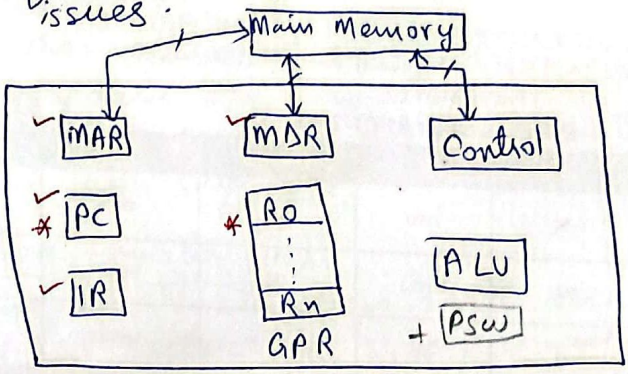
REGISTERS

- Length of various registers and which registers to include are design issues.



user visible *
not user visible ✓
read only +

ALU contain storage location called registers.

* **User Visible Register**: Enable the machine or assembly-language programmer to minimize main memory references by optimizing use of registers.
- They can be referenced by means of machine language programs.

* **control and status registers**: Used by control unit to control the operation of the processor and by privileged, operating system programs to control the execution of programs.

There is not a clean separation of registers into these two categories. For example, on some machines program counter is user visible (e.g: pentium), but on many is not.

USER VISIBLE REGISTER required by machine language

1- **General Purpose Registers**:
- used to temporary store the operands.
- can contain address also in some cases (indirect addressing mode).

2- **Dedicated Data and Address Registers**:
- Data registers hold data only and cannot be implied in the calculation of an operand address.
- Address registers may be devoted to a particular add. mode.
- some address registers are:
  - a- Segment pointers: holds the address of the base of the segment.
  - b- Index registers: used for indexed addressing and may be autoindexed.
  - c- Stack pointer: If there is user visible stack addressing, then this register points to the top of the stack.

In some machines subroutines call result in automatic saving of all user-visible registers, to be restored on return. On other machines, it is the responsibility of the programmer to save the relevant contents of regs.

Scanned with CamScanner

ONTROL & STATUS REGISTERS:

Most commonly found registers are:

ogram Counter (PC):

ontains the address of an instruction to be fetched.
rocessor updates the PC after each instruction fetch so that the PC always points to the next instruction to be executed.
a branch or skip instruction also modify the contents of PC.

Instruction Register (IR)

- Contains the instruction most recently fetched.
- The fetched instruction is loaded into an IR, where the opcode and operand specifiers are analyzed.

Memory address registers (MAR):

Contains the address of an instruction in memory.
- connects directly to the address bus.

Memory ~~dat~~ Buffer register (MBR):

- contains a word of data to be written to memory or word most recently read.
- connects directly to the data bus.

MAR and MBR are not provided in a machine, then some equivalent uffering mechanism must be there whereby the bits to be transferred o the system bus are staged and bits to be read from the data bus are temporarily stored.

he four registers mentioned above are used for the movement data between the processor and memory.

~~PROCESSOR STATUS~~

PROGRAM STATUS WORD (PSW):

- It could be a register or a set of registers.
- contains program status information.
- typically contains condition codes (flag register) and other information (starting address of program in memory, etc).

Condition Codes (Flag Register):

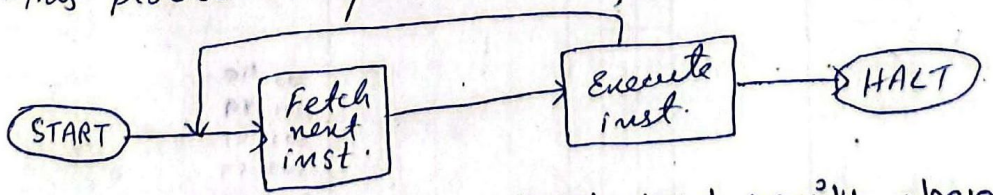
- partially visible to the user. These are read only.
- condition codes are set by the processor hardware as the result of operations. For eg: an arithmetic operation may produce a positive, negative or zero result.
- In addition to the result itself being stored in memory, a condition code is also set. Subsequently the code may be tested as a part of some conditional branch operation.

Scanned with CamScanner

- they make the circuitry complex, so some machines don't include condition codes at all. (eg IA-64 architecture) (12)
- common fields or flag include the following:
  1- Sign: contains the sign bit of the result of the last arithmetic operation.
  2- Zero: Set when the result is 0.
  3- Carry: Set if an operation resulted in a carry into or borrow out of a higher order bit. Used for multiword arithmetic operations.
  4- Equal: Set if a logical compare result is equality.
  5- Overflow: Used to indicate arithmetic overflow.
  6- Interrupt enable/disable: Used to enable or disable interrupts.
  7- Supervisor: Indicates whether the processor is executing in supervisor or user mode. Certain privileged instructions can be executed only in supervisor mode, and certain areas of memory can be accessed only in supervisor mode.

BASIC INSTRUCTION CYCLE:

- A program is a set of instructions stored in memory. (Mostly in consecutive locations).
- The processing required for a single instruction is called an instruction cycle.
- In simplest form, an instruction cycle consists of two steps:
  1- Instruction fetch: fetches the instruction from memory (one at a time).
  2- Instruction execute: executes the instruction. Instruction execution may involve several operations and depends on the nature of instruction.
- This process is repeated until all instructions are executed.

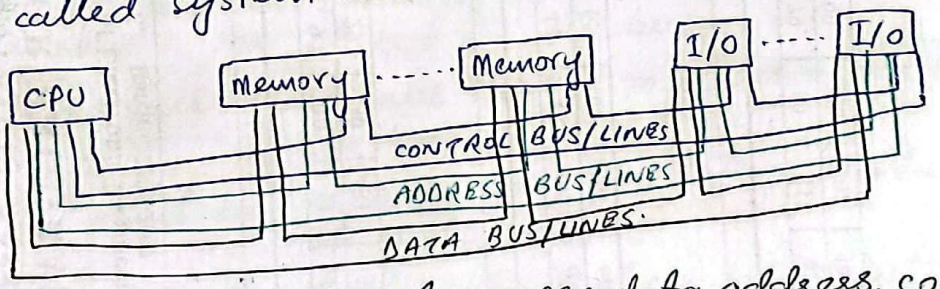


Accumulator: Employed to hold temporarily operands and results of ALU operations.

Scanned with CamScanner

BUS INTERCONNECTION: 3.4

- The most commonly used interconnection structure so far is Bus and vaibus multiple-bus structures.
- A bus is a communication pathway connecting two or more devices.
- A bus is a shared transmission medium.
- Only one device can successfully transmit the data at a time, otherwise the signals will be garbled.
- signal transmitted by one device is available for reception by all other devices attached to the bus.
- A bus consists of multiple lines, each line is capable of transmitting signals representing binary 1 and binary 0.
- A group of bits can be sent simultaneously, thus a reasonable speed is achieved. 8 bits → 8 lines transmit them in parallel.
- Computer system contains a number of different buses that provide pathways between components at various level of computer system hierarchy.
- A bus that connects major computer components (CPU, MM, I/O) is called system bus.



# of lines = width of bus.

- There are three functional groups: data, address, control.
- There may be power distribution lines that supply power to the attached modules.

1- DATA BUS:
- provides path for memory data between system modules.
- if width of data bus increases, performance increases.
- e.g: if data bus is 8-bits wide and instruction is 16-bits wide, then two cycles will be needed to fetch each instruction.
- width is usually in power of 2. (8/16/32/64...).

2- ADDRESS BUS:
- used to designate source or destination of data on data bus.
- also used to address I/O module and I/O ports.
- width of the address bus determines the maximum possible memory capacity of the system.

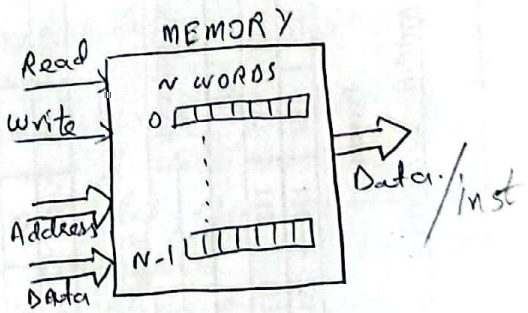
Scanned with CamScanner

INTERCONNECTION STRUCTURES 3.3

- The collection of paths connecting the various modules (processor, memory, I/O) is called interconnection structures.
- Major forms of input and output (exchanges) are as follows:

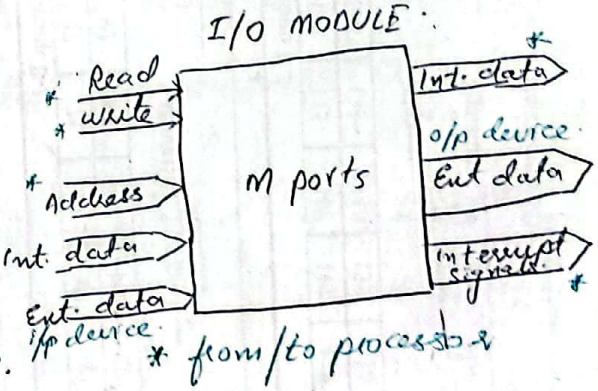
MEMORY:

- two operations possible:
  - read
  - write
- sometimes the memory also sends acknowledgement signals to processor. (asynchronous mode).



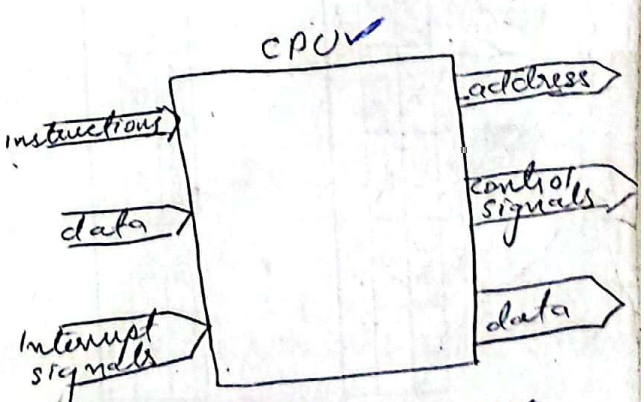
I/O MODULE:

- from an internal point of view (from/to processor), I/O is functionally similar to memory.
- 2 I/O operations, read and write.
- may control more than one I/O device.
- interface to an external device is called port. Each port has a unique address.
- there are external data paths for the input and output of data with an external device.
- it may send INT signal to processor.



PROCESSOR:

- reads instruction and data
- writes out data after processing.
- uses control signals to control the overall operation of system.
- receives interrupt signals.



Thus the interconnection structure must support the following types of transfers:

1- Memory to processor: Processor reads instruction/data from memory.
2- Processor to memory: Processor writes data to memory.
3- I/O to processor: Processor reads (via I/O module) from a device.
4- Processor to I/O: Processor sends data to I/O (via I/O module).
5- I/O to or from memory: an I/O module is allowed to exchange data directly with memory, without processor intervention — DMA (direct memory access)

Scanned with CamScanner

CONTROL BUS:

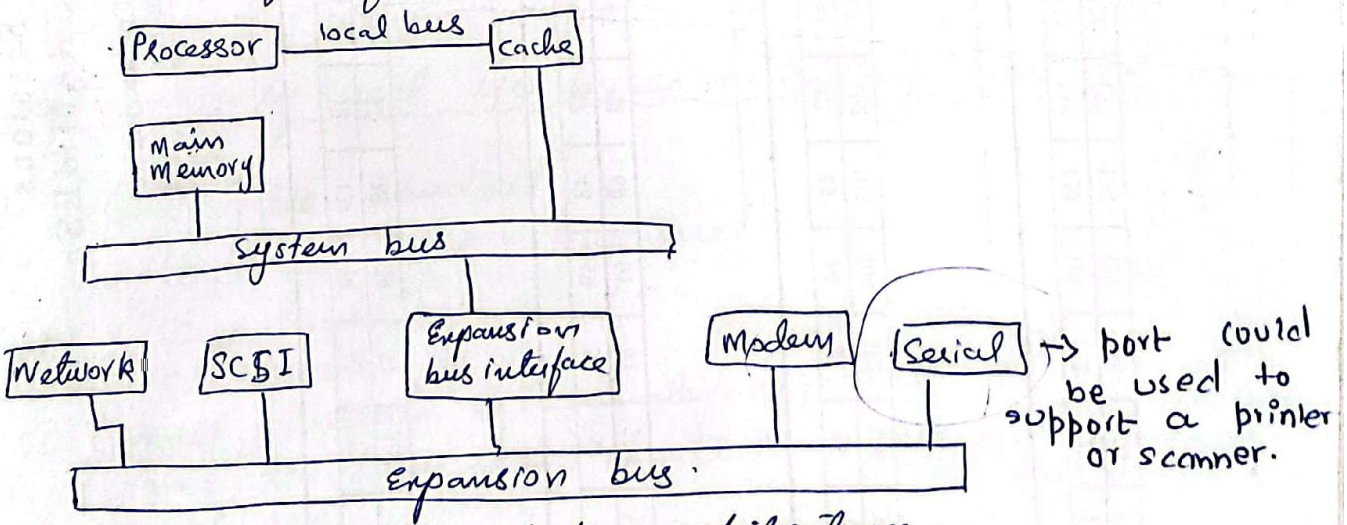
- used to control the access to and use of data and address buses.
- because the data and address lines are shared by all components, there must be a means of controlling their use.
- Control signals/lines transmit both:
  a- timing signals: indicate the validity of data and address information.
  b- command signals: specify operations to be performed.
- typical control lines include:
  1- Memory write: causes data on the bus to be written into the addressed location.
  2- Memory read: causes data from the addressed location to be placed on the bus.
  3- I/O write: causes data on the bus to be output to the addressed output port.
  4- I/O read: causes data from the addressed I/O port to be placed on the bus.
  5- Transfer ~~ack~~ ACK: Indicates that data have been accepted from or placed on the bus.
  6- Bus request: Indicates that a module needs to gain control of the bus.
  7- Bus grant: Indicates that the requesting module has been granted control of the bus.
  8- Interrupt request: Indicates that an interrupt is pending.
  9- Interrupt ACK: Acknowledges that the pending interrupt has been recognized.
  10- Clock: Used to synchronize operations.
  11- Reset: Initializes all modules.
- Before starting any bus operation, a module needs to get control of the bus.
- System bus is usually etched on the board or the components are on the same chip.
- Nowaday, for increasing number of modules ~~to increase the~~ ~~bus~~, bus architectures are replaced by ICN (Interconnection network).
- If a great number of devices are connected to the bus, performance will suffer because of the following two reasons:

Scanned with CamScanner

The more devices attached to the bus, the greater the bus length and hence the greater the propagation delay, more contention. The bus may become a bottleneck as the aggregate data transfer demand approaches the capacity of the bus. This problem can be countered to some extent by increasing the data rate that the bus carry and by using wider buses. But still not sufficient for rapidly growing devices (eg: graphics and video controllers, network interfaces).

BUS HIERARCHIES Page No: 88

- most computer systems use multiple buses, laid out in a hierarchy.
- It is possible to connect I/O controllers directly onto the system bus. A more efficient solution is to use expansion buses.
- An expansion bus interface buffers data transfers between the system bus and the I/O controllers on the expansion bus.
- This arrangement allows the system to support a wide variety of I/O devices and at the same time insulate memory-to-processor traffic from I/O traffic.



Traditional bus architectures.

* shows typical examples of I/O devices that might be attached to the expension bus.
* Networ connections include (LAN) and (WAN)
* SCSI (small computer system interface) is itself a type of bus used to support local disk drivers and other peripherals.

Scanned with CamScanner

Mezzanine Architecture

Some commonly used I/O buses nowadays are:

[See handout]

- PCI (Peripheral component interconnect) 32-bit/64-bit.
- PCI-X (PCI extended) 64-bits
- PCI-e (PCI express) serial
- SCSI (small computer system interface). 16-bit
- SAS (serial attached SCSI) [faster than SCSI & SATA).
- SATA (Serial ATA) AT attachment
- GPU (graphics processing unit)
- AGP (accelerated graphics port)
- USB (universal serial bus)
- Firewire.

iSCSI - ethernet port

SAN - connected to fibre cables

Scanned with CamScanner

LEMENTS OF BUS DESIGN

Bus types
2) Method of arbitration (centralized or distributed)
3) Timing.
4) Bus width.
5) Data transfer type.

1) BUS TYPES:

* Dedicated bus lines: A dedicated bus line is permanently assigned either to one function or to a physical subset of computer components. eg: separate buses for data & address
* Time multiplexed bus lines:
  - address and data may be transmitted over the same set of lines using Address Valid control line.
  - Address valid control line is activated when address is placed on the bus and vice versa when data is placed.
  - adv: save space and cost.
  - disadv: requires complex circuitry
    - potential reduction in performance because certain events that share the same lines, cannot take place in parallel. In 8086/88: half of the 16-bit address lines are multiplexed with data and upper half is multiplexed with status line. In 8085 lower 8-bits are multiplexed with data.

2) METHOD OF ARBITRATION:

Arbitration → problem solving.

- In all but simplest systems, more than one module may need control of the bus. Bus is shared among various module.
- eg: bus can be needed simultaneously by the Pr and by I/O module (DMA operation).
- As only one module at a time can successfully transmit data, so some method of arbitration is needed (to allocate bus to the initiator of the transfer)

* Centralized scheme:
  - a single hardware device, called bus controller or arbiter is responsible for allocating time on the bus.
  - the device may be a separate module or part of the controller.
* Distributed scheme:
  - there is no central controller, rather each module contains access control logic and the modules act together to share the bus.
  - transfer initiator, gets control of the bus → master.
  - other device involved in the exchange → slave.

Scanned with CamScanner

BUS WIDTH ☆

*wider the data bus, greater the number of bits transferred at one time.

*wider the address bus, greater the range of location that can be directly referenced.

(4) DATA TRANSFER TYPES

*Read operation: For dedicated as well as time multiplexed lines:

* some delay access for bus arbitration.

* data onto the data bus as soon as

* locked the data.

Scanned with CamScanner

SEMICONDUCTOR MEMORIES

- RWM — read/write memory
- ROM — read only memory
- RMM — read mostly memory.

RWM (read/write memory)

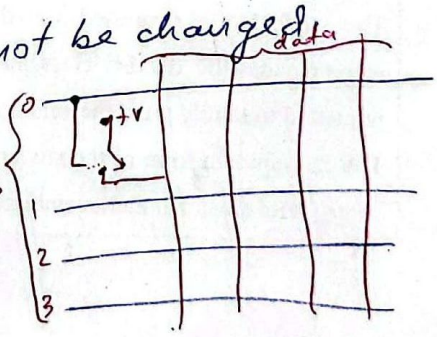
- data can be easily and rapidly read from or written into RAMs.
- reading and writing is accomplished through electrical signals.
- volatile memory.
- could be DRAM or SRAM.

* **DRAM (Dynamic RAM):**
  - made with cells that store data as charge on capacitor
  - capacitors have natural tendency to discharge; they require periodic charge refreshing to maintain charge storage.
  - they are DRO memories.
  - used in RAMs.

* **SRAM (static RAM):**
  - data is stored on flip flops (a logical element)
  - no refreshing is required, therefore fast.
  - expensive - complex circuit (relatively)
  - used in most caches.

ROM (read only memory)

- contains permanent pattern of data that cannot be changed.
- read operations are possible any time.
- write operation is possible only once.
- non-volatile and non erasable.
- used in microwave oven, older BIOS, etc.
- data is actually wired into the chip.



* MASK ROMs - factory programmed ROM.
* PROM (programmable ROM) - can be programmed by the user once.

RMM (read mostly memory)

- read operations are more frequent than write operations.
- read operation is as simple as in RAM but write operation requires especial hardware called ROM programmers.
- non-volatile and erasable.
- these ROMs should be erased before being rewritten.
- erasure also requires especial hardware.
- types:
  * EPROM
  * FLASH
  * EEPROM.

Scanned with CamScanner

	EPROM	FLASH	EEPROM
erasure process	uses uv light	→	electrical voltages are used – in circuit erasure
size	smaller in size than EEPROM (more dense)	← more dense than EEPROMs.	larger in size.
erasure time	20 min or more	1 or a few seconds	≈10 ms
endurance	100 programming cycles before chip is damaged.	100 to 10000 cycles	100000 cycles.
erasure unit	whole contents are erased	sector/block level erasure uses: usb flash memories, cameras, video cards, etc.	byte level erasure is possible.

LOCALITY OF REFERENCE states that a program tends to access any small portion of logical address space at any point of time.
(logical address space is the set of all logical addresses generated by the CPU).

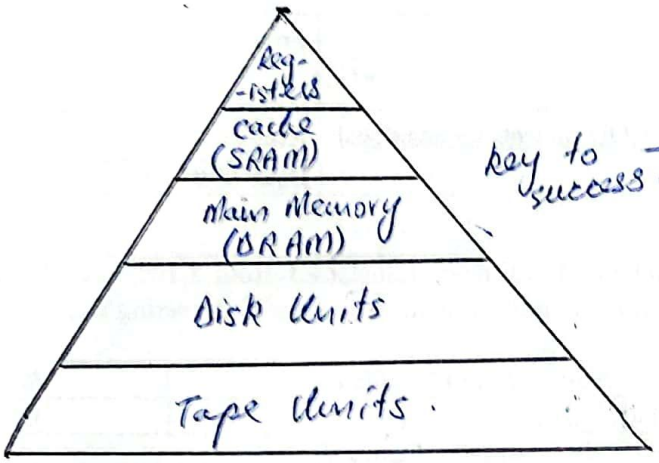
Temporal locality: It is the locality over time. Recently referenced items are likely to be referenced again in the near future.
e.g: loops and subroutines.

Spatial locality: It is the locality over space. Tending to access items whose addresses are near to one another.
e.g: array traversal.

Scanned with CamScanner

MEMORY HIERARCHY

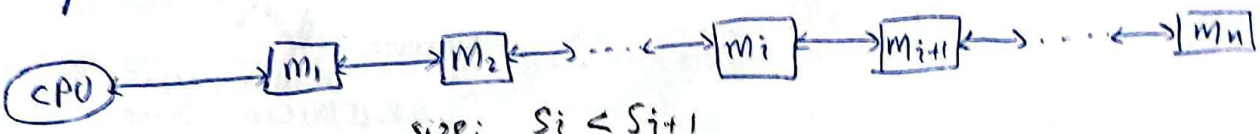
- There is a tradeoff between cost, capacity and speed of a memory.
  - Faster access time, greater cost per bit
  - Greater capacity, (if) smaller cost per bit
  - Greater capacity, slower access time.



key to success. →
- decreasing cost/bit
- increasing capacity
- increasing access time
- decreasing frequency of access of the memory by the processor.
- increasing UOT.

Rationale behind the design of a memory hierarchy:

- cost of a memory varies inversely with the access time.
  $t \propto \frac{1}{\sqrt{cost}}$
- It is therefore impossible to design an entire system using just one technology that offers both low cost and high operating speed (i.e. low access time). Hence a memory system is designed using different physical devices to achieve optimum performance.
- memory hierarchy reduces the speed gap between processor and memory system at reasonable cost.



size: $S_i < S_{i+1}$
access time: $t_i < t_{i+1}$
cost: $c_i > c_{i+1}$
unit of transfer: $x_i < x_{i+1}$
access frequency: $f_i > f_{i+1}$

Hit = a reference generated by CPU is found in M_i
Miss = a reference generated by CPU is not found in M_i

Scanned with CamScanner