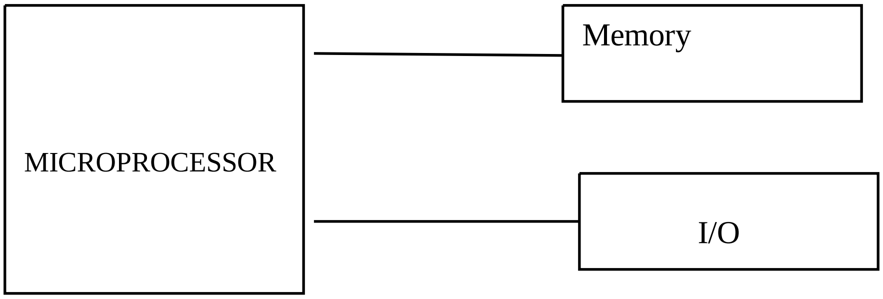
UNIT V

5.1 INPUT-OUTPUT ORGANIZATION

PERIPHERAL DEVICES

- A microprocessor is a multipurpose, programmable logic device that reads binary instructions from a storage device called memory accepts binary data as input and proc esses data according to those instructions and provides results as output.
- A typical programmable machine can be represented with three components: microprocessor, memory and I/O as shown



A Programmable devices

- Microprocessor, memory and I/O Components are work together or interact with each other to perform a given task; thus they compromise a system.
- The physical components of this system are called hardware.
- A set of instructions written for the microprocessor to perform a task is called a progra m.
- A group pf programs is called software.
- The machine (system) represented in figure can be programmed to turn traffic lights on and off, compute mathematical functions or keep track of a guidance system.

The microprocessor applications are classified primarily in two categories:

1. Reprogrammable systems
2. Embedded systems

Reprogrammable Systems

- In reprogrammable systems, such as microcomputers, the microprocessor is used for
- computing and data processing.

- These systems include general – purpose microprocessors capable of handling large data,
- mass storage devices and peripherals.

Embedded Systems

- In embedded systems, the microprocessor is a part of a final product and is available for
- reprogramming to the end user.
- A copying machine is a typical example of an embedded system.

BINARY DIGITS

- The microprocessor operates in binary digits 0 and 1, also known as bits. Bit is an
- abbreviation for term binary digit.
- These digits are represented in terms of electrical voltages in the machine
    - 0 represents low voltage level and
    - 1 represents high voltage level.

A MICROPROCESSOR AS A PROGRAMMABLE DEVICE

- The microprocessor is programmable that can be instructed to perform given tasks within
- its capacity.
- The engineers designing a microprocessor determine a set of tasks the microprocessor should perform
- and design the necessary logic circuits and provide the user with a list of instructions the
- processor will understand.
- For example, an instruction for adding two numbers may look like a group of eight binary
- digits such as 1000 0000.
- These instructions are simply a pattern of 0s and 1s.
- The user (programmer) selects instructions from the list and determines the sequence of execution for a
- given task.
- These instructions are entered or stored in storage, called memory, which can be read
- by the microprocessor.

MEMORY

- Memory is like the pages of a notebook with space for a fixed number of binary numbers
- on each line.
- However, these pages are generally made of semiconductor material.
- Typically, each line is an 8-bit register that can store 8 binary bits, and several of these registers are
- arranged in a sequence called memory.

- These registers are always grouped together in powers of two.

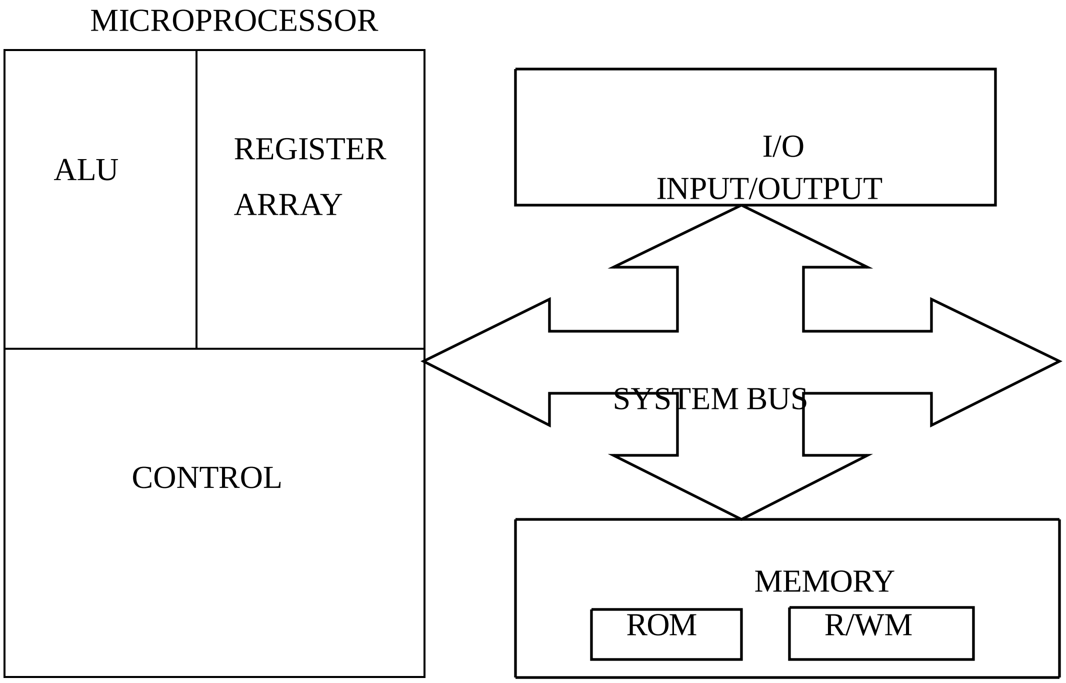
INPUT/OUTPUT

- The user can enter instructions and data into memory through devices such as keyboard or simple switches.
- These devices are called input devices.
- The microprocessor reads the instruction from the memory and processes the data according to those instructions.
- The results can be displayed by a device such as seven segment LEDS or printed by a printer.
- These devices are called output devices.

MICROPROCESSOR AS A CPU

- We can also view the microprocessor as a primary component of a computer.
- The CPU is the primary and central player in communicating with devices such as
- memory, input and output.
- The timing of the communicating process is controlled by the group of circuits called the control unit.
- The CPU contains various registers to store data.
- The arithmetic logic unit (ALU) to perform arithmetic and logical operations, instruction decoders, counters and control lines.
- The CPU reads instruction from memory and performs the task specified.
- It communicates with input/output devices either to accept or to send data.
- With the advent of the integrated circuit technology it became possible to built the CPU on a single chip; this came to be known as microprocessor.

ORGANIZATION OF A MICROPROCESSOR-BASED SYSTEM



Microprocessor-Based Systems with Bus Architecture

MICROPROCESSOR

- The microprocessor is a semiconductor device consisting of electronic logic circuits
- manufactured by using either a large-scale integration (LSI) or very-large-scale integration
- (VCLSI) technique.
- The microprocessor is capable of performing various computing functions and making decisions to change the sequence of program execution,
- in large computers, a CPU implemented on one or more circuit boards performs
- these computing functions. The microprocessor is in many ways similar to the CPU.
- The microprocessor can be divided into three segments for the sake clarity.

Arithmetic/logic unit (ALU)

- Register array
- Control unit.

Arithmetic/Logic Unit

- This is the area of the microprocessor where various computing functions are performe d

- on data.
- The ALU unit performs arithmetic operations as addition and subtraction, and such logic
- operations as AND, OR and exclusive OR.
- Results are stored either in register or in memory.

Register Array

- This area of the microprocessor consists of various registers. These registers are primarily used to store data temporarily during his execution of a program.
- Some of the registers are accessible to the user through instructions.

Control Unit

- The control unit provides the necessary timing and control signals to all the operations in the microcomputers.
- It controls the flow of data between the microprocessor and memory and peripherals.

5.1.1 INPUT – OUTPUT INTERFACE

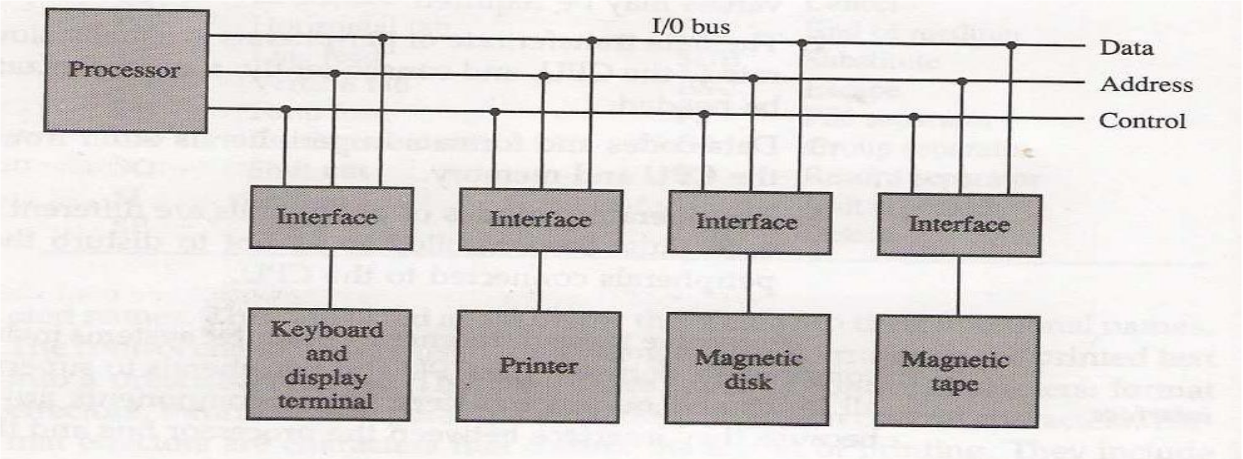
- This provides a method for transferring information between internal storage and external I/O devices.
- **Communication links** are used to interface between peripherals and CPU. They resolve the differences between computer and peripheral.

The differences are

- Peripherals are electromechanical and electromagnetic devices where CPU and memory are electronic devices.
- Data transfer rate of peripherals are slower than transfer rate of CPU. So synchronization is necessary.
- Data codes and formats in peripherals are differ from word format in CPU and memory.
- Operating modes of peripherals are different from each other and it should be controlled so that it will not disturb operations of other peripherals.
- To resolve these differences, a special hardware component is included in computer system between CPU and peripherals to supervise and synchronize all input and output transfers. This component are called **interface units**.

5.1.2 I/O BUS AND INTERFACE MODULES

- The I/O bus consists of data lines, address lines and control lines.
- Each peripheral devices are associated with its interface units.
- Each peripheral has its own controller.



Connection of I/O bus to input-output devices

- The interface Decodes the address and control received from the I/O bus
- Interprets it and provides signal for peripheral controller
- Synchronize the data flow and supervises the transfer between CPU and peripherals.
- The I/O bus from the processor is attached to all the peripheral devices.
- To communicate with a particular device,
- Processor will place a device address on the address line.
- The address decoder in the interface, monitors the address line.
- When the interface detects its own address, it activates the path between the bus line and the device.
- Other interfaces will disable their path.
- Processor will place a function code in the control lines.
- The interface selected responds to the function code and executes it.
- The function code is called as I/O command.

Four types of I/O Commands :-

1. **Control Command** : Activates the peripheral and inform it what to do. This depends on the peripheral and each of it receives its own distinguished sequence of control commands based on its mode of operation.
2. **Status Command**: Test various status conditions in the interface and the peripherals. During transfer, errors may occur which are detected by the interface. These errors are designated by setting bits in status register which is read by the processor at certain intervals.
3. **Data Output Command**: Makes the interface to respond by transferring data from the bus into one of its register. The processor issues data output command. The interface responds to the address and command and transfers the information from the data lines in the bus to its buffer register. The interface then communicates with the peripheral devices and sends the data to it.

4. **Data Input Command:** This is opposite to data output command. The interface receives an item of data from the peripheral device and places it in its buffer register. The processor checks if data are available by means of a status command and issues a data input command. The interface places the data on the data lines for the processor.

5.1.3 I/O VERSUS MEMORY BUS

- Memory bus is used for communication between processor and the memory unit.
- It contains data, address and read/write control lines.
- Three ways that computer bus can be used to communicate with memory and I/O :
- Use two separate buses – one for memory and other for I/O.
- Use one common bus for both memory and I/O but have separate control lines for each.
- Use one common bus for memory and I/O with common control lines.
- In the first method, the computer has independent sets of data, address and control buses, one for accessing memory and the other for I/O. This is done in computers that provide a separate **I/O processor (IOP)** in addition to the CPU.
- The **purpose of IOP** is to provide an independent pathway for the transfer of information between external devices and internal memory. The I/O processor is sometimes called a **data channel**.

5.1.4 ISOLATED VERSUS MEMORY-MAPPED I/O

- Many computers use one common bus to transfer information between memory or I/O and the CPU.The CPU specifies whether the address on the address lines is for a memory word or for an interface register by enabling one of two possible read or write lines.
- The I/O read and I/O write control lines are enabled during an I/O transfer.The memory read and memory write control lines are enables during memory transfer.
- This configuration isolates all I/O interface addresses from the memory addresses and referred as **isolated I/O method** for assigning addresses in a common bus.
- Certain computers employ only one set of read and write signals and do not distinguish between memory and I/O addresses.
- This configuaration is referred to as **memory-mapped I/O**.

- There are no specific inputs or output instructions. The CPU can manipulate I/O data residing in interface registers with the same instruction that are used to manipulate memory words.
- Computer with memory-mapped I/O uses memory-type instructions to access I/O data.
- It allows computer to use the same instruction for either I/O transfer or memory transfer.
- The **advantage** is that the load and store instructions used for reading and writing from memory and can be used to input and output data from I/O registers.

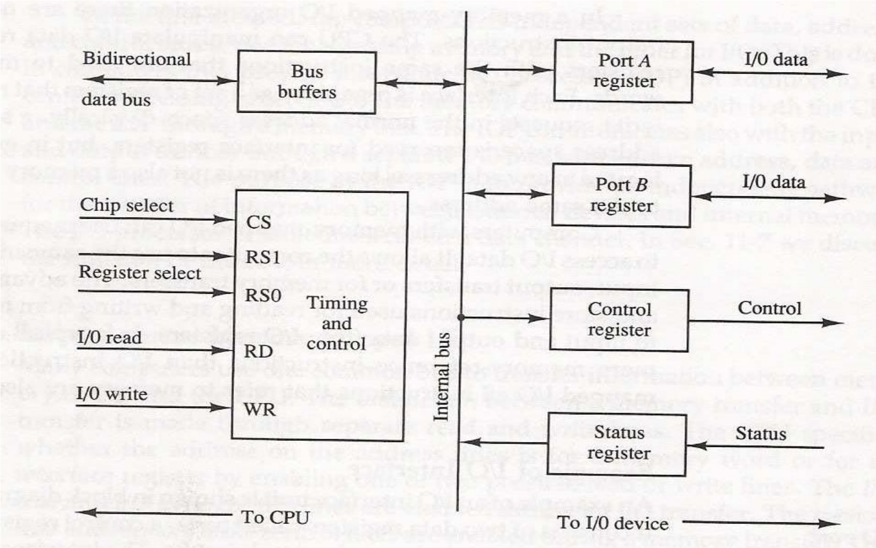
5.1.5 EXAMPLE OF I/O INTERFACE

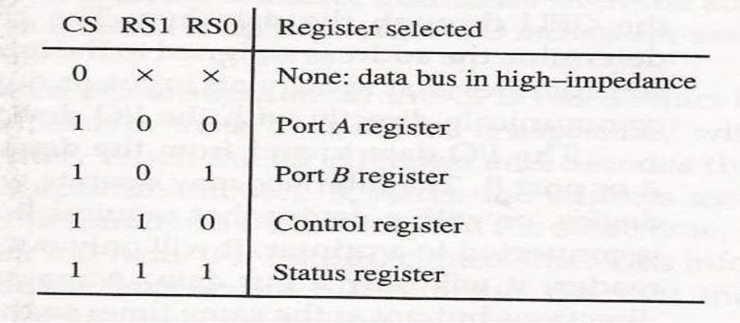
It consists of

1. Two data registers called ports - The I/O data to and from the device can be transferred into either port A or port B.
2. A Control register – receives control information from CPU. By loading appropriate bits into control register, the interface and the I/O device attached to it can be placed in a variety of operating modes.
3. A Status register – The bits in status register are used for status conditions and for recording errors that may occur during the data transfer.
4. Bus buffers
5. Timing and control circuits

- The interface communicates with the CPU through the data bus.
- The Chip select and register select inputs determine the address assigned to the interface
- The I/O read and write are two control lines that specify an input or output respectively.
- The four registers communicate directly with the I/O device attached to the interface.
- The interface operates with an output device or with an input device.
- A circuit is provided to detect the address assigned to the interface registers.
- This circuit enables the chip select (CS) input when the interface is selected by the address bus.
- The two register select inputs RS1 and RS2 are connected to two least significant lines of the address bus.

- These two inputs select one of the four registers in the interface.
- The data transfer between the selected registers and CPU is done via data bus when an I/O read or write signal is enabled.





CS	RS1	RS0	Register selected
0	×	×	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

Example of I/O Interface units

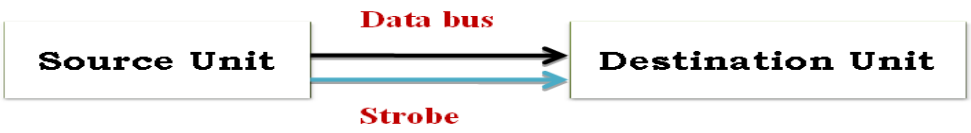
5.1.6 ASYNCHRONOUS DATA TRANSFER

- The internal operations in a digital system are synchronized by means of clock pulses supplied by a common pulse generator.
- Clock pulses are applied to all registers within a unit and all data transfers among internal registers occur simultaneously during the occurrence of a clock pulse.

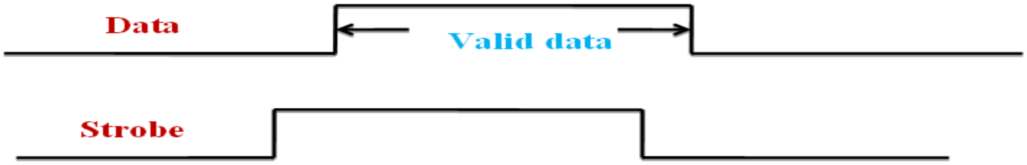
- Two units, such as a CPU and an I/O interface, are designed independently of each other. If the registers in the interface share a common clock with the CPU registers, the transfer between the two units is said to be synchronous.
- In most cases, the internal timing in each unit is independent from the other in that each uses its own private clock for internal registers. In that case, the two units are said to be asynchronous to each other.
- Asynchronous data transfer between two independent units requires that control signals be transmitted between the communicating units to indicate the time at which data is being transmitted.
- One way of achieving this is by means of a strobe pulse supplied by one of the units to indicate to the other unit when the transfer has to occur.
- Another method commonly used is to accompany each data item being transferred with a control signal that indicates the presence of data in the bus.
- The unit receiving the data item responds with another control signal to acknowledge receipt of the data. This type of agreement between two independent units is referred to as handshaking.
- We have to specify the asynchronous transfer between two independent units by means of a timing diagram that shows the timing relationship that must exist between the control signals and the data in the buses.
- The sequence of control during an asynchronous transfer depends on whether the transfer is initiated by the source (transmitting) unit or by the destination (receiving) unit.

5.1.7 STROBE CONTROL

- A Strobe Control employs a single control line to time each transfer
- The strobe may be activated by either the source or the destination unit



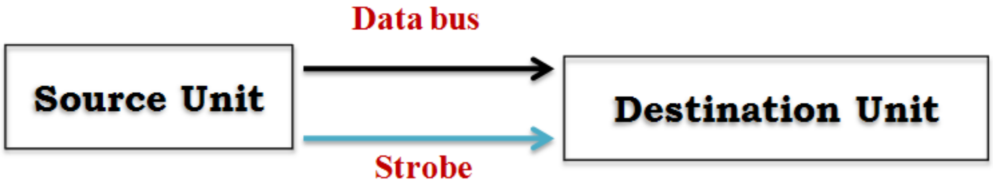
Block Diagram



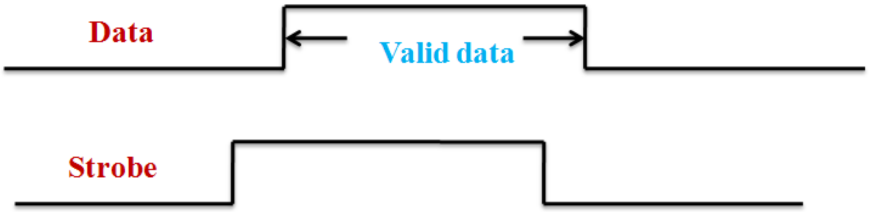
Timing Diagram

Source-Initiated Strobe for Data Transfer

- The data bus carries the binary information from source to destination.
- The strobe is a single line that informs the destination unit when a valid data word is available in the bus.
- The source unit places data on the bus and after a short delay, it activates the strobe pulse.
- The information on the data bus and strobe signal remain active to allow destination unit to receive the data.
- The source removes data from the bus a brief period after it disables its strobe pulse.
- Once the strobe pulse is disabled , the data bus does not contain valid data.



Block Diagram



Timing Diagram

Destination-Initiated Strobe for Data Transfer

- Here the destination unit initiated the data transfer and it activates the strobe pulse, informing source unit to provide data.
- Now the source unit places the information in data bus.
- The falling edge of the strobe pulse is used to trigger the destination register.
- Then the destination unit disables the strobe and the source removes the data after a predefined time interval.

5.1.8 HANDSHAKING

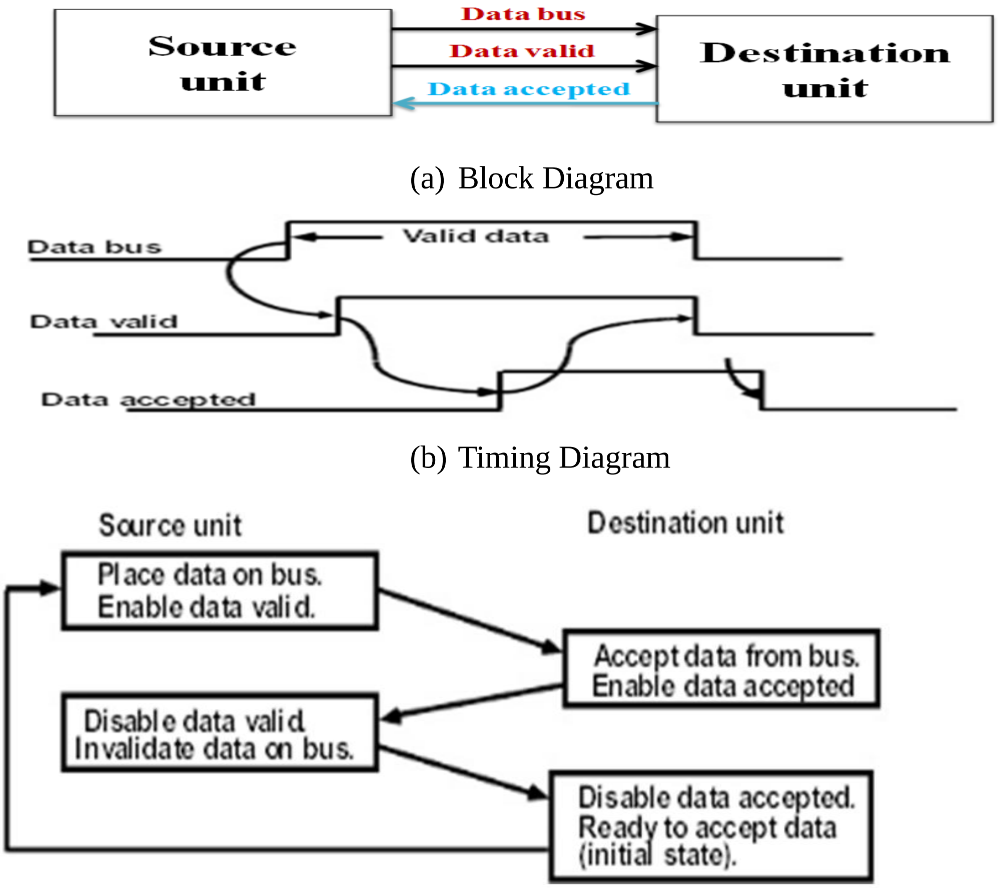
DISADVANTAGES OF STROBE CONTROL

a. The source unit that initiates the transfer has no way of knowing whether the destination unit has actually received data

b. The destination unit that initiates the transfer no way of knowing whether the source has actually placed the data on the bus

- To solve this problem, the HANDSHAKE method introduces a second control signal to provide a
  Replay to the unit that initiates the transfer
- One control line from source to destination is used by the source unit to inform the destination unit
  whether the valid data in the bus.
- Another control line from destination to source is used by the destination unit to inform the source unit
  whether it can accept data.

Source-Initiated transfer using Handshaking



(a) Block Diagram

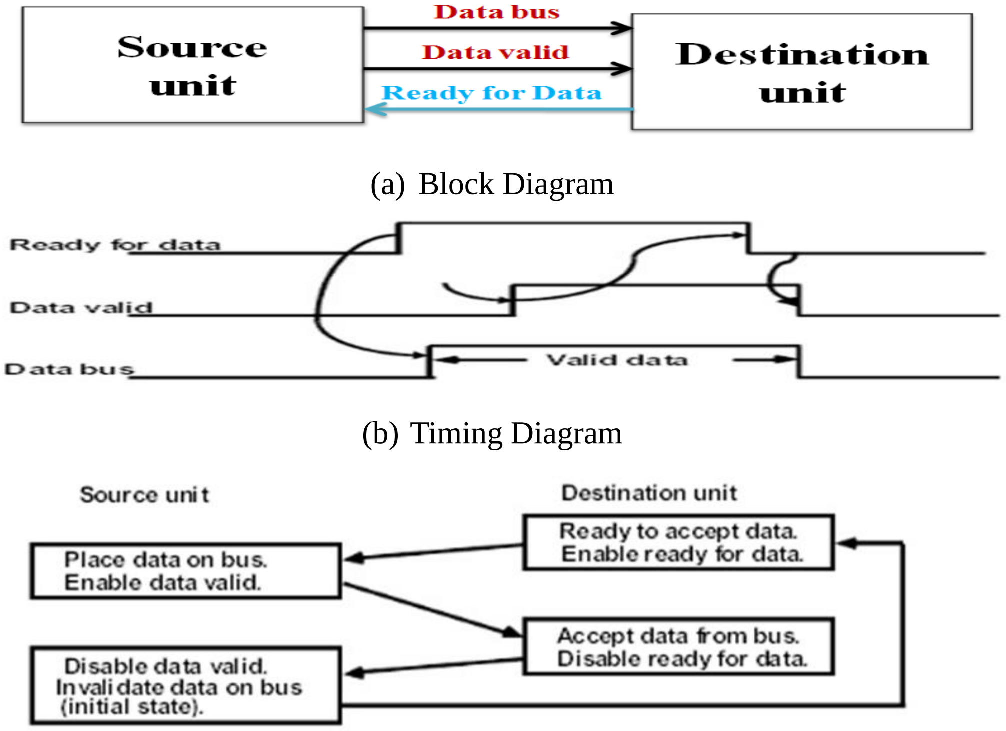
(b) Timing Diagram

(c) Sequence of Events

Two handshaking lines are

- Data valid : generated by the source unit
- Data accepted : generated by the destination unit
- The timing diagram shows the exchange of signals between the two units.
- The sequence of events shows four possible states in the system.
- The source unit initiates the transfer by placing the data on the bus and enables its data valid signal.
- The data accepted signal is activated by the destination unit after it accepts the data.
- The source unit then disables its data valid signal which invalidates the data on the bus.
- The destination unit disables its data accepted signal and the system goes to initial state.

Destination-Initiated transfer using Handshaking


Source unit
Data bus
Data valid
Ready for Data
Destination unit
Ready for data
Data valid
Data bus
Valid data
Source unit
Destination unit
Ready to accept data. Enable ready for data.
Place data on bus. Enable data valid.
Accept data from bus. Disable ready for data.
Disable data valid. Invalidate data on bus (initial state).

(a) Block Diagram

(b) Timing Diagram

(c) Sequence of Events

- The signal generated by the destination unit is “ready for data”.
- The source unit does not place any data on the bus until it receives the ready for data signal from the destination.

- The only difference between source-initiated and destination-initiated transfer is in their choice of initial state.

Timeout mechanism

- Handshaking scheme provides high degree of flexibility and reliability.
- If one unit is faulty, the data transfer is not completed which can be detected by means of timeout mechanism – this produces an alarm if the data transfer is not completed within in a predetermined time.
- This is implemented by means of an internal clock that starts counting time when the unit enables one of its handshaking control signal. If there is no response within a predetermined time then the unit assumes there is an error and produces an alarms.