Interface is the path for communication between two components. Interfacing is of two types, memory interfacing and I/O interfacing.

Memory Interfacing

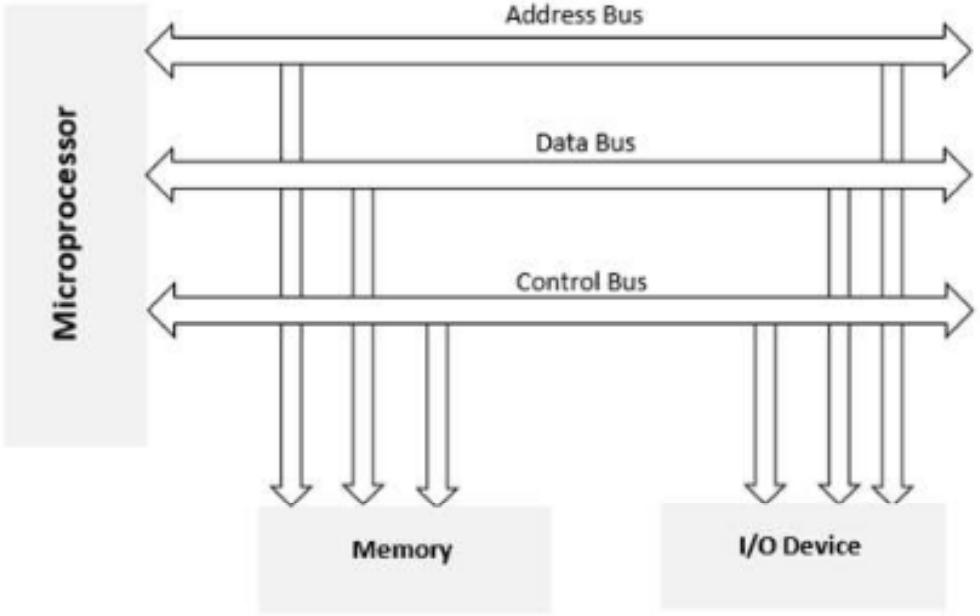
When we are executing any instruction, we need the microprocessor to access the memory for reading instruction codes and the data stored in the memory. For this, both the memory and the microprocessor requires some signals to read from and write to registers.

The interfacing process includes some key factors to match with the memory requirements and microprocessor signals. The interfacing circuit therefore should be designed in such a way that it matches the memory signal requirements with the signals of the microprocessor.

IO Interfacing

There are various communication devices like the keyboard, mouse, printer, etc. So, we need to interface the keyboard and other devices with the microprocessor by using latches and buffers. This type of interfacing is known as I/O interfacing.

Block Diagram of Memory and I/O Interfacing



8085 Interfacing Pins

Following is the list of 8085 pins used for interfacing with other devices −

- A_{15} - A_8 (Higher Address Bus)
- AD_7 - AD_0(Lower Address/Data Bus)
- ALE
- RD
- WR
- READY

Ways of Communication − Microprocessor with the Outside World?

There are two ways of communication in which the microprocessor can connect with the outside world.

- Serial Communication Interface
- Parallel Communication interface

Serial Communication Interface − In this type of communication, the interface gets a single byte of data from the microprocessor and sends it bit by bit to the other system serially and vice-a-versa.

Parallel Communication Interface − In this type of communication, the interface gets a byte of data from the microprocessor and sends it bit by bit to the other systems in simultaneous (or) parallel fashion and vice-a-versa.

Before having a discussion regarding the demerits or merits of I/O mapped I/O and memory-mapped I/O, let us have a generic discussion regarding the difference between I/O mapped I/O and memory mapped I/O.

In Memory Mapped Input Output –

- We allocate a memory address to an Input-Output device.
- Any instructions related to memory can be accessed by this Input-Output device.
- The Input-Output device data are also given to the Arithmetic Logical Unit.

Input-Output Mapped Input Output –

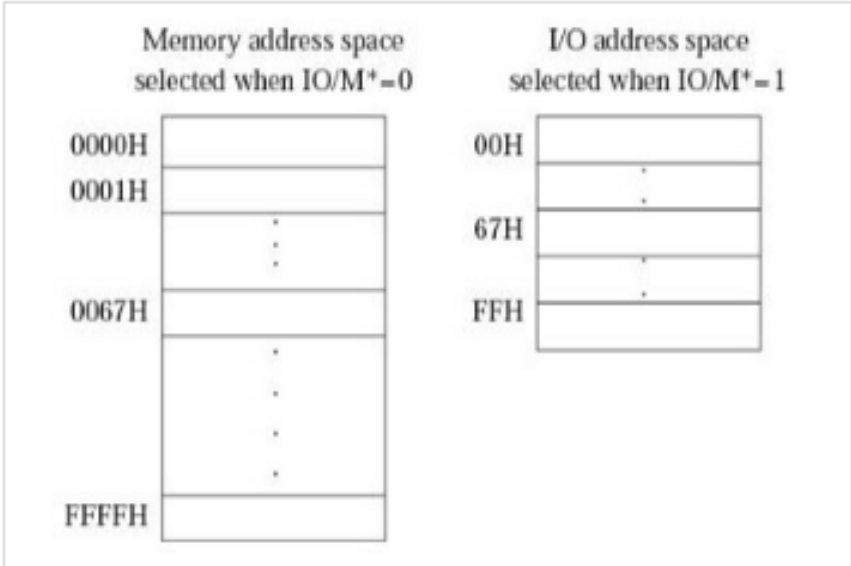
- We give an Input-Output address to an Input-Output device
- Only IN and OUT instructions are accessed by such devices.
- The ALU operations are not directly applicable to such Input-Output data.

So as a summary we can mention that −

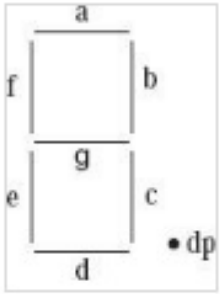
- I/O is any general-purpose port used by processor/controller to handle peripherals connected to it.
- I/O mapped I/Os have a separate address space from the memory. So, total addressed capacity is the number of I/Os connected and a memory connected. Separate I/O-related instructions are used to access I/Os. A separate signal is used for addressing an I/O device.
- Memory-mapped I/Os share the memory space with external memory. So, total addressed capacity is memory connected only. This is underutilisation of resources if your processor supports I/O-mapped I/O. In this case, instructions used to access I/Os are the same as that used for memory.
- Let's take an example of the 8085 processor. It has 16 address lines i.e. addressing capacity of 64 KB memory. It supports I/O-mapped I/Os. It can address up to 256 I/Os.
- If we connect I/Os to it an I/O-mapped I/O then, it can address 256 I/Os + 64 KB memory. And special instructions IN and OUT are used to access the peripherals. Here we fully utilize the addressing capacity of the processor.
- If the peripherals are connected in memory mapped fashion, then total devices it can address is only 64K. This is underutilisation of the resource. And only memory-accessing instructions like MVI, MOV, LOAD, SAVE are used to access the I/O devices.

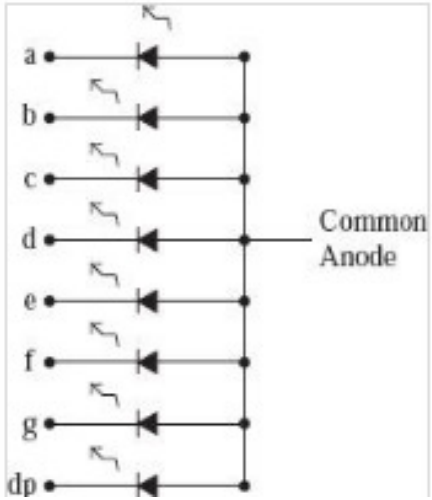
After the discussion stated earlier, it is not possible for us to conclude to which scheme of addressing the Input Output ports is better. Both of them have their advantages and disadvantages. The Intel family of microprocessors like 8085, 8086, 80386, Pentium, and Zilog family of microprocessors like Z-80, Z-8000, etc. provide I/O-mapped I/O facility, in addition to providing memory-mapped I/O. So some I/O ports can be connected as I/O-mapped I/O ports, and some others as memory-mapped I/O ports in an Intel processor-based system. But Motorola family of microprocessors like 6800, 68000, 68020, etc. provide only memory-mapped I/O. Thus, we can say that an Intel processor is better compared with a Motorola processor, as far as addressing of I/O ports is concerned.

After the discussions stated earlier it is not possible for us to conclude to the extent of which scheme of addressing the Input-Output port is better. Both of them have their advantages and disadvantages. The family of microprocessors which belong to the Intel family-like 8085, 8086, 80386, Pentium, and Zilog family of microprocessors like Z-80, Z-8000, etc provide the facility of Input-Output mapping to Input-Output facility. Where in addition, we provide the facility of memory mapped Input Output also In the processor-based system manufactured by Intel. But the family of microprocessors which belong to Motorolla like 6800, 68000, 68020 provides only Memory mapped Input Output. Hence we can conclude that the processor which belong to Intel family is far better compared with the processor of Motorolla as far the addressing of the Input-Output ports is concerned.



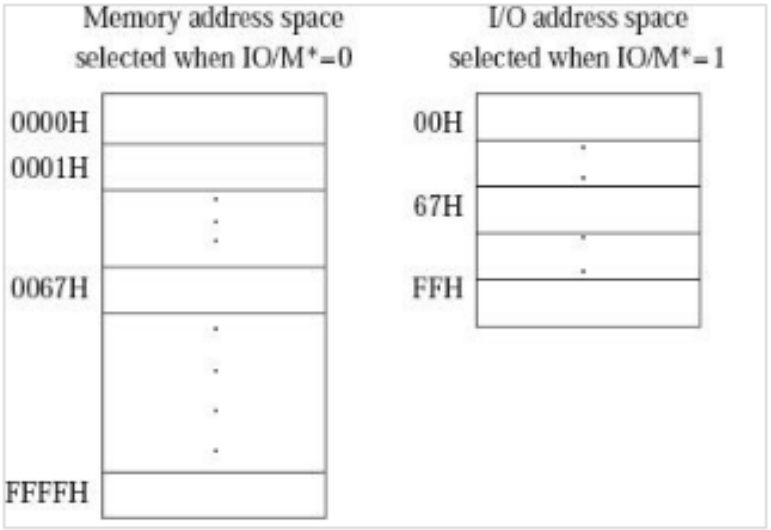
We interface the I/O device in a very segmented manner and is carried systematically. In the interfacing of seven segment display to 8085 microcontroller it is found that An output device which is very common is, especially in the kit of 8085 microprocessor and it is the Light Emitting Diode consisting of seven segments. Moreover, we have eight segments in a LED display consisting of 7 segments which includes '.', consisting of character 8 and having a decimal point just next to it. We denote the segments as 'a, b, c, d, e, f, g, and dp' where dp signifies '.' which is the decimal point resulting to a seven display circuit of LEDs which are seven segmented now the display is visual to the serial transfer of 8085 microcontroller. The portion which is displayed of the interface has 4 LEDS comprises of 7-segments LEDs as we can see from the physical layout of the interface. The connection to the interface is done by the ALS-8085 kit which uses a flat table of 26 crores. We connect the connector C1 to the interface to the Input Output connector P3 on the ALS kit.





Now we shall discuss about merits of I/O mapped I/O and demerits of memory-mapped I/O –

- We use the instructions IN and OUT for the addressing of input output mapped input output ports. The mnemonics of these instructions indicates that the processor is in the process of communication with the Input Output port. For loading the accumulator from memory-mapped input port with address FFF0H we execute the instructions LDA FFF0H.
- There are some microprocessors which consists of a control pin for the selection of memory location or an Input Output port. As an example to be cited as in 8085 processor, the pin IO/M* is used for the selection of an Input Output port or a memory location.
- 256 locations are allocated for the address space of the Input Output port since they are only of size of 8 bits. The address space of memory consists of 64K locations as the addresses of the memory are of length 16 bits.
- If we address the Input Output ports as the memory mapped Input Output ports with the addresses ranging from FF00H-FFF0H then these address ranges should not be allotted for any memory chip.



I/O interfacing

There are two methods of interfacing the Input / Output devices with the microprocessor. They are,

1) Memory mapped I/O and
2) I/O mapped I/O.

Memory mapped I/O

In this method the I/O devices are treated like the memory. A part of the memory address space is used for the I/O devices. The memory mapped I/O scheme is shown in figure.

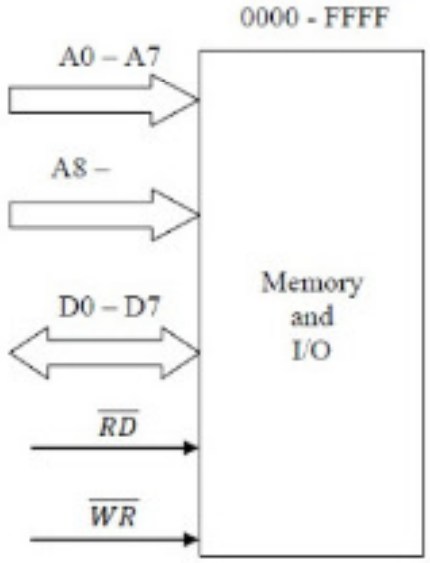


Figure: Memory mapped I/O scheme

- In memory mapped I/O scheme, the same address space is used for both memory and I/O devices.
- The microprocessor uses the sixteen address line A_0 – A_7 and A_8 – A_{15} for the memory as well as for the I/O devices.
- The I/O devices share the address space with the memory. All the memory related instructions are used for addressing I/O devices also.
- No separate IN and OUT instructions are required in memory mapped I/O scheme.
- IO/$\overline{M}$ pin is not required.

Steps for memory operations (memory read and memory write) :

1. When the memory related instructions like LDA and STA are used, the microprocessor places the 16-bit address on the address bus.
2. $\overline{RD}$ is activated for read operation and $\overline{WR}$ is activated for write operation.

Steps for I/O operations (I/O read and I/O write) :

The same steps used for memory operations are used for I/O operations also.

I/O mapped I/O

In this method, I/O devices are treated as I/O devices and memory is treated as memory. Separate address space is used for memory and I/O. The I/O mapped I/O scheme is shown in figure.

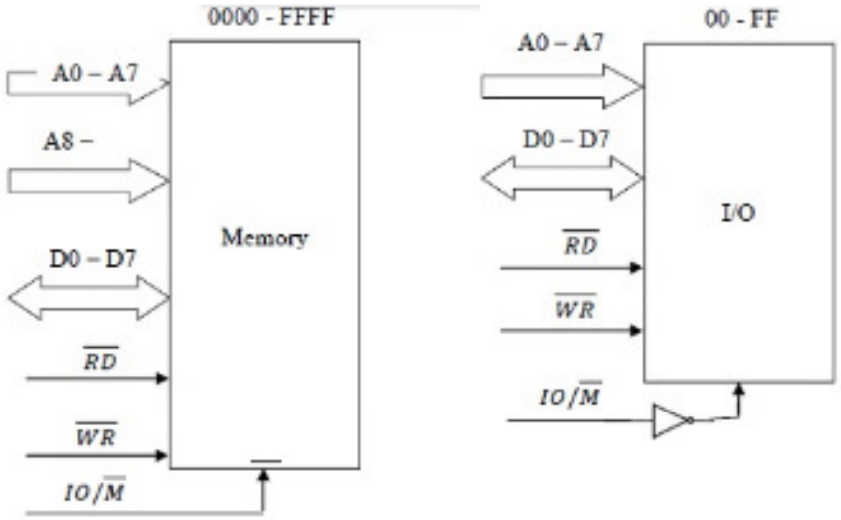


Figure: I/O mapped I/O scheme

• In I/O mapped I/O scheme, the microprocessor uses the sixteen address lines A_0 – A_7 and A_8 – A_{15} for the memory and eight address lines A_0 to A_7 to identify an input / output device.

• Here, the full address space 0000 – FFFF is used for the memory and a separate address space 00 – FF is used for the I/O devices.

• Hence, the microprocessor can address 65536 (2^{16}) memory locations 256 (2^8) input devices and 256 (2^8) output devices separately.

• IN and OUT instructions are used to activate the IO/$\overline{M}$' signal.

• When IO/$\overline{M}$' is low, the memory is selected for reading and writing operations.

• When IO/$\overline{M}$' is high, the I/O port is selected for reading and writing operations.

Steps for memory operations (memory read and memory write) :

1. When the memory related instructions like LDA and STA are used, the microprocessor places the 16-bit address on the address bus.
2. The microprocessor makes the IO/$\overline{M}$ linelow.
3. The microprocessor makes the $\overline{RD}$ low for read operation and $\overline{WR}$ low for write operation.

Steps for I/O operations (I/O read and I/O write) :

1. When the I/O related instructions like IN and OUT are used, the microprocessor places the 8-bit address on the address bus A_0 – A_7 as well as A_8 – A_{15}.
2. IO/$\overline{M}$ line is made high.
3. The microprocessor makes the $\overline{RD}$ low for read operation and $\overline{WR}$ low for write operation.

The 8255A is a general purpose programmable I/O device designed to transfer the data from I/O to interrupt I/O under certain conditions as required. It can be used with almost any microprocessor.

It consists of three 8-bit bidirectional I/O ports (24I/O lines) which can be configured as per the requirement.

Ports of 8255A

8255A has three ports, i.e., PORT A, PORT B, and PORT C.

- **Port A** contains one 8-bit output latch/buffer and one 8-bit input buffer.
- **Port B** is similar to PORT A.
- **Port C** can be split into two parts, i.e. PORT C lower (PC0-PC3) and PORT C upper (PC7-PC4) by the control word.

These three ports are further divided into two groups, i.e. Group A includes PORT A and upper PORT C. Group B includes PORT B and lower PORT C. These two groups can be programmed in three different modes, i.e. the first mode is named as mode 0, the second mode is named as Mode 1 and the third mode is named as Mode 2.

Operating Modes

8255A has three different operating modes −

- **Mode 0** − In this mode, Port A and B is used as two 8-bit ports and Port C as two 4-bit ports. Each port can be programmed in either input mode or output mode where outputs are latched and inputs are not latched. Ports do not have interrupt capability.
- **Mode 1** − In this mode, Port A and B is used as 8-bit I/O ports. They can be configured as either input or output ports. Each port uses three lines from port C as handshake signals. Inputs and outputs are latched.
- **Mode 2** − In this mode, Port A can be configured as the bidirectional port and Port B either in Mode 0 or Mode 1. Port A uses five signals from Port C as handshake signals for data transfer. The remaining three signals from Port C can be used either as simple I/O or as handshake for port B.

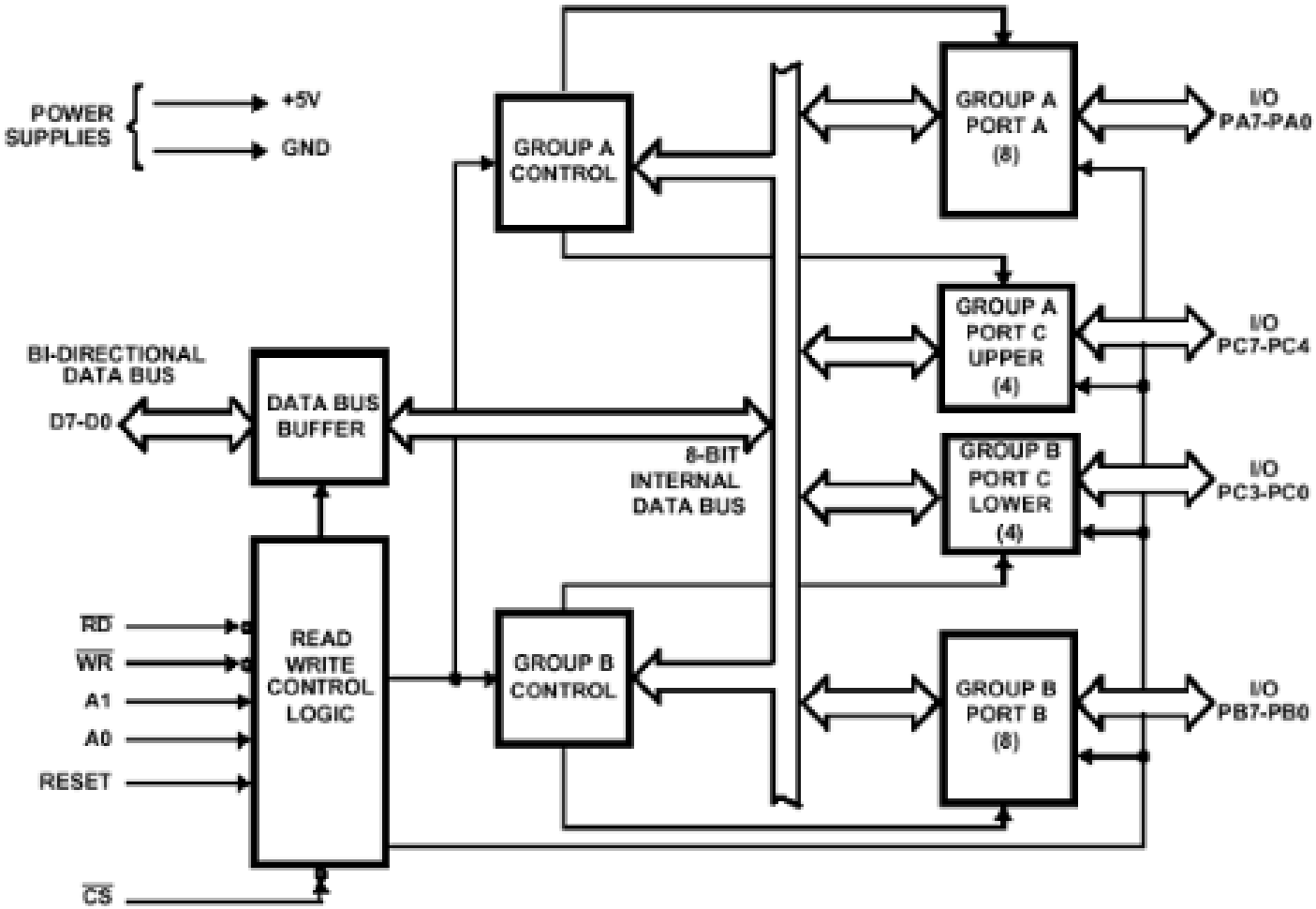
Features of 8255A

The prominent features of 8255A are as follows −

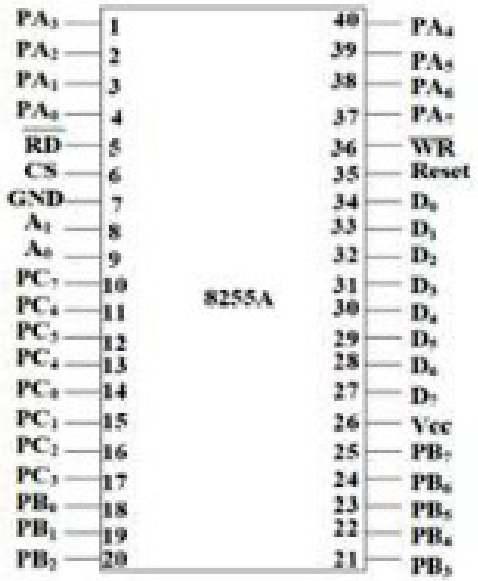
- It consists of 3 8-bit IO ports i.e. PA, PB, and PC.
- Address/data bus must be externally demux'd.
- It is TTL compatible.
- It has improved DC driving capability.

8255 Architecture

The following figure shows the architecture of 8255A –



Let us first take a look at the pin diagram of Intel 8255A –



Now let us discuss the functional description of the pins in 8255A.

Data Bus Buffer

It is a tri-state 8-bit buffer, which is used to interface the microprocessor to the system data bus. Data is transmitted or received by the buffer as per the instructions by the CPU. Control words and status information is also transferred using this bus.

Read/Write Control Logic

This block is responsible for controlling the internal/external transfer of data/control/status word. It accepts the input from the CPU address and control buses, and in turn issues command to both the control groups.

CS

It stands for Chip Select. A LOW on this input selects the chip and enables the communication between the 8255A and the CPU. It is connected to the decoded address, and A_0 & A_1 are connected to the microprocessor address lines.

Their result depends on the following conditions −

CS	A_1	A_0	Result
0	0	0	PORT A
0	0	1	PORT B
0	1	0	PORT C
0	1	1	Control Register
1	X	X	No Selection

WR

It stands for write. This control signal enables the write operation. When this signal goes low, the microprocessor writes into a selected I/O port or control register.

RESET

This is an active high signal. It clears the control register and sets all ports in the input mode.

RD

It stands for Read. This control signal enables the Read operation. When the signal is low, the microprocessor reads the data from the selected I/O port of the 8255.

A_0 and A_1

These input signals work with RD, WR, and one of the control signal. Following is the table showing their various signals with their result.

A_1	A_0	RD	WR	CS	Result
0	0	0	1	0	**Input Operation** PORT A → Data Bus
0	1	0	1	0	PORT B → Data Bus
1	0	0	1	0	PORT C → Data Bus
0	0	1	0	0	**Output Operation** Data Bus → PORT A
0	1	1	0	0	Data Bus → PORT A
1	0	1	0	0	Data Bus → PORT B
1	1	1	0	0	Data Bus → PORT D