**Computer input and output devices**

**How the processor communicates with these devices**

**Bus architecture**

* **Input devices**

An input device is any hardwarecomponentthat allows users to enter data and instructions (programs, commands, and user responses) into a computer. Examples include: Keyboard

Many people use a keyboard as one of their input devices. A **keyboard is an input device** that contains keys users press to enter data and instructions into a computer Desktop computer keyboards typically have from 101 to 105 keys.

All desktop computer keyboards have a typing area that includes the letters of the alphabet, numbers, punctuation marks, and other basic keys. Many desktop

* **Pointing Devices**

A **pointing device is an input device that allows a** user to control a pointer on the screen. In a graphical user interface,

A **pointer is a small symbol on** the screen whose location and shape change as a user moves a pointing device.

A pointing device can be used to move the insertion point; select text, graphics, and other objects; and click buttons, icons, links, and menu commands. **Examples:** touch pad, Track ball, touch screens and touch sensitive pads

* **Output Devices**

An **output device is any type of hardware** component that conveys information to one or more people. Commonly used output devices Include

1. Display devices
2. Printers
3. Speakers
4. Headphones
5. Data projectors
6. Interactive white boards
7. Force-feedback game controllers

**Interfacing I/O devices to the Memory, Processor and OS**

* How is a user I/O request transformed into a device command and communicated to the device?

E.g., file read/write, mouse movement, and keyboard stroke

* How is data actually transferred to or from a memory location?
* What is the role of the operating system?

**Communication between I/O devices and the OS**

For the I/O system to perform its functions right;

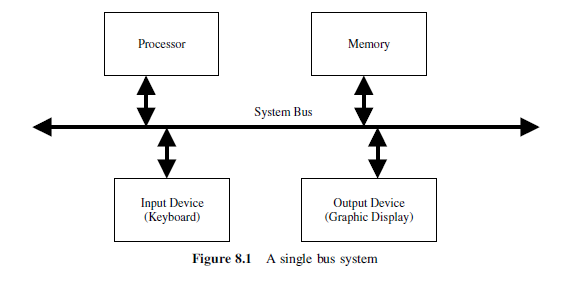
* The operating system must be able to communicate with I/O devices and to prevent user programs from accessing the I/O devices directly
* This communication between the I/O device and the processor is called the I/O protocols and requires a number of I/O registers

**Giving commands to I/O devices**

CPU must be able to address the device and to supply one or more commands methods for addressing the device

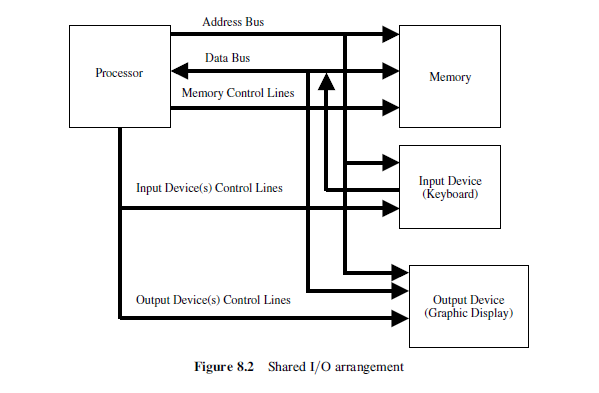
1. Single Bus System
2. Shared Bus system
3. Memory Mapped I/O
4. Programmed I/O

**Single Bus System**

****

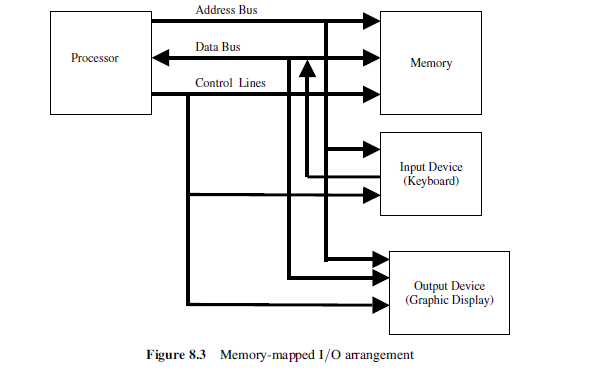
* **In the single bus System there are numbers of input and output register and each belonging to a single input device and same for the output devices and an address is assign to each of the above device and different from the address assign to memory**

**Shared I/O system**

****

* This arrangement, called shared I/O, is shown schematically, In this case, the address and data lines from the CPU can be shared between the memory and the I/O devices. A separate control line will have to be used. This is because of the need for executing input and output instructions. In a typical computer system, there exists more than one input and more than one output device. Therefore, there is a need to have address decoder circuitry for device identification. There is also a need for status registers for each input and output device.

**Memory Mapped I/O**

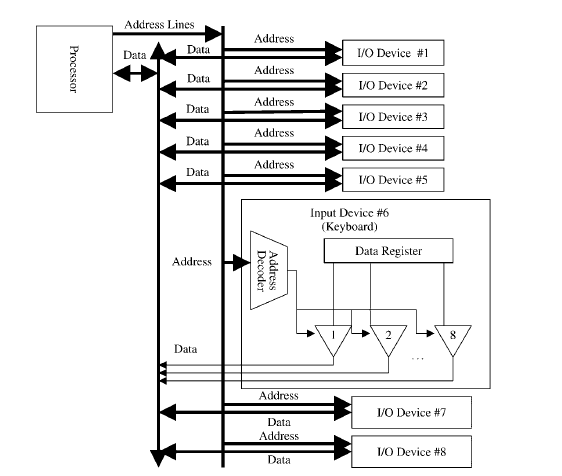
****

Compare to the shared I/O the memory mapped uses same control lines for the I/O and the processor and memory

**Memory-mapped I/O addressing**

* Portions of a program’s address space are assigned to I/O devices
* Reads and writes to these addresses are interpreted as commands to the device
* These memory addresses are not directly accessible by the user programs

**Programmed I/O**

****

In the programmed I/O the protocols has to be programmed in the form of routine that runs over the control of the CPU.

**Communication with the Processor**

**Two methods**

* Polling
* Device status bits are periodically checked to see if it is time for the next I/O operation
* Interrupt-driven I/O
* Device delivers interrupt to the CPU when it requires attention
* Interrupts are like exceptions except that they are not associated with any instruction
* CPU can check before starting a new instruction if an interrupt has been delivered
* Interrupt-handling: Can be vectored or can use a Cause register

**Transferring Data between Device and Memory**

* Direct Memory Access (DMA)
* Data is transferred directly from the device to memory (or vice versa)
* Processor is only involved in

1. Initiating the DMA transfer
2. Handling interrupt at the end of DMA transfer

**DMA**

* Implemented with special controller that transfers data between memory and I/O device independent of the processor
* Three steps in DMA transfers

1. Processor sets up the DMA transfer by supplying identity of device, operation to perform, memory address that is source or destination of data, number of bytes to be transferred
2. DMA controller starts the operation (authorizes for the bus, supplies address, reads or writes data), until the entire block is transferred
3. DMA controller interrupts the processor, which then takes the necessary actions

**Basic I/O system example**

**Computer Buses**

**The bus is a critical component of the computer**

* They are shared components that provide the paths for all parts of the computer to communicate with each other
* They can reduce the complexity of communications between computer components
* They can provide an easy way to evolve a computer system
* add components
* They can be a serious bottleneck if not designed and used appropriately
* As systems grow, they need to evolve hierarchically
* They can be parallel or serial
* They can have data widths larger than the computer word length

**Bus Basics**

* A bus generally contains a set of control lines and a set of data lines
* The control lines are used to signal requests and acknowledgments, and to indicate what type of information is on the data lines
* The data lines of the bus carry information between the source and the destination.
* This information may consist of data, complex commands, or addresses.
* Buses are traditionally classified into two:
* processor-memory buses
* I/O buses

**The processor-memory bus**

* A bus that connects processor and memory
* It is generally short and high speed
* Are interfaced to the memory system to maximize memory- processor bandwidth

**The I/O buses**

* Are lengthy
* can have many types of devices connected to them
* Do not typically interface directly to the memory but use either a processor-memory bus or a backplane bus to connect to memory

**Bus Design:** Things to Consider

**Some of the parameters to consider when designing a computer bus:**

* Accessibility
* Speed
* Reliability
* Interfacing
* Communication protocol
* Shareability
* length

**Bus Communication Protocol**

**Two types:**

* synchronous
* Asynchronous
* Synchronous ( e.g. processor- memory Bus)
* Includes a clock in the control lines and has a fixed protocol for communication that is relative to the clock

**Advantage**

* Involves very little logic and can run very fast

**Disadvantages**

* Every device communicating on the bus must use same clock rate
* To avoid clock skew, the bus lines cannot be long if they are fast.

**Asynchronous (e.g. I/O buses)**

* It is not clocked, so requires a handshaking protocol and additional control lines (ReadReq, Ack, DataRdy)

**Advantages**

* Can accommodate a wide range of devices and device speeds
* Can be lengthened without worrying about clock skew or synchronization problems

**Disadvantage**

* Relatively slow