

Input/Output (I/O)

This section describes the basic input output techniques used by microcomputers to transfer data between the microcomputer and external devices.

- The I/O devices connected to a microcomputer system provide an efficient means of communication between the computer and outside world
- The characteristics of the I/O devices are normally different from those of the microcomputer. (speed of operation, word length)
- To make the characteristics of I/O devices compatible with those of the microcomputer, interface hardware circuitry between the microcomputer and I/O devices is necessary
- There are **two types of I/O devices** i.e. **physical I/O and logical I/O**. When the microcomputer has no Operating System the user must work directly with physical I/O devices and provide detailed I/O design.

Logical I/O

- For a microcomputer with an operating system, the user works with **virtual I/O** devices and he/she does not have to be familiar with the characteristics of the physical I/O devices
- The user performs data transfer between the microcomputer and the physical I/O devices **indirectly by calling the I/O routines** provided by the operating systems using virtual I/O instructions

Physical I/O

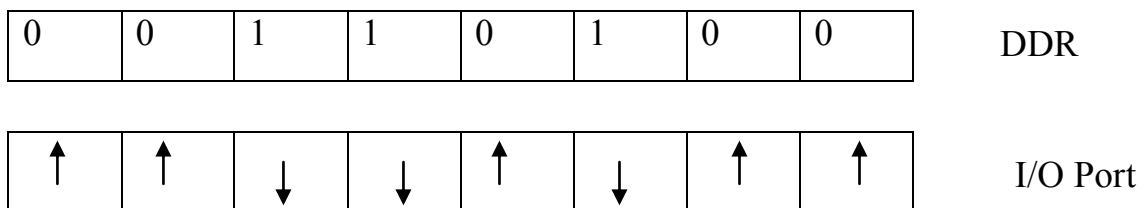
When the microcomputer has no operating system, the user must work directly with physical I/O devices. There are three ways:

1. Programmed I/O
2. Interrupt driven I/O
3. Direct memory access (DMA)

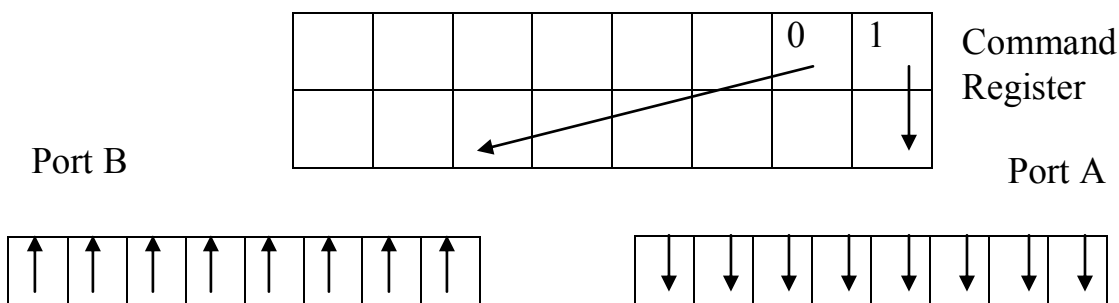
Programmed I/O

- The microcomputer executes a program to communicate with an external device **via a register** called the I/O port for programmed I/O
- I/O ports are usually two types.

- For one type, each bit in the port can be individually configured as either input or output
- For the other type, all bits in the port can be set up as either all parallel inputs or out bits
- Each port can be configured as an input or output port by another register called the **command or data-direction register (DDR)**.
- **The port data register contains the actual input or output data**
- Each bit in the port can usually be set up as an input by respectively writing a 0 or a 1 in the corresponding bit in the data direction register
- **A '1' written to a particular bit in DDR enables the output buffer while a '0' enables the input buffer**
- If DDR contains $(34)_{16}$ the corresponding port is defined as



- For parallel I/O, there is only one register, known as **command register, for all ports.**
- A particular bit in the command register configures all bits in a port as either inputs or outputs



I/O ports are addressed using either standard I/O or memory-mapped I/O techniques.

Standard I/O

- For an 8-bit microprocessor, an **8-bit address is typically used for each I/O port, although it uses 16 bit address for memory location.**
- The microprocessor outputs a high on $\text{IO}/\overline{\text{M}}$ control pin to indicate to memory and I/O chips that an I/O operation is taking place.
- **IN AL, PortA** and **OUT PortA, AL** are example of standard I/O instructions

Memory-Mapped I/O

- In memory-mapped I/O, **MSB** of the address is used to **distinguish between I/O and memory.**
- This reduces the microprocessor's addressing memory by 50%.
- **MOV AL, START** and **MOV START, AL** are example of memory-mapped I/O.

Unconditional and conditional programmed I/O

- The microprocessor can send data to an external device at **any time** during unconditional I/O. The external device **must always be ready** for data transfer
- In conditional I/O, the microprocessor outputs data to an external device **via handshaking**
- The microprocessor input the status of the external device to determine whether the device is ready for the data transfer.

Interrupt driven I/O

- Interrupt I/O is a device initiated I/O transfer. The external device is connected to a pin called, the **interrupt (INT) pin** on the microprocessor chip.
- When the device needs an I/O transfer with the microcomputer, it activates the interrupt pin of the microprocessor chip.
- The microprocessor usually completes the current instruction and saves at least the contents of the current program counter on the stack.
- The microprocessor then automatically loads an address into the program counter to branch to a subroutine like program called the **interrupt service routine**. This program is **written by the user**.

- The **last instruction** in the service routine is a **RETURN**, which normally loads **the return address in the program counter**.
- Then the microprocessor continues to execute the main program.

Interrupt types

There are typically three types of interrupts:

1. External Interrupts
2. Trap or Internal Interrupts
3. Software Interrupts

External Interrupts

External interrupts are initiated through the microprocessor's interrupt pin by external devices. External interrupts can further be divided into two types:

- a) Maskable Interrupts and
- b) Non-maskable interrupts

A maskable interrupt is enabled or disabled by executing instructions such as **EI** or **DI**. If the microcomputer's interrupt is disabled, the microprocessor ignores the maskable interrupt. Some microprocessor, such as 8086, have an interrupt flag bit in the processor's status register. **When the interrupt is disabled, the interrupt flag bit is 1, so no maskable interrupts are recognized by the processor.** The interrupt flag bit resets to zero when the interrupt is enabled.

The non-maskable interrupt has a **higher priority** than the maskable interrupt. If both maskable and non-maskable interrupts are activated at the same time the processor will **service the non-maskable interrupt first**. The non-maskable interrupt is usually used as **power failure interrupt**.

Maskable handshake interrupt

- This interrupt is usually implemented by using two pins – **INTR** and **INTA**.
- When the interrupt pin is activated by an external device, the processor completes the current instruction, saves at least the current program counter onto the stack, and generates an interrupt acknowledge (**INTA**).
- In response to interrupt acknowledge, the external device provides an instruction, such as **CALL**, using external hardware on the data bus of the microcomputer.

- This instruction is then read and executed by the microcomputer to branch to the desired service routine.

Internal Interrupt

Internal interrupts, or **traps**, are activated internally by exceptional conditions such as overflow, division by zero, or execution of an illegal opcode. Traps are handled in the same way as external interrupts. The user writes a service routine to take corrective measures and provide an indication to inform the user that an exceptional condition has occurred.

Software Interrupt or System Call

When interrupt instructions are executed, the microprocessor is interrupted and serviced similarly to external or internal interrupts. Software interrupts are normally used to call the operating system.

Interrupt Address Vector

The technique used to **find the starting address of the interrupt service routine is the use of an interrupt address vector**. In some microprocessors, the manufacturers define the fixed starting address for each interrupt. Other manufacturers use an indirect approach by defining fixed locations where the interrupt address vector is stored.

Direct Memory Access (DMA)

Direct Memory Access is a technique that transfers data between a microcomputer's memory and an I/O device without involving the microprocessor. DMA is widely used in transferring large blocks of data between a peripheral device and the microcomputer's memory. The DMA technique uses a **DMA controller chip** for data transfer operation. The operation is summarized as follows:

- The I/O devices **request DMA operation** via the **DMA request lines** of the controller chip
- The controller chip **activates** the microprocessor **hold pin**, requesting the CPU to release the bus.
- The **processor sends HLDA (hold acknowledgement)** back to the DMA controller, indicating that the bus is disabled.

- The DMA controller **places the current value of its internal registers**, such as the address register and counter, on the **system bus** and sends a **DMA acknowledge** to the peripheral device.
- The DMA controller completes the DMA transfer and releases the buses.

There are three basic types of DMA:

1. Block transfer
2. Cycle stealing and
3. Interleaved DMA

Block Transfer

The DMA controller chip takes the bus from the microcomputer to transfer data between the memory and I/O device. The microprocessor has no access to the bus until the transfer is completed. During this time, the microprocessor can perform internal operations that do not need the bus. Using this method block of data can be transferred.

Cycle Stealing

Typically the microprocessor clock is enabled by ANDing an INHIBIT signal with the system clock. The DMA controller controls the INHIBIT line. During normal operation, the INHIBIT line is HIGH, providing the microprocessor clock. When DMA operation is desired, the controller makes the INHIBIT line low for the clock cycle. The microprocessor is then stopped completely for one cycle. Data transfer between the memory and I/O takes place during this cycle. This method is called cycle stealing.

Interleaved DMA

The DMA controller chip takes over the system bus when the microprocessor is not using it. For example, the microprocessor does not use the bus while incrementing the program counter or perform an ALU operation. The DMA controller chip identifies these cycles and allows the transfers of data between the memory and I/O device.

DMA controller

The DMA controller chip usually has at least three registers normally selected by the **controller's register select (RS)** line: **an address register, a terminal count register and a status register**. Both the address and terminal count registers are initialized by the microprocessor. The address register contains the starting address of the data to be transferred and the terminal count register contains the desired block to be transferred. The status register contains the information such as the completion of DMA transfer.