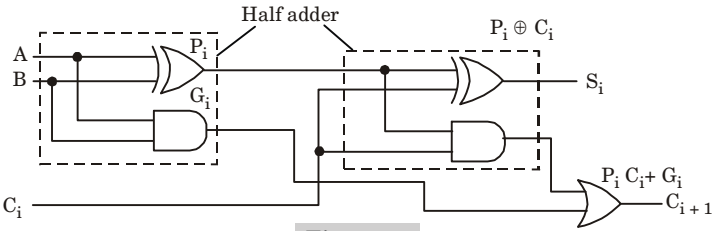


Fig. 2.2.1.

9. Consider the circuit of the full adder shown in Fig. 2.2.1. If we define two new binary variables,

$$P_i = A_i \oplus B_i \, , \, G_i = A_i B_i$$

10. The output sum and carry can respectively be expressed as

$$S_i = P_i \oplus C_i \, , \, C_{i+1} = G_i + P_i C_i$$

11. G_i is called a carry generate, and it produces a carry of 1 when both A_i and B_i are 1, regardless of the input carry C_i. P_i is called a carry propagate, because it determines whether a carry into stage i will propagate into stage $i + 1$.

$$C_0 = \text{input carry}$$

$$C_1 = G_0 + P_0 C_0$$

$$C_2 = G_1 + P_1 C_1 = G_1 + P_1 (G_0 + P_0 C_0)$$

$$= G_1 + P_1 G_0 + P_0 P_1 C_0$$

$$C_3 = G_2 + P_2 C_2 = G_2 + P_2 G_1 + P_2 P_1 G_0 + P_2 P_1 P_0 C_0$$

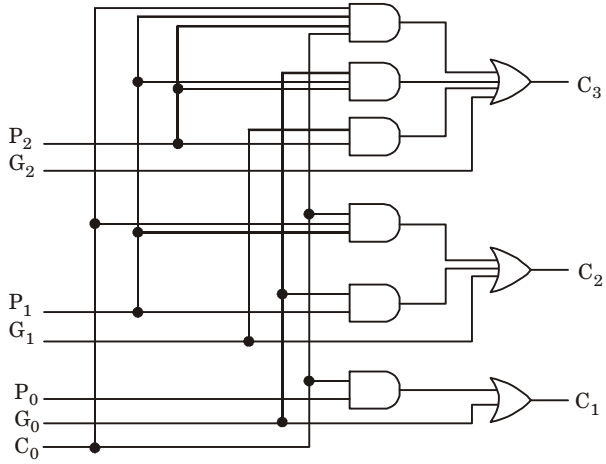


Fig. 2.2.2. Logic diagram of carry look ahead generator.

PART-2

Multiplication : Signed Operand Multiplication, Booth's Algorithm and Array Multipliers.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 2.3. **Explain the Booth's algorithm in depth with the help of flowchart. Give an example for multiplication using Booth's algorithm.**

AKTU 2016-17, Marks 15

OR

Discuss the Booth's algorithm for 2's complement number. Also illustrate it with the some example.

OR

Explain Booth's multiplication algorithm in detail.

AKTU 2017-18, Marks 07

Answer

The algorithm for 2's complement multiplication is as follows :

Step 1 : Load multiplicand in B, multiplier in Q. For negative numbers, 2's complement format to be used.
Step 2 : Initialize the down counter CR by the number of bits involved.
Step 3 : Clean locations A (n-bits) and Q_{n+1} (1-bit).
Step 4 : Check LS bit of Q_n and Q_{n+1} jointly. If the pattern is 00 or 11 then go to Step 5. If 10, then $A = A - B$. If 01, then $A = A + B$.
Step 5 : Perform arithmetic right-shift with A, Q_n and Q_{n+1}. LS of A goes to MS of Q_n and LS of Q goes to Q_{n+1}. Old content of Q_{n+1} is discarded.
Step 6 : Decrement CR by one. If CR is not zero then go to Step 4.
Step 7 : Final result (or the product) is available in A (higher part) and Q_n (lower part).

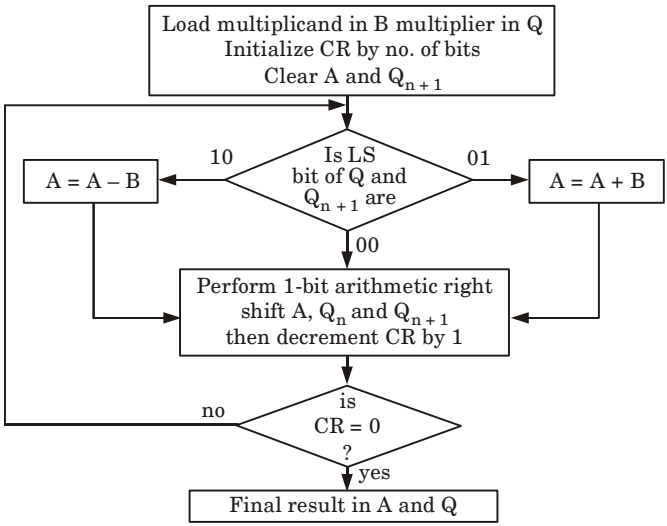


Fig. 2.3.1. 2's Complement multiplication.

Example : Both negative (– 5 × – 4)

Multiplicand (B) ← 1 0 1 1 (– 5)		**Multiplier (Q) ← 1 1 0 0 (– 4)**		
A	***Q_n***	***Q_{n+1}***	**Operation**	**CR**
0 0 0 0	1 1 0 0	0	Initial	4
0 0 0 0	0 1 1 0	0	Shift right	3
0 0 0 0	0 0 1 1	0	Shift right	2
0 1 0 1	0 0 1 1	0	A ← A – B	1
0 0 1 0	1 0 0 1	1	Shift right	
0 0 0 1	0 1 0 0	1	Shift right	0
Result :	0 0 0 1	0 1 0 0	= + 20	

Que 2.4. Explain Booth's algorithm with its hardware implementation.

Answer

1. Fig. 2.4.1 shows the hardware implementation for Booth's algorithm.
2. The circuit is similar to the circuit for positive number multiplication.
3. It consists of an n-bit adder, control logic and four register A, B, Q and Q_{-1}.

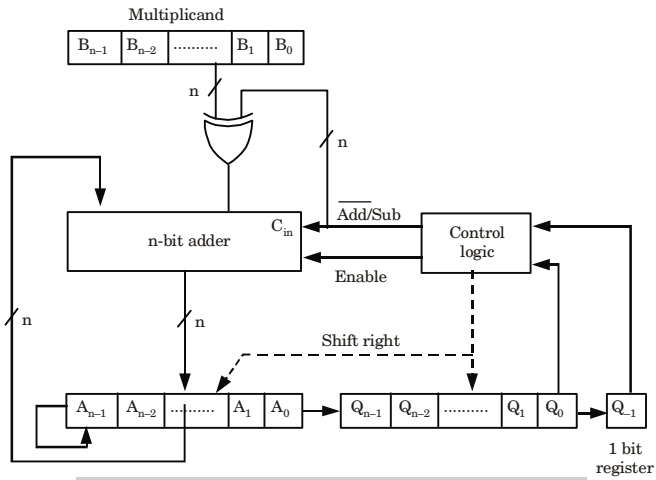


Fig. 2.4.1. Hardware implementation of signed binary multiplication for Booth's algorithm.

4. Multiplier and multiplicand are loaded into register Q and register B respectively.
5. Register A and Q_{-1} and initially set to 0.
6. The n-bit adder performs addition, input of adders comes from multiplicand and content of register A.
7. In case of addition, $\overline{\text{Add}}$ / Sub line is 0, therefore, $C_{in} = 0$ and multiplicand is directly applied as a second input to the n-bit adder.
8. In case of subtraction, $\overline{\text{Add}}$ / Sub line is 1, therefore $C_{in} = 1$ and multiplicand is complemented and then applied to the n-bit adder. As a result, the 2's complement of the multiplicand is added to the content of register A.
9. The control logic scans bit Q_0 and Q_{-1} one at a time and generates the control signals to perform the corresponding function.
10. If the two bits are same (1 – 1 or 0 – 0), then all the bits of A, Q and Q_{-1} register are shifted to right 1 bit without addition or subtraction (Add/Subtract Enable = 0).

11. If the two bits differ, then the multiplicand is added to or subtracted from the A register, depending on the status of bits.
12. After addition or subtraction right shift occurs such that the left most bit of A (A_{n-1}) is not only shifted into A_{n-2}, but also remains in A_{n-1}.
13. This is required to preserve the sign of the number in A and Q.

Que 2.5. **Draw the data path of 2's compliment multiplier. Give the Robertson multiplication algorithm for 2's compliment fractions. Also illustrate the algorithm for 2's compliment fraction by a suitable example.** **AKTU 2017-18, Marks 07**

Answer

Datapath of 2's compliment multiplier : Refer Q. 2.3, Page 2–4B, Unit-2.

Robertson algorithm :

1. $A \leftarrow 0, B \leftarrow$ Multiplicand $Q \leftarrow$ Multiplier and count $\leftarrow n$.
2. If $Q_0 = 1$ then perform $A \leftarrow A + B$.
3. Shift right register $F.A.Q$ by 1 bit $F \leftarrow B\ [n-1]$ AND $Q[0]$ OR F and count $\leftarrow$ count $- 1$.
4. If count > 1 Repeat steps 2 and 3, otherwise if $Q_0 = 1$ then perform $A \leftarrow A - B$ and set $Q\ [0] = 0$.

For example : We perform the multiplication of fraction as :

Multiplicand = 0.625

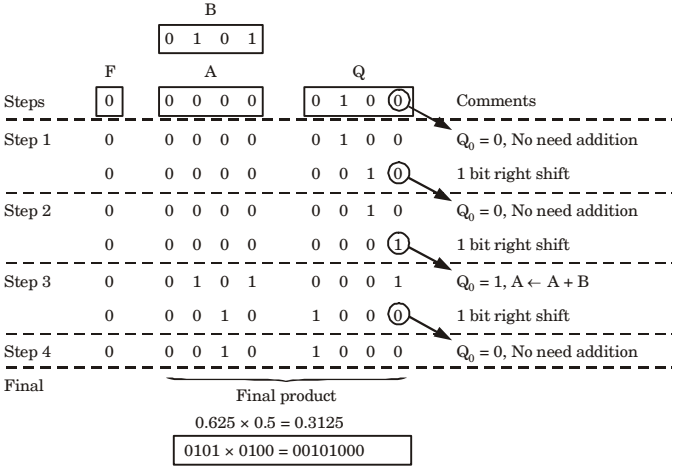
Multiplier = 0.5

Equivalent binary representation 0.625 = 0101

2's complement representation – 0.625 = 1010 + 1 (– 0.625) = 1011

Equivalent binary representation + 0.5 = 0100

2 complement representation – 0.5 = 1011 + 1 – 0.5 = 1100



Que 2.6. **Show step by step the multiplication process using Booth's algorithm when (+ 15) and (– 13) numbers are multiplied. Assume 5-bit registers that hold signed numbers.**

AKTU 2014-15, Marks 10

Answer

15 = 0 1 1 1 1
– 13 = 2's complement of 13 = 1 0 0 1 1
Multiplicand (M) = 0 1 1 1 1, Multiplier = 1 0 0 1 1

A	Q_n	Q_{n+1}	Operation	SC
00000	10011	0		101 (5)
10001	10011	0	A←A – M	
11000	11001	1	Shift	100 (4)
11100	01100	1	Shift	011 (3)
01011	01100	1	A←A + M	
00101	10110	0	Shift	010 (2)
00010	11011	0	Shift	001 (1)
10011	11011	0	A←A – M	
11001	11101	1	Shift	000 (0)
Result = (11001	11101) = – 195(2's complement of +195)			

Que 2.7. **Show the contents of the registers E, A, Q, SC during the process of multiplication of two binary numbers 11111 (multiplicand) 10101 (multiplier). The signs are not included.**

AKTU 2016-17, Marks 10

Answer

Multiplicand B = 11111	**E**	**A**	**Q**	**SC**
	0	**00000**	**10101**	**101**
Q_n = 1 ; add B		00000 11111 11111		
Shift right EAQ	0	01111	11010	100
Q_n = 1 ; add B		01111 11111 00000		
Shift right EAQ	0	00000	01101	011
Q_n = 0 ; shift right EAQ	0	00000	00110	010
Q_n = 0 ; shift right EAQ	0	00000	00011	001
Q_n = 0 ; shift right EAQ	0	00000	00001	000

Que 2.8. **Show the multiplication process using Booth's algorithm when the following numbers are multiplied : (– 13) by (+ 8)**

AKTU 2015-16, Marks 7.5

Answer

True binary equivalent of + 8 = 01000
True binary equivalent of + 13 = 01101
1's complement of + 13 = 10010
+ 1
2's complement of + 13 = 10011 (–13)
Multiplier = 01000
Multiplicand (B) = 10011

A	Q_n	Q_{n+1}	Operation	SC
00000	01000	0		100
00000	00100	0	Ashr AQQ_{n+1}	
00000	00010	0	Ashr AQQ_{n+1}	011
00000	00001	0	Ashr AQQ_{n+1}	010
01101 00110	00001 10000	0 1	Add $\bar{B}$ + 1 to A Ashr AQQ_{n-1}	001
11001 11100	10000 11000	1 0	Add B to A Ashr AQQ_{n+1}	000
Result : 11100	11000 = – 104 (2's complement of (+ 104))			

Que 2.9. **Draw the flowchart of Booth's algorithm for multiplication and show the multiplication process using Booth's algorithm for (– 7) × (+ 3).**

AKTU 2018-19, Marks 07

Answer

Flowchart of Booth's algorithm for multiplication : Refer Q. 2.3, Page 2–4B, Unit-2.

Multiplication : Multiply (– 7) × (+ 3)

Convert (–7) into 2's complement form :

+7 = 0111
1's complement of (+7) = 1000
adding 1 + 1
2's complement of (+ 7) = 1001
(+3) = 0011

A	Q_n	Q_{n+1}	$B = 1001$ $\bar{B} + 1 = 0111$	SC
0000	0011	0	initial values	100
0111			sub B or	
0111			add 0111 to A	
0011	1001	1	Ashr $AQ_n\ Q_{n+1}$	011
0001	1100	1	Ashr $AQ_n\ Q_{n+1}$	010
1001			add 1001	
1010				
1101	0110	0	Ashr $AQ_n\ Q_{n+1}$	001
1110	1011	0	Ashr $AQ_n\ Q_{n+1}$	000

Answer is 11101011

$(-7) \times (+3) = -21 = 11101011$ (2's complement of + 21)

Que 2.10. Explain array multiplier method with the help of example.

Answer

1. The combinational circuit implemented to perform multiplication is called array multiplier.
2. The generalized multiplication process for array multiplier for two unsigned integers : Multiplicand $A = A_3A_2A_1A_0$ and multiplier $B = B_3B_2B_1B_0$ is shown in Fig. 2.10.1.

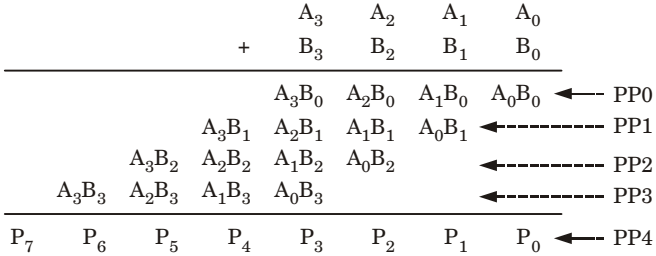


Fig. 2.10.1. Manual multiplication process.

3. Each shifted multiplicand which is multiplied by either 0 or 1 depending on the corresponding multiplier bit is called Partial Product (PP).
4. Each partial product consists of four product components.

$$P_0 = A_0B_0$$
$$P_1 = A_1B_0 + A_0B_1$$

$$P_2 = A_2B_0 + A_1B_1 + A_0B_2$$
$$P_3 = A_3B_0 + A_2B_1 + A_1B_2 + A_0B_3$$
$$P_4 = A_3B_1 + A_2B_2 + A_1B_3$$
$$P_5 = A_3B_2 + A_2B_3$$
$$P_6 = A_3B_3$$

5. The product component bit is a logical AND of multiplier bit B_i and multiplicand bit A_j, *i.e.*, $B_i \times A_j$. Since the arithmetic and logic products coincide in the 1 bit case.
6. Fig. 2.10.2 shows the circuit to add the product components. Here, the product components are represented by AND gates and separated to make space.

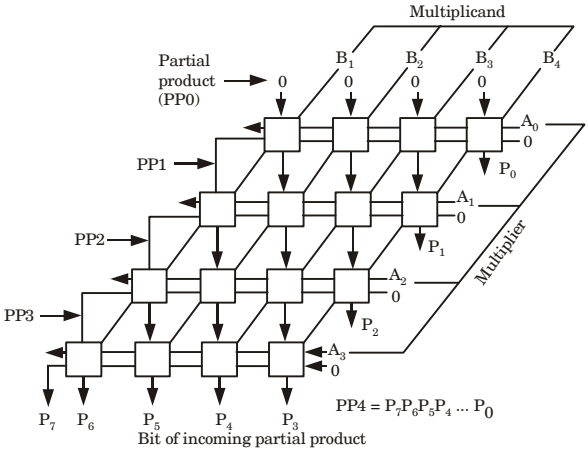


Fig. 2.10.2. Block diagram of combinational multiplier.

7. The full adder block is represented by square block. The carries in each partial product row of full adders are connected to make 4-bit ripple adder.
8. Thus, the first 4-bit ripple adder adds the first two rows of product components to produce the first partial product.
9. The carry output generated is propagated to the most significant product component used to produce the next partial product.
10. The subsequent adders add each partial product with the next product component.

PART-3

Division and Logic Operations.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 2.11. Write down the step for restoring and non-restoring of division operations.

Answer

Restoring division operation :

Step 1 : Shift A and Q left one binary position.

Step 2 : Subtract divisor from A and place answer back in $A(A \leftarrow A - B)$.

Step 3 : If the sign bit of A is 1, set Q_0 to 0 and add divisor back to A (that is, restore A); otherwise, set Q_0 to 1.

Step 4 : Repeat steps 1, 2 and 3 upto n times.

Non-restoring division operation :

Step 1 : If the sign of A is 0, shift A and Q left one bit position and subtract divisor from A; otherwise, shift A and Q left and add divisor to A.

Step 2 : If the sign of A is 0, set Q_0 to 1; otherwise, set Q_0 to 0.

Step 3 : Repeat steps 1 and 2 for n times.

Step 4 : If the sign of A is 1, add divisor to A. Step 4 is required to leave the proper positive remainder in A at the end of n cycles.

Que 2.12. Draw the flow chart for restoring and non-restoring division operation.

Answer

Flowchart for restoring division operation is shown in Fig. 2.12.1.

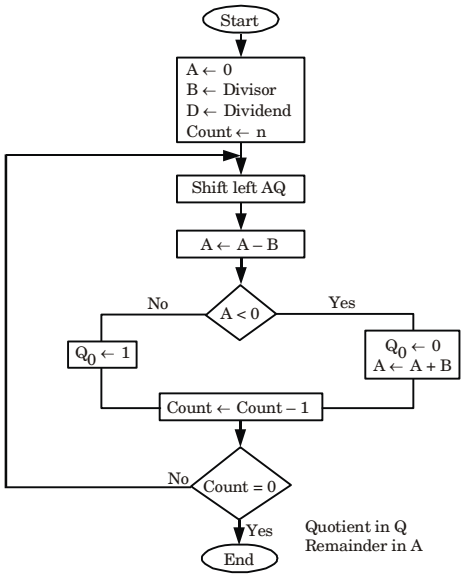


Fig. 2.12.1. Flow chart for restoring division operation.

A flow chart for non-restoring division operation is shown in Fig. 2.12.2.

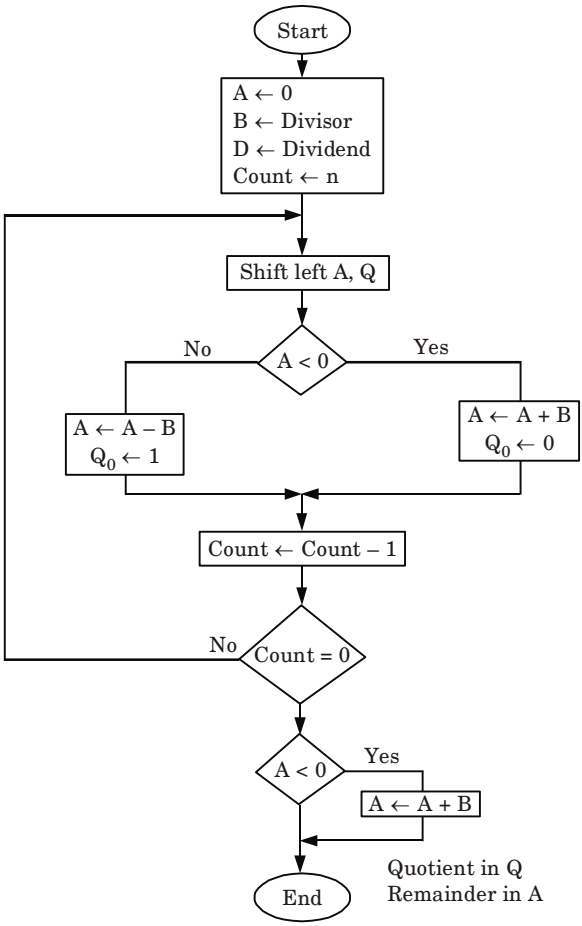


Fig. 2.12.2. Flow chart for non-restoring division operation.

Que 2.13. **Draw the data path of sequential *n*-bit binary divider. Give the non-restoring division algorithm for unsigned integers. Also illustrate algorithm for unsigned integer with a suitable example.** **AKTU 2017-18, Marks 07**

Answer

Datapath of sequential *n*-bit binary divider :

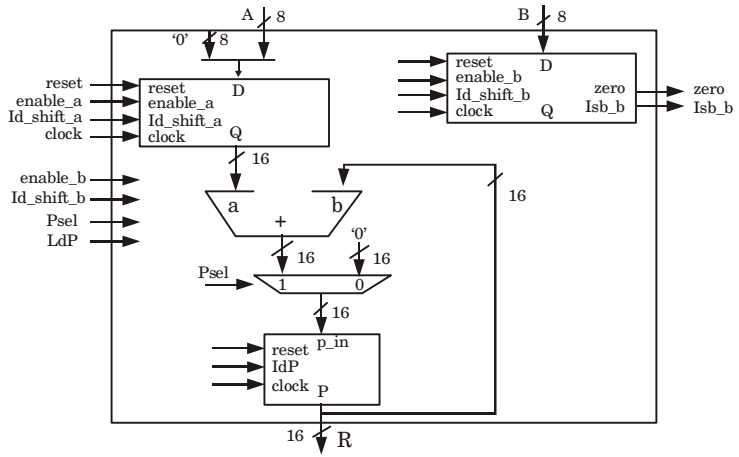


Fig. 2.13.1.

Algorithm for non-restoring division : Refer Q. 2.11, Page 2–12B, Unit-2.

For example, consider 4-bit dividend and 2-bit divisor :

Dividend = 1010, Divisor = 0011

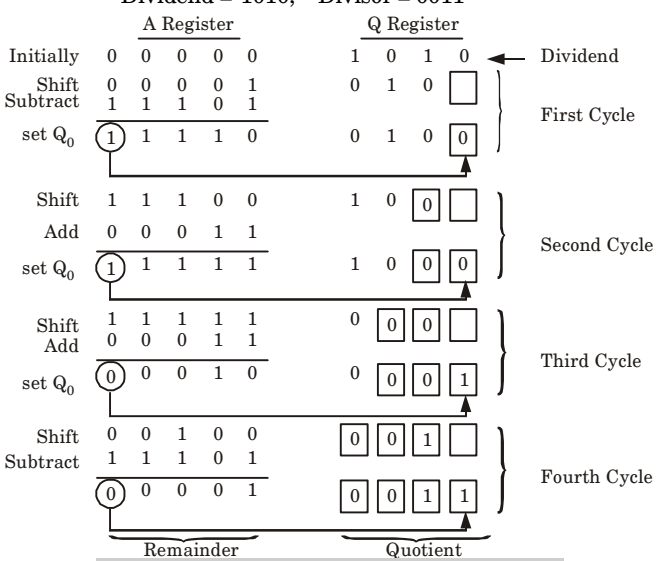


Fig. 2.13.2. A non-restoring division example.

In given example after 4 cycles register A is positive and hence step 3 is not required.

Que 2.14. **Perform the division process of 00001111 by 0011 (use a dividend of 8 bits).** **AKTU 2018-19, Marks 07**

Answer

$B = 0011$ $\quad \overline{B} + 1 = 1101$

Operation	E	A	Q	SC
Dividend in Q, $A = 0$ shl EAQ add $\overline{B} + 1$	0	0000 0001 1101	1111 1110	100
$E = 0$, leave $Q_n = 0$ add B restore partial remainder shl EAQ add $\overline{B} + 1$	0 1 0	1110 0011 0001 0011 1101	1110 1100	011
$E = 1$, set Q_n to 1 shl EAQ add $\overline{B} + 1$	1 0	0000 0001 1101	1101 1010	010
$E = 0$, leave $Q_n = 0$ add B restore partial remainder	0 1	1110 0011 0001	1010 	001
shl EAQ add $\overline{B} + 1$ $E = 1$, set Q_n to 1	0 1	0011 1101 0000 Remainder	0100 0101 Quontient	000

Que 2.15. **What do you mean by overflow ? Describe the overflow detection.**

Answer

Overflow :

1. Overflow is a condition when two numbers with n digits are added and the sum is a number occupying $n + 1$ digit.
2. Overflow is a problem in digital computer because the number of bits cannot be accommodated by an n-bit word.
3. There are following three types of overflow :
   i. **Positive overflow :** Positive overflow refers to integer representations and refers to a number that is larger than that can be represented in a given number of bits.

ii. **Exponent overflow :** Exponent overflow refers to floating point representations and refers to a positive exponent that exceeds the maximum possible exponent value.

iii. **Significand overflow :** Significand overflow occurs when the addition of two significant number of the same sign results in a carry out of the most significant bit.

Overflow detection : An overflow can be detected by observing the carry into the sign bit position. If these two carries are not equal, an overflow condition is produced.

For example :

+ 35	0	100011	– 35	1	011101
+ 40	0	101000	– 40	1	011000
75	1	001011	– 75	10	110101

Two carries are explicitly shown. If the two carries are applied to an exclusive OR gate, an overflow would be detected when the output of the gate is 1.

PART-4

Floating Point Arithmetic Operations, Arithmetic and Logic Unit Design.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

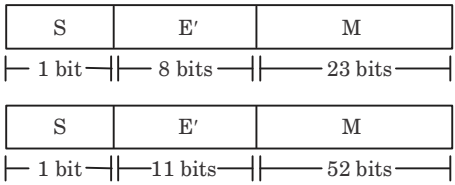
Que 2.16. Explain the basic format used to represent floating point numbers.

Answer

The floating point representation has three fields :

1. **The sign bit :** The sign bit determines whether the number is negative or positive. 0 denotes a positive number and 1 denotes a negative number.
2. **The exponent :** The exponent field needs to represent both positive and negative exponents. To do this, a bias is added to the actual exponent in order to get the stored exponent. For IEEE single precision the exponent field is of 8 bits and has a bias value of 127. For double precision, the exponent field is of 11 bits, and has a bias of 1023.
3. **The mantissa :** The mantissa, also known as the significand, represents the precision bits of the number. It is composed of an implicit leading bit and the fraction bits.

The general structure of floating point number is



Where S is significant (mantissa) digits, E is exponent, B is scaling factor, which is 2 for binary number, 10 for decimal number.

Que 2.17. Write the steps for various floating point arithmetic operations.

Answer

Steps for various floating point arithmetic operations are :

i. Addition and subtraction :
- **Step 1 :** Check for zeros.
- **Step 2 :** Align the mantissas.
- **Step 3 :** Add or subtract the mantissas.
- **Step 4 :** Normalize the result.

ii. Multiplication :
- **Step 1 :** Check for zeros.
- **Step 2 :** Add the exponents.
- **Step 3 :** Multiply the mantissas and determine the sign of the result.
- **Step 4 :** Normalize the product.

iii. Division :
- **Step 1 :** Check for zeros.
- **Step 2 :** Subtract the exponents.
- **Step 3 :** Divide the mantissas and determine the sign of the result.
- **Step 4 :** Normalize the result.

Que 2.18. Explain the function of arithmetic circuit with the help of circuit diagram.

Answer

Arithmetic circuit performs the operation of both addition and subtraction. It has two 4-bit inputs $A_3 A_2 A_1 A_0$ and $B_3 B_2 B_1 B_0$.

4-bit parallel adder/subtractor : It is an arithmetic circuit.

1. The ADD/SUB control line, connected with C_{in} of the full adder, is used to perform the operations of addition and subtraction.
2. The EXOR gates are used as controlled inverters.

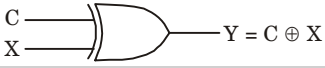


Fig. 2.18.1. Symbol of EXOR gate.

Table : 2.18.1. Truth table of EXOR Gate.

Input		Output
C	*X*	*Y*
0	0	0
0	1	1
1	0	1
1	1	0

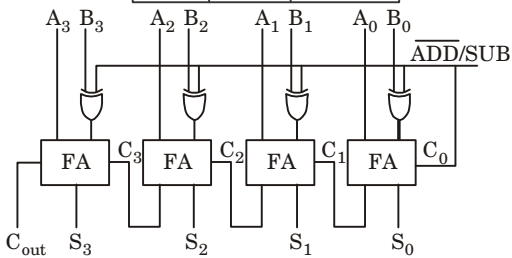


Fig. 2.18.2. A 4-bit parallel binary adder/subtractor.

3. If $C = 0$, the input variable X is either 0 or 1 will be transferred to output terminal.
4. If $C = 1$, the input variable X is either 0 or 1 will be complemented and transferred to output. By using this EXOR gate property we use this gate in the 4-bit adder / subtractor circuit.

Case 1 : ADD/SUB = 1

i. Now, the controlled inverter (EXOR gate) produces the 1's complement of $B_3\,B_2\,B_1\,B_0$. Since 1 is given to C_{in} of the LSB bit of the adder, it is added to the complemented output of EXOR gate output, it is equal to 2's complement of $B_3\,B_2\,B_1\,B_0$.

ii. The 2's complemented $B_3\,B_2\,B_1\,B_0$ will be added to $A_3\,A_2\,A_1\,A_0$ to produce the sum, the produced output of $S_3\,S_2\,S_1\,S_0$ is the difference between $A_3\,A_2\,A_1\,A_0$ and $B_3\,B_2\,B_1\,B_0$.

Case 2 : ADD/SUB = 0.

i. Now, the controlled inverter is transferred $B_3\,B_2\,B_1\,B_0$ four bit to full adder, this 4 bit is added with $A_3A_2A_1A_0$ to produce sum and carry.

Que 2.19. Add – 35 and – 31 in binary using 8-bit registers, in signed 1's complement and signed 2's complement.

AKTU 2014-15, Marks 05

Answer

sign bit ↓

True binary number of 35 = 0 0 1 0 0 0 1 1

True binary number of 31 = 0 0 0 1 1 1 1 1

1's complement of $-35 = 1\ 1\ 0\ 1\ 1\ 1\ 0\ 0$
1's complement of $-31 = +1\ 1\ 1\ 0\ 0\ 0\ 0\ 0$

$1\ 1\ 0\ 1\ 1\ 1\ 1\ 0\ 0$

↑

Discard the carry

2's Complement of $-35 = 1\ 1\ 0\ 1\ 1\ 1\ 0\ 0$

$+1$

$1\ 1\ 0\ 1\ 1\ 1\ 0\ 1$

2's Complement of $-31 = 1\ 1\ 1\ 0\ 0\ 0\ 0\ 0$

$+1$

$1\ 1\ 1\ 0\ 0\ 0\ 0\ 1$

Adding 2's complement of – 35 and – 31

	1	1	0	1	1	1	0	1
+	1	1	1	0	0	0	0	1
1	1	0	1	1	1	1	1	0

↑ ↑

Discard the carry — sign bit

Que 2.20. Draw the block diagram of control unit of basic computer. Explain in detail with control timing diagrams.

AKTU 2016-17, Marks 15

Answer

1. The control unit consists of 2 decoders, 1 sequence counter, number of control logic gates.
2. The instruction in IR is divide into 3 parts : 15^{th} bit to a flip-flop (FF) called I, Operation code, and bits 0 to 11. The Op-code is decoded using 3*8 decoder (D_0 to D_7). Bits 0 to 11 are applied to the control logic gates. The output of a 4-bit sequence counter are decoded into 16 timing signals (T_0 to T_{15}).
3. The SC responds to the positive transition of the clock. Initially CLR I/ P is active, in 1^{st} positive transition SC = 0, timing signal T_0 is active as the output of the decoder. This in turn triggers those registers whose control inputs are connected to T_0. SC is incremented and the timing signals T_0, T_1, T_2, T_3 .... are created. This continues unless SC is cleared. We can clear the SC with decoder output D_3 active, denoted as :

$$D_3T_4 : SC \leftarrow 0$$

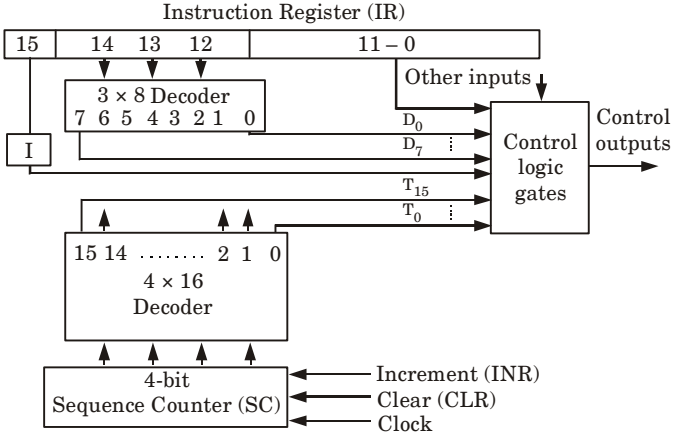


Fig. 2.20.1. Block diagram of control unit.

4. Output D_3 from the operation decoder becomes active at the end of T_2. When T_4 is active, the output of AND gate that implements the control function D_3T_4 becomes active. This signal is applied to CLR input of SC.
5. Example of register transfer : T_0 : AR ← PC (Activities in T_0 will be, Content of PC placed on bus, $S_2S_1S_0 = 010$, LD of AR is active, transfer occurs at the end of positive transition, T_0 is .inactive, T_1 gets active).
6. Timing control is generated by 4-bit sequence counter and 4 × 16 decoder. The SC can be incremented or cleared. $T_0, T_1, T_2, T_3, T_4, T_0$, ...

For example :

Assume : At time T_4, SC is cleared to 0 if decoder output D_3 is active.

$$D_3T_4 : SC \leftarrow 0$$

Timing diagram :

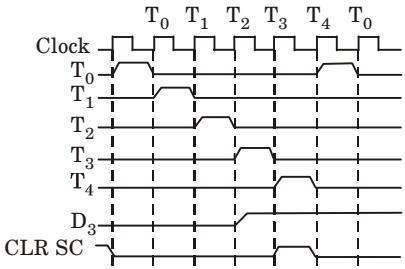


Fig. 2.20.2.

Que 2.21. Draw a flowchart for adding and subtracting two fixed point binary numbers where negative numbers are signed 1's complement presentation. **AKTU 2018-19, Marks 07**

Answer

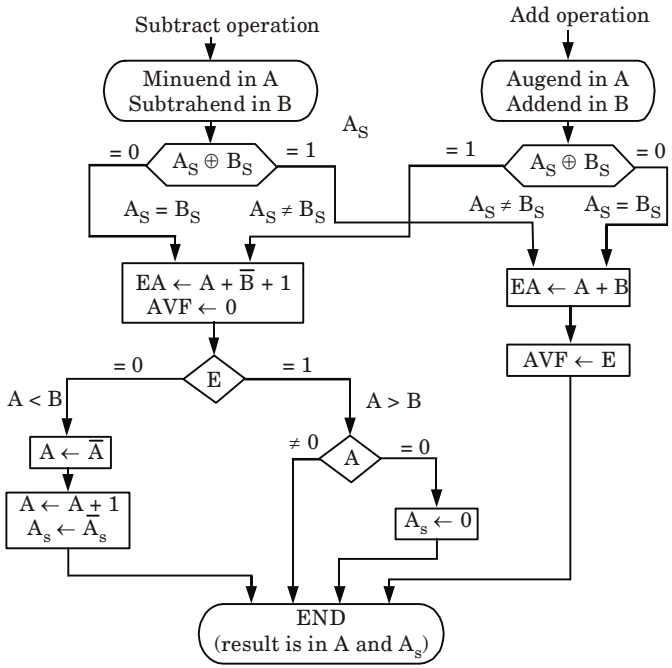


Fig. 2.21.1.

PART-5

IEEE Standard for Floating Point Numbers.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 2.22. **Explain IEEE standard for floating point numbers.**

OR

How floating point numbers are represented in computer, also give IEEE 754 standard 32-bit floating point number format.

AKTU 2017-18, Marks 3.5

Answer

1. The IEEE standard for floating point arithmetic (IEEE 754) is a technical standard for floating point computation.
2. IEEE 754 numbers are divided into two types :
   - **a. Single precision :**
     - i. The floating point numbers in 32-bit are single precision.

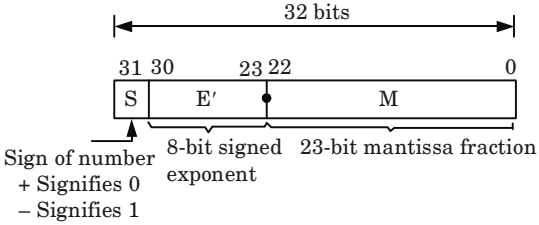


Fig. 2.22.1. Single precision.

     - ii. 32-bit floating point number in single precision is represented as $+ 1M \times 2^E$.
     - iii. The relationship between E and E' in single precision is given as $E' = E + 127$.
     - iv. The 8-bit assigned for exponent E' (Modified exponent) is in the range $0 < E' < 255$ for normal values. Thus the actual exponent is in the range $-127 \le E \le 127$.
     - v. The values 0 and 255 are used to indicates the floating points values of exact 0 and infinite respectively.
   - **b. Double precision :**
     - i. The floating number in 64-bit are double precision. Fig. 2.22.2 show the double precision format of IEEE standard form.
     - ii. The double precision format has increased, exponent and mantissa ranges.
     - iii. The 11 bit assigned for exponent E' has range $0 < E' < 2047$ for normal values. Thus the actual exponent E is in the range $-1022 \le E \le 1023$. The relationship between E and E' in double pricision is given as $E' = E + 1023$.
     - iv. The 52 bit is assigned for mantissa, provides a precision equivalent to about 16 decimal digits.
     - v. 64-bit floating point number in double precision is represented as $\pm 1.M \times 2^E$.

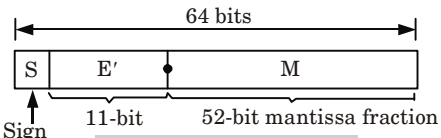


Fig. 2.22.2. Double precision.

Que 2.23. **Represent 1460.125_{10} in single precision and double precision formats.**

Answer

Step 1 : Convert the decimal number in binary format.
For integer,

2	1460	0
2	730	0
2	365	1
2	182	0
2	91	1
2	45	1
2	22	0
2	11	1
2	5	1
2	2	0
	1	

$(1460)_{10} = (10110110100)_2$

For fraction,

$$0.125 \times 2 = 0.250 \Rightarrow 0$$
$$0.250 \times 2 = 0.500 \Rightarrow 0$$
$$0.500 \times 2 = 1.000 \Rightarrow 1$$
$$(0.125)_{10} = (0.001)_2$$
$$(1460.125)_{10} = (10110110100.001)_2$$

Step 2 : Normalize the number.

$$10110110100.001 = 1.0110110100001 \times 2^{10}$$

Single precision :

For a given floating point number,

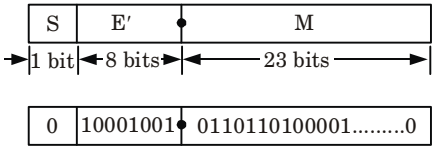
$1.0110110100001 \times 2^{10}$

$$S = 0$$
$$E = 10$$
$$M = 0110110100001$$

Bais (or) modified exponent for single precision

$$E' = 127 + E$$
$$= 127 + 10 = 137_{10}$$
$$= 10001001_2$$

Number is single precision format



Double precision :

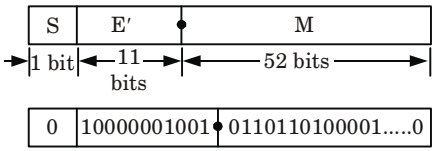
For a given number,

$1.0110110100001 \times 2^{10}$

$S = 0, E = 10, M = 0110110100001$

Bias (or) modified exponent for double precision

$$E' = E + 1023$$
$$= 10 + 1023$$
$$= (1033)_{10} = (10000001001)_2$$



VERY IMPORTANT QUESTIONS

Following questions are very important. These questions may be asked in your SESSIONALS as well as UNIVERSITY EXAMINATION.

Q. 1. Describe sequential Arithmetic and Logic Unit (ALU) using proper diagram.

Ans. Refer Q. 2.1.

Q. 2. Explain Booth's multiplication algorithm in detail.

Ans. Refer Q. 2.3.

Q. 3. Draw the data path of 2's compliment multiplier. Give the Robertson multiplication algorithm for 2's compliment fractions. Also illustrate the algorithm for 2's compliment fraction by a suitable example.

Ans. Refer Q. 2.5.

Q. 4. Show step by step the multiplication process using Booth's algorithm when (+ 15) and (– 13) numbers are multiplied. Assume 5-bit registers that hold signed numbers.

Ans. Refer Q. 2.6.

Q. 5. Show the contents of the registers *E*, *A*, *Q*, *SC* during the process of multiplication of two binary numbers 11111 (multiplicand) 10101 (multiplier). The signs are not included.

Ans. Refer Q. 2.7.

Q. 6. Show the multiplication process using Booth's algorithm when the following numbers are multiplied : (– 13) by (+ 8)

Ans. Refer Q. 2.8.

Q. 7. Draw the flowchart of Booth's algorithm for multiplication and show the multiplication process using Booth's algorithm for (– 7) × (+ 3).

Ans. Refer Q. 2.9.

Q. 8. Draw the data path of sequential *n*-bit binary divider. Give the non-restoring division algorithm for unsigned integers. Also illustrate algorithm for unsigned integer with a suitable example.

Ans. Refer Q. 2.13.

Q. 9. Perform the division process of 00001111 by 0011 (use a dividend of 8 bits).

Ans. Refer Q. 2.14.

Q. 10. Add – 35 and – 31 in binary using 8-bit registers, in signed 1's complement and signed 2's complement.

Ans. Refer Q. 2.19.

Q. 11. Draw the block diagram of control unit of basic computer. Explain in detail with control timing diagrams.

Ans. Refer Q. 2.20.

Q. 12. Draw a flowchart for adding and subtracting two fixed point binary numbers where negative numbers are signed 1's complement presentation.

Ans. Refer Q. 2.21.

Q. 13. How floating point numbers are represented in computer, also give IEEE 754 standard 32-bit floating point number format.

Ans. Refer Q. 2.22.

3
UNIT

Control Unit

CONTENTS

PART-1

Instruction Types, Formats, Instruction Cycles and Sub Cycles (Fetch, Execute etc), Micro-Operation.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 3.1. **What is an instruction in the context of computer organization ? Explain the purpose of the various elements of an instruction with the help of a sample instruction format.**

AKTU 2014-15, Marks 10

Answer

Instruction :

1. Instruction is a command to the processor to perform a given task on specified data.
2. An instruction is a designed binary pattern which is based on the architecture of CPU to perform a specific function. The entire group of instructions is called the instruction set.

Instruction format :

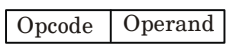


Fig. 3.1.1.

Instruction has two parts opcode and operand,

1. Task to be performed, called the operation code (Opcode), and the data to be operated upon, called the operand.
2. The operands include the input data of the operation and the results that are produced.
3. A computer must have instructions capable of performing four types of operations :
   a. Data transfers between the memory and the CPU registers.
   b. Arithmetic and logic operations on data.
   c. Program sequencing and control.
   d. I/O transfers.
4. The purpose of an instruction is to specify both an operation to be carried out by a CPU or also process the set of operands or data to be used in the operation.

   Example :

This instruction will multiply two operand *A* and *B* and result is stored in *A*.

Que 3.2. Describe the types of instructions on the basis of address fields used in the instruction with example.

Answer

Four types of instructions are available on the basis of referenced address fields :

Now, Let us evaluate the following arithmetic expression, using all instruction types.

$$X = (A + B) * (C + D)$$

1. **Three address instruction :** Computers with three address instruction formats can use each address fields to specify either a processor register or a memory operand. The program in assembly language to evaluate arithmetic expression is shown as :

ADD	R1, A, B	R1 ← M [A] + M [B]
ADD	R2, C, D	R2 ← M [C] + M [D]
MUL	X, R1, R2	M [X] ← R1 * R2

2. **Two address instruction :** In this format each address field can specify either a processor register or a memory word. The assembly language program to evaluate arithmetic expression is as follows :

MOV	R1, A	R1← M [A]
ADD	R1, B	R1 ← R1 + M [B]
MOV	R2, C	R2 ← M [C]
ADD	R2, D	R2 ← R2 + M [D]
MUL	R1, R2	R1 ← R1 * R2
MOV	X, R1	M [X] ← R1

3. **One address instruction :** One address instruction uses an implied accumulator (AC) register for all data manipulation. The program is as follows :

LOAD	A	AC ← M [A]
ADD	B	AC ← AC + M [B]
STORE	T	M [T] ← AC
LOAD	C	AC ← M [C]
ADD	D	AC ← AC + M [D]
MUL	T	AC ← AC * M [T]
STORE	X	M [X] ← AC

All operations are done between the AC register and a memory operand.

4. **Zero address instruction :** A stack organized computer does not use an address field for the instructions ADD and MUL. The PUSH and POP instructions need an address field to specify the operand that communicates with the stack. The program is as follows :

PUSH	A	TOS ← A
PUSH	B	TOS ← B
ADD		TOS ← (A + B)
PUSH	C	TOS ← C

PUSH	D	TOS ← D
ADD		TOS ←(C + D)
MUL		TOS ← (C + D) * (A + B)
POP	X	M [X] ← TOS

where TOS is TOP of the stack.

Que 3.3. **Evaluate the arithmetic statement $X = (A + B)*(C + D)$ using a general register computer with three address, two address and one address instruction format a program to evaluate the expression.** **AKTU 2018-19, Marks 07**

Answer

Three address instruction :

ADD	R1, A, B	R1 ← M[A] + M[B]
ADD	R2, C, D	R2 ← M[C] + M[D]
MUL	X, R1, R2	M[X] ← R1 * R2

Two address instruction :

MOV	R1, A	R1 ← M[A]
ADD	R1, B	R1 ← R1 + M[B]
MOV	R2, C	R2 ← M[C]
ADD	R2, D	R2 ← R2 + M[D]
MUL	R1, R2	R1 ← R1 * R2
MOV	X, R1	M[X] ← R1

One address instruction :

LOAD	A	AC ← M[A]
ADD	B	AC ← A[C] + M[B]
STORE	T	M[T] ← AC
LOAD	C	AC ← M[C]
ADD	D	AC ← AC + M[D]
MUL	T	AC ← AC * M[T]
STORE	X	M[X] ← AC

PART-2

Instruction Cycle and Sub Cycle (Fetch and Execute etc.).

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 3.4. **Define instruction cycle and divide instruction cycle into sub cycles with the help of diagram, explain the sequence in which sub cycles are executed.**

OR

Explain the different cycles of an instruction execution.

OR

Explain all the phases of instruction cycle.

AKTU 2018-19, Marks 3.5

Answer

Instruction cycle :

1. Instruction cycle is a complete process of instruction execution.
2. It is a basic operational process of a computer.
3. It is the process by which a computer retrieves a program instruction from its memory, determines what actions the instruction dictates, and carries out those actions.

The instruction cycle is divided into three sub cycles :

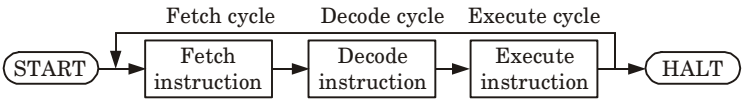


Fig. 3.4.1.

1. **Fetch cycle :** To fetch an opcode from a memory location following steps are performed :
   i. The program counter places the address of the memory location in which the opcode is stored, on the address bus.
   ii. The CPU sends the required memory control signals so as to enable the memory to send the opcode.
   iii. The opcode stored in the memory location is placed on the data bus and transferred to the CPU.
2. **Decode cycle :**
   i. The opcode which is fetched from the memory is placed first of all in the Data Register (DR) (data/address buffer in case of Intel 8085). Thereafter it goes to the Instruction Register (IR).
   ii. From the instruction register it goes to the decoder circuitry, which is within the CPU.
   iii. The decoder circuitry decodes the opcode.
   iv. After the opcode is decoded the CPU comes to know what operation is to be performed, and then execution begins.
3. **Execute cycle :**
   i. In this cycle, function of the instruction is performed.
   ii. If the instruction involves arithmetic or logic, ALU is utilized.

Que 3.5. Write the steps in fetching a word from memory. Differentiate between a branch instruction and call subroutine instruction.

AKTU 2014-15, Marks 10

Answer

Steps in fetching a word from memory are as follows :

Step 1 : The CPU has to perform opcode fetch cycle and operand fetch cycle.

Step 2 : The opcode fetch cycle gives the operation code of fetching a word from memory to the CPU.

Step 3 : The CPU then invokes the operand fetch cycle.

Step 4 : The opcode specifies the address of memory location where the information is stored.

Step 5 : The CPU transfers the address of required word of information to the Address Register (AR), which is connected to the address lines of the memory bus. Hence, the address is transferred to the memory.

Step 6 : The CPU activates the read signal of the memory to indicate that a read operation is needed.

Step 7 : As a result, memory copies data from the addressed register on the data bus.

Step 8 : The CPU then reads this data from the data register and loads it in the specified register.

Step 9 : Memory Functions Completed (MFC) is also used as a control signal for this memory transfer.

Step10 : Memory sets MFC to 1 to indicate that the contents of the specified location have been read and are available on the data bus.

Difference :

S. No.	Branch instruction	Subroutine instruction
1.	Branch instruction is a machine-language or assembly-language instruction.	Subroutine is a control transfer instruction.
2.	It is used to change the sequence of instruction execution.	It is used to call a subroutine.

Que 3.6. **Assuming that all registers initially contain 0, what is the value of R_1 after the following instruction sequence is executed :**

MOV R_1, # 6
MOV R_2, # 5
ADD R_3, R_1, R_1
SUB R_1, R_3, R_2
MUL R_3, R_1, R_1.

Answer

MOV R_1, #6 {Directly move 6 to the register R_1}

MOV R_2, #5	{Take another register in which directly move 5 into it}
ADD R_3, R_1, R_1	{Value stored in R_1 added to R_1 again and then stored in R_3} Thus, R_3 has a value of 12.
SUB R_1, R_3, R_2	{Value of R_2 is subtracted from R_3 and move into the R_1} Thus R_1 has a value of = $R_3 - R_2 = 12 - 5 = 7$
MUL R_3, R_1, R_1	{Multiply R_1 with R_1 and store into R_3}

Thus, the final execution of this statement will give 49 and R_1 contains 7 as value stored in it.

Que 3.7. In an instruction format, there are 16 bits in an instruction word. Bit 0 to 11 convey the address of the memory location for memory related instructions. For non memory instructions these bits convey various register or I/O operations. Bits 12 to 14 show the various basic memory operations such as ADD, AND, LDA etc. Bit 15 shows if the memory is accessed directly or indirectly. For such an instruction format draw block diagram of the control unit of a computer and briefly explain how an instruction will be decoded and executed, by this control unit.

AKTU 2016-17, Marks 10

Answer

1. Consider the instruction code format shown in Fig. 3.7.1(a). It consists of a 3-bit operation code, a 12-bit address, and an indirect address mode bit designated by *I*.
2. The mode bit is 0 for a direct address and 1 for an indirect address.
3. A direct address instruction is shown in Fig. 3.7.1(b). It is placed in address 22 in memory. The *I* bit is 0, so the instruction is recognized as a direct address instruction.
4. The opcode specifies an ADD instruction, and the address part is the binary equivalent of 457.
5. The control finds the operand in memory at address 457 and adds it to the content of *AC*.
6. The instruction in address 35 shown in Fig. 3.7.1(c) has a mode bit $I = 1$. Therefore, it is recognized as an indirect address instruction.
7. The address part is the binary equivalent of 300. The control goes to address 300 to find the address of the operand.
8. The address of the operand in this case is 1350. The operand found in address 1350 is then added to the content of *AC*.
9. The indirect address instruction needs two references to memory to fetch an operand.

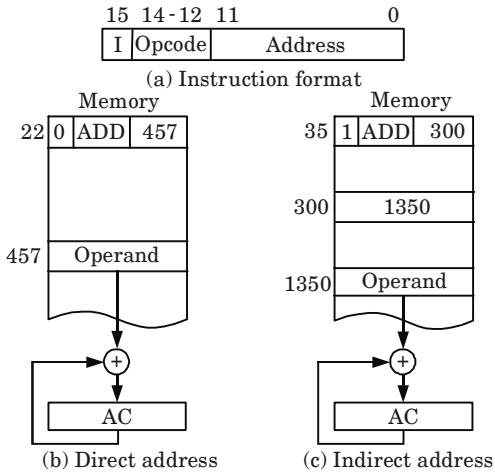


Fig. 3.7.1.

Instruction will be decoded and executed by this control unit :

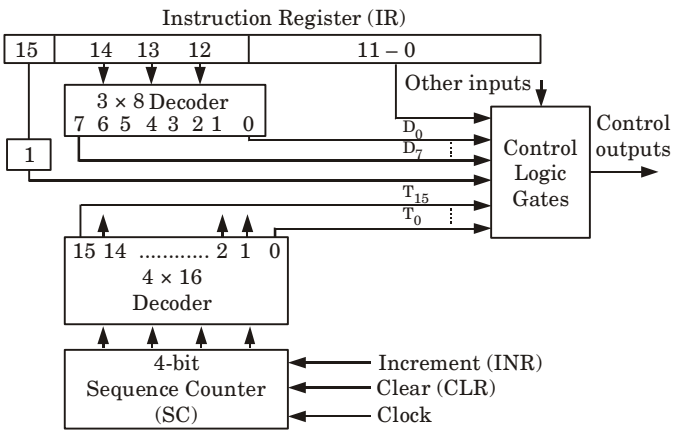


Fig. 3.7.2. Block diagram of control unit of a computer.

1. Control unit consists of :
   i. Instruction register
   ii. Number of control logic gates
   iii. Two decoders
   iv. 4-bit sequence counter
2. An instruction read from memory is placed in the Instruction Register (IR).
3. The instruction register is divided into three parts : the *I* bit, operation code, and address part.

4. First 12-bits (0 – 11) are applied to the control logic gates.
5. The operation code bits (12 – 14) are decoded with a 3 × 8 decoder.
6. The eight outputs (D_0 through D_7) from a decoder go to the control logic gates to perform specific operation.
7. Last bit 15 is transferred to a *I* flip-flop designated by symbol *I*.
8. The 4-bit Sequence Counter (SC) can count in binary from 0 through 15.
9. The counter output is decoded into 16 timing pulses T_0 through T_{15}.
10. The sequence counter can be incremented by INR input or clear by CLR input synchronously.

PART-3

Micro-operation Execution of Complete Instruction.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 3.8. Describe micro-operation and enlist its types.

Answer

1. A micro-operation is a simple operation that can be performed during one clock period.
2. The result of this operation may replace the previous binary information of register or the result may be transferred to another register.
3. Examples of micro-operations are shift, move, count, add and load etc.
4. The micro-operations most often encountered in digital computers are classified into four categories :
   - **i. Register transfer micro-operations :** It transfer binary information from one register to another.
   - **ii. Arithmetic micro-operations :** It perform arithmetic operation on numeric data stored in registers.
   - **iii. Logic micro-operations :** It perform bit manipulation operations on non-numeric data stored in registers.
   - **iv. Shift micro-operations :** It perform shift operations on data stored in registers.

Que 3.9. Write a short note on register transfer micro-operation.

Answer

1. Register transfer is defined as information transfer from one register to another and is designated in symbolic form by means of a replacement operator.

2. The statement denotes a transfer of the content of register $R1$ into register $R2$.

$$R2 \leftarrow R1$$

3. It designates a replacement of the content of $R2$ by the content of $R1$. By definition, the content of the source register $R1$ does not change after the transfer.
4. Every statement written in a register transfer notation implies a hardware construction for implementing the transfer.
5. Fig. 3.9.1 shows the block diagram that depicts the transfer from $R1$ to $R2$. The n outputs of register $R1$ are connected to the n inputs of register $R2$.

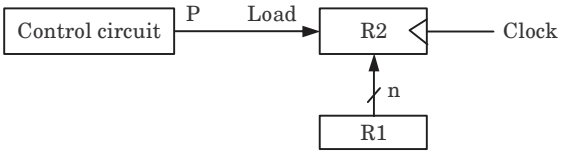


Fig. 3.9.1. Block diagram.

6. The letter n will be used to indicate any number of bits for the register. It will be replaced by an actual number when the length of the register is known.
7. Register $R2$ has a load input that is activated by the control variable P.
8. The basic symbols of the register transfer notations are listed in Table 3.9.1.

Table 3.9.1. Basic symbols for register transfers.

S. No.	Symbol	Description	Examples
1.	Letters (and numerals)	Denotes a register	MAR, $R2$
2.	Parentheses ( )	Denotes a part of a register	$R2(0–7)$, $R2(L)$
3.	Arrow ←	Denotes transfer of information	$R2 \leftarrow R1$
4.	Comma,	Separates two micro-operations	$R2 \leftarrow R1, R1 \leftarrow R2$

Que 3.10. **Write short note on arithmetic micro-operation.**

Answer

1. The basic arithmetic micro-operations are addition, subtraction, increment and decrement. The arithmetic micro-operation defined by the statement, $R3 \leftarrow R1 + R2$
which specifies an addition micro-operation.
2. It states that the contents of register $R1$ are added to the contents of register $R2$ and the sum is transferred to register $R3$.
3. Subtraction is implemented through complementation and addition, which is specified in following statement :

$$R3 \leftarrow R1 + \overline{R2} + 1$$

$\overline{R2}$ is the symbol for the 1's complement of $R2$. Adding 1 to the 1's complement produce the 2's complement.

4. Adding the contents of $R1$ to the 2's complement of $R2$ is equivalent to $R1 - R2$. The other basic arithmetic micro-operations are listed in Table 3.10.1.

Table 3.10.1 : Arithmetic micro-operations.

S. No.	Symbolic designation	Description
1.	$R3 \leftarrow R1 + R2$	Contents of $R1$ plus $R2$ transferred to $R3$.
2.	$R3 \leftarrow R1 - R2$	Contents of $R1$ minus $R2$ transferred to $R3$.
3.	$R2 \leftarrow \overline{R2}$	Complement the contents of $R2$ (1's complement).
4.	$R2 \leftarrow \overline{R2} + 1$	2's complement the contents of $R2$ (negate).
5.	$R3 \leftarrow R1 + \overline{R2} + 1$	$R1$ plus the 2's complement of $R2$ (subtraction).
6.	$R1 \leftarrow R1 + 1$	Increment the contents of $R1$ by one.
7.	$R1 \leftarrow R1 - 1$	Decrement the contents of $R1$ by one.

Que 3.11. Write a short note on logic micro-operation.

Answer

1. Logic micro-operations specify binary operations for strings of bits stored in registers.
2. These operations consider each bit of the register separately and treat the contents of two registers $R1$ and $R2$ symbolized by the statement

$$P : R1 \leftarrow R1 \oplus R2$$

3. It specifies a logic micro-operation is to be executed on the individual bits of the registers provided that the control variable $P = 1$.
4. For example, the content of $R1$ is 1010 and content of $R2$ is 1100. The logic computation :

1 0 1 0 content of $R1$
1 1 0 0 content of $R2$
0 1 1 0 content of $R1$ after $P = 1$

5. There are 16 different logic operations that can be performed with two binary variables.
6. The Boolean functions of two variables x and y are expressed in algebraic form in first column of Table 3.11.1.

Table 3.11.1 : Sixteen logic micro-operations.

S. No.	Boolean function	Micro-operation	Name
1.	$F_0 = 0$	$F \leftarrow 0$	Clear
2.	$F_1 = xy$	$F \leftarrow A \wedge B$	AND

3.	$F_2 = xy'$	$F \leftarrow A \wedge \overline{B}$	
4.	$F_3 = x$	$F \leftarrow A$	Transfer A
5.	$F_4 = x'y$	$F \leftarrow \overline{A} \wedge B$	
6.	$F_5 = y$	$F \leftarrow B$	Transfer B
7.	$F_6 = x \oplus y$	$F \leftarrow A \oplus B$	Exclusive-OR
8.	$F_7 = x + y$	$F \leftarrow A \vee B$	OR
9.	$F_8 = (x + y)'$	$F \leftarrow \overline{A \vee B}$	NOR
10.	$F_9 = (x \oplus y)'$	$F \leftarrow \overline{A \oplus B}$	Exclusive-NOR
11.	$F_{10} = y'$	$F \leftarrow \overline{B}$	Complement B
12.	$F_{11} = x + y'$	$F \leftarrow A \vee \overline{B}$	
13.	$F_{12} = x'$	$F \leftarrow \overline{A}$	Complement A
14.	$F_{13} = x' + y$	$F \leftarrow \overline{A} \vee B$	
15.	$F_{14} = (xy)'$	$F \leftarrow \overline{A \wedge B}$	NAND
16.	$F_{15} = 1$	F← all 1's	Set to all 1's

Que 3.12. Write a short note on shift micro-operations.

OR

List and explain different types of shift micro-operation.

AKTU 2016-17, Marks 15

Answer

Shift micro-operation :

1. Shift micro-operations in computer architecture are those which are used in serial shifting of data present in a register.
2. Shift micro-operations move or shift data in a register bitwise that is, one bit at a time either left or right from its original position.

List of different types of shift micro-operation :

a. Arithmetic shift micro-operation :

i. Arithmetic shift operation shifts signed (positive or negative) binary numbers either left or right by multiplying or dividing by 2.

ii. For arithmetic shift left micro-operation, the value in the register is multiplied by 2 and whereas for arithmetic shift right micro-operation, the value in the register is divided by 2.

iii. In RTL (Register Transfer Language), we can represent this arithmetic shift micro-operations as

$R \leftarrow$ ashl R (arithmetic shift left R (register))

$R \leftarrow$ ashr R (arithmetic shift right R (register))

iv. Fig. 3.12.1 showing arithmetic shift left operation is as follows :

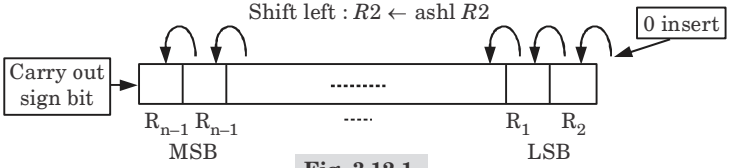


Fig. 3.12.1.

v. Fig. 3.12.2 showing arithmetic shift right operation is as follows :

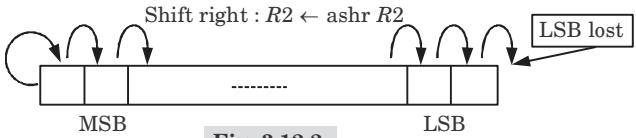


Fig. 3.12.2.

b. Logical shift micro-operation :

i. A logical shift micro-operation transfers a 0 (zero) through the serial input, either from left or right depending on the type.

ii. For logical shift left micro-operation, 0 (zero) is transferred through the right of the data and for the logical shift right micro-operation, 0 (zero is transferred through the left of the data as shown in the Fig. 3.12.3.

iii. Register Transfer Language (RTL) for the logical shift micro-operations can be written as :

$R \leftarrow$ shl R (shift left register (R)).

$R \leftarrow$ shr R (shift right register (R)).

iv. Fig. 3.12.3 showing logical shift left micro-operation on the data in a register.

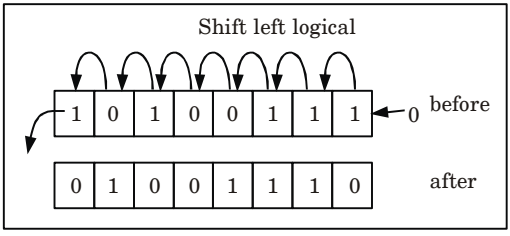


Fig. 3.12.3.

v. Fig. 3.12.4 showing the logical shift right is as follows :

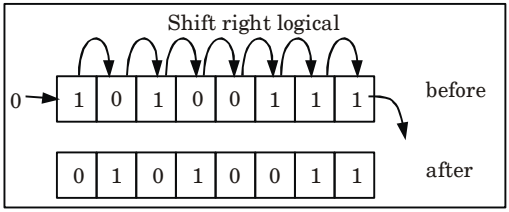


Fig. 3.12.4.

c. Circular shift micro-operation :

i. A circular shift micro-operation performs the shifting of bits from one end of the register to the other end of the register.

ii. In circular shift left operation, the leftmost bit in the register is transferred to the rightmost end and in the circular shift right operation, the rightmost bit in the register is transferred or shifted to the leftmost end of the register as shown in the Fig. 3.12.5 and Fig. 3.12.6 respectively.

iii. Register transfer language for the circular shift micro-operations can be written as :

$R \leftarrow$ cil R (circular shift left register (R)).

$R \leftarrow$ cir R (circular shift right register (R)).

iv. Fig. 3.12.5 showing circular shift left micro-operation.

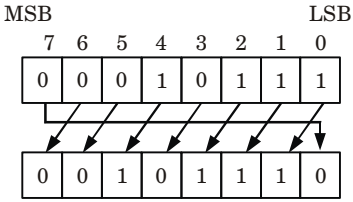


Fig. 3.12.5.

v. Fig. 3.12.6 showing circular shift right micro-operation.

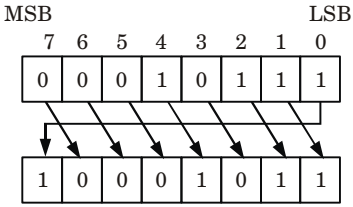


Fig. 3.12.6

Que 3.13. What are the different categories of micro-operations that may be carried out by CPU ? Explain each category of micro-operations giving one example for each.

AKTU 2014-15, Marks 10

Answer

Different categories of micro-operation are :

1. **Register transfer micro-operation :** Refer Q. 3.9, Page 3–9B, Unit-3.
2. **Arithmetic micro-operation :** Refer Q. 3.10, Page 3–10B, Unit-3.
3. **Logic micro-operation :** Refer Q. 3.11, Page 3–11B, Unit-3.
4. **Shift micro-operation :** Refer Q. 3.12, Page 3–12B, Unit-3.

Que 3.14. Discuss the execution of a complete instruction.

Answer

1. The execution of a complete instruction is explained by a simple addition instruction.
2. Consider the instruction,
   $$\text{ADD } [M], R_1$$
   which adds the contents of memory location (address is specified) to the register R_1.
3. The execution of this instruction requires the following steps :
   a. Fetch the instruction.
   b. Fetch the operand.
   c. Perform the addition.
   d. Load the result into R_1.
4. The first step of fetching the instruction is common to execution of all instructions.
5. The remaining steps depend upon the operation to be performed.
6. Instruction execution proceeds as follows :
   Step 1 : The instruction fetch operation is initiated by loading the contents of the Program Counter (PC) into MAR and sending read request to memory to read the instruction from memory (Now waiting for response from the memory)
   Step 2 : Memory read is requested.
   Step 3 : The processor is to be delayed until the MFC (Memory Function Completed) signal is received. Once MFC is received the word fetched from the memory is transferred to instruction register.
   Step 4 : The contents of memory are transferred by memory read operation.
   Step 5 : The contents of R_1 are transferred to one of the inputs of ALU. Now both operands are available in ALU inputs.
   Step 6 : Addition operation is performed.
   Step 7 : The result is transferred to R_1.
   Step 8 : It indicates that the end of the execution of the current instruction, and it causes a new fetch cycle to begin by returning to step 1.

PART-4

Program Control, Reduced Instruction Set Computer.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 3.15. What is CISC ? Explain its characteristics.

AKTU 2015-16, Marks 05

Answer

1. The term "CISC" (Complex Instruction Set Computer or Computing) refers to computers designed with a full set of computer instructions that were intended to provide needed capabilities in the most efficient way.
2. CISC has complex instruction sets variable length encoding of instructions and instruction execution takes varying number of clock cycles.
3. Due to large number of addressing modes for the operations and instructions, the CISC computer generally require fewer instructions to perform the computation.
4. Programs writing for CISC architectures tend to take less space in memory.

Characteristics of CISC :

1. If the frequency of complex operation is high, then the performance of the CISC machine is better to implement.
2. CISC tends to have many instruction formats to accommodate more opcode types and operand addressing method.
3. CISC has many instruction formats.
4. CISC processor provides direct manipulation of operands residing in memory.

Que 3.16. Discuss the advantages and disadvantages of using a variable length instruction format.

Answer

Advantages of using a variable length instruction format :

1. In variable length instructions format, it is easy to provide a large collection of opcodes with different opcode lengths.
2. Addressing can be more flexible with various combinations of register and memory references plus addressing modes.

3. The use of variable length instructions does not remove the desirability of making all of the instruction lengths integrally related to the word length.
4. It is more efficient.
5. It is more compact with a combination of register memory addressing modes.

Disadvantages of using a variable length instruction format :

1. The main disadvantage of variable length instructions is increase in the complexity of the processor.
2. It does not remove desirability of instruction length which is integrally related to word length.

Que 3.17. **Write a short note on RISC.**

AKTU 2014-15, Marks 05

OR

What is RISC ? Explain its various characteristics.

AKTU 2015-16, Marks 05

Answer

1. RISC (Reduced Instruction Set Computer) processor instruction has a fixed length encoding of instruction and each instruction executes in a single clock cycle by hardwired implementation of each instruction.
2. RISC architecture focus on reducing the number of instructions and working with simpler instruction set having limited number of addressing modes and allowing them to execute more instructions in the same amount of time.
3. Programs written for RISC architectures tend to make more space in memory but RISC processor's increased clock rate allows it to execute its program in less time than a CISC processor takes to execute its program.

RISC characteristics :

1. Simple instructions are used in RISC architecture.
2. RISC helps and supports few simple data types and synthesizes complex data types.
3. RISC utilizes simple addressing modes and fixed length instructions for pipelining.
4. RISC permits any register to use in any context.

Que 3.18. **Differentiate between RISC & CISC based microprocessor.**

AKTU 2017-18, Marks 07

OR

Differentiate between complex instruction set computer and reduced instruction set computer.

OR

Give the detailed comparison between RISC and CISC.

AKTU 2016-17, Marks 10

Answer

S. No.	RISC	CISC
1.	Multiple register sets, often consisting of more than 256 registers.	Single register set, often consisting 6 to 16 registers total.
2.	Three register operands allowed per instruction (for example, add R1, R2, R3).	One or two register operands allowed per instruction (for example, add R1, R2).
3.	Parameter passing through efficient on-chip register windows.	Parameter passing through inefficient off-chip memory.
4.	Single-cycle instructions (except for load and store).	Multiple-cycle instructions.
5.	Hardwired control.	Micro-programmed control.
6.	Highly pipelined.	Less pipelined.
7.	Simple instructions are few in number.	There are many complex instructions.
8.	Fixed length instructions.	Variable length instructions.
9.	Complexity in compiler.	Complexity in microcode.
10.	Only load and store instructions can access memory.	Many instructions can access memory.
11.	Few addressing modes.	Many addressing modes.

PART-5

Pipelining.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 3.19. What is pipelining ? How the idea of pipelining used in a computer ?

OR

Write a short note on pipelining. **AKTU 2018-19, Marks 3.5**

Answer

1. Pipelining is a technique of decomposing a sequential process into sub-operations, with each sub-process being executed in a special dedicated segment that operates concurrently with all other segments.
2. The processor executes a program by fetching and executing instructions, one after the other.
3. Let F_i and E_i refer to the fetch and execute steps for instruction I_i.
4. Execution of a program consists of a sequence of fetch and execute steps as shows is Fig. 3.19.1.

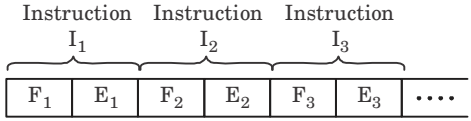


Fig. 3.19.1. Sequential execution.

5. Now consider a computer that has two separate hardware units, one for fetching instructions and another for executing them, as shown in Fig. 3.19.2.
6. The instruction fetched by the fetch unit is deposited in an intermediated storage buffer B_i.
7. The results of execution are deposited in the destination location specified by the instructions.
8. For these purposes, we assume that both the source and destination of the data operated in by the instructions are inside the block labelled "Execution unit".

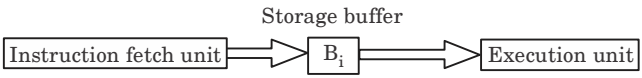


Fig. 3.19.2. Hardware organization.

9. The computer is controlled by a clock whose period is such that the fetch and execute steps of any instruction can be completed in one clock cycle.
10. Operation of the computer proceeds as in Fig. 3.19.3.

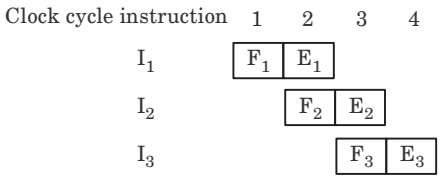


Fig. 3.19.3.

11. In the first clock cycle, the fetch unit fetches an instruction I_1 (step F_1) and stores it in buffer B_i at the end of the clock cycle.

12. In the second clock cycle, the instruction fetch unit proceeds with the fetch operation for instruction I_2 (Step F_2). Meanwhile, the execution unit performs the operation specified by the instruction I_1, which is available to it in buffer B_i (Step E_1). By the end of the second clock cycle, the execution of instruction I_1 is completed and instruction I_2 is available.
13. Instruction I_2 is stored in B_i by replacing I_1, which is no longer needed. Step E_2 is performed by the execution unit during the third clock cycle, while instruction I_3 is being fetched by the fetch unit. In this manner, both fetch and execute units are kept busy all the time.

Que 3.20. How pipelining is classified ?

OR

Write short notes on instruction pipeline.

AKTU 2018-19, Marks 3.5

Answer

Classification of pipeline :

1. Arithmetic pipelining :

a. The arithmetic logic units of a computer can be segmentized for pipeline operations in various data format as shown in Fig. 3.20.1.
b. The four-stage pipeline used in Star-100, the eight-stage pipeline used in the TI-ASC, the up to 14 stages pipeline used in the Cray-1 are the example of arithmetic pipeline.

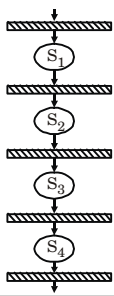


Fig. 3.20.1. Arithmetic pipelining.

2. Processor pipelining :

a. This refers to pipeline processing of the same data stream by a cascade of processors each of which processes a specific task as shown in Fig. 3.20.2.
b. The data stream passes the first processor with results stored in a memory block, which is also accessible, by the second processor.
c. The second processor then passes the refine results to the third, and so on.
d. The pipelining of multiple processors is not yet well accepted as a common practice.

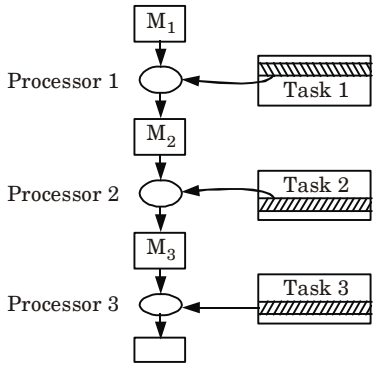


Fig. 3.20.2. Processor pipelining.

3. Instruction pipelining :

a. The execution of a stream of instructions can be pipelined by overlapping the execution of the current instruction with the fetch, decode, and operand fetch of subsequent instructions as shown in Fig. 3.20.3.

b. This technique is also known as instruction look ahead.

c. Almost all high-performance computers are now equipped with instruction-execution pipelines.

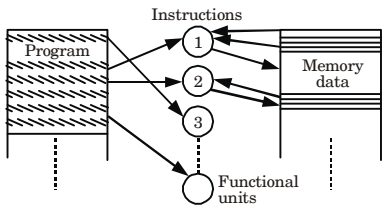


Fig. 3.20.3. Instruction pipelining.

Que 3.21. What are the various pipeline performance measures ?

Answer

There are following terms which are used to measure a pipeline performance :

1. Speed-up 2. Efficiency 3. Throughput

Consider a 'k' segment pipeline with clock cycle time as 'T_p'. Let there be 'n' tasks to be completed in the pipelined processor. Now, the first instruction is going to take 'k' cycles to come out of the pipeline but the other '$n - 1$' instructions will take only '1' cycle each, *i.e.*, a total of '$n - 1$' cycles. So, time taken to execute 'n' instructions in a pipelined processor :

$$ET_{\text{pipeline}} = k + n - 1 \text{ cycles} = (k + n - 1)\, T_p$$

In the same case, for a non-pipelined processor, execution time of 'n' instructions will be :

$$ET_{\text{non-pipeline}} = n * k * T_p$$

So, speed-up (S) of the pipelined processor over non-pipelined processor, when 'n' tasks are executed on the same processor is :

S = Performance of pipelined processor / Performance of non-pipelined processor

As the performance of a processor is inversely proportional to the execution time, we have,

$$S = ET_{\text{non-pipeline}} \; / \; ET_{\text{pipeline}}$$
$$S = [n * k * T_p] \; / \; [(k + n - 1) * T_p]$$
$$S = [n * k] \; / \; [k + n - 1]$$

When the number of tasks 'n' are significantly larger than k, that is, $n >> k$

$$S \approx n * k \; / \; n \approx k$$

where 'k' are the number of stages in the pipeline.

Also, Efficiency = Given speed-up / Max speed up = $S / S_{\max}$

We know that, $S_{\max} = k$

So, Efficiency = S / k

Throughput = Number of instructions / Total time to complete the instructions

So, Throughput = $n \; / \; (k + n - 1) * T_p$

Que 3.22. **Differentiate between linear and non-linear pipeline.**

Answer

S. No.	Linear pipeline	Non-linear pipeline
1.	Static pipeline	Dynamic pipeline
2.	Allows only streamline connections.	Allows feed-forward and feedback connections in addition to the streamline connection.
3.	Function partitioning is easy.	Function partitioning is relatively difficult.
4.	The output is produced from the last stage.	The output is not necessarily produced from the last stage.
5.	The reservation table is trivial in the sense that data flows in linear streamline.	The reservation table is non-trivial in the sense that there is no linear streamline for data flows.
6.	Have single reservation table.	Have more than one reservation table.
7.	All initiations to a static pipeline use the same reservation table.	A dynamic pipeline may allow different initiations to follow a mix reservation tables.
8.	Used to perform fixed function.	Use to perform variable function at different time.

PART-6

Hardwired and Micro-programmed Control : Micro-programme Sequencing.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 3.23. Explain hardwired control unit. What are the methods to design hardwired controllers ? **AKTU 2017-18, Marks 3.5**

OR

What do you understand by hardwired control ? Give various methods to design hardwired control unit. Describe any one method used for designing of hardwired control unit.

Answer

Hardwired control unit :

1. It is a controller as a sequential logic circuit or a finite state machine that generates a sequence of control signals in response to the externally supplied instructions.
2. The control logic is implemented with gates, flip-flops, decoders, and other digital circuits.

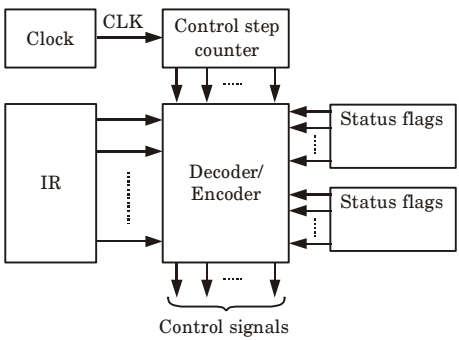


Fig. 3.23.1. General block diagram of hardwired control.

3. A hardwired control requires changes in the wiring among the various components if the design has to be modified or changed.
4. Its input logic signals are transformed into a set of output logic signals are called control signals.
5. Each step in this sequence is completed in one clock cycle.

6. A counter may be used to keep the track of the control steps.
7. The required control signals are determined by the following information :
   a. Contents of the control step counter.
   b. Contents of the instruction register.
   c. Contents of the condition code flags.
   d. External input signals such as MFC and interrupt request.

Methods to design hardwired control : There are four simplified and systematic methods for the design of hardwired controllers.

1. State table method or one-hot method.
2. Delay element method.
3. Sequence-counter method.
4. PLA method.

PLA method :

1. PLA (Programmable Logic Array) is an important device of digital system.
2. We can implement a large combinational logic circuit or sequential logic circuit in a PLA.

Fig. 3.23.2. PLA control unit.

3. It replaces more number of MSI (Medium Scale Integration) and SSI (Small Scale Integration) by its single device chip.
4. The decision logic function and decoder functions are implemented in the Programmable Logic Array (PLA).
5. It is possible to reduce the number of ICs and the number of interconnection wires. The PLA output is the next state of the sequence register.

Que 3.24. Discuss the basic structure of micro-program control unit.

OR

Explain micro-programmed control unit.

Answer

1. Micro-programmed control is a method of control unit design in which the control signal selection and sequencing information is stored in a ROM or RAM called a Control Memory (CM).

2. The sequence of control signals to be generated by the controller can be stored in a special Read Only Memory (ROM) also called Control Memory (CM).
3. Memory control word is written for each micro-operation, and these control words are stored in a serial ascending memory location.
4. The control word is accessed serially (serial access memory) from the control memory.
5. Control words are stored in the ROM permanently.
6. The output of the control memory provides the required control signals.

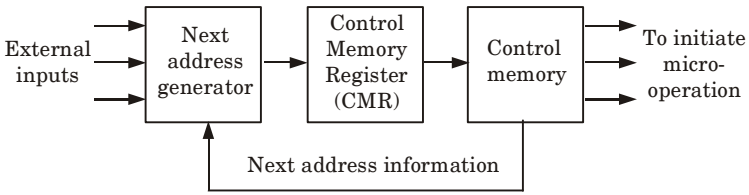


Fig. 3.24.1. Block diagram of micro-programmed control unit.

7. If the control memory is sequentially accessed by incrementing control memory location, then the sequence of control signals stored in successive word of ROM can be generated.
8. The storage of control word in a ROM is often referred to as firmware.

Que 3.25. Compare and contrast hardwired and micro-programmed control units. Also lists their advantages and disadvantages. **AKTU 2014-15, Marks 10**

OR

Explain hardwired and micro-programmed control and compare them.

OR

What are the differences between hardwired and micro-programmed control unit ? **AKTU 2015-16, Marks 05**

OR

Explain the basic concept of hardwired and software control unit with neat diagrams. **AKTU 2018-19, Marks 07**

Answer

Hardwired control : Refer Q. 3.23, Page 3–23B, Unit-3.
Micro-programmed control : Refer Q. 3.24, Page 3–24B, Unit-3.

Comparison between hardwired control and micro-programmed control :

S. No.	Characteristics	Hardwired control	Micro-programmed control
1.	Speed	Fast	Slow
2.	Implementation	Hardware	Software
3.	Flexibility	Not flexible	Flexible
4.	Ability to handle large/ complex instruction set	Somewhat difficult	Easier
5.	Ability to support operating system and diagnostic features	Very difficult	Easy
6.	Design process	Difficult for more operation	Easy
7.	Memory	Not used	Control memory used (RAM or ROM)
8.	Chip area efficiency	Uses less area	Uses more area
9.	Used in	RISC processor	CISC processor
10.	Output generation	On the basis of input signal	On the basis of control line.

Advantage of hardwired control unit :

1. Speed is high.

Disadvantages of hardwired control unit :

1. Expensive to implement.
2. More error prone.
3. Contain complex logic.

Advantages of micro-programmed control unit :

1. Cheaper to implement.
2. Less error prone.
3. Contain very simple piece of logic.

Disadvantage of micro-programmed control unit :

1. Speed is slow.

Que 3.26. What is micro-programmed control unit ? Give the basic structure of micro-programmed control unit. Also discuss the micro-instruction format and the control unit organization for a typical micro-programmed controllers using suitable diagram.

AKTU 2017-18, Marks 07

Answer

Micro-programmed control unit and its structure : Refer Q. 3.24, Page 3–24B, Unit-3.

Micro-instruction formats : The micro-instruction format for the control memory is shown in the Fig. 3.26.1.

F1, F2, F3 : Micro-operation fields
CD : Condition for branching
BR : Branch field
AD : Address field

Fig. 3.26.1. Micro-instruction format.

The 20 bits of the micro-instruction are divided into four functional parts as follows :

1. The three fields F1, F2, and F3 specify micro-operations for the computer. The micro-operations subdivided into three fields of three bits each. The three bits in each field are encoded to specify seven distinct micro-operations. So, this gives a total of 21 micro-operations.
2. The CD field selects status bit conditions.
3. The BR field specifies the type of branch to use.
4. The AD field contains a branch address. The address field is seven bits wide since the control memory has 128 = 27 words.

Organization of micro-programmed control unit :

1. The general configuration of a micro-programmed control unit is demonstrated in the block diagram of Fig. 3.26.2.
2. The control memory is assumed to be a ROM, within which all control information is permanently stored.

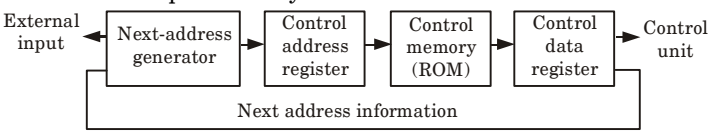


Fig. 3.26.2. Micro-programmed control organization.

3. The control memory address register specifies the address of the micro-instruction, and the control data register holds the micro-instruction read from memory.
4. The micro-instruction contains a control word that specifies one or more micro-operations for the data processor. Once these operations are executed, the control must determine the next address.
5. The location of the next micro-instruction may be the one next in sequence, or it may be located somewhere else in the control memory.
6. While the micro-operations are being executed, the next address is computed in the next address generator circuit and then transferred into the control address register to read the next micro-instruction.

7. Thus a micro-instruction contains bits for initiating micro-operations in the data processor part and bits that determine the address sequence for the control memory.
8. The next address generator is sometimes called a micro-program sequencer, as it determines the address sequence that is read from control memory.
9. Typical functions of a micro-program sequencer are incrementing the control address register by one, loading into the control address register an address from control memory, transferring an external address, or loading an initial address to start the control operations.
10. The control data register holds the present micro-instruction while the next address is computed and read from memory.

Que 3.27. Explain micro-program sequencer with block diagram. Compare horizontal and vertical organization.

OR

Write a short note on micro-program sequencer for control memory.

AKTU 2014-15, Marks 05

OR

Explain micro-program sequencer for a control memory using a suitable block diagram. **AKTU 2016-17, Marks 10**

OR

What is a micro-program sequencer ? With block diagram, explain the working of micro-program sequencer.

AKTU 2018-19, Marks 07

Answer

1. Micro-program sequencer is a general purpose building block for micro-programmed control unit.
2. The basic components of a micro-programmed control unit are the control memory and the circuit that selects the next address.
3. The address selection part is called micro-program sequencer.
4. The main purpose of micro-program sequencer is to present an address to the control memory so that micro-instruction may be read and executed.
5. The next address logic of the sequencer determines the specific address source to be loaded into the control address register.
6. The choice of the address source is guided by the next address information bits that sequencer receives from the present micro-instruction.
7. All the instructions are loaded in the control memory.

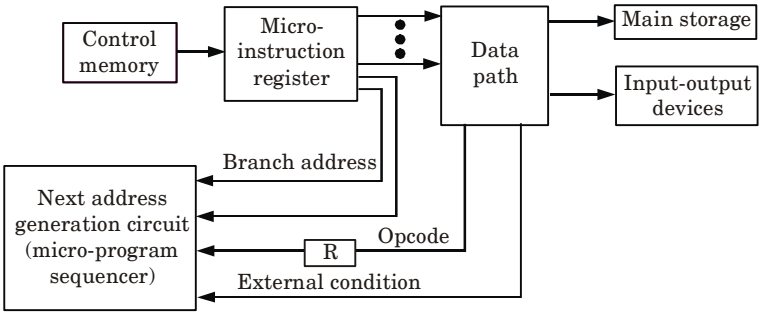


Fig. 3.27.1. Block diagram of micro-programmed control with micro-program sequencer.

8. The present micro-instruction is placed in micro-instruction register for execution.

S. No.	Horizontal Organization	Vertical Organization
1.	Long format.	Short format.
2.	Ability to express a high degree of parallelism.	Limited ability to express parallel micro-operations.
3.	Little encoding of control information.	Considerable encoding of the control information.
4.	Useful when higher operating speed is desired.	Slower operating speed.

Que 3.28. Describe micro-program sequencing in detail.

Answer

1. Micro-program sequencing is a process of controlling the generation of next address.
2. Micro-program sequencing is done with the help of micro-program sequencer.
3. A micro-program sequencer attached to a control memory inputs certain bits of the micro-instruction, from which it determines the next address for control memory.
4. A typical sequencer provides the following address-sequencing capabilities :
   a. Increment the present address for control memory.
   b. Branches to an address as specified by the address field of the micro-instruction.

c. Branches to a given address if a specified status bit is equal to 1.
d. Transfer control to a new address as specified by an external source (Instruction Register).
e. Has a facility for subroutine calls and returns.

5. Depending on the current micro-instruction condition flags, and the contents of the instruction register, a control memory address must be generated for the next micro-instruction.

PART-7

Concept of Horizontal and Vertical Micro-programming.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 3.29. Explain the concept of vertical and horizontal multi-programming.

Answer

Vertical micro-programming :

1. In the case of vertical micro-programming, each line of the micro-program represents a micro-instruction which specifies one or more micro-operations.
2. One micro-instruction gets executed during each step of the control sequence. One can use a straight binary code to specify each micro-operation.
3. Rather than provide for only one micro-operation per step and use a single decoder for the entire micro-instruction field, it is in fact possible to partition this field into a number of mutually exclusive subfields which can be independently decoded in separate decoder.
4. This approach yields a more cost-effective design. If the system design needs in all a total of N different micro-operations one will have to provide for $\log_2 N$ bit to specify a micro-operation.
5. If only one micro-operation per micro-instruction is allowed, then one would require $\log_2 N$ bit per micro-instruction.

Horizontal micro-programming :

1. In horizontal micro-programming, one associates each bit of the micro-instruction with a specific micro-operation (bit I to represent micro-operation I).
2. A specific micro-operation is executed during a micro-instruction step only if the corresponding bit is one.

3. For instance, micro-operation I is executed if the I^{th} bit is one; it is not executed otherwise.
4. A micro-code decoder will no longer be required in this case because no coding is involved.
5. This scheme will require a total of N bits per micro-instruction (to accommodate N different micro-operations).
6. The advantage with horizontal micro-programming is that several micro-operations can be executed during each micro-instruction step.
7. In fact, this scheme permits one to execute all the micro-operations provided in the computer in a single micro-instruction.

Que 3.30. Briefly define the following terms :

i. Micro-operation
ii. Micro-instruction
iii. Micro-program
iv. Micro-code
v. Control memory

AKTU 2015-16, Marks 10

Answer

i. Micro-operation :

1. Micro-operation is a set of operations that the processor unit has to perform to execute the major phases of instruction cycle.
2. The instruction cycle has three major phases of fetch, decode and execute.
3. The primary function of a CPU is to execute sequence of instructions which is in accordance with the instruction cycle.

ii. Micro-instructions :

1. Micro-instructions are the individual control words in the micro routine.
2. This contains the control signals for sequencing information.

iii. Micro-program : Micro-program is a micro-code in a particular processor implementation. Writing micro-code is often called micro-programming.

iv. Micro-code :

1. Micro-code is a layer of hardware-level instructions and/or data structures involved in the implementation of higher level machine code instructions in many computers and other processors.
2. It helps to separate the machine instructions from the underlying electronics so that instructions can be designed and altered more freely.

v. Control memory :

1. Control memory is a Random Access Memory (RAM) consisting of addressable storage registers.

2. It is used as a temporary storage for data.
3. Access to control memory data requires less time than to main memory, this speeds up CPU operation.

Que 3.31. **Write an assembly level program for the following pseudo code :**

SUM = 0
SUM = SUM + A + B
DIF = DIF-C
SUM = SUM + DIF

AKTU 2016-17, Marks 10

Answer

```
CLA                 /SUM = 0
STA SUM
LDA SUM             /Load current sum
ADD A               /Add A to SUM
ADD B               /Add B to SUM
STA SUM             /Save SUM
LDA C               /Load C to AC
CMA                 /Create 2's complement
INC
ADD DIF             /Subtract C from DIF
STA DIF             /Save DIF
ADD SUM             /Add SUM to DIF
STA SUM             /Save SUM
HLT                 /Halt
```

Que 3.32. **Explain 4-bit incrementer with a necessary diagram.**

AKTU 2016-17, Marks 15

Answer

1. The diagram of a 4-bit combinational circuit incrementer is shown in Fig. 3.32.1.
2. One of the inputs to the least significant Half Adder (HA) is connected to logic-1 and the other input is connected to the least significant bit of the number to be incremented.
3. The output carry from one half-adder is connected to one of the inputs of the next-higher-order half-adder.
4. The circuit receives the four bits from A_0 through A_3 adds one to it, and generates the incremented output in S_0 through S_3.
5. The output carry C_4 will be 1 only after incrementing binary 1111. This also causes outputs S_0 through S_3 to go to 0.
6. This micro-operation is easily implemented with a binary counter.

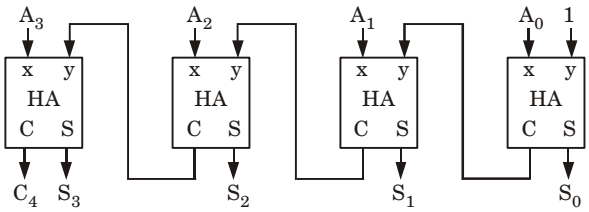


Fig. 3.32.1. 4-bit binary incrementer.

7. Every time the count enable is active, the clock pulse transition increments the content of the register by one.
8. There may be occasions when the increment micro-operation must be done with a combinational circuit independent of a particular register.
9. This can be accomplished by means of half adders connected in cascade.

Que 3.33. Write a program loop using a pointer and a counter to clear the contents of hex locations 500 to 5FF with 0.

AKTU 2016-17, Marks 15

Answer

LDA NBR	/	Initialize counter
CMA	/	2's complement of NBR INC
STA CTR	/	save -NBR to counter
LDA ADR	/	Save start address
STA PTR	/	Initialize pointer PTR
LOP, CLA	/	Clear AC
STA PTR I	/	Reset memory word
ISZ PTR	/	Increment pointer
ISZ CTR	/	increment counter
BUN LOP	/	Branch to LOP (CTR < 0)
HLT	/	Halt when CTR = 0
NBR, HEX FF	/	NBR of cleared words
CTR, –	/	Counter
ADR, HEX 500	/	Start address
PTR, –	/	Pointer

Que 3.34. Demonstrate the process of second pass of assembler using a suitable diagram.

AKTU 2016-17, Marks 15

Answer

1. Machine instructions are translated during the second pass by means of table-lookup procedures.
2. A table-lookup procedure is a search of table entries to determine whether a specific item matches one of the items stored in the table.
3. The assembler uses four tables.
4. Any symbol that is encountered in the program must be available as an entry in one of these tables; otherwise, the symbol cannot be interpreted.

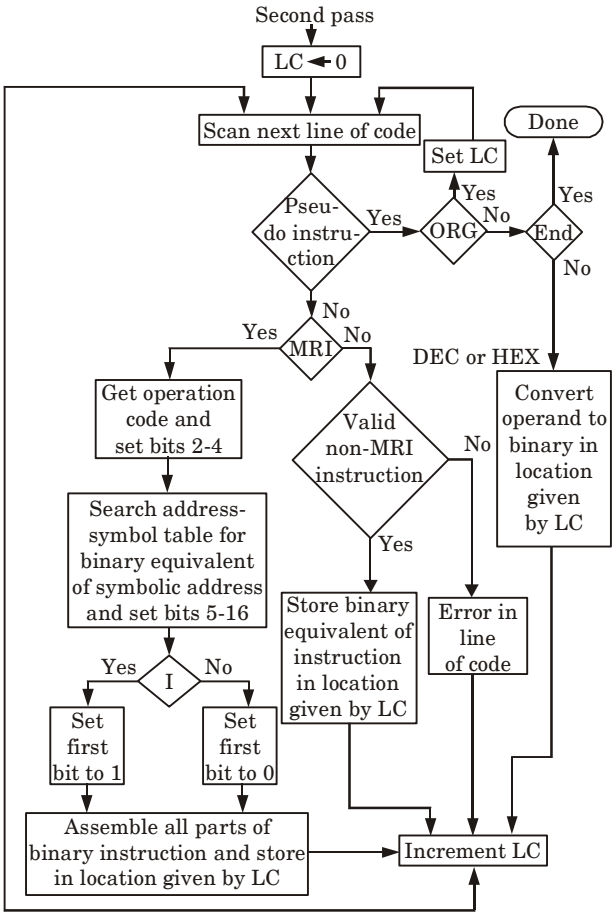


Fig. 3.34.1.

5. We assign the following names to the four tables :
   a. Pseudo instruction table
   b. MRI table
   c. Non-MRI table
   d. Address symbol table
6. The entries of the pseudo instruction table are the four symbols ORG, END, DEC, and HEX.
7. Each entry refers the assembler to a subroutine that processes the pseudo instruction when encountered in the program.
8. The MRI table contains the seven symbols of the memory-reference instructions and their 3-bit operation code equivalent.
9. The non-MRI table contains the symbols for the 18 register-reference and input-output instructions and their 16-bit binary code equivalent.
10. The assembler searches these tables to find the symbol that it is currently processing in order to determine its binary value.

VERY IMPORTANT QUESTIONS

Following questions are very important. These questions may be asked in your SESSIONALS as well as UNIVERSITY EXAMINATION.

Q. 1. What is an instruction in the context of computer organization ? Explain the purpose of the various elements of an instruction with the help of a sample instruction format.

Ans. Refer Q. 3.1.

Q. 2. Evaluate the arithmetic statement $X = (A + B)*(C + D)$ using a general register computer with three address, two address and one address instruction format a program to evaluate the expression.

Ans. Refer Q. 3.3.

Q. 3. Explain all the phases of instruction cycle.

Ans. Refer Q. 3.4.

Q. 4. Write the steps in fetching a word from memory. Differentiate between a branch instruction and call subroutine instruction.

Ans. Refer Q. 3.5.

Q. 5. In an instruction format, there are 16 bits in an instruction word. Bit 0 to 11 convey the address of the memory location for memory related instructions. For non memory instructions these bits convey various register or I/O operations. Bits 12 to 14 show the various basic memory operations such as ADD, AND, LDA etc. Bit 15 shows if the memory is accessed directly or indirectly. For such an instruction format draw block diagram of the control unit of a computer and briefly explain how an instruction will be decoded and executed, by this control unit.

Ans. Refer Q. 3.7.

Q. 6. List and explain different types of shift micro-operation.

Ans. Refer Q. 3.12.

Q. 7. What are the different categories of micro-operations that may be carried out by CPU ? Explain each category of micro-operations giving one example for each.

Ans. Refer Q. 3.13.

Q. 8. What is CISC ? Explain its characteristics.

Ans. Refer Q. 2.15.

Q. 9. What is RISC ? Explain its various characteristics.

Ans. Refer Q. 3.17.

Q. 10. Give the detailed comparison between RISC and CISC.

Ans. Refer Q. 3.18.

Q. 11. Write a short note on pipelining.

Ans. Refer Q. 3.19.

Q. 12. Explain the basic concept of hardwired and software control unit with neat diagrams.

Ans. Refer Q. 3.25.

Q. 13. What is micro-programmed control unit ? Give the basic structure of micro-programmed control unit. Also discuss the micro-instruction format and the control unit organization for a typical micro-programmed controllers using suitable diagram.

Ans. Refer Q. 3.26.

Q. 14. Write a short note on micro-program sequencer for control memory.

Ans. Refer Q. 3.27.

Q. 15. Briefly define the following terms :

i. **Micro-operation**
ii. **Micro-instruction**
iii. **Micro-program**
iv. **Micro-code**
v. **Control memory**

Ans. Refer Q. 3.30.

Q. 16. Write an assembly level program for the following pseudo code :

```
SUM = 0
SUM = SUM + A + B
DIF = DIF-C
SUM = SUM + DIF
```

Ans. Refer Q. 3.31.

Q. 17. Explain 4-bit incrementer with a necessary diagram.

Ans. Refer Q. 3.32.

Q. 18. Write a program loop using a pointer and a counter to clear the contents of hex locations 500 to 5FF with 0.

Ans. Refer Q. 3.33.

Q. 19. Demonstrate the process of second pass of assembler using a suitable diagram.

Ans. Refer Q. 3.34.

4
UNIT

Memory

CONTENTS

PART-1

Memory : Basic Concept and Hierarchy.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 4.1. Explain concept of memory. Describe memory hierarchy.

Answer

1. The memory of a computer holds (stores) program instructions (what to do), data (information), operand (manipulated or operated upon data), and calculations (ALU results).
2. The CPU controls the information stored in memory.
3. Information is fetched, manipulated (under program control) and written (or written back) into memory for immediate or later use.
4. The internal memory of a computer is also referred to as main memory, global memory, main storage.
5. The secondary or auxiliary memory (also called mass storage) is provided by various peripheral devices.

Memory hierarchy :

1. The memory hierarchy system consists of all storage devices employed in a computer system from the slow but high capacity auxiliary memory to a relatively faster main memory, to as even smaller and faster cache memory accessible to the high speed processing logic.
2. Fig. 4.1.1 shows the typical memory hierarchy. All the memory devices are shown in the Fig. 4.1.1.

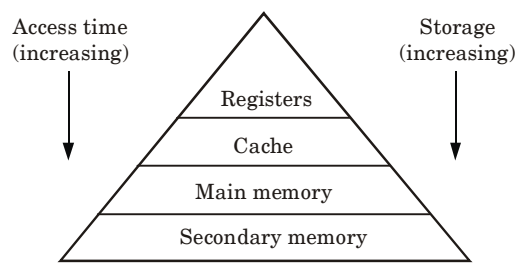


Fig. 4.1.1. Memory hierarchy.

Memory hierarchy is layered into these layers :

1. **Registers :** Internal register in a CPU is used for holding variables and temporary results. Internal registers have a very small storage; however they can be accessed instantly.
2. **Cache memory :** Cache is used by the CPU for holding data in the memory which is being accessed over and over again. It acts as a buffer between the CPU and the main memory.
3. **Main memory :**
   a. Main memory is large and fairly fast external memory which stores active data and programs.
   b. Storage location in memory is directly addressed by the CPU's load and store instructions.
   c. Access time is larger because of its large capacity and as it is physically separated from the CPU.
4. **Secondary/auxiliary memory :**
   a. Secondary/auxiliary memory are giant in capacity but comparatively very slow than all the other types of memory.
   b. It stores large data files, programs and files that will not be required continuously by the CPU.
   c. It also acts as an overflow memory when the capacity of the main memory exceeded.

Que 4.2. **Why is memory system of a computer organized as a hierarchy ? Discuss the basic elements of a memory hierarchy.**

Answer

1. Memory system of a computer is organized as a hierarchy to optimize the use of different types of memories and to achieve greater efficiency and economy.
2. Memory is organized in a hierarchy with the highest performance and in general the most expensive devices at the top, and with progressively lower performance and less costly devices in succeeding layers.

Elements of memory hierarchy : Refer Q. 4.1, Page 4–2B, Unit-4.

PART-2

Semiconductor RAM Memories.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 4.3. **Explain semiconductor RAM. Enlist the types of semiconductor memory.**

OR

Explain dynamic RAM and static RAM.

Answer

Semiconductor RAM :

1. Semiconductor Random Access Memory (RAM) is a form of computer data storage which stores frequently used program instructions to increase the general speed of a system.
2. A random access memory device allows data items to be read or written in almost the same amount of time irrespective of the physical location of data inside the memory.

Types of semiconductor RAM are :

1. DRAM :

a. Dynamic RAM is a form of random access memory.

b. DRAM uses a capacitor to store each bit of data, and the level of charge on each capacitor determines whether that bit is a logical 1 or 0.

c. However these capacitors do not hold their charge indefinitely, and therefore the data needs to be refreshed periodically.

d. DRAM is the form of semiconductor memory that is often used in equipment including personal computers and workstations where it forms the main RAM for the computer.

2. SRAM :

a. Static Random Access Memory is a semiconductor memory in which, the data does not need to be refreshed dynamically.

b. SRAM is volatile in nature.

c. It consumes more power, is less dense and more expensive than DRAM.

d. SRAM is used for cache memory.

Que 4.4. **Give the structure of commercial 8M × 8 bit DRAM chip.**

AKTU 2017-18, Marks 07

Answer

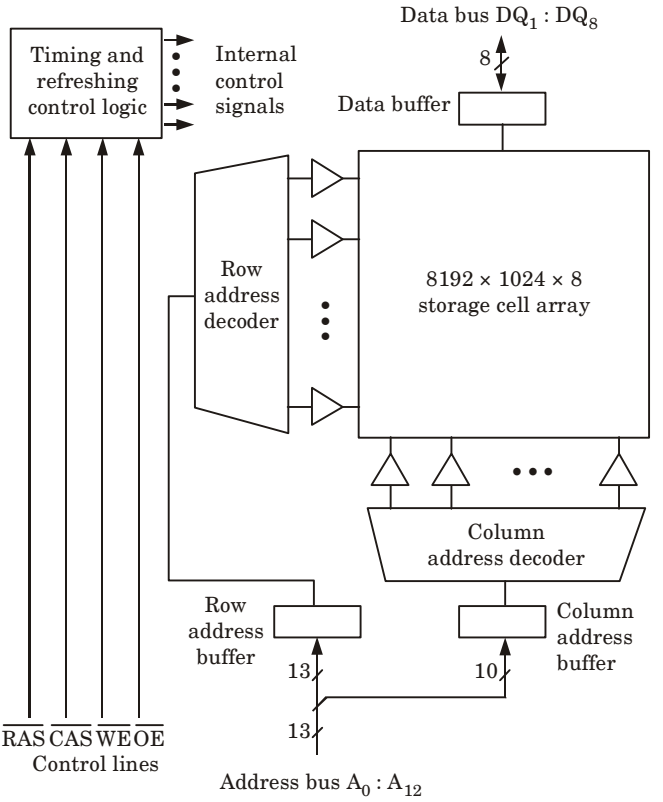


Fig. 4.4.1. Structure of a commercial 8M × 8-bit DRAM chip.

PART-3

2D & 2½ D Memory Organization.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 4.5. Explain 2D, 2 $\frac{1}{2}$ D memory organization.

OR

Write short note on organization of 2D and 2.5D memory organization. **AKTU 2014-15, Marks 05**

Answer

2D organization :

1. The cells are organized in the form of a two-dimensional array with rows and columns.
2. Each row refers to word line. For 4-bit per word memory, 4 cells are interconnected to a word line. Each column in the array refers to a bit line.

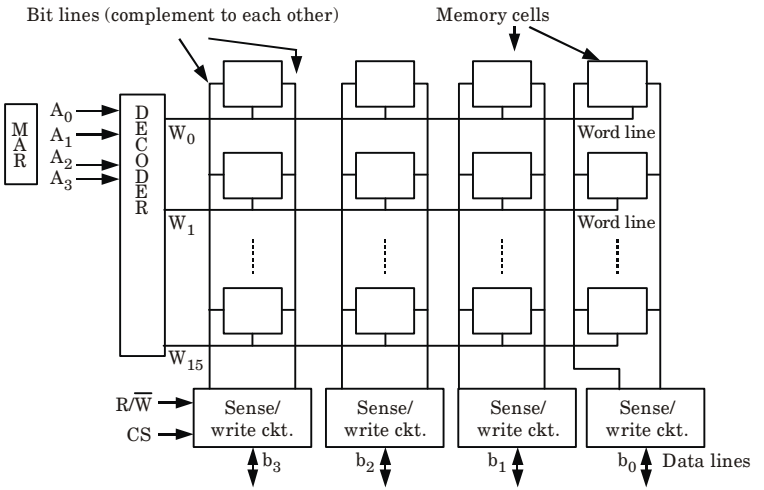


Fig. 4.5.1. 2D organization of a memory chip of size 16 × 4.

3. The Memory Address Register (MAR) holds the address of the location where read/write operation is executed. In Fig. 4.5.1, MAR has 4-bit lines.
4. The content of MAR is decoded by an address decoder on the chip to activate each word line.
5. The cells in each column are connected to a sense/write circuit by two bit lines. Two bit lines are complement to each other.
6. The sense/write circuits are activated by the Chip Select (CS) lines. The sense/write circuits are connected to the data lines of the chip.
7. During a read operation, these circuits sense or read the information stored in the cells selected by a word line and transmit this information to the data lines.

8. During a write operation, the sense/write circuits receive or write input information from the data lines and store it in the selected cells.

2.5D organization :

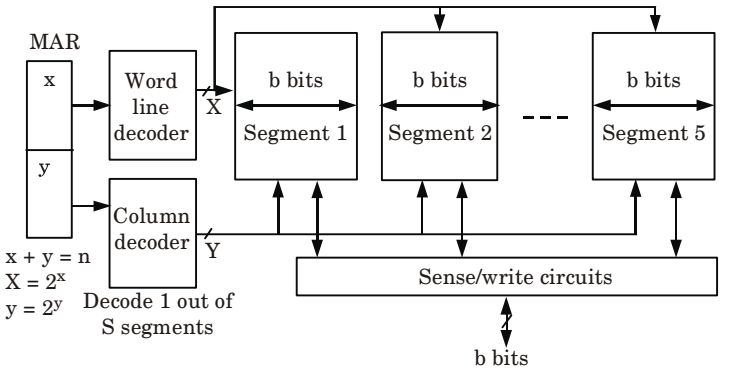


Fig. 4.5.2. 2.5D organization.

1. In 2.5D organization there exists a segment.
2. The content of MAR is divided into two parts–x and y number of bits.
3. The number of segments S is equal to 2^y.
4. $X = 2^x$ drive lines are fed into the cell array and y number of bits decode one bit line out of S lines fed into a segment of the array. In total, there are Sb number of bit lines for a b bit per word memory.
5. Thus for any given address in the MAR, the column decoder decodes b out of Sb bit lines by using the y bits of the MAR while a particular word line is activated by using the x bits.
6. Thus only the b numbers of bits in the array are accessed by enabling the word line and b number of bit lines simultaneously.
7. Though 2.5D organized memory may need lesser chip decoding logic, it suffers from one drawback. With high density chips, a simple failure, such as external pin connection opening or a failure on one bit can render the entire chip inoperative.

PART-4

ROM Memories.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 4.6. What is ROM ? Explain the types of ROM.

OR

Explain the semiconductor based ROM memories.

Answer

ROM :

1. The Read Only Memory is a type of semiconductor memory that is designed to hold data that is either permanent or will not change frequently. It is also known as non-volatile memory.

Types of ROM :

i. Programmable read only memory (PROM) :

a. It is one-time programmable ROM (OTP) and can be written to or programmed via a special device called a PROM programmer.

b. Typically, this device uses high voltages to permanently destroy or create internal links (fuses or antifuses) within the chip.

c. Consequently, a PROM can only be programmed once.

ii. Erasable programmable read only memory (EPROM) :

a. It can be erased by exposure to strong ultraviolet light (typically for 10 minutes or longer), then rewritten with a process that again requires application of higher than usual voltage.

b. Repeated exposure to UV light will eventually wear out an EPROM, but the endurance of most EPROM chips exceeds 1000 cycles of erasing and reprogramming.

c. EPROM chip packages can often be identified by the prominent quartz "window" which allows UV light to enter.

d. After programming, the window is typically covered with a label to prevent accidental erasure.

e. Some EPROM chips are factory-erased before they are packaged, and include no window, these are effectively PROM.

iii. Electrically erasable programmable read only memory (EEPROM) :

a. It is based on a similar semiconductor structure to EPROM, but allows its entire contents (or selected banks) to be electrically erased, then rewritten electrically, so that they need not be removed from the computer (or camera, MP3 player, etc.).

b. Writing or flashing an EEPROM is much slower (milliseconds per bit) than reading from a ROM or writing to a RAM (nanoseconds in both cases).

iv. Mask programming :

a. It is done by the company during fabrication process of the unit.

b. The procedure for fabricating a ROM requires that the customer fills out the truth table he wishes the ROM to satisfy.

Que 4.7. How main memory is useful in computer system ? Explain the memory address map of RAM and ROM.

AKTU 2016-17, Marks 15

Answer

Main memory is useful in computer system :

1. The main memory occupies a central position in a computer system.
2. It is able to communicate directly with the CPU and with auxiliary memory devices through an I/O processor.
3. It is relatively large and fast.

Memory address map of RAM and ROM :

1. Memory address map is a pictorial representation of assigned address space for each chip in the system.
2. To demonstrate with a particular example, assume that a computer system needs 512 bytes of RAM and 512 bytes of ROM.
3. The RAM and ROM chips to be used are specified in Fig. 4.7.1. The memory address map for this configuration is shown in Table 1.
4. The component column specifies whether a RAM or a ROM chip is used. The hexadecimal address column assigns a range of hexadecimal equivalent addresses for each chip. The address bus lines are listed in the third column.
5. The selection between RAM and ROM is achieved through bus line 10. The RAMs are selected when the bit in this line is 0, and the ROM when the bit is 1.

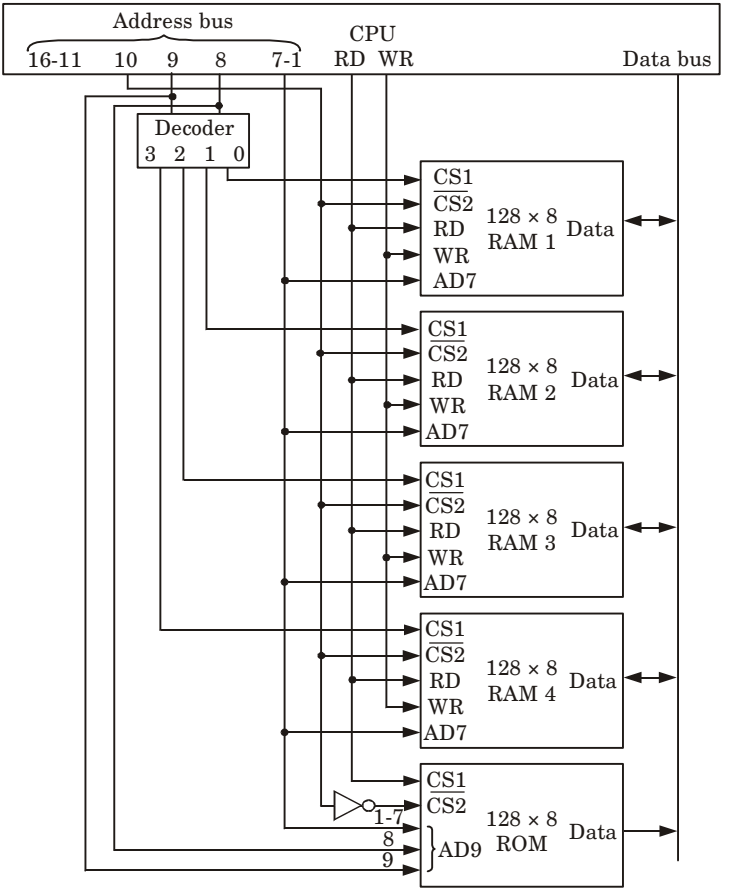


Fig. 4.7.1.

Table 4.7.1 : Memory address map.

Component	Hexadecimal Address	Address bus									
		10	9	8	7	6	5	4	3	2	1
RAM1	0000-007F	0	0	0	×	×	×	×	×	×	×
RAM2	0080-00FF	0	0	1	×	×	×	×	×	×	×
RAM3	0100-017F	0	1	0	×	×	×	×	×	×	×
RAM4	0180-01FF	0	1	1	×	×	×	×	×	×	×
ROM	0200-03FF	1	×	×	×	×	×	×	×	×	×

6. The × under the address bus lines designate those lines that must be connected to the address inputs in each chip.
7. The RAM chips have 128 bytes and need seven address lines.
8. The ROM chip has 512 bytes and need 9 address lines.
9. It is now necessary to distinguish between four RAM chips by assigning to each a different address.

Que 4.8. **A computer uses RAM chips of 1024*1 capacity.**

i. How many chips are needed and how should their address lines be connected to provide a memory capacity of 1024*8 ?

ii. How many chips are needed to provide a memory capacity of 16 KB ? Explain in words how the chips are to be connected to the address bus. **AKTU 2015-16, Marks 10**

Answer

i. Available size of RAM chips = 1024×1

Required memory capacity = 1024 bytes

$= 1024 \times 8$

Number of chips required = $\frac{1024 \times 8}{1024 \times 1} = 8$ chips

So, 8 chips are needed with address line connected in parallel.

ii. To provide a memory capacity of 16K bytes, chips required are $16 \times 8 = 128$ chips

Number of address line for 16 K = 14 ($16\ K = 2^{14}$)

So, 14 lines to specify chip address.

Que 4.9. **A ROM chip of 1024*8 has four select inputs and operates from a 5 volt power supply. How many pins are needed for the IC package ? Draw a block diagram and label all input and output terminals in the ROM.** **AKTU 2015-16, Marks 10**

Answer

Size of ROM Chip = 1024×8

Number of input = 10 pin [$2^{10} = 1024$]

Number of output = 8 pin

Number of chip select = 4 pin

Power = 2 pin

Total 24 pins are required.

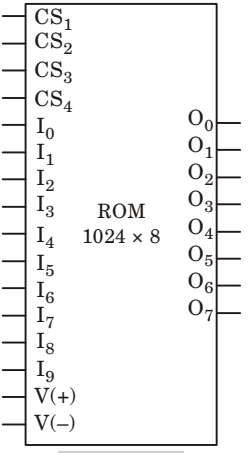


Fig. 4.9.1.

Que 4.10. **A computer uses a memory unit with 256 K words of 32 bits each. A binary instruction code is stored in one word of memory. The instruction has four parts : an indirect bit, an operation code, a register code part to specific one of 64 register and an address part.**

i. How many bits are there in the operation code, the register code part and the address part ?

ii. Draw the instruction word format and indicate the number of bits in each part.

iii. How many bits are there in the data and address inputs of the memory ?

AKTU 2018-19, Marks 07

Answer

a. **Address :** $2^8 * 2^{10} = 2^{18} = 18$ bits
Register : 64 registers = $2^6 = 6$ bits
OP code : (Total bit – Indirect bit – Address bit – Register bit) = (32 – 1 – 18 – 6) bits = 7 bits

b.

I	OP	Register	Address

31 30 23 17 0

c. **Number of bit in address inputs :** 18
Number of bit in data inputs : 32

PART-5

Cache Memories : Concept and Design Issues and Performance.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 4.11. Write short note on cache memory.

Answer

1. Cache memory is a small-sized type of volatile computer memory that provides high-speed data access to a processor and stores frequently used computer programs, applications and data.
2. It stores and retains data only until a computer is powered up.
3. Cache memory provides faster data storage and access by storing an instance of programs and data routinely accessed by the processor.
4. Thus, when a processor requests data that already has an instance in the cache memory, it does not need to go to the main memory or the hard disk to fetch the data.
5. Cache memory can be primary or secondary cache memory, where primary cache memory is directly integrated or closest to the processor.
6. In addition to hardware-based cache, cache memory also can be a disk cache, where a reserved portion on a disk stores and provide access to frequently accessed data/applications from the disk.

Que 4.12. Discuss the design issues in cache design.

Answer

The primary elements having strong influence on cache design are :

i. Cache size : There are two important factor in deciding cache size at the time of its design.

a. Size should be small enough so that its cost is very close to the main memory.

b. Size should be large enough so that most of the memory reference should be available in the cache so that average access time will be close to cache alone.

ii. Block size :

a. Block size of cache is an important factor in cache performance.

b. Larger block size could store the desired word as well as some adjacent word also. As a result hit ratio will increase automatically.

c. A small block size result in frequent replacement of data shortly after it is fetched increasing the overhead in cache operation.

d. As the block size increases from smaller to larger, the hit ratio will increase first and on increasing the block size further the hit ratio began to decrease.

iii. **Associativity :** It is a basic tradeoff between parallel searching versus constraints on which addresses can be stored.

iv. **Write strategy :** Its main concern is related to when do we write cache contents to main memory.

Que 4.13. **How the performance of cache memory is measured ?**

Answer

1. The performance of cache memory is frequently measured in terms of a quantity called hit ratio.
2. When the CPU refers to memory and finds the word in cache, it is said to produce a hit.
3. If the word is not found in cache, it is in main memory and it counts as a miss.
4. The ratio of the number of hits divided by the total CPU references to memory (hits plus misses) is the hit ratio.
5. The hit ratio is best measured experimentally by running representative programs in the computer and measuring the number of hits and misses during a given interval of time.

PART-6

Address Mapping and Replacement.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 4.14. **What is meant by cache mapping ? What are different types of mapping ? Discuss different mapping techniques with examples.**

OR

Discuss the various types of address mapping used in cache memory. **AKTU 2017-18, Marks 07**

Answer

Cache mapping :

1. Cache mapping is the method by which the contents of main memory are brought into the cache and referenced by the CPU. The mapping method used directly affects the performance of the entire computer system.
2. Mapping is a process to discuss possible methods for specifying where memory blocks are placed in the cache. Mapping function dictates how the cache is organized.

Types of mapping :

1. Direct mapping :

a. The direct mapping technique is simple and inexpensive to implement.

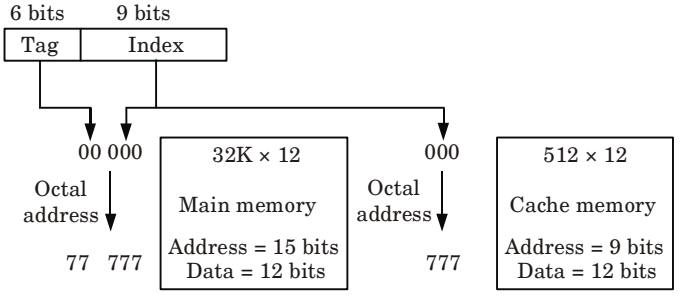


Fig. 4.14.1.

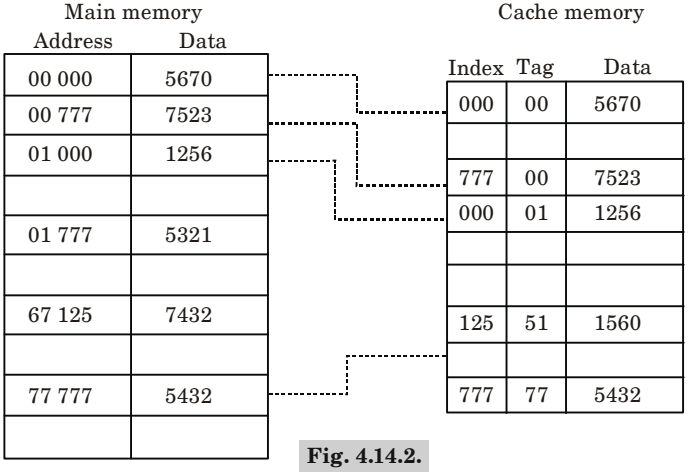


Fig. 4.14.2.

b. When the CPU wants to access data from memory, it places an address. The index field of CPU address is used to access address.

c. The tag field of CPU address is compared with the associated tag in the word read from the cache.

d. If the tag-bits of CPU address are matched with the tag-bits of cache, then there is a hit and the required data word is read from cache.

e. If there is no match, then there is a miss and the required data word is stored in main memory. It is then transferred from main memory to cache memory with the new tag.

2. Associative mapping :

a. An associative mapping uses an associative memory.

b. This memory is being accessed using its contents.

c. Each line of cache memory will accommodate the address (main memory) and the contents of that address from the main memory.

d. That is why this memory is also called Content Addressable Memory (CAM). It allows each block of main memory to be stored in the cache.

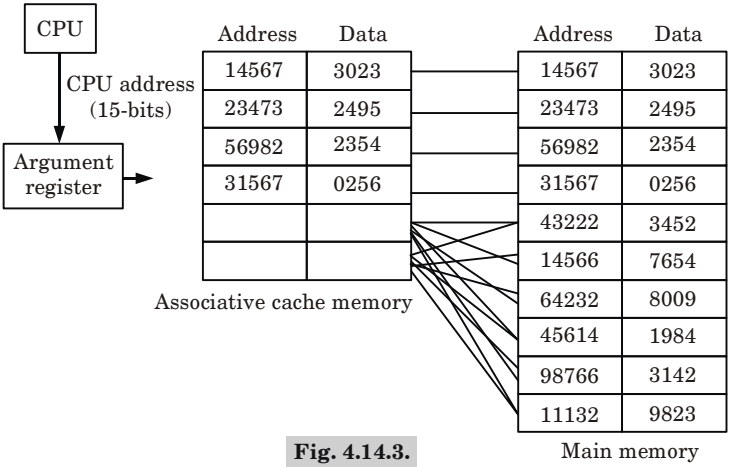


Fig. 4.14.3.

3. Set associative mapping :

a. In set associative mapping, each cache location can have more than one pair of tag + data items.

b. It combines the best of the direct mapping cache and the more flexible mapping of the fully associative cache.

c. That is more than one pair of tag and data are residing at the same location of cache memory. If one cache location is holding two pair of tag + data items, that is called 2-way set associative mapping.

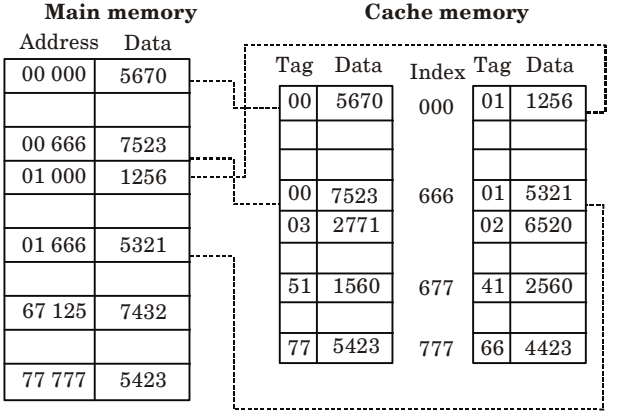


Fig. 4.14.4.

Que 4.15. What do you mean by cache memory ? How does it affect the performance of the computer system ? An eight-way set associative cache is used in computer in which the real memory size 2^{32} bytes. The line size is 16 bytes, and there are 2^{10} lines per set. Calculate the cache size and tag length.

Answer

Cache memory : Refer Q. 4.11, Page 4–13B, Unit-4.

Cache affect the performance as follows :

1. The CPU to work at its maximum efficiency, the data transfer from the other hardware must be as fast as its speed. The purpose of a cache is to ensure this smooth and fast transition of data transfer from the hardware to the CPU.
2. As CPU speed increased to the point where the RAM is no longer able to catch up, the transferring of information again become a serious problem. To solve this issue, a cache, which was effectively a small and extremely fast memory, was added to the processor to store immediate instruction from the RAM. Since the cache runs at the same speed of the CPU, it can rapidly provide information to the CPU at the shortest time without any delay.

Numerical :

Given, Main memory size = 2^{32} bytes = 2^{35} bits

Line size = Block size = 16 bytes = 2^7 bits

Lines/set = 2^{10}

Since, 8-way set associative so, cache have 8 set

No. of set = 8

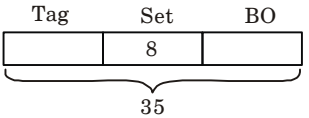
Total no. of lines = No. of sets × No. of lines/set

$= 8 \times 2^{10} = 2^{13}$

Cache size = No. of sets × No. of lines /set × size of line

$= 8 \times 2^{13} \times 2^{7}$ bits $= 2^{23}$ bits

Cache size = 8 MBytes



Block offset = $\log_2$ (line size) = $\log_2 (2^7) = 7$ bits

No. of set = 8

Physical address (PA) = 35

Tag size = PA – (Block offset + No. of set)

$= 35 - (7 + 8)$

Tag size = 20 bit

Que 4.16. **Consider a cache uses a direct mapping scheme. The size of main memory is 4 K bytes and word size of cache is 2 bytes. The size of cache memory is 128 bytes. Find the following :**

i. The size of main memory address (assume each byte of main memory has an address)

ii. Address of cache block

iii. How many memory location address will be translated to cache address/block/location ?

iv. How can it be determined if the content of specified main memory address in cache ? **AKTU 2014-15, Marks 10**

Answer

Given, Size of main memory = 4 K bytes

Size of cache memory = 128 bytes

Word size of cache = 2 bytes

i. Size of main memory address :

Since, size of main memory = 4 K bytes = 2^{12} bytes

So, the size of main memory address = 12

ii. Address of cache block :

Since, size of cache memory = 128 bytes

Block size = word size of cache = 2 bytes

$\therefore$ Number of lines = $\frac{128}{2} = 64$

So, address of cache block is from 0 to 63

iii. Since, size of main memory = 4 K bytes = 4096 bytes

So, number of block in main memory = 4096

and number of block in cache = 64

and each block of cache = 2 bytes

So, $\left(\frac{4096}{(64 \times 2)}\right)$ = 32 memory location address will be translated to cache address/block/location.

iv. We can determine the content of specified main memory address in cache using this :

$$i \bmod 2^k$$

where, i = particular memory address

k = line number

Que 4.17. A two way set associative cache memory uses blocks of 4 words. The cache can accommodate a total of 2048 words from memory. The main memory size is 128 K × 32.

i. Formulate all pertinent information required to construct the cache memory.

ii. What is the size of the cache memory ?

AKTU 2018-19, Marks 07

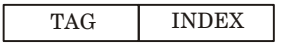
Answer

i. Main memory size = 128K × 32 = 2^{17}

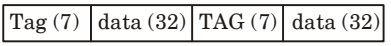
Cache size = 2048 words

Set size of 2 cache can accommodate = 2048/2 = 1024 words of cache

Block size = 4 words



ii.



Size of cache memory = 1024 × 2 (7 + 32) = 1024 × 78

Que 4.18. What is the distinction between spatial locality and temporal locality ?

AKTU 2015-16, Marks 7.5

Answer

Difference :

S. No.	Spatial locality	Temporal locality
1.	Spatial locality refers to the tendency of execution to involve a number of memory locations that are clustered.	Temporal locality refers to the tendency for a processor to access memory locations that have been used recently.
2.	The spatial locality means that instructions stored near-by to the recently executed instructions are also likely to be executed soon.	The temporal locality means that a recently executed instruction is likely to be executed again very soon.
3.	The spatial aspect suggests that instead of bringing just one item from the main memory to the cache, it is wise to bring several items that reside at adjacent addresses as well.	The temporal aspect of the locality reference suggests that whenever information of instruction and data is first needed, this information should be brought into cache where it will hopefully remain until it is needed again.

Que 4.19. Write short note on write through and write back policy of cache memory.

Answer

Writing policy of cache memory :

i. Write through :

a. This is simplest technique.

b. Using this technique, all write operations are made to main memory as well as to the cache, ensuring that main memory is always valid.

c. The main disadvantage of this technique is that it generates substantial memory traffic and may create a bottleneck.

ii. Write back :

a. It is another technique which minimizes memory writes.

b. With write back, updates are made only in the cache.

c. When an update occurs, an UPDATE bit associated with the slot is set. Then, when a block is replaced it is written back to main memory if and only if the UPDATE bit is set.

d. The problem with write back is that portions of main memory are invalid and hence accesses by I/O modules can be allowed only through the cache.

Que 4.20. Explain replacement algorithm in brief.

Answer

When a main memory block needs to be brought in while all the cache memory blocks are occupied, one of them has to be replaced. This is known as block replacement.

The replacement algorithms are given as follows :

1. Optimal replacement :

a. In this policy, replace the block which is no longer needed in the future.

b. If all blocks currently in cache memory will be used again, replace the one which will not be used in the future for the longest time.

c. The optimal replacement is obviously the best but is not realistic, simply because when a block will be needed in the future is usually not known ahead of time.

2. LRU (Least Recently Used) :

a. In this policy, replace the block in cache memory that has not been used for the longest time, *i.e.*, the least recently used (LRU) block.

b. The LRU is sub-optimal based on the temporal locality of reference, *i.e.*, memory items that are recently referenced are more likely to be referenced soon than those which have not been referenced for a longer time.

3. FIFO (First in First Out) :

a. In this policy, replace the block that has been in cache memory for the longest time.

b. FIFO is simplest replacement algorithm than LRU.

4. Random selection :

a. In this policy, replace a randomly selected block among all blocks currently in cache memory.

b. In random selection, select the block and discard it to make space when necessary.

PART-7

Auxiliary Memories : Magnetic Disk, Magnetic Tape and Optical Disks, Virtual Memory : Concept Implementation.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 4.21. Explain auxiliary memory. What are the commonly used auxiliary memory ?

Answer

1. Auxiliary memory is a higher capacity external memory.
2. It is non-volatile memory that is not accessible by the CPU, because it is not accessed via the input / output channels.

Types of auxiliary memory :

1. Magnetic disks :

a. A disk is a circular plate constructed of metal or of plastic coated with a magnetizable material.

b. Data are recorded on and later retrieved from the disk via a conducting coil, named the head.

c. During a read and write operation, the head is stationary while the plate rotates beneath it.

2. Magnetic tape :

a. A magnetic tape consists of the electrical, mechanical and electronic components to provide the parts and control mechanism for a magnetic tape unit.

b. The tape itself is a strip of plastic coated with a magnetic recording medium.

c. Bits are recorded as magnetic spots on the tape along several tracks.

d. Usually, seven or nine bits are recorded simultaneously to form a character together with a parity bit.

e. Read/write heads are mounted one in each track so that data can be recorded and read as sequence of characters.

3. Flash memory :

a. An electronic non-volatile computer storage device that can be electrically erased and reprogrammed, and works without any moving parts.

b. Examples of this are flash drives, memory cards and solid state drives.

4. Optical disk :

a. A storage medium from which data is read and written by lasers.

b. Optical disks can store much more data up to 6 gigabytes more than most portable magnetic media, such as floppies.

c. There are three basic types of optical disks: CD/DVD/BD-ROM (read-only), WORM (write-once read-many) & EO (erasable optical disks).

Que 4.22. **A moving arm disc storage device has the following specifications :**
Number of Tracks per recording surface = 200
Disc rotation speed = 2400 revolution/minute
Track-storage capacity = 62500 bits
Estimate the average latency and data transfer rate of this device.

AKTU 2017-18, Marks 07

Answer

Disk rotation speed = 2400 rpm

As we know that average latency = $\frac{1}{2}$ × Rotation time

$\because$ 2400 rotation in one minute so the time for one rotation = $\frac{1}{2} * \frac{60}{2400}$ s

$$= \frac{3}{240} \text{ s}$$
$$= 12.5 \text{ ms}$$

Track storage capacity = 62500 bits
And in one rotation head cover entire track, so, disk transfer rate

$$= 62500 * \frac{2400}{60} \text{ s}$$

(where 2400/60 is number of rotations per second)

$$= 2.5 * 10^6 \text{ bps}$$

Que 4.23. **What is virtual memory ?**

OR

Write a short note on virtual memory. **AKTU 2014-15, Marks 05**

Answer

1. Virtual memory is a memory management capability of an OS that uses hardware and software to allow a computer to compensate for physical memory shortages by temporarily transferring data from Random Access Memory (RAM) to disk storage.

2. Virtual address space is increased using active memory in RAM and inactive memory in Hard Disk Drives (HDDs) to form contiguous addresses that hold both the application and its data.
3. A system using virtual memory can load larger programs or multiple programs running at the same time, allowing each one to operate as if it has infinite memory and without having to purchase more RAM.
4. Virtual memory is a facility that allows program to address memory from local point of view, without regard to the amount of main memory.
5. A virtual memory system provides a mechanism for translating program generated addresses into correct main memory locations.
6. This is done dynamically, while programs are being executed in the CPU.

Que 4.24. Explain the following memory schemes discussing why needed the :

i. Interleaved memory

ii. Associative memory **AKTU 2014-15, Marks 10**

OR

Explain the working principle of associative memory.

Answer

i. Interleaved memory :

1. Interleaved memory is a design made to compensate for the relatively slow speed of Dynamic Random Access Memory (DRAM).
2. This is done by spreading memory addresses evenly across memory banks.
3. Thus, in contiguous memory, reads and writes are done using each memory bank in turn, resulting in higher memory throughputs due to reduced waiting for memory banks to become ready for desired operations.
4. As shown in Fig. 4.24.1, the lower order k bits of the address are used to select the module (Memory bank) and higher order m bits give a unique memory location in the memory bank that is selected by the lower order k bits.
5. Thus in this way consecutive memory locations are stored on different memory banks.
6. Whenever requests to access consecutive memory locations are being made several memory banks are kept busy at any point in time.
7. This results in faster access to a block of data in the memory and also results in higher overall utilization of the memory system as a whole.

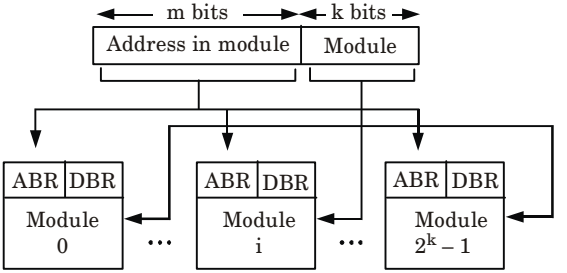


Fig. 4.24.1.

8. If k bits are allotted for selecting the bank as shown in the Fig. 4.24.1, there have to be total $2k$ banks. This ensures that there are no gaps of non-existent memory locations.

ii. Associative memory :

1. Associative memory is a memory in which location is accessed by a field of data word stored in the memory rather than by any address.
2. It can be viewed as a random access type memory which in addition to having a physically wired-in addressing mechanism also has wired-in logic for bit comparison.
3. This logic circuit enables comparison of desired bit positions of all the words with a specified input key.
4. This comparison is done simultaneously for all the words.
5. This is also called Content Addressable Memory (CAM).

Working principle of associative memory :

1. The mask register specifies the key field.
2. Input data is simultaneously compared with the key field of each word.
3. The select circuit implements two functions :
   a. It stores the word location (s) for which match has occurred.
   b. It reads out the word(s) in predetermined order for the match position (s).
4. Thus, a word stored in the associative memory is a pair (key, Data). Any subfield of the word can be specified as the key.
5. The read or write instruction is preceded by the match instruction having the format.

   Match key, Input data
6. The read/write operation can next be performed on each of the words for which match signal is generated.

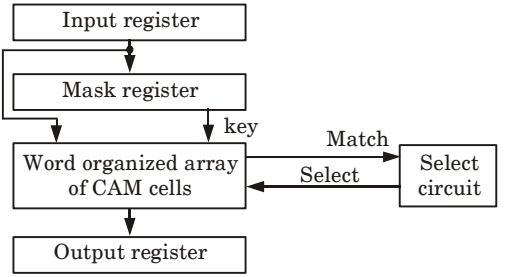


Fig. 4.24.2. Organization of an associative memory.

Que 4.25. **What is associative memory ? Explain with the help of a block diagram. Also mention the situation in which associative memory can be effective utilized.** **AKTU 2018-19, Marks 07**

Answer

Associative memory and block diagram : Refer Q. 4.24, Page 4–24B, Unit-4.

Associative memory is effectively utilized when doing a large number of pattern match and lookup.

Que 4.26. **What do you mean by CAM ? Explain its major characteristics.** **AKTU 2015-16, Marks 10**

Answer

CAM :

1. Content Addressable Memory (CAM) is computer memory that operates like a hardware search engine for search-intensive applications.
2. CAM is capable of searching its entire contents in a single clock cycle.
3. It does that by pairing the SRAM-based memory with additional logic comparison circuitry that is active on every clock cycle.
4. The way CAM functions is almost the opposite of Random Access Memory (RAM).
5. Data stored on CAM, can be accessed by searching for the content itself, and the memory retrieves the addresses where that content can be found.
6. Because of its parallel nature, CAM is much faster than RAM for searching.

The major characteristics are :

1. This memory is accessed simultaneously and in parallel on the basis of data content rather than by specific address or location.
2. This memory is capable of finding an empty unused location to store the word.
3. This memory is uniquely suited to do parallel searches by data association.
4. Each cell must have storage capability as well as logic circuits for matching its content with an external argument.

Que 4.27. Discuss the conceptual organization of a multilevel memory system used in computers.

Answer

Conceptual organization of a multilevel memory system consist of basically four elements *i.e.*, CPU registers, cache memory, main memory and secondary memory.

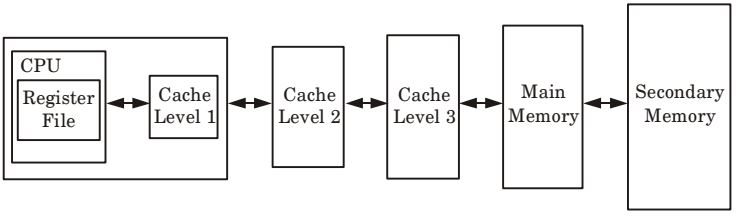


Fig. 4.27.1. Conceptual organization of multilevel memories in a computer system.

1. **CPU registers :**
   a. The high-speed registers in CPU are used as the temporary storage for instructions and data.
   b. They usually form a general purpose register file for storing data as it is processed. A capacity of 32 data words is typical for a register file and each register can be accessed within a single clock cycle that is in a few nano seconds.
2. **Cache memory :**
   a. Cache memory is positioned logically between register files and the main memory.
   b. Its capacity is less than main memory but access time is much lesser, that is this types of memories are much faster than the main memory.

   It operates in three levels :

   a. Level 1 (L1) cache : It is built directly into the processor chip. It has small capacity from 8 kb to 128 kb.

b. Level 2 (L2) cache : It is built on current processors on processor chip. It has capacity from 64 kb to 16 MB.

c. Level 3 (L3) cache : This cache is separate from processor chip on the motherboard. Its capacity is up to 8 MB.

3. **Main or primary memory :** Refer Q. 4.1, Page, Unit-4.
4. **Secondary memory :** Refer Q. 4.1, Page, Unit-4.

Que 4.28. What do you mean by locality of reference ? Explain with suitable example. **AKTU 2017-18, Marks 07**

Answer

Locality of reference is a term for the phenomenon in which the same values or related storage locations are frequently accessed, depending on the memory access pattern.

Example :

1. Take the example of an operating system. Ideally, we would like an unlimited amount of main memory, instantly accessible.
2. In practice, we have a limited amount of main memory, and because it is cheaper, a very large amount of secondary memory.
3. However the trade-off is that secondary memory tends to be several orders of magnitude slower than primary memory.
4. We can approach the ideal by keeping the more often used data in main memory, and everything else in secondary memory.
5. Because of the principle of Locality of Reference, we can be sure that most memory references will be to locations already stored in main memory, thereby improving efficiency and providing a flat memory model.
6. This scheme is used in modern operating systems and is called virtual memory. Virtual memory gives users the appearance of unlimited primary memory by transparently utilizing secondary memory.

VERY IMPORTANT QUESTIONS

Following questions are very important. These questions may be asked in your SESSIONALS as well as UNIVERSITY EXAMINATION.

Q. 1. Give the structure of commercial 8M × 8 bit DRAM chip.

Ans. Refer Q. 4.4.

Q. 2. Write short note on organization of 2D and 2.5D memory organization.

Ans. Refer Q. 4.5.

Q. 3. How main memory is useful in computer system ? Explain the memory address map of RAM and ROM.

Ans. Refer Q. 4.7.

Q. 4. A computer uses RAM chips of 1024*1 capacity.

i. How many chips are needed and how should their address lines be connected to provide a memory capacity of 1024*8 ?

ii. How many chips are needed to provide a memory capacity of 16 KB ? Explain in words how the chips are to be connected to the address bus.

Ans. Refer Q. 4.8.

Q. 5. A ROM chip of 1024*8 has four select inputs and operates from a 5 volt power supply. How many pins are needed for the IC package ? Draw a block diagram and label all input and output terminals in the ROM.

Ans. Refer Q. 4.9.

Q. 6. A computer uses a memory unit with 256 K words of 32 bits each. A binary instruction code is stored in one word of memory. The instruction has four parts : an indirect bit, an operation code, a register code part to specific one of 64 register and an address part.

i. How many bits are there in the operation code, the register code part and the address part ?

ii. Draw the instruction word format and indicate the number of bits in each part.

iii. How many bits are there in the data and address inputs of the memory ?

Ans. Refer Q. 4.10.

Q. 7. Discuss the various types of address mapping used in cache memory.

Ans. Refer Q. 4.14.

Q. 8. Consider a cache uses a direct mapping scheme. The size of main memory is 4 K bytes and word size of cache is 2 bytes. The size of cache memory is 128 bytes. Find the following :

i. The size of main memory address (assume each byte of main memory has an address)

ii. Address of cache block

iii. How many memory location address will be translated to cache address/block/location ?

iv. How can it be determined if the content of specified main memory address in cache ?

Ans. Refer Q. 4.16.

Q. 9. A two way set associative cache memory uses blocks of 4 words. The cache can accommodate a total of 2048 words from memory. The main memory size is 128 K × 32.

i. Formulate all pertinent information required to construct the cache memory.

ii. What is the size of the cache memory ?

Ans. Refer Q. 4.17.

Q. 10. What is the distinction between spatial locality and temporal locality ?

Ans. Refer Q. 4.18.

Q. 11. A moving arm disc storage device has the following specifications :
Number of Tracks per recording surface = 200
Disc rotation speed = 2400 revolution/minute
Track-storage capacity = 62500 bits
Estimate the average latency and data transfer rate of this device.

Ans. Refer Q. 4.22.

Q. 12. Write a short note on virtual memory.

Ans. Refer Q. 4.23.

Q. 13. Explain the following memory schemes discussing why needed the :

i. Interleaved memory

ii. Associative memory

Ans. Refer Q. 4.24.

Q. 14. What is associative memory ? Explain with the help of a block diagram. Also mention the situation in which associative memory can be effective utilized.

Ans. Refer Q. 4.25.

Q. 15. What do you mean by CAM ? Explain its major characteristics.

Ans. Refer Q. 4.26.

Q. 16. What do you mean by locality of reference ? Explain with suitable example.

Ans. Refer Q. 4.28.

5
UNIT

Input / Output

CONTENTS

PART-1

Input / Output : Peripheral Devices, I / O Interface, I / O Ports.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 5.1. Explain the term peripheral devices.

Answer

1. Peripheral devices are the computer devices that are connected to the computer externally such as printer, scanner, keyboard, mouse, tape device, microphone and external modem. It can be internal such as CD-ROM or internal modem.
2. Peripheral devices can be classified according to their functions :
   - **i. Input :** Input devices are the type of computer devices that are used to provide the control signals to the computer. Keyboard and mouse are the examples of the input devices.
   - **ii. Output :** Output devices are the devices that are used to display the results. Printer, scanner, speaker and the monitor are the examples of the output devices.
   - **iii. Storage :** A storage device is a device that is used to store the information such as hard-disk drive, flash drive, floppy disk and the tape drive.

Que 5.2. Describe I/O interface. Why they are needed ?

OR

Why input-output interface is required ? Describe in detail.

AKTU 2015-16, Marks 15

Answer

I/O interface provides a method of transferring information between internal storage and external I/O devices.

The major requirements for an I/O module can be given as :

1. **Processor communication :** This involves the following tasks :
   - **a.** Exchange of data between processor and I/O module.
   - **b. Command decoding :** The I/O module for a disk drive may accept the following commands from the processor : READ SECTOR, WRITE SECTOR, SEEK track, etc.

c. **Status reporting :** The device must be able to report its status to the processor. For example, disk drive busy, ready etc.

d. Status reporting may also involve reporting various errors.

e. **Address recognition :** Each I/O device has a unique address and the I/O module must recognize this address.

2. **Device communication :** The I/O module is able to perform device communication such as status reporting.

3. **Control and timing :** The I/O module is able to co-ordinate the flow of data between the internal resources (such as processor, memory) and external devices.

4. **Data buffering :**

   a. This is necessary as there is a speed mismatch between speed of data transfer between processor and memory and external devices.

   b. Data coming from the main memory are sent to an I/O module in a rapid burst.

   c. The data is buffered in the I/O module and then sent to the peripheral device at its rate.

5. **Error detection :**

   a. The I/O module is able to detect errors and report them to the processor.

   b. These errors may be mechanical errors (such as paper jam in a printer), or changes in the bit pattern of transmitted data. A common way of detecting such errors is by using parity bits.

Que 5.3. **Explain I/O bus and I/O command.**

Answer

I/O bus :

1. The I/O bus consists of data lines, address lines and control lines.
2. It acts as a communication link between processor and several peripheral devices.
3. It comprises of magnetic tape, magnetic disk, printer and terminal.
4. The I/O bus from processor is attached to all peripheral interfaces.

I/O command :

When an address is in the address lines, at the same time, the processor provides a function code in control lines which is referred to as I/O command. It is of four types :

i. **Control command :** It is issued to activate the peripheral and to inform it what to do.

ii. **Status command :** It is used to test various status conditions in the interface and peripheral.

iii. **Output data command :** It causes the interface to respond by transferring data from bus into one of its registers.

iv. **Input data command :** It is the opposite of data output. The interface receives an item of data from peripheral and places it in its buffer register.

PART-2

Interrupts : Interrupt Hardware, Types of Interrupts and Exceptions.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 5.4. Write a short note on interrupts.

AKTU 2014-15, Marks 05

OR

Define interrupt. When a device interrupt occurs how does the processor determine which device has issued the interrupt ?

Answer

Interrupt : An interrupt is a signal sent by an I/O interface to the CPU when it is ready to send information to the memory or receive information from the memory.

Identifying the source of an interrupt :

Two methods are available to determine the interrupting device :

a. Polled interrupts :

1. On receiving an interrupt request the micro-processor will execute a routine causing it to poll each of the devices in turn.
2. Devices have a status register containing one or more interrupt request bits.
3. If a device caused the interrupt, its interrupt flag bit will be set.
4. The appropriate service routine will then be selected and the device serviced.
5. Polling can be very inefficient if there are many devices capable of causing an interrupt.
6. This method uses only software methods to identify an interrupting device.

b. Vectored interrupts :

1. This is the method used in modern computers.
2. It is sometimes referred to as hardware identification since additional hardware is required.

3. Each vector is identified by an interrupt number from 0 to 255 and provides the processor with the means of addressing the appropriate interrupt handler.
4. A vector address is determined by multiplying its vector number by 4.

Que 5.5. Explain the sequence that takes place when an interrupt occurs. **AKTU 2015-16, Marks 10**

Answer

When an interrupt occurs, sequence of following six steps takes place :

1. Interrupt recognition :

a. The interrupt recognition is recognized by the processor of an interrupt request due to activation of an interrupt request line or an internal mechanism.

b. In this step, the processor can determine which device or CPU, component made the request.

2. Status saving :

a. The goal of this step is to make the interrupt sequence transparent to the interrupted process.

b. Therefore, the processor saves the flags and registers that may be changed by the interrupt service routine so that they may be restored after the service routine is finished.

3. Interrupt masking :

a. For the first few steps of the sequence, all interrupts are masked out so that no other interrupt may be processed before the processor status is saved.

b. The mask is then set to accept interrupts of higher priority.

4. Interrupt acknowledgment :

a. At some point, the processor must acknowledge the interrupt being serviced, so that the interrupting device becomes free to continue its task.

b. One of the ways is to have an external signal line denoted to interrupt acknowledge.

5. Interrupt service routine :

a. At this point, the processor initiates the interrupt service routine.

b. The address of the routine can be obtained in several ways, depending on the system architecture.

c. The simplest is found in the polling method, in which one routine polls each device to find which one interrupted.

6. Restoration and return :

a. After the interrupt service routine has completed its processing, it restores all the registers it has changed, and the processor restores all the registers and flags that were saved at the initiation of the interrupt routine.

b. If this is done correctly, the processor should have the same status as before the interrupt was recognized.

Que 5.6. How system resolve the priority of interrupt ?

OR

Explain polling and daisy chaining method.

Answer

There are two methods of system to resolve the priority of interrupt :

a. Polling method :

1. When interrupt requests arrive from two or more devices simultaneously, the processor has to decide which request should be serviced first and which one should be delayed.
2. The processor takes the decision with the help of interrupt priorities. It accepts the request having the highest priority.
3. In this case polling is used to identify the interrupting device, priority is automatically assigned by the order in which devices are polled.
4. Therefore, no further arrangement is required to accommodate simultaneous interrupt requests. However, the priority of any device is usually determined by the way the device is connected to the processor.

b. Chaining method :

1. Most common way to connect the devices is to form a daisy chain, as shown in Fig. 5.6.1.
2. The Interrupt Request line (INTR) is common to all the devices and the Interrupt Acknowledge line (INTA) is connected in a daisy chain model.
3. In daisy chain fashion the signal is allowed to propagate serially through the devices.
4. When more than one devices issue an interrupt request, the INTR line is activated and processor responds by setting the INTA line.

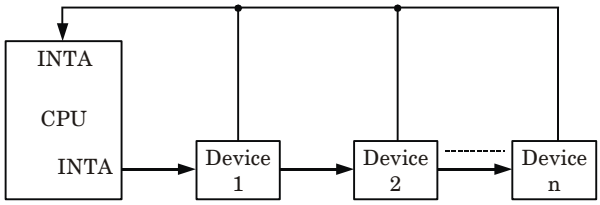


Fig. 5.6.1. Interrupt priority system using daisy chain.

5. This signal is received by device 1. Device 1 passes the signal to the device 2 only if it requires any service.
6. If device 1 requires service, it blocks the INTA line and puts its identification code on the data lines. Therefore, in daisy chain arrangement, the device that is electrically closest to the processor has the highest priority.

Que 5.7. How interrupts are classified ?

Answer

Basically the interrupts can be classified in the following three ways :

1. Hardware and software interrupts :

a. The interrupts initiated by an external hardware by sending an appropriate signal to the interrupt pin of the CPU is called hardware interrupt.

b. The software interrupts are program instructions. These instructions are inserted at desired location in a program. While running a program, if software interrupt instruction is encountered the CPU initiates an interrupt.

2. Vectored and non-vectored interrupts :

a. When an interrupt signal is accepted by the CPU, and the program control automatically branches to a specific address (called vector address) then the interrupt is called vectored interrupt.

b. In non-vectored interrupts the interrupting device should supply the address of the ISR to be executed in response to the interrupt.

3. Maskable and non-maskable interrupts :

a. The interrupts whose request can be either accepted or rejected by the CPU are called maskable interrupts.

b. The interrupts whose request has to be definitely accepted by the CPU are called non-maskable interrupts.

Que 5.8. **Explain the difference between vectored and non-vectored interrupt. Explain stating examples of each.**

AKTU 2018-19, Marks 07

Answer

S. No.	Vectored interrupt	Non-vectored interrupt
1.	Vectored interrupt are those interrupt that generates the interrupt request, identifies itself directly to the processor.	Non-vectored interrupt are those in which vector address is not pre-defined.
2.	Vector interrupt have fixed memory location for transfer of control for normal execution.	Non-vectored interrupt do not have fixed memory location for transfer of control for normal execution.
3.	Vectored interrupt has memory address.	A non-vectored interrupt do not have memory address.
4.	The vectored interrupt allows the CPU to be able to know what ISR to carry out in software.	When a non-vectored interrupt received, it jump into the program counter to fixed address in hardware.
5.	Response time is low.	Response time is high.
6.	TRAP is a vectored interrupt.	INTR is non-vectored interrupt.

Que 5.9. **Explain the types of interrupt on the basis of timer.**

Answer

There are following types of interrupt on this basis of timer :

1. **Level-triggered :**
   a. A level-triggered interrupt is a class of interrupts where the presence of an unserviced interrupt is indicated by a high level (1), or low level (0), of the interrupt request line.
   b. A device wishing to signal an interrupt drives line to its active level, and then holds it at that level until serviced.
2. **Edge-triggered :**
   a. An edge-triggered interrupt is a class of interrupts that are signaled by a level transition on the interrupt line, either a falling edge (1 to 0) or a rising edge (0 to 1).

b. A device wishing to signal an interrupt drives a pulse onto the line and then releases the line to its quiescent state.

c. If the pulse is too short to be detected by polled I/O then special hardware may be required to detect the edge.

3. Hybrid :

a. Some systems use a hybrid of level-triggered and edge-triggered signaling. The hardware not only looks for an edge, but it also verifies that the interrupt signal stays active for a certain period of time.

b. A common use of a hybrid interrupt is for the NMI (non-maskable interrupt) input.

c. Because NMIs generally signal major or even catastrophic system events, a good implementation of this signal tries to ensure that the interrupt is valid by verifying that it remains active for a period of time.

PART-3

Modes of Data Transfer : Programmed I/O.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 5.10. Write a short note on programmed I/O.

AKTU 2014-15, Marks 05

OR

Discuss the programmed I/O method for controlling input-output operations.

Answer

Programmed I/O :

1. When the CPU is executing a program and executes an instruction relating to I/O, it executes that instruction by issuing a command to the appropriate I/O module.

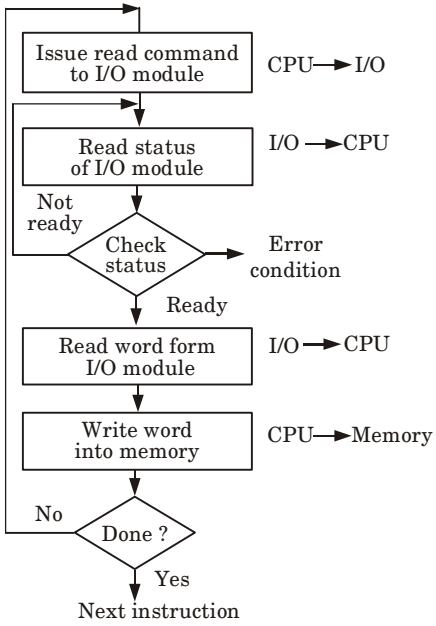


Fig. 5.10.1. Programmed I/O.

2. With programmed I/O, the I/O module will perform the requested action and then set the appropriate bits in the I/O status register.
3. The I/O module takes no further action to alert the CPU.
4. It does not interrupt the CPU.
5. Thus, it is the responsibility of CPU to periodically check the status of the I/O module until it finds that the operation is complete.

PART-4

Interrupt Initiated I/O and Direct Memory Access, I/O Channels and Processors.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 5.11. **What is interrupt initiated I/O ?**

Answer

1. Interrupt initiated I/O is a mode of data transfer which removes the drawback of the programmed I/O mode.
2. The CPU issues commands to the I/O module then proceeds with its normal work until interrupted by I/O device on completion of its work.
3. For input, the device interrupts the CPU when new data has arrived and is ready to be retrieved by the system processor.
4. The actual actions to perform depend on whether the device uses I/O ports, memory mapping.
5. For output, the device delivers an interrupt either when it is ready to accept new data or to acknowledge a successful data transfer.
6. Memory-mapped and DMA-capable devices usually generate interrupts to tell the system that they are done with the buffer.
7. Although interrupt relieves the CPU of having to wait for the devices, but it is still inefficient in data transfer of large amount because the CPU has to transfer the data word by word between I/O module and memory.

Que 5.12. **Write short note on DMA.** **AKTU 2014-15, Marks 05**

OR

Explain the working of DMA controller with the help of suitable diagrams. **AKTU 2017-18, Marks 07**

OR

Write a short note on DMA based data transfer.

AKTU 2018-19, Marks 07

OR

Give the block diagram of DMA controller. Why are the read and write control lines in a DMA controller bidirectional ?

AKTU 2018-19, Marks 07

Answer

DMA :

1. DMA stands for "Direct Memory Access" and is a method of transferring data from the computer's RAM to another part of the computer without processing it using the CPU.
2. While most data that is input or output from our computer is processed by the CPU, some data does not require processing, or can be processed by another device.
3. DMA can save processing time and is a more efficient way to move data from the computer's memory to other devices.
4. In order for devices to use direct memory access, they must be assigned to a DMA channel. Each type of port on a computer has a set of DMA channels that can be assigned to each connected device.

5. For example, a PCI controller and a hard drive controller each have their own set of DMA channels.

Block diagram for DMA :

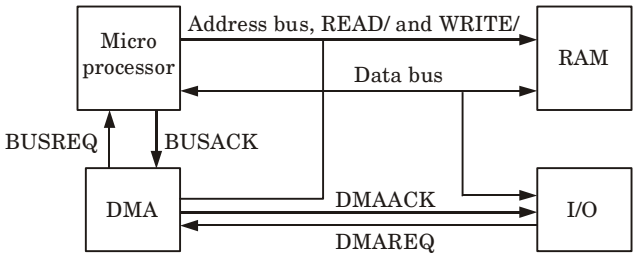


Fig. 5.12.1.

Working of DMA controller :

1. When the peripheral device sends a DMA request, the DMA controller activates the BR line, informing the CPU to relinquish the buses.
2. The CPU responds with its BG line, informing the DMA that its buses are disabled.

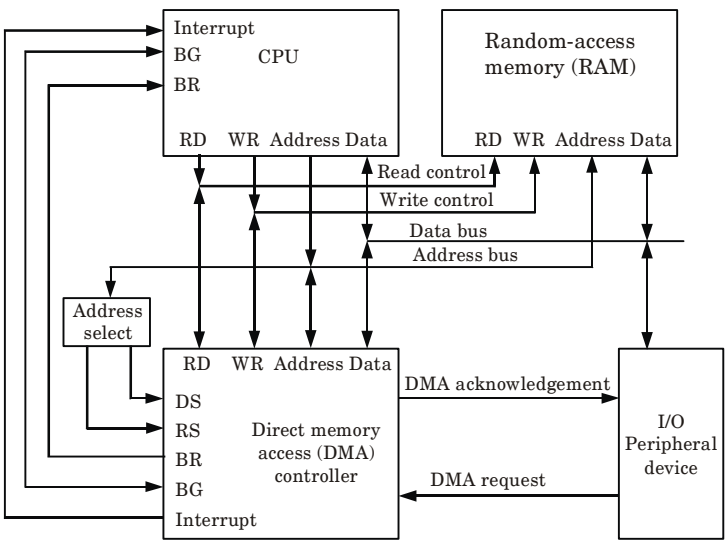


Fig. 5.12.2. DMA transfer in a computer system.

3. The DMA then puts the current value of its address register into the address bus, initiates the RD or WR signal, and sends a DMA acknowledge to the peripheral device.
4. The direction of transfer depends on the status of the BG line.

a. When BG = 0, the RD and WR are input lines allowing the CPU to communicate with the internal DMA registers.

b. When BG = 1, the RD and WR are output lines from the DMA controller to the random access memory to specify the read or write operation for the data.

5. When the peripheral device receives a DMA acknowledge, it puts a word in the data bus (for write) or receives a word from the data bus (for read).

6. Thus, the DMA controls the read or write operations and supplies the address for the memory.

7. The peripheral unit can communicate with memory through the data bus for direct transfer between the two units while the CPU is momentarily disabled.

Reason for bidirectional read and write control lines : Read and write control lines in a DMA controller is bidirectional because the microprocessor fetch (read) the data from the memory and write data to the memory.

Que 5.13. What is the difference between isolated I/O and memory mapped I/O ? Explain the advantages and disadvantages of each.

AKTU 2014-15, Marks 10

Answer

Difference between isolated I/O and memory mapped I/O :

S. No.	Isolated I/O	Memory mapped I/O
1.	Isolated I/O uses separate memory space.	Memory mapped I/O uses memory from the main memory.
2.	Limited instructions can be used. Those are IN, OUT, INS, OUTS.	Any instruction which references to memory can be used.
3.	The addresses for isolated I/O devices are called ports.	Memory mapped I/O devices are treated as memory locations on the memory map.
4.	Efficient I/O operations due to separate bus.	Inefficient I/O operations due to single bus for data and addressing.
5.	Comparatively larger in size.	Smaller in size.
6.	Uses complex internal logic.	Common internal logic for memory and I/O devices.
7.	Slower operations.	Faster operations.

Advantage of isolated I/O :

1. The devices of I/O are treated in a separate domain as compared to memory.
2. A total of 1MB address space is allowed for memory applications.
3. In order to maximize the I/O operations (isolated) separate instructions are always provided to perform these operations.

Disadvantage of isolated I/O :

1. The data transfer only occurs between the I/O port and the registers.

Advantages of memory mapped I/O :

1. I/O intensive operation is fast.
2. The SQLite library needs less RAM.

Disadvantages of memory mapped I/O :

1. If an I/O error on a memory-mapped file cannot be caught by the application, results in a program crash.
2. Performance is reduced by the use of memory-mapped I/O.

Que 5.14. What do you mean by Input-Output (I/O) processor ?

Answer

1. IOP is designed to handle the details of I/O processing. Unlike the DMA controller that must be set up entirely by the CPU, the IOP can fetch and execute its own instructions.
2. IOP instructions are specifically designed to facilitate I/O transfers.

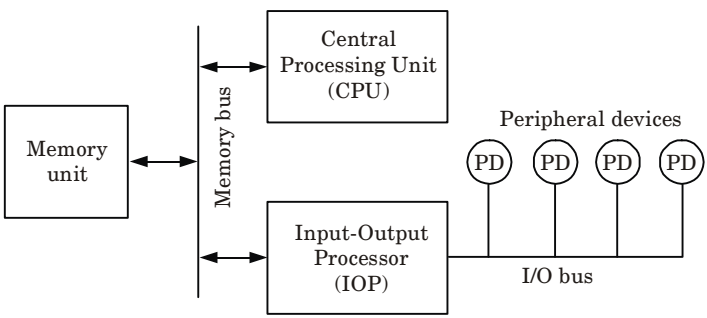


Fig. 4.14.1. The block diagram of a computer with IOP.

3. The memory unit occupies central position and can communicate with each processor by means of direct memory access.
4. The CPU is responsible for processing data. The IOP provides a path for transfer of data between various peripheral devices and the memory unit.

5. The CPU assigns the task of initiating the I/O program. From then on the IOP operates independent of the CPU and continues to transfer data from external devices and memory.

Que 5.15. Explain I/O channels with its types.

Answer

1. A channel is an independent hardware component that co-ordinate all I/O to a set of controllers. Computer systems that use I/O channel have special hardware components that handle all I/O operations.
2. Channels use separate, independent and low cost processors for its functioning which are called channel processors.
3. Channel processors are simple, but contains sufficient memory to handle all I/O tasks.
4. When I/O transfer is complete or an error is detected, the channel controller communicates with the CPU using an interrupt, and informs CPU about the error or the task completion.
5. Each channel supports one or more controllers or devices. Channel programs contain list of commands to the channel itself and for various connected controllers or devices.

Types of I/O Channels :

1. **Multiplexer :** The Multiplexer channel can be connected to a number of slow and medium speed devices. It is capable of operating number of I/O devices simultaneously.
2. **Selector :** This channel can handle only one I/O operation at a time and is used to control one high speed device at a time.
3. **Block-Multiplexer :** It combines the features of both multiplexer and selector channels.

PART-5

Serial Communication : Synchronous and Asynchronous Communication, Standard Communication Interfaces.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 5.16. What do you mean by serial communication ? What are the transmission modes of serial communication ?

Answer

1. In serial communication bits are transferred one after the other over a single communication path.
2. Serial communication is a device communication protocol that is standard on almost every PC.
3. A given transmission on a communication channel between two machines can occur in several different ways.

Modes of serial communication :

1. Simplex connection :

a. A simplex connection is a connection in which the data flows in only one direction, from the transmitter to the receiver.

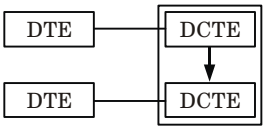


Fig. 5.16.1. Simplex connection.

2. Half-duplex connection :

a. A half-duplex connection (sometimes called an alternating connection or semi-duplex) is a connection in which the data flows in one direction or the other, but not both at the same time.

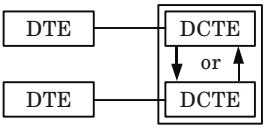


Fig. 5.16.2. Half-duplex connection.

b. With this type of connection, each end of the connection transmits in turn.

3. Full-duplex connection :

a. A full-duplex connection is a connection in which the data flow in both directions simultaneously.

b. This can be achieved by means of a four-wire link, with a different pair of wires dedicated to each direction of transmission.

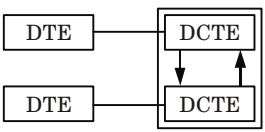


Fig. 5.16.3. Full-duplex connection.

Que 5.17. Explain synchronous communication and asynchronous communication.

Answer

Synchronous communication :

1. In the synchronous communication scheme, after a fixed number of data bytes, a special bit pattern called SYNC is sent as shown in Fig. 5.17.1.

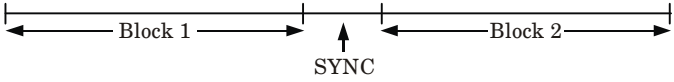


Fig. 5.17.1. Synchronous communication.

2. There is no gap between adjacent characters in the synchronous communication.
3. There is a continuous stream of data bits coming at a fixed speed in a synchronous communication scheme.
4. Synchronous communication is used generally when two computers are communicating to each other or when a buffered terminal is communicating to the computer.

Asynchronous communication :

1. In the asynchronous communication scheme, each character includes start and stop bits, as shown in Fig. 5.17.2.

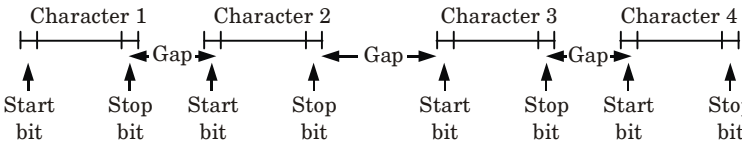


Fig. 5.17.2. Asynchronous communication.

2. There are some gaps between adjacent characters in the asynchronous communication.
3. In the asynchronous communication scheme, the bits within a character frame (including start, parity and stop bits) are sent at the baud rate.
4. Asynchronous communication is used when slow speed peripherals communicate with the computer.

Que 5.18. Discuss the advantages and disadvantages of synchronous and asynchronous transmission.

Answer

Advantages of synchronous transmission :

1. Lower overhead and thus, greater throughput.

Disadvantages of synchronous transmission :

1. Slightly more complex.
2. Hardware is more expensive.

Advantages of asynchronous transmission :

1. Simple and does not require synchronization of both communication sides.

2. Hardware are cheaper as clock is not required.
3. Set-up is very fast, so well suited for applications where messages are generated at irregular intervals.

Disadvantages of asynchronous transmission :

1. Large relative overhead, a high proportion of the transmitted bits are uniquely for control purposes and thus carry no useful information.

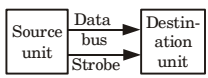
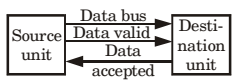
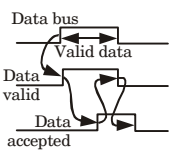
Que 5.19. Differentiate among :

i. Strobe control and handshaking asynchronous data transfer modes.
ii. Processor and IOP.
iii. Synchronous and asynchronous transmission.
iv. Character-oriented and Bit-oriented protocols.
v. DMA and interrupt initiated I/O techniques.

AKTU 2015-16, Marks 15

Answer

i. Strobe control and handshaking asynchronous data transfer modes :

S. No.	Parameter	Strobe control	Handshaking
1.	Control line	It employs a single control line to time each transfer.	It employs more than single control line to time each transfer.
2.	Acknowledgement	Reply message is not present.	Reply message is present.
3.	Block diagram		
4.	Timing diagram		

ii. Processor and IOP :

S. No.	Processor	IOP
1.	Processor is CPU.	IOP is port of CPU processing.
2.	Handles arithmetic and logical tasks.	Handles only I/O processing.
3.	DMA controller is set-up by CPU.	IOP is a processor with DMA.

iii. Synchronous and asynchronous transmission :

S. No.	Synchronous transmission	Asynchronous transmission
1.	Transmitter and receivers are synchronized by clock.	Transmitter and receivers are not synchronized by clock.
2.	Data bits are transmitted with synchronization of clock.	Bits of data are transmitted at constant rate.
3.	Data transfer takes place in blocks.	Data transfer is character oriented.

iv. Character-oriented and Bit-oriented protocols :

S. No.	Character-oriented protocol	Bit-oriented protocol
1.	The character-oriented protocol is based on the binary code of a character set.	The bit-oriented protocol does not use characters in its control field and is independent of any particular code.
2.	The code has 128 characters, of which 95 are graphics characters and 33 are control characters.	It allows the transmission of serial bit stream of any length without the implication of character boundaries.

Message format for character-oriented protocol :

SYN	SYN	SOH	Header	STX	Text	ETX	BCl

Frame format for bit-oriented protocol :

Flag 01111110	Address 8 bits	Control 8 bits	Information any number of bits	Frame check 16 bits	Flag 01111110

v. DMA and interrupt initiated I/O techniques :

S. No.	DMA	Interrupt initiated I/O
1.	As DMA initializes, CPU become idle.	CPU executes the current program, during the interrupt initiated I/O technique.
2.	As the DMA disables, memory buses are returned to CPU and CPU starts executing its program.	After the transfer, CPU returns to the previous program to continue.

Que 5.20. **Describe the subroutine. Write a program which move the block of data.** **AKTU 2016-17, Marks 7.5**

OR

Write a note on subroutines. **AKTU 2016-17, Marks 15**

Answer

Subroutine :

1. A subroutine is a set of common instructions that can be used in a program many times.
2. A subroutine consists of a self-contained sequence of instructions that carries out a given task.
3. Each time that a subroutine is used in the main part of the program, a branch is executed to the beginning of the subroutine.
4. After the subroutine has been executed, a branch is made back to the main program.
5. A branch can be made to the subroutine from any part of the main program.
6. Because branching to a subroutine and returning to the main program is such a common operation, all computers provide special instructions to facilitate subroutine entry and return.

Program :

```
LXI H, XX50H        ; Set up HL as a pointer for the source memory
LXI D, XX70H        ; Set up DE as a pointer for the destination
                      memory
MVI B, 10H          ; Set up B as byte counter
NEXT : MOV A,M      ; Get data byte from the source memory
STAX D              ; Store the data byte in destination memory
INX H
INX D               ; Get ready to transfer next byte
DCR B
JNZ NEXT            ; Go back to get next byte if byte counter ≠ 0
HLT
```

VERY IMPORTANT QUESTIONS

Following questions are very important. These questions may be asked in your SESSIONALS as well as UNIVERSITY EXAMINATION.

Q. 1. Why input-output interface is required ? Describe in detail.
Ans. Refer Q. 5.2.

Q. 2. Write a short note on interrupts.
Ans. Refer Q. 5.4.

Q. 3. Explain the sequence that takes place when an interrupt occurs.
Ans. Refer Q. 5.5.

Q. 4. Explain the difference between vectored and non-vectored interrupt. Explain stating examples of each.
Ans. Refer Q. 5.8.

Q. 5. Write a short note on programmed I/O.
Ans. Refer Q. 5.10.

Q. 6. Write a short note on DMA based data transfer.
Ans. Refer Q. 5.12.

Q. 7. What is the difference between isolated I/O and memory mapped I/O ? Explain the advantages and disadvantages of each.
Ans. Refer Q. 5.13.

Q. 8. Differentiate among :
i. Strobe control and handshaking asynchronous data transfer modes.
ii. Processor and IOP.
iii. Synchronous and asynchronous transmission.
iv. Character-oriented and Bit-oriented protocols.
v. DMA and interrupt initiated I/O techniques.
Ans. Refer Q. 5.19.

Q. 9. Describe the subroutine. Write a program which move the block of data.
Ans. Refer Q. 5.20.

1 UNIT

Central Processing Unit (2 Marks Questions)

1.1. What are the various ways of specifying the binary point in a register ?

Ans. **There are two ways of specifying the binary point in a register which are :**

i. By giving it a fixed position.
ii. By employing a floating-point representation.

1.2. What are the various facts related to bus and bus system ?

Ans. **Various facts related to bus and bus system are :**

i. A bus system will multiplex k registers of n bits each to produce an n-line common bus.
ii. The number of multiplexers needed to construct the bus is equal to n, the number of bits in each register.
iii. The size of each multiplexer must be $k \times 1$, since it multiplexes k data lines.

1.3. Give various advantages of polling method.

Ans. **Various advantages of polling method are :**

i. The priority can be changed by altering the polling sequence stored in the controller.
ii. If the one module fails, entire system does not fail.

1.4. Discuss the basic component of register transfer logic.

Ans. **Basic components of register transfer logic are :**

i. Registers and their functions
ii. Information
iii. Operations
iv. Control function

1.5. What is the relation between bus width and number of bit transferred ?

Ans. Bus width is directly proportional to the number of bit transferred. The wider the data bus, then greater will be the number of bits transferred at one time.

1.6. Define memory transfer.

Ans. Memory transfer involves basic operation like fetch (read) or store (write). The fetch operation transfers a copy of content from memory location to CPU. The store operation transfers the word information from CPU to specific memory location.

1.7. Define bus transfer.

Ans. The data transfer between various blocks connected to the common bus is called bus transfer. Common bus system is shared by all the units.

1.8. Explain control word. **AKTU 2015-16, Marks 02**

Ans. Control word is defined as a word whose individual bits represent the various control signals. Therefore each of the control steps in the control sequence of an instruction defines a unique combination of 0s and 1s.

1.9. Compare register stack and memory stack.

Ans.

S. No.	Register stack	Memory stack
1.	A stack can be placed in a portion of a logical memory or can be organized as a collection of number of memory words or registers.	The stack is implemented as a standalone and also implemented as a random access memory attached to CPU.
2.	The stack pointer register (SP) contains a binary number whose value is equal to address of the word that is currently on top of the stack.	The implementation of a stack in the CPU is done by assigning a portion of memory to a stack operation.

2 UNIT

Arithmetic and Logic Unit (2 Marks Questions)

2.1. What is look ahead carry adders ?

Ans. A look ahead carry adder is a type of adder which improves the speed by reducing the amount of time required to determine carry bits.

2.2. What is arithmetic and logic circuit ?

Ans. **Arithmetic circuit :** It is a digital circuit which performs only arithmetic operations such as addition, subtraction etc.
Logic circuit : It is a digital circuit which perform only logic operations such as AND, OR and NOT etc.

2.3. Define following terms :
i. RTL **ii. Micro-operation**

AKTU 2016-17, Marks 02

Ans.

i. RTL : Register Transfer Language (RTL) is a convenient tool for describing the internal organization of digital computers in concise and precise manner. It can also be used to facilitate the design process of digital systems.

ii. Micro-operation : The processor unit has to perform a set of operations to execute the major phases of instruction cycle these set of operations called micro-operations.

2.4. What is the main advantage of RTL ?

AKTU 2015-16, Marks 02

Ans. **Advantage of RTL are :**

i. It uses register's as a primitive component in the digital system instead of flip-flops and gates.
ii. It describes the information flow and processing tasks among the data stored in the registers in a concise and precise manner.
iii. It uses a set of expressions and statements which resemble the statements used in programming languages.
iv. The presentation of digital functions in register transfer logic is very user friendly.

2.5. What is the need of having many addressing modes in machine ?

Ans. **Need of having many addressing modes in machine is due to following reason :**

i. The way of operands are chosen during program execution is dependent on the addressing mode of the instruction.

ii. The addressing mode specifies a rule for interpreting or modifying the address field of the instruction before the operand is actually referenced.

2.6. How subtraction operation and other operations can be simplified in a digital system ?

Ans. Subtraction operation and other operations can be simplified in a digital system by using complement method. For each number system, there are two types of complement :

i. r's complement ii. $(r-1)$'s complement

2.7. How many flip-flops are needed for 4-bit decimal code and 4385 in BCD representations ?

Ans. For 4-bit decimal code, there are 4 flip-flops used, each for one bit. For 4385 in BCD representation, there are 16 flip-flops used.

2.8. State the condition for floating-point number to become normalized.

Ans. A floating-point number is said to be normalized if the most significant digit of the mantissa is non-zero.

2.9. When exponent overflow and underflow occur ?

Ans. Exponent overflow occurs when a positive exponent exceeds the maximum possible exponent value.

Exponent underflow occurs when a negative exponent exceeds the maximum possible exponent value. In such cases, the number is designed is zero.

2.10. Perform the following operation on signed numbers using 2's compliment method : $(56)_{10} + (-27)_{10}$.

AKTU 2017-18, Marks 02

Ans.

	$56 =$	111000	(binary form)
	$+56 =$	0111000	(signed binary form)
	$-27 =$	011011	(binary form)
		100100	(1's complement)
		$+1$	
		100101	(2's complement)
	$-27 =$	1100101	(signed binary form)
now, $(+56)_{10} + (-27)_{10} =$		0111000	
		1100101	
		①0011101	
	$=$	$(0011101)_2$	(signed binary number)
	$=$	$(29)_{10}$	

☺☺☺

3 UNIT

Control Unit (2 Marks Questions)

3.1. Explain one, two and three address instruction.

AKTU 2016-17, Marks 02

Ans.

i. **One address instruction :** One address instruction uses an implied accumulator (AC) register for all data manipulation.

ii. **Two address instruction :** In this format each address field can specify either a processor register or a memory word.

iii. **Three address instruction :** Three address instruction formats can use each address fields to specify either a processor register or a memory operand.

3.2. What are the various facts related to operation code ?

Ans. Various facts related to operation code are :

i. The number of bits required for the operation code of an instruction depends on the total number of operations available in the computer.

ii. The operation code must consist of at least n bits for a given 2^n distinct operations.

3.3. Define the necessary factors for instruction sequencing.

Ans. Necessary factors for instruction sequencing are :

i. It needs a counter to calculate the address of next instruction after execution of current instruction is completed.

ii. It is also necessary to provide a register in the control unit for storing the instruction code.

3.4. What operations are included in micro-operations ?

Ans. Micro-operations includes :

i. Transfer a word of data from one CPU register to another or to the ALU.

ii. Perform the arithmetic or logic operations on the data from the CPU registers and store the result in a CPU register.

iii. Fetch a word of data from specified memory location and load them into a CPU register.

iv. Store a word of data from a CPU register into a specified memory location.

3.5. Define the following terms : **AKTU 2016-17, Marks 02**

i. Effective address

ii. Immediate instruction

Ans.

i. Effective address : Effective address is the address of the operand in a computation-type instruction or the target address in a branch-type instruction.

ii. Immediate instruction : An immediate mode instruction has an operand field rather than an address field. The operand field contains the actual operand to be used in conjunction with the operation specified in the instruction.

3.6. What does the processor do when an interrupt is pending ?

Ans. If an interrupt is pending, the processor does the following :

i. It suspends execution of the current program being executed and saves its context.

ii. It sets the program counter to the starting address of an interrupt handler routine.

3.7. Define the goal of CISC architecture.

Ans. The goal of CISC architecture is to provide a single machine instruction for each statement that is written in high-level language.

3.8. Compare horizontal and vertical organization.

Ans.

S. No.	Horizontal organization	Vertical organization
1.	Long formats.	Short formats.
2.	Ability to express a high degree of parallelism.	Limited ability to express parallel micro-operations.
3.	Little encoding of control information.	Considerable encoding of the control information.
4.	Useful when higher operating speed is desired.	Slower operating speed.

3.9. Describe the micro-program sequencing.

Ans. The simple approach of micro-programming is sequential execution of micro-instructions, except for the branch at the end of the fetch phase.

3.10. What is the problem with simple micro-instruction ?

Ans. The micro-program requires several branch micro-instructions. These instructions perform no useful operation in data path. They are needed only to determine the address of next micro-instruction. Thus, they de-tract the operating speed of computer.

3.11. List two important instruction set design issues.

AKTU 2015-16, Marks 02

Ans. **Two important instruction set design issues are :**

i. **Data types :** The various types of data upon which operations are performed.

ii. **Registers :** Number of CPU registers that can be referenced by instructions and their use.

3.12. Define sequencer. **AKTU 2016-17, Marks 02**

Ans. A sequencer generates the addresses used to step through the micro-program of a control store. It is used as a part of control unit of CPU for address ranges.

3.13. List the two techniques used for grouping the control signals. **AKTU 2015-16, Marks 02**

Ans. **The two techniques used for grouping the control signal are :**

i. Hardwired control unit

ii. Micro-programmed control unit

3.14. List three types of control signals.

AKTU 2015-16, 2018-19; Marks 02

Ans. **Three types of control signals are :**

i. ALU

ii. Data paths

iii. System

3.15. Draw the block diagram of micro-program sequencer.

AKTU 2015-16, Marks 02

Ans.

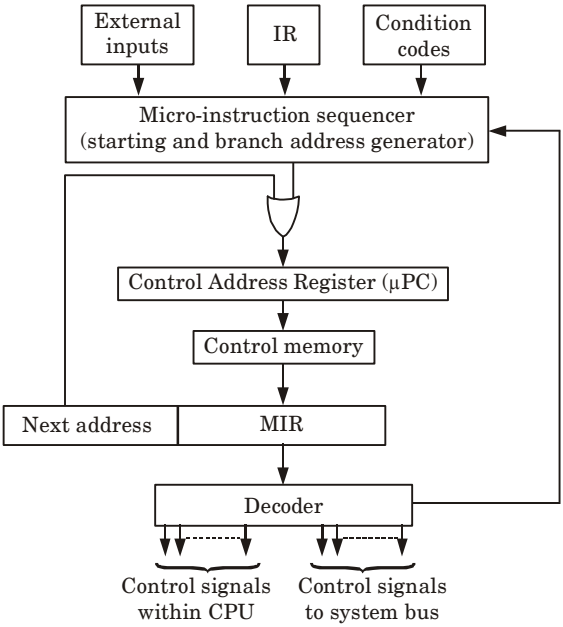


Fig. 3.15.1.

3.16. Write short note on pipelining process.

AKTU 2017-18, Marks 02

Ans. Pipelining means realizing temporal parallelism in an economical way. In this, the problem is divided into a series of tasks that have to be completed one after the other.

3.17. Differentiate between horizontal and vertical microprogramming.

AKTU 2017-18, Marks 02

Ans.

S. No.	Horizontal microprogramming	Vertical microprogramming
1.	In horizontal micro-programming, one associates each bit of the micro-instruction with a specific micro-operation (bit I to represent micro-operation I).	In the case of vertical micro-programming, each line of the micro-program represents a micro-instruction which specifies one or more micro-operations.
2.	A specific micro-operation is executed during a micro-instruction step only if the corresponding bit is one.	One micro-instruction gets executed during each step of the control sequence. One can use a straight binary code to specify each micro-operation.

3.18. What are the difference between horizontal and vertical micro codes ? **AKTU 2018-19, Marks 02**

Ans.

S. No.	Horizontal micro code	Vertical micro code
1.	In this types of code the micro code contains the control signal without any intermediary.	In case of vertical micro code every action is encoded in density.
2.	Horizontal micro code instruction contain a lot of signals and hence due to that the number of bits also increase.	Vertical micro code are slower but they take less space and their actions at execution time need to be decoded to a signal.

4 UNIT

Memory (2 Marks Questions)

4.1. What are the requirements of memory ?

Ans. **There are three requirements of memory which are following :**

i. It should be fast.
ii. It should be large.
iii. It should be inexpensive.

4.2. Differentiate between SRAM and DRAM.

AKTU 2018-19, Marks 02

Ans.

S. No.	Static RAM	Dynamic RAM
1.	Static RAM contains less memory cells per unit area.	Dynamic RAM contains more memory cells as compared to static RAM per unit area.
2.	It has less access time hence faster memories.	Its access time is greater than static RAMs.
3.	Cost is more.	Cost is less.

4.3. What do you mean by programming of ROM ?

Ans. The blowing of fuses in a cell, according to the truth table is called programming of ROM. The PROMs are one time programmable. Once programmed, the information stored is permanent.

4.4. Give the difference between PROM and EEPROM.

Ans.

S. No.	PROM	EEPROM
i.	It is one-time programmable ROM.	It can be programmed more than once.
ii.	PROM destroys the entire data when applied with high voltage.	It allows selective erasing at the register level.

4.5. Why auxiliary storage is organized in records or blocks ?

Ans. Auxiliary storage is organized in records or blocks because the seek time is usually much longer than the transfer time.

4.6. What is CAM ? **AKTU 2015-16, 2017-18; Marks 02**

Ans. A Content Addressable Memory (CAM) is a circuit that combines comparison and storage in a single device. Instead of supplying an address and reading a word like a RAM, we supply the data and the CAM looks to see if it has a copy and returns the index of the matching row.

4.7. Which of L1 and L2 cache is faster ?

AKTU 2015-16, 2017-18; Marks 02

Ans.

1. All processors rely on L1 cache, which is usually located on the processor and is very fast memory and expensive.
2. L2 cache is slower, bigger and cheaper than L1 cache.

4.8. What is cache memory used for ? **AKTU 2016-17, Marks 02**

Ans.

1. Cache memory is used to store frequently used data or instructions.
2. Cache memory is used to improve computer performance by reducing its access time.
3. A cache holds instructions and data that are likely to be needed for the CPU's next operation.

4.9. Give the disadvantage of direct mapping.

Ans. The disadvantage of direct mapping is that the hit ratio can drop considerably if two or more words whose addresses have the same index but different tags are accessed repeatedly.

4.10. Define access time, seek time and latency time.

Ans. **Access time :** The disk access time is the time delay between receiving an address and the beginning of actual data transfer.

Seek time : The seek time is the time required to move the read/write head to the proper track.

Latency time : The rotational delay also known as latency time is the amount of time that elapses after the head is positioned over the correct track until the starting position of the addressed section passes under the read/write head.

4.11. Discuss the advantages of erasable optical disk.

Ans. **Advantages of erasable optical disk are :**

i. Provide high storage capacity about 650 Mbytes of data on 5.25 inch disk.
ii. It is portable and can easily be carried from one computer to another.
iii. It is highly reliable and has longer life.

4.12. State the disadvantages of erasable optical disk.

Ans. **Disadvantages of erasable optical disk are :**

i. It uses constant angular velocity method, therefore lot of storage space is wasted in the outer tracks.
ii. Overwriting data on magneto-optic media is slower than for magnetic media, since one revolution is required to erase a bit and a second is required to write back to that location.

4.13. What is memory management unit ?

Ans. Memory Management Unit (MMU) is the hardware component in the computer that handles virtual memory. Any request for data is sent to the MMU which then determines the location of the information; whether it is in RAM or on a permanent storage drive. The MMU is usually located on our machine's CPU and holds a table for matching physical memory addresses to virtual ones.

4.14. Explain the following terms :

i. PSW

ii. Delayed load **AKTU 2016-17, Marks 02**

Ans.

i. PSW (Program Status Word) :

1. The collection of all status bit conditions in the CPU is sometimes called a program status word or PSW.
2. The PSW is stored in a separate hardware register and contains the status information that characterizes the state of the CPU.
3. It includes the status bits from the last ALU operation and it specifies the interrupts that are allowed to occur and whether the CPU is operating in a supervisor or user mode.

ii. Delayed load :

1. It is up to the compiler to make sure that the instruction following the load instruction uses the data fetched from memory.
2. If the compiler cannot find a useful instruction to put after the load, it inserts a no-op (no-operation) instruction.
3. This is a type of instruction that is fetched from memory but has no operation, thus wasting a clock cycle.
4. This concept of delaying the use of the data loaded from memory is referred to as delayed load.

4.15. What do you understand by locality of reference ?

AKTU 2018-19, Marks 02

Ans. Locality of reference is a term for the phenomenon in which the same values or related storage locations are frequently accessed, depending on the memory access pattern.

4.16. Write the difference between RAM & ROM.

AKTU 2017-18, Marks 02

Ans.

S. No.	RAM	ROM
1.	A random access memory (RAM) device allows data items to be read or written in almost the same amount of time irrespective of the physical location of data inside the memory.	The read only memory (ROM) is a type of semiconductor memory that is designed to hold data that is either permanent or will not change frequently. It is also known as non-volatile memory.
2.	Types of RAM : i. Dynamic random access memory (DRAM) ii. Static random access memory (SRAM)	Types of ROM : i. Programmable read only memory (PROM) ii. Erasable programmable read only memory (EPROM)

5
UNIT

Input/Output (2 Marks Questions)

5.1. Why I/O devices cannot be connected directly to the system bus ?

Ans. I/O devices cannot be connected directly to the system bus for the following reasons :

i. A variety of peripherals with different methods of operation are available. So, it would be impractical to incorporate the necessary logic within the CPU to control a range of devices.

ii. Generally, the peripherals used in a computer system have different data formats and word lengths than that of CPU used in it.

5.2. What is function of I/O interface ?

Ans. Function of I/O interface are :

1. Input-output interface enables transfer of data between internal storage and external I/O devices.
2. In order to interface peripherals with the CPU, I/O interfaces contain special communication links. These communication links are used to overcome, the difference between the CPU and peripheral such as data transfer speed, mode of operation, etc.

5.3. Compare memory and I/O bus.

Ans.

S. No.	Memory bus	I/O bus
1.	Memory bus shares entire address range.	I/O bus shares only I/O address range.
2.	Memory bus width is greater than I/O bus width.	I/O bus width is smaller than memory bus width.
3.	Memory bus includes data bus, address bus and control signals to access memory.	I/O bus includes data bus, address bus and control signals to access I/O.

5.4. State the drawbacks of programmed I/O and interrupt driven I/O.

Ans. **Drawbacks of programmed I/O and interrupt driven I/O :**

i. The time that the CPU spends testing I/O device status and executing a number of instructions for I/O data transfer can often be better spent on other task.

ii. The I/O transfer rate is limited by the speed with which the CPU can test and service a device.

5.5. Compare programmed I/O and interrupt driven I/O.

Ans.

S. No.	Programmed I/O	Interrupt driven I/O
1.	It is implemented without interrupt hardware support.	It is implemented using interrupt hardware support.
2.	It does not depend on interrupt status.	Interrupt must be enabled to process interrupt driven I/O.
3.	It does not need initialization of stack.	It needs initialization of stack.

5.6. Give comparison between I/O program controlled transfer and DMA transfer.

Ans.

S. No.	I/O program controlled transfer	DMA transfer
1.	Software controlled data transfer.	Hardware controlled data transfer.
2.	Data transfer speed is low.	Data transfer speed is high.
3.	CPU is involved in the transfer.	CPU is not involved in the transfer.
4.	Extra hardware is not required.	DMA controller is required to carry-out data transfer.

5.7. State the characteristics of I/O channel.

Ans.

i. An I/O channel has a special-purpose processor.

ii. The I/O instructions are stored in main memory.

iii. The I/O program specifies the devices, the area of memory storage, priority and actions to be taken for certain error conditions.

5.8. Explain the type of I/O channels.

Ans. **There are two main types of I/O channels :**

i. Selector channel

ii. Multiplexer channel

5.9. What are the modes of data transfer ?

AKTU 2016-17, Marks 02

Ans. **Modes of data transfer are :**

i. **Programmed I/O :** Programmed I/O operations are the result of I/O instructions written in computer program.

ii. **Interrupt-driven I/O :** This mode avoids the drawbacks of programmed I/O by using interrupt request.

iii. **Direct memory access :** The interface transfers data onto and out of the memory unit through memory bus.

5.10. What is an interrupt ? **AKTU 2016-17, Marks 02**

Ans.

1. An interrupt is a signal sent by an I/O interface to the CPU when it is ready to send information to the memory or receive information from the memory.
2. When a CPU receives an interrupt signal it stops executing current normal program.
3. After stopping it saves the state of various registers in stack.
4. When this is done CPU executes a subroutine in order to perform the specific task requested by the interrupt.

5.11. Differentiate between synchronous and asynchronous transmission. **AKTU 2016-17, Marks 02**

Ans.

S. No.	Synchronous serial communication	Asynchronous serial communication
1.	Transmitter and receivers are synchronized by clock.	Transmitter and receivers are not synchronized by clock.
2.	Data bits are transmitted with synchronization of clock.	Data bits are transmitted at constant rate.
3.	Data transfer takes place in blocks.	Data transfer is character-oriented.

5.12. Name the different types of I/O bus.

Ans. There are four types of I/O bus which are :

i. Control command ii. Status command

iii. Output data command iv. Input data command

5.13. Why are read and write control lines in a DMA controller bidirectional ? **AKTU 2015-16, Marks 02**

Ans. Read and write control lines in a DMA controller are bidirectional so that :

1. When a BG input is 0, the CPU can communicate with the DMA registers through the data bus to read from or write to the DMA registers.

2. When BG is 1, the CPU has relinquished the buses and the DMA can communicate directly with the memory by specifying an address in the address bus and activating RD or WR.

5.14. What is the use of modem in synchronous communication ? **AKTU 2015-16, Marks 02**

Ans. A modem converts digital signals into audio tones to be transmitted over telephone lines and also converts audio tones from the line to digital signals for machine use. The modems used in synchronous transmission have internal clocks that are set to the frequency that bits are being transmitted in the communication line.

5.15. Describe cycle stealing in DMA.

AKTU 2018-19, Marks 02

Ans.

1. In Direct Memory Access (DMA), cycle stealing is a method of allowing I/O controllers to read or write RAM without interfering with the CPU.
2. DMA controllers can operate in cycle stealing mode in which controller take over the bus for each byte of data to be transferred and then return control to the CPU.

5.16. Write the difference between serial and parallel communication. **AKTU 2017-18, Marks 02**

Ans.

Basis for comparison	Serial communication	Parallel communication
Meaning	Data flows in bi-direction, bit by bit	Multiple lines are used to send data *i.e.*, 8 bits or 1 byte at a time
Cost	Economical	Expensive
Bits transferred at 1 clock pulse	1 bit	8 bits or 1 byte
Speed	Slow	Fast
Applications	Used for long distance communication. For example, computer to computer.	Short distance. For example, computer to printer.

B. Tech.

(SEM. IV) EVEN SEMESTER THEORY EXAMINATION, 2014-15

COMPUTER ORGANIZATION

Time : 3 Hours **Total Marks : 100**

SECTION – A

Note : **1.** Attempt **all** questions.
2. Make suitable assumptions wherever necessary.

1. Attempt any **two** parts of the following : **(10 × 2 = 20)**

a. What is a multiplexer and demultiplexer ? Explain how an 8 × 1 multiplexer can designed using two 4 × 1 multiplexers.

b.

i. Simplify the following function using *K*-map and draw the circuit using AND, OR, NOT gates.
$F(A, B, C, D) = s(0, 2, 8, 9, 10, 11, 13, 15)$

ii. Add – 35 and – 31 in binary using 8-bit registers, in signed 1's complement and signed 2's complement.

c. Show step by step the multiplication process using booth's algorithm when (+ 15) and (– 13) numbers are multiplied. Assume 5 – bit registers that hold signed numbers.

2. Attempt any **two** parts of the following : **(10 × 2 = 20)**

a. What is an instruction in the context of computer organization ? Explain the purpose of the various elements of an instruction with the help of a sample instruction format.

b. Explain the following addressing modes with the help of an example each :

i. Direct **ii. Register indirect**
iii. Implied **iv. Immediate**
v. Indexed

c. Write the steps in fetching a word from memory. Differentiate between a branch instruction and call subroutine instruction.

3. Attempt any **two** parts of the following : **(10 × 2 = 20)**

a. Compare and contrast hardwired and micro programmed control units. Also lists their advantages and disadvantages.

b. What are the different categories of micro-operations that may be carried out by CPU ? Explain each category of micro-operations giving one example for each.

c. Write short notes on the following :

i. Microprogram sequencer for control memory.

ii. RISC.

4. Attempt any **two** parts of the following : **(10 × 2 = 20)**

a. What is the difference between isolated I/O and memory mapped I/O ? Explain the advantages and disadvantages of each.

b. Consider a cache uses a direct mapping scheme. The size of main memory is 4K bytes and word size of cache is 2 bytes. The size of cache memory is 128 bytes. Find the following :

i. The size of main memory address (assume each byte of main memory has an address)

ii. Address of cache block

iii. How many memory location address will be translated to cache address/block/location ?

iv. How can it be determined if the content of specified main memory address in cache.

c. Explain the following memory schemes discussing why needed the :

i. Interleaved memory ii. Associative memory

5. Write short notes on any **four** of the following : **(5 × 4 = 20)**

a. Interrupt **b. Bus arbitration**

c. Virtual memory **d. Organization of 2D and 2.5D**

e. Programmed I/O **f. DMA**

SOLUTION OF PAPER (2014-15)

SECTION – A

Note : **1.** Attempt **all** questions.
2. Make suitable assumptions wherever necessary.

1. Attempt any **two** parts of the following : **(10 × 2 = 20)**

a. What is a multiplexer and demultiplexer ? Explain how an 8 × 1 multiplexer can designed using two 4 × 1 multiplexers.

Ans. This question is out of syllabus since session 2017-18.

b.

i. Simplify the following function using *K*-map and draw the circuit using AND, OR, NOT gates.
$F(A, B, C, D) = s(0, 2, 8, 9, 10, 11, 13, 15)$

Ans. This question is out of syllabus since session 2017-18.

ii. Add – 35 and – 31 in binary using 8-bit registers, in signed 1's complement and signed 2's complement.

Ans.

sign bit ↓

True binary number of 35 = 0 0 1 0 0 0 1 1
True binary number of 31 = 0 0 0 1 1 1 1 1
1's complement of – 35 = 1 1 0 1 1 1 0 0
1's complement of – 31 = + 1 1 1 0 0 0 0 0
1 1 0 1 1 1 1 0 0

↑ Discard the carry

2's Complement of – 35 = 1 1 0 1 1 1 0 0
+ 1
1 1 0 1 1 1 0 1

2's Complement of – 31 = 1 1 1 0 0 0 0 0
+ 1
1 1 1 0 0 0 0 1

Adding 2's complement of – 35 and – 31

			1	1	0	1	1	1	0	1
+		1	1	1	0	0	0	0	1	
1		1	0	1	1	1	1	1	0	

↑ Discard the carry ↑ sign bit

c. Show step by step the multiplication process using booth's algorithm when (+ 15) and (– 13) numbers are multiplied. Assume 5 – bit registers that hold signed numbers.

Ans. 15 = 0 1 1 1 1
– 13 = 2's complement of 13 = 1 0 0 1 1
Multiplicand (M) = 0 1 1 1 1, Multiplier = 1 0 0 1 1

A	Q_n	Q_{n+1}	Operation	SC
00000	10011	0		101 (5)
10001	10011	0	A←A – M	
11000	11001	1	Shift	100 (4)
11100	01100	1	Shift	011 (3)
01011	01100	1	A←A + M	
00101	10110	0	Shift	010 (2)
00010	11011	0	Shift	001 (1)
10011	11011	0	A←A – M	
11001	11101	1	Shift	000 (0)
Result = (11001	11101) = – 195(2's complement of +195)			

2. Attempt any **two** parts of the following : **(10 × 2 = 20)**

a. What is an instruction in the context of computer organization ? Explain the purpose of the various elements of an instruction with the help of a sample instruction format.

Ans. **Instruction :**

1. Instruction is a command to the processor to perform a given task on specified data.
2. An instruction is a designed binary pattern which is based on the architecture of CPU to perform a specific function. The entire group of instructions is called the instruction set.

Instruction format :

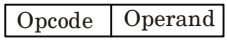


Fig. 1.

Instruction has two parts opcode and operand,

1. Task to be performed, called the operation code (Opcode), and the data to be operated upon, called the operand.
2. The operands include the input data of the operation and the results that are produced.
3. A computer must have instructions capable of performing four types of operations :

a. Data transfers between the memory and the CPU registers.
b. Arithmetic and logic operations on data.
c. Program sequencing and control.
d. I/O transfers.

4. The purpose of an instruction is to specify both an operation to be carried out by a CPU or also process the set of operands or data to be used in the operation.

Example :

This instruction will multiply two operand *A* and *B* and result is stored in *A*.

b. Explain the following addressing modes with the help of an example each :

i. Direct **ii. Register indirect**
iii. Implied **iv. Immediate**
v. Indexed

Ans.

i. Direct :

1. A very simple form of addressing is direct addressing, in which the address field contain the effective address of the operand : $EA = A$
where, EA = Actual (effective) address of the location containing the referenced operand.
A = Contents of the address field in the instruction.

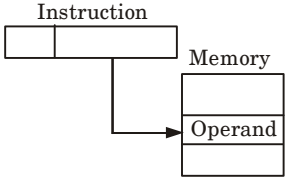


Fig. 2. Direct.

2. A direct address in instruction needs two reference to memory :

a. Read instruction b. Read operand

ii. Register indirect mode :

1. Register indirect mode is similar to indirect addressing.
2. The only difference is whether the address field refers to a memory location or a register.
3. Thus, for register indirect address, $EA = (R)$

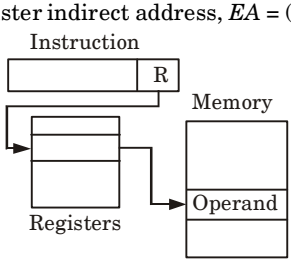


Fig. 3. Register indirect.

iii. Implied mode :

1. In this mode, the operands are specified implicitly in the definition of the instruction.
2. All register reference instructions that use an accumulator are implied mode instructions.
3. Zero address instructions in a stack-organized computer are implied mode instruction since the operands are implied to be on top of the stack. It is also known as stack addressing mode.

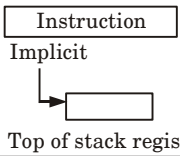


Fig. 4. Implied mode.

iv. Immediate mode :

1. In this mode, the operand is specified in the instruction itself.
2. The operand field contains the actual operand to be used in conjunction with the operation specified in the instruction.

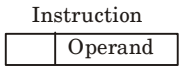


Fig. 5. Immediate mode.

v. Indexed :

1. The effective address of the operand is generated by adding a constant value to the contents of a register.
2. The register used may be either a special register for this purpose or more commonly, it may be any one of a set of general purpose registers in the CPU.
3. It is referred to as an index register. We indicate the index mode symbolically as, $X(R)$
   where X denotes a constant and R is the name of register involved.
4. The effective address of the operand is given by, $EA = X + [R]$
5. In the process of generating the effective address, the contents of the index register are not changed.

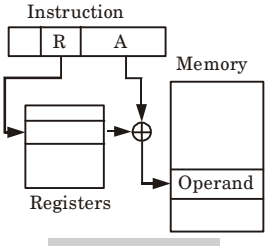


Fig. 6. Indexed.

Example :

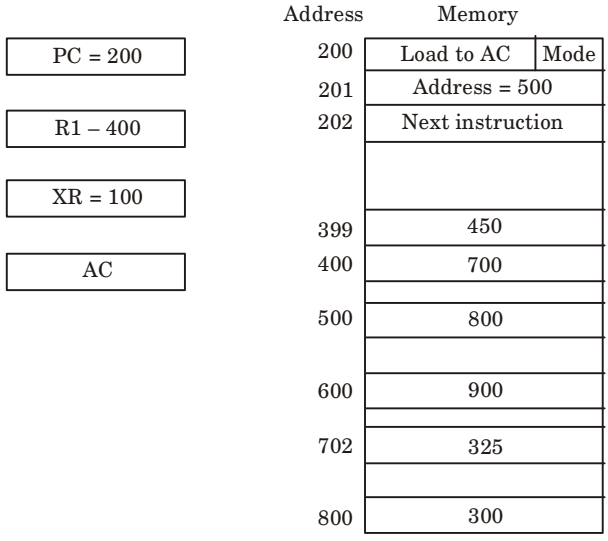


Fig. 7.

Addressing Mode	Effective Address	Content of AC
Direct address	500	800
Immediate operand	201	500
Indirect address	800	300
Indexed address	600	900
Implied	–	400
Register indirect	400	700

c. Write the steps in fetching a word from memory. Differentiate between a branch instruction and call subroutine instruction.

Ans. Steps in fetching a word from memory are as follows :

Step 1 : The CPU has to perform opcode fetch cycle and operand fetch cycle.

Step 2 : The opcode fetch cycle gives the operation code of fetching a word from memory to the CPU.

Step 3 : The CPU then invokes the operand fetch cycle.

Step 4 : The opcode specifies the address of memory location where the information is stored.

Step 5 : The CPU transfers the address of required word of information to the Address Register (AR), which is connected to the address lines of the memory bus. Hence, the address is transferred to the memory.

Step 6 : The CPU activates the read signal of the memory to indicate that a read operation is needed.

Step 7 : As a result, memory copies data from the addressed register on the data bus.

Step 8 : The CPU then reads this data from the data register and loads it in the specified register.

Step 9 : Memory Functions Completed (MFC) is also used as a control signal for this memory transfer.

Step10 : Memory sets MFC to 1 to indicate that the contents of the specified location have been read and are available on the data bus.

Difference :

S. No.	Branch instruction	Subroutine instruction
1.	Branch instruction is a machine-language or assembly-language instruction.	Subroutine is a control transfer instruction.
2.	It is used to change the sequence of instruction	It is used to call a subroutine.

3. Attempt any **two** parts of the following : **(10 × 2 = 20)**

a. Compare and contrast hardwired and micro programmed control units. Also lists their advantages and disadvantages.

Ans. Comparison between hardwired control and micro-programmed control :

S. No.	Characteristics	Hardwired control	Micro-programmed control
1.	Speed	Fast	Slow
2.	Implementation	Hardware	Software
3.	Flexibility	Not flexible	Flexible
4.	Ability to handle large/ complex instruction set	Somewhat difficult	Easier

5.	Ability to support operating system and diagnostic features	Very difficult	Easy
6.	Design process	Difficult for more operation	Easy
7.	Memory	Not used	Control memory used (RAM or ROM)
8.	Chip area efficiency	Uses less area	Uses more area
9.	Used in	RISC processor	CISC processor
10.	Output generation	On the basis of input signal	On the basis of control line.

Advantage of hardwired control unit :

1. Speed is high.

Disadvantages of hardwired control unit :

1. Expensive to implement.
2. More error prone.
3. Contain complex logic.

Advantages of micro-programmed control unit :

1. Cheaper to implement.
2. Less error prone.
3. Contain very simple piece of logic.

Disadvantage of micro-programmed control unit :

1. Speed is slow.

b. What are the different categories of micro-operations that may be carried out by CPU ? Explain each category of micro-operations giving one example for each.

Ans. Different categories of micro-operation are :

i. Register transfer micro-operation :

1. Register transfer is defined as information transfer from one register to another and is designated in symbolic form by means of a replacement operator.
2. The statement denotes a transfer of the content of register $R1$ into register $R2$.

$$R2 \leftarrow R1$$

3. It designates a replacement of the content of $R2$ by the content of $R1$. By definition, the content of the source register $R1$ does not change after the transfer.
4. Every statement written in a register transfer notation implies a hardware construction for implementing the transfer.

5. Fig. 1 shows the block diagram that depicts the transfer from $R1$ to $R2$. The n outputs of register $R1$ are connected to the n inputs of register $R2$.

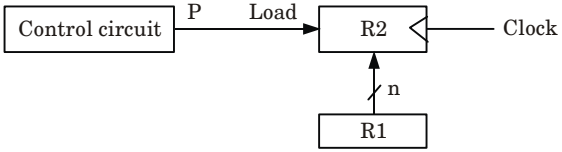


Fig. 8. Block diagram.

6. The letter n will be used to indicate any number of bits for the register. It will be replaced by an actual number when the length of the register is known.
7. Register $R2$ has a load input that is activated by the control variable P.

ii. Arithmetic micro-operation :

1. The basic arithmetic micro-operations are addition, subtraction, increment and decrement. The arithmetic micro-operation defined by the statement, $R3 \leftarrow R1 + R2$ which specifies an addition micro-operation.
2. It states that the contents of register $R1$ are added to the contents of register $R2$ and the sum is transferred to register $R3$.
3. Subtraction is implemented through complementation and addition, which is specified in following statement :

$$R3 \leftarrow R1 + \overline{R2} + 1$$

$\overline{R2}$ is the symbol for the 1's complement of $R2$. Adding 1 to the 1's complement produce the 2's complement.

4. Adding the contents of $R1$ to the 2's complement of $R2$ is equivalent to $R1 - R2$.

iii. Logic micro-operation :

1. Logic micro-operations specify binary operations for strings of bits stored in registers.
2. These operations consider each bit of the register separately and treat the contents of two registers $R1$ and $R2$ symbolized by the statement

$$P : R1 \leftarrow R1 \oplus R2$$

3. It specifies a logic micro-operation is to be executed on the individual bits of the registers provided that the control variable $P = 1$.
4. For example, the content of $R1$ is 1010 and content of $R2$ is 1100. The logic computation :

1 0 1 0	content of $R1$
1 1 0 0	content of $R2$
0 1 1 0	content of $R1$ after $P = 1$

5. There are 16 different logic operations that can be performed with two binary variables.

iv. Shift micro-operation :

1. Shift micro-operations in computer architecture are those which are used in serial shifting of data present in a register.
2. Shift micro-operations move or shift data in a register bitwise that is, one bit at a time either left or right from its original position.

List of different types of shift micro-operation :

a. Arithmetic shift micro-operation
b. Logical shift micro-operation
c. Circular shift micro-operation

c. Write short notes on the following :

i. Microprogram sequencer for control memory.

Ans.

1. Micro-program sequencer is a general purpose building block for micro-programmed control unit.
2. The basic components of a micro-programmed control unit are the control memory and the circuit that selects the next address.
3. The address selection part is called micro-program sequencer.
4. The main purpose of micro-program sequencer is to present an address to the control memory so that micro-instruction may be read and executed.
5. The next address logic of the sequencer determines the specific address source to be loaded into the control address register.
6. The choice of the address source is guided by the next address information bits that sequencer receives from the present micro-instruction.
7. All the instructions are loaded in the control memory.

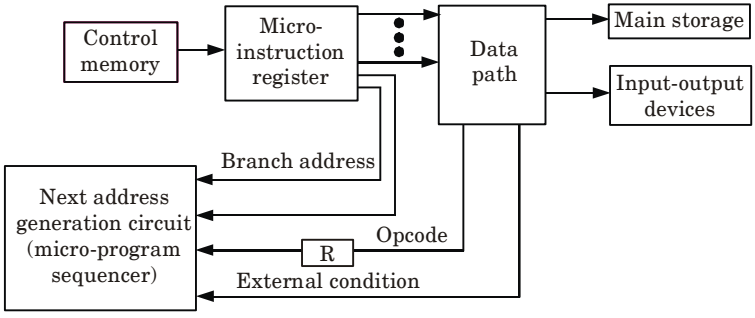


Fig. 9. Block diagram of micro-programmed control with micro-program sequencer.

8. The present micro-instruction is placed in micro-instruction register for execution.

ii. RISC.

Ans.

1. RISC (Reduced Instruction Set Computer) processor instruction has a fixed length encoding of instruction and each instruction executes in a single clock cycle by hardwired implementation of each instruction.
2. RISC architecture focus on reducing the number of instructions and working with simpler instruction set having limited number of addressing modes and allowing them to execute more instructions in the same amount of time.
3. Programs written for RISC architectures tend to make more space in memory but RISC processor's increased clock rate allows it to execute its program in less time than a CISC processor takes to execute its program.

RISC characteristics :

1. Simple instructions are used in RISC architecture.
2. RISC helps and supports few simple data types and synthesizes complex data types.
3. RISC utilizes simple addressing modes and fixed length instructions for pipelining.
4. RISC permits any register to use in any context.

4. Attempt any **two** parts of the following : **(10 × 2 = 20)**

a. What is the difference between isolated I/O and memory mapped I/O ? Explain the advantages and disadvantages of each.

Ans. **Difference between isolated I/O and memory mapped I/O :**

S. No.	Isolated I/O	Memory mapped I/O
1.	Isolated I/O uses separate memory space.	Memory mapped I/O uses memory from the main memory.
2.	Limited instructions can be used. Those are IN, OUT, INS, OUTS.	Any instruction which references to memory can be used.
3.	The addresses for isolated I/O devices are called ports.	Memory mapped I/O devices are treated as memory locations on the memory map.
4.	Efficient I/O operations due to separate bus.	Inefficient I/O operations due to single bus for data and addressing.
5.	Comparatively larger in size.	Smaller in size.
6.	Uses complex internal logic.	Common internal logic for memory and I/O devices.
7.	Slower operations.	Faster operations.

Advantage of isolated I/O :

1. The devices of I/O are treated in a separate domain as compared to memory.
2. A total of 1MB address space is allowed for memory applications.
3. In order to maximize the I/O operations (isolated) separate instructions are always provided to perform these operations.

Disadvantage of isolated I/O :

1. The data transfer only occurs between the I/O port and the registers.

Advantages of memory mapped I/O :

1. I/O intensive operation is fast.
2. The SQLite library needs less RAM.

Disadvantages of memory mapped I/O :

1. If an I/O error on a memory-mapped file cannot be caught by the application, results in a program crash.
2. Performance is reduced by the use of memory-mapped I/O.

b. Consider a cache uses a direct mapping scheme. The size of main memory is 4K bytes and word size of cache is 2 bytes. The size of cache memory is 128 bytes. Find the following :

i. The size of main memory address (assume each byte of main memory has an address)

ii. Address of cache block

iii. How many memory location address will be translated to cache address/block/location ?

iv. How can it be determined if the content of specified main memory address in cache.

Ans. Given,Size of main memory = 4 K bytes
Size of cache memory = 128 bytes
Word size of cache = 2 bytes

i. Size of main memory address :
Since, size of main memory = 4 K bytes = 2^{12} bytes
So, the size of main memory address = 12

ii. Address of cache block :
Since, size of cache memory = 128 bytes
Block size = word size of cache = 2 bytes

$$\therefore \quad \text{Number of lines} = \frac{128}{2} = 64$$

So, address of cache block is from 0 to 63

iii. Since, size of main memory = 4 K bytes = 4096 bytes
So, number of block in main memory = 4096
and number of block in cache = 64
and each block of cache = 2 bytes

So, $\left(\frac{4096}{(64 \times 2)}\right)$ = 32 memory location address will be translated to cache address/block/location.

iv. We can determine the content of specified main memory address in cache using this :

$$i \bmod 2^k$$

where, i = particular memory address
k = line number

c. Explain the following memory schemes discussing why needed the :

i. Interleaved memory ii. Associative memory

Ans.

i. Interleaved memory :

1. Interleaved memory is a design made to compensate for the relatively slow speed of Dynamic Random Access Memory (DRAM).
2. This is done by spreading memory addresses evenly across memory banks.
3. Thus, in contiguous memory, reads and writes are done using each memory bank in turn, resulting in higher memory throughputs due to reduced waiting for memory banks to become ready for desired operations.
4. As shown in Fig. 10, the lower order k bits of the address are used to select the module (Memory bank) and higher order m bits give a unique memory location in the memory bank that is selected by the lower order k bits.
5. Thus in this way consecutive memory locations are stored on different memory banks.
6. Whenever requests to access consecutive memory locations are being made several memory banks are kept busy at any point in time.
7. This results in faster access to a block of data in the memory and also results in higher overall utilization of the memory system as a whole.

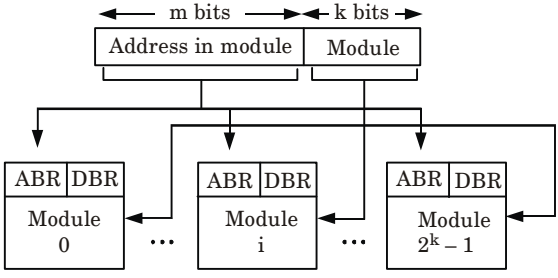


Fig. 10.

8. If k bits are allotted for selecting the bank as shown in the Fig. 10, there have to be total $2k$ banks. This ensures that there are no gaps of non-existent memory locations.

ii. Associative memory :

1. Associative memory is a memory in which location is accessed by a field of data word stored in the memory rather than by any address.
2. It can be viewed as a random access type memory which in addition to having a physically wired-in addressing mechanism also has wired-in logic for bit comparison.
3. This logic circuit enables comparison of desired bit positions of all the words with a specified input key.
4. This comparison is done simultaneously for all the words.
5. This is also called Content Addressable Memory (CAM).

5. Write short notes on any **four** of the following : **(5 × 4 = 20)**

a. Interrupt **b. Bus arbitration**
c. Virtual memory **d. Organization of 2D and 2.5D**
e. Programmed I/O **f. DMA**

Ans.

a. Interrupt :

Interrupt : An interrupt is a signal sent by an I/O interface to the CPU when it is ready to send information to the memory or receive information from the memory.

Identifying the source of an interrupt :

Two methods are available to determine the interrupting device :

i. Polled interrupts :

1. On receiving an interrupt request the micro-processor will execute a routine causing it to poll each of the devices in turn.
2. Devices have a status register containing one or more interrupt request bits.
3. If a device caused the interrupt, its interrupt flag bit will be set.
4. The appropriate service routine will then be selected and the device serviced.
5. Polling can be very inefficient if there are many devices capable of causing an interrupt.
6. This method uses only software methods to identify an interrupting device.

ii. Vectored interrupts :

1. This is the method used in modern computers.
2. It is sometimes referred to as hardware identification since additional hardware is required.
3. Each vector is identified by an interrupt number from 0 to 255 and provides the processor with the means of addressing the appropriate interrupt handler.
4. A vector address is determined by multiplying its vector number by 4.

b. Bus arbitration :

1. Bus arbitration is a mechanism which decides the selection of current master to access bus.

2. Among several masters and slave units that are connected to a shared bus, it may happen that more than one master or slave units will request access to the bus at the same time.
3. In such situation, bus access is given to the master having highest priority.
4. Three different mechanisms are commonly used for this :

i. Daisy chaining :

a. Daisy chaining method is cheaper and simple method.
b. All master make use of the same line for bus request.
c. The bus grant signal serially propagates through each master until it encounters the first one that is requesting.

ii. Parallel arbitration : The parallel arbitration consists of priority encoder and a decoder. In this mechanism, each bus arbiter has a bus request output line and input line.

iii. Independent priority : In this each master has separate pair of bus request and bus grant lines and each pair has a priority assigned to it.

c. Virtual memory :

1. Virtual memory is a memory management capability of an OS that uses hardware and software to allow a computer to compensate for physical memory shortages by temporarily transferring data from Random Access Memory (RAM) to disk storage.
2. Virtual address space is increased using active memory in RAM and inactive memory in Hard Disk Drives (HDDs) to form contiguous addresses that hold both the application and its data.
3. A system using virtual memory can load larger programs or multiple programs running at the same time, allowing each one to operate as if it has infinite memory and without having to purchase more RAM.
4. Virtual memory is a facility that allows program to address memory from local point of view, without regard to the amount of main memory.
5. A virtual memory system provides a mechanism for translating program generated addresses into correct main memory locations.
6. This is done dynamically, while programs are being executed in the CPU.

d. Organization of 2D and 2.5D :

2D organization :

1. The cells are organized in the form of a two-dimensional array with rows and columns.
2. Each row refers to word line. For 4-bit per word memory, 4 cells are interconnected to a word line. Each column in the array refers to a bit line.
3. The Memory Address Register (MAR) holds the address of the location where read/write operation is executed. In Fig. 11, MAR has 4-bit lines.

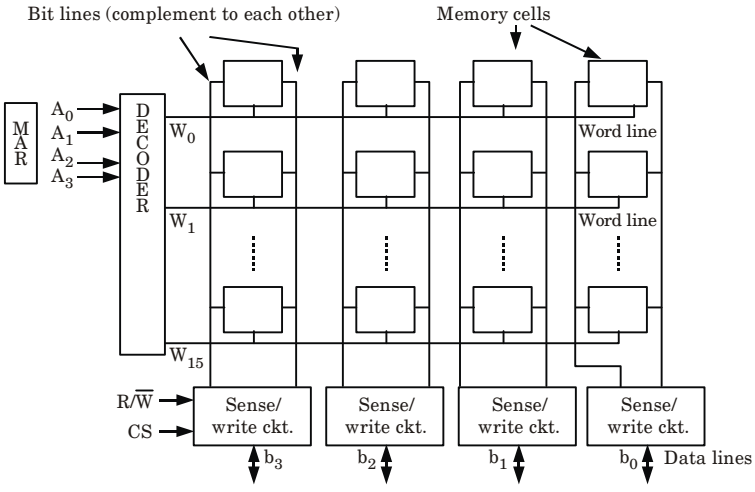


Fig. 11. 2D organization of a memory chip of size 16 × 4.

4. The content of MAR is decoded by an address decoder on the chip to activate each word line.
5. The cells in each column are connected to a sense/write circuit by two bit lines. Two bit lines are complement to each other.
6. The sense/write circuits are activated by the Chip Select (CS) lines. The sense/write circuits are connected to the data lines of the chip.
7. During a read operation, these circuits sense or read the information stored in the cells selected by a word line and transmit this information to the data lines.
8. During a write operation, the sense/write circuits receive or write input information from the data lines and store it in the selected cells.

2.5D organization :

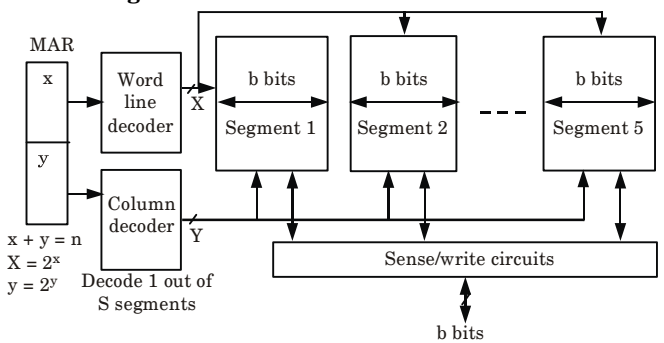


Fig. 12. 2.5D organization.

1. In 2.5D organization there exists a segment.
2. The content of MAR is divided into two parts–x and y number of bits.
3. The number of segments S is equal to 2^y.
4. $X = 2^x$ drive lines are fed into the cell array and y number of bits decode one bit line out of S lines fed into a segment of the array. In total, there are Sb number of bit lines for a b bit per word memory.
5. Thus for any given address in the MAR, the column decoder decodes b out of Sb bit lines by using the y bits of the MAR while a particular word line is activated by using the x bits.
6. Thus only the b numbers of bits in the array are accessed by enabling the word line and b number of bit lines simultaneously.
7. Though 2.5D organized memory may need lesser chip decoding logic, it suffers from one drawback. With high density chips, a simple failure, such as external pin connection opening or a failure on one bit can render the entire chip inoperative.

e. Programmed I/O :

1. When the CPU is executing a program and executes an instruction relating to I/O, it executes that instruction by issuing a command to the appropriate I/O module.

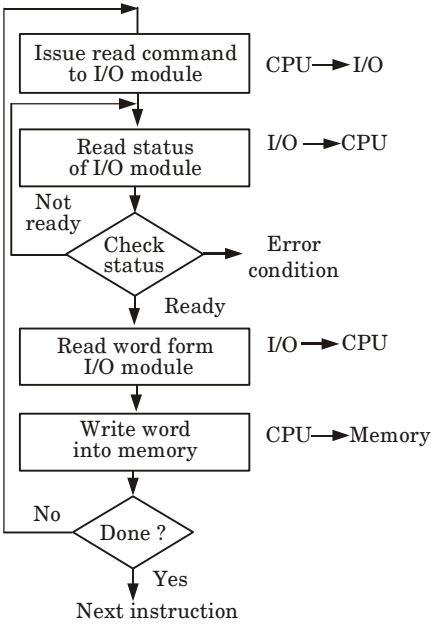


Fig. 13. Programmed I/O.

2. With programmed I/O, the I/O module will perform the requested action and then set the appropriate bits in the I/O status register.
3. The I/O module takes no further action to alert the CPU.
4. It does not interrupt the CPU.
5. Thus, it is the responsibility of CPU to periodically check the status of the I/O module until it finds that the operation is complete.

f. DMA :

1. DMA stands for "Direct Memory Access" and is a method of transferring data from the computer's RAM to another part of the computer without processing it using the CPU.
2. While most data that is input or output from our computer is processed by the CPU, some data does not require processing, or can be processed by another device.
3. DMA can save processing time and is a more efficient way to move data from the computer's memory to other devices.
4. In order for devices to use direct memory access, they must be assigned to a DMA channel. Each type of port on a computer has a set of DMA channels that can be assigned to each connected device.
5. For example, a PCI controller and a hard drive controller each have their own set of DMA channels.

Block diagram for DMA :

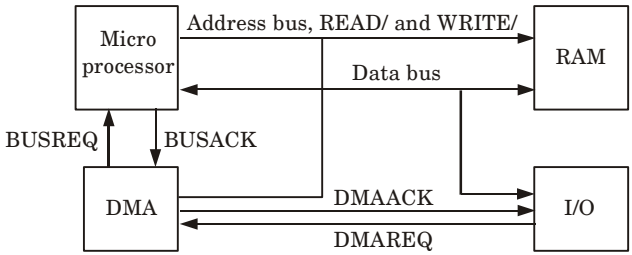


Fig. 14.

☺☺☺

B.Tech.

(SEM. V) ODD SEMESTER THEORY EXAMINATION, 2015-16

COMPUTER ARCHITECTURE

Time : 3 Hours **Total Marks : 100**

SECTION – A

Note : Attempt all parts. All parts carry equal marks. Write answer of each part in short. **(2 × 10 = 20)**

1. a. **What is the main advantage of RTL ?**

   b. **Define control word.**

   c. **Give block diagram of microprogram sequencer.**

   d. **Why are read and write control lines in a DMA controller bidirectional ?**

   e. **List two important instruction set design issues.**

   f. **List the two techniques used for grouping the control signals.**

   g. **Which of L1 and L2 cache is faster ?**

   h. **What is the use of modem in synchronous communication ?**

   i. **What is CAM ?**

   j. **List three types of control signals.**

SECTION – B

Note : Attempt any **five** questions from this section. **(10 × 5 = 50)**

2. **Discuss the advantages and disadvantages of polling and daisy chaining bus arbitration schemes.**

3. **Briefly define the following terms :**
   i. **Micro operation**
   ii. **Micro instruction**
   iii. **Micro program**
   iv. **Micro code**

v. **Control memory**

4. **What do you mean by CAM ? Explain its major characteristics.**

5. **Explain various types of processor organization.**

6. **Explain the sequence that takes place when an interrupt occurs.**

7. **A computer uses RAM chips of 1024*1 capacity.**
   i. **How many chips are needed and how should their address lines be connected to provide a memory capacity of 1024*8 ?**
   ii. **How many chips are needed to provide a memory capacity of 16 KB ? Explain in words how the chips are to be connected to the address bus.**

8. **A ROM chip of 1024*8 has four select inputs and operates from a 5 volt power supply. How many pins are needed for the IC package ? Draw a block diagram and label all input and output terminals in the ROM.**

9. i. **What are the differences between hardwired and microprogrammed control unit ?**

   ii. **What is RISC ? Explain its various characteristics.**

SECTION-C

Note : Attempt any **two** questions from this section. **(15 × 2 = 30)**

10. i. **What is the distinction between spatial locality and temporal locality ?**

    ii. **Show the multiplication process using Booth's algorithm when the following numbers are multiplied : (– 13) by (+ 8)**

11. **Why input output interface is required ? Describe in detail.**

12. **Differentiate among :**
    i. **Strobe control and handshaking asynchronous data transfer modes.**
    ii. **Processor and IOP.**
    iii. **Synchronous and asynchronous transmission.**
    iv. **Character-oriented and Bit-oriented protocols.**
    v. **DMA and interrupt initiated I/O techniques.**

☺☺☺

SOLUTION OF PAPER (2015-16)

SECTION – A

Note : Attempt all parts. All parts carry equal marks. Write answer of each part in short. **(2 × 10 = 20)**

1. a. What is the main advantage of RTL ?

Ans. **Advantage of RTL are :**

i. It uses register's as a primitive component in the digital system instead of flip-flops and gates.

ii. It describes the information flow and processing tasks among the data stored in the registers in a concise and precise manner.

iii. It uses a set of expressions and statements which resemble the statements used in programming languages.

iv. The presentation of digital functions in register transfer logic is very user friendly.

b. Define control word.

Ans. Control word is defined as a word whose individual bits represent the various control signals. Therefore each of the control steps in the control sequence of an instruction defines a unique combination of 0s and 1s.

c. Give block diagram of microprogram sequencer.

Ans.

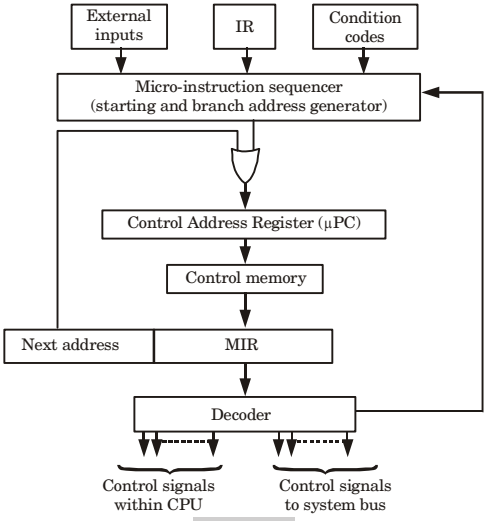


Fig. 1.

d. Why are read and write control lines in a DMA controller bidirectional ?

Ans. Read and write control lines in a DMA controller are bidirectional so that :

1. When a BG input is 0, the CPU can communicate with the DMA registers through the data bus to read from or write to the DMA registers.
2. When BG is 1, the CPU has relinquished the buses and the DMA can communicate directly with the memory by specifying an address in the address bus and activating RD or WR.

e. List two important instruction set design issues.

Ans. **Two important instruction set design issues are :**

1. Number of operations and type of operation to be provided and how must complexity they should support.
2. Data types of operands upon which operations to be performed.

f. List the two techniques used for grouping the control signals.

Ans. **The two techniques used for grouping the control signal are :**

i. Hardwired control unit
ii. Micro-programmed control unit

g. Which of L1 and L2 cache is faster ?

Ans.

1. All processors rely on L1 cache, which is usually located on the processor and is very fast memory and expensive.
2. L2 cache is slower, bigger and cheaper than L1 cache.

h. What is the use of modem in synchronous communication ?

Ans. A modem converts digital signals into audio tones to be transmitted over telephone lines and also converts audio tones from the line to digital signals for machine use. The modems used in synchronous transmission have internal clocks that are set to the frequency that bits are being transmitted in the communication line.

i. What is CAM ?

Ans. A Content Addressable Memory (CAM) is a circuit that combines comparison and storage in a single device. Instead of supplying an address and reading a word like a RAM, we supply the data and the CAM looks to see if it has a copy and returns the index of the matching row.

j. List three types of control signals.

Ans. **Three types of control signals are :**

i. ALU

ii. Data paths
iii. System

SECTION – B

Note : Attempt any **five** questions from this section. **(10 × 5 = 50)**

2. Discuss the advantages and disadvantages of polling and daisy chaining bus arbitration schemes.

Ans. **Daisy chaining :**

1. In this, all masters make use of the same line for bus request.
2. The bus grant signal serially propagates through each master until it encounters the first one that is requesting access to the bus.
3. This master blocks the propagation of the bus grant signal, activates the busy line and gains control of the bus.
4. Therefore any other requesting module will not receive the grant signal and hence cannot get the bus access.

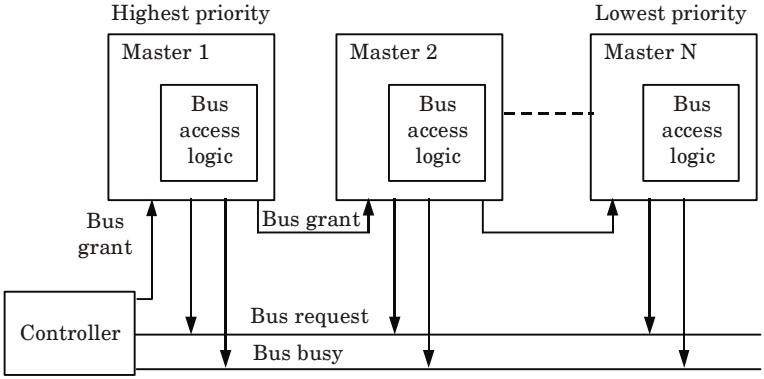


Fig. 2. Daisy chaining method.

Advantages of daisy chaining :

1. It is a simple and cheaper method.
2. It requires the least number of lines and this number is independent of the number of masters in the system.

Disadvantages of daisy chaining :

1. The propagation time delay of bus grant signal is proportional to the number of masters in the system. This makes arbitration time slow and hence limits the number of master in the system.
2. The priority of the master is fixed by the physical location of master.
3. Failure of any one master causes the whole system to fail.

Advantages of polling bus arbitration :

1. If the one module fails entire system does not fail.
2. The priority can be changed by altering the polling sequence stored in the controller.

Disadvantages of polling bus arbitration :

1. It requires more bus request and grant signals ($2 \times n$ signals for n modules).
2. Polling overhead can consume a lot of CPU time.

3. Briefly define the following terms :

i. Micro operation
ii. Micro instruction
iii. Micro program
iv. Micro code
v. Control memory

Ans.

i. Micro-operation :

1. Micro-operation is a set of operations that the processor unit has to perform to execute the major phases of instruction cycle.
2. The instruction cycle has three major phases of fetch, decode and execute.
3. The primary function of a CPU is to execute sequence of instructions which is in accordance with the instruction cycle.

ii. Micro-instructions :

1. Micro-instructions are the individual control words in the micro routine.
2. This contains the control signals for sequencing information.

iii. Micro-program : Micro-program is a micro-code in a particular processor implementation. Writing micro-code is often called micro-programming.

iv. Micro-code :

1. Micro-code is a layer of hardware-level instructions and/or data structures involved in the implementation of higher level machine code instructions in many computers and other processors.
2. It helps to separate the machine instructions from the underlying electronics so that instructions can be designed and altered more freely.

v. Control memory :

1. Control memory is a Random Access Memory (RAM) consisting of addressable storage registers.
2. It is used as a temporary storage for data.
3. Access to control memory data requires less time than to main memory, this speeds up CPU operation.

4. What do you mean by CAM ? Explain its major characteristics.

Ans. CAM :

1. Content Addressable Memory (CAM) is computer memory that operates like a hardware search engine for search-intensive applications.

2. CAM is capable of searching its entire contents in a single clock cycle.
3. It does that by pairing the SRAM-based memory with additional logic comparison circuitry that is active on every clock cycle.
4. The way CAM functions is almost the opposite of Random Access Memory (RAM).
5. Data stored on CAM, can be accessed by searching for the content itself, and the memory retrieves the addresses where that content can be found.
6. Because of its parallel nature, CAM is much faster than RAM for searching.

The major characteristics are :

1. This memory is accessed simultaneously and in parallel on the basis of data content rather than by specific address or location.
2. This memory is capable of finding an empty unused location to store the word.
3. This memory is uniquely suited to do parallel searches by data association.
4. Each cell must have storage capability as well as logic circuits for matching its content with an external argument.

5. Explain various types of processor organization.

Ans. Types of processor organizations :

a. General-purpose register based :

1. In this organization, the registers communicate with each other not only for direct data transfers, but also while performing various micro-operations.
2. Seven registers are used for general purpose, the output of each register is connected to two multiplexer (MUXs) inputs.
3. Three select lines are used to select any one of the seven registers and the contents of selected registers are supplied to the inputs of ALU.
4. The buses *A* and *B* are used to form the inputs to the common arithmetic logic unit (ALU).
5. The operation to be performed is selected in the ALU and is determined by the arithmetic or logic micro-operation by using function select lines (OPR).
6. The result of the micro-operation is available as output data and also goes into the inputs of all the registers.
7. Any one of the destination register receives the information from the output bus which is selected by a decoder.

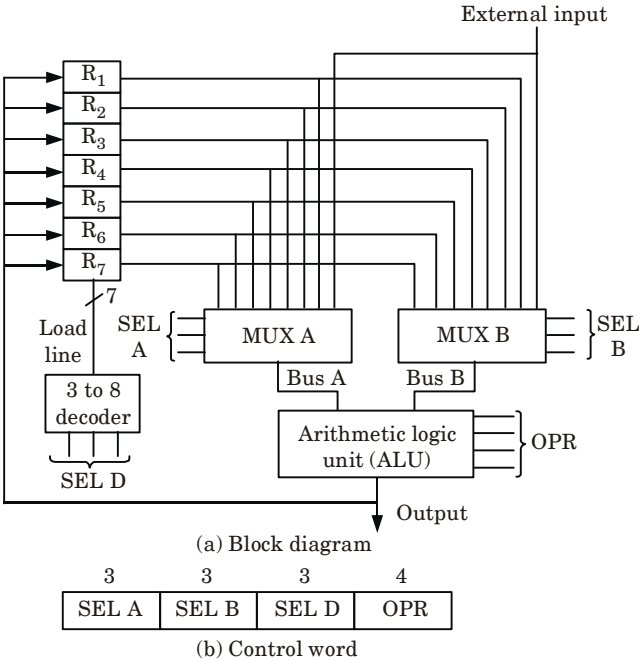


Fig. 3. General-purpose register based organization.

b. Stack based :

1. A stack is an ordered set of elements in which only one element can be accessed at a time.
2. The point of access is called the top of the stack.
3. The number of elements in the stack or length of the stack is variable.
4. Items may only be added or deleted from the top of the stack.
5. A stack is also known as a pushdown list or a Last-In-First-Out (LIFO) list.

Organization of register stack :

Consider the organization of a 64-word register stack as illustrated in Fig. 4.

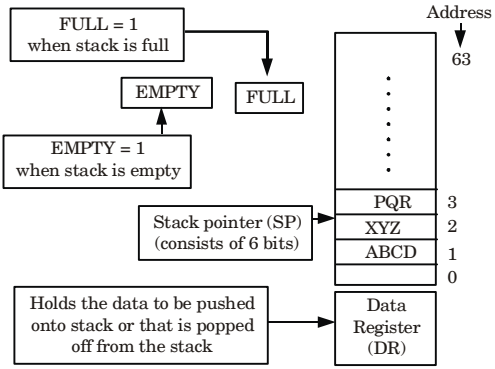


Fig. 4. Block diagram of 64-word stack.

c. Accumulator based :

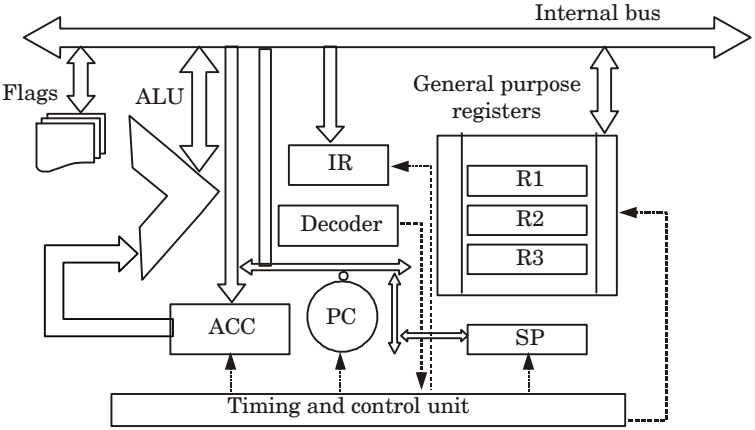


Fig. 5. Block diagram of accumulator based CPU organization.

1. **ALU :** A most generic computer system is composed of a unit to do arithmetic, shift and logical micro-operations commonly known as ALU of CPU.
2. **Program Counter (PC) :** This keeps track of the instruction address in memory from where the next instruction needs to be fetched. The instructions are stored in memory in an order decided by programmer.
3. **General purpose registers (R1, R2, R3) :** It suggests that the registers are involved in operations like load inputs, store intermediate results of arithmetic, logical and shift micro-operations. The initial inputs are loaded into registers from memory and final results are later moved into memory.

4. **Accumulator (ACC) :** This block acts as the default temporary storage register location for all mathematical operations in ALU.
5. **Instruction Register IR and Decoder :** After instruction is fetched from the memory its stored in Instruction Register. The instruction is then decode by the decoder.
6. **Stack pointer (SP) :** Stack pointer is involved in managing the stack transfers during and program execution.
7. **Timing and control unit :** This block manages the sequencing of events on a timeline between various components of a CPU. All the blocks are controlled in a manner to optimize the computational power of the unit by minimizing the failures.
8. **Flags :** Flags are also registers or bits inside registers which are set or cleared for a particular condition on an arithmetic operation. Some of the most common flags are :
   i. **Sign :** Is used to identify the set/reset of most significant bit of the result.
   ii. **Carry :** Is used to identify, a carry during addition, or borrow during subtraction/comparison.
   iii. **Parity :** Set if the parity is even. Refer parity from here.
   iv. **Zero :** To identify when the result is equal to zero.
9. **Bus sub-system :** All the data transfers in-between memory and CPU registers including instruction fetches are carried over bus.

6. Explain the sequence that takes place when an interrupt occurs.

Ans. When an interrupt occurs, sequence of following six steps takes place :

1. **Interrupt recognition :**
   a. The interrupt recognition is recognized by the processor of an interrupt request due to activation of an interrupt request line or an internal mechanism.
   b. In this step, the processor can determine which device or CPU, component made the request.
2. **Status saving :**
   a. The goal of this step is to make the interrupt sequence transparent to the interrupted process.
   b. Therefore, the processor saves the flags and registers that may be changed by the interrupt service routine so that they may be restored after the service routine is finished.
3. **Interrupt masking :**
   a. For the first few steps of the sequence, all interrupts are masked out so that no other interrupt may be processed before the processor status is saved.

b. The mask is then set to accept interrupts of higher priority.

4. Interrupt acknowledgment :

a. At some point, the processor must acknowledge the interrupt being serviced, so that the interrupting device becomes free to continue its task.

b. One of the ways is to have an external signal line denoted to interrupt acknowledge.

5. Interrupt service routine :

a. At this point, the processor initiates the interrupt service routine.

b. The address of the routine can be obtained in several ways, depending on the system architecture.

c. The simplest is found in the polling method, in which one routine polls each device to find which one interrupted.

6. Restoration and return :

a. After the interrupt service routine has completed its processing, it restores all the registers it has changed, and the processor restores all the registers and flags that were saved at the initiation of the interrupt routine.

b. If this is done correctly, the processor should have the same status as before the interrupt was recognized.

7. A computer uses RAM chips of 1024*1 capacity.

i. How many chips are needed and how should their address lines be connected to provide a memory capacity of 1024*8 ?

ii. How many chips are needed to provide a memory capacity of 16 KB ? Explain in words how the chips are to be connected to the address bus.

Ans.

i. Available size of RAM chips = 1024×1

Required memory capacity = 1024 bytes

$= 1024 \times 8$

$$\text{Number of chips required} = \frac{1024 \times 8}{1024 \times 1} = 8 \text{ chips}$$

So, 8 chips are needed with address line connected in parallel.

ii. To provide a memory capacity of 16K bytes, chips required are $16 \times 8 = 128$ chips

Number of address line for 16 K = 14 ($16\ K = 2^{14}$)

So, 14 lines to specify chip address.

8. A ROM chip of 1024*8 has four select inputs and operates from a 5 volt power supply. How many pins are needed for the IC package ? Draw a block diagram and label all input and output terminals in the ROM.

Ans. Size of ROM Chip = 1024 × 8
Number of input = 10 pin [2^{10} = 1024]
Number of output = 8 pin
Number of chip select = 4 pin
Power = 2 pin
Total 24 pins are required.

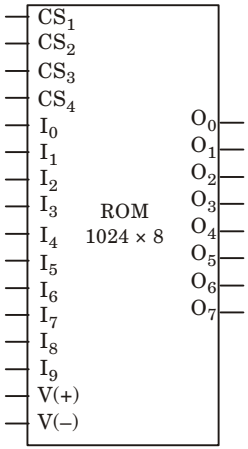


Fig. 6.

9. i. What are the differences between hardwired and microprogrammed control unit ?

Ans. **Comparison between hardwired control and micro-programmed control :**

S. No.	Characteristics	Hardwired control	Micro-programmed control
1.	Speed	Fast	Slow
2.	Implementation	Hardware	Software
3.	Flexibility	Not flexible	Flexible
4.	Ability to handle large/ complex instruction set	Somewhat difficult	Easier
5.	Ability to support operating system and diagnostic features	Very difficult	Easy

6.	Design process	Difficult for more operation	Easy
7.	Memory	Not used	Control memory used (RAM or ROM)
8.	Chip area efficiency	Uses less area	Uses more area
9.	Used in	RISC processor	CISC processor
10.	Output generation	On the basis of input signal	On the basis of control line.

ii. What is RISC ? Explain its various characteristics.

Ans.

1. RISC (Reduced Instruction Set Computer) processor instruction has a fixed length encoding of instruction and each instruction executes in a single clock cycle by hardwired implementation of each instruction.
2. RISC architecture focus on reducing the number of instructions and working with simpler instruction set having limited number of addressing modes and allowing them to execute more instructions in the same amount of time.
3. Programs written for RISC architectures tend to make more space in memory but RISC processor's increased clock rate allows it to execute its program in less time than a CISC processor takes to execute its program.

RISC characteristics :

1. Simple instructions are used in RISC architecture.
2. RISC helps and supports few simple data types and synthesizes complex data types.
3. RISC utilizes simple addressing modes and fixed length instructions for pipelining.
4. RISC permits any register to use in any context.

SECTION-C

Note : Attempt any **two** questions from this section. **(15 × 2 = 30)**

10. i. What is the distinction between spatial locality and temporal locality ?

Ans. **Difference :**

S. No.	Spatial locality	Temporal locality
1.	Spatial locality refers to the tendency of execution to involve a number of memory locations that are clustered.	Temporal locality refers to the tendency for a processor to access memory locations that have been used recently.
2.	The spatial locality means that instructions stored near-by to the recently executed instructions are also likely to be executed soon.	The temporal locality means that a recently executed instruction is likely to be executed again very soon.
3.	The spatial aspect suggests that instead of bringing just one item from the main memory to the cache, it is wise to bring several items that reside at adjacent addresses as well.	The temporal aspect of the locality reference suggests that whenever information of instruction and data is first needed, this information should be brought into cache where it will hopefully remain until it is needed again.

ii. Show the multiplication process using Booth's algorithm when the following numbers are multiplied : (– 13) by (+ 8)

Ans.

True binary equivalent of + 8 = 01000
True binary equivalent of + 13 = 01101
1's complement of + 13 = 10010
+ 1
2's complement of + 13 = 10011 (–13)
Multiplier = 01000
Multiplicand (B) = 10011

A	Q_n	Q_{n+1}	Operation	SC
00000	01000	0		100
00000	00100	0	Ashr AQQ_{n+1}	
00000	00010	0	Ashr AQQ_{n+1}	011
00000	00001	0	Ashr AQQ_{n+1}	010
01101 00110	00001 10000	0 1	Add $\bar{B} + 1$ to A Ashr AQQ_{n-1}	001
11001 11100	10000 11000	1 0	Add B to A Ashr AQQ_{n+1}	000
Result : 11100	11000 = – 104 (2's complement of (+ 104))			

11. Why input output interface is required ? Describe in detail.

Ans.

i. I/O interface provides a method of transferring information between internal storage and external I/O devices.

ii. In order to interface peripherals with the CPU, I/O interfaces contain special communication links. These communication links are used to overcome, the differences between the CPU and peripherals such as data transfer speed, mode of operation, etc.

The major requirements for an I/O module can be given as :

1. **Processor communication :** This involves the following tasks :
   - **a.** Exchange of data between processor and I/O module.
   - **b. Command decoding :** The I/O module for a disk drive may accept the following commands from the processor : READ SECTOR, WRITE SECTOR, SEEK track, etc.
   - **c. Status reporting :** The device must be able to report its status to the processor. For example, disk drive busy, ready etc.
   - **d.** Status reporting may also involve reporting various errors.
   - **e. Address recognition :** Each I/O device has a unique address and the I/O module must recognize this address.
2. **Device communication :** The I/O module is able to perform device communication such as status reporting.
3. **Control and timing :** The I/O module is able to co-ordinate the flow of data between the internal resources (such as processor, memory) and external devices.
4. **Data buffering :**
   - a. This is necessary as there is a speed mismatch between speed of data transfer between processor and memory and external devices.
   - b. Data coming from the main memory are sent to an I/O module in a rapid burst.
   - c. The data is buffered in the I/O module and then sent to the peripheral device at its rate.
5. **Error detection :**
   - a. The I/O module is able to detect errors and report them to the processor.
   - b. These errors may be mechanical errors (such as paper jam in a printer), or changes in the bit pattern of transmitted data. A common way of detecting such errors is by using parity bits.

12. Differentiate among :

i. Strobe control and handshaking asynchronous data transfer modes.

ii. Processor and IOP.

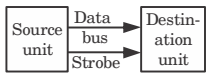
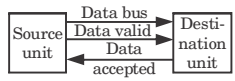
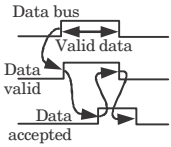
iii. Synchronous and asynchronous transmission.

iv. Character-oriented and Bit-oriented protocols.

v. DMA and interrupt initiated I/O techniques.

Ans.

i. Strobe control and handshaking asynchronous data transfer modes :

S. No.	Parameter	Strobe control	Handshaking
1.	Control line	It employs a single control line to time each transfer.	It employs more than single control line to time each transfer.
2.	Acknowledgement	Reply message is not present.	Reply message is present.
3.	Block diagram		
4.	Timing diagram		

ii. Processor and IOP :

S. No.	Processor	IOP
1.	Processor is CPU.	IOP is port of CPU processing.
2.	Handles arithmetic and logical tasks.	Handles only I/O processing.
3.	DMA controller is set-up by CPU.	IOP is a processor with DMA.

iii. Synchronous and asynchronous transmission :

S. No.	Synchronous transmission	Asynchronous transmission
1.	Transmitter and receivers are synchronized by clock.	Transmitter and receivers are not synchronized by clock.
2.	Data bits are transmitted with synchronization of clock.	Bits of data are transmitted at constant rate.
3.	Data transfer takes place in blocks.	Data transfer is character oriented.

iv. Character-oriented and Bit-oriented protocols :

S. No.	Character-oriented protocol	Bit-oriented protocol
1.	The character-oriented protocol is based on the binary code of a character set.	The bit-oriented protocol does not use characters in its control field and is independent of any particular code.
2.	The code has 128 characters, of which 95 are graphics characters and 33 are control characters.	It allows the transmission of serial bit stream of any length without the implication of character boundaries.

Message format for character-oriented protocol :

SYN	SYN	SOH	Header	STX	Text	ETX	BCl

Frame format for bit-oriented protocol :

Flag 01111110	Address 8 bits	Control 8 bits	Information any number of bits	Frame check 16 bits	Flag 01111110

v. DMA and interrupt initiated I/O techniques :

S. No.	DMA	Interrupt initiated I/O
1.	As DMA initializes, CPU become idle.	CPU executes the current program, during the interrupt initiated I/O technique.
2.	As the DMA disables, memory buses are returned to CPU and CPU starts executing its program.	After the transfer, CPU returns to the previous program to continue.

☺☺☺

B.Tech.
(SEM. V) ODD SEMESTER THEORY EXAMINATION, 2016-17
COMPUTER ARCHITECTURE

Time : 3 Hours **Total Marks : 100**

SECTION – A

Note : Attempt **all** questions : **(2 × 10 = 20)**

1. **Define following terms :**
   i. **RTL**
   ii. **Micro-operation**
2. **Define sequencer.**
3. **Explain one, two and three address instruction.**
4. **Define the following terms :**
   i. **Effective address**
   ii. **Immediate instruction**
5. **Explain the following terms :**
   i. **PSW**
   ii. **Delayed load**
6. **Differentiate SIMD and MIMD.**
7. **What are the modes of data transfer ?**
8. **What is an interrupt ?**
9. **Differentiate between synchronous and asynchronous transmission.**
10. **What is cache memory used for ?**

SECTION – B

Note : Attempt any **five** questions : **(10 × 5 = 50)**

1. **Show the contents of the registers E, A, Q, SC during the process of multiplication of two binary numbers 11111 (multiplicand) and 10101 (multiplier). The signs are not included.**

2. **In an instruction format, there are 16 bits in an instruction word. Bit 0 to 11 convey the address of the memory location for memory related instructions. For non memory instructions these bits convey various register or I/O operations. Bits 12 to 14 show the various basic memory operations such as ADD, AND, LDA etc. Bit 15 shows if the memory is accessed directly or indirectly. For such an instruction format draw block diagram of the control unit of a computer and briefly explain how an instruction will be decoded and executed, by this control unit.**

3. **Write an assembly level program for the following pseudocode :**

```
SUM = 0
SUM = SUM + A + B
DIF = DIF-C
SUM = SUM + DIF
```

4. **Explain microprogram sequencer for a control memory using a suitable block diagram.**

5. **Give the detailed comparison between RISC and CISC.**

Section-C

Note : Attempt any **two** questions : **(15 × 2 = 30)**

1. **Explain the Booth's algorithm in depth with the help of flowchart. Give an example for multiplication using Booth's algorithm.**

2. **How main memory is useful in computer system ? Explain the memory address map of RAM and ROM.**

3. a. **Draw a block diagram of a computer's CPU showing all the basic building blocks such as program counter, accumulator, address and data registers, instruction register, control unit etc., and describe how such an arrangement can work as a computer, if connected properly to memory, input / output etc.**

   b. **Describe the subroutine. Write a program which move the block of data.**

4. **Explain the operation of three state bus buffers and show its use in design of common bus.**

5. **Explain 4-bit incrementer with a necessary diagram.**

6. **Write a program loop using a pointer and a counter to clear the contents of hex locations 500 to 5FF with 0.**

7. **Demonstrate the process of second pass of assembler using a suitable diagram.**

8. **Explain :**
   i. **Vector processing**
   ii. **Vector operations**
   Explain how matrix multiplication is carried out on a computer supporting vector computations.

9. **Explain Flynn's classification of computers.**

10. **How addressing mode is significant for referring memory ? List and explain different types of addressing modes.**

11. **What is a memory stack ? Explain its role in managing subroutines with the help of neat diagrams.**

12. **What is stack ? Give the organization of register stack with all necessary elements and explain the working of push and pop operations.**

13. **Write a note on subroutines.**

14. **Draw the block diagram of control unit of basic computer. Explain in detail with control timing diagrams.**

15. **List and explain different types of shift micro-operation.**

SOLUTION OF PAPER (2016-17)

SECTION – A

Note : Attempt **all** questions : **(2 × 10 = 20)**

1. Define following terms :

i. RTL

ii. Micro-operation

Ans.

i. RTL : Register Transfer Language (RTL) is a convenient tool for describing the internal organization of digital computers in concise and precise manner. It can also be used to facilitate the design process of digital systems.

ii. Micro-operation : The processor unit has to perform a set of operations to execute the major phases of instruction cycle these set of operations called micro-operations.

2. Define sequencer.

Ans. A sequencer generates the addresses used to step through the micro-program of a control store. It is used as a part of control unit of CPU for address ranges.

3. Explain one, two and three address instruction.

Ans.

i. One address instruction : One address instruction uses an implied accumulator (AC) register for all data manipulation.

ii. Two address instruction : In this format each address field can specify either a processor register or a memory word.

iii. Three address instruction : Three address instruction formats can use each address fields to specify either a processor register or a memory operand.

4. Define the following terms :

i. Effective address

ii. Immediate instruction

Ans.

i. Effective address : Effective address is the address of the operand in a computation-type instruction or the target address in a branch-type instruction.

ii. Immediate instruction : An immediate mode instruction has an operand field rather than an address field. The operand field contains the actual operand to be used in conjunction with the operation specified in the instruction.

5. Explain the following terms :

i. PSW **ii. Delayed load**

Ans.

i. PSW (Program Status Word) :

1. The collection of all status bit conditions in the CPU is sometimes called a program status word or PSW.
2. The PSW is stored in a separate hardware register and contains the status information that characterizes the state of the CPU.
3. It includes the status bits from the last ALU operation and it specifies the interrupts that are allowed to occur and whether the CPU is operating in a supervisor or user mode.

ii. Delayed load :

1. It is up to the compiler to make sure that the instruction following the load instruction uses the data fetched from memory.
2. If the compiler cannot find a useful instruction to put after the load, it inserts a no-op (no-operation) instruction.
3. This is a type of instruction that is fetched from memory but has no operation, thus wasting a clock cycle.
4. This concept of delaying the use of the data loaded from memory is referred to as delayed load.

6. Differentiate SIMD and MIMD.

Ans.

S. No.	SIMD	MIMD
1.	It stands for Single Instruction Stream, Multiple Data Stream.	It stands for Multiple Instruction Stream, Multiple Data Stream.
2.	In this, a parallel computer consists of *N* identical processors.	In this, we have *N* processors, *N* streams of instructions and *N* streams of data.

7. What are the modes of data transfer ?

Ans. Modes of data transfer are :

i. Programmed I/O : Programmed I/O operations are the result of I/O instructions written in computer program.

ii. Interrupt-driven I/O : This mode avoids the drawbacks of programmed I/O by using interrupt request.

iii. Direct memory access : The interface transfers data onto and out of the memory unit through memory bus.

8. What is an interrupt ?

Ans.

1. An interrupt is a signal sent by an I/O interface to the CPU when it is ready to send information to the memory or receive information from the memory.

2. When a CPU receives an interrupt signal it stops executing current normal program.
3. After stopping it saves the state of various registers in stack.
4. When this is done CPU executes a subroutine in order to perform the specific task requested by the interrupt.

9. Differentiate between synchronous and asynchronous transmission.

Ans.

S. No.	Synchronous serial communication	Asynchronous serial communication
1.	Transmitter and receivers are synchronized by clock.	Transmitter and receivers are not synchronized by clock.
2.	Data bits are transmitted with synchronization of clock.	Data bits are transmitted at constant rate.
3.	Data transfer takes place in blocks.	Data transfer is character-oriented.

10. What is cache memory used for ?

Ans.

1. Cache memory is used to store frequently used data or instructions.
2. Cache memory is used to improve computer performance by reducing its access time.
3. A cache holds instructions and data that are likely to be needed for the CPU's next operation.

SECTION – B

Note : Attempt any **five** questions : **(10 × 5 = 50)**

1. Show the contents of the registers E, A, Q, SC during the process of multiplication of two binary numbers 11111 (multiplicand) and 10101 (multiplier). The signs are not included.

Ans.

Multiplicand *B* = 11111	***E***	***A***	***Q***	***SC***
	0	**00000**	**10101**	**101**
Q_n = 1 ; add *B*		00000 11111		
		11111		
Shift right EAQ	0	01111	11010	100
Q_n = 1 ; add *B*		01111 11111		
		00000		
Shift right EAQ	0	00000	01101	011

Q_n = 0 ; shift right EAQ	0	00000	00110	010
Q_n = 0 ; shift right EAQ	0	00000	00011	001
Q_n = 0 ; shift right EAQ	0	00000	00001	000

2. In an instruction format, there are 16 bits in an instruction word. Bit 0 to 11 convey the address of the memory location for memory related instructions. For non memory instructions these bits convey various register or I/O operations. Bits 12 to 14 show the various basic memory operations such as ADD, AND, LDA etc. Bit 15 shows if the memory is accessed directly or indirectly. For such an instruction format draw block diagram of the control unit of a computer and briefly explain how an instruction will be decoded and executed, by this control unit.

Ans.

1. Consider the instruction code format shown in Fig. 1(a). It consists of a 3-bit operation code, a 12-bit address, and an indirect address mode bit designated by *I*.
2. The mode bit is 0 for a direct address and 1 for an indirect address.
3. A direct address instruction is shown in Fig. 1(b). It is placed in address 22 in memory. The *I* bit is 0, so the instruction is recognized as a direct address instruction.

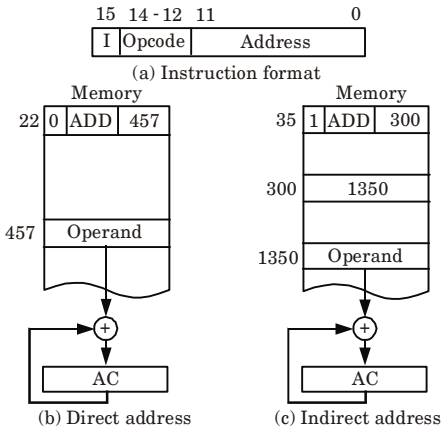


Fig. 1.

4. The opcode specifies an ADD instruction, and the address part is the binary equivalent of 457.
5. The control finds the operand in memory at address 457 and adds it to the content of *AC*.
6. The instruction in address 35 shown in Fig. 1(c) has a mode bit *I* = 1. Therefore, it is recognized as an indirect address instruction.

7. The address part is the binary equivalent of 300. The control goes to address 300 to find the address of the operand.
8. The address of the operand in this case is 1350. The operand found in address 1350 is then added to the content of *AC*.
9. The indirect address instruction needs two references to memory to fetch an operand.

Instruction will be decoded and executed by this control unit :

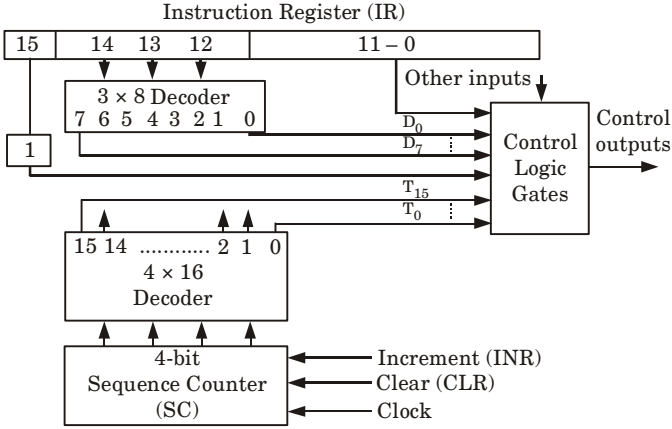


Fig. 2. Block diagram of control unit of a computer.

1. Control unit consists of :

i. Instruction register
ii. Number of control logic gates
iii. Two decoders
iv. 4-bit sequence counter

2. An instruction read from memory is placed in the Instruction Register (IR).
3. The instruction register is divided into three parts : the *I* bit, operation code, and address part.
4. First 12-bits (0 – 11) are applied to the control logic gates.
5. The operation code bits (12 – 14) are decoded with a 3 × 8 decoder.
6. The eight outputs (D_0 through D_7) from a decoder go to the control logic gates to perform specific operation.
7. Last bit 15 is transferred to a *I* flip-flop designated by symbol *I*.
8. The 4-bit Sequence Counter (SC) can count in binary from 0 through 15.
9. The counter output is decoded into 16 timing pulses T_0 through T_{15}.
10. The sequence counter can be incremented by INR input or clear by CLR input synchronously.

3. Write an assembly level program for the following pseudocode :

SUM = 0
SUM = SUM + A + B
DIF = DIF-C
SUM = SUM + DIF

Ans.

```
CLA                 /SUM = 0
STA SUM
LDA SUM             /Load current sum
ADD A               /Add A to SUM
ADD B               /Add B to SUM
STA SUM             /Save SUM
LDA C               /Load C to AC
CMA                 /Create 2's complement
INC
ADD DIF             /Subtract C from DIF
STA DIF             /Save DIF
ADD SUM             /Add SUM to DIF
STA SUM             /Save SUM
HLT                 /Halt
```

4. Explain microprogram sequencer for a control memory using a suitable block diagram.

Ans.

1. Micro-program sequencer is a general purpose building block for micro-programmed control unit.
2. The basic components of a micro-programmed control unit are the control memory and the circuit that selects the next address.

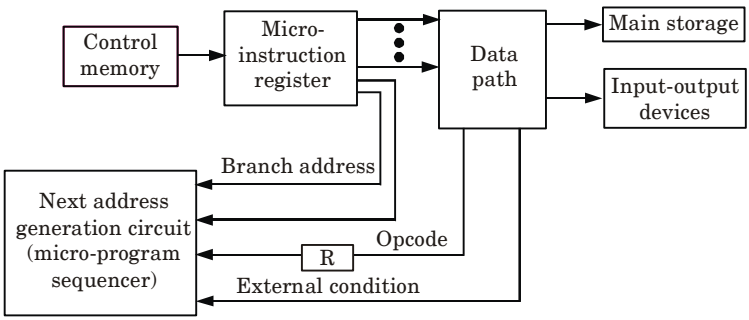


Fig. 3. Block diagram of micro-programmed control with micro-program sequencer.

3. The address selection part is called micro-program sequencer.
4. The main purpose of micro-program sequencer is to present an address to the control memory so that micro-instruction may be read and executed.

5. The next address logic of the sequencer determines the specific address source to be loaded into the control address register.
6. The choice of the address source is guided by the next address information bits that sequencer receives from the present micro-instruction.
7. All the instructions are loaded in the control memory.
8. The present micro-instruction is placed in micro-instruction register for execution.

5. Give the detailed comparison between RISC and CISC.

Ans.

S. No.	RISC	CISC
1.	Multiple register sets, often consisting of more than 256 registers.	Single register set, often consisting 6 to 16 registers total.
2.	Three register operands allowed per instruction (for example, add R1, R2, R3).	One or two register operands allowed per instruction (for example, add R1, R2).
3.	Parameter passing through efficient on-chip register windows.	Parameter passing through inefficient off-chip memory.
4.	Single-cycle instructions (except for load and store).	Multiple-cycle instructions.
5.	Hardwired control.	Micro-programmed control.
6.	Highly pipelined.	Less pipelined.
7.	Simple instructions are few in number.	There are many complex instructions.
8.	Fixed length instructions.	Variable length instructions.
9.	Complexity in compiler.	Complexity in microcode.
10.	Only load and store instructions can access memory.	Many instructions can access memory.
11.	Few addressing modes.	Many addressing modes.

Section-C

Note : Attempt any **two** questions : **(15 × 2 = 30)**

1. Explain the Booth's algorithm in depth with the help of flowchart. Give an example for multiplication using Booth's algorithm.

Ans. The algorithm for 2's complement multiplication is as follows :
Step 1 : Load multiplicand in B, multiplier in Q. For negative numbers, 2's complement format to be used.

Step 2 : Initialize the down counter CR by the number of bits involved.
Step 3 : Clean locations A (n-bits) and Q_{n+1} (1-bit).
Step 4 : Check LS bit of Q_n and Q_{n+1} jointly. If the pattern is 00 or 11 then go to Step 5. If 10, then $A = A - B$. If 01, then $A = A + B$.
Step 5 : Perform arithmetic right-shift with A, Q_n and Q_{n+1}. LS of A goes to MS of Q_n and LS of Q goes to Q_{n+1}. Old content of Q_{n+1} is discarded.
Step 6 : Decrement CR by one. If CR is not zero then go to Step 4.
Step 7 : Final result (or the product) is available in A (higher part) and Q_n (lower part).

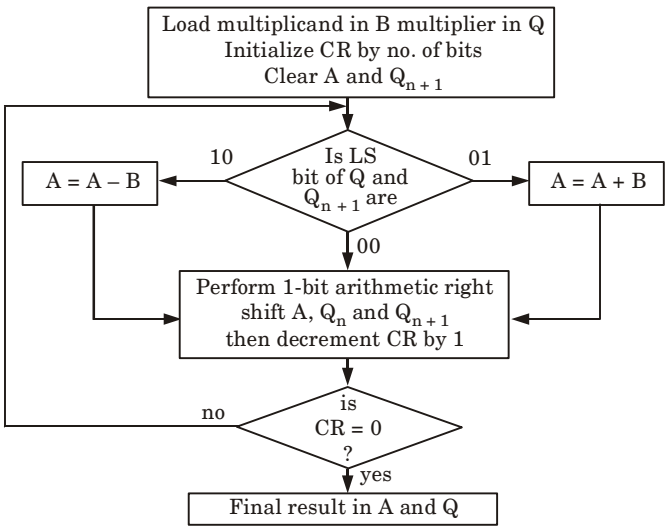


Fig. 4. 2's Complement multiplication.

Example : Both negative (– 5 × – 4)

Multiplicand (B) ← 1 0 1 1 (– 5)		**Multiplier (Q) ← 1 1 0 0 (– 4)**		
A	***Q_n***	***Q_{n+1}***	**Operation**	**CR**
0 0 0 0	1 1 0 0	0	Initial	4
0 0 0 0	0 1 1 0	0	Shift right	3
0 0 0 0	0 0 1 1	0	Shift right	2
0 1 0 1	0 0 1 1	0	A ← A – B	1
0 0 1 0	1 0 0 1	1	Shift right	
0 0 0 1	0 1 0 0	1	Shift right	0
Result : 0 0 0 1	0 1 0 0	= + 20		

2. How main memory is useful in computer system ? Explain the memory address map of RAM and ROM.

Ans. **Main memory is useful in computer system :**

1. The main memory occupies a central position in a computer system.
2. It is able to communicate directly with the CPU and with auxiliary memory devices through an I/O processor.
3. It is relatively large and fast.

Memory address map of RAM and ROM :

1. Memory address map is a pictorial representation of assigned address space for each chip in the system.

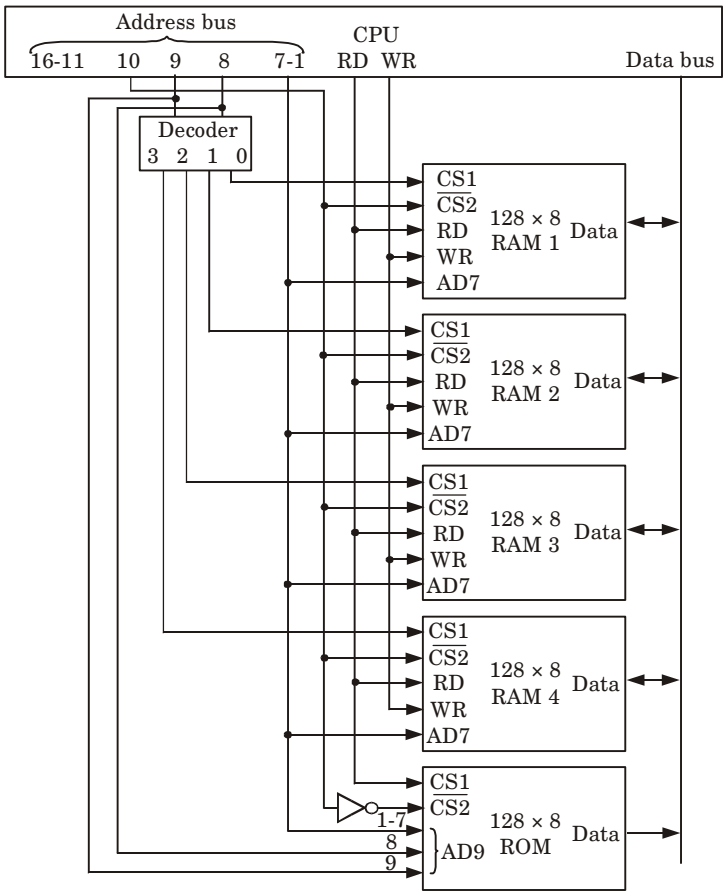


Fig. 5.

2. To demonstrate with a particular example, assume that a computer system needs 512 bytes of RAM and 512 bytes of ROM.

3. The RAM and ROM chips to be used are specified in Fig. 5. The memory address map for this configuration is shown in Table 1.
4. The component column specifies whether a RAM or a ROM chip is used. The hexadecimal address column assigns a range of hexadecimal equivalent addresses for each chip. The address bus lines are listed in the third column.
5. The selection between RAM and ROM is achieved through bus line 10. The RAMs are selected when the bit in this line is 0, and the ROM when the bit is 1.

Table 1 : Memory address map.

Component	Hexadecimal Address	Address bus									
		10	9	8	7	6	5	4	3	2	1
RAM1	0000-007F	0	0	0	×	×	×	×	×	×	×
RAM2	0080-00FF	0	0	1	×	×	×	×	×	×	×
RAM3	0100-017F	0	1	0	×	×	×	×	×	×	×
RAM4	0180-01FF	0	1	1	×	×	×	×	×	×	×
ROM	0200-03FF	1	×	×	×	×	×	×	×	×	×

6. The × under the address bus lines designate those lines that must be connected to the address inputs in each chip.
7. The RAM chips have 128 bytes and need seven address lines.
8. The ROM chip has 512 bytes and need 9 address lines.
9. It is now necessary to distinguish between four RAM chips by assigning to each a different address.

3. a. Draw a block diagram of a computer's CPU showing all the basic building blocks such as program counter, accumulator, address and data registers, instruction register, control unit etc., and describe how such an arrangement can work as a computer, if connected properly to memory, input / output etc.

Ans. **Block diagram of computer's CPU :**
A computer performs five major operations. These are :

1. It accepts data or instructions as input.
2. It stores data and instruction.
3. It processes data as per the instructions.
4. It controls all operations inside a computer.
5. It gives results in the form of output.

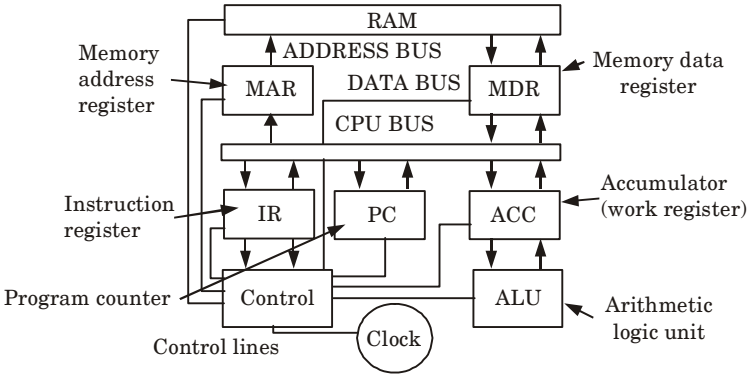


Fig. 6.

Arrangement of CPU, memory, input/output to work as a computer :

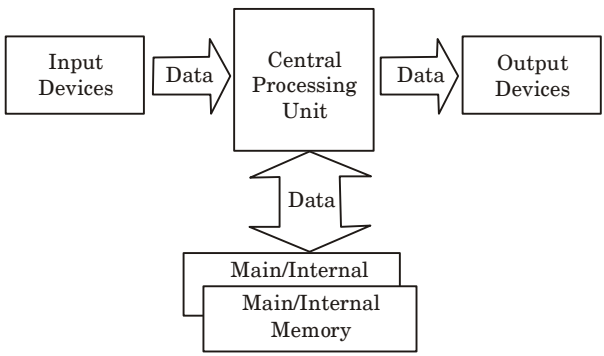


Fig. 7.

a. **Input unit :** This unit is used for entering data and programs into the computer system by the user for processing.
b. **Storage unit :** The storage unit is used for storing data and instructions before and after processing.
c. **Output unit :** The output unit is used for storing the result as output produced by the computer after processing.
d. **Processing unit :** The task of performing operations like arithmetic and logical operations is called processing.
   The Central Processing Unit (CPU) takes data and instructions from the storage unit and makes all sorts of calculations based on the instructions given and the type of data provided. It is then sent back to the storage unit.

b. Describe the subroutine. Write a program which move the block of data.

Ans. **Subroutine :**

1. A subroutine is a set of common instructions that can be used in a program many times.
2. A subroutine consists of a self-contained sequence of instructions that carries out a given task.
3. Each time that a subroutine is used in the main part of the program, a branch is executed to the beginning of the subroutine.
4. After the subroutine has been executed, a branch is made back to the main program.
5. A branch can be made to the subroutine from any part of the main program.
6. Because branching to a subroutine and returning to the main program is such a common operation, all computers provide special instructions to facilitate subroutine entry and return.

Program :

```
LXI H, XX50H        ; Set up HL as a pointer for the source memory
LXI D, XX70H        ; Set up DE as a pointer for the destination
                      memory
MVI B, 10H          ; Set up B as byte counter
NEXT : MOV A,M      ; Get data byte from the source memory
STAX D              ; Store the data byte in destination memory
INX H
INX D               ; Get ready to transfer next byte
DCR B
JNZ NEXT            ; Go back to get next byte if byte counter ≠ 0
HLT
```

4. Explain the operation of three state bus buffers and show its use in design of common bus.

Ans.

1. A three state gate is a digital circuit that exhibits three states.
2. Two of the states are signals equivalent to logic 1 and 0.
3. The third state is a high-impedance state.
4. The high-impedance state behaves like an open circuit, which means that the output is disconnected and does not have logic significance.
5. Three state gates may perform any conventional logic, such as AND or NAND.
6. However, the one most commonly used in the design of a bus system is the buffer gate.
7. The graphic symbol of a three state buffer gate is shown in Fig. 8.

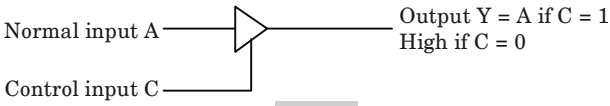


Fig. 8.

8. The control input determines the output state.
9. When the control input is equal to 1, the output is enabled and the gate behaves like any conventional buffer, with the output equal to the normal input.
10. When the control input is 0, the output is disabled and the gate goes to a high state, regardless of the value in the normal input.
11. A large number of three state gate outputs can be connected with wires to form a common bus line without endangering loading effects.

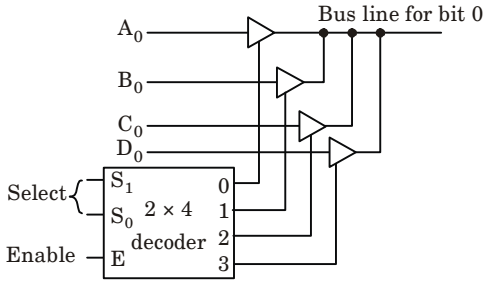


Fig. 9.

12. The outputs of four buffers are connected together to form a single bus line.
13. The control inputs to the buffers determine which of the four normal inputs will communicate with the bus line.
14. Not more than one buffer may be in the active state at any given time.
15. To construct a common bus for four registers of *n* bits each using three state buffers, we need *n* circuits with four buffers in each.
16. Each group of four buffers receives one significant bit from the four registers.
17. Only one decoder is necessary to select between the four registers.

5. Explain 4-bit incrementer with a necessary diagram.

Ans.

1. The diagram of a 4-bit combinational circuit incrementer is shown in Fig. 10.
2. One of the inputs to the least significant Half Adder (HA) is connected to logic-1 and the other input is connected to the least significant bit of the number to be incremented.
3. The output carry from one half-adder is connected to one of the inputs of the next-higher-order half-adder.

4. The circuit receives the four bits from A_0 through A_3 adds one to it, and generates the incremented output in S_0 through S_3.
5. The output carry C_4 will be 1 only after incrementing binary 1111. This also causes outputs S_0 through S_3 to go to 0.
6. This micro-operation is easily implemented with a binary counter.

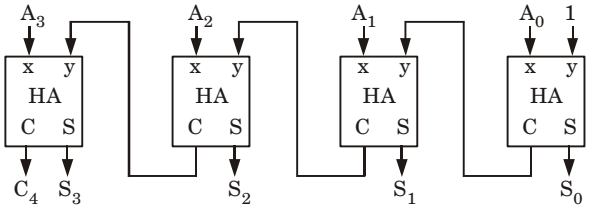


Fig. 10. 4-bit binary incrementer.

7. Every time the count enable is active, the clock pulse transition increments the content of the register by one.
8. There may be occasions when the increment micro-operation must be done with a combinational circuit independent of a particular register.
9. This can be accomplished by means of half adders connected in cascade.

6. Write a program loop using a pointer and a counter to clear the contents of hex locations 500 to 5FF with 0.

Ans.

```
LDA NBR      /  Initialize counter
CMA          /  2's complement of NBR INC
STA CTR      /  save -NBR to counter
LDA ADR      /  Save start address
STA PTR      /  Initialize pointer PTR
LOP, CLA     /  Clear AC
STA PTR I    /  Reset memory word
ISZ PTR      /  Increment pointer
ISZ CTR      /  increment counter
BUN LOP      /  Branch to LOP (CTR < 0)
HLT          /  Halt when CTR = 0
NBR, HEX FF  /  NBR of cleared words
CTR, –       /  Counter
ADR, HEX 500 /  Start address
PTR, –       /  Pointer
```

7. Demonstrate the process of second pass of assembler using a suitable diagram.

Ans.

1. Machine instructions are translated during the second pass by means of table-lookup procedures.

2. A table-lookup procedure is a search of table entries to determine whether a specific item matches one of the items stored in the table.
3. The assembler uses four tables.
4. Any symbol that is encountered in the program must be available as an entry in one of these tables; otherwise, the symbol cannot be interpreted.

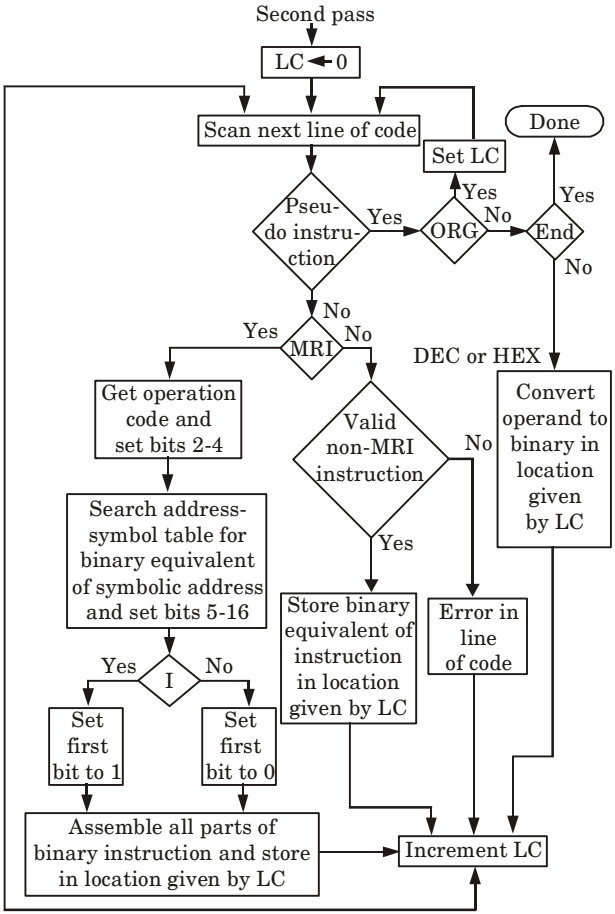


Fig. 11.

5. We assign the following names to the four tables :
   a. Pseudo instruction table
   b. MRI table
   c. Non-MRI table
   d. Address symbol table

6. The entries of the pseudo instruction table are the four symbols ORG, END, DEC, and HEX.
7. Each entry refers the assembler to a subroutine that processes the pseudo instruction when encountered in the program.
8. The MRI table contains the seven symbols of the memory-reference instructions and their 3-bit operation code equivalent.
9. The non-MRI table contains the symbols for the 18 register-reference and input-output instructions and their 16-bit binary code equivalent.
10. The assembler searches these tables to find the symbol that it is currently processing in order to determine its binary value.

8. Explain :

i. Vector processing

ii. Vector operations

Explain how matrix multiplication is carried out on a computer supporting vector computations.

Ans. This question is out of syllabus since session 2018-19.

9. Explain Flynn's classification of computers.

Ans. This question is out of syllabus since session 2018-19.

10. How addressing mode is significant for referring memory ? List and explain different types of addressing modes.

Ans.

1. The addressing mode is significant for referring memory as it is a code that tells the control unit how to obtain the Effective Address (EA) from the displacement.
2. Addressing mode is a rule of calculation, or a function that use displacement as its main argument and other hardware component as (such as PC, registers and memory locations) as secondary arguments and produce the EA as a result.

Types of addressing modes :

i. Direct :

1. A very simple form of addressing is direct addressing, in which the address field contain the effective address of the operand : $EA = A$ where, EA = Actual (effective) address of the location containing the referenced operand.

A = Contents of the address field in the instruction.

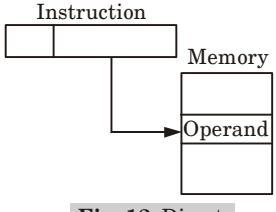


Fig. 12. Direct.

2. A direct address in instruction needs two reference to memory :
   a. Read instruction
   b. Read operand

ii. Displacement addressing :

1. A very powerful mode of addressing combines the capabilities of direct addressing and register indirect addressing.
2. It is known by a variety of names depending upon the content of its use but the basic mechanism is the same.
3. Displacement addressing requires that the instruction have two address fields, at least one of which is explicit.
4. The value contained in one address field (value = A) is used directly.
5. The other address field or an implicit reference based on opcode, refers to a register whose contents are added to A to produce the effective address.

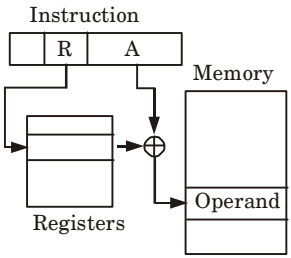


Fig. 13. Displacement addressing.

iii. Relative addressing :

1. Relative addressing means that the next instruction to be carried out is an offset number of locations away, relative to the address of the current instruction.
2. Consider this bit of pseudo-code
   Jump + 3 if accumulator = = 2
   Code executed if accumulator is NOT = 2
   Jump + 5 (unconditional relative jump to avoid the next line of code)
   acc : (code executed if accumulator is = 2)
3. In the code, the first line of code is checking to see if the accumulator has the value of 2 then the next instruction is 3 lines away.
4. This is called a conditional jump and it is making use of relative addressing.

iv. Register indirect mode :

1. Register indirect mode is similar to indirect addressing.
2. The only difference is whether the address field refers to a memory location or a register.
3. Thus, for register indirect address, $EA = (R)$

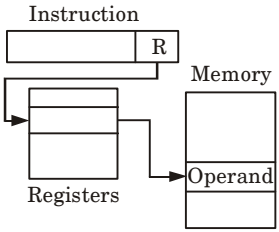


Fig. 14. Register indirect.

v. Implied mode :

1. In this mode, the operands are specified implicitly in the definition of the instruction.
2. All register reference instructions that use an accumulator are implied mode instructions.
3. Zero address instructions in a stack-organized computer are implied mode instruction since the operands are implied to be on top of the stack. It is also known as stack addressing mode.

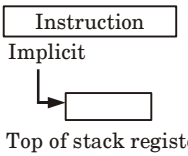


Fig. 15. Implied mode.

vi. Immediate mode :

1. In this mode, the operand is specified in the instruction itself.
2. The operand field contains the actual operand to be used in conjunction with the operation specified in the instruction.

Fig. 16. Immediate mode.

vii. Indexed :

1. The effective address of the operand is generated by adding a constant value to the contents of a register.
2. The register used may be either a special register for this purpose or more commonly, it may be any one of a set of general purpose registers in the CPU.
3. It is referred to as an index register. We indicate the index mode symbolically as, $X(R)$
   where X denotes a constant and R is the name of register involved.
4. The effective address of the operand is given by, $EA = X + [R]$
5. In the process of generating the effective address, the contents of the index register are not changed.

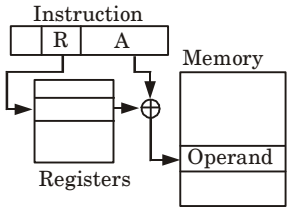


Fig. 17. Indexed.

11. What is a memory stack ? Explain its role in managing subroutines with the help of neat diagrams.

Ans. **Memory stack :** Memory stack is a series of memory spaces that is used in the processes that is done by processor and is temporarily stored in registers.

Role in managing subroutines :

1. The stack supports program execution by maintaining automatic process-state data.
2. If the main routine of a program, for example, invokes function *a* ( ), which in turn invokes function *b* ( ), function *b* ( ) will eventually return control to function *a* ( ), which in turn will return control to the main ( ) function as shown in Fig. 18.
3. To return control to the proper location, the sequence of return addresses must be stored.
4. A stack is well suited for maintaining this information because it is a dynamic data structure that can support any level of nesting within memory constraints.

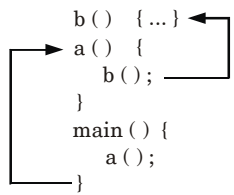


Fig. 18. Stack management.

5. When a subroutine is called, the address of the next instruction to execute in the calling routine is pushed onto the stack.
6. When the subroutine returns, this return address is popped from the stack, and program execution jumps to the specified location as shown in Fig. 19.

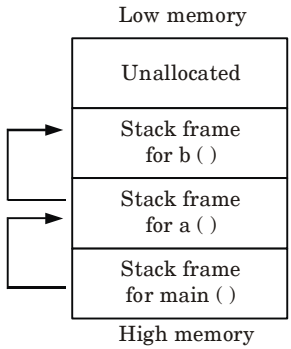


Fig. 19. Calling a subroutine.

7. The information maintained in the stack reflects the execution state of the process at any given instant.
8. In addition to the return address, the stack is used to store the arguments to the subroutine as well as local (or automatic) variables.
9. Information pushed onto the stack as a result of a function call is called a frame. The address of the current frame is stored in the frame or base pointer register.
10. When a subroutine is called, the frame pointer for the calling routine is also pushed onto the stack so that it can be restored when the subroutine exits.

12. What is stack ? Give the organization of register stack with all necessary elements and explain the working of push and pop operations.

Ans.

1. A stack is an ordered set of elements in which only one element can be accessed at a time.
2. The point of access is called the top of the stack.
3. The number of elements in the stack or length of the stack is variable.
4. Items may only be added or deleted from the top of the stack.
5. A stack is also known as a pushdown list or a Last-In-First-Out (LIFO) list.

Organization of register stack :

Consider the organization of a 64-word register stack as illustrated in Fig. 20.

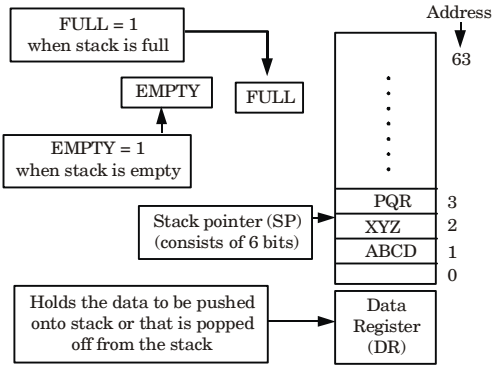


Fig. 20. Block diagram of 64-word stack.

The four separate registers used in the organization are :

1. **Stack Pointer register (SP) :** It contains a value in binary each of 6 bits, which is the address of the top of the stack. Here, the stack pointer SP contains 6 bits and SP cannot contain a value greater than 111111 *i.e.*, value 63.
2. **FULL register :** It can store 1 bit information. It is set to 1 when the stack is full.
3. **EMPTY register :** It can store 1 bit information. It is set to 1 when stack is empty.
4. **Data Register (DR) :** It holds the data to be written into or to be read from the stack.

Working of POP and PUSH :

POP (Performed if stack is not empty *i.e.*, if EMPTY = 0) :

DR ← M[SP]	Read item from the top of stack
SP ← SP – 1	Decrement stack pointer
If (SP = 0) then (EMPTY ← 1)	Check if stack is empty
FULL ← 0	Mark the stack not full

PUSH (Performed if stack is not full *i.e.*, if FULL = 0) :

SP ← SP + 1	Increment stack pointer
M[SP] ← DR	Write item on top of the stack
If (SP = 0) then (FULL ← 1)	Check if stack is full
EMPTY ← 0	Mark the stack not empty

13. Write a note on subroutines.

Ans. **Subroutine :**

1. A subroutine is a set of common instructions that can be used in a program many times.
2. A subroutine consists of a self-contained sequence of instructions that carries out a given task.
3. Each time that a subroutine is used in the main part of the program, a branch is executed to the beginning of the subroutine.

4. After the subroutine has been executed, a branch is made back to the main program.
5. A branch can be made to the subroutine from any part of the main program.
6. Because branching to a subroutine and returning to the main program is such a common operation, all computers provide special instructions to facilitate subroutine entry and return.

14. Draw the block diagram of control unit of basic computer. Explain in detail with control timing diagrams.

Ans.

1. The control unit consists of 2 decoders, 1 sequence counter, number of control logic gates.
2. The instruction in IR is divide into 3 parts : 15th bit to a flip-flop (FF) called I, Operation code, and bits 0 to 11. The Op-code is decoded using 3*8 decoder (D_0 to D_7). Bits 0 to 11 are applied to the control logic gates. The output of a 4-bit sequence counter are decoded into 16 timing signals (T_0 to T_{15}).
3. The SC responds to the positive transition of the clock. Initially CLR I/ P is active, in 1st positive transition SC = 0, timing signal T_0 is active as the output of the decoder. This in turn triggers those registers whose control inputs are connected to T_0. SC is incremented and the timing signals T_0, T_1, T_2, T_3 .... are created. This continues unless SC is cleared. We can clear the SC with decoder output D_3 active, denoted as :

$$D_3T_4 : SC \leftarrow 0$$

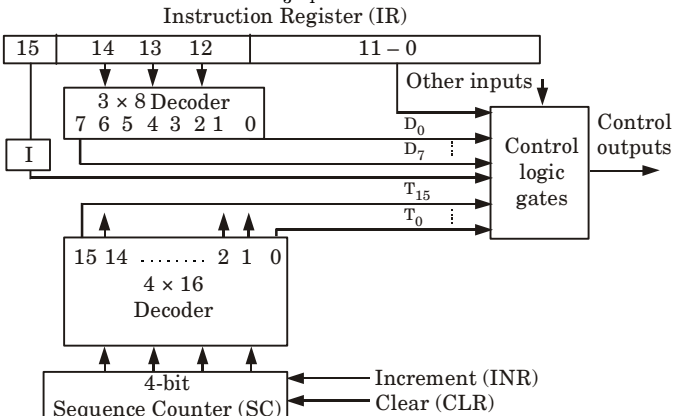


Fig. 21. Block diagram of control unit.

4. Output D_3 from the operation decoder becomes active at the end of T_2. When T_4 is active, the output of AND gate that implements the control function D_3T_4 becomes active. This signal is applied to CLR input of SC.

5. Example of register transfer : $T_0 : AR \leftarrow PC$ (Activities in T_0 will be, Content of PC placed on bus, $S_2S_1S_0 = 010$, LD of AR is active, transfer occurs at the end of positive transition, T_0 is .inactive, T_1 gets active).
6. Timing control is generated by 4-bit sequence counter and 4 × 16 decoder. The SC can be incremented or cleared. $T_0, T_1, T_2, T_3, T_4, T_0$, ...

For example :

Assume : At time T_4, SC is cleared to 0 if decoder output D_3 is active.

$D_3T_4 :$ $SC \leftarrow 0$

Timing diagram :

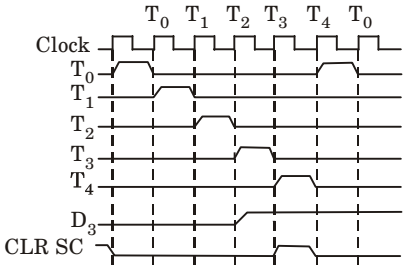


Fig. 22.

15. List and explain different types of shift micro-operation.

Ans. **Shift micro-operation :**

1. Shift micro-operations in computer architecture are those which are used in serial shifting of data present in a register.
2. Shift micro-operations move or shift data in a register bitwise that is, one bit at a time either left or right from its original position.

List of different types of shift micro-operation :

a. Arithmetic shift micro-operation :

i. Arithmetic shift operation shifts signed (positive or negative) binarynumbers either left or right by multiplying or dividing by 2.

ii. For arithmetic shift left micro-operation, the value in the register is multiplied by 2 and whereas for arithmetic shift right micro-operation, the value in the register is divided by 2.

iii. In RTL (Register Transfer Language), we can represent this arithmetic shift micro-operations as

$R \leftarrow$ ashl R (arithmetic shift left R (register))

$R \leftarrow$ ashr R (arithmetic shift right R (register))

iv. Fig. 23 showing arithmetic shift left operation is as follows :

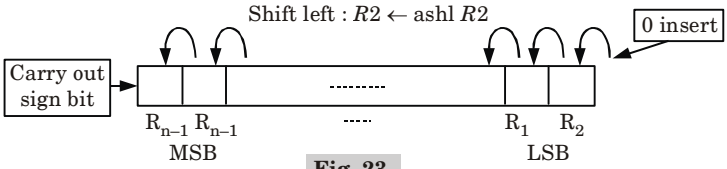


Fig. 23.

v. Fig. 24 showing arithmetic shift right operation is as follows :

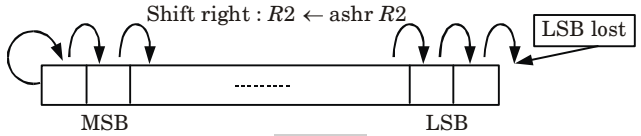


Fig. 24.

b. Logical shift micro-operation :

i. A logical shift micro-operation transfers a 0 (zero) through the serial input, either from left or right depending on the type.

ii. For logical shift left micro-operation, 0 (zero) is transferred through the right of the data and for the logical shift right micro-operation, 0 (zero is transferred through the left of the data as shown in the Fig. 25.

iii. Register Transfer Language (RTL) for the logical shift micro-operations can be written as :
$R \leftarrow$ shl R (shift left register (R)).
$R \leftarrow$ shr R (shift right register (R)).

iv. Fig. 25 showing logical shift left micro-operation on the data in a register.

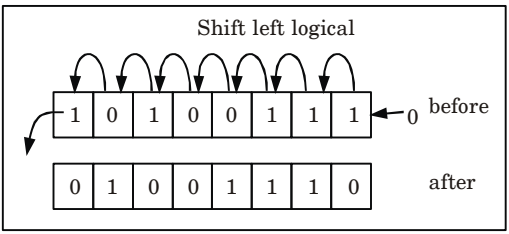


Fig. 25.

v. Fig. 26 showing the logical shift right is as follows :

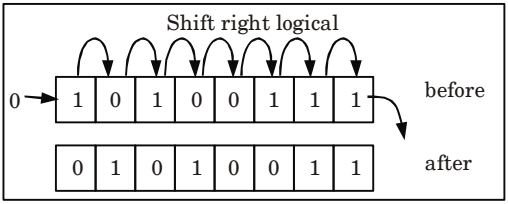


Fig. 26.

c. Circular shift micro-operation :

i. A circular shift micro-operation performs the shifting of bits from one end of the register to the other end of the register.

ii. In circular shift left operation, the leftmost bit in the register is transferred to the rightmost end and in the circular shift right operation, the rightmost bit in the register is transferred or shifted to the leftmost end of the register as shown in the Fig. 28 and Fig. 29 respectively.

iii. Register transfer language for the circular shift micro-operations can be written as :
$R \leftarrow$ cil R (circular shift left register (R)).
$R \leftarrow$ cir R (circular shift right register (R)).

iv. Fig. 28 showing circular shift left micro-operation.

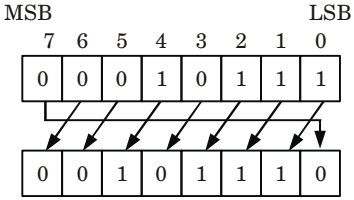


Fig. 27.

v. Fig. 29 showing circular shift right micro-operation.

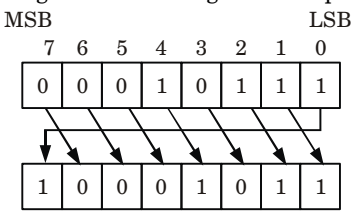


Fig. 28.

☺☺☺

B.Tech.

(SEM. III) ODD SEMESTER THEORY EXAMINATION, 2017-18
COMPUTER ORGANIZATION & ARCHITECTURE

Time : 3 Hours **Max. Marks : 70**

Note : 1. Attempt **all** Sections. If require any missing data; then choose suitably.

SECTION – A

1. Attempt **all** questions in brief. **(2 × 7 = 14)**

a. Draw the circuit diagram of D flip-flop.

b. Write the difference between RAM & ROM.

c. Write short note on pipelining process.

d. Write the difference between serial and parallel communication.

e. Perform the following operation on signed numbers using 2's compliment method : $(56)_{10} + (-27)_{10}$.

f. Write speed up performance laws.

g. Differentiate between horizontal and vertical microprogramming.

SECTION-B

2. Attempt any **three** of the following : **(7 × 3 = 21)**

a. What is programmable logic device ? List various techniques to program PLD. Explain any one technique with example.

b. i. Draw the block diagram for a small accumulator based CPU.

ii. How floating point numbers are represented in computer, also give IEEE 754 standard 32-bit floating point number format.

c. Draw the data path of sequential *n*-bit binary divider. Give the non-restoring division algorithm for unsigned integers.

Also illustrate algorithm for unsigned integer with a suitable example.

d. What is micro programmed control unit ? Give the basic structure of micro programmed control unit. Also discuss the microinstruction format and the control unit organization for a typical micro programmed controllers using suitable diagram.

e. What do you mean by locality of reference ? Explain with suitable example.

SECTION-C

3. Attempt any **one** part of the following : **(7 × 1 = 7)**

a. Differentiate between RISC & CISC based microprocessor.

b. Explain booths multiplication algorithm in detail.

4. Attempt any **one** part of the following : **(7 × 1 = 7)**

a. Draw the data path of 2's compliment multiplier. Give the Robertson multiplication algorithm for 2's compliment fractions. Also illustrate the algorithm for 2's compliment fraction by a suitable example.

b. Describe sequential Arithmetic & Logic Unit (ALU) using proper diagram.

5. Attempt any **one** part of the following : **(7 × 1 = 7)**

a. Give the structure of commercial 8M × 8 bit DRAM chip.

b. Explain the working of DMA controller with help of suitable diagrams.

6. Attempt any **one** part of the following : **(7 × 1 = 7)**

a. What is hardwired control ? List various design methods for hardwired control. Discuss in detail using diagram any one of the method for designing GCD processor.

b. How pipeline performance can be measured ? Discuss. Give a space time diagram for visualizing the pipeline behaviour for a four stage pipeline.

7. Attempt any **one** part of the following : **(7 × 1 = 7)**

a. Discuss the various types of address mapping used in cache memory.

b. A moving arm disc storage device has the following specifications :
Number of Tracks per recording surface = 200
Disc rotation speed = 2400 revolution/minute
Track-storage capacity = 62500 bits
Estimate the average latency and data transfer rate of this device.

SOLUTION OF PAPER (2017-18)

Note : **1.** Attempt **all** Sections. If require any missing data; then choose suitably.

SECTION – A

1. Attempt **all** questions in brief. **(2 × 7 = 14)**

a. Draw the circuit diagram of D flip-flop.

Ans. This question is out of syllabus since session 2018-19.

b. Write the difference between RAM & ROM.

Ans.

S. No.	RAM	ROM
1.	A random access memory (RAM) device allows data items to be read or written in almost the same amount of time irrespective of the physical location of data inside the memory.	The read only memory (ROM) is a type of semiconductor memory that is designed to hold data that is either permanent or will not change frequently. It is also known as non-volatile memory.
2.	Types of RAM : i. Dynamic random access memory (DRAM) ii. Static random access memory (SRAM)	Types of ROM : i. Programmable read only memory (PROM) ii. Erasable programmable read only memory (EPROM)

c. Write short note on pipelining process.

Ans. Pipelining means realizing temporal parallelism in an economical way. In this, the problem is divided into a series of tasks that have to be completed one after the other.

d. Write the difference between serial and parallel communication.

Ans.

Basis for comparison	Serial communication	Parallel communication
Meaning	Data flows in bi-direction, bit by bit	Multiple lines are used to send data *i.e.*, 8 bits or 1 byte at a time
Cost	Economical	Expensive
Bits transferred at 1 clock pulse	1 bit	8 bits or 1 byte
Speed	Slow	Fast
Applications	Used for long distance communication. For example, computer to computer.	Short distance. For example, computer to printer.

e. Perform the following operation on signed numbers using 2's compliment method : $(56)_{10} + (-27)_{10}$.

Ans.

$56 = 111000$ (binary form)

$+56 = 0111000$ (signed binary form)

$-27 = 011011$ (binary form)

100100 (1's complement)

$+1$

100101 (2's complement)

$-27 = 1100101$ (signed binary form)

now, $(+56)_{10} + (-27)_{10} = 0111000$

1100101

$\overline{\textcircled{1}0011101}$

$= (0011101)_2$ (signed binary number)

$= (29)_{10}$

f. Write speed up performance laws.

Ans. Speed up performance laws are as follow :

1. **Amdahl's law :** According to Amdahl's law, if the fraction of computation that cannot be divided into concurrent tasks is, f and no overhead occurs when the computation is divided into concurrent parts, the time to perform the computation with n processors is given by,

$$ft_s + (1 - f)\, f_s/n$$

2. **Gustafson's law :** This law states that any sufficiently large problem can be efficiently parallelized, the sequential operations

will no longer be a bottleneck if the number of parallel operations in the problem is scaled-up sufficiently.

3. **Sun and Ni's law :** This law is a generalization of Amdahl's law and Gustafson's law. This model maximizes the use of both CPU and memory capacity.

g. Differentiate between horizontal and vertical microprogramming.

Ans.

S. No.	Horizontal microprogramming	Vertical microprogramming
1.	In horizontal micro-programming, one associates each bit of the micro-instruction with a specific micro-operation (bit I to represent micro-operation I).	In the case of vertical micro-programming, each line of the micro-program represents a micro-instruction which specifies one or more micro-operations.
2.	A specific micro-operation is executed during a micro-instruction step only if the corresponding bit is one.	One micro-instruction gets executed during each step of the control sequence. One can use a straight binary code to specify each micro-operation.

SECTION-B

2. Attempt any **three** of the following : **(7 × 3 = 21)**

a. What is programmable logic device ? List various techniques to program PLD. Explain any one technique with example.

Ans.

1. PLDs are semiconductor devices that can be programmed to obtain required logic device.
2. Because of the advantage of re-programmability, they have replaced special purpose logic devices like logic gates, flip-flops, counters and multiplexers in many semicustom applications.
3. It consists of arrays of AND and OR gates, which can be programmed to realize required logic function.
4. The process of entering the information into these devices is known as programming.
5. Basically, users can program these devices or ICs electrically in order to implement the Boolean functions based on the requirement.

Various techniques to program PLD are :

1. Programmable Read Only Memory (PROM)
2. Programmable Logic Array (PLA)
3. Programmable Array Logic (PAL)

i. PAL is a programmable logic device that has Programmable AND array & fixed OR array.

ii. The advantage of PAL is that we can generate only the required product terms of Boolean function instead of generating all the min terms by using programmable AND gates.

iii. The block diagram of PAL is shown in Fig. 1.

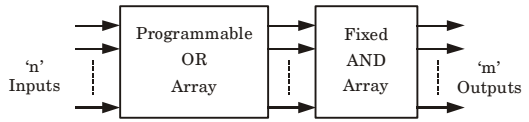


Fig. 1.

Example :

Full adder using PAL : There are two functions used for the implementation of full adder :

$$S = \bar{A}\bar{B}C + \bar{A}B\bar{C} + A\bar{B}\bar{C} + ABC$$

$$C = AB\bar{C} + A\bar{B}C + \bar{A}BC + ABC$$

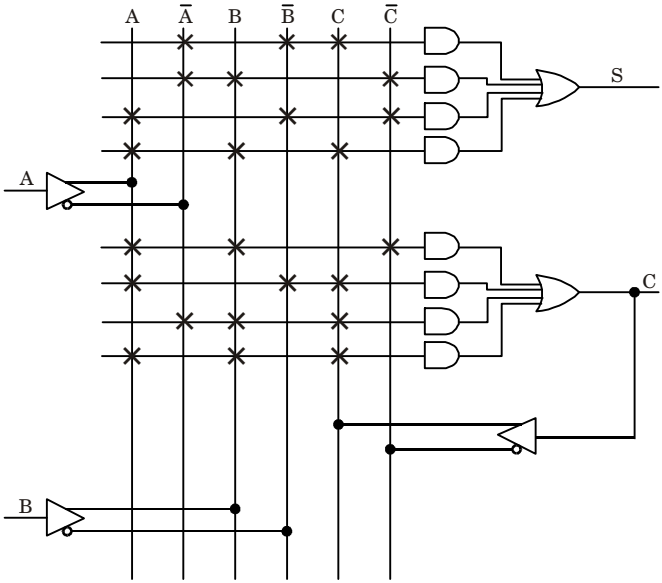


Fig. 2.

b. i. Draw the block diagram for a small accumulator based CPU.

ii. How floating point numbers are represented in computer, also give IEEE 754 standard 32-bit floating point number format.

Ans.

i.

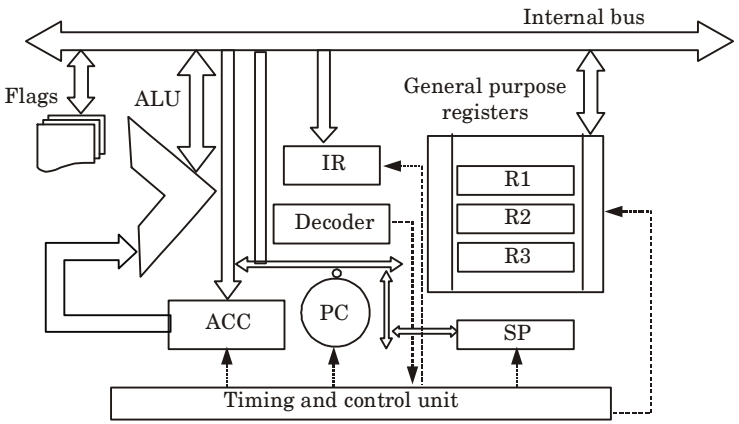
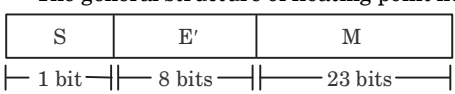


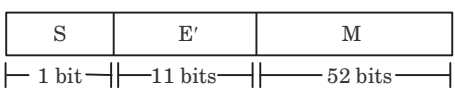
Fig. 3. Block diagram of accumulator based CPU organization.

ii. Floating point number : The floating point representation has three fields :

1. **The sign bit :** The sign bit determines whether the number is negative or positive. 0 denotes a positive number and 1 denotes a negative number.
2. **The exponent :** The exponent field needs to represent both positive and negative exponents. To do this, a bias is added to the actual exponent in order to get the stored exponent. For IEEE single precision the exponent field is of 8 bits and has a bias value of 127. For double precision, the exponent field is of 11 bits, and has a bias of 1023.
3. **The mantissa :** The mantissa, also known as the significand, represents the precision bits of the number. It is composed of an implicit leading bit and the fraction bits.

The general structure of floating point number is





Where S is significant (mantissa) digits, E is exponent, B is scaling factor, which is 2 for binary number, 10 for decimal number.

c. Draw the data path of sequential n-bit binary divider. Give the non-restoring division algorithm for unsigned integers.

Also illustrate algorithm for unsigned integer with a suitable example.

Ans. **Datapath of sequential *n*-bit binary divider :**

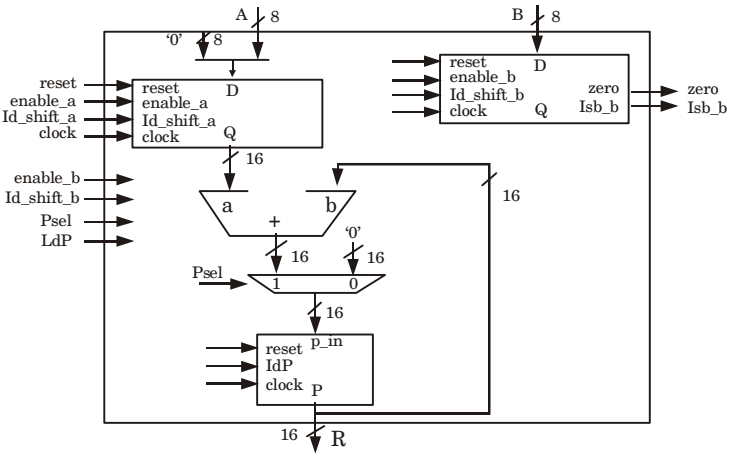


Fig. 4.

Algorithm for non-restoring division :

Restoring division operation :

Step 1 : Shift A and Q left one binary position.

Step 2 : Subtract divisor from A and place answer back in $A(A \leftarrow A - B)$.

Step 3 : If the sign bit of A is 1, set Q_0 to 0 and add divisor back to A (that is, restore A); otherwise, set Q_0 to 1.

Step 4 : Repeat steps 1, 2 and 3 upto n times.

Non-restoring division operation :

Step 1 : If the sign of A is 0, shift A and Q left one bit position and subtract divisor from A; otherwise, shift A and Q left and add divisor to A.

Step 2 : If the sign of A is 0, set Q_0 to 1; otherwise, set Q_0 to 0.

Step 3 : Repeat steps 1 and 2 for n times.

Step 4 : If the sign of A is 1, add divisor to A. Step 4 is required to leave the proper positive remainder in A at the end of n cycles.

For example, consider 4-bit dividend and 2-bit divisor :

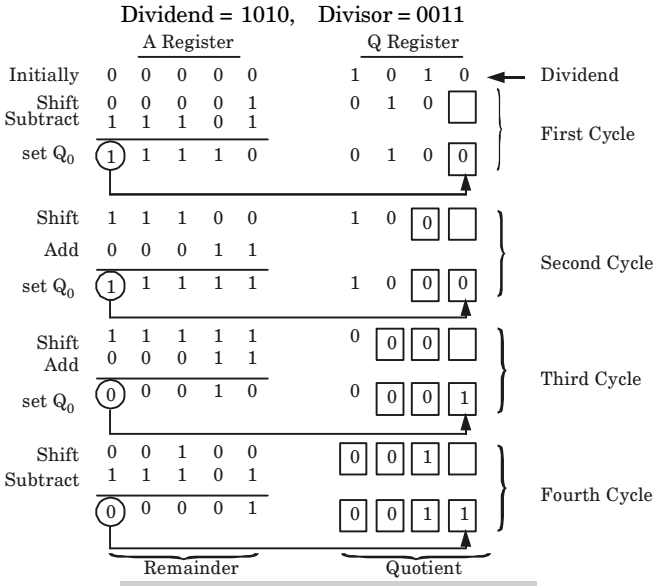


Fig. 5. A non-restoring division example.

In given example after 4 cycles register *A* is positive and hence step 3 is not required.

d. What is micro programmed control unit ? Give the basic structure of micro programmed control unit. Also discuss the microinstruction format and the control unit organization for a typical micro programmed controllers using suitable diagram.

Ans. **Micro-programmed control unit and its structure :**

1. Micro-programmed control is a method of control unit design in which the control signal selection and sequencing information is stored in a ROM or RAM called a Control Memory (CM).
2. The sequence of control signals to be generated by the controller can be stored in a special Read Only Memory (ROM) also called Control Memory (CM).
3. Memory control word is written for each micro-operation, and these control words are stored in a serial ascending memory location.
4. The control word is accessed serially (serial access memory) from the control memory.
5. Control words are stored in the ROM permanently.
6. The output of the control memory provides the required control signals.

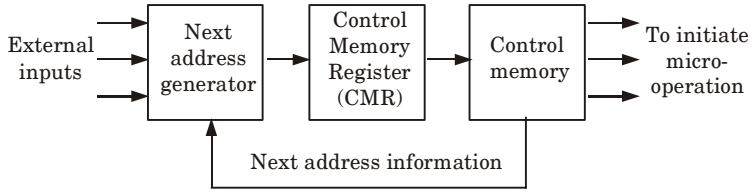


Fig. 6. Block diagram of micro-programmed control unit.

7. If the control memory is sequentially accessed by incrementing control memory location, then the sequence of control signals stored in successive word of ROM can be generated.
8. The storage of control word in a ROM is often referred to as firmware.

Micro-instruction formats : The micro-instruction format for the control memory is shown in the Fig. 7.

F1, F2, F3 : Micro-operation fields
CD : Condition for branching
BR : Branch field
AD : Address field

Fig. 7. Micro-instruction format.

The 20 bits of the micro-instruction are divided into four functional parts as follows :

1. The three fields F1, F2, and F3 specify micro-operations for the computer. The micro-operations subdivided into three fields of three bits each. The three bits in each field are encoded to specify seven distinct micro-operations. So, this gives a total of 21 micro-operations.
2. The CD field selects status bit conditions.
3. The BR field specifies the type of branch to use.
4. The AD field contains a branch address. The address field is seven bits wide since the control memory has 128 = 27 words.

Organization of micro-programmed control unit :

1. The general configuration of a micro-programmed control unit is demonstrated in the block diagram of Fig. 8.
2. The control memory is assumed to be a ROM, within which all control information is permanently stored.

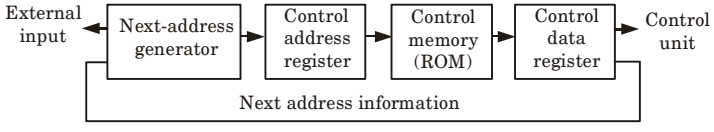


Fig. 8. Micro-programmed control organization.

3. The control memory address register specifies the address of the micro-instruction, and the control data register holds the micro-instruction read from memory.
4. The micro-instruction contains a control word that specifies one or more micro-operations for the data processor. Once these operations are executed, the control must determine the next address.
5. The location of the next micro-instruction may be the one next in sequence, or it may be located somewhere else in the control memory.
6. While the micro-operations are being executed, the next address is computed in the next address generator circuit and then transferred into the control address register to read the next micro-instruction.
7. Thus a micro-instruction contains bits for initiating micro-operations in the data processor part and bits that determine the address sequence for the control memory.
8. The next address generator is sometimes called a micro-program sequencer, as it determines the address sequence that is read from control memory.
9. Typical functions of a micro-program sequencer are incrementing the control address register by one, loading into the control address register an address from control memory, transferring an external address, or loading an initial address to start the control operations.
10. The control data register holds the present micro-instruction while the next address is computed and read from memory.

e. What do you mean by locality of reference ? Explain with suitable example.

Ans. Locality of reference is a term for the phenomenon in which the same values or related storage locations are frequently accessed, depending on the memory access pattern.

Example :

1. Take the example of an operating system. Ideally, we would like an unlimited amount of main memory, instantly accessible.
2. In practice, we have a limited amount of main memory, and because it is cheaper, a very large amount of secondary memory.
3. However the trade-off is that secondary memory tends to be several orders of magnitude slower than primary memory.
4. We can approach the ideal by keeping the more often used data in main memory, and everything else in secondary memory.
5. Because of the principle of Locality of Reference, we can be sure that most memory references will be to locations already stored in main memory, thereby improving efficiency and providing a flat memory model.
6. This scheme is used in modern operating systems and is called virtual memory. Virtual memory gives users the appearance of unlimited primary memory by transparently utilizing secondary memory.

SECTION-C

3. Attempt any **one** part of the following : **(7 × 1 = 7)**

a. Differentiate between RISC & CISC based microprocessor.

Ans.

S. No.	RISC	CISC
1.	Multiple register sets, often consisting of more than 256 registers.	Single register set, often consisting 6 to 16 registers total.
2.	Three register operands allowed per instruction (for example, add R1, R2, R3).	One or two register operands allowed per instruction (for example, add R1, R2).
3.	Parameter passing through efficient on-chip register windows.	Parameter passing through inefficient off-chip memory.
4.	Single-cycle instructions (except for load and store).	Multiple-cycle instructions.
5.	Hardwired control.	Micro-programmed control.
6.	Highly pipelined.	Less pipelined.
7.	Simple instructions are few in number.	There are many complex instructions.
8.	Fixed length instructions.	Variable length instructions.
9.	Complexity in compiler.	Complexity in microcode.
10.	Only load and store instructions can access memory.	Many instructions can access memory.
11.	Few addressing modes.	Many addressing modes.

b. Explain booths multiplication algorithm in detail.

Ans. The algorithm for 2's complement multiplication is as follows :

Step 1 : Load multiplicand in B, multiplier in Q. For negative numbers, 2's complement format to be used.

Step 2 : Initialize the down counter CR by the number of bits involved.

Step 3 : Clean locations A (n-bits) and Q_{n+1} (1-bit).

Step 4 : Check LS bit of Q_n and Q_{n+1} jointly. If the pattern is 00 or 11 then go to Step 5. If 10, then $A = A - B$. If 01, then $A = A + B$.

Step 5 : Perform arithmetic right-shift with A, Q_n and Q_{n+1}. LS of A goes to MS of Q_n and LS of Q goes to Q_{n+1}. Old content of Q_{n+1} is discarded.

Step 6 : Decrement CR by one. If CR is not zero then go to Step 4.
Step 7 : Final result (or the product) is available in A (higher part) and Q_n (lower part).

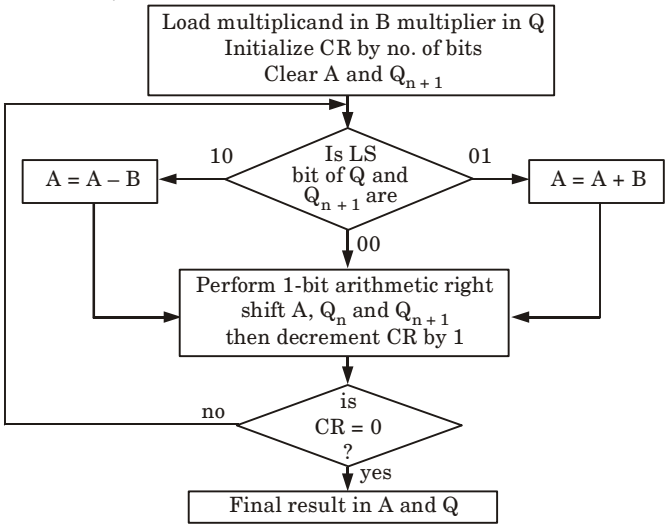


Fig. 9. 2's complement multiplication.

Example : Both negative (– 5 × – 4)

Multiplicand (B) ← 1 0 1 1 (– 5)			**Multiplier (Q) ← 1 1 0 0 (– 4)**	
A	***Q_n***	***Q_{n+1}***	**Operation**	**CR**
0 0 0 0	1 1 0 0	0	Initial	4
0 0 0 0	0 1 1 0	0	Shift right	3
0 0 0 0	0 0 1 1	0	Shift right	2
0 1 0 1	0 0 1 1	0	A ← A – B	1
0 0 1 0	1 0 0 1	1	Shift right	
0 0 0 1	0 1 0 0	1	Shift right	0
Result : 0 0 0 1	0 1 0 0	= + 20		

4. Attempt any **one** part of the following : **(7 × 1 = 7)**

a. Draw the data path of 2's compliment multiplier. Give the Robertson multiplication algorithm for 2's compliment fractions. Also illustrate the algorithm for 2's compliment fraction by a suitable example.

Ans. Data path of 2's compliment multiplier :

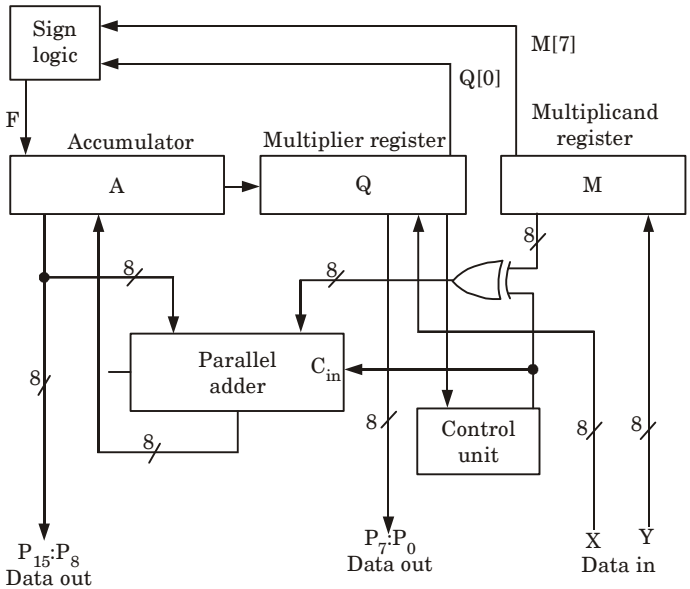


Fig. 10.

Robertson algorithm :

1. $A \leftarrow 0, B \leftarrow$ Multiplicand $Q \leftarrow$ Multiplier and count $\leftarrow n$.
2. If $Q_0 = 1$ then perform $A \leftarrow A + B$.
3. Shift right register $F.A.Q$ by 1 bit $F \leftarrow B\ [n-1]$ AND $Q[0]$ OR F and count $\leftarrow$ count $- 1$.
4. If count > 1 Repeat steps 2 and 3, otherwise if $Q_0 = 1$ then perform $A \leftarrow A - B$ and set $Q\ [0] = 0$.

For example : We perform the multiplication of fraction as :

Multiplicand = 0.625

Multiplier = 0.5

Equivalent binary representation 0.625 = 0101

2's complement representation – 0.625 = 1010 + 1 (– 0.625) = 1011

Equivalent binary representation + 0.5 = 0100

2 complement representation – 0.5 = 1011 + 1 – 0.5 = 1100

Steps	F	A	Q	Comments
		B = 0 1 0 1		
	0	0 0 0 0	0 1 0 0	
Step 1	0	0 0 0 0	0 1 0 0	Q_0 = 0, No need addition
	0	0 0 0 0	0 0 1 0	1 bit right shift
Step 2	0	0 0 0 0	0 0 1 0	Q_0 = 0, No need addition
	0	0 0 0 0	0 0 0 1	1 bit right shift
Step 3	0	0 1 0 1	0 0 0 1	Q_0 = 1, A ← A + B
	0	0 0 1 0	1 0 0 0	1 bit right shift
Step 4	0	0 0 1 0	1 0 0 0	Q_0 = 0, No need addition
Final		Final product		

$0.625 \times 0.5 = 0.3125$

$0101 \times 0100 = 00101000$

b. Describe sequential Arithmetic & Logic Unit (ALU) using proper diagram.

Ans.

1. By combining arithmetic and logic circuits with the help of multiplexer, we can get the arithmetic and logic units.

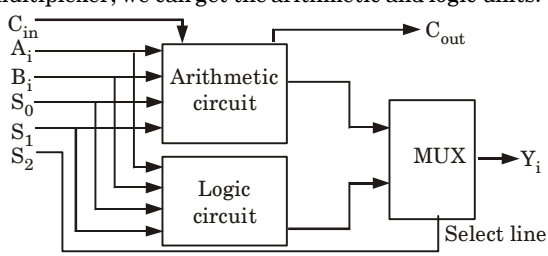


Fig. 11. Block diagram of ALU.

2. When the mode select line $S_2 = 0$, this ALU acts as an arithmetic circuit, so the output of arithmetic circuit is transferred as final output.
3. Otherwise ($S_2 = 1$) the output of the logic circuit is transferred as final output.
4. Based on the mode select S_2 and input carry C_{in}, we increase or decrease the number of arithmetic and logic operations.
5. When $S_2 = 0$, the ALU performs arithmetic operation and when $S_2 = 1$, with $C_{in} = 0$, the ALU performs logic operations.
6. We know that the carry input is not required in the logic circuits.

7. When logic operation is selected ($S_2 = 1$), the carry input must be zero.
8. This given us the output sum in full adder circuit as

$$Y_i = A_i \oplus B_i \oplus C_i \qquad (\because \ C_i = 0)$$
$$Y_i = A_i \oplus B_i$$

5. Attempt any **one** part of the following : **(7 × 1 = 7)**

a. Give the structure of commercial 8M × 8 bit DRAM chip.

Ans.

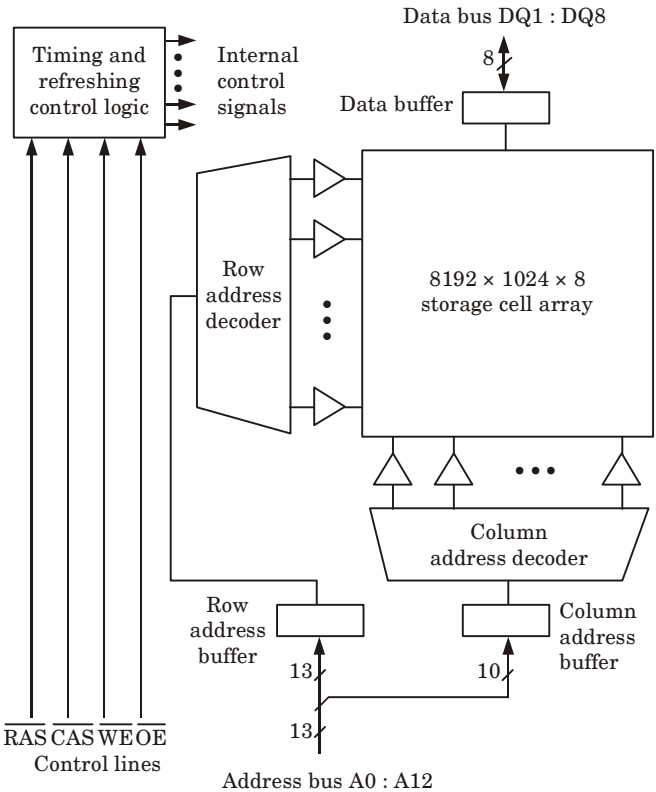


Fig. 12. Structure of a commerical 8M × 8-bit DRAM chip.

b. Explain the working of DMA controller with help of suitable diagrams.

Ans. **Working of DMA controller :**

1. When the peripheral device sends a DMA request, the DMA controller activates the BR line, informing the CPU to relinquish the buses.

2. The CPU responds with its BG line, informing the DMA that its buses are disabled.
3. The DMA then puts the current value of its address register into the address bus, initiates the RD or WR signal, and sends a DMA acknowledge to the peripheral device.
4. The direction of transfer depends on the status of the BG line.

a. When BG = 0, the RD and WR are input lines allowing the CPU to communicate with the internal DMA registers.

b. When BG = 1, the RD and WR are output lines from the DMA controller to the random access memory to specify the read or write operation for the data.

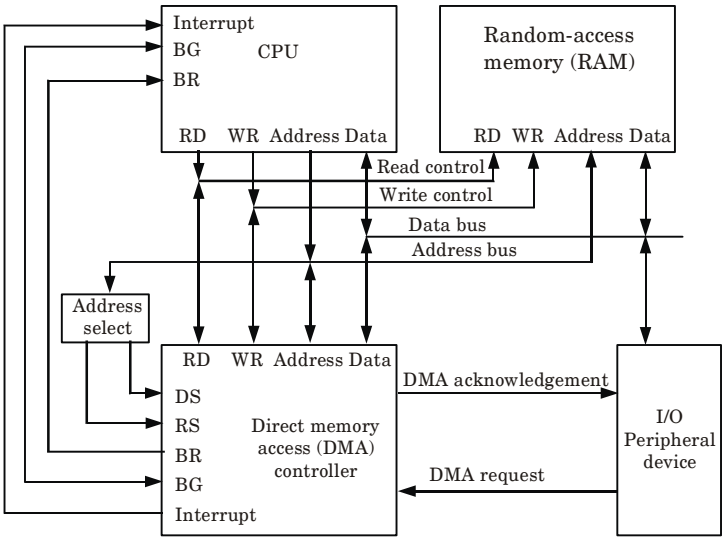


Fig. 13. DMA transfer in a computer system.

5. When the peripheral device receives a DMA acknowledge, it puts a word in the data bus (for write) or receives a word from the data bus (for read).
6. Thus, the DMA controls the read or write operations and supplies the address for the memory.
7. The peripheral unit can communicate with memory through the data bus for direct transfer between the two units while the CPU is momentarily disabled.

6. Attempt any **one** part of the following : **(7 × 1 = 7)**

a. What is hardwired control ? List various design methods for hardwired control. Discuss in detail using diagram any one of the method for designing GCD processor.

Ans. **Hardwired control and design methods :**

1. It is a controller as a sequential logic circuit or a finite state machine that generates a sequence of control signals in response to the externally supplied instructions.
2. The control logic is implemented with gates, flip-flops, decoders, and other digital circuits.

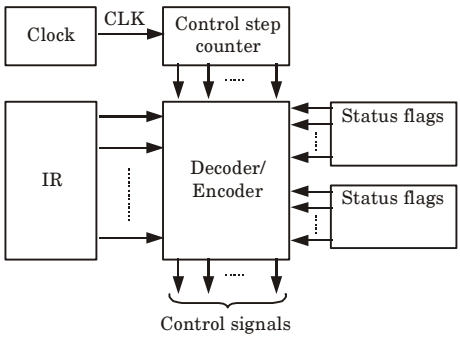


Fig. 14. General block diagramof hardwired control.

Methods to design hardwired control : There are four simplified and systematic methods for the design of hardwired controllers.

1. State table method or one-hot method.
2. Delay element method.
3. Sequence-counter method.
4. PLA method.

FSM based design of GCD processor :

1. First write algorithms for calculation of GCD, after that data paths and control unit is designed for GCD processor.
2. In this model, both the FSM and the data path circuits are manually constructed as separate units.
3. The FSM and the data path are connected together in an enclosing unit using the control and status signals.

```
1.  Input X
2.  Input Y
3.  While (X! = Y)
4.  {
5.  IF (X > Y)
6.  THEN X = X – Y
7.  ELSE Y = Y – X
8.  END IF
9.  }
10. OUTPUT X
```

Fig. 15. Euclid's algorithm.

4. The algorithm shown in Fig. 15 has five data manipulation statements in lines 1, 2, 5, 7, and 10.
5. There are two conditional tests in lines 3 and 4.
6. We can conclude that the data path requires two 8-bit registers, one for variable *X*, and one for variable *Y*, and a subtractor.
7. The dedicated data path is shown in Fig. 15.
8. In Fig. 16, we have a 2-to-1 mux for the input of each register because for each register, we need to initially load it with an input number, and subsequently load it with the result from the subtractor.
9. The two control signals, In_*X* and In_*Y*, select which of the two sources are to be loaded into the registers *X* and *Y* respectively.
10. The two control signals, load_*X* and load_*Y*, load a value into the respective register.

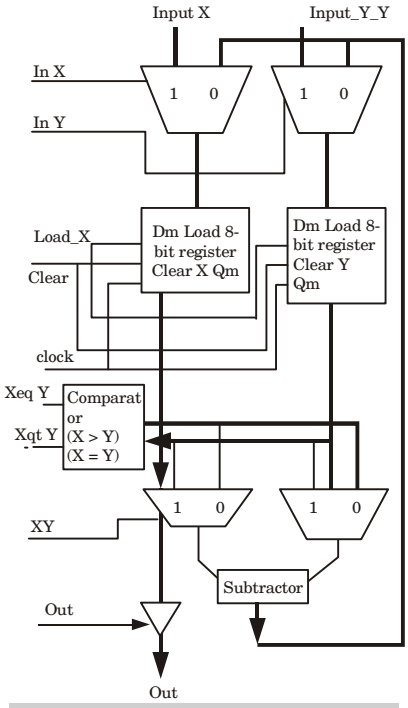


Fig. 16. Datapaths of GCD processor.

b. How pipeline performance can be measured ? Discuss. Give a space time diagram for visualizing the pipeline behaviour for a four stage pipeline.

Ans. **Pipeline performance :**

There are following terms which are used to measure a pipeline performance :

1. Speed-up 2. Efficiency 3. Throughput

Consider a 'k' segment pipeline with clock cycle time as 'T_p'. Let there be 'n' tasks to be completed in the pipelined processor. Now, the first instruction is going to take 'k' cycles to come out of the pipeline but the other '$n - 1$' instructions will take only '1' cycle each, *i.e.*, a total of '$n - 1$' cycles. So, time taken to execute 'n' instructions in a pipelined processor :

$$ET_{\text{pipeline}} = k + n - 1 \text{ cycles} = (k + n - 1)\, T_p$$

In the same case, for a non-pipelined processor, execution time of 'n' instructions will be :

$$ET_{\text{non-pipeline}} = n * k * T_p$$

So, speed-up (S) of the pipelined processor over non-pipelined processor, when 'n' tasks are executed on the same processor is :

S = Performance of pipelined processor / Performance of non-pipelined processor

As the performance of a processor is inversely proportional to the execution time, we have,

$$S = ET_{\text{non-pipeline}} \;/\; ET_{\text{pipeline}}$$
$$S = [n * k * T_p] \;/\; [(k + n - 1) * T_p]$$
$$S = [n * k] \;/\; [k + n - 1]$$

When the number of tasks 'n' are significantly larger than k, that is, $n >> k$

$$S \approx n * k / n \approx k$$

where 'k' are the number of stages in the pipeline.

Also, Efficiency = Given speed-up / Max speed up = $S / S_{\max}$

We know that, $S_{\max} = k$

So, Efficiency = S / k

Throughput = Number of instructions / Total time to complete the instructions

So, Throughput = $n / (k + n - 1) * T_p$

Space time graph :

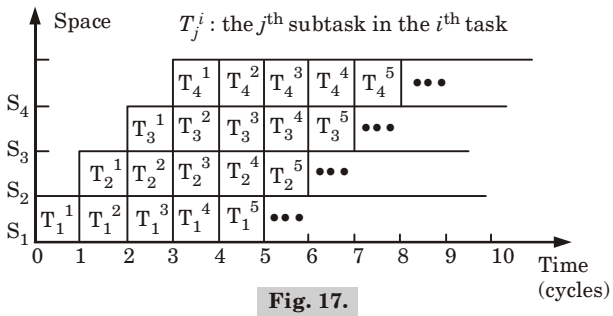


Fig. 17.

7. Attempt any **one** part of the following : **(7 × 1 = 7)**

a. Discuss the various types of address mapping used in cache memory.

Ans. **Types of address mapping :**

1. Direct mapping :

a. The direct mapping technique is simple and inexpensive to implement.

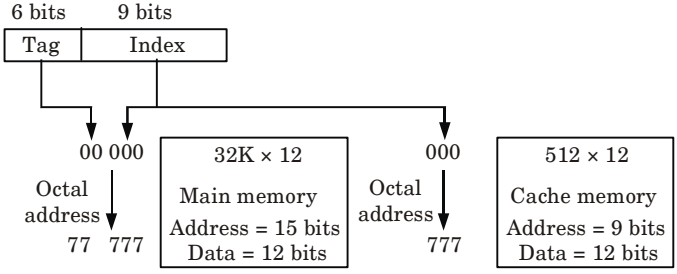


Fig. 18.

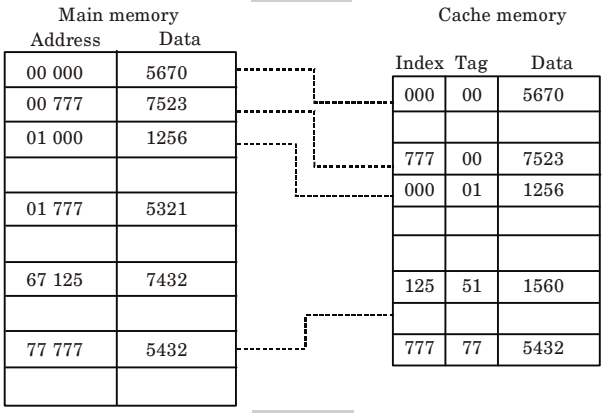


Fig. 19.

b. When the CPU wants to access data from memory, it places an address. The index field of CPU address is used to access address.

c. The tag field of CPU address is compared with the associated tag in the word read from the cache.

d. If the tag-bits of CPU address are matched with the tag-bits of cache, then there is a hit and the required data word is read from cache.

e. If there is no match, then there is a miss and the required data word is stored in main memory. It is then transferred from main memory to cache memory with the new tag.

2. Associative mapping :

a. An associative mapping uses an associative memory.

b. This memory is being accessed using its contents.

c. Each line of cache memory will accommodate the address (main memory) and the contents of that address from the main memory.

d. That is why this memory is also called Content Addressable Memory (CAM). It allows each block of main memory to be stored in the cache.

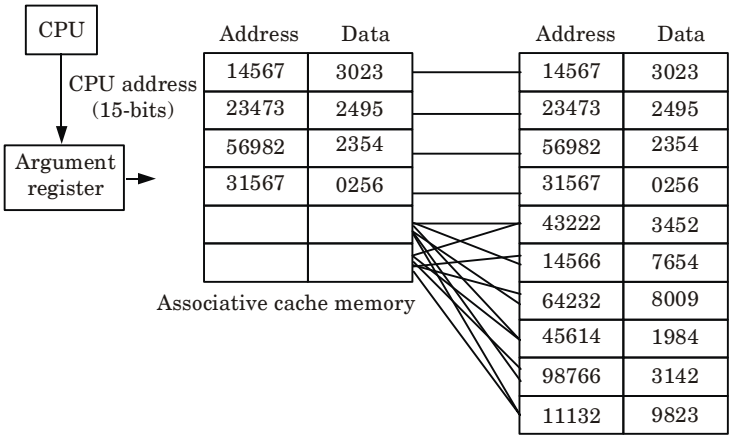


Fig. 20.

3. Set associative mapping :

a. In set associative mapping, each cache location can have more than one pair of tag + data items.

b. It combines the best of the direct mapping cache and the more flexible mapping of the fully associative cache.

c. That is more than one pair of tag and data are residing at the same location of cache memory. If one cache location is holding two pair of tag + data items, that is called 2-way set associative mapping.

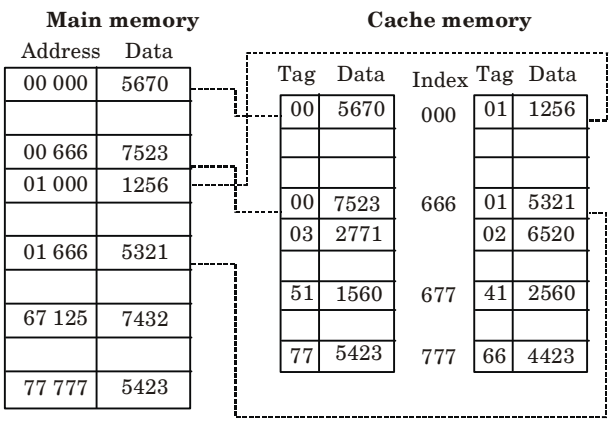


Fig. 21.

b. A moving arm disc storage device has the following specifications :
Number of Tracks per recording surface = 200
Disc rotation speed = 2400 revolution/minute
Track-storage capacity = 62500 bits
Estimate the average latency and data transfer rate of this device.

Ans. Disk rotation speed = 2400 rpm

As we know that average latency = $\frac{1}{2}$ × Rotation time

$\because$ 2400 rotation in one minute so the time for one rotation

$$= \frac{1}{2} * \frac{60}{2400} \text{ s} = \frac{3}{240} \text{ s} = 12.5 \text{ ms}$$

Track storage capacity = 62500 bits
And in one rotation head cover entire track, so, disk transfer rate

$$= 62500 * \frac{2400}{60} \text{ s}$$

(where 2400/60 is number of rotations per second)

$$= 2.5 * 10^6 \text{ bps}$$

☺☺☺

B.Tech.

(SEM. III) ODD SEMESTER THEORY EXAMINATION, 2018-19 COMPUTER ORGANIZATION & ARCHITECTURE

Time : 3 Hours **Max. Marks : 70**

Note : 1. Attempt **all** Sections. If require any missing data; then choose suitably.

SECTION – A

1. Attempt **all** questions in brief. **(2 × 7 = 14)**

a. What do you understand by locality of reference ?

b. Which of the following architecture is/are not suitable for realizing SIMD ?

c. What is the difference between RAM and DRAM ?

d. What are the difference between horizontal and vertical micro codes ?

e. Describe cycle stealing in DMA.

f. List three types of control signals.

g. Define the role of MIMD in computer architecture.

SECTION-B

2. Attempt any **three** of the following : **(7 × 3 = 21)**

a. Evaluate the arithmetic statement $X = (A + B)*(C + D)$ using a general register computer with three address, two address and one address instruction format a program to evaluate the expression.

b. Perform the division process of 00001111 by 0011 (use a dividend of 8 bits).

c. A two way set associative cache memory uses blocks of 4 words. The cache can accommodate total of 2048 words from memory. The main memory size is 128 K × 32.

i. Formulate all pertinent information required to construct the cache memory.
ii. What is the size of cache memory ?

d. What is associative memory ? Explain with the help of a block diagram. Also mention the situation in which associative memory can be effective utilized.

e. A computer uses a memory unit with 256 *K* words of 32 bits each. A binary instruction code is stored in one word of memory. The instruction has four parts : an indirect bit, an operation code, a register code part to specific one of 64 register and an address part.
i. How many bits are there in the operation code, the register code part and the address part ?
ii. Draw the instruction word format and indicate the number of bits in each part.
iii. How many bits are there in the data and address inputs of the memory ?

SECTION-C

3. Attempt any **one** part of the following : **(7 × 1 = 7)**
a. Write short notes on :
i. Instruction pipeline
ii. DMA based data transfer

b. Explain the difference between vectored and non-vectored interrupt. Explain stating examples of each.

4. Attempt any **one** part of the following : **(7 × 1 = 7)**
a. Draw the flowchart of Booth's algorithm for multiplication and show the multiplication process using Booth's algorithm for (– 7) × (+ 3).

b. Write short notes on :
i. Amdahl's law **ii. Pipelining**

5. Attempt any **one** part of the following : **(7 × 1 = 7)**
a. What is a microprogram sequencer ? With block diagram, explain the working of microprogram sequencer.

b. Draw a flowchart for adding and subtracting two fixed point binary numbers where negative numbers are signed 1's complement presentation.

6. Attempt any **one** part of the following : **(7 × 1 = 7)**

a. Give the block diagram of DMA controller. Why are the read and write control lines in a DMA controller bidirectional ?

b. Explain all the phases of instruction cycle.

7. Attempt any **one** part of the following : **(7 × 1 = 7)**

a. Explain the basic concept of hardwired and software control unit with neat diagrams.

b.

1	2	3	4	5	6	
S1	X					X
S2		X			X	
S3			X			
S4				X		
S5		X				X

For the following reservation table :

i. Calculate the set of the forbidden latencies and collision vector.

ii. Draw a state diagram, showing all possible initial sequences (cycles) without a collision in the pipeline.

iii. Simple Cycles (SC)

iv. Greedy cycles among simple cycles

v. MAL (Minimum Average Latency)

vi. What is the minimum allowed constant cycles ?

vii. Maximum throughput

viii. Throughput if the minimum constant cycle is used.

☺☺☺

SOLUTION OF PAPER (2018-19)

SECTION – A

1. Attempt **all** questions in brief. **(2 × 7 = 14)**

a. What do you understand by locality of reference ?

Ans. Locality of reference is a term for the phenomenon in which the same values or related storage locations are frequently accessed, depending on the memory access pattern.

b. Which of the following architecture is/are not suitable for realizing SIMD ?

Ans. Von-neumann architecture is not suitable for realizing SIMD.

c. What is the difference between RAM and DRAM ?

Ans.

S. No.	Static RAM	Dynamic RAM
1.	Static RAM contains less memory cells per unit area.	Dynamic RAM contains more memory cells as compared to static RAM per unit area.
2.	It has less access time hence faster memories.	Its access time is greater than static RAMs.
3.	Cost is more.	Cost is less.

d. What are the difference between horizontal and vertical micro codes ?

Ans.

S. No.	Horizontal micro code	Vertical micro code
1.	In this types of code the micro code contains the control signal without any intermediary.	In case of vertical micro code every action is encoded in density.
2.	Horizontal micro code instruction contain a lot of signals and hence due to that the number of bits also increase.	Vertical micro code are slower but they take less space and their actions at execution time need to be decoded to a signal.

e. Describe cycle stealing in DMA.

Ans.

1. In Direct Memory Access (DMA), cycle stealing is a method of allowing I/O controllers to read or write RAM without interfering with the CPU.
2. DMA controllers can operate in cycle stealing mode in which controller take over the bus for each byte of data to be transferred and then return control to the CPU.

f. List three types of control signals.

Ans. **Three types of control signals are :**

i. ALU
ii. Data paths
iii. System

g. Define the role of MIMD in computer architecture.

Ans. The role of MIMD is to provide an environment for the computer processors to operates on programs which have their own instruction and data.

SECTION-B

2. Attempt any **three** of the following : **(7 × 3 = 21)**

a. Evaluate the arithmetic statement $X = (A + B)*(C + D)$ using a general register computer with three address, two address and one address instruction format a program to evaluate the expression.

Ans.

Three address instruction :

ADD	R1, A, B	R1 ← M[A] + M[B]
ADD	R2, C, D	R2 ← M[C] + M[D]
MUL	X, R1, R2	M[X] ← R1 * R2

Two address instruction :

MOV	R1, A	R1 ← M[A]
ADD	R1, B	R1 ← R1 + M[B]
MOV	R2, C	R2 ← M[C]
ADD	R2, D	R2 ← R2 + M[D]
MUL	R1, R2	R1 ← R1 * R2
MOV	X, R1	M[X] ← R1

One address instruction :

LOAD	A	AC ← M[A]
ADD	B	AC ← A[C] + M[B]
STORE	T	M[T] ← AC
LOAD	C	AC ← M[C]
ADD	D	AC ← AC + M[D]
MUL	T	AC ← AC * M[T]
STORE	X	M[X] ← AC

b. Perform the division process of 00001111 by 0011 (use a dividend of 8 bits).

Ans.

B = 0011 $\quad \overline{B} + 1 = 1101$

Operation	E	A	Q	SC
Dividend in Q, A = 0 shl EAQ add $\overline{B} + 1$	0	0000 0001 1101	1111 1110	100
E = 0, leave $Q_n = 0$ add B restore partial remainder shl EAQ add $\overline{B} + 1$	0 1 0	1110 0011 0001 0011 1101	1110 1100	011
E = 1, set Q_n to 1 shl EAQ add $\overline{B} + 1$	1 0	0000 0001 1101	1101 1010	010
E = 0, leave $Q_n = 0$ add B restore partial remainder	0 1	1110 0011 0001	1010	001
shl EAQ add $\overline{B} + 1$ E = 1, set Q_n to 1	0 1	0011 1101 0000 Remainder	0100 0101 Quotient	000

c. A two way set associative cache memory uses blocks of 4 words. The cache can accommodate total of 2048 words from memory. The main memory size is 128 *K* × 32.

i. Formulate all pertinent information required to construct the cache memory.

ii. What is the size of cache memory ?

Ans.

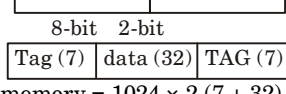
i. Main memory size = 128K × 32 = 2^{17}

Cache size =2048 words

Set size of 2 cache can accommodate = 2048/2 = 1024 words of cache

Block size =4 words

ii.



Size of cache memory = 1024 × 2 (7 + 32) = 1024 × 78

d. What is associative memory ? Explain with the help of a block diagram. Also mention the situation in which associative memory can be effective utilized.

Ans. **Associative memory :**

1. Associative memory is a memory in which location is accessed by a field of data word stored in the memory rather than by any address.
2. It can be viewed as a random access type memory which in addition to having a physically wired-in addressing mechanism also has wired-in logic for bit comparison.
3. This logic circuit enables comparison of desired bit positions of all the words with a specified input key.
4. This comparison is done simultaneously for all the words.
5. This is also called Content Addressable Memory (CAM).

Working principle of associative memory :

1. The mask register specifies the key field.
2. Input data is simultaneously compared with the key field of each word.
3. The select circuit implements two functions :
   a. It stores the word location (s) for which match has occurred.
   b. It reads out the word(s) in predetermined order for the match position (s).
4. Thus, a word stored in the associative memory is a pair (key, Data). Any subfield of the word can be specified as the key.
5. The read or write instruction is preceded by the match instruction having the format.
   Match key, Input data
6. The read/write operation can next be performed on each of the words for which match signal is generated.

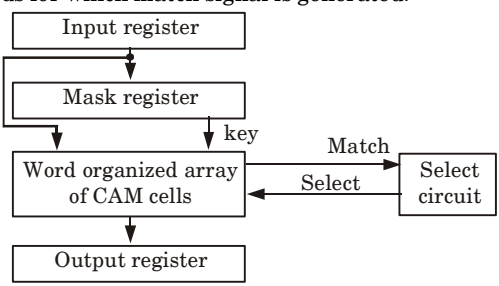


Fig. 1. Organization of an associative memory.

Associative memory is effectively utilized when doing a large number of pattern match and lookup.

e. A computer uses a memory unit with 256 *K* words of 32 bits each. A binary instruction code is stored in one word of memory. The instruction has four parts : an indirect bit, an operation code, a register code part to specific one of 64 register and an address part.

i. How many bits are there in the operation code, the register code part and the address part ?

ii. Draw the instruction word format and indicate the number of bits in each part.

iii. How many bits are there in the data and address inputs of the memory ?

Ans.

a. Address : $2^8 * 2^{10} = 2^{18} = 18$ bits
Register : 64 registers = $2^6 = 6$ bits
OP code : (Total bit – Indirect bit – Address bit – Register bit) = (32 – 1 – 18 – 6) bits = 7 bits

b.

c. Number of bit in address inputs : 18
Number of bit in data inputs : 32

SECTION-C

3. Attempt any **one** part of the following : **(7 × 1 = 7)**

a. Write short notes on :

i. Instruction pipeline

ii. DMA based data transfer

Ans.

i. Instruction pipelining :

a. The execution of a stream of instructions can be pipelined by overlapping the execution of the current instruction with the fetch, decode, and operand fetch of subsequent instructions as shown in Fig. 2.

b. This technique is also known as instruction look ahead.

c. Almost all high-performance computers are now equipped with instruction-execution pipelines.

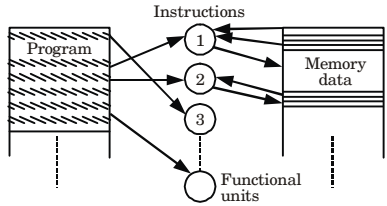


Fig. 2. Instruction pipelining.

ii. DMA based data transfer :

1. DMA stands for "Direct Memory Access" and is a method of transferring data from the computer's RAM to another part of the computer without processing it using the CPU.
2. While most data that is input or output from our computer is processed by the CPU, some data does not require processing, or can be processed by another device.

3. DMA can save processing time and is a more efficient way to move data from the computer's memory to other devices.
4. In order for devices to use direct memory access, they must be assigned to a DMA channel. Each type of port on a computer has a set of DMA channels that can be assigned to each connected device.
5. For example, a PCI controller and a hard drive controller each have their own set of DMA channels.

Block diagram for DMA :

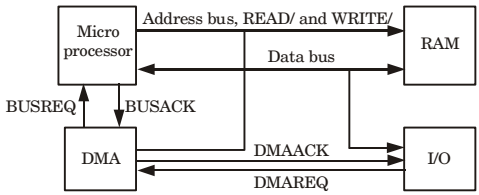


Fig. 3.

b. Explain the difference between vectored and non-vectored interrupt. Explain stating examples of each.

Ans.

S. No.	Vectored interrupt	Non-vectored interrupt
1.	Vectored interrupt are those interrupt that generates the interrupt request, identifies itself directly to the processor.	Non-vectored interrupt are those in which vector address is not pre-defined.
2.	Vector interrupt have fixed memory location for transfer of control for normal execution.	Non-vectored interrupt do not have fixed memory location for transfer of control for normal execution.
3.	Vectored interrupt has memory address.	A non-vectored interrupt do not have memory address.
4.	The vectored interrupt allows the CPU to be able to know what ISR to carry out in software.	When a non-vectored interrupt received, it jump into the program counter to fixed address in hardware.
5.	Response time is low.	Response time is high.
6.	TRAP is a vectored interrupt.	INTR is non-vectored interrupt.

4. Attempt any **one** part of the following : **(7 × 1 = 7)**

a. Draw the flowchart of Booth's algorithm for multiplication and show the multiplication process using Booth's algorithm for (– 7) × (+ 3).

Ans. Flowchart of Booth's algorithm for multiplication :

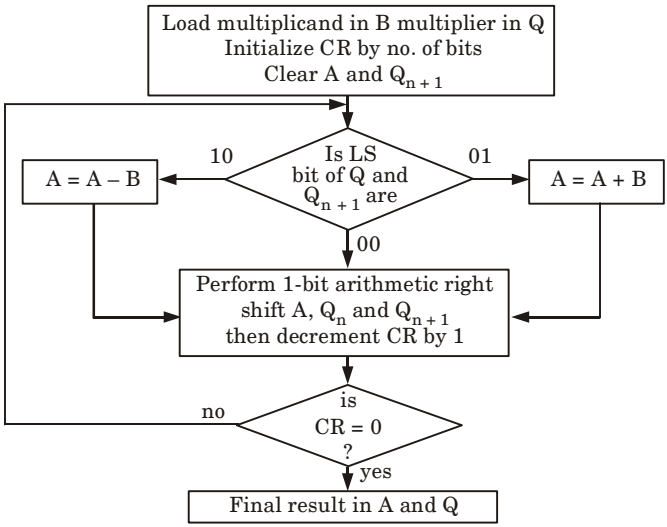


Fig. 4. 2's Complement multiplication.

Multiplication : Multiply (– 7) × (+ 3)

Convert (–7) into 2's complement form :

	+7 = 0111
1's complement of (+7)	= 1000
adding 1	+ 1
2's complement of (+ 7)	= 1001

(+3) = 0011

A	Q_n	Q_{n+1}	$B = 1001$ $\bar{B} + 1 = 0111$	SC
0000	0011	0	initial values	100
0111			sub B or	
0111			add 0111 to A	
0011	1001	1	Ashr $AQ_n\ Q_{n+1}$	011
0001	1100	1	Ashr $AQ_n\ Q_{n+1}$	010
1001			add 1001	
1010				
1101	0110	0	Ashr $AQ_n\ Q_{n+1}$	001
1110	1011	0	Ashr $AQ_n\ Q_{n+1}$	000

Answer is 11101011
$(-7) \times (+3) = -21 = 11101011$ (2's complement of + 21)

b. Write short notes on :

i. Amdahl's law **ii. Pipelining**

Ans.

i. Amdahl's law :

According to Amdahl's law, if the fraction of computation that cannot be divided into concurrent tasks is, f and no overhead occurs when the computation is divided into concurrent parts, the time to perform the computation with n processors is given by,

$$ft_s + (1 - f)\, f_s/n$$

ii. Pipelining :

1. Pipelining is a technique of decomposing a sequential process into sub-operations, with each sub-process being executed in a special dedicated segment that operates concurrently with all other segments.
2. The processor executes a program by fetching and executing instructions, one after the other.
3. Let F_i and E_i refer to the fetch and execute steps for instruction I_i.
4. Execution of a program consists of a sequence of fetch and execute steps as shows is Fig. 5.

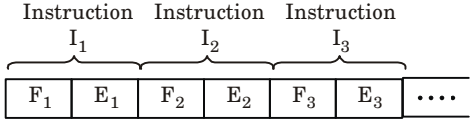


Fig. 5. Sequential execution.

5. Now consider a computer that has two separate hardware units, one for fetching instructions and another for executing them, as shown in Fig. 6.
6. The instruction fetched by the fetch unit is deposited in an intermediated storage buffer B_i.
7. The results of execution are deposited in the destination location specified by the instructions.
8. For these purposes, we assume that both the source and destination of the data operated in by the instructions are inside the block labelled "Execution unit".

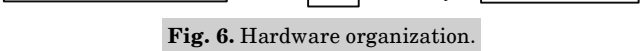


Fig. 6. Hardware organization.

9. The computer is controlled by a clock whose period is such that the fetch and execute steps of any instruction can be completed in one clock cycle.

5. Attempt any **one** part of the following : **(7 × 1 = 7)**

a. What is a microprogram sequencer ? With block diagram, explain the working of microprogram sequencer.

Ans.

1. Micro-program sequencer is a general purpose building block for micro-programmed control unit.
2. The basic components of a micro-programmed control unit are the control memory and the circuit that selects the next address.
3. The address selection part is called micro-program sequencer.
4. The main purpose of micro-program sequencer is to present an address to the control memory so that micro-instruction may be read and executed.
5. The next address logic of the sequencer determines the specific address source to be loaded into the control address register.
6. The choice of the address source is guided by the next address information bits that sequencer receives from the present micro-instruction.
7. All the instructions are loaded in the control memory.
8. The present micro-instruction is placed in micro-instruction register for execution.

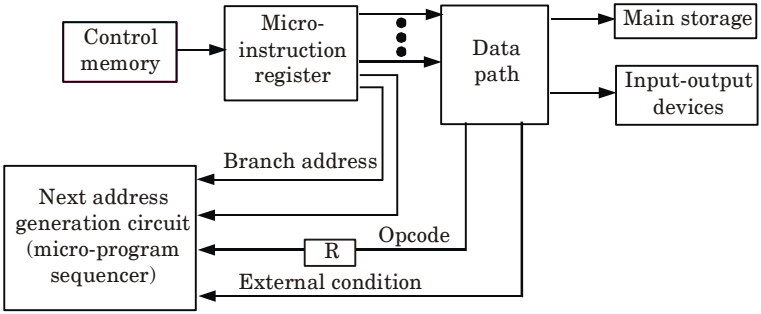


Fig. 7. Block diagram of micro-programmed control with micro-program sequencer.

b. Draw a flowchart for adding and subtracting two fixed point binary numbers where negative numbers are signed 1's complement presentation.

Ans.

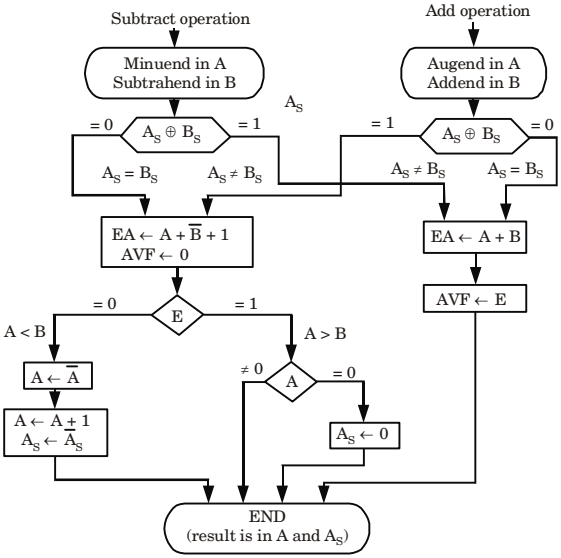


Fig. 8.

6. Attempt any **one** part of the following : **(7 × 1 = 7)**

a. Give the block diagram of DMA controller. Why are the read and write control lines in a DMA controller bidirectional ?

Ans. **Block diagram of DMA controller :**

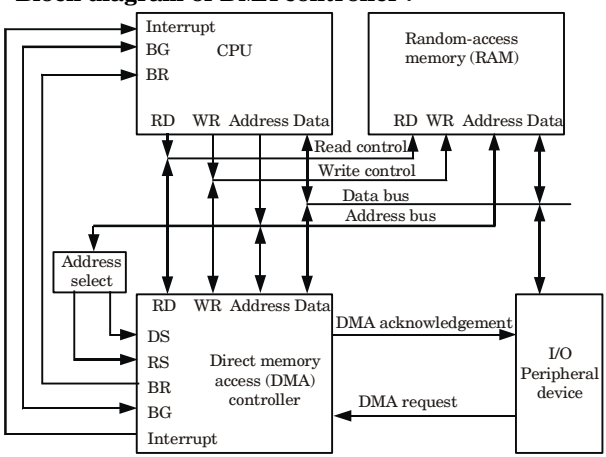


Fig. 9. DMA transfer in a computer system.

Reason for bidirectional read and write control lines : Read and write control lines in a DMA controller is bidirectional because the microprocessor fetch (read) the data from the memory and write data to the memory.

b. Explain all the phases of instruction cycle.

Ans. **Instruction cycle :**

1. Instruction cycle is a complete process of instruction execution.
2. It is a basic operational process of a computer.
3. It is the process by which a computer retrieves a program instruction from its memory, determines what actions the instruction dictates, and carries out those actions.

The instruction cycle is divided into three sub cycles :

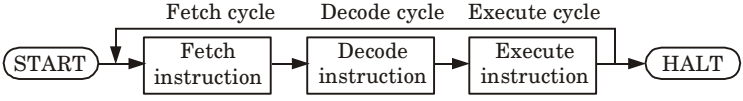


Fig. 10.

1. Fetch cycle : To fetch an opcode from a memory location following steps are performed :

i. The program counter places the address of the memory location in which the opcode is stored, on the address bus.
ii. The CPU sends the required memory control signals so as to enable the memory to send the opcode.
iii. The opcode stored in the memory location is placed on the data bus and transferred to the CPU.

2. Decode cycle :

i. The opcode which is fetched from the memory is placed first of all in the Data Register (DR) (data/address buffer in case of Intel 8085). Thereafter it goes to the Instruction Register (IR).
ii. From the instruction register it goes to the decoder circuitry, which is within the CPU.
iii. The decoder circuitry decodes the opcode.
iv. After the opcode is decoded the CPU comes to know what operation is to be performed, and then execution begins.

3. Execute cycle :

i. In this cycle, function of the instruction is performed.
ii. If the instruction involves arithmetic or logic, ALU is utilized.

7. Attempt any **one** part of the following : **(7 × 1 = 7)**

a. Explain the basic concept of hardwired and software control unit with neat diagrams.

Ans. **Hardwired control :**

1. It is a controller as a sequential logic circuit or a finite state machine that generates a sequence of control signals in response to the externally supplied instructions.

2. The control logic is implemented with gates, flip-flops, decoders, and other digital circuits.

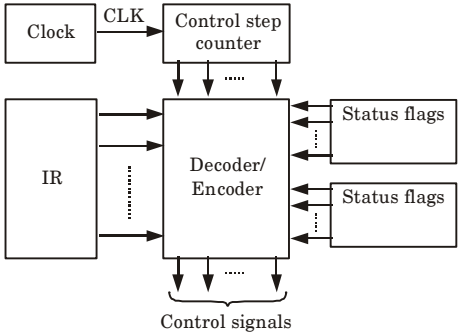


Fig. 11. General block diagram of hardwired control.

3. A hardwired control requires changes in the wiring among the various components if the design has to be modified or changed.
4. Its input logic signals are transformed into a set of output logic signals are called control signals.
5. Each step in this sequence is completed in one clock cycle.
6. A counter may be used to keep the track of the control steps.
7. The required control signals are determined by the following information :

a. Contents of the control step counter.
b. Contents of the instruction register.
c. Contents of the condition code flags.
d. External input signals such as MFC and interrupt request.

Micro-programmed control :

1. Micro-programmed control is a method of control unit design in which the control signal selection and sequencing information is stored in a ROM or RAM called a Control Memory (CM).
2. The sequence of control signals to be generated by the controller can be stored in a special Read Only Memory (ROM) also called Control Memory (CM).
3. Memory control word is written for each micro-operation, and these control words are stored in a serial ascending memory location.
4. The control word is accessed serially (serial access memory) from the control memory.
5. Control words are stored in the ROM permanently.
6. The output of the control memory provides the required control signals.
7. If the control memory is sequentially accessed by incrementing control memory location, then the sequence of control signals stored in successive word of ROM can be generated.

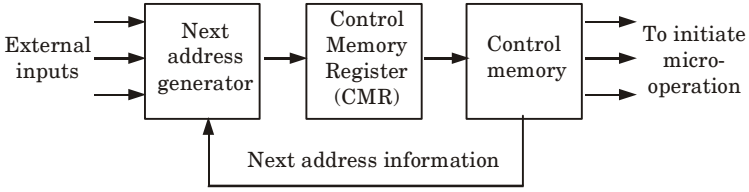


Fig. 12. Block diagram of micro-programmed control unit.

8. The storage of control word in a ROM is often referred to as firmware.

b.

1	2	3	4	5	6	
S1	X					X
S2		X			X	
S3			X			
S4				X		
S5		X				X

For the following reservation table :

i. Calculate the set of the forbidden latencies and collision vector.

ii. Draw a state diagram, showing all possible initial sequences (cycles) without a collision in the pipeline.

iii. Simple Cycles (SC)

iv. Greedy cycles among simple cycles

v. MAL (Minimum Average Latency)

vi. What is the minimum allowed constant cycles ?

vii. Maximum throughput

viii. Throughput if the minimum constant cycle is used.

Ans.

i. The forbidden latencies are 3, 4, and 5 (S1 : 5 ; S2 : 3 ; S3 : 0 ; S4 : 0 and S5 : 4), so that the collision vector is C_x = 11100 where the permissible latencies are 1 and 2.

ii. State diagram can be obtained by tracing each C_x shift as followed :

1st shift

shifted bit	01110
C_x	11100
new value	11110

2nd shift

shifted bit	00111
C_x	11100
new value	11111

1-shift from 2nd shift

shifted bit	01111
C_x	11100
new value	11111

2-shift from 2nd shift

shifted bit	00111
C_x	11100
new value	11111

2-shift from 1st shift

shifted bit	00111
C_x	11100
new value	11111

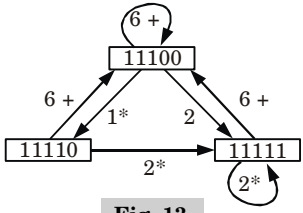


Fig. 13.

iii. The simple cycles are (2), (6), (1, 6) and (2, 6).

iv. The greedy cycles are (2) and (1, 6).

v. According to the lowest greedy cycle's average latency, the MAL (Minimum Average Latency) is 2.

vi. Minimum allowed constant cycle is 2.

vii. The maximum throughput is $\frac{1}{\text{MAL}} = \frac{1}{2} = 0.5$ *i.e.*, 50 %.

viii. The minimum constant cycle is 2, so that the maximum throughput does not change, only 50%.

☺☺☺

B.Tech.

(SEM. III) ODD SEMESTER THEORY EXAMINATION, 2019-20 COMPUTER ORGANIZATION AND ARCHITECTURE

Time : 3 Hours **Max. Marks : 100**

Note : 1. Attempt **all** Sections. If require any missing data; then choose suitably.

SECTION – A

1. Attempt **all** questions in brief. **(2 × 10 = 20)**

a. Define the term computer architecture.

b. Draw the basic functional units of a computer.

c. Perform the 2's complement subtraction of smaller number (101011) from larger number (111001).

d. What is the role of multiplexer and decoder ?

e. Write the differences between RISC and CISC.

f. What are the types of microinstructions available ?

g. What is SRAM and DRAM ?

h. What is the difference between 2D and $2^{1/2}$ D memory organization ?

i. What is I/O control method ?

j. What is bus arbitration ?

SECTION-B

2. Attempt any **three** of the following : **(10 × 3 = 30)**

a. Convert the following arithmetic expressions from infix to reverse polish notation :

i. A*B + C*D + E*F

ii. A*[B + C*CD + E]/F*(G + H)

b. Design a 4-bit carry look ahead adder and explain its operation with an example.

c.

i. Draw the timing diagram for a instruction cycle and explain.

ii. Give a note on subroutine.

d. What do you mean by virtual memory ? Discuss how paging helps in implementing virtual memory.

e. What is DMA ? Describe how DMA is used to transfer data from peripherals.

SECTION-C

3. Attempt any **one** part of the following : **(10 × 1 = 10)**

a. Describe in detail the different kinds of addressing modes with an example.

b. Discuss stack organization. Explain the following in details :

i. Register stack

ii. Memory stack

4. Attempt any **one** part of the following : **(10 × 1 = 10)**

a. Represent the following decimal number in IEEE standard floating-point format in a single precision method (32-bit) representation method :

i. $(65.175)_{10}$

ii. $(-307.1875)_{10}$

b. Using Booth's algorithm perform the multiplication on the following 8-bit unsigned integer 10110011 * 11010101.

5. Attempt any **one** part of the following : **(10 × 1 = 10)**

a. What is parallelism and pipelining in computer architecture ?

b. Explain the organization of microprogrammed control unit in detail.

6. Attempt any **one** part of the following : **(10 × 1 = 10)**

a. Discuss the different mapping techniques used in cache memories and their relative merits and demerits.

b. RAM chip 4096 × 8 bits has two enable lines. How many pins are needed for the integrated circuits package ? Draw a block diagram and label all input and outputs of the RAM. What is main feature of random access memory ?

7. Attempt any **one** part of the following : **(10 × 1 = 10)**

a. Write down the difference between isolated I/O and memory mapped I/O. Also discuss advantages and disadvantages of isolated I/O and memory mapped I/O.

b.

i. Discuss the design of a typical input or output interface.

ii. What are interrupts ? How are they handled ?

SOLUTION OF PAPER (2019-20)

SECTION – A

1. Attempt **all** questions in brief. **(2 × 10 = 20)**

a. Define the term computer architecture.

Ans. Computer architecture refers to those attributes of a system that are visible to a programmer or, those attributes that have a direct impact on the logical execution of a program.

b. Draw the basic functional units of a computer.

Ans. The main functional units of a digital computer are shown in Fig. 1.

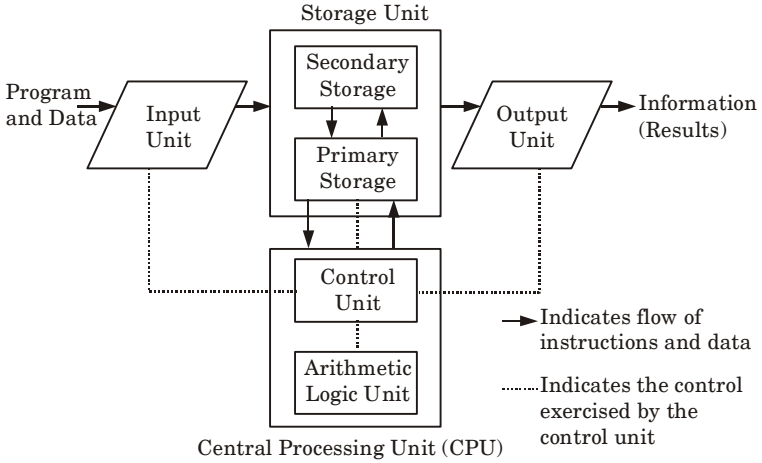


Fig. 1. Functional unit of digital computer.

c. Perform the 2's complement subtraction of smaller number (101011) from larger number (111001).

Ans. (111001) – (101011)

Step 1 : 1's complement of 101011 = 0 1 0 1 0 0

Step 2 : 2's complement of 101011 = 0 1 0 1 0 0

$$\begin{array}{r} +\ 1 \\ \hline 0\,1\,0\,1\,0\,1 \end{array}$$

Step 3 : Adding minuend and 2's complement obtained in Step 2.

$$\begin{array}{rr} & 1\,1\,1\,0\,0\,1 \\ + & 0\,1\,0\,1\,0\,1 \\ \hline \text{Carry} \rightarrow & \boxed{1}\,0\,0\,1\,1\,1\,0 \end{array}$$

The sum produce a carry because minuend > subtrahend. So, discard carry and write the remaining bits.

(1 1 1 0 0 1) – (101011) = 0 0 1 1 1 0

d. What is the role of multiplexer and decoder ?

Ans. Role of multiplexer is to combine multiple inputs into a single data stream.
Role of decoder is to translate a code into a sets of control signals.

e. Write the differences between RISC and CISC.

Ans.

S. No.	RISC	CISC
1.	Highly pipelined.	Less pipelined.
2.	Fixed length instructions.	Variable length instructions.
3.	Few addressing modes.	Many addressing modes.

f. What are the types of microinstructions available ?

Ans. Following types of microinstructions are available :
i. Vertical / horizontal ii. Packed / unpacked
iii. Hard / soft microprogramming iv. Direct / indirect encoding

g. What is SRAM and DRAM ?

Ans. **SRAM :** Static Random Access Memory is a semiconductor memory in which, the data does not need to be refreshed dynamically. SRAM is volatile in nature. It consumes more power.
DRAM : Dynamic RAM is a form of random access memory. DRAM uses a capacitor to store each bit of data, and the level of charge on each capacitor determines whether that bit is a logical 1 or 0.

h. What is the difference between 2D and $2^{1/2}$ D memory organization ?

Ans.

S. No.	2D memory organizations	$2^{1/2}$ D memory organizations
1.	In 2D organization, hardware is fixed.	In $2^{1/2}$ D organization, hardware changes.
2.	2D organization requires more number of gates.	$2^{1/2}$ D organization requires less number of gates.
3.	Error correction is not possible in the 2D organization.	In $2^{1/2}$ D organization, error correction is easy.

i. What is I/O control method ?

Ans. Usually microprocessor controls the process of data transfer between the microprocessor and the peripherals. However, sometimes peripherals also controls the data transfer. During such time I/O control methods are used. The I/O control methods

are employed when the peripheral is capable of transferring the data at higher or slower speed than the processor.

j. What is bus arbitration ?

Ans.

1. Bus arbitration is a mechanism which decides the selection of current master to access bus.
2. Among several masters and slave units that are connected to a shared bus, it may happen that more than one master or slave units will request access to the bus at the same time.

SECTION-B

2. Attempt any **three** of the following : **(10 × 3 = 30)**

a. Convert the following arithmetic expressions from infix to reverse polish notation :

i. A*B + C*D + E*F

ii. A*[B + C*CD + E]/F*(G + H)

Ans.

i. A*B + C*D + E*F

$\underbrace{AB*}_{X}$ + C*D + E*F

X + C*D + E*F X = AB*

X + $\underbrace{CD*}_{Y}$ + E*F

X + Y + E*F Y = CD*

X + Y + $\underbrace{EF*}_{Z}$

$\underbrace{X + Y}_{Z}$ + Z Z = EF*

$\underbrace{XY + +}_{P}$ Z

P + Z P = XY +

$\underbrace{PZ +}_{Q}$ Q = PZ +

On putting value
= PZ +
= XY + Z +
= XY + EF*+
= XCD* + EF * +
= AB * CD * + EF * +

ii. A * [B + C * CD + E] / F * (G + H)
Assume D + E in the bracket then expression becomes
A * [B + C * (D + E)] / F * (G + H)

$A * [B + C * \underbrace{DE +}_{T_1}] / F * (G + H)$

$A * [B + C * T_1] / F * (G + H)$ $T_1 = DE +$

$A * [B + \underbrace{C T_1 *}_{T_2}] / F * (G + H)$

$A * [B + T_2] / F * (G + H)$ $T_2 = CT_1 * +$

$A * \underbrace{B T_2}_{T_3} / F * (G + H)$

$A * T_3 / F * (G + H)$ $T_3 = BT_2 +$

$A * T_3 / F * \underbrace{GH +}_{T_4}$

$A * T_3 / F * T_4$ $T_4 = GH +$

$A * \underbrace{T_3 F /}_{T_5} * T_4$

$A * T_5 * T_4$ $T_5 = T_3 F/$

$\underbrace{A T_5 *}_{T_6} T_4$

$T_6 * T_4$ $T_6 = AT_5 *$

$\underbrace{T_6 T_4 *}_{T_7}$ $T_7 = T_6 T_4 *$

T_7

on putting value of T_7

$= T_6 T_4 *$

$= AT_5 * GH + *$

$= A T_3 F/ * GH + *$

$= A BT_2 + F / * GH + *$

$= A BCT_1 * + F / * GH + *$

$= A BCDE + * + F / GH + *$

$= A BCDE + * F / GH + *$

b. Design a 4-bit carry look ahead adder and explain its operation with an example.

Ans.

1. A Carry Look Ahead adder (CLA) or fast adder is a type of adder used in digital logic.
2. A carry look ahead adder improves speed by reducing the amount of time required to determine carry bits.
3. The carry look ahead adder calculates one or more carry bits before the sum, which reduces the waiting time to calculate the result of the larger-value bits of the adder.
4. Carry look ahead depends on two things :

a. Calculating for each digit position whether that position is going to propagate a carry if one comes in from the right.
b. Combining these calculated values to be able to deduce quickly whether, for each group of digits, that group is going to propagate a carry that comes in from the right.

5. Carry look ahead logic uses the concepts of generating and propagating carries.
6. The addition of two binary numbers in parallel implies that all the bits of the augend and addend are available for computation at the same time.
7. The carry propagation time is an important attribute of the adder because it limits the speed with which two numbers are added.
8. A solution is to increase the complexity of the equipment in such a way that the carry delay time is reduced.

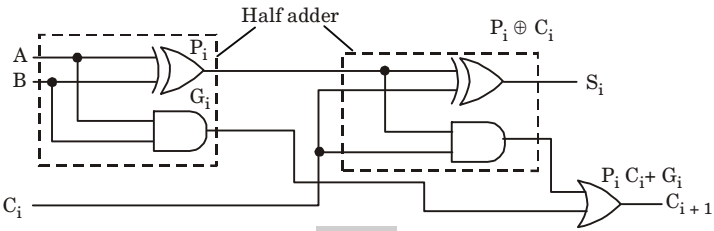


Fig. 2.

9. Consider the circuit of the full adder shown in Fig. 3. If we define two new binary variables,

$$P_i = A_i \oplus B_i \, , \, G_i = A_i B_i$$

10. The output sum and carry can respectively be expressed as

$$S_i = P_i \oplus C_i \, , \, C_{i+1} = G_i + P_i C_i$$

11. G_i is called a carry generate, and it produces a carry of 1 when both A_i and B_i are 1, regardless of the input carry C_i. P_i is called a carry propagate, because it determines whether a carry into stage i will propagate into stage $i + 1$.

$$C_0 = \text{input carry}$$

$i = 0$ $\quad C_1 = G_0 + P_0 C_0$

$i = 1$ $\quad C_2 = G_1 + P_1 C_1 = G_1 + P_1 (G_0 + P_0 C_0)$

$$= G_1 + P_1 G_0 + P_0 P_1 C_0$$

$i = 2$ $\quad C_3 = G_2 + P_2 C_2 = G_2 + P_2 G_1 + P_2 P_1 G_0 + P_2 P_1 P_0 C_0$

$i = 3$ $\quad C_4 = G_3 + P_3 C_3$

$$= G_3 + P_3 (G_2 + P_2 G_1 + P_2 P_1 G_0 + P_3 P_2 P_1 P_0 C_0)$$

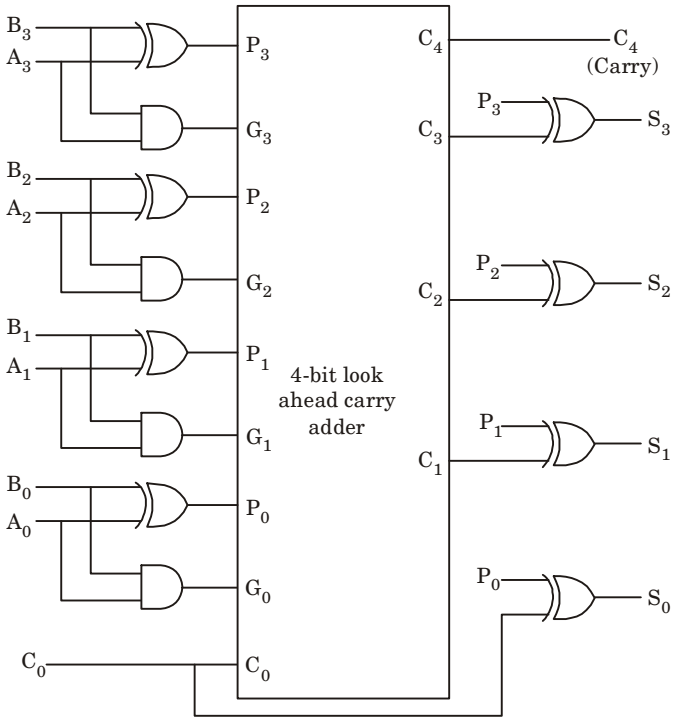


Fig. 3. A 4-bit look-ahead carry adder.

c.

i. Draw the timing diagram for a instruction cycle and explain.

ii. Give a note on subroutine.

Ans.

i. Let consider the instruction cycle for instruction STAX rp.
This instruction stores the contents of A register in memory whose address is specified by register pair (BC or DE).
It requires the following machine cycles :

1. **Opcode fetch :** Program counter places the memory address on low order and high order address bus. This machine cycle is required for reading the opcode of STAX rp (e.g. 0211 for STAX B) into the microprocessor and decode it.
2. **Memory write :** Higher order address is obtained from higher order register and lower address is obtained from lower order register of the specified register pair. The contents of the accumulator are stored into the addressed memory location. Thus memory write machine cycle is required for writing the data from the microprocessor (A register) to the addressed memory location.

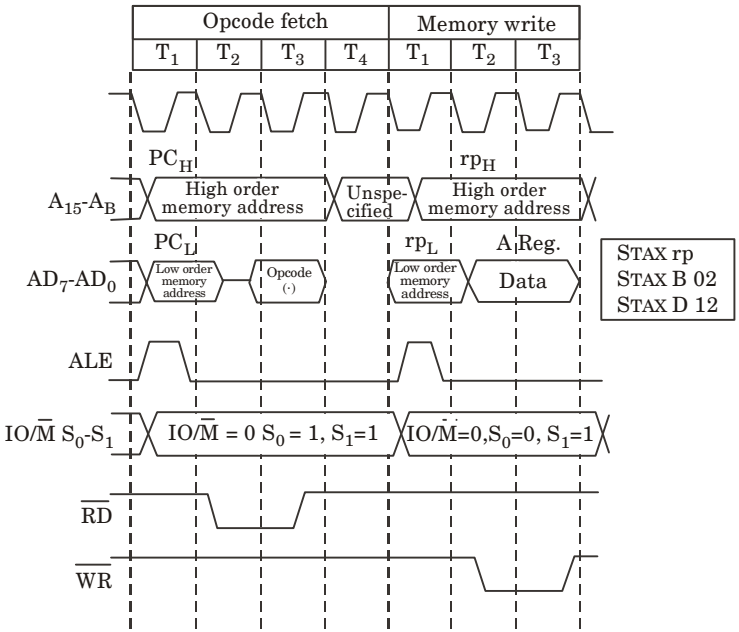


Fig. 4.

ii. Subroutine :

1. A subroutine is a set of common instructions that can be used in a program many times.
2. A subroutine consists of a self-contained sequence of instructions that carries out a given task.
3. Each time that a subroutine is used in the main part of the program, a branch is executed to the beginning of the subroutine.
4. After the subroutine has been executed, a branch is made back to the main program.
5. A branch can be made to the subroutine from any part of the main program.
6. Because branching to a subroutine and returning to the main program is such a common operation, all computers provide special instructions to facilitate subroutine entry and return.

d. What do you mean by virtual memory ? Discuss how paging helps in implementing virtual memory.

Ans. Virtual memory :

1. Virtual memory is a memory management capability of an OS that uses hardware and software to allow a computer to compensate for

physical memory shortages by temporarily transferring data from Random Access Memory (RAM) to disk storage.

2. Virtual address space is increased using active memory in RAM and inactive memory in Hard Disk Drives (HDDs) to form contiguous addresses that hold both the application and its data.

Paging helps in implementing virtual memory as :

1. Virtual memory space is divided into equal size pages.
2. Main memory space is divided into equal size page frames each frame can hold any page from virtual memory.
3. When CPU wants to access page, it first looks into main memory. If it is found in main memory then it is called Hit and page is transfer from main memory to CPU.
4. If CPU needs page that is not present in main memory then it is called as page fault. The page has to be loaded from virtual memory to main memory.
5. There are different page replacement schemes such as FIFO, LRU, LFU etc.
6. During page replacement, if the old page has been modified in the main memory, then it needs to be first copied into the virtual memory and then replaced. CPU keeps track of such updated pages by maintaining dirty bit for each page. When page is updated in main memory dirty bit is set then this dirty page is first copied into virtual memory and then replaced.
7. Pages are loaded into main memory only when required by the CPU, then it is called demand paging. Thus pages are loaded only after page faults.

e. What is DMA ? Describe how DMA is used to transfer data from peripherals.

Ans. DMA :

1. DMA stands for "Direct Memory Access" and is a method of transferring data from the computer's RAM to another part of the computer without processing it using the CPU.
2. While most data that is input or output from our computer is processed by the CPU, some data does not require processing, or can be processed by another device.
3. DMA can save processing time and is a more efficient way to move data from the computer's memory to other devices.
4. In order for devices to use direct memory access, they must be assigned to a DMA channel. Each type of port on a computer has a set of DMA channels that can be assigned to each connected device.
5. For example, a PCI controller and a hard drive controller each have their own set of DMA channels.

Block diagram for DMA :

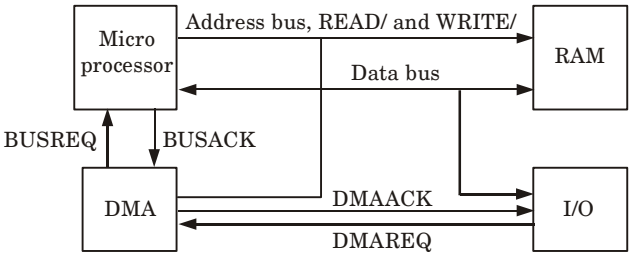


Fig. 5.

Working of DMA controller :

1. When the peripheral device sends a DMA request, the DMA controller activates the BR line, informing the CPU to relinquish the buses.
2. The CPU responds with its BG line, informing the DMA that its buses are disabled.

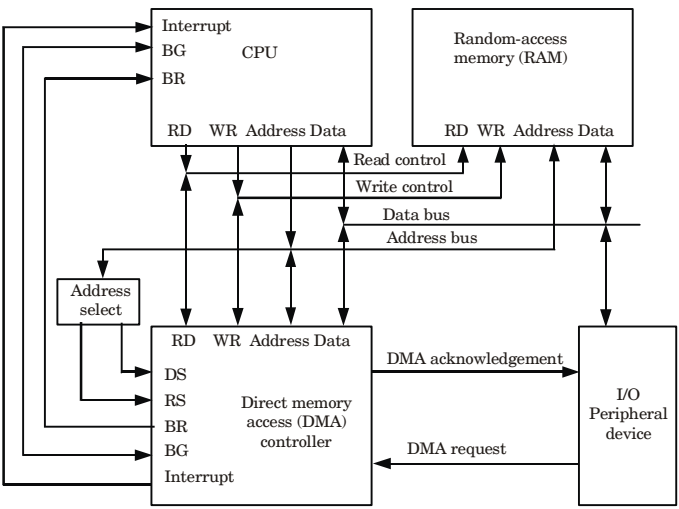


Fig. 6. DMA transfer in a computer system.

3. The DMA then puts the current value of its address register into the address bus, initiates the RD or WR signal, and sends a DMA acknowledge to the peripheral device.
4. The direction of transfer depends on the status of the BG line :
   a. When BG = 0, the RD and WR are input lines allowing the CPU to communicate with the internal DMA registers.

b. When BG = 1, the RD and WR are output lines from the DMA controller to the random access memory to specify the read or write operation for the data.

5. When the peripheral device receives a DMA acknowledge, it puts a word in the data bus (for write) or receives a word from the data bus (for read).
6. Thus, the DMA controls the read or write operations and supplies the address for the memory.
7. The peripheral unit can communicate with memory through the data bus for direct transfer between the two units while the CPU is momentarily disabled.

SECTION-C

3. Attempt any **one** part of the following : **(10 × 1 = 10)**

a. Describe in detail the different kinds of addressing modes with an example.

Ans.

i. Direct :

1. A very simple form of addressing is direct addressing, in which the address field contain the effective address of the operand : $EA = A$

   where, EA = Actual (effective) address of the location containing the referenced operand.

   A = Contents of the address field in the instruction.

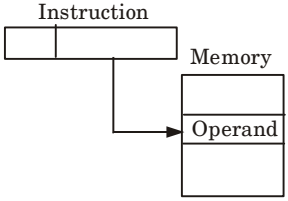


Fig. 7. Direct.

2. A direct address in instruction needs two reference to memory :
   a. Read instruction
   b. Read operand

ii. Displacement addressing :

1. A very powerful mode of addressing combines the capabilities of direct addressing and register indirect addressing.
2. It is known by a variety of names depending upon the content of its use but the basic mechanism is the same.
3. Displacement addressing requires that the instruction have two address fields, at least one of which is explicit.
4. The value contained in one address field (value = A) is used directly.

5. The other address field or an implicit reference based on opcode, refers to a register whose contents are added to *A* to produce the effective address.

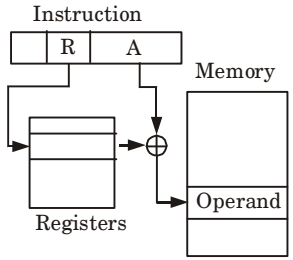


Fig. 8. Displacement addressing.

iii. Relative addressing :

1. Relative addressing means that the next instruction to be carried out is an offset number of locations away, relative to the address of the current instruction.
2. Consider this bit of pseudo-code
   Jump + 3 if accumulator = = 2
   Code executed if accumulator is NOT = 2
   Jump + 5 (unconditional relative jump to avoid the next line of code)
   acc : (code executed if accumulator is = 2)
3. In the code, the first line of code is checking to see if the accumulator has the value of 2 then the next instruction is 3 lines away.
4. This is called a conditional jump and it is making use of relative addressing.

iv. Register indirect mode :

1. Register indirect mode is similar to indirect addressing.
2. The only difference is whether the address field refers to a memory location or a register.
3. Thus, for register indirect address, $EA = (R)$

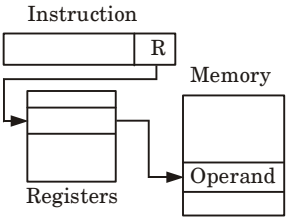


Fig. 9. Register indirect.

v. Implied mode :

1. In this mode, the operands are specified implicitly in the definition of the instruction.

2. All register reference instructions that use an accumulator are implied mode instructions.
3. Zero address instructions in a stack-organized computer are implied mode instruction since the operands are implied to be on top of the stack. It is also known as stack addressing mode.

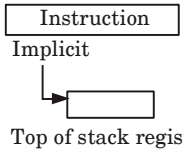


Fig. 10. Implied mode.

vi. Immediate mode :

1. In this mode, the operand is specified in the instruction itself.
2. The operand field contains the actual operand to be used in conjunction with the operation specified in the instruction.

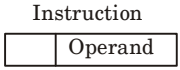


Fig. 11. Immediate mode.

vii. Indexed :

1. The effective address of the operand is generated by adding a constant value to the contents of a register.
2. The register used may be either a special register for this purpose or more commonly, it may be any one of a set of general purpose registers in the CPU.
3. It is referred to as an index register. We indicate the index mode symbolically as, $X(R)$
   where X denotes a constant and R is the name of register involved.
4. The effective address of the operand is given by, $EA = X + [R]$
5. In the process of generating the effective address, the contents of the index register are not changed.

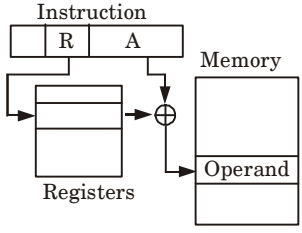


Fig. 12. Indexed.

Example :

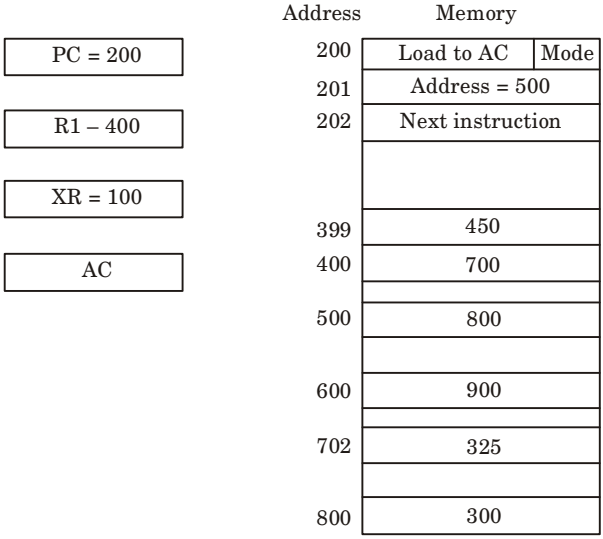


Fig. 13.

Addressing Mode	Effective Address	Content of AC
Direct address	500	800
Immediate operand	201	500
Indirect address	800	300
Indexed address	600	900
Implied	–	400
Register indirect	400	700

b. Discuss stack organization. Explain the following in details :

i. Register stack

ii. Memory stack

Ans.

1. A stack is an ordered set of elements in which only one element can be accessed at a time.
2. The point of access is called the top of the stack.
3. The number of elements in the stack or length of the stack is variable.

4. Items may only be added or deleted from the top of the stack.
5. A stack is also known as a pushdown list or a Last-In-First-Out (LIFO) list.

i. Organization of register stack :

Consider the organization of a 64-word register stack as illustrated in Fig. 14.

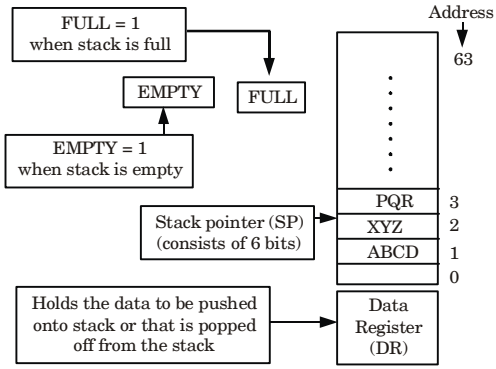


Fig. 14. Block diagram of 64-word stack.

The four separate registers used in the organization are :

1. **Stack Pointer register (SP) :** It contains a value in binary each of 6 bits, which is the address of the top of the stack. Here, the stack pointer SP contains 6 bits and SP cannot contain a value greater than 111111 *i.e.*, value 63.
2. **FULL register :** It can store 1 bit information. It is set to 1 when the stack is full.
3. **EMPTY register :** It can store 1 bit information. It is set to 1 when stack is empty.
4. **Data Register (DR) :** It holds the data to be written into or to be read from the stack.

ii. Memory stack : Memory stack is a series of memory spaces that is used in the processes that is done by processor and is temporarily stored in registers.

Role in managing subroutines :

1. The stack supports program execution by maintaining automatic process-state data.
2. If the main routine of a program, for example, invokes function *a* ( ), which in turn invokes function *b* ( ), function *b* ( ) will eventually return control to function *a* ( ), which in turn will return control to the main ( ) function as shown in Fig. 15.
3. To return control to the proper location, the sequence of return addresses must be stored.

4. A stack is well suited for maintaining this information because it is a dynamic data structure that can support any level of nesting within memory constraints.

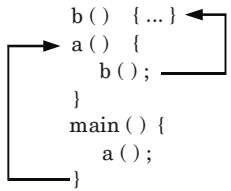


Fig. 15. Stack management.

5. When a subroutine is called, the address of the next instruction to execute in the calling routine is pushed onto the stack.
6. When the subroutine returns, this return address is popped from the stack, and program execution jumps to the specified location as shown in Fig. 16.

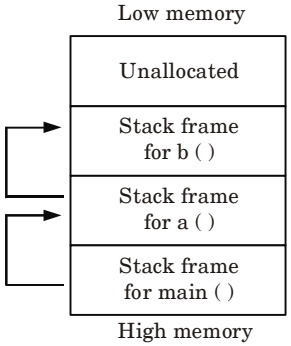


Fig. 16. Calling a subroutine.

7. The information maintained in the stack reflects the execution state of the process at any given instant.
8. In addition to the return address, the stack is used to store the arguments to the subroutine as well as local (or automatic) variables.
9. Information pushed onto the stack as a result of a function call is called a frame. The address of the current frame is stored in the frame or base pointer register.
10. When a subroutine is called, the frame pointer for the calling routine is also pushed onto the stack so that it can be restored when the subroutine exits.

4. Attempt any **one** part of the following : **(10 × 1 = 10)**

a. Represent the following decimal number in IEEE standard floating-point format in a single precision method (32-bit) representation method :

i. $(65.175)_{10}$
ii. $(-307.1875)_{10}$

Ans.

i. $(65.175)_{10}$

Step 1 : Convert the decimal number in binary format

Integer format :

2	65	
2	32	1
2	16	0
2	8	0
2	4	0
2	2	0
	1	0

$(65)_{10} = (1\ 0\ 0\ 0\ 0\ 0\ 1)_2$

Fractional format :

$0.175 \times 2 = 0.35 \Rightarrow 0$
$0.35 \times 2 = 0.7 \Rightarrow 0$
$0.7 \times 2 = 1.4 \Rightarrow 1$
$0.4 \times 2 = 0.8 \Rightarrow 0$
$0.8 \times 2 = 1.6 \Rightarrow 1$
$0.6 \times 2 = 1.2 \Rightarrow 1$
$0.2 \times 2 = 0.4 \Rightarrow 0$

$(0.175)_{10} = (0.0010110)_2$

$(65.175)_{10} = (1000001.0010110)_2$

Step 2 : Normalize the number.

$(1000001.0010110) = 1.000001001 \times 2^6$

Step 3 : Representing the floating point in single precision :

For a given member,

$S = 0$
$E = 6$
$M = 0000010010110$

Bias of single precision format = 127

$E' = 127 + E$
$= 127 + 6$
$= (133)_{10}$
$= (1011101)_2$

Number in single precision format

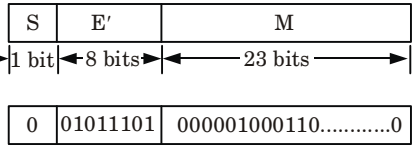


Fig. 17.

ii. $(-307.1875)_{10}$

Step 1 : Convert the decimal number in binary format

Integer format :

2	307	
2	153	1
2	76	1
2	38	0
2	19	0
2	9	1
2	4	1
2	2	0
2	1	0

$(307)_{10} = (100110011)_2$

Fractional format :

$0.1875 \times 2 = 0.3750 \Rightarrow 0$

$0.3750 \times 2 = 0.750 \Rightarrow 0$

$0.750 \times 2 = 1.5 \Rightarrow 1$

$0.5 \times 2 = 1.0 \Rightarrow 1$

$(0.1875)_{10} = (0.0011)_2$

$(307.1875)_{10} = (100110011.0011)_2$

$-307.1875_{10} = -100110011.0011_2$

Step 2 : Normalize the number

$-100110011.001 = -1.00110011001 \times 2^8$

Step 3 : Representing the number in single precision.

For a given number

$S = 1$ (given number is negative number)

$E = 8$

$M = 00110011001$

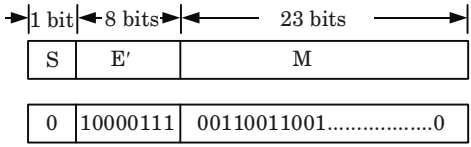
Bias of single precision format is = 127

$E' = 127 + E = 127 + 8$

$= (135)_{10}$

$= (1000\ 0111)_2$

Number in single precision format



b. Using Booth's algorithm perform the multiplication on the following 8-bit unsigned integer 10110011 * 11010101.

Ans. Multiplicand (M) = 10110011, Multiplier = 11010101

A	Q	Q_{n+1}	Operation	SC
00000000	11010101	0		1000
01001101 00100110	11010101 11101010	0 1	$A \rightarrow A - M$ Ashr A, Q, Q_{n+1}	0111
11011001 11101100	11101010 11110101	1 0	$A \rightarrow A + M$ Ashr A, Q, Q_{n+1}	0110
00111001 00011100	11110101 11111010	0 1	$A \rightarrow A - M$ Ashr A, Q, Q_{n+1}	0101
11001111 11100111	11111010 11111101	1 0	$A \rightarrow A + M$ Ashr A, Q, Q_{n+1}	0100
00110100 00011010	11111101 01111110	0 1	$A \rightarrow A - M$ Ashr A, Q, Q_{n+1}	0011
11001101 11100110	01111110 10111111	1 0	$A \rightarrow A + M$ Ashr A, Q, Q_{n+1}	0010
00110011 00011001	10111111 11011111	0 1	$A \rightarrow A - M$ Ashr A, Q, Q_{n+1}	0001
00001100	11101111	1		0000
Result = 00001100 11101111 (+ 33 11)				

5. Attempt any **one** part of the following : **(10 × 1 = 10)**

a. What is parallelism and pipelining in computer architecture ?

Ans. **Parallelism in computer architecture :**

1. Executing two or more operations at the same time is known as parallelism.
2. The purpose of parallel processing (parallelism) is to speed up the computer processing capability *i.e.*, it increases the computational speed.
3. It also increases throughput, *i.e.*, amount of processing that can be accomplished during a given interval of time.
4. Improves the performance of the computer for a given clock speed.
5. Two or more ALUs in CPU can work concurrently to increase throughput. The system may have two or more processors operating concurrently.

Pipelining :

1. Pipelining is a technique of decomposing a sequential process into sub-operations, with each sub-process being executed in a special dedicated segment that operates concurrently with all other segments.

2. The processor executes a program by fetching and executing instructions, one after the other.
3. Let F_i and E_i refer to the fetch and execute steps for instruction I_i.
4. Execution of a program consists of a sequence of fetch and execute steps as shows is Fig. 18.

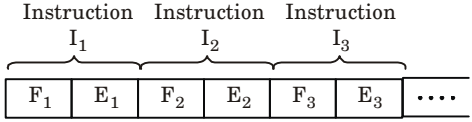


Fig. 18. Sequential execution.

5. Now consider a computer that has two separate hardware units, one for fetching instructions and another for executing them, as shown in Fig. 19.
6. The instruction fetched by the fetch unit is deposited in an intermediated storage buffer B_i.
7. The results of execution are deposited in the destination location specified by the instructions.
8. For these purposes, we assume that both the source and destination of the data operated in by the instructions are inside the block labelled "Execution unit".

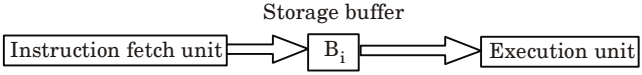


Fig. 19. Hardware organization.

9. The computer is controlled by a clock whose period is such that the fetch and execute steps of any instruction can be completed in one clock cycle.

b. Explain the organization of microprogrammed control unit in detail.

Ans. **Organization of micro-programmed control unit :**

1. The general configuration of a micro-programmed control unit is demonstrated in the block diagram of Fig. 20.
2. The control memory is assumed to be a ROM, within which all control information is permanently stored.

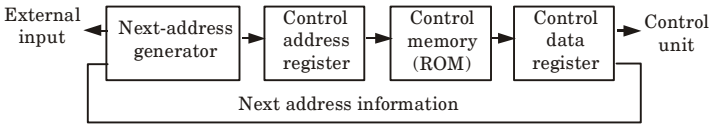


Fig. 20. Micro-programmed control organization.

3. The control memory address register specifies the address of the micro-instruction, and the control data register holds the micro-instruction read from memory.

4. The micro-instruction contains a control word that specifies one or more micro-operations for the data processor. Once these operations are executed, the control must determine the next address.
5. The location of the next micro-instruction may be the one next in sequence, or it may be located somewhere else in the control memory.
6. While the micro-operations are being executed, the next address is computed in the next address generator circuit and then transferred into the control address register to read the next micro-instruction.
7. Thus a micro-instruction contains bits for initiating micro-operations in the data processor part and bits that determine the address sequence for the control memory.
8. The next address generator is sometimes called a micro-program sequencer, as it determines the address sequence that is read from control memory.
9. Typical functions of a micro-program sequencer are incrementing the control address register by one, loading into the control address register an address from control memory, transferring an external address, or loading an initial address to start the control operations.
10. The control data register holds the present micro-instruction while the next address is computed and read from memory.

6. Attempt any **one** part of the following : **(10 × 1 = 10)**

a. Discuss the different mapping techniques used in cache memories and their relative merits and demerits.

Ans. **Cache mapping :**

1. Cache mapping is the method by which the contents of main memory are brought into the cache and referenced by the CPU. The mapping method used directly affects the performance of the entire computer system.
2. Mapping is a process to discuss possible methods for specifying where memory blocks are placed in the cache. Mapping function dictates how the cache is organized.

Types of mapping :

1. Direct mapping :

a. The direct mapping technique is simple and inexpensive to implement.

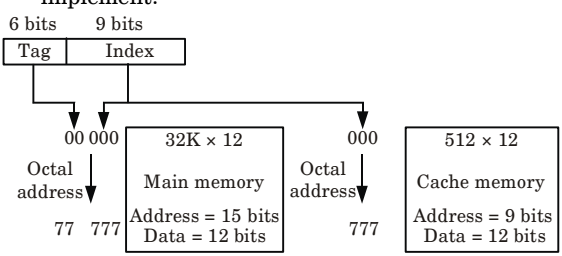


Fig. 21. Functional unit of digital computer.

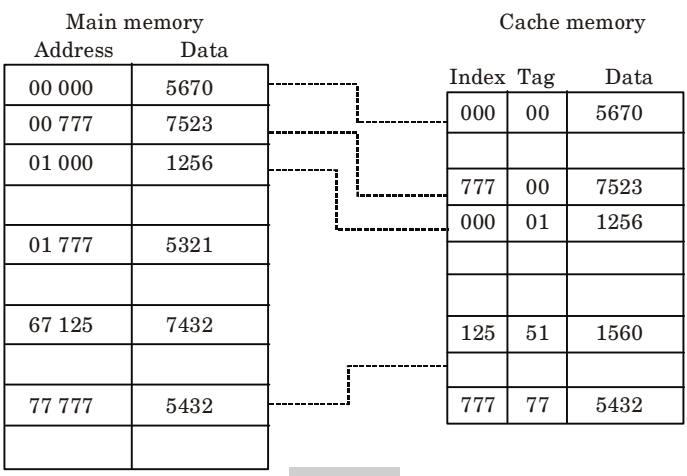


Fig. 22.

b. When the CPU wants to access data from memory, it places an address. The index field of CPU address is used to access address.
c. The tag field of CPU address is compared with the associated tag in the word read from the cache.
d. If the tag-bits of CPU address are matched with the tag-bits of cache, then there is a hit and the required data word is read from cache.
e. If there is no match, then there is a miss and the required data word is stored in main memory. It is then transferred from main memory to cache memory with the new tag.

Merits of direct mapping :

a. Direct mapping is simplest type of cache memory mapping.
b. Here only tag field is required to match while searching word that is why it fastest cache.
c. Direct mapping cache is less expensive compared to associative cache mapping.

Demerits of direct mapping :

a. The performance of direct mapping cache is not good as requires replacement for data-tag value.

2. Associative mapping :

a. An associative mapping uses an associative memory.
b. This memory is being accessed using its contents.
c. Each line of cache memory will accommodate the address (main memory) and the contents of that address from the main memory.
d. That is why this memory is also called Content Addressable Memory (CAM). It allows each block of main memory to be stored in the cache.

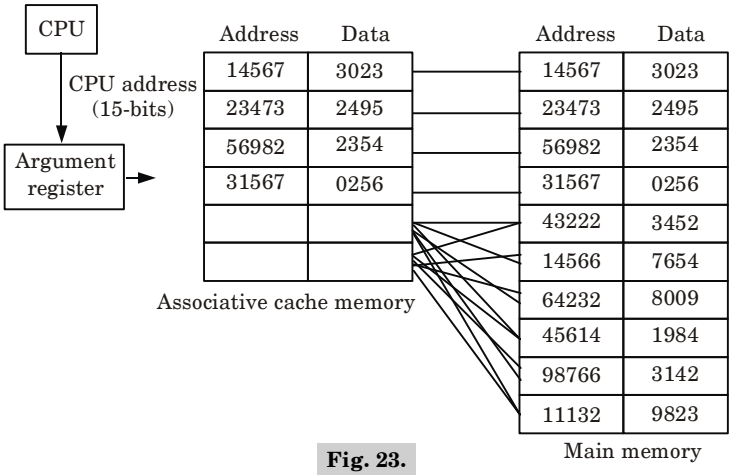


Fig. 23.

Merits of associative mapping :

a. Associative mapping is fast.

b. Associative mapping is easy to implement.

Demerits of associative mapping :

a. Cache Memory implementing associative mapping is expensive as it requires to store address along with the data.

3. Set associative mapping :

a. In set associative mapping, each cache location can have more than one pair of tag + data items.

b. It combines the best of the direct mapping cache and the more flexible mapping of the fully associative cache.

c. That is more than one pair of tag and data are residing at the same location of cache memory. If one cache location is holding two pair of tag + data items, that is called 2-way set associative mapping.

Merits of Set-Associative mapping :

a. Set-Associative cache memory has highest hit-ratio compared two previous two cache memory discussed above. Thus its performance is considerably better.

Demerits of Set-Associative mapping :

a. Set-Associative cache memory is very expensive. As the set size increases the cost increases.

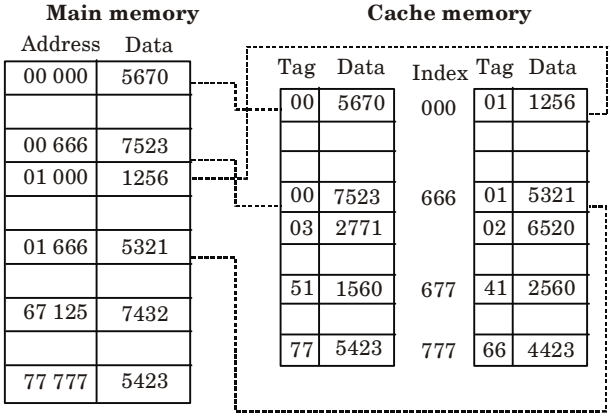


Fig. 24.

b. RAM chip 4096 × 8 bits has two enable lines. How many pins are needed for the integrated circuits package ? Draw a block diagram and label all input and outputs of the RAM. What is main feature of random access memory ?

Ans. Size of RAM chip = 4096 × 8
Number of input = 14 pins (2^{14} = 4096)
Number of output = 8 pin
Number of chip select = 2 pin
Number of power pin = 2 pin
Total pins required = 14 + 8 + 2 + 2 = 26 pins

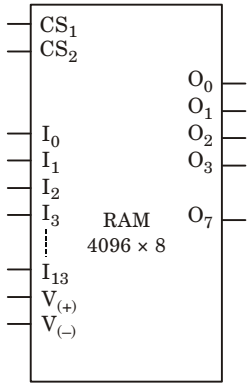


Fig. 25.

Features of random access memory :

1. RAM is volatile memory and require power to store data.
2. RAM is used to store the data for processing on CPU.
3. RAM chips often range in storage capacity from 1 GB to 256 GB.
4. RAM chip is used to increase the speed of the computer.

7. Attempt any **one** part of the following : **(10 × 1 = 10)**

a. Write down the difference between isolated I/O and memory mapped I/O. Also discuss advantages and disadvantages of isolated I/O and memory mapped I/O.

Ans. Difference between isolated I/O and memory mapped I/O :

S. No.	Isolated I/O	Memory mapped I/O
1.	Isolated I/O uses separate memory space.	Memory mapped I/O uses memory from the main memory.
2.	Limited instructions can be used. Those are IN, OUT, INS, OUTS.	Any instruction which references to memory can be used.
3.	The addresses for isolated I/O devices are called ports.	Memory mapped I/O devices are treated as memory locations on the memory map.
4.	Efficient I/O operations due to separate bus.	Inefficient I/O operations due to single bus for data and addressing.
5.	Comparatively larger in size.	Smaller in size.
6.	Uses complex internal logic.	Common internal logic for memory and I/O devices.
7.	Slower operations.	Faster operations.

Advantage of isolated I/O :

1. The devices of I/O are treated in a separate domain as compared to memory.
2. A total of 1MB address space is allowed for memory applications.
3. In order to maximize the I/O operations (isolated) separate instructions are always provided to perform these operations.

Disadvantage of isolated I/O :

1. The data transfer only occurs between the I/O port and the registers.

Advantages of memory mapped I/O :

1. I/O intensive operation is fast.
2. The SQLite library needs less RAM.

Disadvantages of memory mapped I/O :

1. If an I/O error on a memory-mapped file cannot be caught by the application, results in a program crash.
2. Performance is reduced by the use of memory-mapped I/O.

b.

i. Discuss the design of a typical input or output interface.

ii. What are interrupts ? How are they handled ?

Ans.

i. 1. The I/O interface includes control and timing requirements to coordinate the flow of traffic between internal resources (memory, system bus) and external devices.

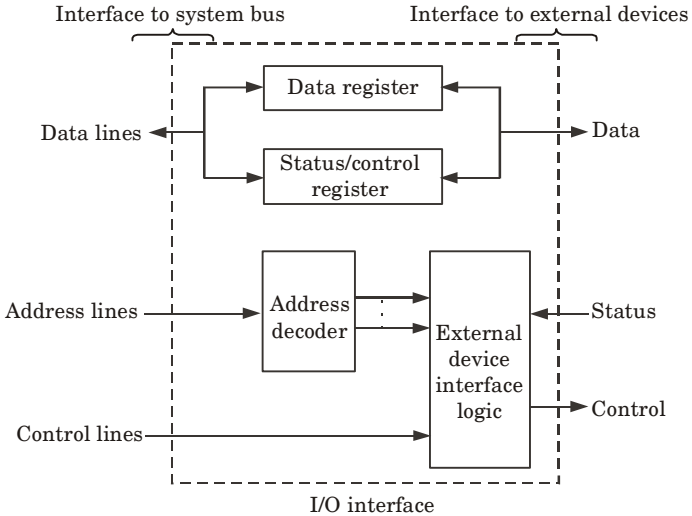


Fig. 26.

2. The I/O interface must able to perform device communication which involves commands, status information and data.
3. Data buffering is also an essential task of an I/O interface. Data transfer rates of peripheral devices are quite higher than that of processor and memory.
4. The data coming from memory or processor are sent to an I/O interface, buffered and then sent to the peripheral device at its data rate.
5. Data are buffered in I/O interface so as not to he up the memory in a slow transfer operation.
6. Thus the I/O interface must be able to operate at both peripheral and memory speeds.
7. I/O interface is also responsible for error detection and for reporting errors to the processor.
8. As shown in the Fig. 26, I/O interface consist: of data register, status/control register, address decoder and external device interface logic.
9. The data register holds the data being transferred to or from the processor.

10. The status/control register contains information relevant to the operation of the I/O device. Both data and status/control registers are connected to the data bus.
11. Address lines drive the address decoder. The address decoder enables the device to recognize its address when address appears on the address lines.
12. The external device interface logic accepts inputs from address decoder, processor control lines and status signal from the I/O device and generates control signals to control the direction and speed of data transfer between processor and I/O devices.

ii. Interrupts :

Interrupt : An interrupt is a signal sent by an I/O interface to the CPU when it is ready to send information to the memory or receive information from the memory.

Identifying the source of an interrupt :

Two methods are available to determine the interrupting device :

a. Polled interrupts :
b. Vectored interrupts :

Handling interrupt :

There are two methods of system to resolve the priority of interrupt :

a. Polling method :

1. When interrupt requests arrive from two or more devices simultaneously, the processor has to decide which request should be serviced first and which one should be delayed.
2. The processor takes the decision with the help of interrupt priorities. It accepts the request having the highest priority.
3. In this case polling is used to identify the interrupting device, priority is automatically assigned by the order in which devices are polled.
4. Therefore, no further arrangement is required to accommodate simultaneous interrupt requests. However, the priority of any device is usually determined by the way the device is connected to the processor.

b. Chaining method :

1. Most common way to connect the devices is to form a daisy chain, as shown in Fig. 27.

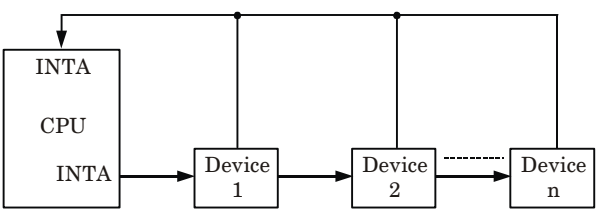


Fig. 27. Interrupt priority system using daisy chain.

2. The Interrupt Request line (INTR) is common to all the devices and the Interrupt Acknowledge line (INTA) is connected in a daisy chain model.
3. In daisy chain fashion the signal is allowed to propagate serially through the devices.
4. When more than one devices issue an interrupt request, the INTR line is activated and processor responds by setting the INTA line.
5. This signal is received by device 1. Device 1 passes the signal to the device 2 only if it requires any service.
6. If device 1 requires service, it blocks the INTA line and puts its identification code on the data lines. Therefore, in daisy chain arrangement, the device that is electrically closest to the processor has the highest priority.