Data Transfer Schemes

Data between the μP and I/O devices is transferred according to schemes that may fall into one of the following categories:

1. Programmed data transfer, and
2. Direct memory access memory transfer.

Programmed data transfers are generally used when a relatively small amount of data is transferred using relatively slow I/O devices, e.g. some A/D, D/A converters. In these schemes, usually one byte or word of data is transferred at a time. When a large block of data is to be transferred, DMA is used. DMA is used for transferring data from peripheral mass storage devices like a hard disk or a high-speed line printer.

Programmed data transfer can be further classified as:

1. Synchronous transfer,
2. Asynchronous transfer (or handshaking),
3. Interrupt driven transfer.

All these schemes require both hardware and software for their implementation. Within a μC, more than one scheme can be used for interfacing various I/O devices.

Programmed Data Transfer -- Synchronous Mode

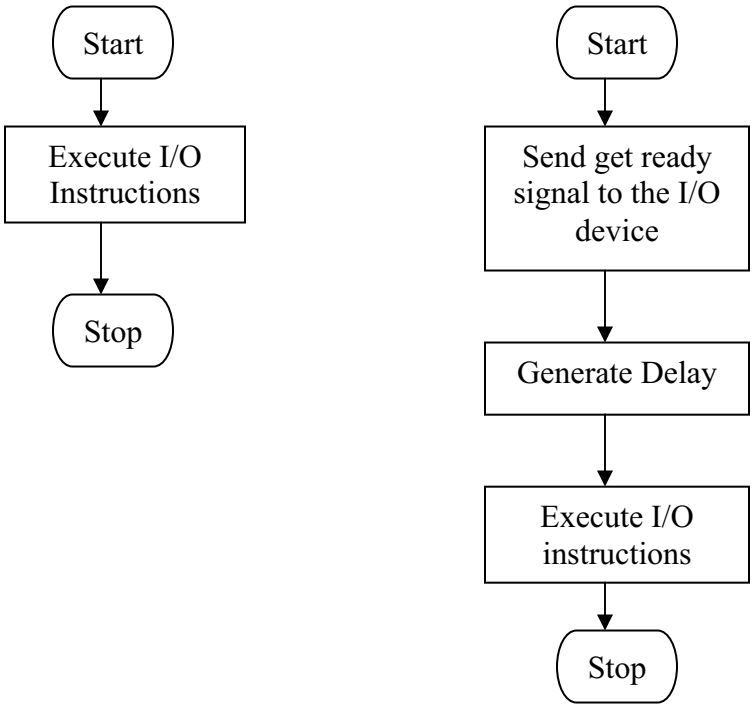
This is the simplest of all the data transfer schemes. Synchronous mode of data transfer is used for I/O devices whose timing characteristics are precisely known, or fast enough to be compatible in speed with the communicating MPU. Whenever data is to be obtained from the device, or transferred to the device, the user program may issue a suitable instruction for the device. At the end of the execution of this instruction, the transfer would have been completed. Thus, if an output device is connected to the 8085 in the memory mapped mode, the instruction MOV M, A may be used for transferring ACC contents to the device, assuming that the address of the device is already stored in the HL register pair. If the device is connected in I/O mapped mode, then the OUT instruction may be issued. The flowchart for this mode is shown in fig 1(a).

The synchronous mode can also be applied to slow I/O devices, if the timing characteristics of these devices are precisely known. In this case the data transfer is initiated by requesting the I/O device to get ready and hen the MPU waits for some predetermined time, usually by generating a delay, and then executes the I/O instruction to complete the data transfer. The flowchart is shown in fig 1(b).

Data Transfer Schemes

Fig 1: Flowchart for the synchronous mode of data transfer:

a) Speed compatible with MPU b) I/O with known speed characteristics



Programmed Data Transfer - Asynchronous Mode

When the I/O device speed and µp speed do not match, asynchronous mode may be used. The key feature of this mode is that, the MPU initiates data transfer by requesting the device to get ready and then goes on checking its status. The I/O instruction is executed only when the I/O device is ready to accept or supply data. So each data transfer is preceded by a 'request ready' signal generated by the MPU and an acknowledgement signal issued by the I/O device. The method is known as handshaking mode of data transfer, as it resembles the back-and-forth movement of hands involved in handshaking, to coordinate the data transfer. The flowchart for this mode of data transfer is shown in fig 2

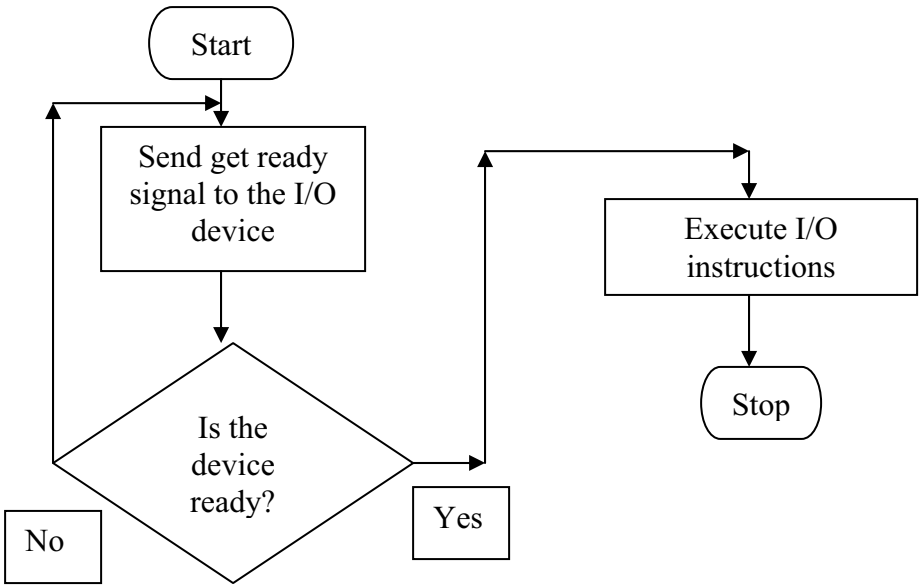
The asynchronous mode is ideal for reconciling the timing differences between the MPU and I/O devices. However, an important disadvantage is that a lot of precious MPU time is wasted during the looping to check the I/O device status. The wastage of the MPU time may be prohibitive or impractical in many situations.

Data Transfer Schemes

Algorithm

```
begin
   issue instruction to the device to get ready;
      repeat
            test device ready flag;
            until device ready flag;
      issue instructions to transfer data;
  end
```

Fig 2: Flowchart for the asynchronous mode of data transfer



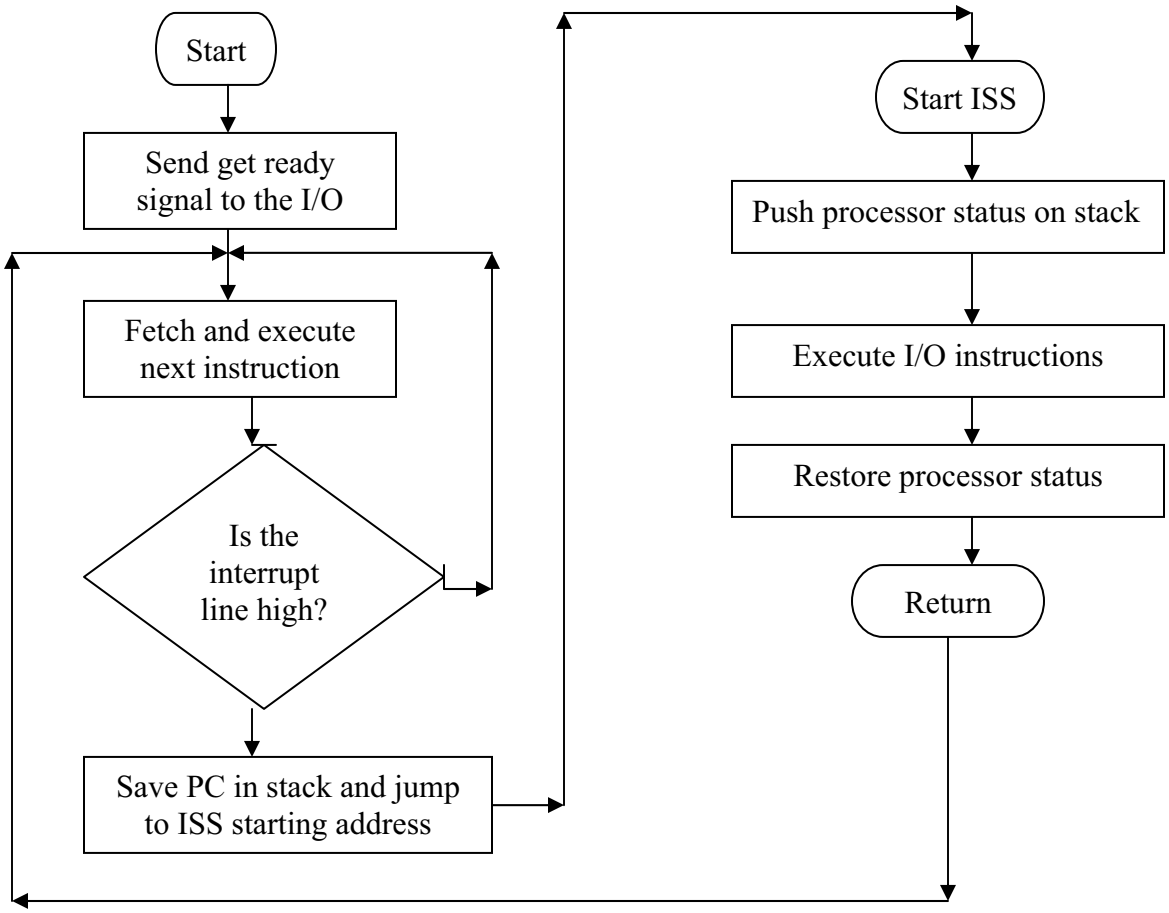
Programmed Data transfer -- Interrupt driven Mode

To overcome the drawbacks of the synchronous and asynchronous modes of data transfer, another programmed data transfer called Interrupt driven mode has been designed. The basic idea of this mode is that, the processor initiates the data by requesting the device 'to get ready' and then goes on executing its original program instead of wasting its time by continuously monitoring the status of the I/O device. When the device is ready to accept or supply data, it informs the processor through some special control line known as interrupt line. In response to this, the processor completes the execution current instruction. Then, instead of executing the next instruction, it saves the PC (and other registers as determined by the

architecture of the MPU) in stack and branches to a predetermined location, which is the starting address of a subroutine called interrupt service subroutine (ISS). The ISS saves the processor status on the stack, completes the data transfer with the I/O responsible for interrupt, restores the processor status and then returns to the original program that the MPU was executing prior to the interrupt. Fig 3 shows the flowchart of this mode of data transfer.

In this mode, the time needed by the I/O device to get ready after receiving the 'get ready' is utilized by the MPU. However, the MPU incurs some overhead in storing and restoring the processor status to and from the stack.

Fig 3 Data transfer in interrupt driven mode



DMA mode of data transfer

For any MPU, there are instructions for data transfer from memory or I/O device to the MPU registers and vice versa. But there does not exist any instruction for data transfer between

memory and I/O devices directly. So, using any one of the programmed I/O modes, data transfer between memory and I/O devices can be done in two steps using one of the MPU registers (usually the accumulator) as via-media. This makes the programmed I/O data transfer modes inherently slow. If a large block of data is to be transferred between memory and a fast I/O device, the overhead incurred may be prohibitive. The direct memory access (DMA) mode has been designed to overcome this problem. The main idea underlying this method is that, the MPU is dissociated from the data transfer process by tristating its system bus. A direct link is established between the memory and I/O device, and an external circuit known as DMA controller controls data transfer. Here, the data transfer rate is only limited by the minimum speed of either of the two devices. However, to facilitate DMA mode of data transfer, the MPU must be equipped with the following features:

1. An input control line, through which the I/O device, via the DMA controller, requests the MPU for DMA data transfer.
2. An output control line, through which the MPU informs the DMA grant.
3. The MPU must be able to tristate the address, data and necessary control lines throughout the DMA data transfer duration.

Almost all the MPUs provide these features. The DMA controller must also perform the following functions:

1. Interface the MPU buses with the I/O device.

2. Generate DMA request signal.

3. In response to the DMA grant signal from the MPU, it must control the address bus and the control lines needed fro data transfer.

4. It must hold the information about the number of bytes to be transferred, along with starting address of the data in memory, so that it can sequentially generate the RAM address one by one and can withdraw the DMA request when the last byte of data transfer is over.

The sequence of events the take place is shown by the flow diagram of fig 4. Two different types of DMA transfer are possible. In the first case, once initiated, the data transfer process does not stop until the complete block is transferred. Therefore, this is known as the block DMA mode. If, however, the MPU cannot be kept inactive for the long duration needed to transfer the complete block of data, or there is significant delay between the consecutive data,

the other scheme known as the cycle stealing DMA mode can be used. In this case, one or two bytes of data are transferred on being granted the DMA by the MPU, and then withdraw the DMA request. After certain time the DMA controller interrupts the MPU, indicating the end of the DMA.

Fig 4. Flowchart of the data transfer in DMA mode