System Architecture
frist class

④ $(x + x\bar{y} + xy)(x\bar{z})$

$(x + x\bar{y} + xy)(\bar{x} + z)$

$0 + 0 + 0 + xz + x\bar{y}z + xyz$

$xz + xyz$

$= xz$

⑪

$\bar{x}yz + \bar{x}y\bar{z} + xz \rightarrow$

$\Rightarrow \bar{x}y + xz$

x	y	z	(a)	(b)
0	0	0	0	0
0	0	1	0	0
0	1	0	1	1
0	1	1	1	1
1	0	0	0	0
1	0	1	1	1
1	1	0	0	0
1	1	1	1	1

Minterm ÷ which all variable appears exactly one (product term)

maxterm ÷ (sum term) .....

Table ÷

x	y	z	product term
0	0	0	$\bar{x}\bar{y}\bar{z}$
0	0	1	$\bar{x}\bar{y}z$
0	1	0	$\bar{x}y\bar{z}$
1			
		1	

symbol	m_0	m_1	m_2	...	m_7
m_0	1	0	0	...	0
m_1	0	1	0	...	0
m_2	0	0	1	...	0

x	y	z	sum term	symbol	M_0	M_1	M_2	...	M_7
0	0	0	$x+y+z$	M_0	0	1	1	...	1
0	0	1	$x+y+\bar{z}$	M_1	1	0	1	...	1

$F(x,y,z) = \Sigma m(0,2,5,7)$

$F(x,y,z) = \Pi M(1,3,4,6)$

Computer Basic → M.R

Morris mano Book

(+) Register transfer language :- Digital system design invariable uses a modular approach the modulars are constructed from such digital component, as resisters decoders, arithmatic elements and and central logic. various modules are interconnected with common data & central path. (this language express the symbolic form in microoperat" sequance among the registers of a digital module)

Register they contain and operation that are performed on data stored in them. the operations executed on data stored in registers. micro-operation is elemantry operation performe on information stored in one/more registers. (the sequance of m-operation performed on binary information stored in registers.)

m-operation are - shift, count, clear, & load.

Adopt a suitable symboles to describe the sequance of transfer beta registers various arithmetic and logic m-operator associated with transfer symbolic notation used to m-opration transfer among registers called register 'transfer language.

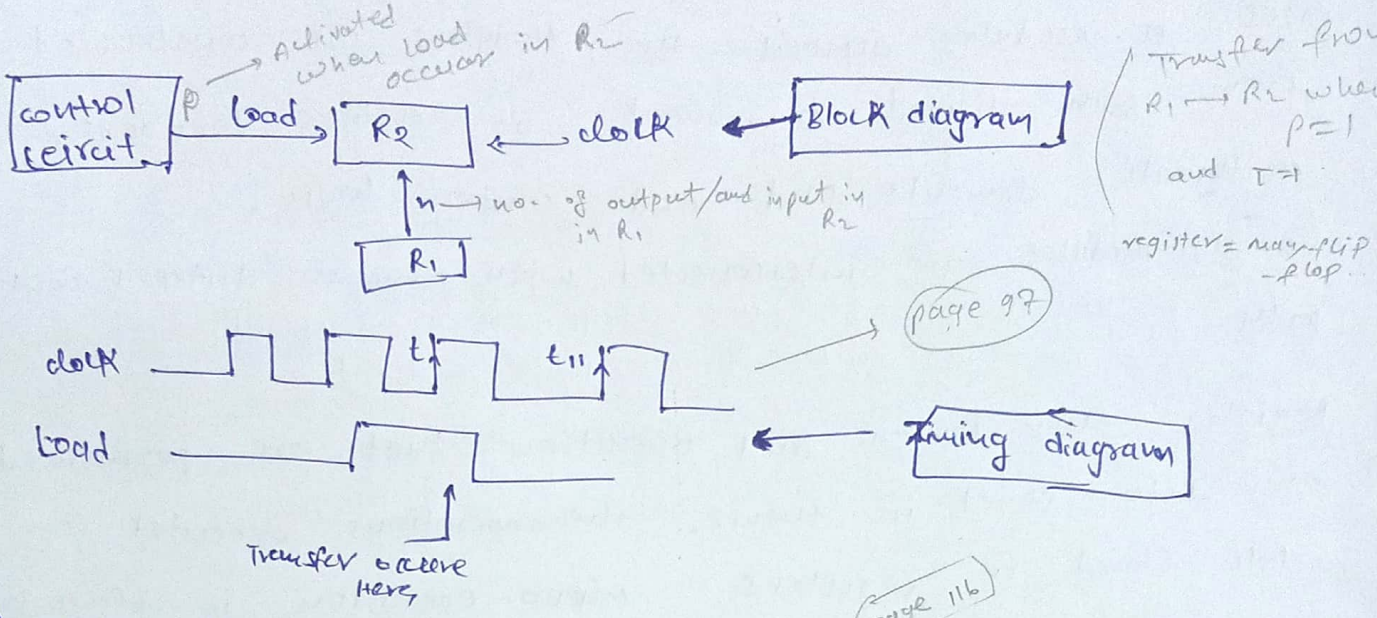
(+) Register Transfer :- it is a collection of data or information stored in a computer's memory which can be accessed quickly and used for various operat" the register that hold an adress.

for memory unit memory address register (MAR)

others are program counter (PC) Instruction register (IR), proceor Register. Information Transfer from one register to another by use symbolic form of replacement operation $R_2 \leftarrow R_1$ Represent information transfer

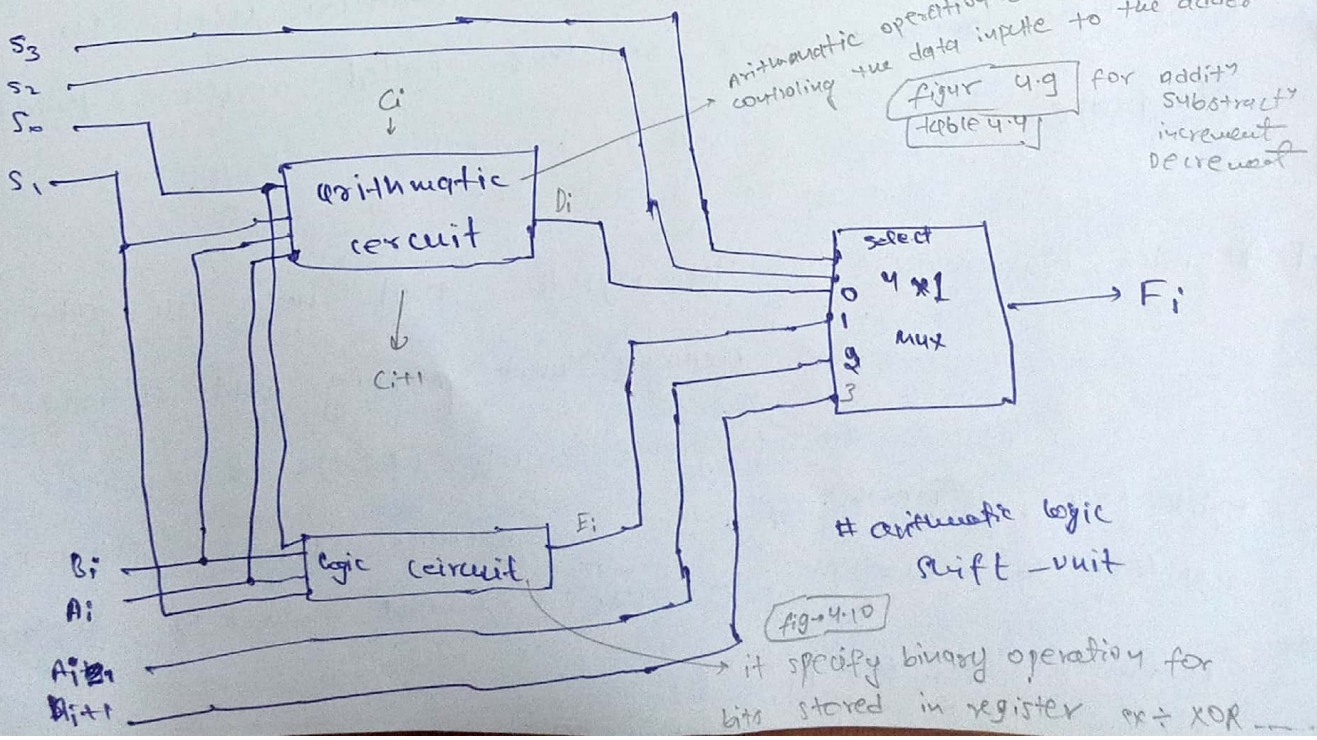
control variable from register transfer operation by control funct^{n}.

control funct^{n} a boolen variable is equal to either 1 or 0.



(+) Arithmetic logic shift unit :- page 116

computer system employ a ~~u~~ of storage register connected to a common operational unit, ALU to perform the ~~u~~ - operation, the contents of specified registers are placed in ALU perform an operation and transfer the results to specific register.



this diagram provide eight arithmatic operation four logic operat^{n}, two shift operat^{n} each operation selected with five variables S_3, S_2, S_1, S_0, c_i

table 4.7

Functional table :-

S_3	S_2	S_1	S_0	c_i	operation	
0	0	0	0	0	$F = A$	Transfer A
0	0	0	0	1	$F = A+B$	
0	0	0	1	0	$F = A \wedge B$	(logic AND)
0	0	0	1	1	$F = A \vee B$	
0	0	1	0	0	$F = A \oplus B$	
0	0	1	0	1	$F = \bar{A}$	
0	0	1	1	0	$F = \text{shr } A$	(shift right)
0	0	1	1	1	$F = \text{shl } A$	
0	1	0	0	x	$F = A - B - 1$	
0	1	0	0	x	$F = A - B$	
0	1	0	1	x	$F = A + 1$	
0	1	1	0	x	$F = A$	
1	0	x	x	x	$F = 0$	

Next chapter

Basic computer organisat^{n} and Design :-

Instruction code :

the operat^{n} of computer defined by → internal register, timing, & control structure, and set of instruction.

the internal organisation of a digital system is sequence of micro operation. it performs on data stored in its register

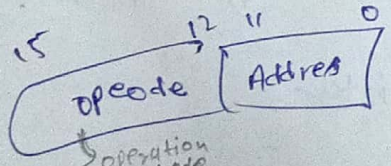
computer instruction is a binary code that specifye a sequnce of the operation. for the computer.

computer read the instruction from memory placed it in control register. the control then interprets the binary code of instruction and proceeds to ~~the~~ ~~execute~~ occur.

((An instruction code is a group of bits that instruct the computer to perform specific operation)). It ~~indicis~~ is divided into parts of the instruction code in operational part (operat^{n} code) ↓ it define addit^{n}, subst^{n}... consists of at least bits for 2^{n} operation (for n bit)

(ii) stored - prog. - organisation :-

Instruct^{n} code format have two part first part specifie the operat^{n} to be perform and second part address. Memory address tells the control where to find an ~~operation~~ operand in Memory. this operand is read from memory used as data to be operated on together with data stored in processor register.



memory unit 4096 word needs 12 bits to specify address $2^{12} = 4096$

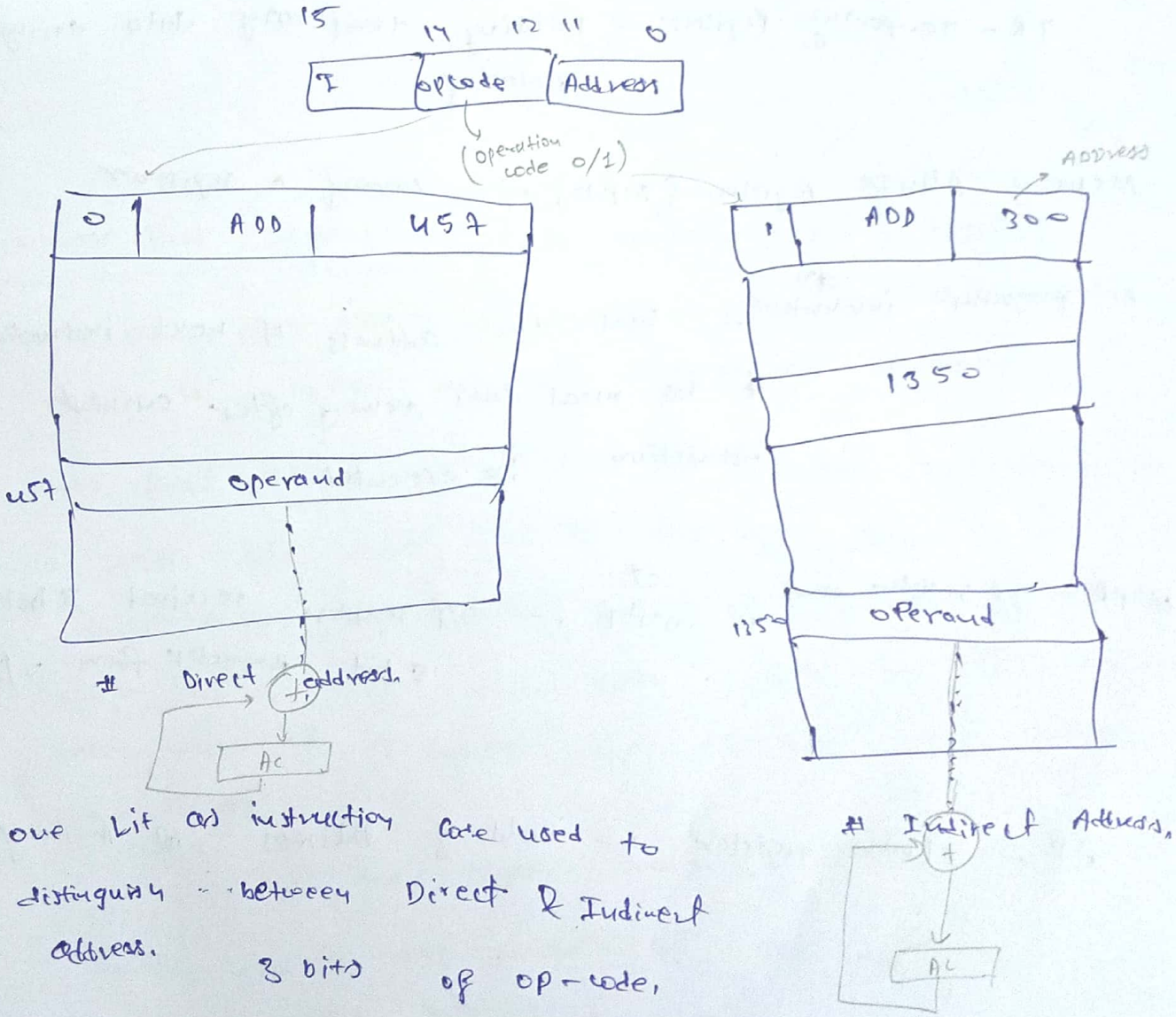
Instruction format

Instruct^{n} :- code is 16 bits have a bits operation code op code computer have single processor register. assign to it (accumulator)

(a register use to store intermediate result of arithmatic and logical operation)

Indirect Address :-

second part specifies the address of op operand the instruction is said direct address. In Indirect address bit in the second part of the instruction,



one bit in instruction code used to distinguish between Direct & Indirect address.

3 bits of op-code,

Computer Register :- computer instruction are stored in consecutive memory location and execute sequentially one at a time. control read the instruction from specific address in memory execute it. It then continue by reading next instruction, so on.

there are register are :-

1) DR - data register (16 bit) :→ hold operand read from memory

AC - Accumulator (16) - general purpose processing register

IR - instruction Register :- instruction read from memory is placed.

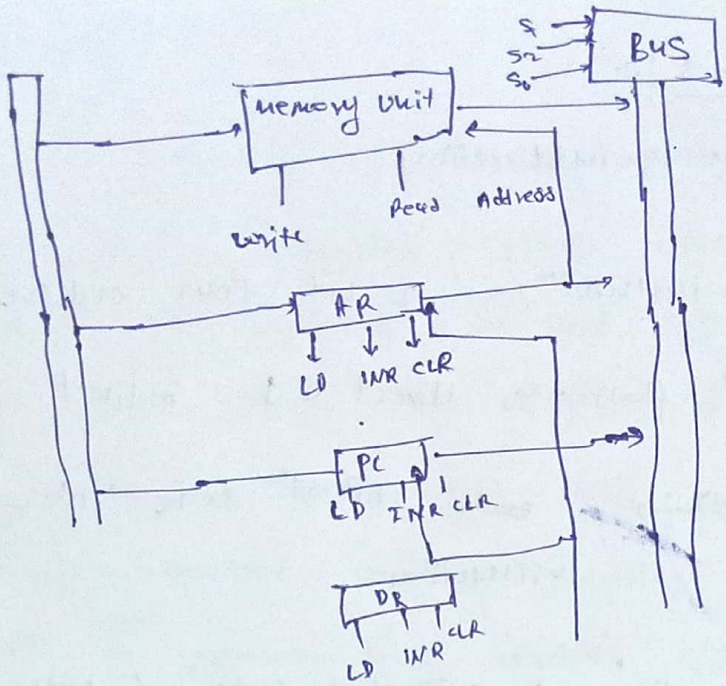
TR - temporary Register — holding temporary data during processing.

Memory Address Register (MAR) :- memory a register

* PC program (12) counter — (hold the address of next instruction) to be read from memory after current instruction is executed

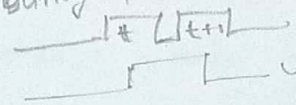
INPR — (IP) register and — OUTR (8) — O/P register received & hold 8 bit character from I/P divice

AR - Address register :- hold of Address of memory.



Basic computer registers connected to a common bus

particular register whose LD (load) inpute is enabled receive the data from bus during the next clock pulse



not clear

common bus system :- Basic computer has 8 register memory unit. path provide to transfer instruction from one register to another and b/w memory and register.

More efficient scheme transfering instruct to use common bus the specific o/p that selected for bus line determine by from binary value of selective variable S_2, S_1, S_0 o/p of DR in 3. when, $S_0 S_1 S_2 = 011$ five register have connected 3 control i/p LD - load INR - increment CLR - clear.

The clock transition at end of the cycle transfer the content of the bus into registered destination register;

① computer instructions :-

Basic computer has 3-instruction

(I) Memory register instruct^{n} :- 12 bits four address, one bit four - 0, direct 0 1, indirect

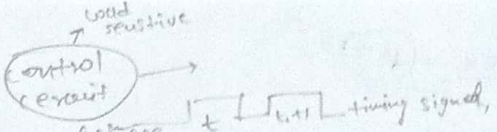
(II) Register-ref:- Tush^{n} → operat^{n} code '111' with '0' bit instruction

(III) I-p & O-p instruction → operation code '111' with '1' bit

Diagram

figure 5-5 Basic computer instruction formate.

⊕ Timing and control :-



Timing for all registers in basic computer is controlled by a master clock generator. clock pulses are applied to all flip-flop and registers in the system. the clock pulse do not change the state of a registers unless register inable by a control logic signal. control signal generator in control unit and provide control inputs for multiplexer in common bus control i/p in processor registers and ~~n~~ operation for accumulator,

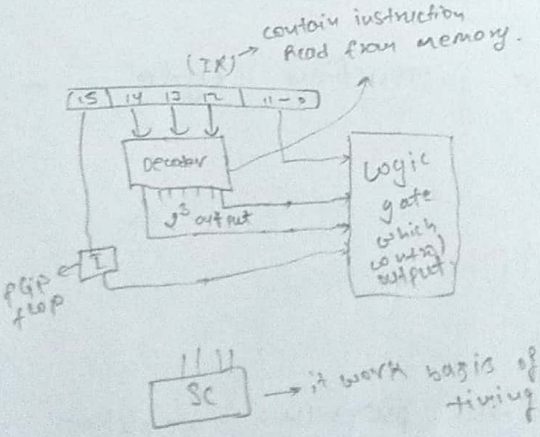
there are two major types of control organisation

① H/w control

② ~~n~~ programed control

(i) the control logic is impleted with gates, flip-flop decoder and other digital circuit.

② the control instruction stored in control memory.



figur 5-6

figur note 2 decoder a sequnce control and no. of control logic gate.

Instruct^n register divided into 3 parts. Decoder $D_0 - D_7$

explanation – (139) page

one clock cycle → activate To which Trigered its register
it happend till CLR inpute is active,
and then again start with T_5 ---- to T_{15} in
the basis of clock cycle and back to T_0 again

figur 5.9 (140)

Summary :- Sc instruct the decoder by the help of timing signal clock pulse and decoder instruct logic gate Hence in the Basis of signal gate CLR or Increment the inputs,

5-5 instruction cycle :-

See fig 5-7 for detail

(program executed in computer by going through cycle)

for each instruction cycle in turn sub-divided into sequence of subcycle. or phases.

1. Fetch an instruction from memory. — By instruct register
2. Decode the instruct — Decoder in binary
3. Read the effective address from memory. if the instruction has an indirect address. — Read By Sc and clock pulse and send it to logic gate then
4. Execute the instruction, — Output executed,

Figur 5-9 flowchart for instruction cycle → (144)

(141) page

(+) fetch & decode :— See (*) above as summary

Initially the program, computer pc is loaded with address of first instruction in the program sequence counter, (sc) is clear to 0 provided by decoded timing signal. T_0, after each clock pulse sc. is increment by one. so timing signal -

structure.

Address register

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)

(143)

(#) Determine the type of instruction:- During time T_3 the control unit determines the type of instruction ——— $D_7=0$ Memory ref. instruction.

(+) Register - Reference instructions:- Table 5-3

(147)

(+) 5.6 Memory-Reference instructions:- Table 5-4

AND to AC
ADD to AC
LDA → load to AC
STA → store AC
BUN
BSA
ISZ

(152)

(+) Input/output and interrupt :-

Input-output configuration :-

——— with the AC in parallel

the 1 bite input flag FGI is a control flip-flop

new instruct^n is available in i/p device clear to 0

when ——— FGI is cleard to zero

Page 155 Program interrupt :- 3rd line, last 4th line

figure 5.13 (156)

Multiple bus organisation of computer :-

next chapter 6

① program of Basic computer
② Machine language
③ Assembly language
④ Rule of the language
⑤ Assembler :-

page (189)

① second pass :- Machine instruction are translated during a second pass by means of table lookup procedure, the assembler use four table And symbol that. - —

① pseudoinstruction table :- Entries are ORG, END, DEC, HEX.

② MRI table :- contains 7 symbols of memory reference instruction.

③ non-MRI-Table- contains the symbol for the 16 register. ref. input-output reference instruction with 16 bit binary code equation.

④ Address symbol table :- it generate during first pass of assembly process.

(see flowchart) Fig 6.2 (I & II)

page 190 line 2 → LC is initialy set to '0' - - - to the last paragraph

2nd para- if the symbol - - - 3rd line

6-5 program loops :- is a seqn. of instruct" that executed many times.