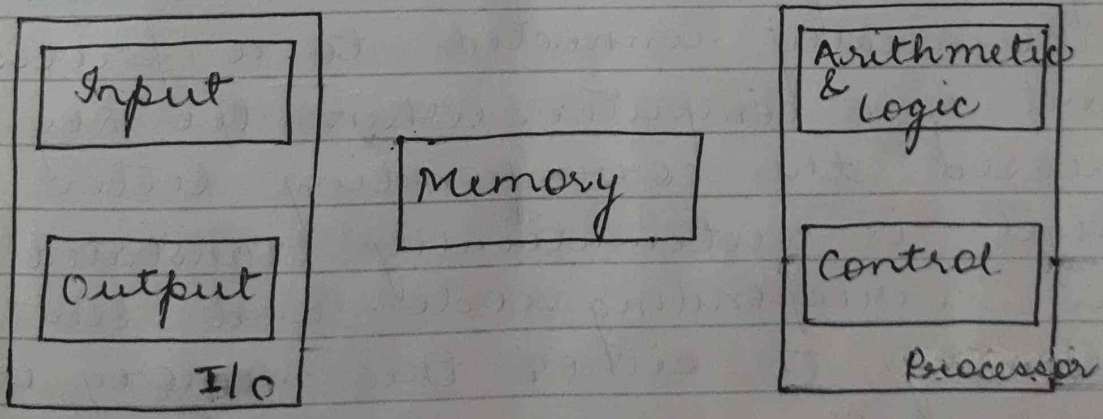
* **Digital Computer or Computer** - A fast electronic calculating machine that accepts digitized input information, process it according to a list of internally stored instruction and produces the resulting output information.

The list of instruction is called a computer program and the internal storage is called computer memory.

* **Functional Units** -

A computer consists of 5 functionally independent main parts -

1. Input
2. Memory
3. Arithmetic and Logic
4. Output
5. control



→ The input unit accepts coded informatio from human operators from electromechan -cal devices such as the keyboard of a vide terminal or from other computers over digital communication lines.

→ The information received is either stored in memory or immediately used by the arithmetic and logic circuitry to perform the operations.

→ The processing steps are determined by the program stored in the memory.

→ Finally, the results are sent back to the outside world through the output unit.

→ All of these actions are coordinated by the control unit.

① Input Unit – computers accepts the coded information through input units, which read the data. The most well known input device is the keyboard of vidéo terminal, which is electronically connected to a processing part of a computer. When the key is pressed the corresponding letter or digit is automatically translated into its corresponding code and sent directly to either the memory or processor.

e.g. Joysticks, Trackballs, & mice etc.

② Memory Unit – The function is to store program and data. There are 2 classes –

a) **Primary Storage** – Primary storage or main memory is the fast memory that operates at electronic speed. Programs are stored in main memory when they are being executed.

Main memory contains a large no. of semiconductor storage cells, each capable of storing one bit of information. These cells are processed in groups (read or write operation) of fixed size called words. The contents of one word (containing n bits) can be stored or retrieved in one basic operation.

To provide easy access to any word in memory a distinct address is associated with each word location. Number of bits in each word is called word length which is 16 to 64. A program must reside in main memory during execution. It is essential to be able to access any word location within the main memory as quickly as possible.

Random Access Memory – Memory in which any location can be ~~executed~~ reached in a short and fixed amount of time after specifying its address. The time required to access ~~as~~ one word is called memory access time. It typically ranges from 10 to 100 nanosecond for most modern computers.

b) Secondary Storage – Secondary storage is used when large amounts of data have to be stored, if some of the data need not be accessed frequently.

e.g. Magnetic Tape & Disks

③ Arithmetic & Logic Unit – Most computer operations are executed in ALU of processor.

e.g. Addition of 2 no's which are located in main memory. Addition is carried out in arithmetic unit.

Any other operation is initiated by bringing the required operands into the ALU, where the necessary operation is performed. Not all operands used in a computation reside in main memory.

Processor contain a no. of high speed storage elements called registers, which may be used for temporary storage of frequently used operands. Each register can store one word of data. Access time to registers are 5 to 10 times faster than access time to memory.

The control and ALU are usually many times faster than other devices connected to a computer system. This enables a single processor to control

a no. of external devices. This is possible because the vast difference in speed enables the fast processor to organize and control the activity of many slower devices.

④ **Output Function** - Its function is to send processed results to the outside world.
e.g. High Speed Printer - 10,000 lines/min.
Some units such as video terminals and graphics displays, provide both an O/P function and i/p function. The reason for using the single name of i/o unit.

⑤ **Control Unit** - The memory, ALU, input and output units store and process information and perform input and output operation. The operation of these units must be co-ordinated in some way, this is the task of control unit. It is the nerve ~~centre~~ center that sends control signal to other units and senses their data.

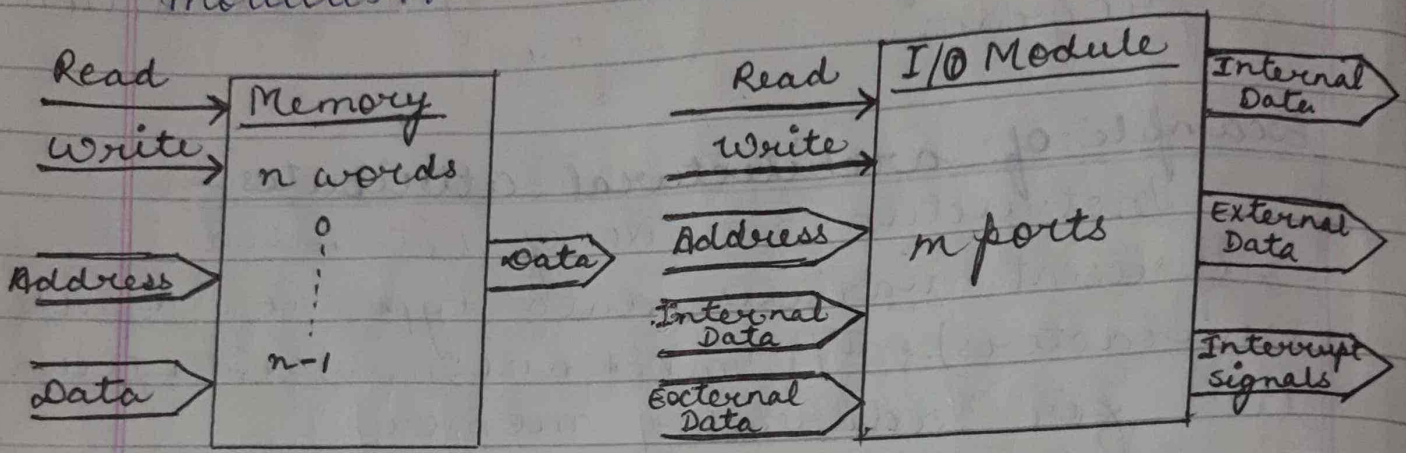
I/O transfer are controlled by software instruction and informati to be transferred. The actual timing signal that govern the transfer are govern by the control circuits.

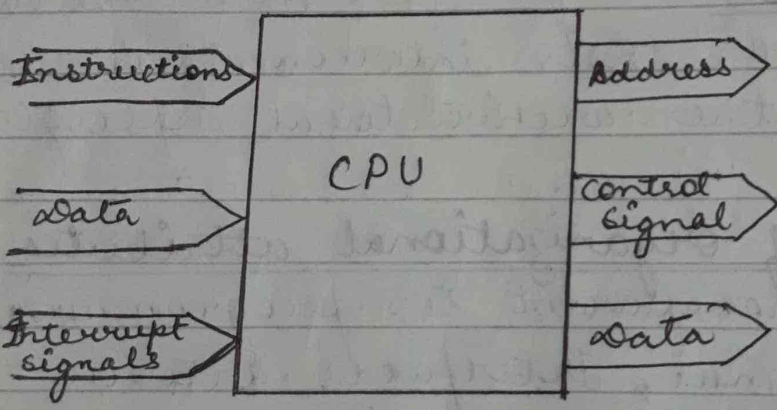
Data transfer between the processor and the memory are also controlled by the control unit

Functional Unit Interconnection (Interconnection Structures) A computer consists of a set of components or modules of 3 basic types (processor, memory, I/O) that communicate with each other. In effect, a computer is a network of basic modules. Thus, there must be path for connecting

the modules.

The collection of paths connecting the various modules is called the interconnection structure. The design of this structure will depend on the exchanges that must be made between modules.





Computer Modules

Memory - A memory module will consist of n words of equal length. Each word is assigned a unique numerical address (0, 1, - - -, n-1). A word of data can be read from or written into the memory. The nature of the operation is indicated by read or write control signal. The location of for the operation

is specified by an address

I/O Module - I/O is functionally similar to memory. There are 2 operations - read and write. An I/O module may control more than one external device. We can refer to each of the interfaces to an external device as a port and give each a unique address (0, ..., m-1). In addition, there are external data paths for the i/p and o/p of data with an external device. Finally, an I/O module may be able to send interrupt signals to the processor.

Processor - The processor reads in instructions and data, writes out data after processing and uses control signals to control the overall operation of the system. It also receives interrupt signals.

The interconnection structure may support the following type of transfers:

1) Memory to Processor - The processor reads an instruction or a unit of data from memory.

2) Processor to Memory - The processor writes a unit of data to memory.

3) I/O to processor - The processor reads data from an I/O device through I/O module.

is specified by an address.

I/O Module - I/O is functionally similar to memory. There are 2 operations - read and write. An I/O module may control more than one external device. We can refer to each of the interfaces to an external device as a port and give each a unique address (0, ..., m-1). In addition, there are external data paths for the i/p and o/p of data with an external device. Finally, an I/O module may be able to send interrupt signals to the processor.

Processor - The processor reads in instruction and data, writes out data after processing and uses control signals to control the overall operation of the system. It also receives interrupt signals.

The interconnection structure may support the following type of transfers:-

1) Memory to Processor - The processor reads an instruction or a unit of data from memory.

2) Processor to Memory - The processor writes a unit of data to memory.

3) I/O to processor - The processor reads data from an I/O device through I/O module.

4) Processor to I/O — The processor sends data to the I/O device.

5) I/O to or from Memory - For these 2 cases, I/O module is allowed to exchange data directly from memory, without going through the processor, using Direct Memory Access (DMA).

Bus Interconnection

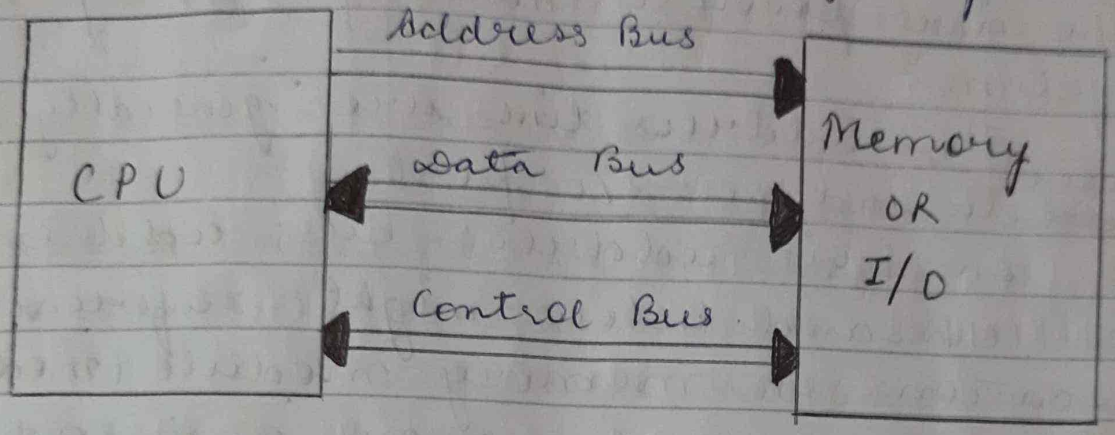
A Bus is a communication pathway connecting 2 or more devices. It is a shared transmission medium. Multiple devices connect to the bus, and a signal transmitted by any one device is available for reception by all other devices attached to the bus. If two devices transmit ~~over~~ during the same time period, their signal will overlap. Only one device at a time can successfully transmit.

A bus consisting of multiple comm^n pathways or lines. Each line is capable of transmitting signals representing 0 and 1. A sequence of binary digits can be transmitted across a single line. Several lines of a bus can be used to transmit binary digits in parallel.

Computer system contain a no. of different buses that provide pathways between components at various levels of the computer system hierarchy.

A bus that connects major components (processor, memory, I/O) is called a system bus. A system bus consist typically of 50 to 100's of seperate lines. Each line is assigned a particular meaning or function.

On any bus the lines can be classified into 3 functional groups -



1. Data Lines
2. Address Lines
3. control lines

In addition, there may be power distribution lines that supply power to the attached modules.

(1) Data lines - These lines collectively are called data bus. It may consist of from 32 - 100's of seperate lines. The no. of lines is referred as width of the data bus because each line can carry 1 bit of information at a time. The width of the data bus is a key factor in determining overall system performance.

If data bus is 8 bit wide & each instruction is 16 bit long, then processor must access memory module twice during each instruct

cycl

(2) Address Lines – Used to designate the source or destination of the data on the data bus.

e.g. If CPU wishes to read a word (8, 16 or 32 bits) of data from memory, it puts the address of the desired word on the address lines. Clearly, the width of address bus determines the max. possible memory capacity of the system.

The address lines are generally also used to address I/O ports.

e.g. On an 8-bit address bus, address 01111111 and below might reference locations in memory module (module 0) with 128 words of memory and address 10000000 and above refer to devices attached to I/O. (module 1).

(3) Control Lines – The control lines are used to control the access to and the use of the data and address lines. Control signals transmit both command and timing information between system modules. Timing signals indicate the validity of data and address information. Command signals specify operations to be performed.

* Memory write – Causes data on the bus to be written into the address location.

* Memory Read – Causes data from the address location to be placed

on the bus.

* I/O Write - Causes data on the bus to be output to the addressed I/O port.
* I/O Read - Causes data from the addressed I/O port to be placed on the bus.
* Transfer ACK - Indicates data have been accepted from or placed on the bus.
* Bus Request - Module need to gain control of the bus.
* Bus Grant - A requesting module has been granted control of the bus.
* Interrupt Request - An interrupt is pending.
* Interrupt ACK - Acknowledge that the pending interrupt has been recognized.
* Clock - Used to synchronize operations.
* Reset - Initialize all modules.

Operation of Bus -

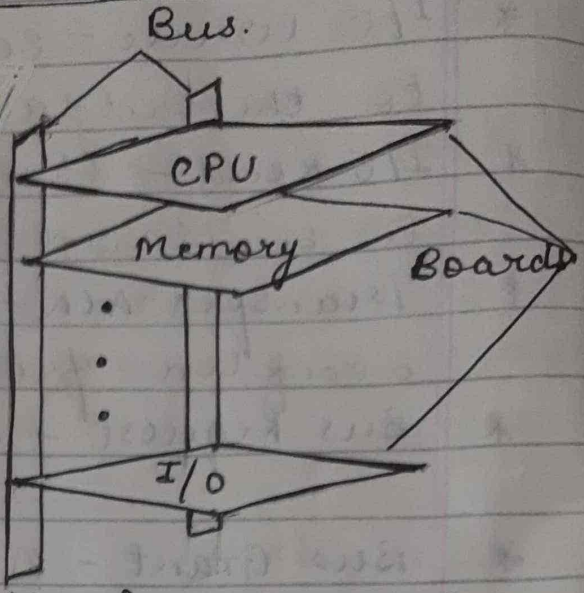
(1) If one module wishes to send data to another, it must do 2 things -

a) Obtain the use of bus
b) Transfer data via the bus

(2) If one module wishes to request data from another module -

a) obtain the use of bus.
b) Transfer a request to the other module over the appropriate control and address lines. It must then wait for that second module to send the data.

Physical Realization of Bus Architecture



Physical Realization of Bus Architecture

The system bus is actually a no. of parallel electronic conductors. These conductors are metal lines etched in a card or board (printed circuit board).

In this example, the bus consists of 2 vertical column of conductors. At regular intervals along the columns, there are attachment points in the form of slots that extend of horizontally to support a printed circuit board. Each of the major system components occupies one or more boards and plugs into the bus at these slots. This arrangement is more convenient. A small computer system may be acquired and expanded later (more memory, more I/O) by adding more boards. If a component on a board fails, that board can easily be removed and replaced.

1 **Register Transfer Language** – Digital module contain the registers and perform the operations on the data stored on them, called microoperation. A microoperation is a primary or elementary operation. The result of the operation may replace the previous information of the register or may be transferred to another register.

Example of Microoperation – Shift, Count, Clear, and load.

This language uses some predefined symbolic notation for each microoperation.

Register Transfer – The term 'register transfer' implies the availability of h/w logic ckt's that can perform the stated microoperation and transfer the result of the operation to the same or another register.

A register transfer language (RTL) can be used to describe the sequence of microoperation among the registers in symbolic form.

Features of RTL

* A symbolic language.
* Convenient tool for describing the internal organization of digital computers.
* Used to facilitate the design process of digital system.

The internal h/w organization of a digital computer is best defined by –

a) The set of register it contain and their function
b) The sequence of microoperation performed.
c) The control that initiate the sequence of microoperations.

The RTL adopted here is believed to be as simple as possible.

Register - Registers are designated by capital letter (sometimes followed by numerals) to denote the function of the register.

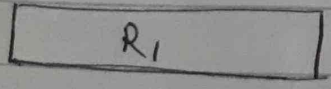
e.g.

Memory Address Register (MAR) - Register that holds an address of memory unit is usually called MAR.

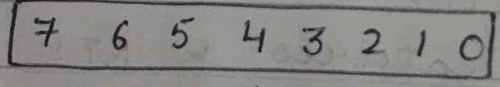
Program Counter (PC)

Instruction Register (IR)

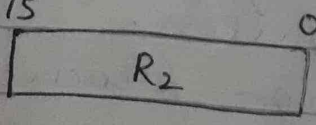
Processor Register (R1)



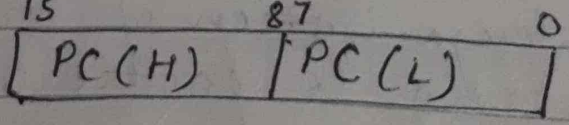
a) Register R



b) Showing individual Bits



c) numbering of bits



d) Divided into 2 parts
H - High Byte
L - Low Byte

Block Diagram of Register

Information Transfer (Register Transfer) –

$$R_2 \leftarrow R_1$$

denotes a transfer of content of register R_1 into register R_2. It designates a replacement of the content of R_2 by the content of R_1. The content of the source register R_1 does not change after the transfer. Transfer implies that circuits are available from O/P of source register to the I/P of designated register.

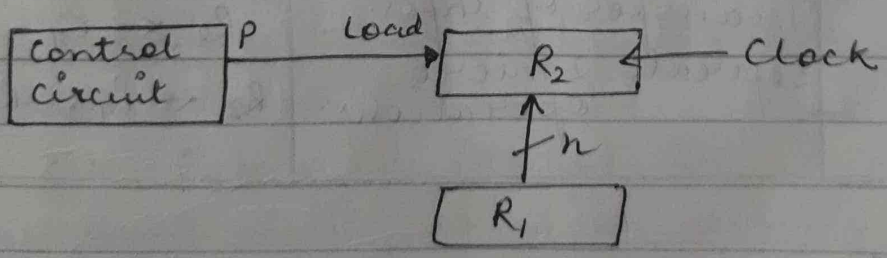
$$\text{If } (P=1) \text{ then } (R_2 \leftarrow R_1)$$

where P is a control signal generated in the control section.

Control Function – A control function is a boolean variable that is equal to 0 or 1.

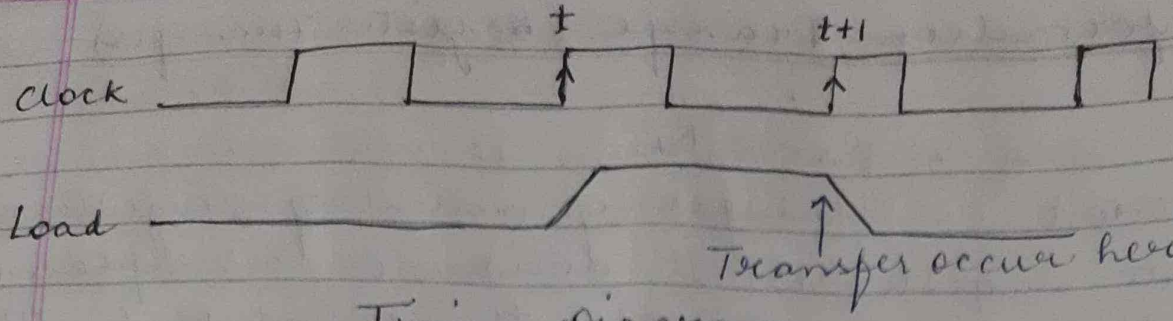
$$P: R_2 \leftarrow R_1$$

The control condition is terminated with a colon. Transfer operation be executed by the h/w only if P=1.



Block diagram (Transfer from R_1 to R_2 when P=1)

n output of register R_1 are connected to n input of R_2. n is used to indicate any no. of bits for the register. R_2 has a load i/p which is activated by P control variable, which is synchronized with the same clock applied to R_2.



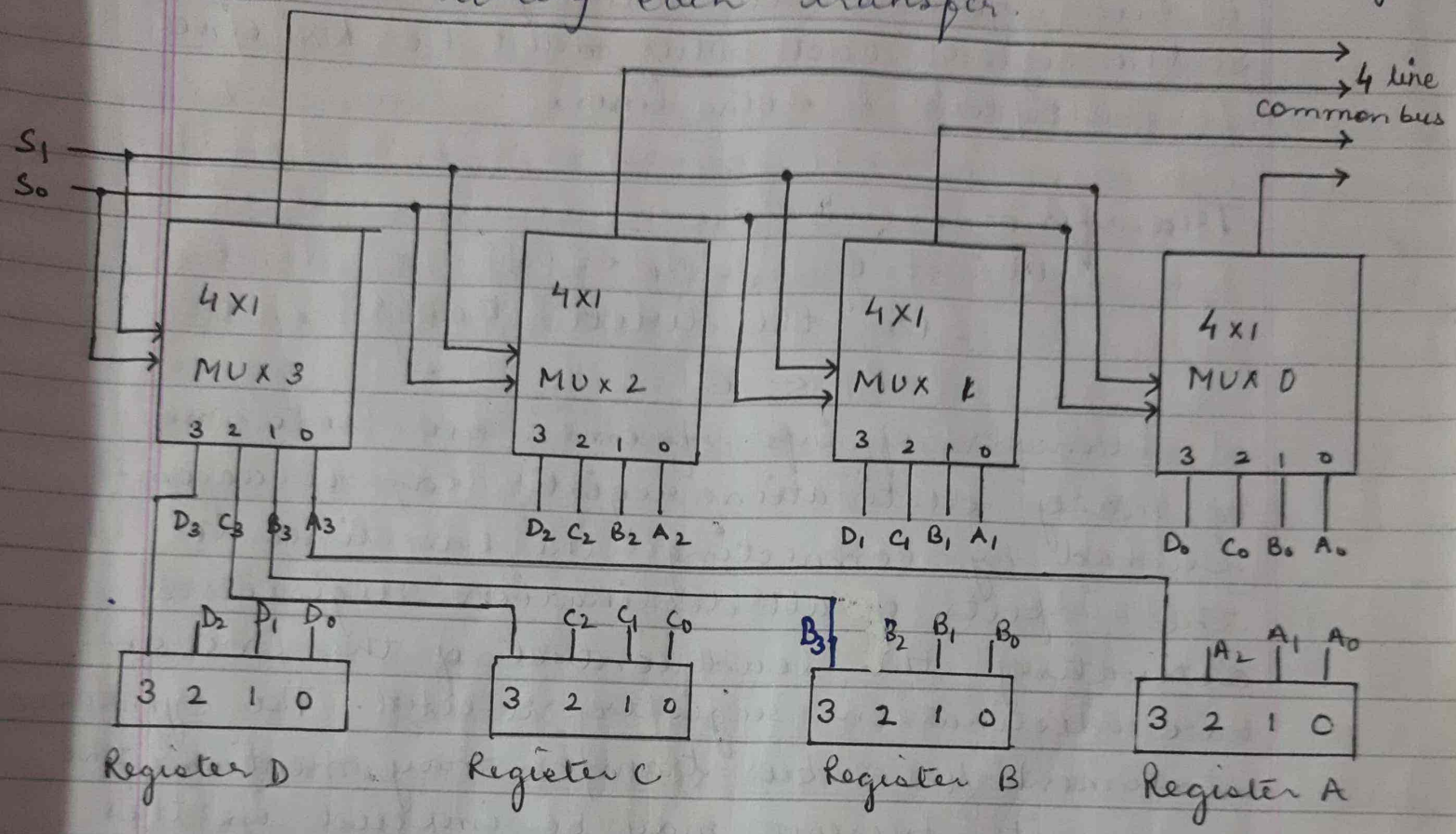
Timing Diagram

P is activated in the control section by rising edge of the clock pulse at time t. The next (+)ve transition of clock at time t+1 finds the load active and data are loaded into R_2 in parallel. P may go back to 0 at time t+1, otherwise the transfer will occur with every clock pulse transition while P remains active. Even though P becomes active just after time t, the actual transfer does not occur until the register is triggered by the next (+)ve transition at time t+1.

Symbol	Description	Example
Letters or Numerals	Denotes a register	MAR, R2
Parentheses ( )	Part of a register	R2 (0-7), R2 (L)
Arrow (←)	Transfer of info.	$R_2 \leftarrow R_1$
Comma ,	seperate 2 micro operations	$R_2 \leftarrow R_1$, $R_1 \leftarrow R_2$

Bus Transfer — A more efficient scheme for transferring information b/w registers in a multiple register configuration is a common bus system. A bus structure consist of a set of common lines, one for each bit of register, through which

binary info. is transferred one at a time. Ctrl signal determine which register is selected by the bus during each transfer.



One way of constructing a common bus system is with multiplexers. MUX 0 multiplexes four 0 bits of the registers etc.

Bus Selection - S_0 and S_1 are 2 selection lines. When

$S_1 S_0 = 00$ (zero), the zero data input of all four multiplexers are selected and applied to the ~~otu~~ output that form the bus. This causes bus line to receive the content of register A.

When $S_1 S_0 = 01$ (register B is selected) and so on.

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

No. of multiplexer = n (no. of bits in each register)

A bus system will multiplex k registers. So the size of each Mux must be k×1 since it multiplexes k data lines.

Transfer -

$BUS \leftarrow C$, $R_1 \leftarrow BUS$

or the direct transfer

$R_1 \leftarrow C$

A transfer of info. from a bus into one of many destination register can be accomplished by connecting the bus lines to the inputs of all destination registers & activating the load control of the particular destination register selected. The symbolic statement for a bus transfer may mention the bus or its presence may be implied in this statement. The register transfer is symbolized as ~~follows:~~ given above.

The content of register C is placed on the bus and the content of the bus is loaded into R_1, by activating its load control input

Memory Transfer -

Read - The transfer of information from a memory word to the outside environment is called a read operation.

Write - The transfer of new information to

be stored into the memory is called a write operation.

Word - A word will be symbolized by the letter M. A particular word is selected by the memory address during the transfer. It is necessary to specify the address of M when writing memory transfer operation. This will be done by enclosing the address in square brackets following the letter M.

Address Register (AR) - A memory unit that receives the address from a register, called AR.

Data Register - The data are transferred to another register called the data register (DR)

Read : DR ← M[AR] {Memory Read}

A write operation transfers the content of a data registers to a memory word M selected by the address. Assume that the i/p data are in register R_1 and is in AR.

write : M[AR] ← R_1 {Memory write}

This causes a transfer of information from R_1 into the memory word M selected by the address in AR.

Bus Arbitration

Computer system contain a no. of buses at various levels. The CPU contains a no. of internal buses for transferring information between processor register and ALU. A memory bus consists of lines for transferring data, address and

read / write information. An I/O is used to transfer information to and from input and output devices.

A bus that connects major components in a multiprocessor system such as CPU's, I/O's & memory is called a system bus. The physical ckt's of a system bus are contained in a no. of identical printed circuit boards. Each board belongs to a particular module. Circuit are connected in parallel through connectors. Each pin in each circuit connector is connected by wire to the corresponding pin of all other connectors in other boards.

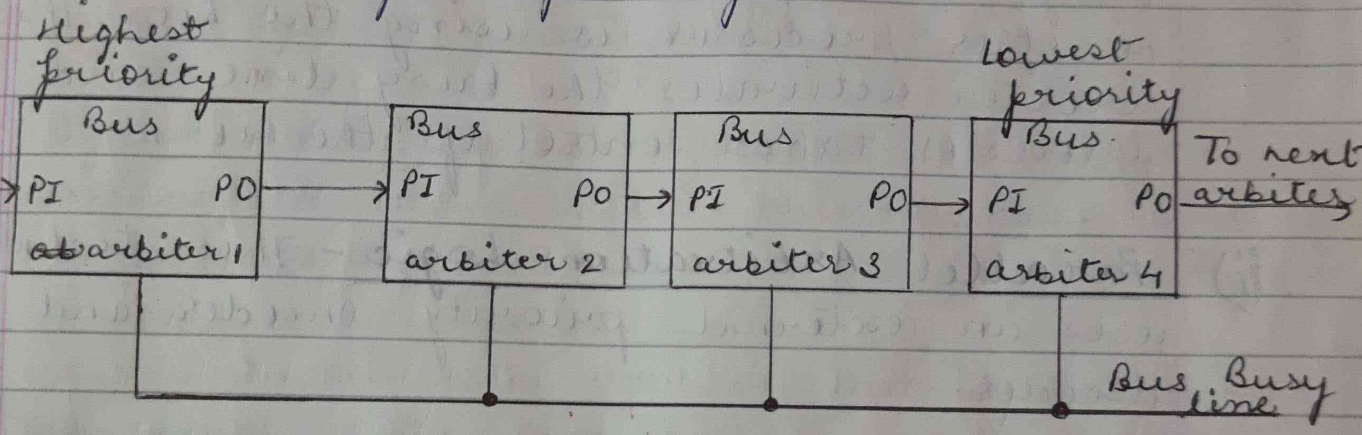
Arbitration - The processor in a shared memory multiprocessor system request access to common memory or other common resources through the system bus. If no other processor is currently utilizing the bus, the requesting processor may be granted access immediately. However, the requesting processor must wait if another processor is currently utilizing the system bus. Other processor may request the system bus at the same time. Arbitration must then be performed to resolve this multiple contention for shared resources.

Types of Arbitration

i) Serial Arbitration Procedure
ii) Parallel " "
iii) Dynamic " "

i) Serial Arbitration Procedure - Arbitration procedure service all processor request on the basis of established priorities. The serial priority resolving technique is obtained from a daisy-chain connection of bus arbitration circuits.

The processors connected to the system bus are assigned priority according to their position along the priority control line. The device closest to the priority line is assigned the highest priority.



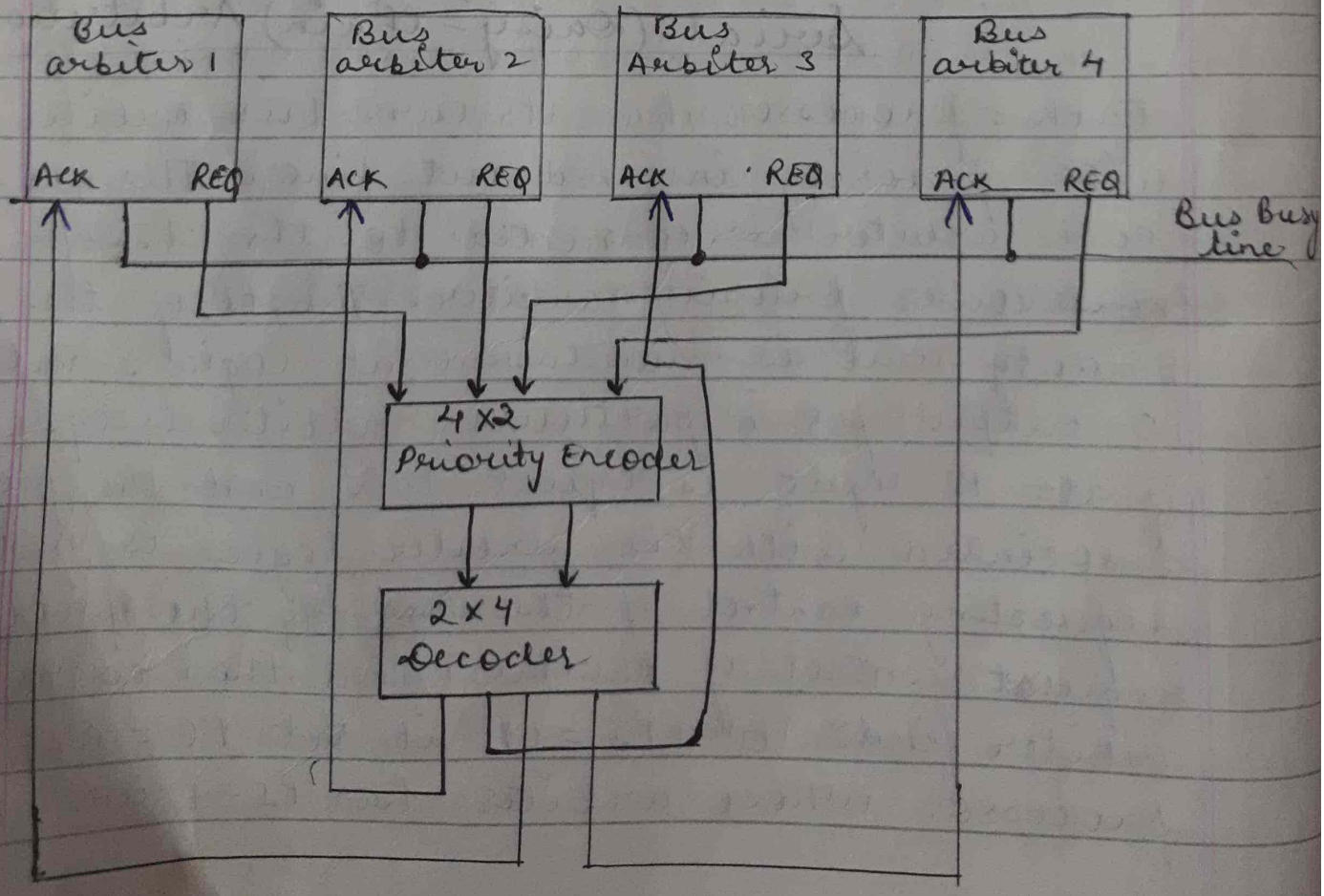
Serial (Daisy-Chain) Arbitration

Each processor has its own bus arbiter logic with priority in and out lines. The PO of each arbiter is connected to the PI of the next lower priority arbiter. The PI of the highest priority unit is maintained at logic 1 value. The PO output for a particular arbiter is equal to 1 if its PI input is equal to 1 and the processor associated with the arbiter logic is not requesting control of the bus. If the processor request control of the bus and the corresponding arbiter finds its PI = 1, it sets PO = 0. Thus the processor whose arbiter has PI = 1 and PO = 0

is given control of the bus.

A processor may be in middle of a bus operation when a higher priority processor requests the bus, the lower priority processor must complete its bus operation before it relinquishes control of the bus. The busy bus line provides a mechanism for an orderly transfer of control. The busy line provides a wired OR logic connection. When an arbit receives control of the bus it examines the busy line. If the line is inactive, it means no other processor is using the bus. The arbiter activates the busy line and its processor takes control of the bus.

ii) Parallel Arbitration Logic - This technique uses an external priority encoder and a decoder.



Each arbiter has a bus request output line & acknowledgement input line. Each arbiter enables the request line when its processor is requesting access to the system bus. The processor takes ctrl of the bus if its acknowledge input line is enabled. The bus busy line provides an orderly xfer of control. The 8/3 of encoder generates a 2-bit code which represents the highest priority unit among those requesting the bus. The 2-bit code from the encoder output drives the 2x4 decoder which enables the proper ack line to grant bus access to the highest-priority unit.

ii) Dynamic Arbitration Algorithm – The two bus arbitration use a static priority algo. since the priority of each device is fixed by the way it is connected to the bus.

In contrast, Dynamic Priority algo gives the system the capability for changing the priority of the devices while the system is in operation.

Procedures –

a) Time Slice – This algo. allocates a fixed length time-slice of bus time that is offered sequentially to each processor in round robin fashion.

) Polling – The bus system that uses polling, the bus grant signal is replaced by a set of lines called poll lines which are connected to all units. These lines are used by the bus controller to define an address for each device connected to the bus. The bus controller sequences through

the addressess in a prescribed manner. When a processor that requires access recognizes its address, it activates the bus busy line and then access the bus.

c) Least Recently Used – LRU algo. gives the highest priority to the requesting device that has not used the bus for the longest interval. Priorities are dynamically changed to give every device an opportunity to access the bus.

d) FIFO – First-come, first served. The bus controller establishes a queue arranged according to the time that the bus request arrives.

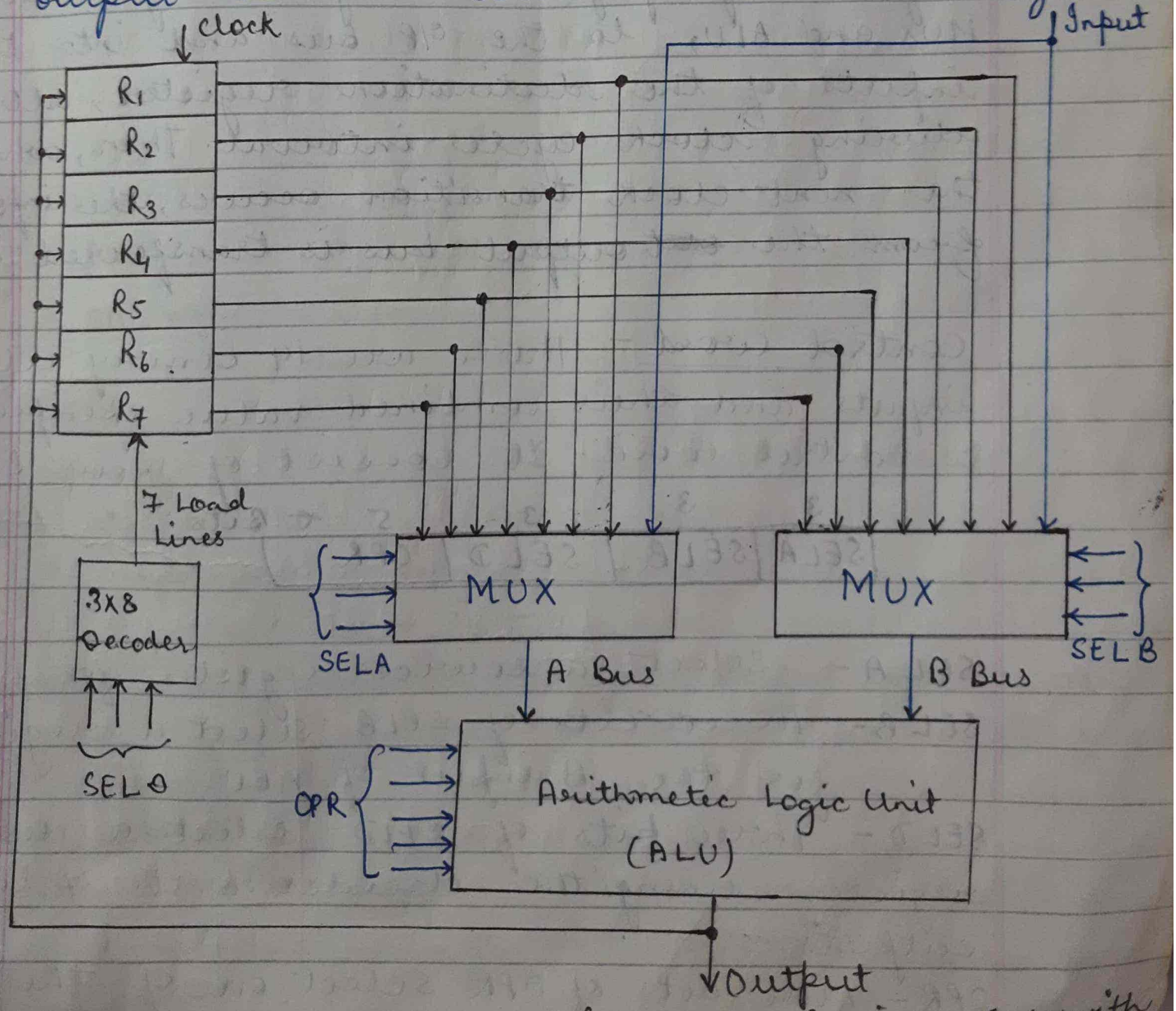
e) Rotating Daisy Chain – A dynamic extension of the daisy chain algorithm. There is no central bus controller. The priority line is connected from the priority-out of the last device back to the priority-in of the first device in a closed loop. Whichever device has access to the bus serve as a bus controller for the following arbitration. Each arbiter priority for a given bus cycle is determined by its position along the bus priority line from the arbiter whose processor is currently controlling the bus. Once an arbiter release the bus, it has the lowest priority.

General Register Organization - Memory locations are needed for storing pointer, counter etc. Having to refer to memory locations for such applications is time consuming because memory access is the most time consuming operation in computer.

It is more efficient and convenient to store these intermediate values in processor registers, and connect these large no. of registers through a common bus. The registers communicate with each other not only for direct data transfer but also while performing various microoperations.

Bus System - A bus organization for 7 CPU

is shown. The output of each register is connected to two MUX to form two buses A and B. The selection lines in each MUX select one register or input data for each bus. The A and B buses form input to ALU. The operation selected in ALU determines the arithmetic and logic microoperation that is to be performed. The result of microoperation is available for O/P data and also goes into the input of all registers. The register that receives the O/P from output bus is selected by a decoder. The decoder activates one of the register load inputs, thus providing transfer path between output bus and selected destination register.



Block Diagram Register set with common ALU.

For e.g. - To perform the operation
$R_1 \leftarrow R_2 + R_3$.

Inputs are -

a) MUX A selector (SELA) - To place the contents of R_2 into bus A.

b) MUX B selector (SELB) - To place the contents of R_3 into bus B.

c) ALU operation selector (OPR) - To provide the arithmetic addition operation A+B.

d) Decoder Destination Selector (SELD) - To transfer the content of output bus into R_1.

The data from the two source registers propogate through the gates in the MUX and ALU, to the O/P bus and into the inputs of the destination register, all during clock cycle interval. Then, when the next clock transition occurs, the information from the output bus is transferred into R

Control word - There are 14 binary selection inputs and their combined value specifies a control word. It consist of four fields

3	3	3	5 ← Bits
SELA	SELB	SELD	OPR

SELA - Select a source register for A input

SELB - Three bits of SELB select a register for the B input of ALU.

SELD - Three bits of SELD select a destination register using the decoder & its 7 load outputs.

OPR - Five bits of OPR select one of the operation in the ALU.

ALU – The ALU provides arithmetic and logic operation. In addition, CPU must provide shift operations. The shifter may be placed in the input of ALU to provide preshift capability or at the output of ALU to provide postshifting capability.

OPR Slect	operation	Symbol
00000	Transfer A	TSFA
00001	Increment A	INCA
00010	add A+B	ADD
00101	Subtract A-B	SUB
00110	Decrement A	DECA
01000	AND A & B	AND
01010	OR A & B	OR
01100	XOR A & B	XOR
01110	Complement A	COMA
10000	Shift right A	SHRA
11000	Shift left A	SHLA

e.g. of microoperation

$R_1 \leftarrow R_2 - R_3$

Field	SELA	SELB	SELD	OPR
Symbol :	R_2	R_3	R_1	SUB
control word :	010	011	001	00101

Example of Increment

$R_6 \leftarrow R_6 + 1$

increment and transfer operations do not use B input of the ALU. We assign 000 to any unused field

control word : 110 000 110 00001

The direct transfer of data from i/p to o/p is accomplished with a ctrl word of all 0's.

Many other microoperations can be generated in the CPU. The most efficient way to generate control word is to store them in memory unit referred to as control Memory. By reading consecutive control word from memory it is possible to initiate the desired sequence of microoperations for the CPU. This type of control is referred to as microprogrammed control.

Stack Organization -

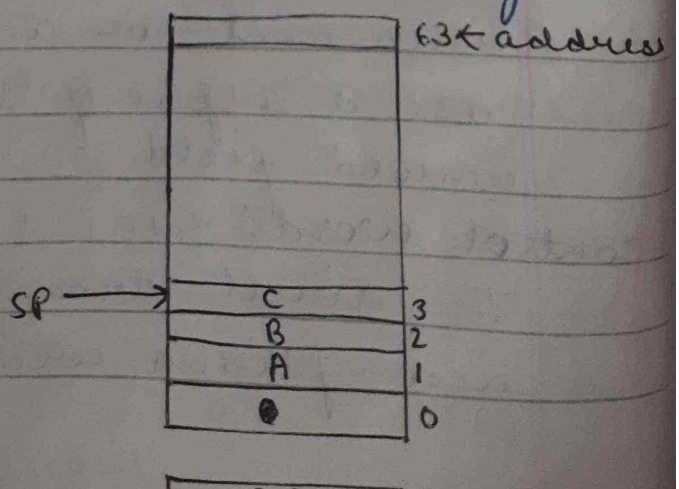
LIFO - Last In First Out

Stack Pointer - Stack is a memory unit with an address register that can count only after an initial value is loaded into it. The register that holds the address for the stack is called stack pointer because its value always point at the top item in the stack.

operations - i) PUSH
ii) POP

Register Stack - A stack can be organized as a collection of a finite no. of registers. The SP contains a binary no. whose value is equal to the address of word that is currently on top of the stack.

FULL EMTY



Block Diagram of 64-word Stack

In 64 word stack SP contains 6 bits because $2^6 = 64$ when 63 is incremented by 1.

$111111 + 1 = 1000000$ (Zero) – 6 LSB.

when 0 is decremented by 1.

$000000 - 1 = 111111$ (63)

The one bit register FULL = 1 if stack is full

EMTY = 1 if stack is empty

DR – Data register that holds the binary data to be written into or read out of the stack.

Push If the stack is not full (if FULL = = 0), a new item is inserted with a push operation.

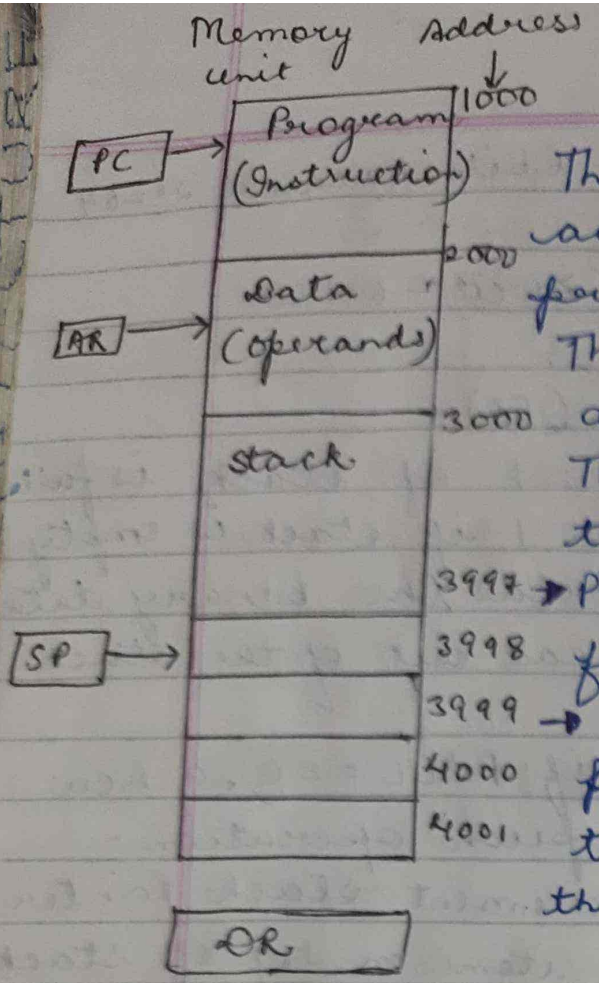
```
SP ← SP+1          Increment stack pointer
M[SP] ← DR         Write item on top of stack
If (SP = 0) then (FULL ← 1)   check if stack is full
   EMTY ← 0        Mark stack not empty
```

Pop A new item is deleted from the stack if stack is not empty (EMTY = 0)

```
DR ← M[SP]         Read item from top of stack
SP ← SP-1          Decrement stack pointer
If (SP = 0) then (EMTY ← 1)   check if stack is empty
   FULL ← 0        mark the stack not full.
```

Note that an erroneous operation will result if the stacked is pushed when FULL = 1 and Popped when EMTY = 1

Memory Stack – A stack can be implemented in RAM attached to CPU, by assigning a portion of memory to a stack operation and using a processor register as a stack pointer. The SP points at the top of stack.



Computer memory with program, data & stack segment

The program counter (PC) points at the address of next instruction in the program.

The address register (AR) points at array of data.

The three registers are connected to a common address bus.

PC is used during the fetch phase to read an instruction.

AR is used during the execute phase to read an operand. SP is used to push or pop items into or from the stack.

The first item in stack is at address 4000 and last at 3000.

Push operation - $SP \leftarrow SP - 1$

$M[SP] \leftarrow DR$.

A memory write operation inserts the word from DR into top of stack

Pop operation -

$DR \leftarrow M[SP]$

$SP \leftarrow SP + 1$

Stack limits - The stack limits (stack overflow or stack underflow) can be checked by using 2 processor registers -

(1) One to hold upper limit (3000 in this case)

(2) Other to hold lower limit (4001 " ")

After a push operation, SP is compared with upper limit. After a pop operation, SP is compared with lower limit.

Advantage of Memory Stack – CPU can refer to it without specifying an address, since the address is always available & automatically updated in stack pointer.

Reverse Polish Notation – Stack is very efficient for evaluating arithmetic expression

Infix Notation	A + B
Prefix or Polish Notation	+ A B
Postfix Or Reverse Polish Notation	A B +

Scan the expression from left to right. When an operator is reached, perform the operation.

eg i) A * B + C * D.

A B * C D * +

ii) (A+B) * [C * (D+E) + F]

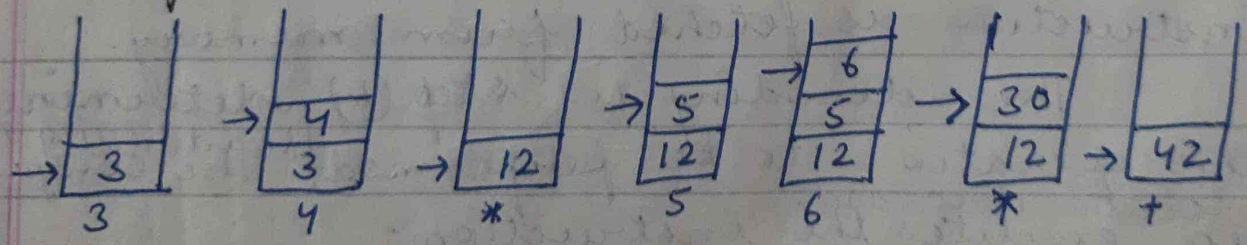
A B + D E + C * F + *

RPN, combined with a stack arrangement of registers is the most efficient way known for evaluating arithmetic expression i.e. parantheis free polish notation.

for e.g. (3 * 4) + (5 * 6)

In RPN – 3 4 * 5 6 * +

Stack operation –



Most compiler, irrespective of their CPU organization convert all arithmetic expression into polish notation because this is the most efficient way for translating arithmetic exp into machine language instruction.

Addressing Modes - The way the operands are chosen during program execution is dependent on the addresing mode of the instruction. The addresing mode specifies a rule for interpreting or modifying the address field of the instruction before the operand is actually referenced.

Use -

a) To give programming versatility.
b) To reduce the no. of bits in addressing field of instruction.

The availability of the addressing modes give the experienced programmer flexibility for writing programs that are more efficient w.r.t. the no. of instruction and execution time.

The basic operation cycle of computer -

a) Fetch the instruction from memory.
b) Decode the instruction
c) Execute the instruction

Program Counter (PC) - PC keeps track of the instruction in the program stored in memory. It holds the address of an instruction to be executed next and is incremented each time an instruction is fetched from memory.

The decoding in step (b) determines the operation to be performed, the addressing mode & location of operand. The computer then executes the instruction.

The various methods of accessing data is called as the 'Addresing Modes'.

Mode Field - Operation code specifies the

operation to be performed.

Instruction format for mode field

Opcode	Mode	Address.

Mode field is used to show the operands needed for the operation.
There may or may not be address field in the instruction. If there is an address field, it may designate a memory address or a processor register.

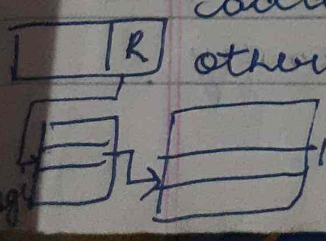
1) Implied Mode - Need no address field (In this the operands are specified implicitly in the definition of the instruction)
e.g. The instruction 'complement accumulator'

Instruction

2) Immediate Mode - (The operand is specified in the instruction itself.) It has an operand field rather than an address field.
operand field = Actual operands + operation
It is useful for initializing registers to constant value. (The second word of the instruction is taken as a operand rather than an address.)

3) Register Mode - (When the address field specifies a processor register or memory word. the instruction is said to be in register mode.) The operands are in registers that reside within the CPU. A particular register is selected from a register field.
eg. - 8085 instruction in register mode is MOV A, B which xfer the content of register B into AC A.

4) Register Indirect Mode - (The instruction specifies a register in the CPU whose contents give the address of the operand in memory.) In other word 'the selected register contains



EA - Sometimes address field of the instruction is the address of ~~instruction~~ operand but sometimes not. It gives the required address after some calculation. The address obtained is called EA.

the address of operand rather than the operand itself.

Advantage - The address field of instruction uses fewer bits to select a register.

5) Autoincrement OR Autodecrement Mode - It is similar to register indirect mode except that the register is incremented or decremented after (or before) its value is used to access memory. This can be achieved by using increment or decrement instruction.,

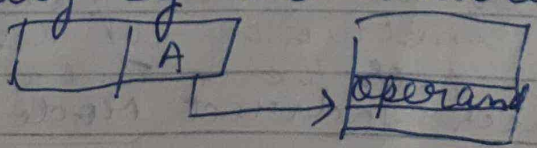
The address field of an instruction is used by the control unit in CPU to obtain the operand from memory. Sometimes the value given in the address field is the address of operand but sometimes just an address from which the address of operand is calculated.

Effective Address - Defined to be the memory address obtained from the computation dictated by the given addressing mode. In a system with virtual address EA is virtual add. or a register, In asystem without virtual memory EA is the ~~is~~ main memory address or a register.

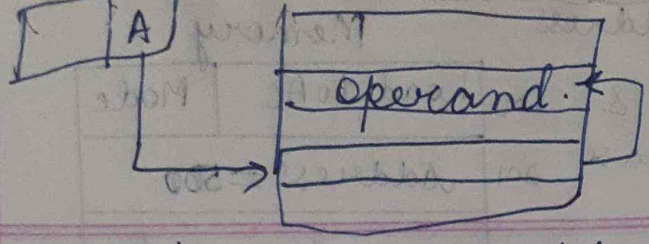
6) Direct Addressing Mode

Effective address = Address part of Instruction

operand reside in memory & its address is given directly by the address field of instruction.



7) Indirect Address Mode - Address field of instruction gives the address where the effective address is stored in memory.



(EA) Effective address = address part of instruction + content of CPU register

8) Relative Address Mode - The content of PC is added to the address part of instruction in order to obtain the effective address. Address part usually is signed number.

e.g. Assume PC contains no = 825 &
Address part = 24.
location 825 is read from memory & PC is incremented by 1 ~~to~~ to 826.
Effective address = 826 + 24 = 850.

9) Indexed Addressing Modes - (The content of an index register is ~~added to~~ added to the address part of the instruction to obtain EA. Index register is special CPU register that contains index value) Address field defines the beginning address of data array. Each operand is stored relative to beginning address. Difference b/w beginning address and address of operand is index value.

10) Base Register Addressing Mode - The content of base register is added to the address part of the instruction to obtain the effective address. This mode is used in computer to facilitate the relocation of programs in memory.

Addressing Mode	Effective Address	Content to AC
Register	—	400
Immediate operand	201	500
Register Indirect Mode	400	700

The value in PC after the fetch phase & during the execute phase is 202.

PC = 200

R1 = 400

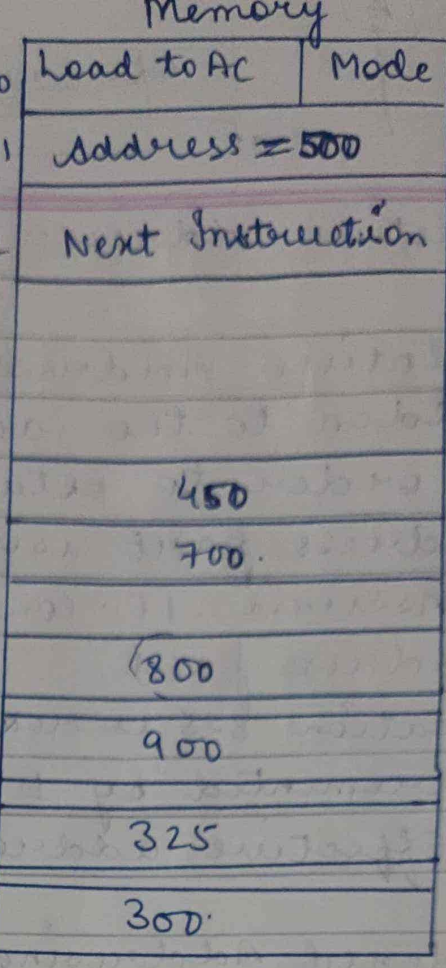
XR = 100

AC

R1 = Processor Register

XR = Index Register

AC receives the operand after the instruction is executed.



Numerical Example for addressing modes

Addressing Mode	Effective Address	Content to AC
Autoincrement	400	700
Autodecrement	399	450
Direct address	500	800
Indirect address	800	300
Relative address	500+202 = 702	325
Index Address	100+500 = 600	900

Processor Organization – The processor part of a computer CPU is sometimes referred as the data path of the CPU because the processor forms the paths ~~referred to as the data~~ for the data transfers between registers in the unit. The various paths are said to be controlled by means of gates that open the required paths and close all others.

The value in PC after the fetch phase & during the execute phase is 202.

PC = 200

R1 = 400

XR = 100

AC

R1 = Processor Register

XR = Index Register

AC receives the operand after the instruction is executed.

Address	Memory	
200	Load to AC	Mode
201	Address = 500	
202	Next Instruction	
399	450	
400	700	
500	800	
600	900	
702	325	
800	300	

Numerical Example for addressing modes

Addressing Mode	Effective Address	Content to AC
Autoincrement	400	700
Autodecrement	399	450
Direct address	500	800
Indirect address	800	300
Relative address	500+202 = 702	325
Index Address	100+500 = 600	900

Processor Organization – The processor part of a computer CPU is sometimes referred to as the data path of the CPU because the processor forms the paths ~~referred to as the data~~ for the data transfers between the registers in the unit. The various paths are said to be controlled by means of gates that open the required paths and close all others.

The execution unit (Data Processing Unit) contains a set of registers for storing data and an ALU for execution of arithmetic & logic operations. The processor can be designed to fulfill the requirements of a set of data paths for specific applications.

The processor has three main parts-

i) Register - The register is a storage device that is used to store words. The registers are used to transfer data and are also used for some microoperations. The group of register is sometimes called scratch pad memory. It is also called a register file.

Types -

a) Register Bank OR Register file - The CPU of a system is usually equipped with a no. of general purpose registers fabricated as a small fast memory. Certain registers of a register bank may be reserved for specific function such as stack pointer or program counter.

b) Accumulator Register - used to perform arithmetic & logic operation and the result is stored in accumulator.

c) Index Register - Used to perform modification of an operation address.

d) Valid Area Register - Used to define the upper and lower address of the memory address within which the user's program or data block reside.

ii) Arithmetic & Logic Circuit - when the processors are enclosed in an IC package it is called bit slice microprocessor or ~~arith~~ ALU.

iii) Control Unit - It generates sequence of control signals which dictate instructions to all the devices to perform their operations. Each device perform the corresponding operation, after receiving the control signal from the control unit.