MAR	Holds address of memory unit
PC	Program Counter
IR	Instruction Register
R_1	Processor Register

Figure 1.1: Block diagram of register

Figure 1.2: Transfer from R1 to R2 when P = 1

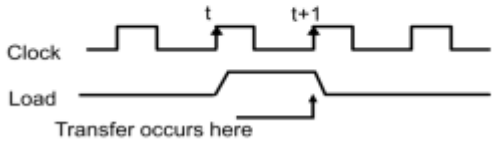


Figure 1.3: Timing diagram

Symbol	Description	Examples
Letters (and numerals)	Denotes a register	MAR, R2
Parentheses ( )	Denotes a part of a register	R 2(0-7), R2(L)
Arrow ←	Denotes transfer of information	R2←R1
Comma ,	Separates two micro operations	R2←R1, R1←R2

Table 1.1: Basic Symbols for Register Transfers

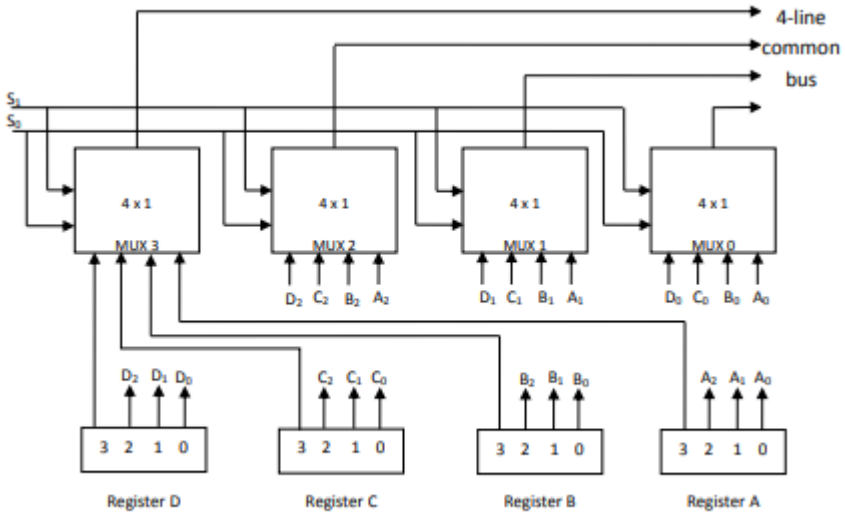


Figure 1.4: Bus system for four registers

S_1	S_0	Register selected
0	0	A
0	1	B
1	0	C
1	1	D

Table 1.2: Function Table for Bus

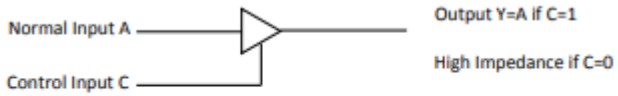


Figure 1.5: Graphic symbols for three-state buffer

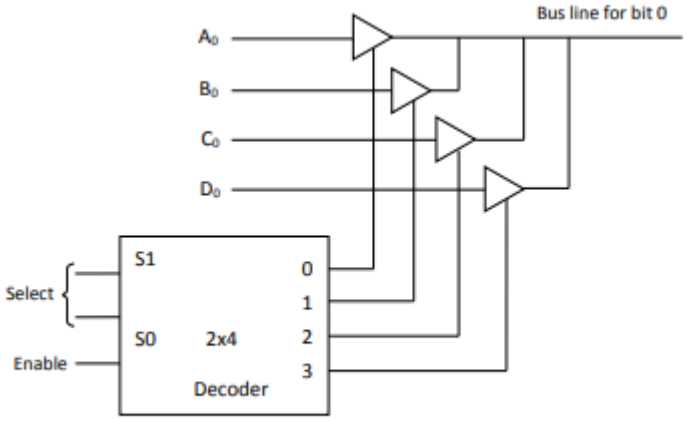
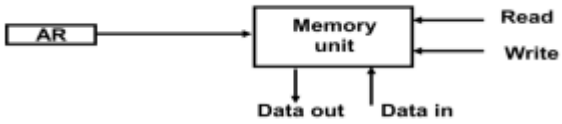


Figure 1.6: Bus line with three state-buffers



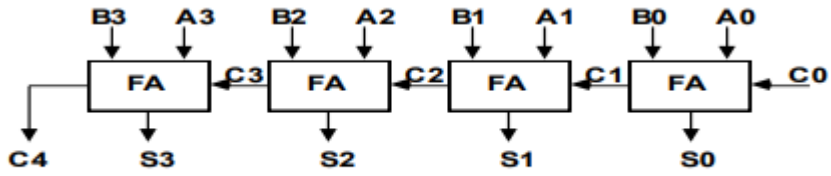


Figure 1.7: 4-bit binary adder

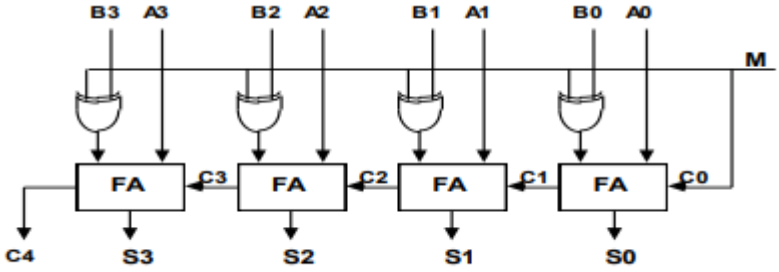


Figure 1.8: 4-bit Adder-Subtractor

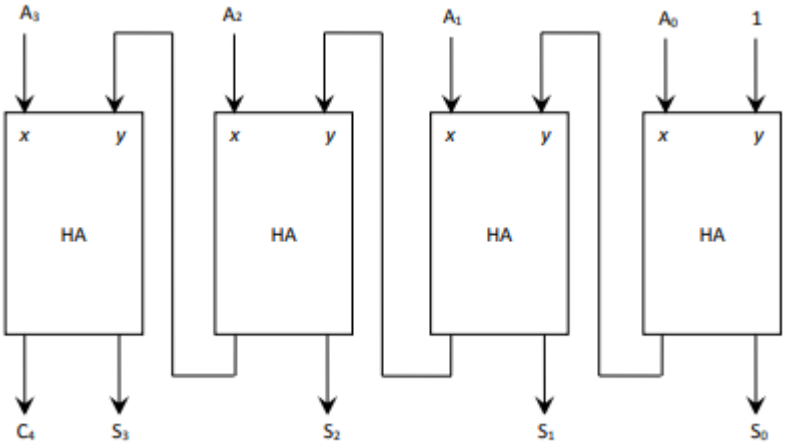


Figure 1.9: 4-bit binary incrementer

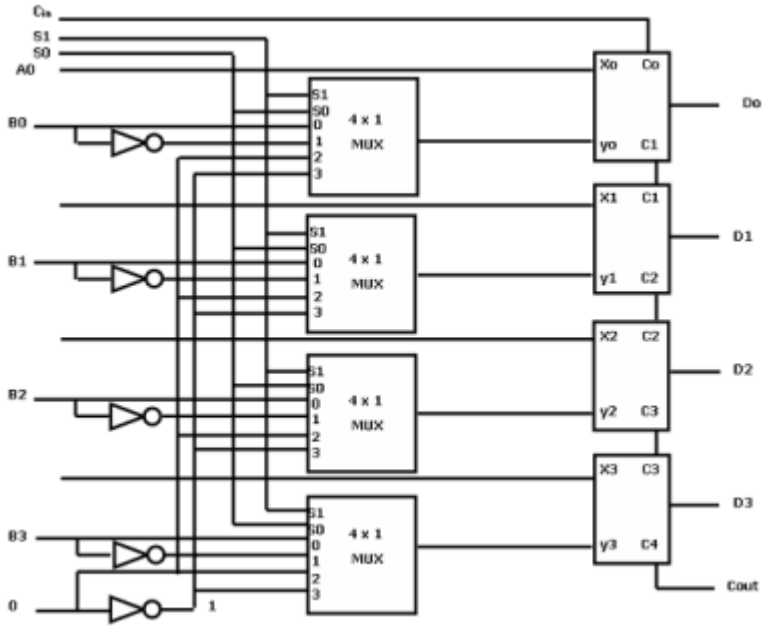


Figure 1.10: 4-bit arithmetic circuit

Select S_1 S_0 C_{in}			Input Y	Output D = A + Y + Cin	Microoperation
0	0	0	B	D = A + B	Add
0	0	1	B	D = A + B + 1	Add with Carry
0	1	0	B'	D = A + B'	Subtract with Borrow
0	1	1	B'	D = A + B' + 1	Subtract
1	0	0	0	D = A	Transfer A
1	0	1	0	D = A + 1	Increment A
1	1	0	1	D = A – 1	Decrement A
1	1	1	1	D = A	Transfer A

TABLE 1.3: 4-4 Arithmetic Circuit Function Table

x	Y	F_0	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8	F_9	F_{10}	F_{11}	F_{12}	F_{13}	F_{14}	F_{15}
0	**0**	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	**1**	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	**0**	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	**1**	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

TABLE 1.4: Truth Tables for 16 Functions of Two Variables

Boolean function	Microoperation	Name
$F_0 = 0$	F←0	Clear
$F_1 = xy$	F←A ∧ B	AND
$F_2 = xy'$	F←A ∧ $\overline{B}$	
$F_3 = x$	F←A	Transfer A
$F_4 = x'y$	F←$\overline{A}$ ∧ B	
$F_5 = y$	F←B	Transfer B
$F_6 = x \oplus y$	F←A⊕B	Exclusive-OR
$F_7 = x + y$	F←A∨B	OR
$F_8 = (x+ y)'$	F←$\overline{A \vee B}$	NOR
$f_9 = (x \oplus y)'$	F←$\overline{A \oplus B}$	Exclusive-NOR
$F_{10} = y'$	F←$\overline{B}$	Complement B
$F_{11} = x + y'$	F←A ∨ $\overline{B}$	
$F_{12} = x'$	F←$\overline{A}$	Complement A
$F_{13} = x' + y$	F←$\overline{A}$ ∨ B	
$F_{14} = (xy)'$	F←$\overline{A \wedge B}$	NAND

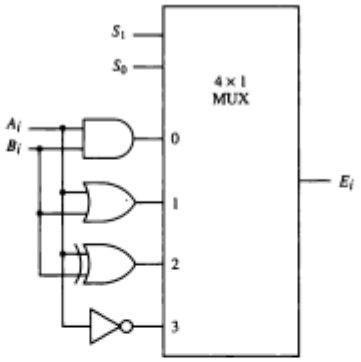
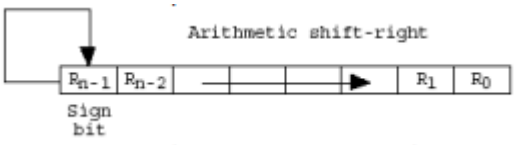


Figure 1.11 : One stage of logic circuit

S_1	S_0	Output	Operation
0	0	E = A ∧ B	AND
0	1	E = A ∨ B	OR
1	0	E = A ⊕ B	XOR
1	1	E = $\overline{A}$	Compliment

Table 1.6: Function table



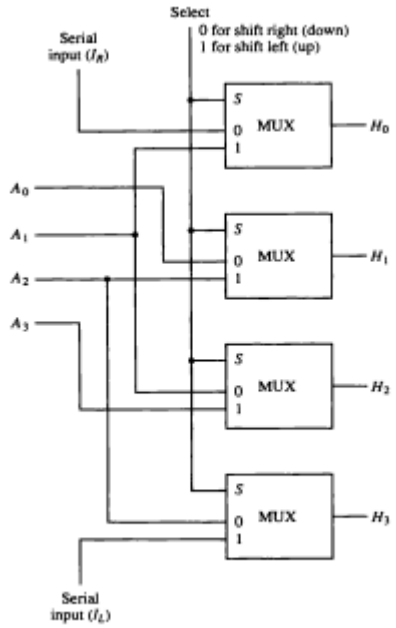


Figure 1.12: 4-bit combinational circuit shifter

Select	Output			
S	H_0	H_1	H_2	H_3
0	I_R	A_0	A_1	A_2
1	A_1	A_2	A_3	I_L

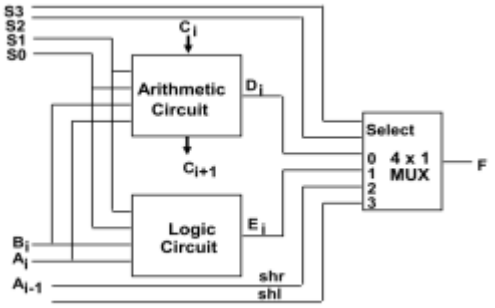


Figure 1.13: One stage of arithmetic logic shift unit

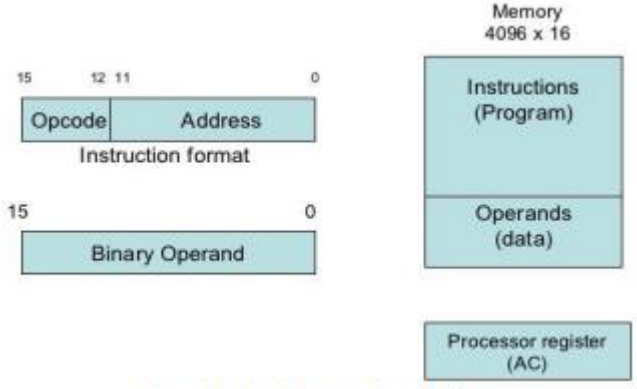


Figure 2.1: Stored Program Organization

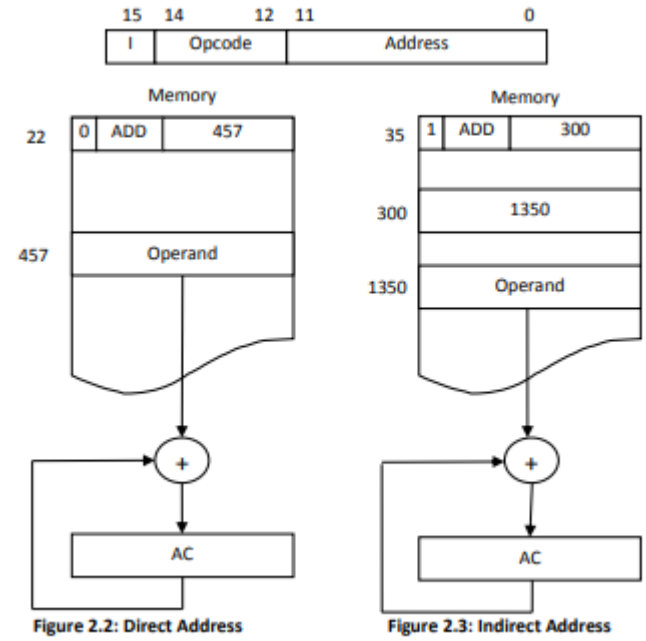


Figure 2.2: Direct Address

Figure 2.3: Indirect Address

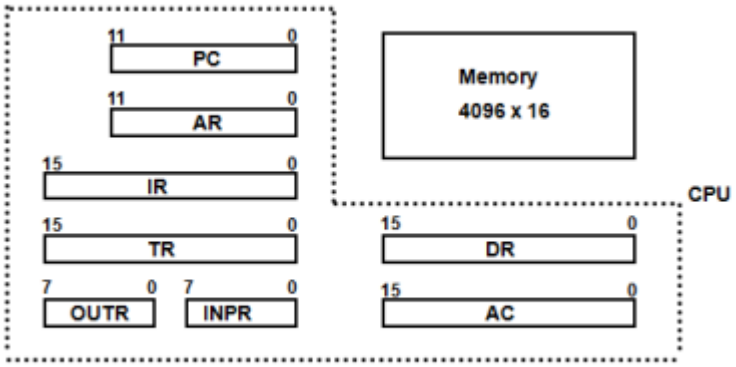


Figure 2.4: Basic Computer Register and Memory

Register Symbol	Bits	Register Name	Function
DR	16	Data register	Holds memory operand
AR	12	Address register	Holds address for memory
AC	16	Accumulator	Processor register
IR	16	Instruction register	Holds instruction code
PC	12	Program counter	Holds address of instruction
TR	16	Temporary register	Holds temporary data
INPR	8	Input register	Holds input character
OUTR	8	Output register	Holds output character

Table 2.1: List of Registers for Basic Computer

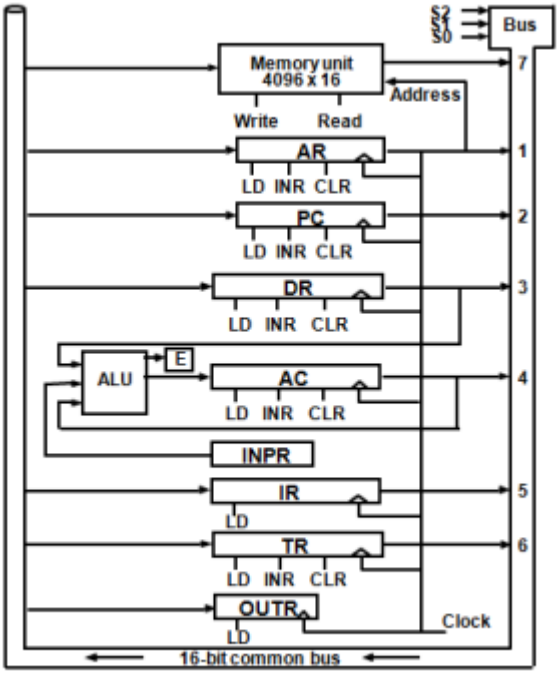


Figure 2.5: Basic computer registers connected to a common bus

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

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

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

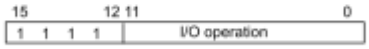


Figure 2.6: Basic computer instruction format

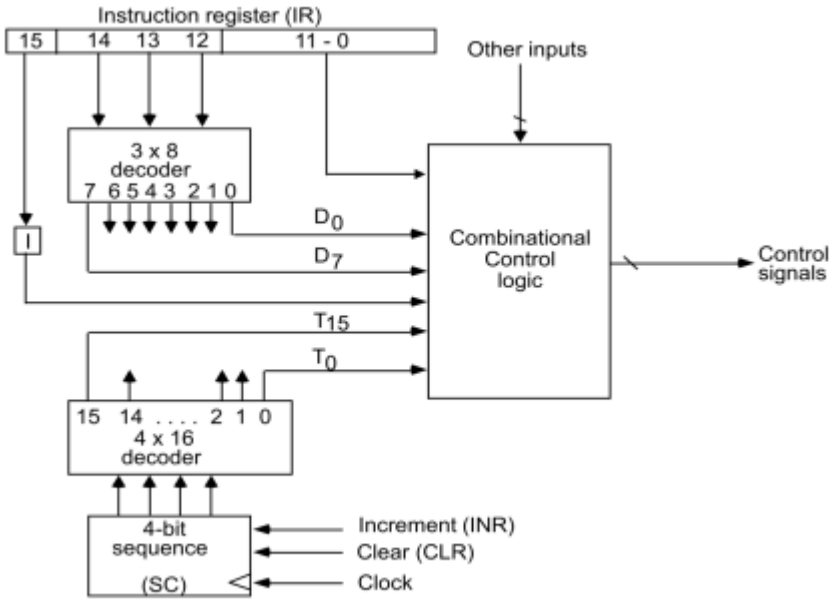


Figure 2.7: Control unit of basic computer

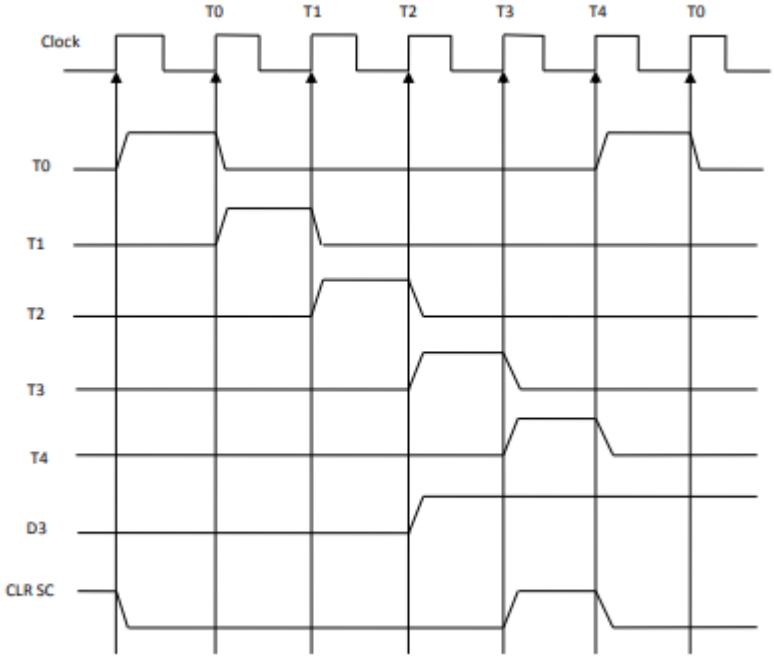


Figure 2.8: Example of control timing signals

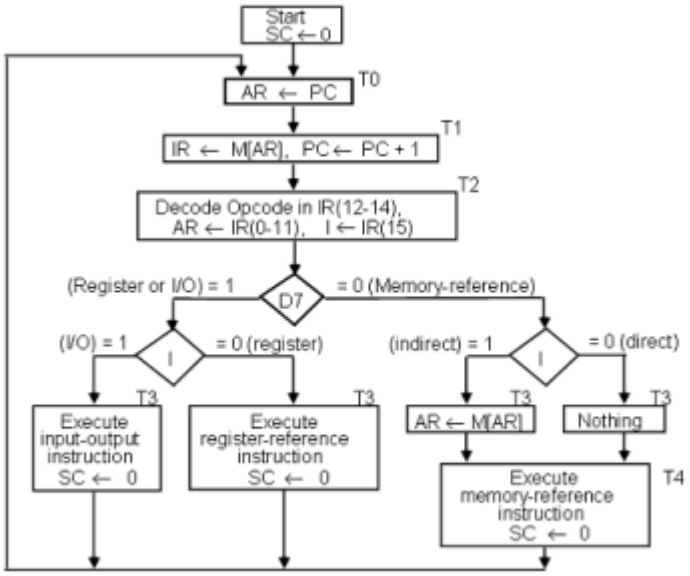
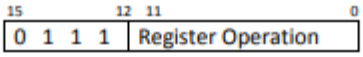
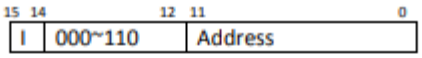


Figure 2.9: Flowchart for instruction cycle (initial configuration)



- There are 12 register-reference instructions listed below:

	r:	SC←0	Clear SC
CLA	rB_{11}:	AC ← 0	Clear AC
CLE	rB_{10}:	E ← 0	Clear E
CMA	rB_9:	AC ← AC'	Complement AC
CME	rB_8:	E ← E'	Complement E
CIR	rB_7:	AC ← shr AC, AC(15) ← E, E ← AC(0)	Circular Right
CIL	rB_6:	AC ← shl AC, AC(0) ← E, E ← AC(15)	Circular Left
INC	rB_5:	AC ← AC + 1	Increment AC
SPA	rB_4:	if (AC(15) = 0) then (PC ← PC+1)	Skip if positive
SNA	rB_3:	if (AC(15) = 1) then (PC ← PC+1	Skip if negative
SZA	rB_2:	if (AC = 0) then (PC ← PC+1)	Skip if AC is zero
SZE	rB_1:	if (E = 0) then (PC ← PC+1)	Skip if E is zero
HLT	rB_0:	S ← 0 (S is a start-stop flip-flop)	Halt computer



- The following table lists seven memory-reference instructions.

Symbol	Operation Decoder	Symbolic Description
AND	D_0	AC ← AC ∧ M[AR]
ADD	D_1	AC ← AC + M[AR], E ← C_{out}
LDA	D_2	AC ← M[AR]
STA	D_3	M[AR] ← AC
BUN	D_4	PC ← AR
BSA	D_5	M[AR] ← PC, PC ← AR + 1
ISZ	D_6	M[AR] ← M[AR] + 1, if M[AR] + 1 = 0 then PC ← PC+1

M[AR] ← PC, PC ← AR + 1
M[135] ←21, PC ←135 + 1 = 136

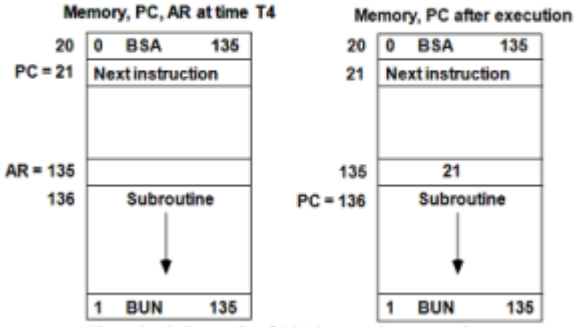


Figure2.10: Example of BSA instruction execution

D_6T_4: DR ← M[AR]

D_6T_5: DR ← DR + 1

D_6T_4: M[AR] ← DR, if (DR = 0) then (PC ← PC + 1), SC ← 0

Control Flowchart

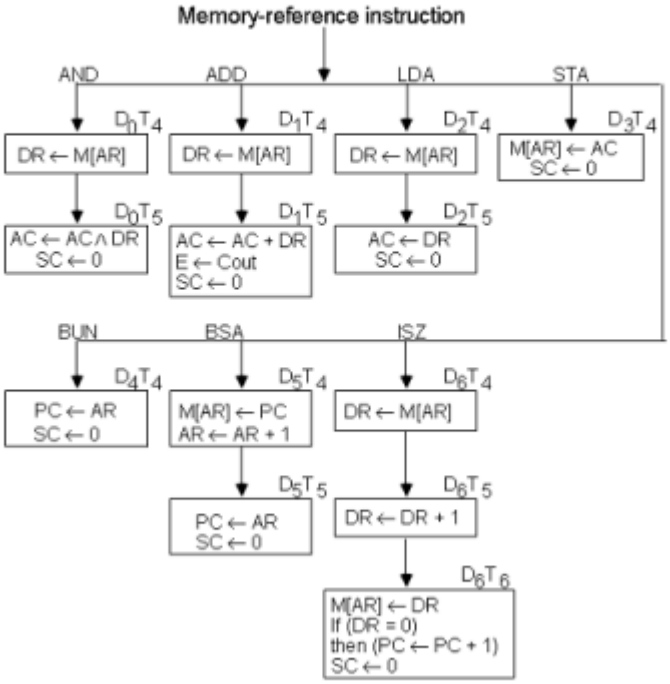


Figure 2.11: Flowchart for memory-reference instructions

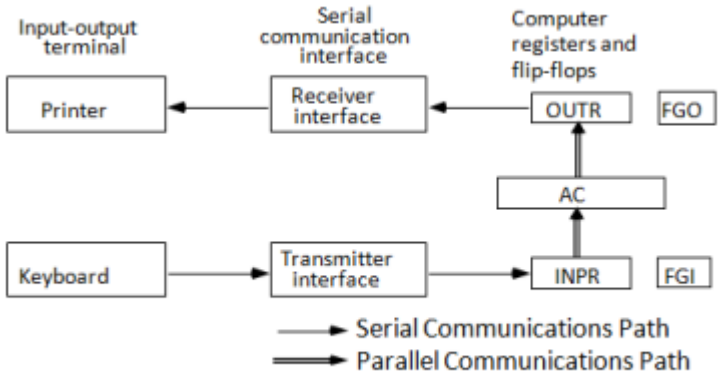


Figure 2.12: Input-output configuration

INP	AC(0-7) ← INPR, FGI ← 0	Input char. to AC
OUT	OUTR ← AC(0-7), FGO ← 0	Output char. from AC
SKI	if(FGI = 1) then (PC ← PC + 1)	Skip on input flag
SKO	if(FGO = 1) then (PC ← PC + 1)	Skip on output flag
ION	IEN ← 1	Interrupt enable on
IOF	IEN ← 0	Interrupt enable off

Table 2.2: Input Output Instructions

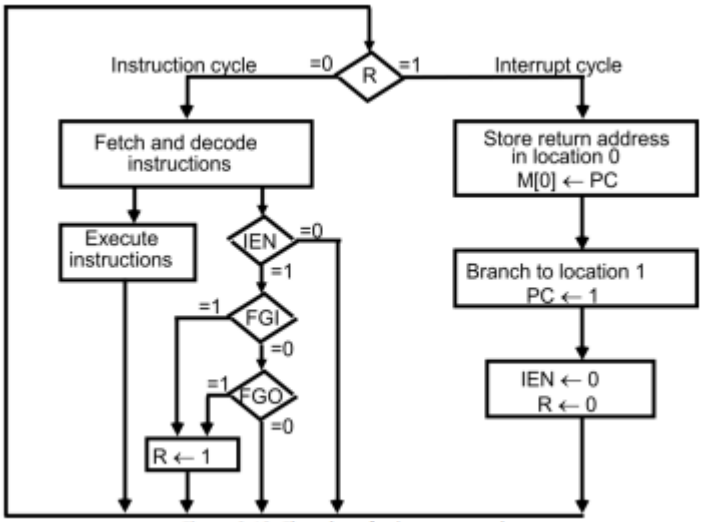


Figure 2.13: Flowchart for interrupt cycle

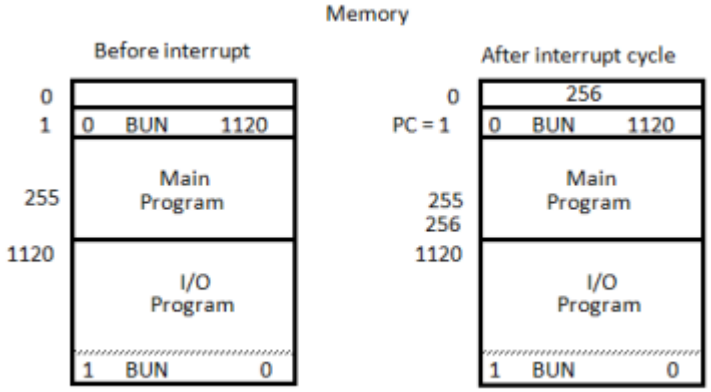


Figure 2.14: Demonstration of the interrupt cycle

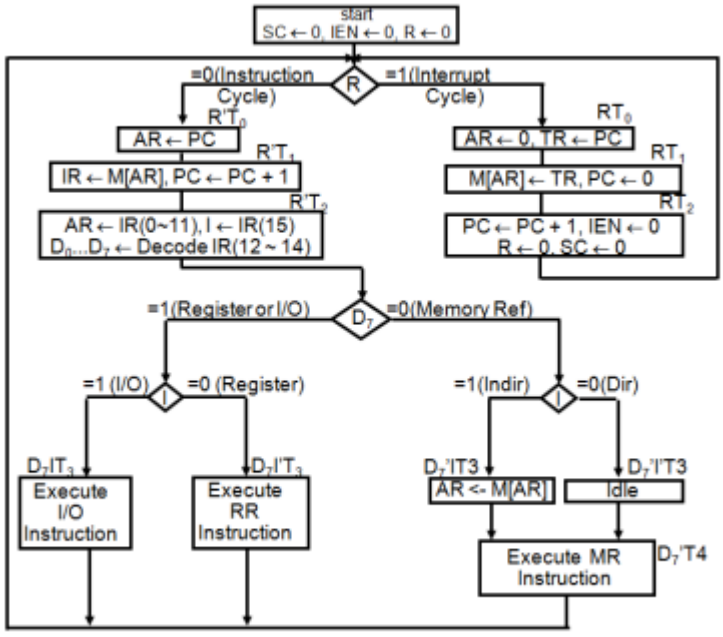


Figure 2.15: Flowchart for computer operation

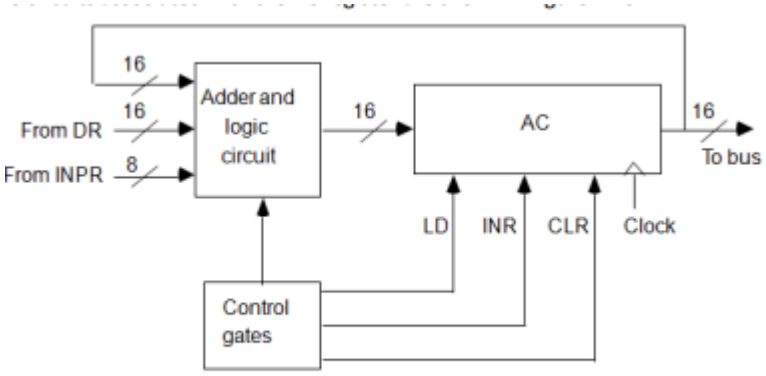


Figure 2.16: Circuits associated with AC

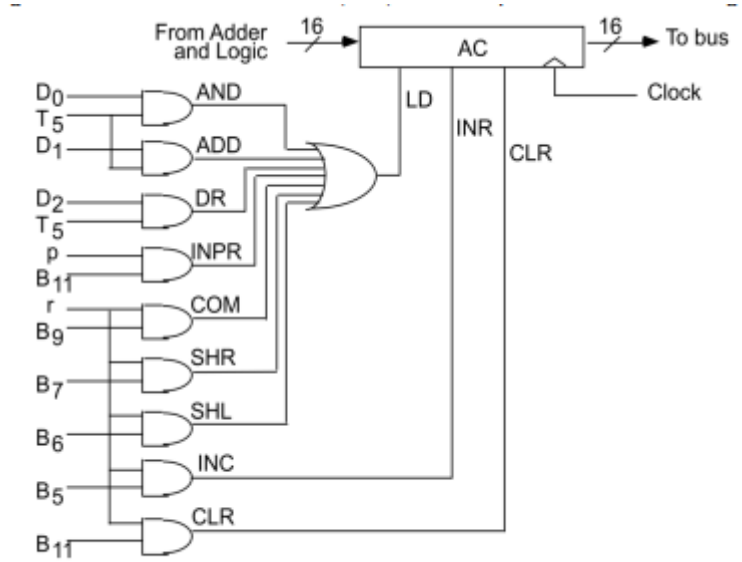


Figure 2.17: Gate structure for controlling the LD, INR, and CLR of AC

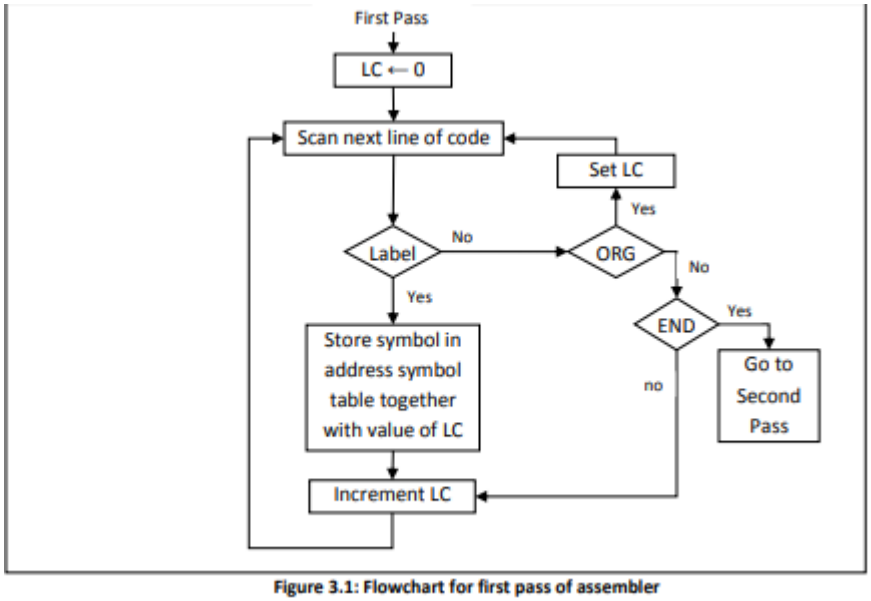


Figure 3.1: Flowchart for first pass of assembler

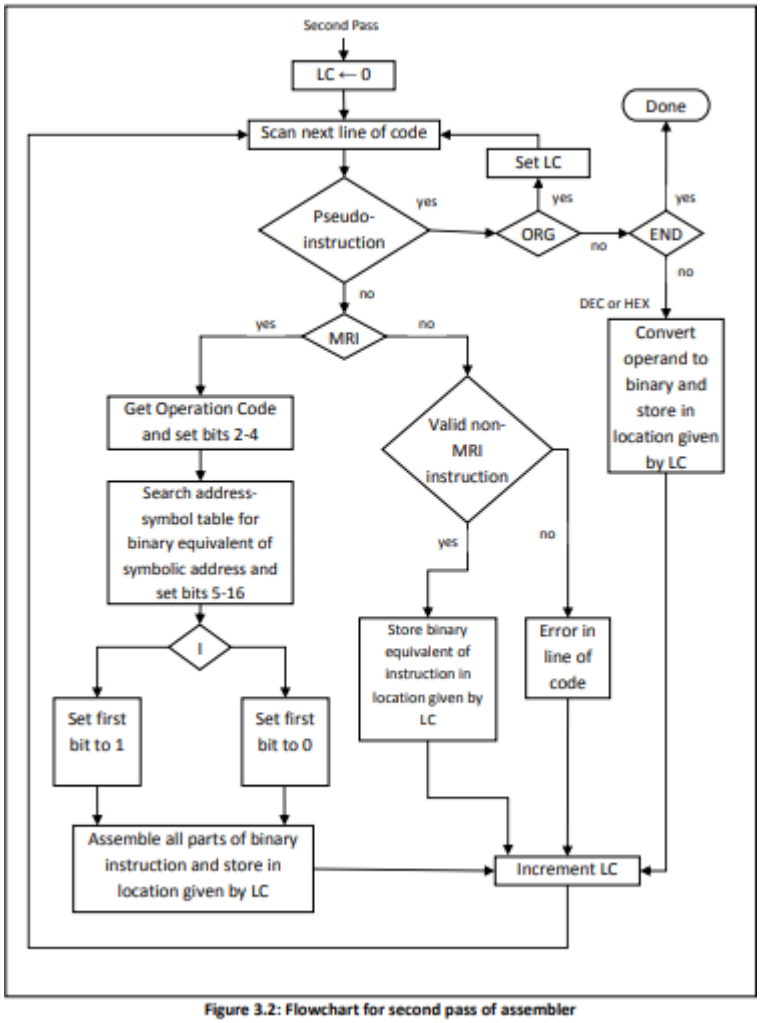


Figure 3.2: Flowchart for second pass of assembler

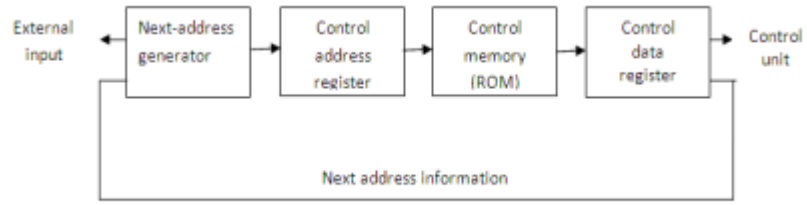


figure 4.1: Micro-programmed control organization

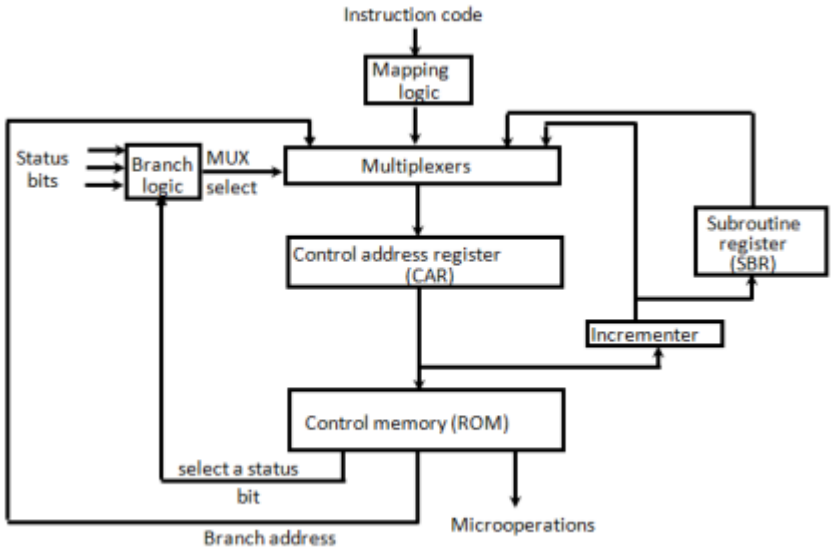


Figure 4.2: Selection of address for control memory

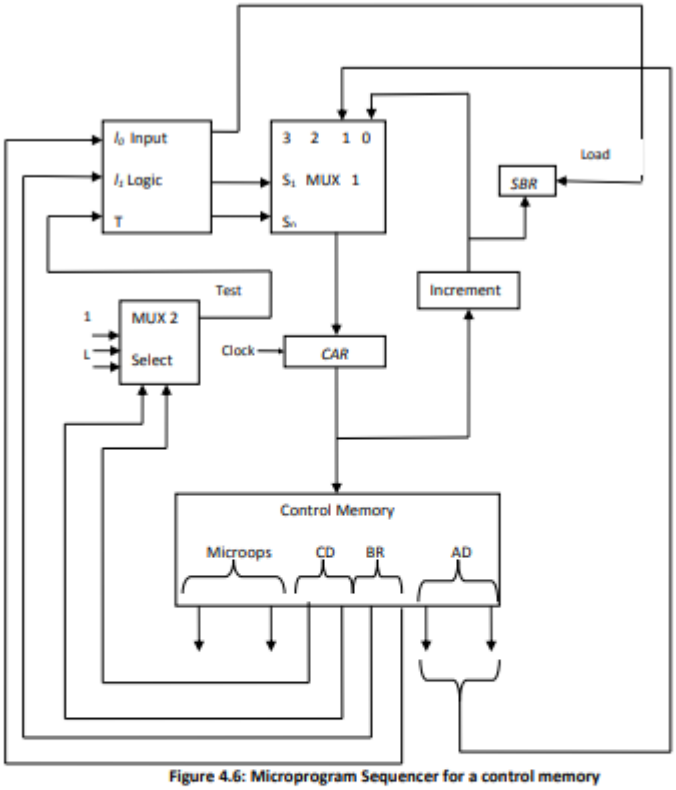


Figure 4.6: Microprogram Sequencer for a control memory

Input Logic : Truth Table

BR	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	X	1	0	0
1 1	1	1	X	1	1	0

Table 4.4: Input Logic Truth Table for Microprogram Sequencer

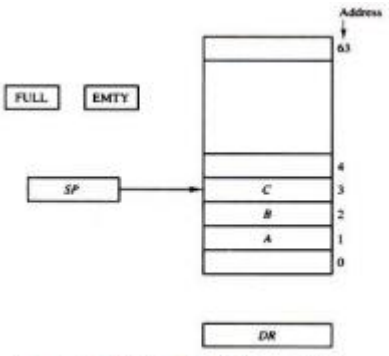


Figure 5.1: Block diagram of a 64-word stack

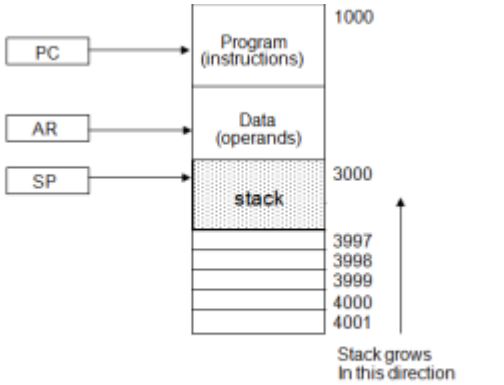


Figure 5.2: Computer memory with program, data, and stack segments

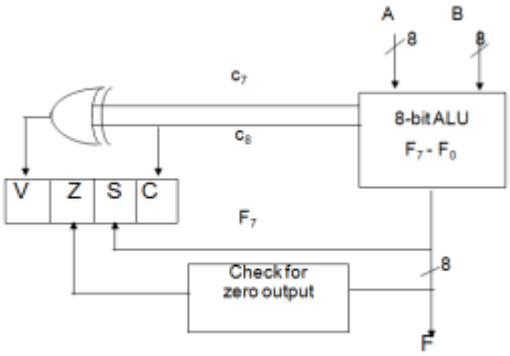
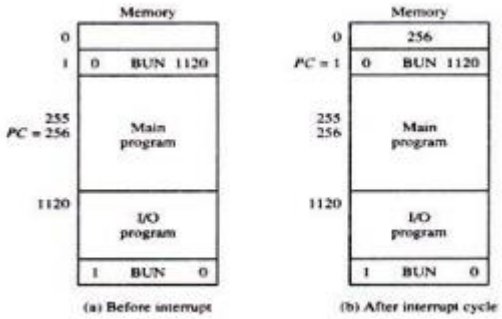


Figure 5.3: Status Register Bits



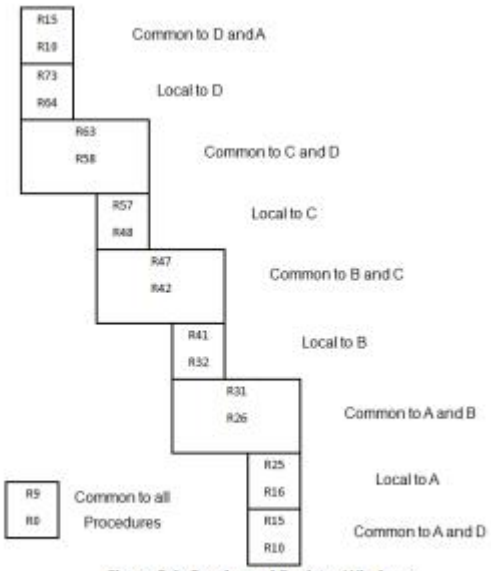
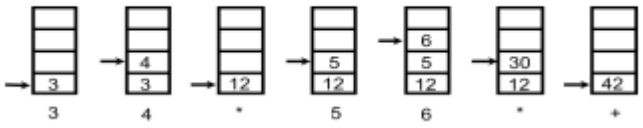


Figure 5.4: Overlapped Register Windows

$(3 * 4) + (5 * 6) \Rightarrow 3\ 4\ *\ 5\ 6\ *\ +$



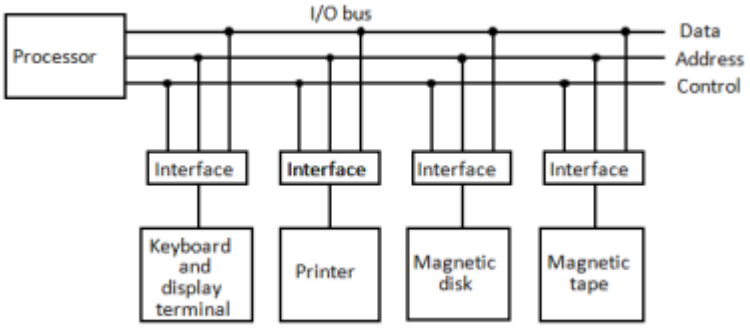
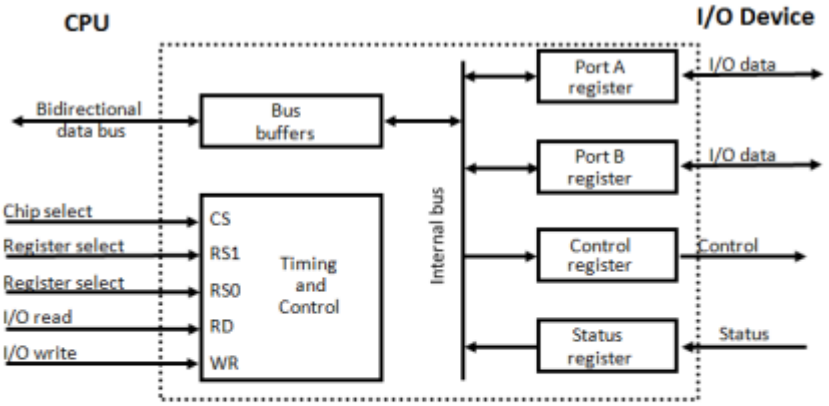


Figure 8.1: Connection of I/O bus to input-output device.



CS	RS1	RS0	Register Selected
0	X	X	None: data bus in high impedance
1	0	0	Port A register
1	0	1	Port B register
1	1	0	Control register
1	1	1	Status register

Figure 8.2: Example of I/O interface unit

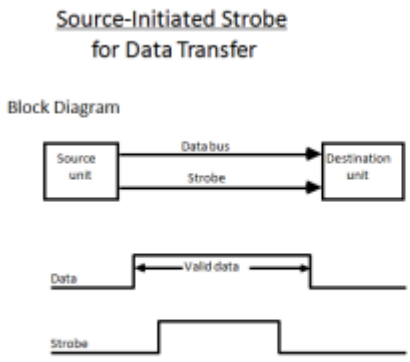


Figure 8.3: Source-initiated strobe for data transfer

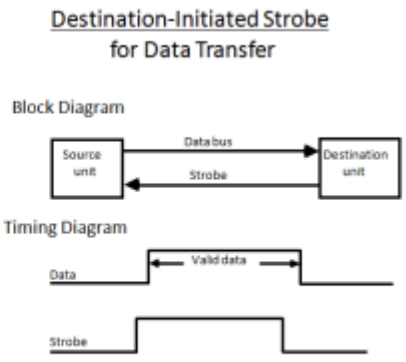


Figure 8.4: Destination-initiated strobe for data transfer

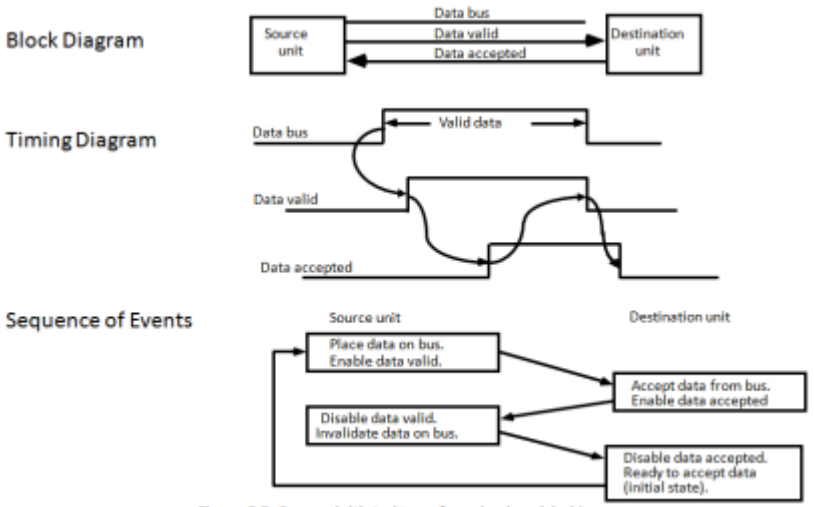


Figure 8.5: Source-initiated transfer using handshaking

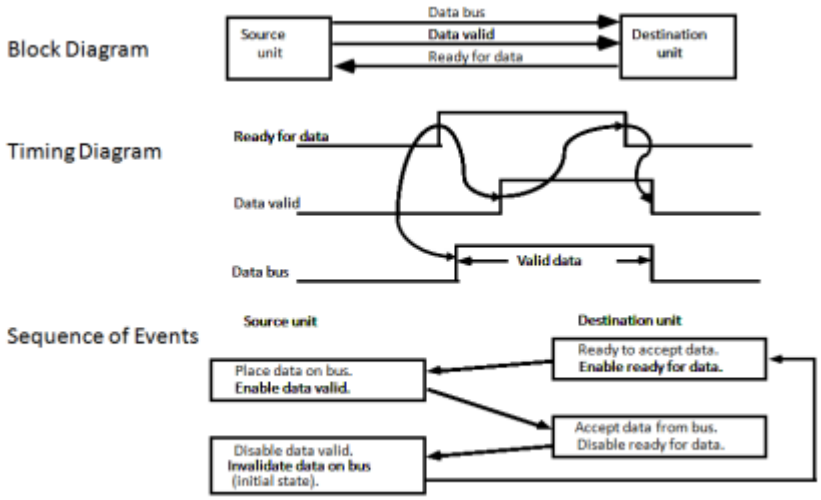


Figure 8.6: Destination-initiated transfer using handshaking

Programmed I/O:

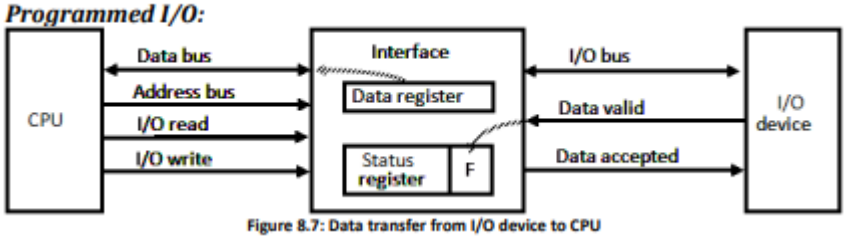


Figure 8.7: Data transfer from I/O device to CPU

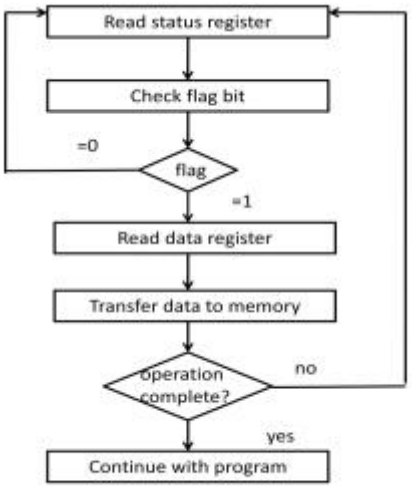


Figure 8.8: Flowchart for CPU program to input data

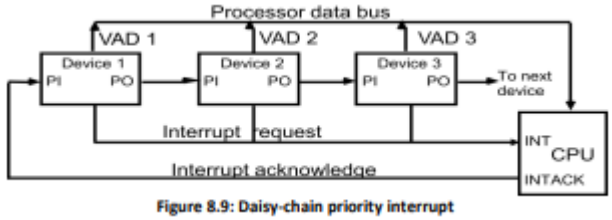


Figure 8.9: Daisy-chain priority interrupt

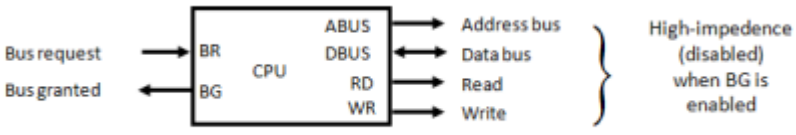


Figure 8.10: CPU bus signals for DMA transfer

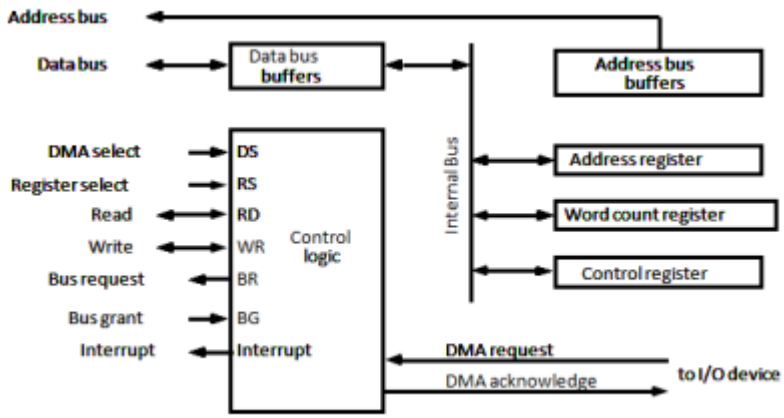


Figure 8.11: Block diagram of DMA controller

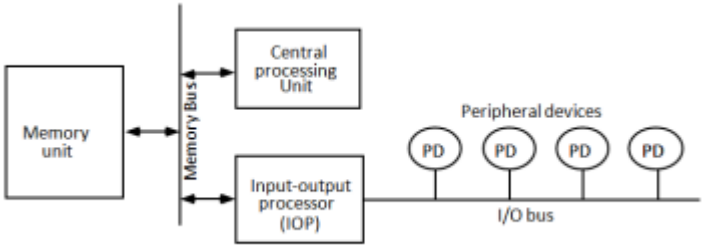


Figure 8.12: Block diagram of a computer with I/O processor

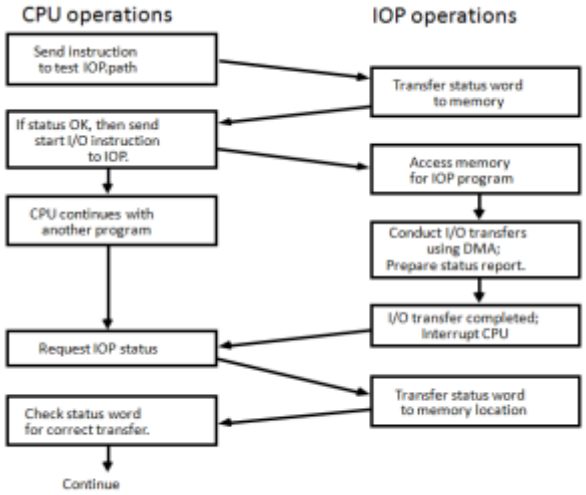


Figure 8.13: CPU-IOP communication

Memory address map of RAM and ROM

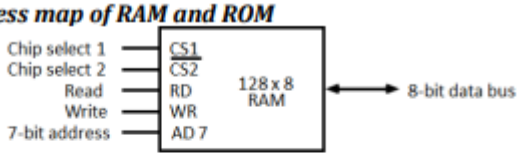


Figure 9.1: Typical RAM chip

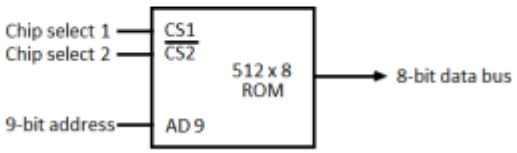


Figure 9.2: Typical ROM chip

Component	Hexa address	Address bus 10 9 8 7 6 5 4 3 2 1
RAM 1	0000 - 007F	0 0 0 x x x x x x x
RAM 2	0080 - 00FF	0 0 1 x x x x x x x
RAM 3	0100 - 017F	0 1 0 x x x x x x x
RAM 4	0180 - 01FF	0 1 1 x x x x x x x
ROM	0200 - 03FF	1 x x x x x x x x x

Table 9.1: Memory Address Map for Micro-procomputer

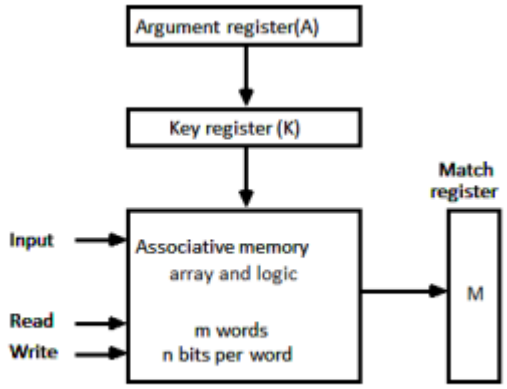
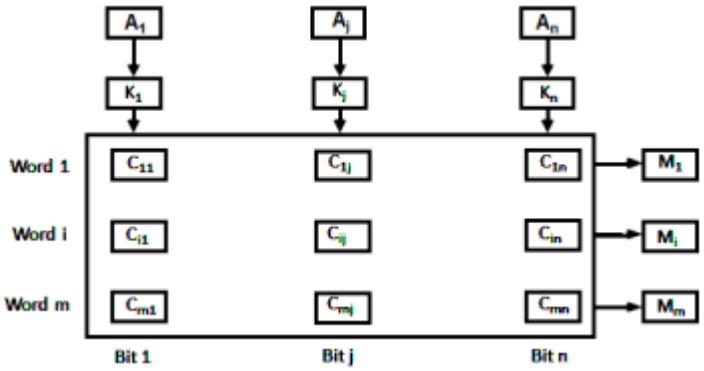


Figure 9.3: Block diagram of associative memory



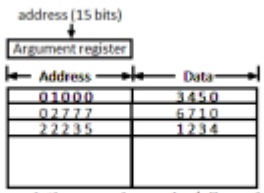


Figure 9.5: Associative mapping cache (all numbers in octal)

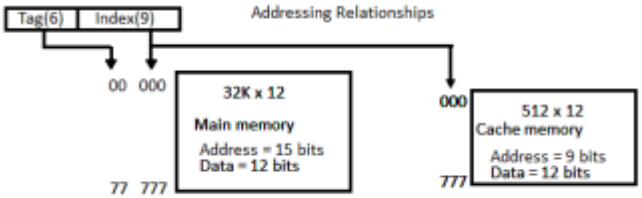


Figure 9.6: Addressing relationships between main and cache memories

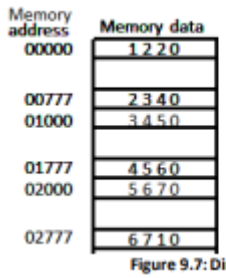
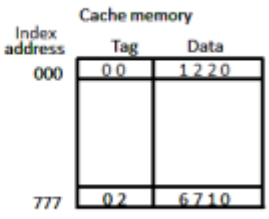


Figure 9.7: Direct mapping cache organization



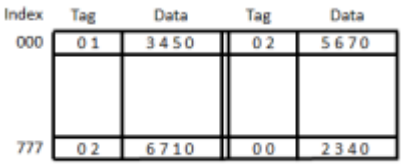


Figure 9.8: Two-way set-associative mapping cache

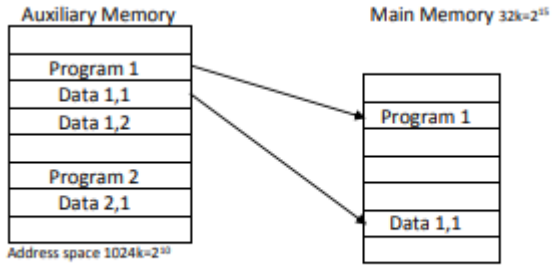


Figure 9.9: Relation between address and memory space in a virtual memory system

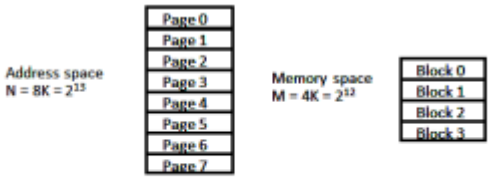


Figure 9.10 Address and Memory space split into group of 1K words

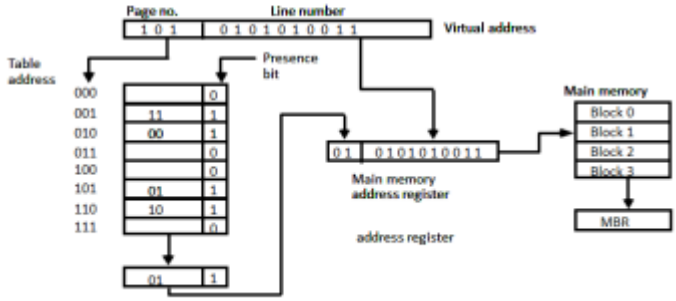


Figure 9.11: Memory table in paged system

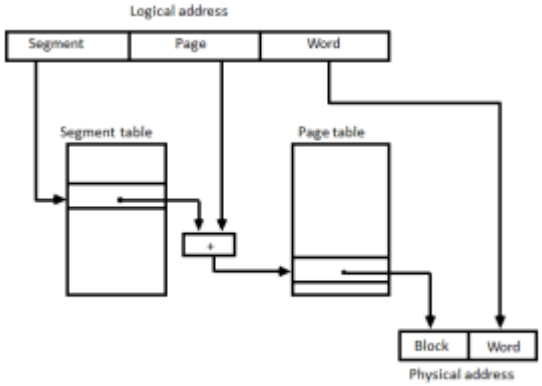


Figure 9.12: Logical to physical address mapping