COMMON BUS SYSTEM.

Chapter 5.

Basic Computer ← 8 Registers, A Memory Unit, Control Unit

Paths must be provided to transfer information from 1 Register to another & b/w Memory & Registers.

No. of Wires will be Excessive if Comm. are made b/w Outputs of each register & the Inputs of the other registers. A more efficient scheme is to transfer info using Common Bus.

In Last Lect., We have studied, how to Construct a bus system using MUX or 3 state buffer gates.

Bus Output is selected using S_2 S_1 S_0 Selection lines.

Eg:- AR is 3.

The Output of AR is placed onto bus when $S_2 S_1 S_0 = 011$

:→ Register whose LD input enabled Rec. the data from the bus.

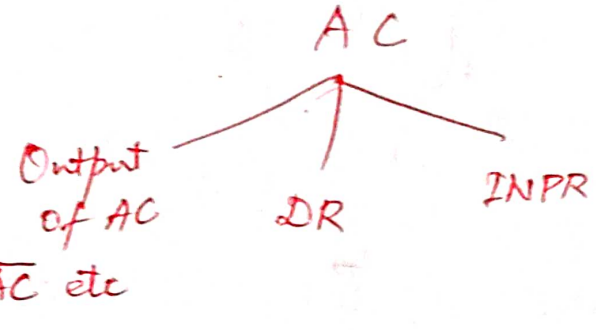
* Memory rec the contents of the bus when its Write input is Activated.
* Memory places its 16 bit Output onto the bus when read input is activated & $S_2 S_1 S_0 = 111$.
* 4 Registers AR, AC, IR & TR — 16 bits
  PC — 12 bits (∵ of Address)
  ↓
  4 Higher bits are set to 0 while sending/rec Data.

INPR, OUTR — 8 bits

↙ Provides info to the bus

↘ Receives the info from bus through AC.

Memory Address :- AR is used to specify a memory address.
By using AR, we eliminate the need for AD BUS
Inputs of AC come from ADDER & LOGIC Circuit.
AC sec input from 3 sets.



eg:- $\overline{AC}$ etc

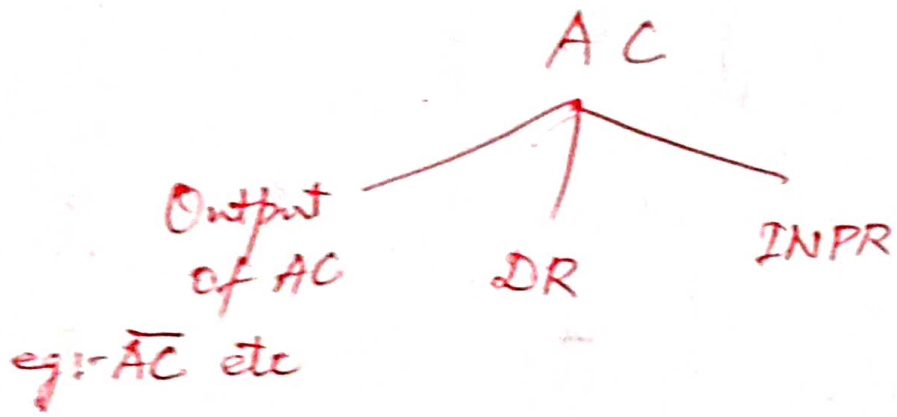
Eg:- DR ← AC & AC ← DR

Memory Address :- AR is used to specify a memory address.

By using AR, we eliminate the need for ADD BUS

Inputs of AC come from ADDER & LOGIC Circuit.

AC see input from 3 sets.



Eg:- DR ← AC & AC ← DR

Amritsar College of Engg. & Technology
AMRITSAR.

Nº 009460

Chapter 5

Name..Parvitar..Singh.....
College Roll No.....
Branch..C.S.E...Sem..III.....
Subject..Comp: Arch.....
Test I/II/III .....
Date....13/10/11.....

Signature of Invigilator

No. of Sheet attached.....

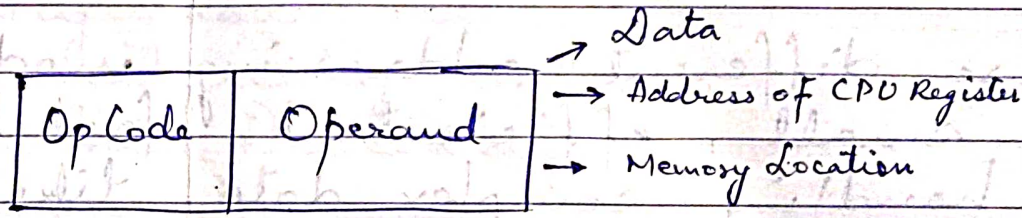
V.V. Imp

5. ADDRESSING MODES

COMPUTER ARCHITECTURE
CS-201

INTRODUCTION :-

A term addressing mode refer's to the mechanism for specifying operand's. An operand may be specified as the part of the instruction or the reference of the memory location's can be given. An operand could also be an address of the CPU Registers.



DEFINITION :-

The operand (data) of an instruction is stored in main memory or CPU registers. Different methods are used for specifying the operand in an instruction are called Addressing modes. A computer may use one or more addressing mode in its organization.

✱ Need of Addressing Modes :-

The choice of addressing mode govern's
- i) Efficiency (time as well as space)
- ii) Economy
- iii) Programming Flexibility.

These different addressing modes are helpful in writing small, efficient & fast programs. These modes are handling complex data like indexing of an array, looping, program relocation & pointers to memory.

Register addressing is very efficient as no memory fetch is required to bring the data. In addition, very few bits are required to address a register. Thus, we save both in terms of time and space.

"Types of Addressing Modes"

1. Immediate addressing mode.
2. Implied addressing mode.
3. Direct or absolute addressing mode.
4. Indirect addressing mode.
5. Register addressing mode.
6. Register Indirect addressing mode.
7. Auto Increment or Auto Decrement
8. Relative Addressing mode.
9. Index Addressing mode.
10. Base Register Addressing mode.

1. Immediate Addressing :-

Operand is mentioned explicitly in the instruction. The instruction is having operand field rather than address field. Hence, no fetch operation is required. This mode provides data to its op code immediately. That is why this mode is called Immediate Addressing Mode.

Syntax :-

Op Code	Operand

Disadvantages :-

(i) The value of data is limited by the length of the operand field in the instruction.

Advantages :-

(i) No additional memory fetch accesses are reqd for fetching the operand.

(ii) It is very fast.

Examples :-

(i) Add R_1, 15

$R_1 \leftarrow R_1 + 15$

(ii) Move R_0, 300

$R_{i0} \leftarrow 300$

or

MOV R_0, #300

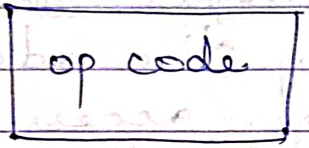
In some assembly language, '#' is put in front of immediate data.

Inherent or

ii) Implied or Implicit Addressing Mode :-

In this mode, the operand's are specified implicitly in the definition of instruction. This means operands are not present in the instruction.

Syntax:



Examples:-

Top of stack

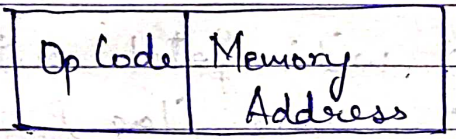
(i) POP → It will fetch out the contents from TOS
(ii) STC (Set carry flag)
(iii) HLT (stop processing)
(iv) CMP (complement AC)
(v) CIR (circular shift right)
(vi) CIL (Circular shift left)
(vii) INC (Increment AC)

All these instructions do not have any specified operand field. This type of addressing mode is limited to specified no. of instructions.

(iii) Direct Addressing Mode :-

In this mode, the operands are residing in memory. The address of the memory location where operands are residing is specified in the instruction itself as memory address. That means the address of the operand is given by the address field of the inst. Here only one memory access is required for operand fetch. That is why it is termed as direct addressing

Syntax



Effective Address

Example's :-

(i) LDA M[X]

AC ← M[X]

This instruction will load the contents of memory location having address X to AC.

(ii) Mov R1, M[X]

R1 ← M[X]

Amritsar College of Engg. & Technology
AMRITSAR.

Nº 009461

Addressing Modes Continue .....

(1)

Signature of Invigilator

No. of Sheet attached _____

Name.....

College Roll No.....

Branch.....Sem.....

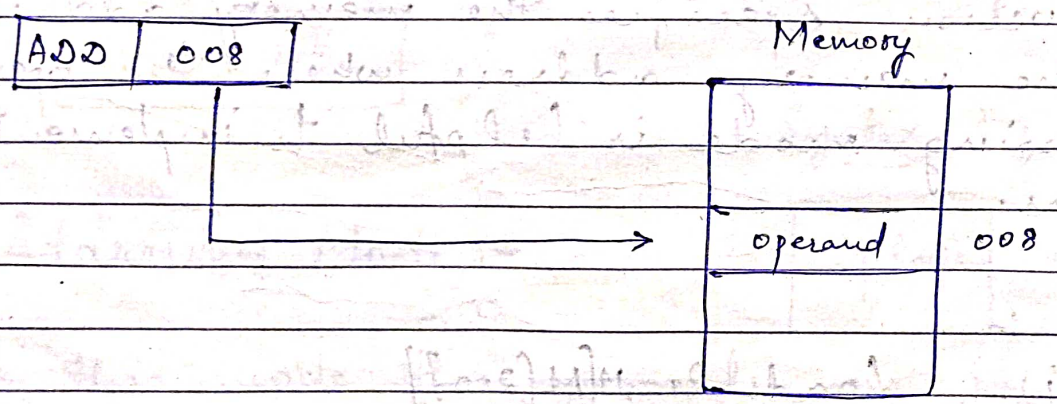
Subject.. Comp:. Architecture

Test I/II/III .....

Date 13/10/2011

i) ADD 008

The data value found at memory address 008 is added into AC.



AC = 500

M[008] = 100

After Execution :- AC ← AC + M[008]

AC ← 500 + 100

AC ← 600

AF → address field

4. Indirect Addressing Mode :-

Syntax :-

OpCode	Memory Address

Effective Address = Memory [AF]

The memory address field of the instruction specifies the memory address where the operands are residing. That means the instruction specifies the memory address which further contains the memory address where the operand is residing. This addressing mode is helpful to implement the Concept of pointers.

Examples :-

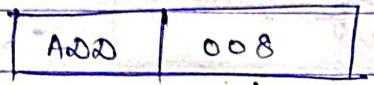
(i) Load R1, M[M[200]]

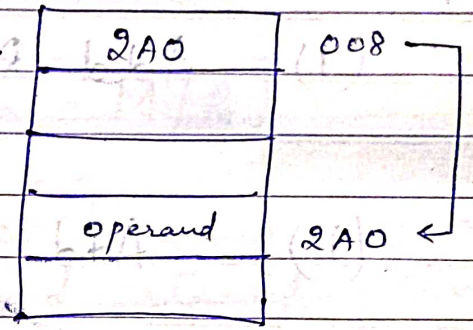
M[200]	300
M[300]	operand (400)

R1 ← 400

(ii) Add 008

Instruction.





AC = 320

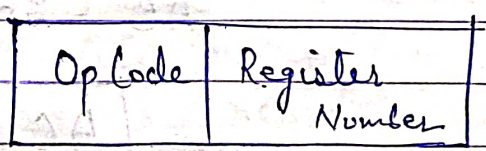
operand = 100

AC ← 320 + 100

AC = 420

5. Register Addressing Mode :-

Syntax :-



In this mode, the operands are in the one of the CPU registers. The register address (or no) is given in the instruction. This addressing mode provides fast access of operands without any memory access. It is even faster than direct addressing mode.

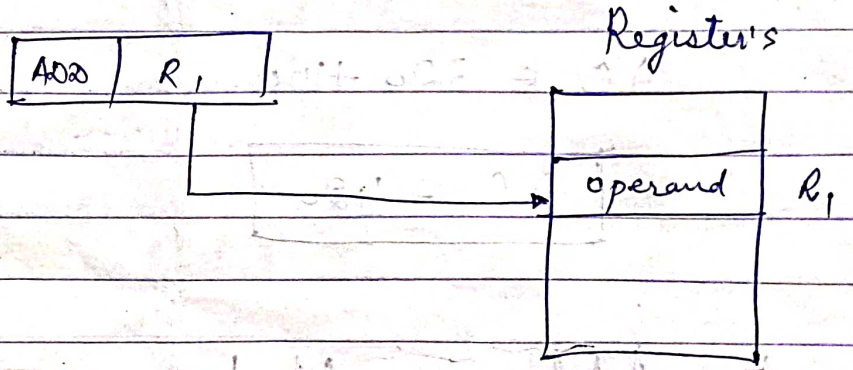
This mode is also called register direct addressing mode.

Example's :-

(i) Add R_1, R_2

$R_1 \leftarrow R_1 + R_2$

(ii) Add R_1



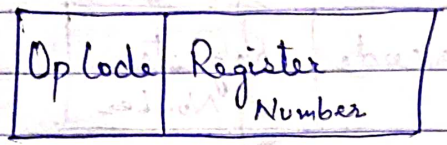
$R_1 = 300$

$AC = 200$

$AC \leftarrow 500$

6. Register Indirect Addressing Mode :-

Syntax :-



Effective Address = M[R]

In this mode, register contains the memory address of operand rather than the operand itself. That means register contains the address of operand than that of operand itself. This mode uses register pair to contain 16 bit address of operand

Example :- LDAX B AC ← M[BC]

LDAX B means load the AC with the contents of memory location whose address is residing in the register pair BC.

This mode is helpful for quick access of memory location in array.

When the address stored in the register refers to a table of data in memory, it is necessary to inc. or dec the register after every access to a table. This can be achieved using this mode.

7. Auto Increment or Auto Decrement mode :-

Syntax :-

Opcode	Register Number

In auto-increment the register value is incremented after the execution of the instruction.

In auto-decrement the register value is decremented before the execution of instruction. This is required in sequential execution of the instructions in the program.

Example :-

* (i) LDAX B AC ← M[BC]

LDAX B means load AC with contents of memory location whose addressing is residing in register pair B & C.

In auto increment the value of register pair BC is incremented after the execution of instruction.

In auto decrement the value of register pair BC is decremented before the execution of the instruction.

④ MOVE (R2), +R0 :- It copies the contents of R0 to R2 Then the value is inc. by 1.

⑤ MOVE &R1, -(R0) :- Initially content R0 is dec. Then copied to R1.

(c) Move R1, 1010 → list of elements is stored starting from 1010
Add AC, (R1)+ /* contents of mem. loc. 1010 are added to AC & subsequently, R1 is inc by 1 */

Thus,

ADD AC, (R1)+

⇓

ADD AC, R1
R1 ← R1+1

After accessing the operand, the contents of this register are incremented to point to the next item in the list (array).

8. Relative Addressing Mode :-

Syntax :- | Opcode | Offset |

EA = Address field of inst.
+
Content's of PC

In this mode, the address part (offset) of instruction is added to program counter to get the effective addr. At this EA, the operands are residing in memory.

Since, offset part can be -ive or +ive.

Examples:- (i) JUMP +8 (PC)

EA = PC + 8

PC = 300 ✓ (Offset Part is +8)

∴ EA = 308

i.e at 308, operands are residing.

(ii) JUMP -8 (PC)

EA = PC - 8

∴ EA = 292

i.e at 292, operands are residing.

⇒ This mode is used for branch address.

⇒ This mode provides quick address calculation with memory acces

Amritsar College of Engg. & Technology
AMRITSAR.

Nº 009462

Addressing Modes
Continue .....

(ii)

Signature of Invigilator No. of Sheet attached _____

Name.....
College Roll No.....
Branch.....Sem.....
Subject.....
Test I/II/III .....
Date.....

9) Index Addressing Mode :-

Syntax :-

Op Code	Address

In this mode, the address part of instruction is added to the contents of index register to get the effective address. Index register is one of the registers of CPU containing index value. That index value is termed as offset or displacement.

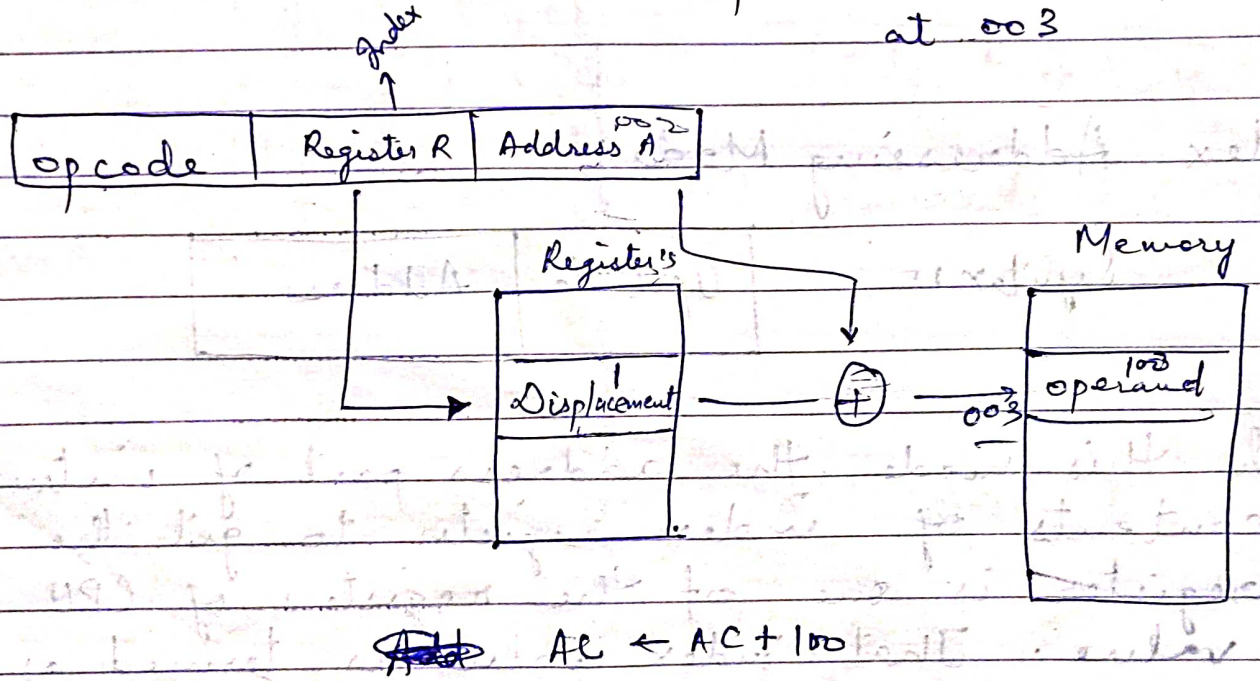
EA = Address field + Contents of Index Register.

Example:- Add 002

R_1 → Index Register 1 holds the value 1

EA = 002 + 1 = (003)

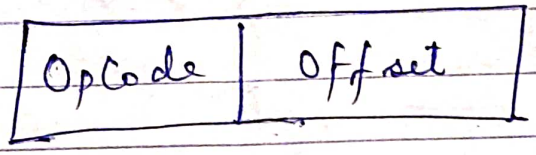
operand resides at 003



~~Add~~ AC ← AC + 100

(10) Base Register Addressing Mode :-

Syntax :-



EA = Address field + Contents of Base Register.

In this mode, the offset part of instruction is added to the contents of base register to get the EA. At this EA the operand is residing.

Usage :-

(i) It is quite useful for accessing array elements as well as characters in strings.

(ii) Relocation of program in memory. That is generally required in multiprogramming environment