Interface circuits

The **I/O interface circuit** is a mediator between the I/O device and the system to which this I/O has to be connected. In our earlier content bus structure, we had discussed a little about the I/O interface where we have seen that one end of the I/O interface is connected to the system bus and the other is connected to the input device.

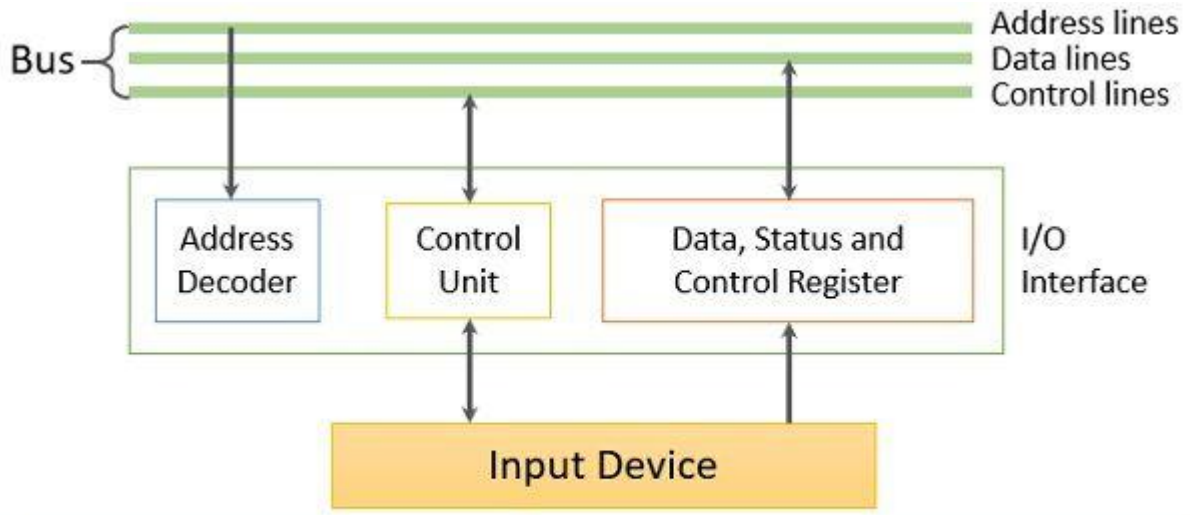
In this context, we will discuss the I/O interface circuit in more detail along with its entire functioning. We will also discuss two variants of interface circuit design i.e., parallel interface and the serial interface.

I/O Interface Circuit

The I/O interface circuit is circuitry that is designed to link the I/O devices to the processor. Now the question is why do we require an interface circuit?

We know that every component or module of the computer has its distinct capabilities and processing speed. For example, the processing speed of the CPU is much higher than the other components of the computer such as keyboard, display, etc.

So, we need a mediator to make the computer communicate with the I/O modules. This mediator is referred to as an interface circuit. Observe the figure below, we can easily see that the one end of the interface circuit is connected to the system bus line i.e., address line, data line, and control line.



Bus Structure for I/O Interface of an Input Device

The address line is decoded by the interface circuit to determine if the processor has addressed this particular I/O device or not. The control line is decoded to identify

which kind of operation is requested by the processor. The data line is used to transfer the data between I/O and the processor.

The other side of the interface circuit has the connections that are essential to transfer data between the I/O interface circuit and the I/O device. And this side of the I/O interface is referred to as the *port*. The port of the I/O interface can be a parallel port or a serial port. We will discuss these ports in the section ahead.

But before discussing these ports let us take a brief outlook of what are the features of the I/O interface circuit.

1. The interface circuit has a **data register** that stores the data temporarily while the data is being exhanged between I/O and processor.
2. The interface circuit also has a **status register**, the bits in the status register indicate the processor whether the I/O device is set for the transmission or not.
3. The interface circuit also has the **control register**, the bits in the control register indicate the type of operation (read or write) requested by the processor to the I/O interface.
4. The interface circuit also has **address decoding circuitry** which decodes the address over the address line to determine whether it is being addressed by the processor.
5. The interface circuitry also generates the **timing signals** that synchronize the operation between the processor and the I/O device.
6. The interface circuit is also responsible for the **format conversion** that is essential for exchanging data between the processor and the I/O interface.

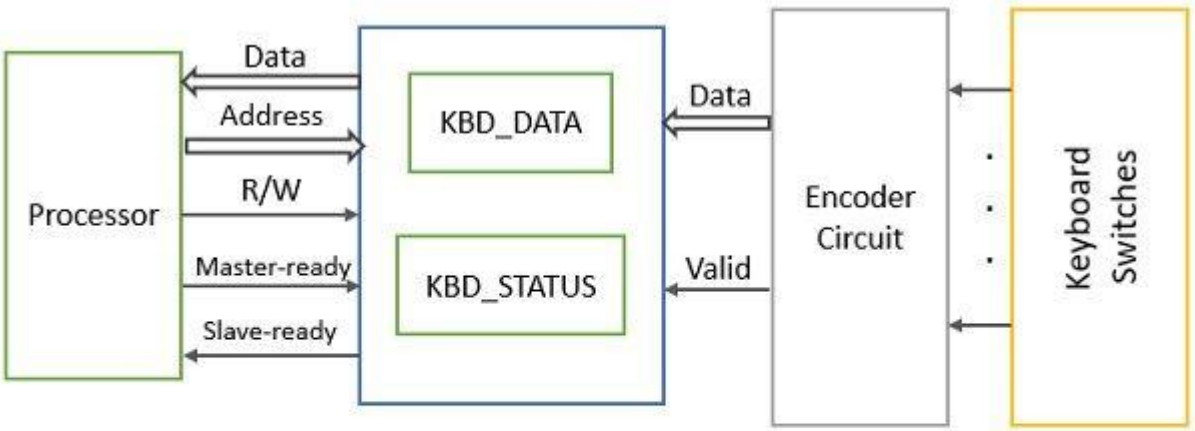
Now, let us learn about the parallel port and the serial port of the I/O interface circuit.

Parallel Port

To understand the interface circuit with a parallel port we will take the example of two I/O devices. First, we will study an input device i.e., a keyboard that has an 8-bit input port, and then an output device i.e., a display that has an 8-bit output port. Here multiple bits are transferred at once.

Input Port

Observe the parallel input port that connects the keyboard to the processor. Now, whenever the key is tapped on the keyboard an electrical connection is established that generates an electrical signal. This signal is encoded by the encoder to convert it into ASCII code for the corresponding character pressed at the keyboard.



Parallel Input Port that connect Keyboard and Processor

The encoder then outputs one byte of data that presents the character encoded by the encoder along with one valid bit. This valid bit changes its status from 0 to 1 when the key is pressed. So, when the valid bit is 1 the ASCII code of the corresponding character is loaded to the KBD_DATA register of the input interface circuit.

Now, when the data is loaded into the KBD_DATA register the KIN status flag present in the KBD_STATUS register is set to1. Which causes the processor to read the data from KBD_DATA.

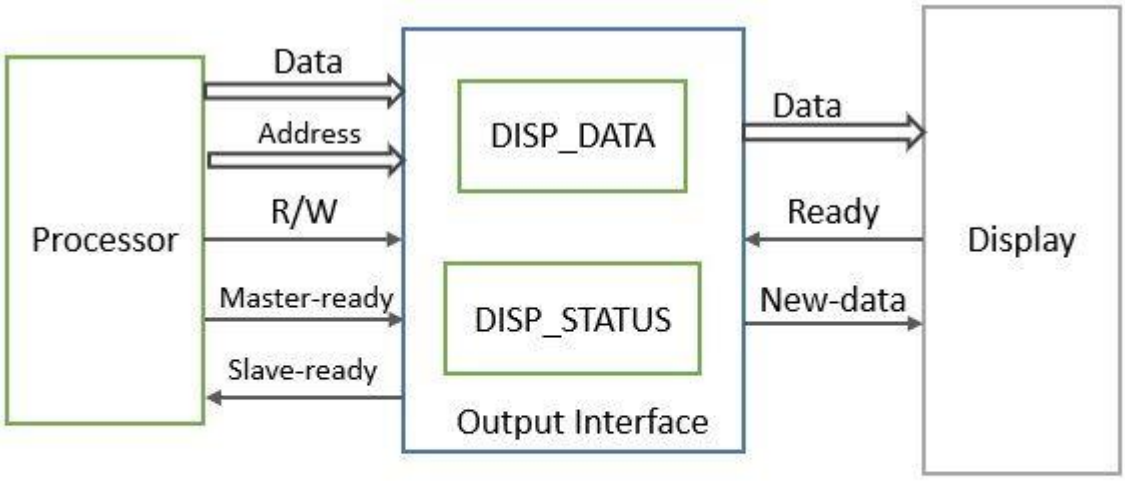
Once the processor reads the data from KBD_DATA register the KIN flag is again set to 0. Here the input interface is connected to the processor using an asynchronous bus.

So, the way they alert each other is using the master ready line and the slave ready line. Whenever the processor is ready to accept the data, it activates its master-ready line and whenever the interface is ready with the data to transmit it to the processor it activates its slave-ready line.

The bus connecting processor and interface has one more control line i.e., R/W which is set to one for reading operation.

Output Port

Observe the output interface shown in the figure below that connects the display and processor. The display device uses two handshake signals that are ready and new data and the other master and slave-ready.



Parallel Output Port that connect Display and Processor

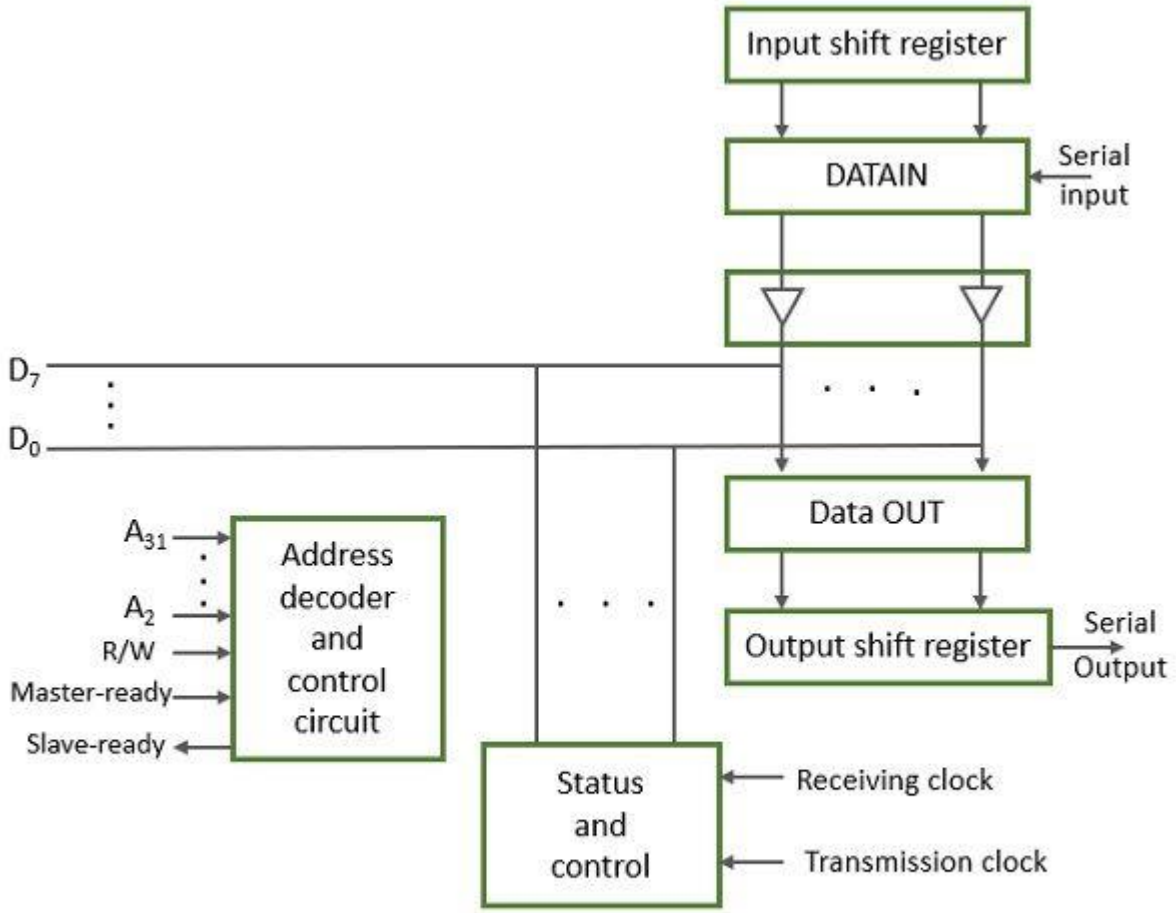
When the display unit is ready to display a character, it activates its ready line to 1 which setups the DOUT flag in the DISP_STATUS register to 1. This indicates the processor and the processor places the character to the DISP_DATA register.

As soon as the processor loads the character in the DISP_DATA the DOUT flag setbacks to 0 and the New-data line to 1. Now as the display senses that the new-data line is activated it turns the ready line to 0 and accepts the character in the DISP_DATA register to display it.

Serial Port

Opposite to the parallel port, the serial port connects the processor to devices that transmit only one bit at a time. Here on the device side, the data is transferred in the bit-serial pattern, and on the processor side, the data is transferred in the bit-parallel pattern.

The transformation of the format from serial to parallel i.e., from device to processor, and from parallel to serial i.e., from processor to device is made possible with the help of shift registers (input shift register & output shift register).



Serial Interface Circuit

Observe the figure above to understand the functioning of the serial interface at the device side. The input shift register accepts the one bit at a time in a bit-serial fashion till it receives all 8 bits. When all the 8 bits are received by the input shift register it loads its content into the DATA IN register parallelly. In a similar fashion, the content of the DATA OUT register is transferred in parallel to the output shift register.

The serial interface port connected to the processor via system bus functions similarly to the parallel port. The status and control block has two status flags SIN and SOUT. The SIN flag is set to 1 when the I/O device inputs the data into the DATA IN register through the input shift register and the SIN flag is cleared to 0 when the processor reads the data from the DATA IN register.

When the value of the SOUT register is 1 it indicates to the processor that the DATA OUT register is available to receive new data from the processor. The processor writes the data into the DATA OUT register and sets the SOUT flag to 0 and when the output shift register reads the data from the DATA OUT register sets back SOUT to 1.

This makes the transmission convenient between the device that transmits and receives one bit at a time and the processor that transmits and receives multiple bits at a time.

The serial interface does not have any clock line to carry timing information. So, the timing information must be embedded with the transmitted data using the encoding scheme. There are two techniques to do this.

Asynchronous Serial Transmission

In the asynchronous transmission, the clock used by the transmitter and receiver is not synchronized. So, the bits to be transmitted are grouped into a group of 6 to 8 bits which has a defined starting bit and ending bit. The start bit has a logic value 0 and the stop bit has a logic value 1.

The data received at the receiver end is recognized by this start and stop bit. This approach is useful where is the transmission is slow.

Synchronous Serial Transmission

The start and stop bit we used in the asynchronous transmission provides the correct timing information but this approach is not useful where the transmission speed is high.

So, in the synchronous transmission, the receiver generates the clock that is synchronized with the clock of the transmitter. This lets the transmitting large blocks of data at a high speed.

This is all about the interface circuit that is an intermediatory circuit between the I/O device and processor. The parallel interface is faster, costly, and efficient for the devices that are at a closer distance to the processor. Whereas the serial interface is is slow, less costly, and efficient for long-distance connection.