CH-7

7.1 External devices →

External devices that provide a means of exchanging data between the external enviroment & the computer. An external device attaches to the computer by a link to an I/O module. The link is used to exchange control, status, & data between I/O module & the external device. An external device is called as peripheral device.

→

Fig 7.1 Generic model of an I/O module.

(A) classification of External devices →

① Human readable → suitable for communicating with the computer user.
Ex → printer, keyboard, screen

② Machine readable → suitable for communicating with equipment.
Ex → magnetic disk & tape systems, & sensors & actuators (monitoring and control)

③ Communicatim → suitable for communicating with remote devices.
Ex → modem, NIC (Network Interface card)

Generic Model of an I/O Module

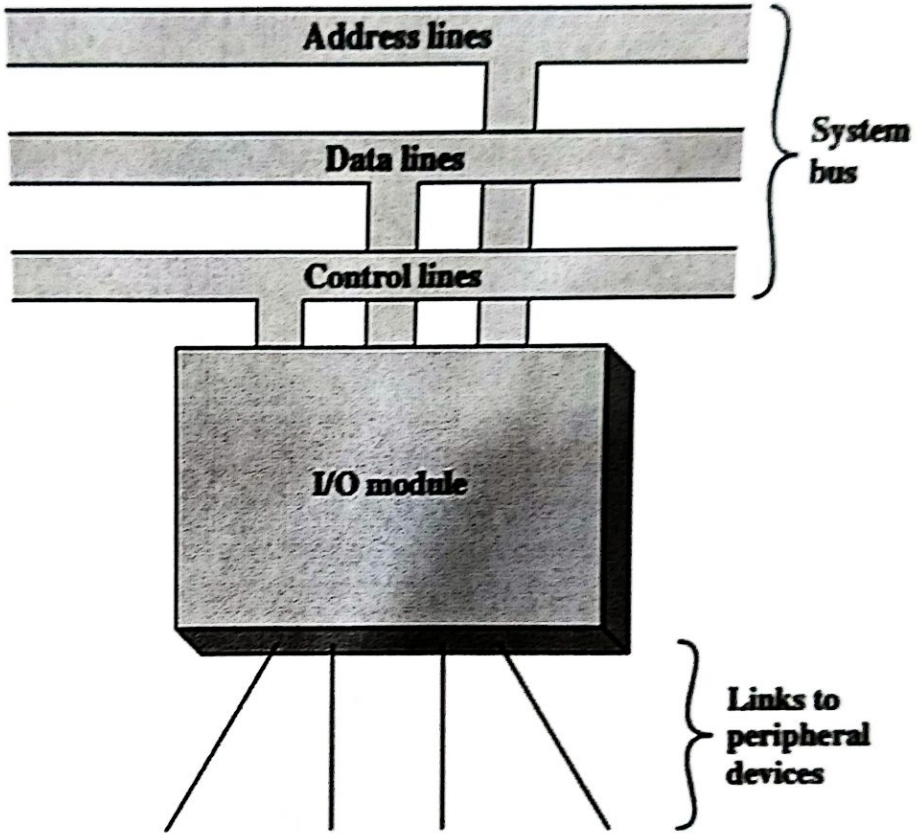


Fig.7.1: Generic Model of an I/O Module

(B) Nature of external devices:

* The interfacing to the I/O module is in the form of control, data, & status signals

① Control signal → It determine the function that the device will perform such as send data to the I/O module (INPUT or READ), accept data from I/O module (OUTPUT or WRITE), report status or perform some control function particular to the device (Ex - position a disk head)

② Data → Data are in the form of a set of bits to be sent to or received from the I/O module.

③ status signal → It indecate the state of the device.
Ex → READY / NOT-READY to show wheather the device is ready for data transfer.

* The essential blocks external device:

① Control logic → control logic associated with the device, controls the device's operation in response to direction from the I/O module.

(2) Transducer → It converts data from electrical to other forms of energy during output & from other forms to electrical during input.

(3) Buffer → Buffer can hold data temporarily during transmission between I/O module & external environ

Fig 7.2: Block diagram of an External device

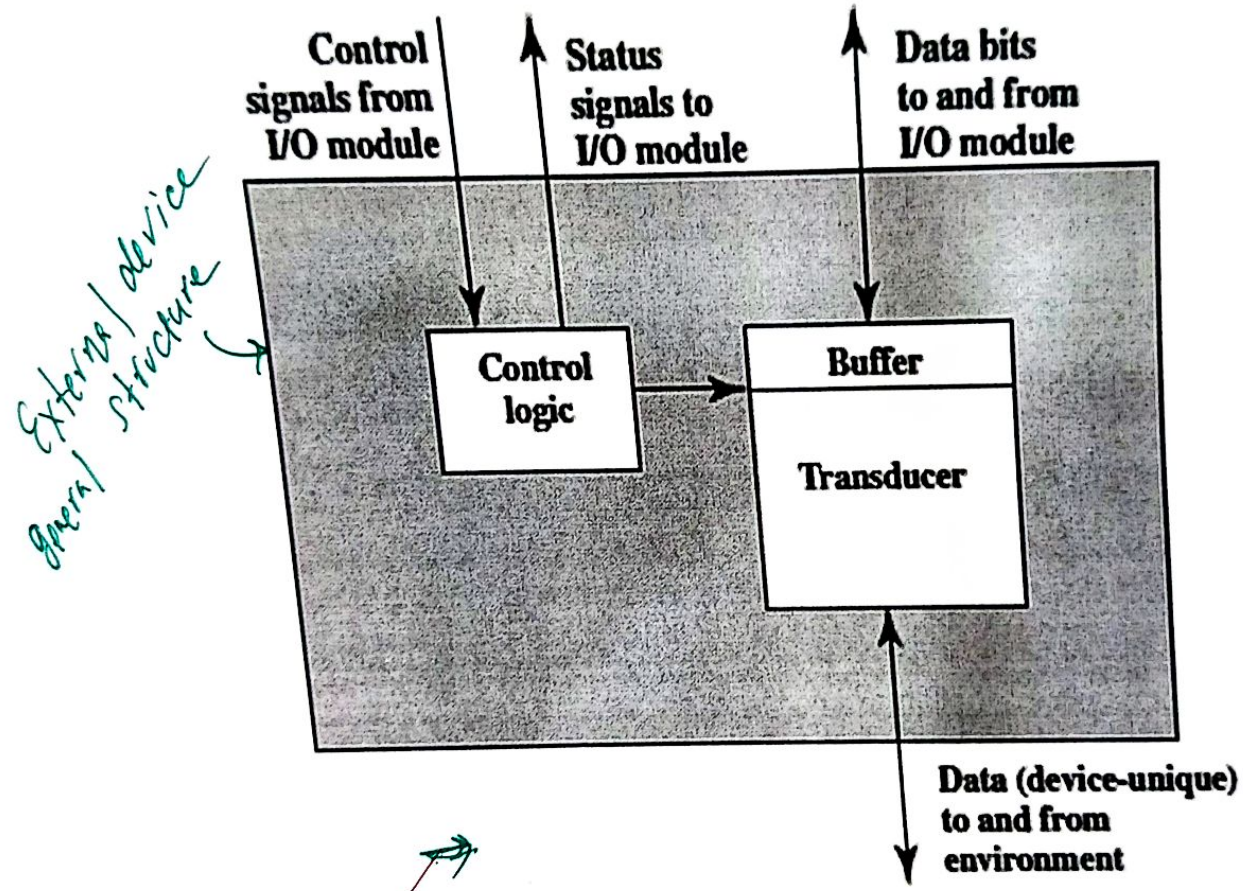
keyboard/Monitor →

→ When the user depresses a key, this generate an electronic signal, that is interpreted by the transducer in the keyboard & translated into the bit pattern of the corresponding IRA (International Reference Alphabet) code.

→ This bit pattern is then transmitted to the I/O module in the computer.

→ At the computer, the text can be stored in the same IRA code.

Block Diagram of an External Device



External device general structure

7.2

→ On the output, IRA code characters are transmitted to an external device from the I/O module. The transducer at the device interprets this code & sends the required electronic signal to the output device.

Disk Drive →

A disk drive contains electronics for exchanging data, control, & status signals with an I/O module plus the electronics for controlling the disk read/write mechanism.

7.2 I/O Module:

(A) I/O Module functions →

The major functions of I/O module:

1. control & timing
2. Processor communicatim
3. Device communicatim
4. Data buffering
5. Error detectim.

① Control & Timing →

I/O module are designed to manage the data transactions between the internal system (main memory & system bus) & peripheral devices.

② Processor communicatim →

This critical functim of an I/O module involves as:

(i) Command decodng → Receive & decode commands sent from processor.

(ii) Data Exchange → Exchngge data between peripherals, processors, & the main memory.

(iii) Status reporting → Communicate the status of peripherals to the processor.

(iv) Address decoding → Organizes each of the peripherals connected to the I/O module by managing their unique address.

(3) Device communication →

I/O module can facilitate communication between connected peripheral devices.

(4) Data Buffering →

With data buffering, I/O module can manage the transfer speed of data sent by the processor to peripheral devices. This compensates for the latency of peripheral devices.

(5) Error detecting →

I/O modules have the ability to detect errors & report them to the CPU. One way that I/O modules detect errors is with the parity bit method.

(B) I/O module structure →

I/O module manage the communication between a CPU & external devices, such as transfer of data, the management of power loads, & the control of machine functions. The structure of I/O module is shown in fig 7.3.

(C) I/O Techniques:

For I/O operations, there are 3-techniques are used:

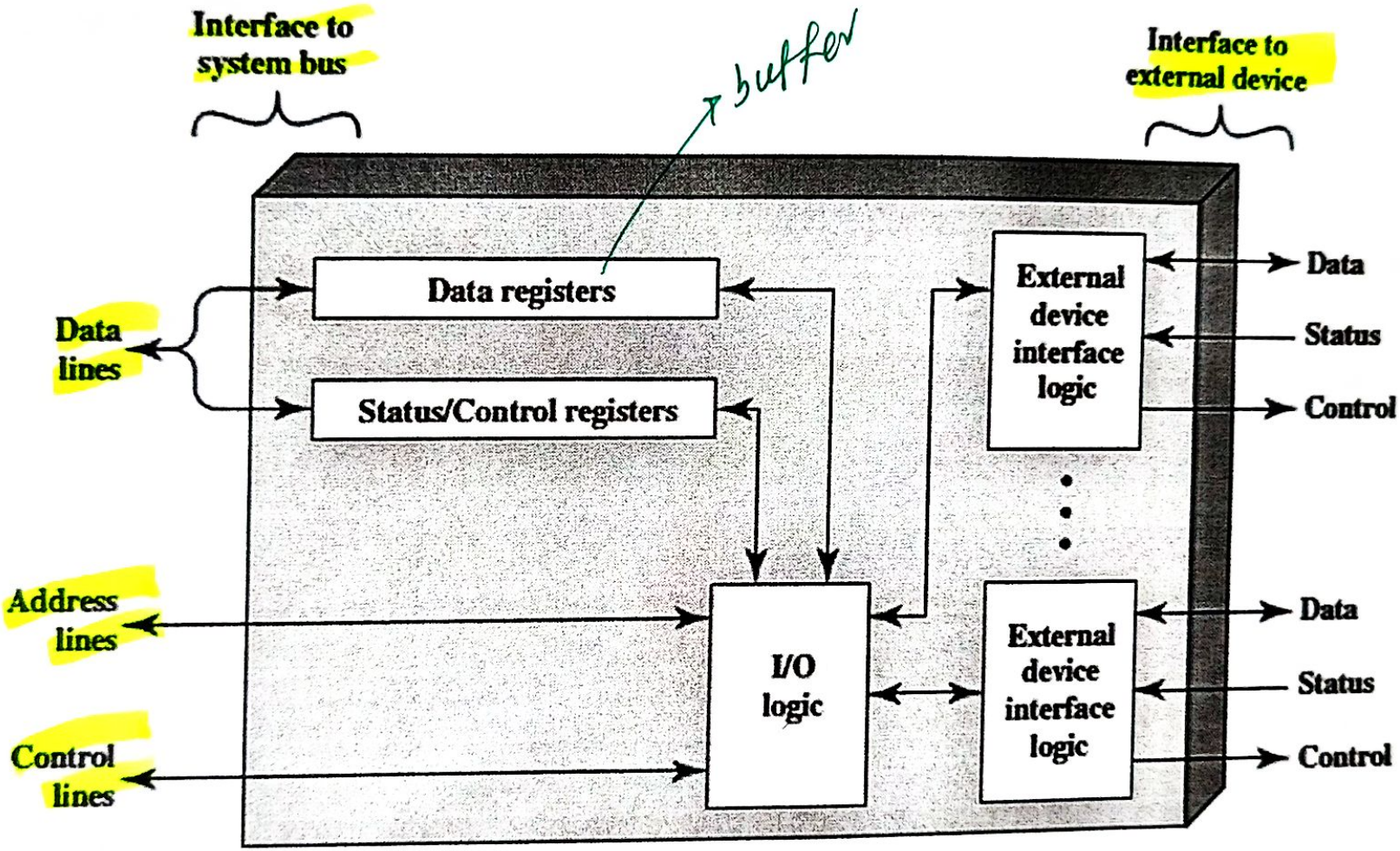
1. Programmed I/O
2. Interrupt-driven I/O
3. Direct Memory Access (DMA)

7.3

1. Programmed I/O

With programmed I/O, data are exchanged between processor & the I/O module. The processor executes a program that gives it direct control of the I/O operation including sensing device status, sending a read or write command, & transferring the data. When the processor issues a command to the I/O module, processor must wait until the I/O operation is complete. If the processor is faster than the I/O module, this is waste of processor time.

I/O MODULE STRUCTURE



F.y 7.3 Block diagram of I/O structure.

(i) I/O commands:

There are 4-types of I/O commands that an I/O module receives from processor. The I/O commands that the processor issues to an I/O module to execute the instructions.

① **control:** It uses to activate a peripheral & tell it what to do.

Ex → a Magnetic-tap unit may be instructed to rewind or to move forward one record.

② **Test:** It uses to test various status conditions associated with an I/O module & it's peripherals.

③ **Read:** It causes the I/O module to obtain an item of data from the peripheral & place it in an internal buffer.

④ **Write:** It causes the I/O module to take an item of data from the data bus & subsequently transmit that data item to the peripheral.

(ii) I/O Instructions →

I/O-related instructions are instructions that the processor fetches from memory.

When the processor, main memory, & I/O share a common bus, two modes of addressing are possible:

① Memory-mapped I/O
② Isolated I/O

① Memory-mapped I/O → With memory-mapped I/O, there is a single address space for memory locations & I/O devices. With memory-mapped I/O, a single read line & a single write line are needed on the bus.

Ex → With 10 address lines, total of $2^{10} = 1024$ memory locations & I/O addresses can be supported, in any combination.

(2) Isolated I/O → With isolated I/O, address space for I/O is isolated from that for memory. The full range of addresses are available for both I/O devices & memory locations.

Ex → With 10 address lines, the system may now support both 1024 memory locations & 1024 I/O addresses.

Difference between Isolated I/O & Memory-mapped I/O:

Isolated I/O	Memory-Mapped I/O
Isolated I/O uses separate memory space	Memory-mapped I/O uses memory from the main memory.
Limited instructions can be used, these are IN, OUT, INS OUTS	Any instruction which references to memory can be used.
The address for Isolated I/O devices are called Ports.	Memory mapped I/O devices are treated as memory locations on the memory map.

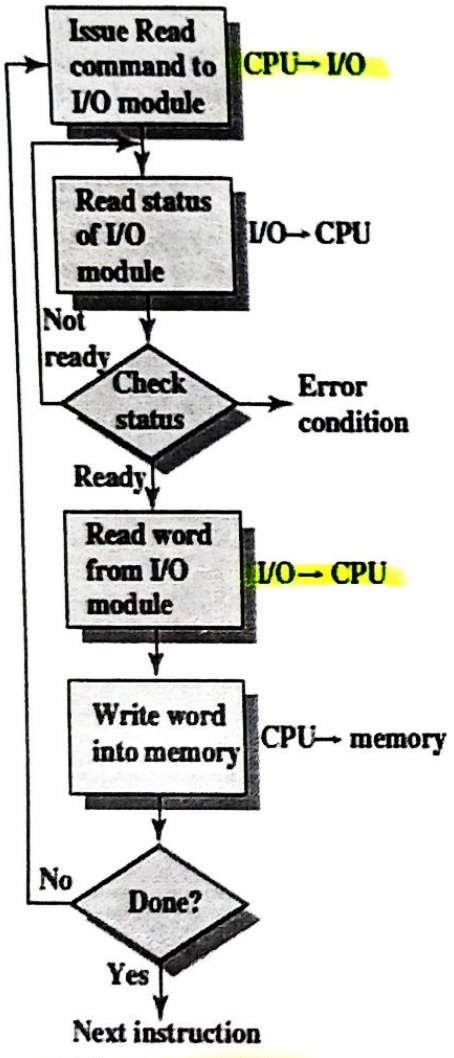
Advantages & disadvantages of programmed I/O →

Advantages	① Simple to implement ② Very little hardware support
Disadvantages	① busy waiting ② ties up CPU for long period with no useful work.

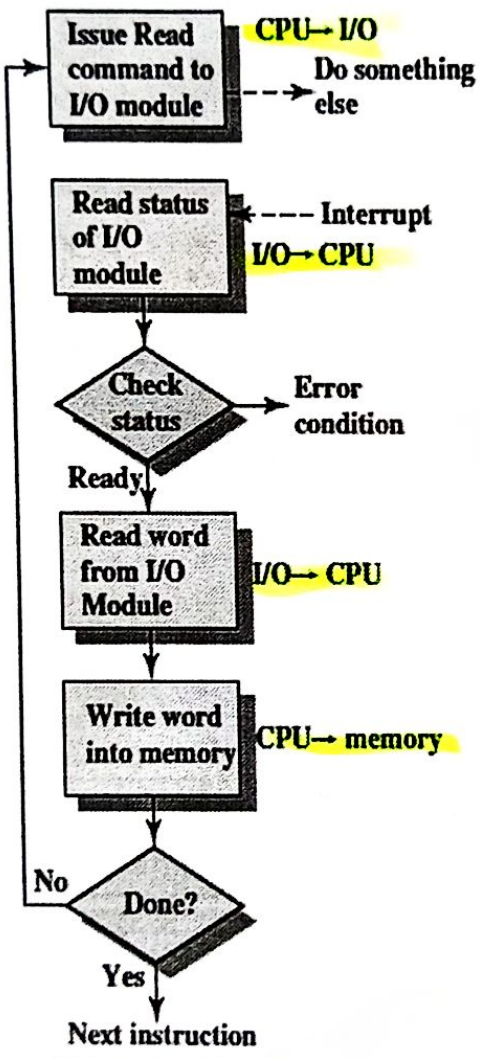
* Programmed I/O technique is used for input data transfer

Flow-Chart is shown in fig 7.4 (a)

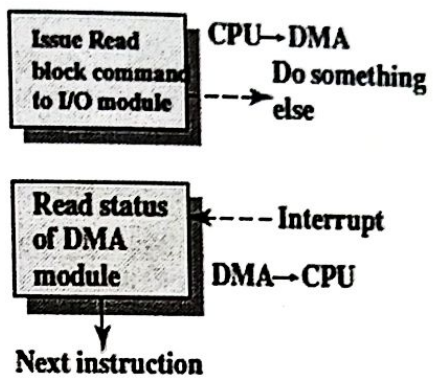
Three Techniques for Input of a Block of Data



(a) Programmed I/O



(b) Interrupt-driven I/O



(c) Direct memory access

This flowchart highlights the main disadvantage of programmed I/O technique:

Data are read in one word (e.g., 16 bits) at a time

For each word that is read in, the processor must remain in a status-Checking cycle until it determines that the word is available in the I/O module's data register.

it is a time-consuming process that keeps the processor busy needlessly.

Fig 7-14

82C59A interput Controller (P-267)

7.4 Interrupt-driven I/O →

With interrupt-driven I/O, the processor does not wait until the I/O operation is completed. The processor performs other tasks while the I/O operation is being performed. When the I/O operation is completed, the I/O module interrupts the processor letting the processor knows the operation is completed & leads to perform the interrupt-handler program.

Interrupt-driven I/O is used for input data transfer Flow-chart is shown in Fig 7.4(b)

Interrupt processing →

When I/O device completes an I/O operation, the following sequence of hardware events occurs:

1. The device issues an interrupt signal to the processor.
2. The processor finishes execution of the current instruction before responding to the interrupt.
3. After responding the interrupt, it sends an acknowledgment signal to the device. The acknowledgment allows the device to remove it's interrupt signal.

PSW → Program status Word register
PC → program counter register.

to begin again

(4) The processor now needs to prepare to transfer control to the interrupt routine. To begin, it needs to save information needed to resume the current program at the point of interrupt. The minimum information required is (a) the status of processor, which is contained in a register called PSW, and (b) the location of next instruction to be executed, which is contained in PC. These can be pushed onto the system control stack & then saved.

(5) The processor now loads the PC with entry location of the interrupt handling program that will respond to this interrupt.

Once the PC has been loaded, the processor proceeds to the next instruction, which begins with an instruction fetch, the result is that control is transferred to the interrupt-handler program.

(6) When interrupt processing is complete, the saved register values are retrieved from the stack & restore to the registers (PC & PSW).

82C59A Interrupt controller →

The Intel 80386 (processor) provides a single interrupt request (INTR) & a single interrupt acknowledge (INTA) line. To allow the 80386 to handle a variety of devices, it is usually configured with an external interrupt arbiter (82C59A). External devices are connected to the 82C59A, which in turn connects to the 80386.

Fig 7.8 shows the use of the 82C59A to connect multiple I/O modules for the 80386. A single 82C59A can handle up to eight modules. It control for more than eight modules is required, a cascade arrangement can be used to hand up to 64 modules.

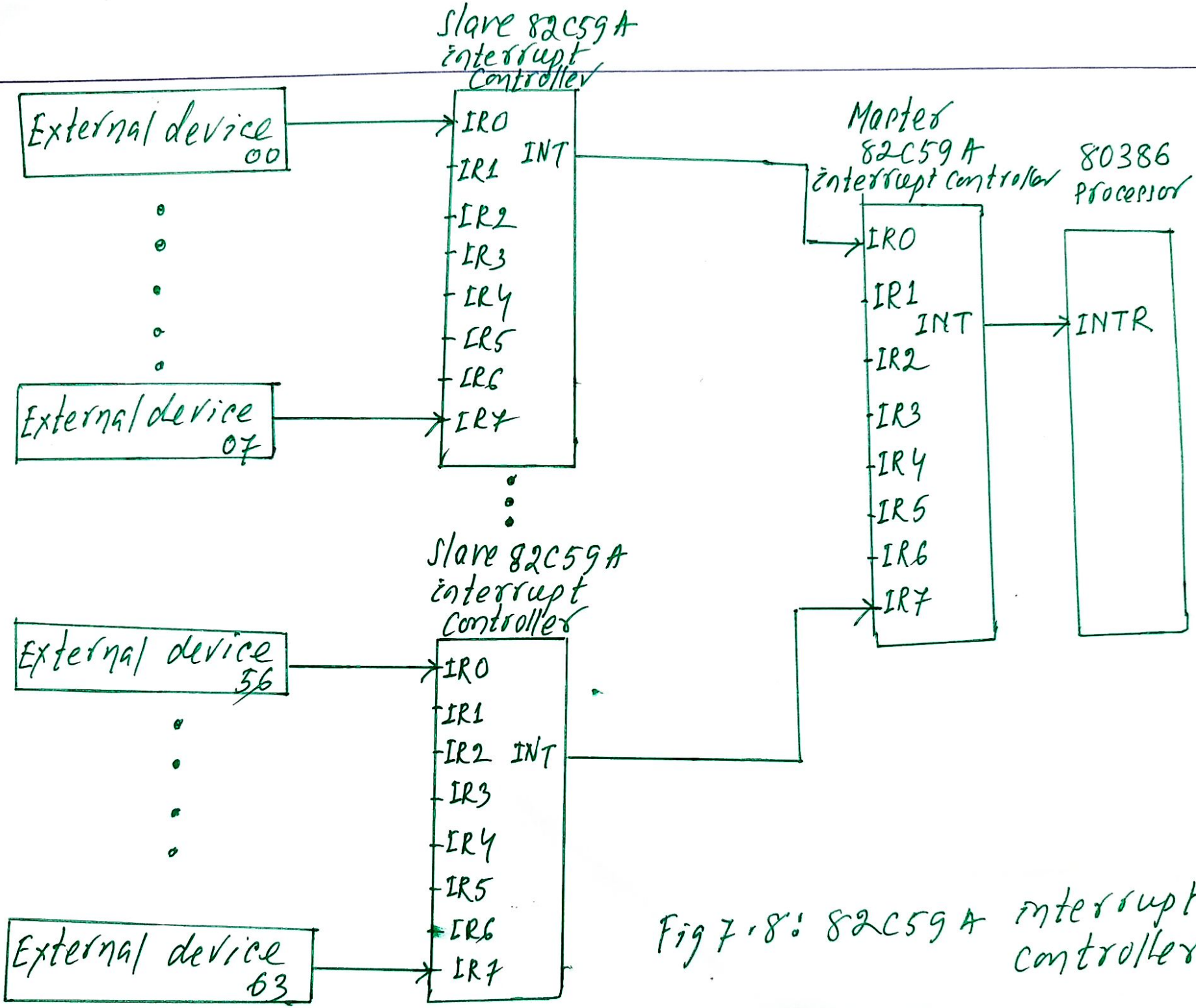


Fig 7.8: 82C59A interrupt controller.

8255A Programmable peripheral interface (I/O module) →

8255A is a I/O module, originally designed for use with the 80386 processor. Fig 7.9 is the block diagram of 8255A I/O module. 8255A includes the following lines.

D_0-D_7: These are the data I/O lines for the device. All information read from & written to the 8255A occurs via these data lines.

$\overline{CS}$ (chip select input): If it is a logical 0, Mp can read & write to the 8255A.

$\overline{RD}$ (Read input): If it is a logical 0, the 8255A data outputs are enabled onto the system data bus.

$\overline{WR}$ (Write input): if it is a logical 0, data are written to the 8255A from the system data bus.

RESET: If it is a logical 1, All the peripheral ports are set to the input mode.

PA0-PA7, PB0-PB7, PC0-PC7: These signal lines are used as 8-bit I/O ports. They can be connected to peripheral devices.

A_0, A_1: The logical combination of these two input lines determine which internal register of the 8255A data are written to or read from.

The right side of 8255A is the external interface. The 24 I/O lines are divided into three 8-bit groups (A, B, C). Each group can function as an 8-bit I/O port, thus providing connection for three peripheral devices. In addition, group C is subdivided into 4-bit groups (C_A, C_B), which may be used in conjunction with the A & B I/O ports. Group C lines carry control & status signal.

The left side of 8255A is the internal interface to the Mp system bus. It includes an 8-bit bidirectional data bus (D_0-D_7), used to transfer data between Mp & the I/O ports

The Mp controls the 8255A by help of an 8-bit control register in the processor. The processor can set the value of the control register to specify a variety of operating modes.

The two address lines specify one of the three I/O ports or the control registers, as follows:

A_1	A_0	Select
0	0	Port A
0	1	Port B
1	0	Port C
1	1	Control register

Thus, when Mp sets both A_1 & A_2 to 1, the 8255A interprets the 8-bit value on the data bus as a control word. When Mp transfers an 8-bit control word with $D_7 = 1$ (Fig 7.10 a), the control word is used to configure the operating mode of the 24 I/O lines. The three modes:

Mode 0: This is the basic I/O mode. The three groups (A, B, C) act as I/O ports. Each port can be designated as input or output. Data may only be sent to a port if the port is defined as output, & data may only be read from a port, if the port is set to input.

Mode 1: In this mode, port A & B can be configured as either input or output, & lines from port C serve as control lines for A & B. The control signals serve two principal purposes: handshaking & interrupt request.

Mode 2: This is a bidirectional mode. In this mode, port A can be configured as either the input or output for bidirectional trafic on port B. Again port C lines are used for control signaling.

When the Mp sets $D_7 = 0$ (fig 7.10(b)), the control word is used to program the bit values of port C individually. This feature is rarely used.