Q. No	Question (s)	Marks	BL	CO
	UNIT - II			
1	**a.** Define microprogrammed control unit	**1M**	L2	C213.1
	b. Name the three types of CPU Organizations	**1M**		
	c. Name any five addressing modes	**1M**		
	d. Which status bits are affected after execution of COMPARE Instruction	**1M**		
	e. Write the relation between the number of processor registers and the address bits required to address each register	**1M**		
2	**a.** Explain the following instructions. i. PUSH, ii. POP, iii. CMP	**3M**		
	b. Discuss about micro program sequencer.	**3M**		
	c. Explain the following instructions: i. BR and ii. JMP	**3M**		
	d. Explain address sequencing for control memory	**3M**		
	e. Differentiate between a branch instruction, a call subroutine instruction, and a program interrupt.	**3M**		
3	**a.** List and explain the unsigned CMP Program Control Instructions.	**5M**		
	b. With a neat diagram, the working of microprogram sequencer	**5M**		
	c. Explain with a neat diagram, how the status bits are affected during a ALU operation	**5M**		
	d. Explain the mapping of instructions.	**5M**		
	e. explain the signed CMP Program Control Instructions.	**5M**		
4	**a.** For the operation X = (A + B) x (C + D) , write i. Three address instruction code, ii. Zero Address Instruction code and iii. RISC Instruction set Where, A, B, C, D, X are memory locations	**10M**		
	b. Explain following addressing modes with a numerical example i. Direct Addressing Mode ii. Register Indirect Addressing Mode iii. Relative addressing mode iv. Indirect Addressing Mode	**10M**		
	c. Explain the Data Transfer instructions in detail.	**10M**		

ANSWERS:

1a. Define microprogrammed control unit
Ans: A control unit whose binary control variables are stored in the memory is called a microprogrammed control unit

1b. Name the three types of CPU Organizations
Ans*:*

i. Single Accumulator Organization
ii. General Register Organization
iii. Stack Organization

1c. Name any five addressing modes
Ans:

i. Implied Addressing Mode
ii. Immediate Addressing Mode
iii. Register Addressing Mode
iv. Register Indirect Addressing Mode
v. Autoincrement and Autodecrement Addressing Mode
vi. Direct Addressing Mode
vii. Indirect Addressing Mode
viii. Relative Addressing Mode
ix. Indexed Addressing Mode
x. Base Register Addressing Mode

1d. Which status bits are affected after execution of COMPARE Instruction?
Ans: carry, zero, sign status bits are affected.

1e. Write the relation between the number of processor registers and the address bits required to address each register
Ans: For 2^k registers we required k-bit address to represent a register

2a. Explain the following instructions. i. PUSH, ii. POP, iii. CMP
Ans:

PUSH: The operation of insertion is called *push* (or push-down) because it can be thought of as the result of pushing a new item on top. When an item is inserted into the stack, the SP is incremented by 1 and SP points to the top of the stack. When the stack is full, then FULL=1 and EMPTY=0

POP: The operation of deletion is called *pop* (or pop-up) because it can be thought of as the result of removing one item so that the stack pops up. When an item is deleted from the stack, the SP is decremented by one and SP points to the top of the stack. When all the items are deleted from stack, EMPTY=1 and FULL=0

CMP: is used to compare 2 operands. Basically, subtraction operations performed here. CMP instruction does not have source and destination, however upon execution CMP instruction, the status bits are affected.

2b. Discuss about micro program sequencer

Ans.

1) The control memory with fields F1, F2, F3, CD, BR, AD forms the heart of the operation. the CD output bits are connected as select line to MUX2. BR output bits are connecting as input to the input logic. AD bits are connecting to MUX1 as the address bits. The CD field is connected as select line to MUX2. MUX2 has 4 input lines with the input 1, I, S, Z. if bit selected is 1, then the T =1 else T=0.

2) Control Address Register (CAR) as we know, gets the address from 4 different paths viz. Incrementor, SBR, Mapping, and branched address bits (AD)

3) **Input logic** with 3 inputs I_0, I_1, T with 3 outputs S_0, S_1, L. Variables S_0, S_1 will decide one of the source addresses for CAR. Variable L enables the Load input in SBR. The binary values of the two selection variables determine the path in the multiplexer as shown below.

2c. Explain the following instructions: i. BR and ii. JMP

Ans.

2d. Explain address sequencing for control memory

Ans.

The transformation from the instruction code bits to the control memory where the routine is location is referred as *mapping* process. Once the required routine is reached, the microinstruction that executes the instruction may be sequenced by incrementing the control address register.

When the execution of the instruction is completed, the control must return to the fetch routine. This is accomplished by executing a unconditional branch microinstruction to the first address of the fetch routine.

The address sequencing capabilities required in a control memory are:

i. Incrementing the control address register.
ii. Conditional or unconditional branch, depending on status bit conditions.
iii. A mapping process from the bits of the instruction to an address for control memory.
iv. A facility for subroutine call and return.

The block diagram below shows the control memory and the associated hardware required to generate the address of next microinstruction. The microinstruction in the control memory contains set of bit to initiate the microoperation and other bits to specify how the next address is calculated.

The Control Address Register (CAR) receives address from 4 paths through a multiplexer as shown in the figure.

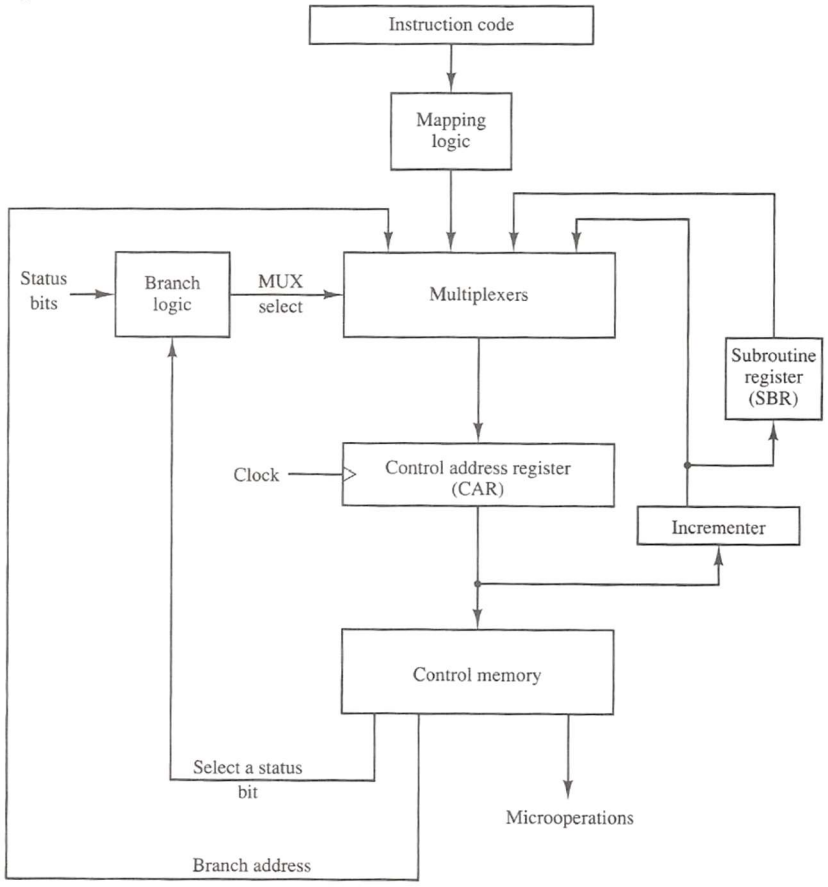
The incrementor increments the content of control address register by one, to select the next microinstruction in sequence.

Branching is achieved by specifying the branch address in one of the fields of microinstruction.

Conditional branching is achieved by using part of the microinstruction to select specific status bits in order to determine its condition.

An external memory is transferred into control memory via a mapping logic (*explained next*).

The return address for the subroutine in stored in a special register whose value is then used when the microprogram wished to return the address.



2e. Differentiate between a branch instruction, a call subroutine instruction, and a program interrupt

Ans:

BRANCH	CALL	INTERRUPT
Program control is transferred to a memory location which is in the main program	Program Control is transferred to a memory location which is not a part of main program	Interrupt is an event that indicates the CPU to perform a specific task immediately

Value of Program Counter (PC) is not transferred to stack	Value of Program Counter (PC) is transferred to stack	an interrupt is an event that is triggered by external components that alert the CPU to perform a certain action
After JUMP, there is no return instruction	After CALL, there is a return instruction	
Initialization of SP (Stack Pointer) is not mandatory	Initialization of SP (Stack Pointer) is mandatory	

3a. List and explain the unsigned CMP Program Control Instructions
Ans:

Unsigned compare conditions ($A - B$)

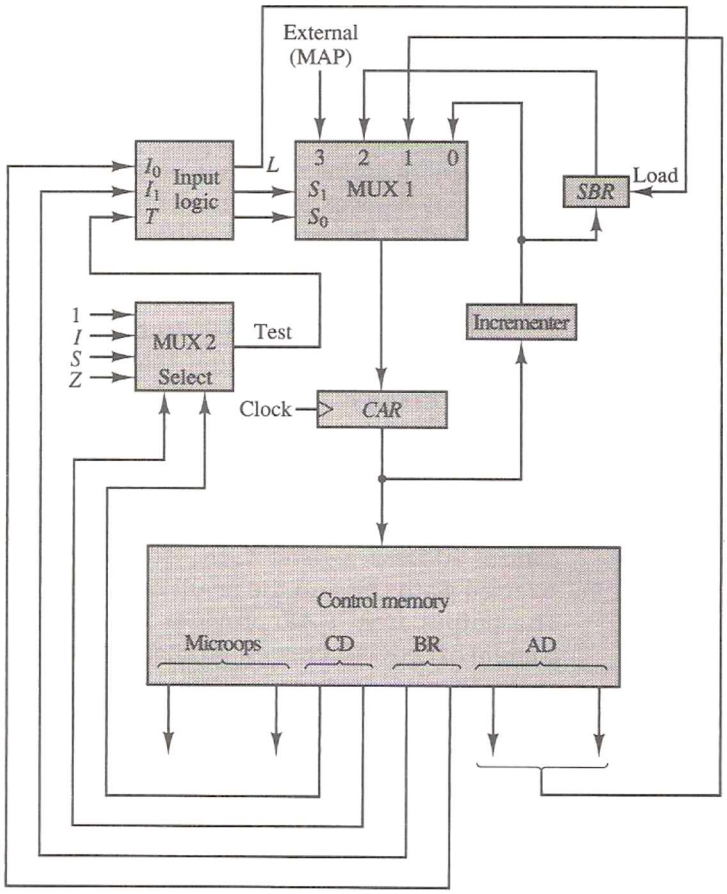
BHI	Branch if higher	$A > B$
BHE	Branch if higher or equal	$A \geq B$
BLO	Branch if lower	$A < B$
BLOE	Branch if lower or equal	$A \leq B$
BE	Branch if equal	$A = B$
BNE	Branch if not equal	$A \neq B$

The above instructions are applicable only for the unsigned comparison. The terminology used for the results after comparison are Higher, Equal and Lower

3b. With a neat diagram, the working of microprogram sequencer
Ans:

The fig shows the microprogram sequencer where in all the concepts learned earlier are used. Following are the blocks are their operation

4) The control memory with fields F1, F2, F3, CD, BR, AD forms the heart of the operation. the CD output bits are connected as select line to MUX2. BR output bits are connecting as input to the input logic. AD bits are connecting to MUX1 as the address bits. The CD field is connected as select line to MUX2. MUX2 has 4 input lines with the input 1, I, S, Z. if bit selected is 1, then the T =1 else T=0.

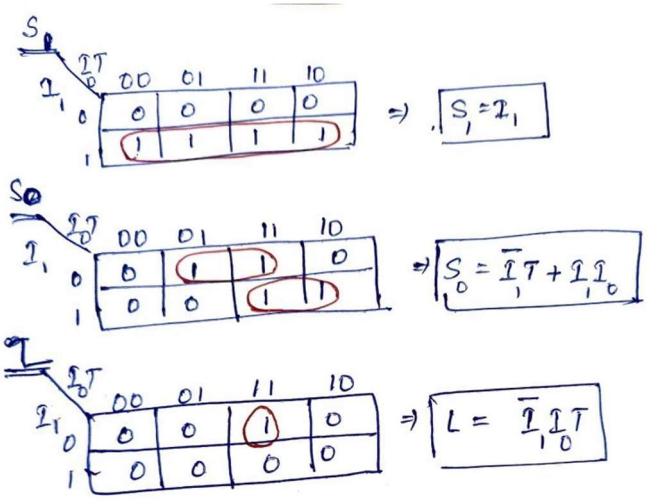


5) Control Address Register (CAR) as we know, gets the address from 4 different paths viz. Incrementor, SBR, Mapping, and branched address bits(AD)

6) **Input logic** with 3 inputs I_0, I_1, T with 3 outputs S_0, S_1, L. Variables S_0, S_1 will decide one of the source addresses for CAR. Variable L enables the Load input in SBR. The binary values of the two selection variables determine the path in the multiplexer as shown below.

BR Field	Input			MUX 1		Load *SBR*
	I_1	I_0	T	S_1	S_0	L
0 0	0	0	0	0	0	0
0 0	0	0	1	0	1	0
0 1	0	1	0	0	0	0
0 1	0	1	1	0	1	1
1 0	1	0	×	1	0	0
1 1	1	1	×	1	1	0

the expressions for L, S1 and S0 are calculated using a k-map as follows and we required 1 inverter, 1 OR gate and 3 AND gates



3c. Explain with a neat diagram, how the status bits are affected during a ALU operation

Ans:

Status bits are also called as Condition-codes or flag bits. The outcome of the operations in the ALU are connected to the status bits as shown below with a 4-bit status register. The status bits are *SIGN(S), ZERO(Z), CARRY(C), OVERFLOW(V):*

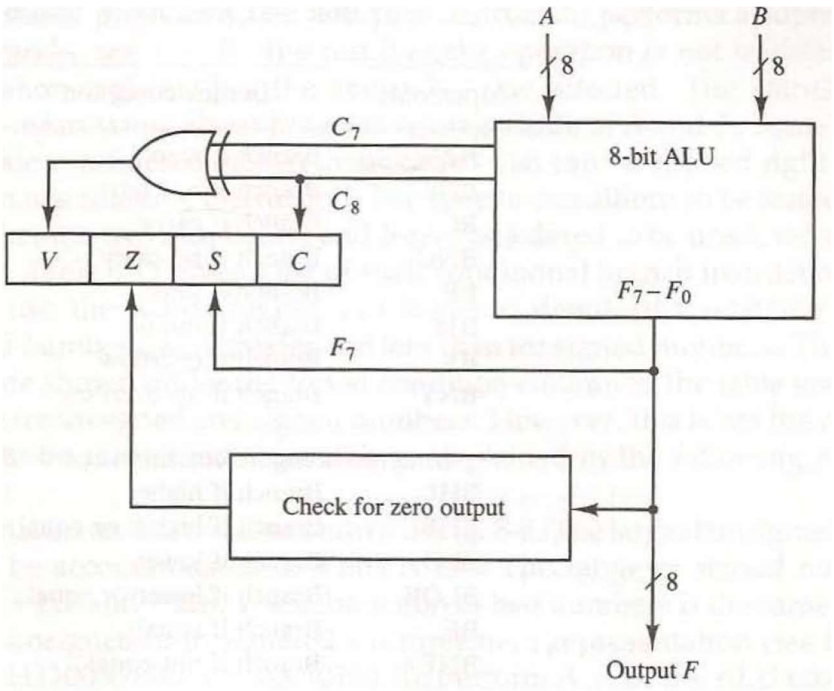
Consider 2 inputs A ($A_7 – A_0$) and B ($B_7 – B_0$), each of 8-bit in length is connected to the ALU. Let F ($F_7 – F_0$) be the output of ALU. Let the carry bits generated during ALU operation be C_0-C_7 and let C_8 be the final carry generated due to bits A_7 and B_7.

CARRY (C): *Bit C is set to 1 if the carry bit C8 is 1*
Bit C is reset/cleared to 0 if the carry bit C8 is 0.

SIGN(S): *Bit S is set to 1 if the bit F_7 is 1*
Bit S is reset/clared to 0 if the bit F7 is 0

ZERO(Z): *if all bits of result is 0 then bit Z is set to 1, else it is reset/cleared to 0*

OVERFLOW(V): overflow is the condition when negative numbers are in 2's complement representation.
If XOR of bits C7 and C8 results in 1, bit V is set to 1 and
if the XOR results in a 0, bit V is cleared/reset to 0



Example:

Carry bits= **1 1** 1 1 1 1 1 1 0 — carry bits C_7 and C_8 are 1

A = 1 0 1 0 0 1 1 1

\+ B = 0 1 1 1 1 0 1 1

F = **1** 0 0 1 0 0 0 1 0 — $F_7 = 0$

Thus, C = *1, V =0, Z=0, S= 0*

3d. Explain the mapping of instructions.

Ans:

Consider a computer with main memory capacity of 2048 x 16. The instruction format for this as shown below. It consists of 11-bit address, 4-bit opcode and 1-bit addressing mode. With 4-bit opcode we have 16 distinct operations.

15	14 — 11	10 — 0
I	**OPCODE**	**ADDRESS**

Assuming that the control memory capacity is 128 x 20. i.e., it has 128 control words with each control word of length 20-bits. To address 128(2^7) memory locations, we need 7-bit address.

We know that group of control words is a *Routine.* Let 4 control words form a routine; with this we have 16 routines. All these 16 routines have to be mapped to the 16 distinct operations of the instruction code. These routines are divided into two, first part is the *opcode routine (8)* and the second is *co-routine (8)* as shown in Fig.3

The 4-bit opcode of the instruction has to be mapped with the 7-bit address of the Routine. The mapping as done as follows

In Fig.4 by observation, the MSB of the opcode routine is 0 and the LSB bits of every routine is 00. By appending a 0 to the MSB and 00 to the LSB of opcode, mapping is done.

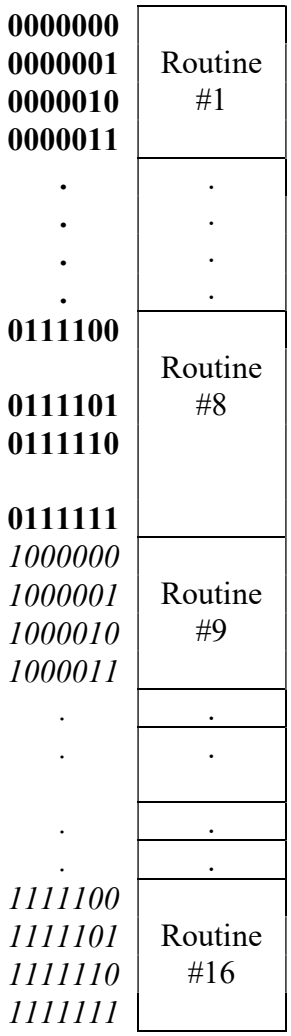


Fig:3 *Routines*

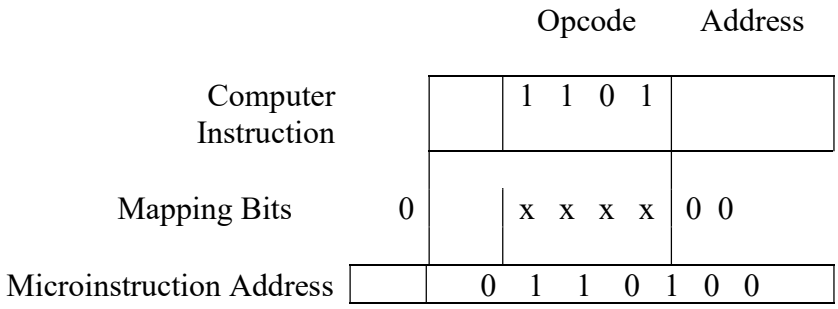


Fig:4 *Mapping from Instruction code to Microinstruction address*

3e) Explain the signed CMP program control instructions.
Ans:

Signed compare conditions ($A - B$)		
BGT	Branch if greater than	$A > B$
BGE	Branch if greater or equal	$A \geq B$
BLT	Branch if less than	$A < B$
BLE	Branch if less or equal	$A \leq B$
BE	Branch if equal	$A = B$
BNE	Branch if not equal	$A \neq B$

The above instructions are applicable for signed comparison, The terminology used for the results after comparison are Greater, Equal, Lesser

4a. Explain following addressing modes with a numerical example.
i. Three address instruction code
ii. Zero Address Instruction code and
iii. RISC Instruction set
Where, A, B, C, D, X are memory locations

Ans: i. Three Address Instructions: In this type, the instruction use *3 address fields.* Also, the address fields can be a register address and memory address

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

ii. Zero Address Instruction code: A stack is a storage device that stored information in such a manner that the items stored last is the first item retrieved. In simple words, stack is a last-in first-out (LIFO). Here 2 instructions PUSH and POP are used where PUSH is used to sent the data into the stack memory and POP is used to sent the data out from the stack memory

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

iii.RISC Instruction Set:
In the RISC architecture, the LOAD and STORE instructions are restricted to data transfer between memory and processor registers only. And all the other instructions should only use the processor register (memory is not involved)**.** There is no restriction on the use of address fields. Instructions for X=(A+B)*(C+D) is:

LOAD R1, A	R1←M[A]
LOAD R2, B	R2←M[B]
LOAD R3, A	R3←M[C]
LOAD R4, B	R4←M[D]
ADD R1, R1, R2	R1← R1 + R2
ADD R3, R3, R4	R3← R3 + R4
MUL R1, R1, R3	R1← R1*R2
STORE X, R1	M[X]←R1

4b. Explain following addressing modes with a numerical example

i. Direct Addressing Mode
ii. Register Indirect Addressing Mode
iii. Relative addressing mode
iv. Indirect Addressing Mode

Ans: Direct Addressing Mode:

In this addressing mode, the effective address is the address filed of the instruction. The operand resides in the memory and the effective address is given in the instruction.

Register Indirect Addressing Mode

In this addressing mode, the content present in the register gives the address of the operand i.e., the register holds the effective address, not the operand. Before using the register indirect addressing mode, the programmer needs to load the effective address into the processor register in the previous instruction.

Relative addressing mode

In this addressing mode, the effective address is the sum of content of program counter and the address part of the instruction

effective address = content of program counter + address part of instruction

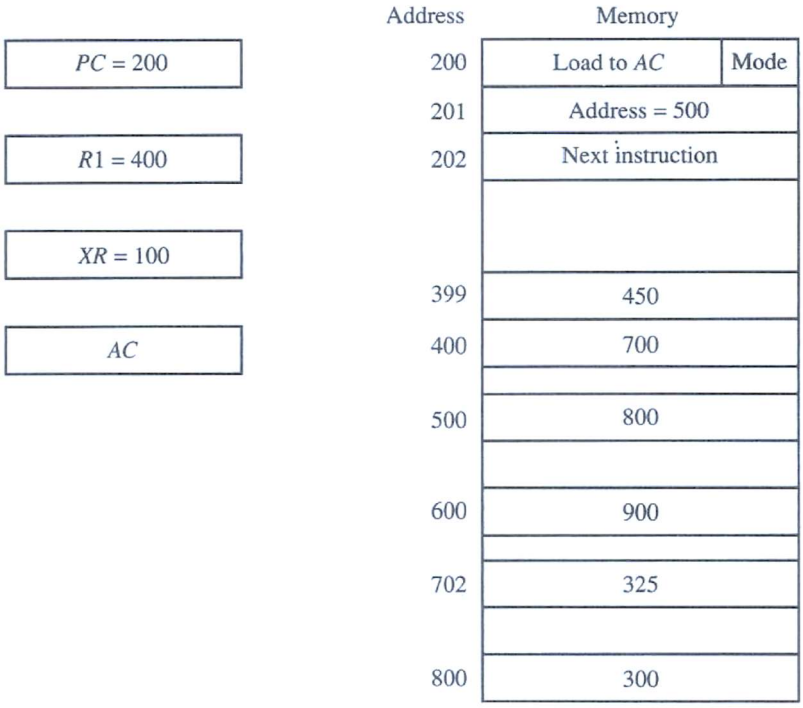
Indirect Addressing Mode

In this addressing mode,

effective address = data present in the address mentioned in instruction

the register used in the computation can be a program counter, indexed register or a base register

Numerical Example:



Addressing Mode	Effective Address	Data Loaded to AC
Direct Memory	500	800
Indirect Memory	800	300
Relative	202+500 = 702	325
Register Indirect	400	700

4c. Explain the Data Transfer instructions in detail

Ans.

Data transfer instructions move data from one place in the computer to another without changing the data content. The most common transfers are between memory and processor registers, between processor registers and input or output, and between the processor registers themselves. The table below shows the list of 8 data transfer instructions.

Name	Mnemonic	Comment
Load	LD	Load the AC with data from memory
Store	ST	Store the memory with data from AC
Move	MOV	Moves the data to/from memory/register
Exchange	XCH	Exchanges content of
Input	IN	Inputs the data from input device
Output	OUT	Outputs the data to output device
Push	PUSH	Inserts data into the stack
Pop	POP	Delete the data from the stack.