GAIN MORE KNOWLEDGE
REACH GREATER HEIGHTS

PRESIDENCY COLLEGE
(Autonomous)

COMPUTER ARCHITECTURE

Reaccredited by NAAC with A+

By
Mr N Kartik / Ms Vasantha
Assistant Professor, Dept of Computer Applications,
Presidency College, Bangalore-24

Presidency Group
OVER 40 YEARS OF ACADEMIC WISDOM

GAIN MORE KNOWLEDGE
REACH GREATER HEIGHTS

PRESIDENCY COLLEGE
(AUTONOMOUS)
AFFILIATED TO BENGALURU CITY UNIVERSITY, APPROVED BY AICTE, DELHI & RECOGNISED BY THE GOVT. OF KARNATAKA
RE-ACCREDITED BY NAAC WITH 'A+' GRADE

GAIN MORE KNOWLEDGE
REACH GREATER HEIGHTS

Presidency College
(Autonomous)

Reaccredited by NAAC with A+

Presidency Group

OVER 40 YEARS OF ACADEMIC WISDOM

Syllabus

- Instruction codes
- Computer Registers
- **Computer instructions and Instruction cycle**
- **Timing and Control**
- Memory-Reference Instructions
- Input-output, and interrupt
- Central processing unit: Stack organization
- **Instruction Formats**
- **Addressing Modes**
- Data Transfer and Manipulation
- Complex Instruction Set Computer (CISC)
- Reduced Instruction Set Computer (RISC)

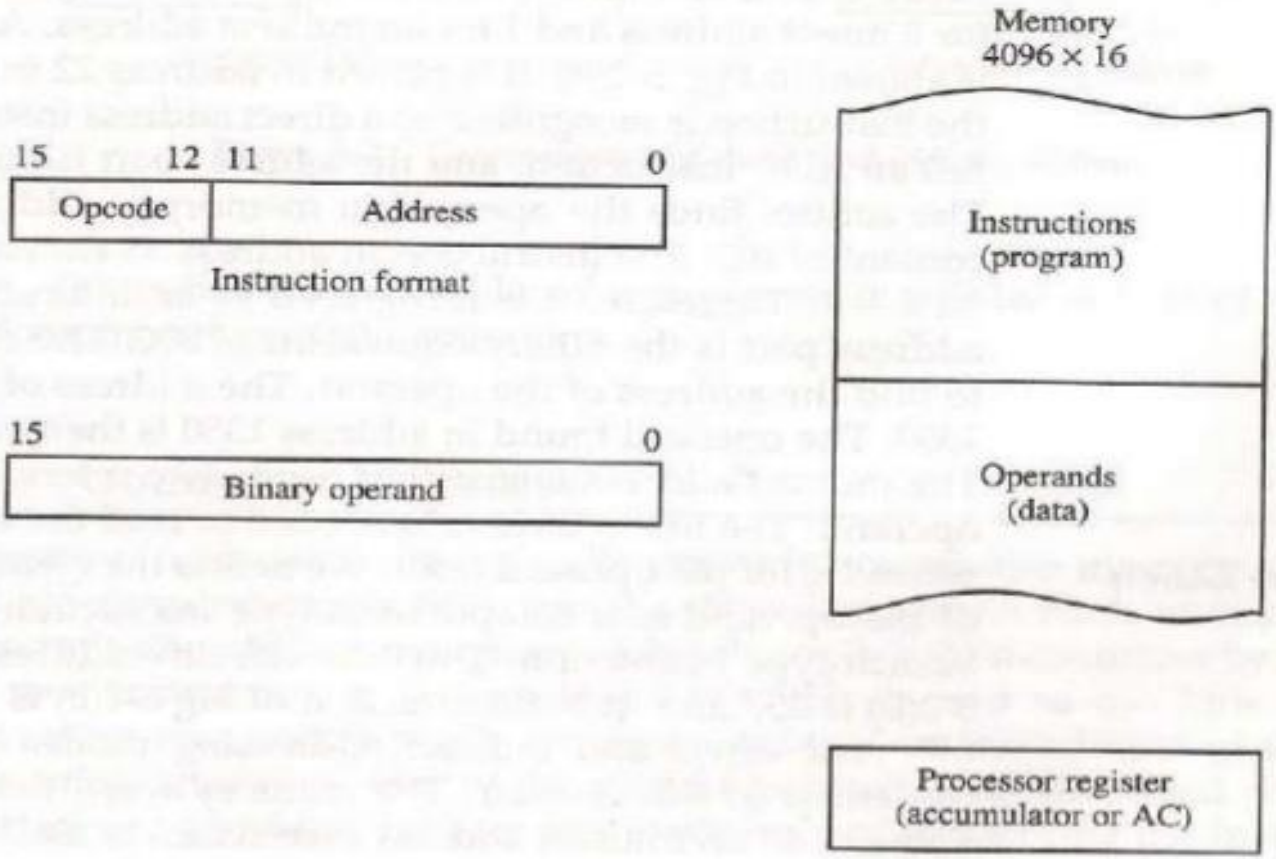
Instruction Codes

- The user of a computer can control the process by means of a program
- **A program is a set of instructions that specify the operations, operand, and the sequence (control)**
- **A instruction is a binary code that specifies a sequence of microoperations**
- Instruction codes together with data are stored in memory (Stored Program Concept)

Instruction Codes

- The computer reads each instruction from memory and places it in a control register. The control then interprets the binary code of the instruction and proceeds to execute it by issuing a sequence of micro operations.
- **Instruction Code :** A group of bits that instruct the computer to perform a specific operation. It is usually divided into parts
- **Operation Code :** The most basic part of an instruction code
- A group of bits that define such operations as add, subtract, multiply, shift, and complement

Figure Stored program organization.



Computer Instructions

- 3 instruction code formats
- Each format has 16 bits
- Operation code (op code) part of the instruction contains three bits and the meaning of the remaining 13 bits depends on the op code encountered.

Computer Instructions

- **Memory reference instruction** uses 12 bits to specify an address and one bit to specify the addressing mode I. I=0 for direct, I=1 for indirect.
- **Register reference instructions** are recognized by op code 111 with a 0 in the leftmost bit.
- **Input output instruction** is recognized by op code 111 with a 1 in the leftmost bit.
- Register and I/O does not need a reference to memory so remaining 12 bits are used to specify the operation or test to be executed.

Computer Instructions

Memory-Reference Instructions (OP-code = 000 ~ 110)

Register-Reference Instructions (OP-code = 111, I = 0)

Input-Output Instructions (OP-code =111, I = 1)

Computer Instructions

Symbol	Hex Code I = 0	Hex Code I = 1	Description
AND	0xxx	8xxx	AND memory word to AC
ADD	1xxx	9xxx	Add memory word to AC
LDA	2xxx	Axxx	Load AC from memory
STA	3xxx	Bxxx	Store content of AC into memory
BUN	4xxx	Cxxx	Branch unconditionally
BSA	5xxx	Dxxx	Branch and save return address
ISZ	6xxx	Exxx	Increment and skip if zero
CLA	7800		Clear AC
CLE	7400		Clear E
CMA	7200		Complement AC
CME	7100		Complement E
CIR	7080		Circulate right AC and E
CIL	7040		Circulate left AC and E
INC	7020		Increment AC
SPA	7010		Skip next instr. if AC is positive
SNA	7008		Skip next instr. if AC is negative
SZA	7004		Skip next instr. if AC is zero
SZE	7002		Skip next instr. if E is zero
HLT	7001		Halt computer
INP	F800		Input character to AC
OUT	F400		Output character from AC
SKI	F200		Skip on input flag
SKO	F100		Skip on output flag
ION	F080		Interrupt on
IOF	F040		Interrupt off

Computer Instructions

- The set of instructions are said to be complete if the computer includes enough instructions in each of following categories:

  1. Arithmetic, logical and shift instructions
  2. Instructions for moving information to and from memory and processor registers.
  3. Program control instructions together with instructions that check status condition.
  4. Input and output instructions

Timing and Control

- The timing for all registers in the basic computer is controlled by a master clock generator.

Timing and Control

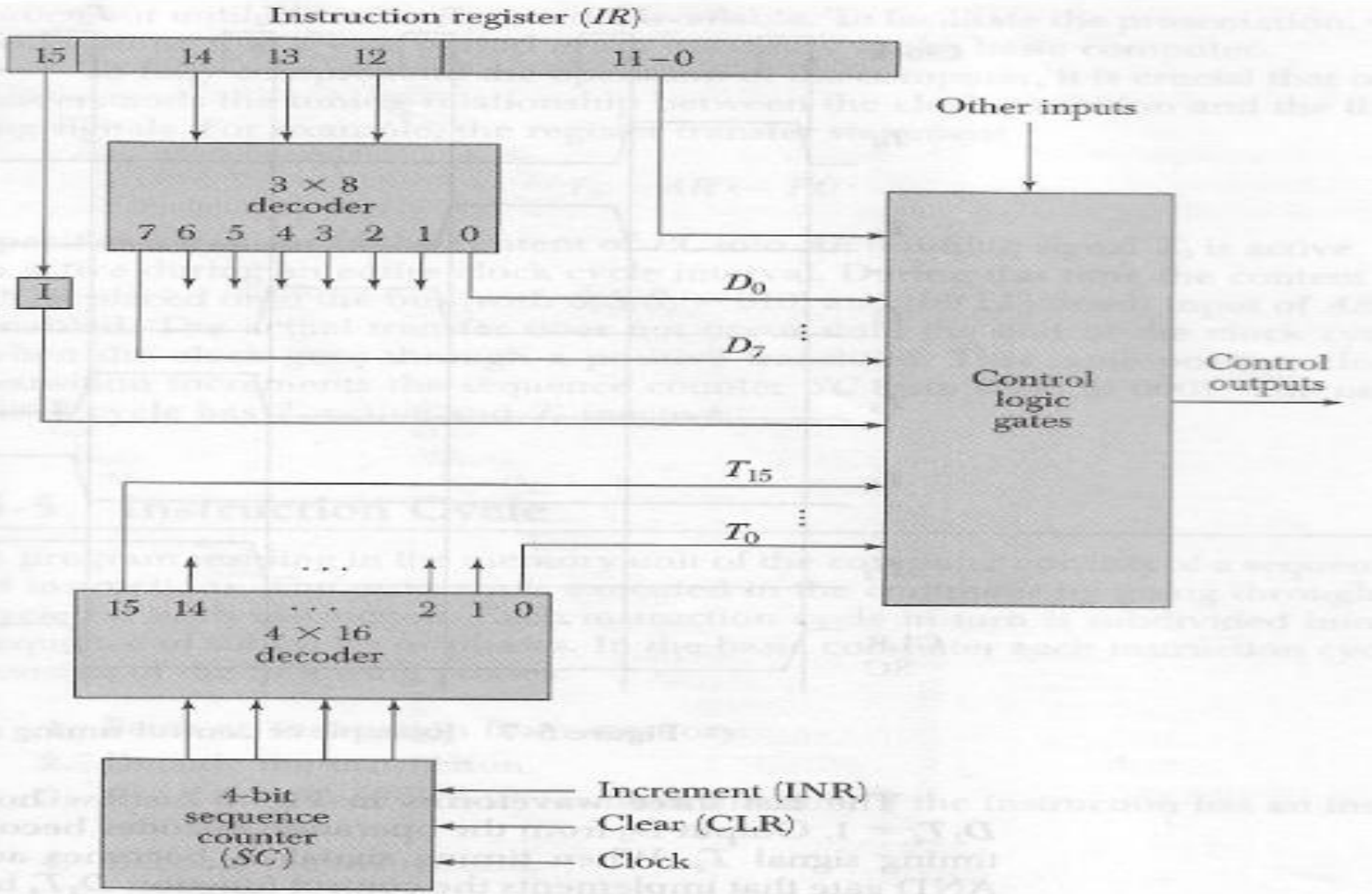


Figure 5-6 Control unit of basic computer.

Timing and Control

- Block Diagram-

  2 Decoders, 1 Sequence counter, number of control logic gates
- An instruction read from memory is placed in the instruction register (IR).
- Instruction register is divided into three parts:

  i) the I bit

  ii) the operation code

  iii) bits 0 through 11

Timing and Control

- Opcode in bits 12 through 14 are decoded with a 3X8 decoder. Eight outputs of the decoder are designated by the symbols D_0 through D_7.
- Bit 15 of the instruction is transferred to a flip flop designated by symbol I.
- Bits 0 through 11 are applied to the control logic gates.
- The 4 bit sequence counter can count in binary from 0 through 15.
- The outputs of the counter are decoded into 16 timing signals T_0 through T_{15}.

Timing and Control

- The sequence counter can be incremented or cleared synchronously.
- The counter is incremented to provide the sequence of timing signals, counter can be cleared.

Instruction Cycle

- In the basic computer each instruction cycle consists of following phases:
  1. Fetch an instruction from memory
  2. Decode the instruction
  3. Read the effective address from memory if the instruction has an indirect address
  4. Execute the instruction

Instruction cycle-Fetch and Decode

- Program counter (PC) is loaded with the address of first instruction in the program
- The sequence counter SC is cleared to 0, providing a decoded timing signal T_0. After each clock pulse, SC is incremented by 1, so that timing signal go through sequence T_0, T_1, T_2…
- Micro operation for fetch and decode
- **T_0: AR<-PC**

  T_1: IR<- M(AR), PC<-PC+1

  T_2: D_0…D_7<-Decode IR(12-14), AR<-IR(0-11), I<-IR(15)

Instruction cycle-Fetch and Decode

- Since only AR is connected to the address inputs of memory, it is necessary to transfer the address from PC to AR during the clock transition associated with timing signal T_0
- The instruction read from memory is then placed in the instruction register IR with the clock transition associated with the timing
  signal T_1. At the same time PC is incremented by one to prepare it for the address of the next instruction in the program
- At time T_2, the operation code in IR is decoded, the indirect bit is transferred to flip flop I, and the address part of the instruction is transferred to AR

GAIN MORE KNOWLEDGE
REACH GREATER HEIGHTS

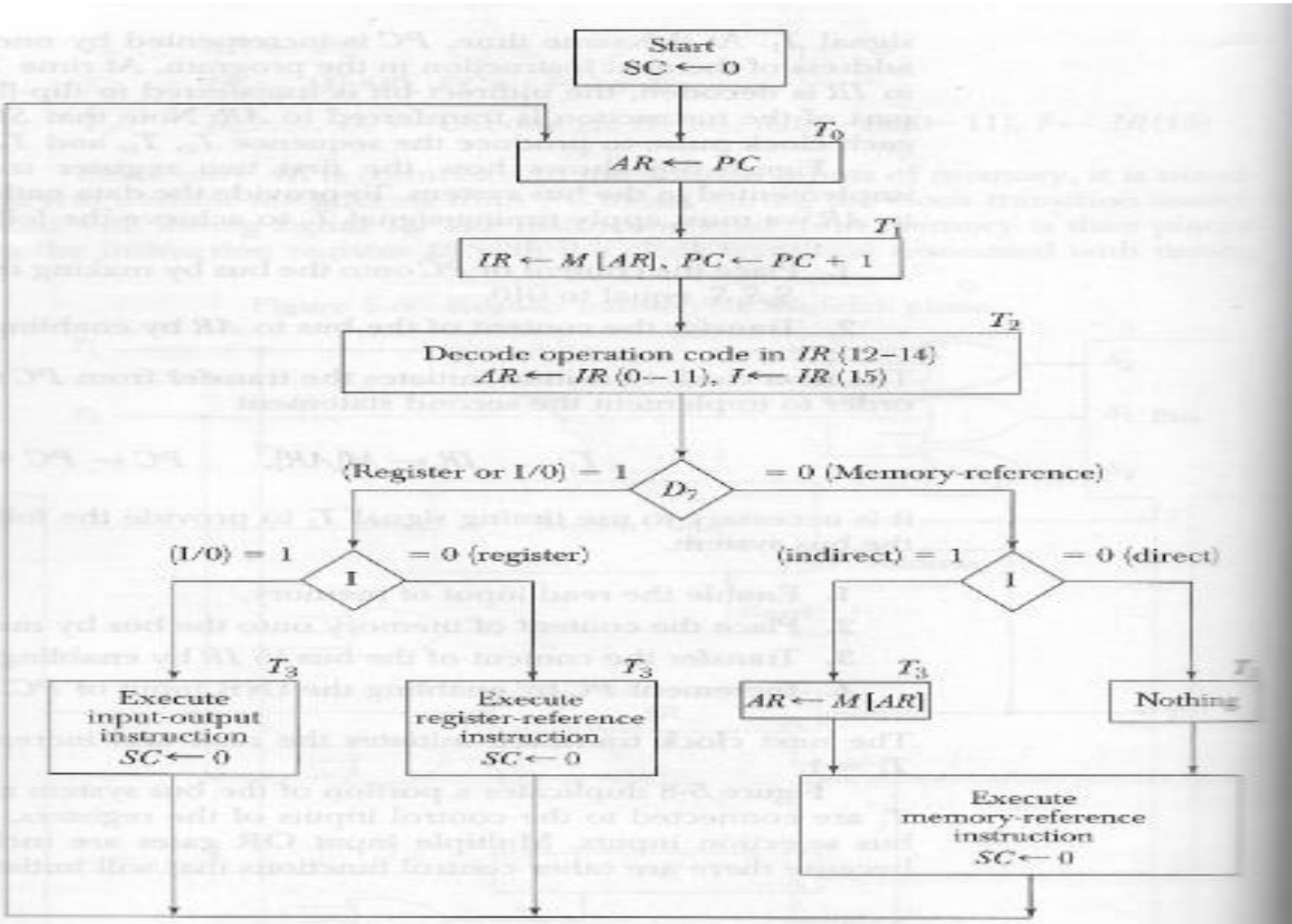
Presidency College
(Autonomous)

Reaccredited by NAAC with A+

Presidency Group

OVER 40 YEARS OF ACADEMIC WISDOM

Instruction cycle



Instruction Formats

- The bits of the instruction are divided into groups called fields
- Most common fields found in instruction formats are:
  1. An operation code field that specifies the operation to be performed.
  2. An address field that designates a memory address or a processor register.
  3. A mode field that specifies the way the operand or the effective address is determined.

Types of instruction format

- Three address instruction
- Two address instruction
- One address instruction
- Zero address instruction

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

Three Address Instruction

- Computers with three address instruction formats can use each address fields to specify either a processor register or a memory operand
- Advantage- It results in short programs when evaluating arithmetic expression
- Disadvantage- binary coded instructions require too many bits to specify three address

Three Address Instruction

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

```
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

- Each address field can specify either a processor register or a memory word

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

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

- Uses an implied accumulator (AC) register for all data manipulation

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

** T is the address of a temporary memory location required for storing the intermediate result

Zero Address Instruction

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

-

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

**TOS-Top of stack

Addressing Mode

- The addressing mode gives or indicates a rule to identify the operand location
- Different addressing modes:
  1. Implied mode
  2. Immediate mode
  3. Register mode
  4. Register indirect mode
  5. Auto increment or auto decrement mode
  6. Direct address mode
  7. Indirect address mode
  8. Relative address mode
  9. Indexed addressing mode
  10. Base register index mode

PC, Mode Field, EA

- Program Counter (PC) – PC is a register that keeps track of the instructions in the program stored in memory
- Mode field- the mode field is used to locate the operands needed for the operation

Opcode	Mode	Address

- Effective address (EA)- it is defined to be the memory address obtained from the computation dictated by the given addressing mode

GAIN MORE KNOWLEDGE
REACH GREATER HEIGHTS

Presidency College
(Autonomous)

Reaccredited by NAAC with A+

Presidency Group

OVER 40 YEARS OF ACADEMIC WISDOM

Implied Mode

- In this mode the operands are specified implicitly in the definition of the instruction
  - Ex- Accumulator and shift based
  - CLA- Clear accumulator
  - CMA-Complement accumulator

Immediate Mode

- In this mode the operand is specified in the instruction itself
- These instructions are useful for initializing registers to a constant value
- Second part of data contain address only
  - Ex ADD 5 AC<-AC+5

Register Mode

- In this mode the operands are in registers that reside within CPU
- Ex ADD R1 AC<-AC+R1

Auto increment and decrement

- The register is incremented or decremented after its value is used to access memory
- When the address stored in the register refers to a table of data in memory, it is necessary to increment or decrement the register

Direct Address Mode

- In this mode the effective address is equal to the address part of the instruction
- The operands resides in memory and its address is given directly by the address field of the instruction
- Second part of instruction holds the address of operand
- Ex- ADD 1000

  1000 (address)->50 (operand)

Indirect Address Mode

- In this mode the address field of the instruction gives the address where the effective address is stored in memory
- Effective address = address part of instruction + content of CPU register
- Ex- ADD 1000(address)

  1000(address)- 1100(address)

  1100(address)- 50(operand)

Relative Address Mode

- In this mode the content of the program counter is added to the address part of the instruction in order to obtain the effective address
- **Effective address = PC + address part of the instruction**

GAIN MORE KNOWLEDGE
REACH GREATER HEIGHTS

Presidency College
(Autonomous)

Reaccredited by NAAC with A+

Presidency Group

OVER 40 YEARS OF ACADEMIC WISDOM

Indexed Addressing Mode

- In this mode the content of an index register is added to the address part of the instruction to obtain the effective address.
- **Effective address = Index Register value + address part of the instruction**

Base Register Addressing Mode

- In this mode the content of a base register is added to the address part of the instruction to obtain the effective address
- **Effective address = Base Register value + address part of the instruction**

Data Transfer and Manipulation

- Most computer instructions can be classified into three categories:

  1. Data transfer instructions
  2. Data manipulation instructions
  3. Program control instructions- provide decision making capabilities and change the path

Data Transfer Instructions

- Cause transfer of data from one location to another without changing the binary information content
- Most common transfers are:

1. Between memory and processor registers
2. Between processor registers and input or output
3. Between the processor registers themselves

GAIN MORE KNOWLEDGE
REACH GREATER HEIGHTS

Presidency College
(Autonomous)

Reaccredited by NAAC with A+

Presidency Group

OVER 40 YEARS OF ACADEMIC WISDOM

Data Transfer Instructions

Name	Mnemonic
Load	LD
Store	ST
Move	MOV
Exchange	XCH
Input	IN
Output	OUT
Push	PUSH
Pop	POP

Data Manipulation Instructions

- The data manipulation instructions in a typical computer are divided into-
- i. Arithmetic instruction

  ii. Logical and bit manipulation instructions

  iii. Shift instructions

Arithmetic Instruction

Name	Mnemonic
Increment	INC
Decrement	DEC
Add	ADD
Subtract	SUB
Multiply	MUL
Divide	DIV
Add with carry	ADDC
Subtract with borrow	SUBB
Negate (2's complement)	NEG

Arithmetic Instruction

- A special carry flip flop is used to store the carry from an operation
- The instruction "add with carry" performs the addition on two operands plus the value of carry
- The "subtract with borrow" instruction subtracts two words and a borrow which may have resulted from previous subtraction
- Mnemonics for three add instruction that specify different data types:
  - ADDI- add two binary integer
  - ADDF- add two floating point number
  - ADDD- add two decimal numbers in BCD

Logical and Bit Manipulation Instructions

- Logical instructions perform binary operations on strings of bits stored in registers

Name of instruction	Mnemonic
Clear	CLR
Complement	COM
AND	AND
OR	OR
Exclusive OR	XOR
Clear Carry	CLRC
Set Carry	SETC
Complement carry	COMC
Enable interrupt	EI
Disable interrupt	DI

Shift Instructions

Name	Mnemonics	
Logical shift right	SHR	/2
Logical shift left	SHL	*2
Arithmetic shift right	SHRA	
Arithmetic shift left	SHLA	
Rotate right	ROR	
Rotate left	ROL	
Rotate right through carry	RORC	
Rotate left through carry	ROLC	

CISC

- A computer with a large number of instructions is classified as complex instruction set computer
- Characteristics-

  1. A large number of instructions- typically from 100 to 250 instructions
  2. Some instructions that perform specialized tasks and are used infrequently
  3. A large variety of addressing modes- typically from 5 to 20 different modes
  4. Variable length instruction format
  5. Instruction that manipulate operands in memory

RISC

- RISC involves an attempt to reduce execution time by simplifying the instruction set of the computer.
- Characteristics-
  1. Relatively few instructions
  2. Relatively few addressing modes
  3. Memory access limited to load and store instructions
  4. All operations done within the registers of the CPU
  5. Fixed length, easily decoded instruction format
  6. Single cycle instruction execution
  7. Hardwired rather than micro programmed control