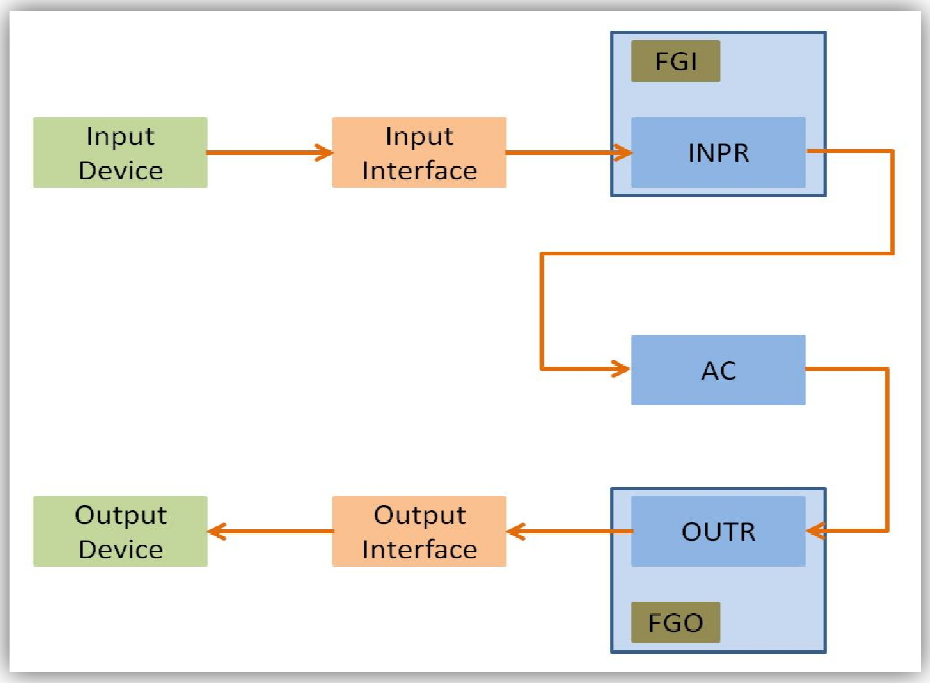
Course Name: A Level (1st Sem) Subject: CO

Topic: I/O Organization and Interrupt Cycle Date: 21-05-20

I/O Organization: Input or Output processes are a little complex than other memory or register reference processes. One most important thing behind it is the frequent interaction of h/w devices. To let the respective h/w do its task, two separate registers are used within the CPU.

These are: INPR (Input Register) and OUTR (Output Register). Both of these registers are of 8 bit size, hence despite of the type of data they hold, we say that an INPR holds an input character that comes from some input device whereas an OUTR holds an output character that goes to some output device.

Two very specific flip-flops FGI and FGO are used to denote the forecast of some input or output operation about to happen. Set status of FGI tells that some input device wants to use INPR for its input data. Similarly, set status of FGO tells that some output data is ready to be taken from OUTR for output devices.



Now the question is that, since stored program execution is a continuous task, what will be the actual time to trigger the I/O operation? The answer is unveiled when we determine the type of I/O transfer. Actually we have three types of I/O transfer- the one is Programmed I/O, the second is Interrupt I/O & the third one is DMA. Out of all these, DMA will be discussed later.

Programmed I/O: The systematic approach is that CPU, while doing it's own tasks, should check regularly the status of FGI and FGO flip-flops. If any of these flip-flops are set, the respective I/O process can start abruptly. This method is good enough but quite resource

© Shashi Kant Mani Tripathi

consuming. It is very clear that in case of a long silence in FGI and FGO, the CPU is wasting it's time by regularly checking their status.

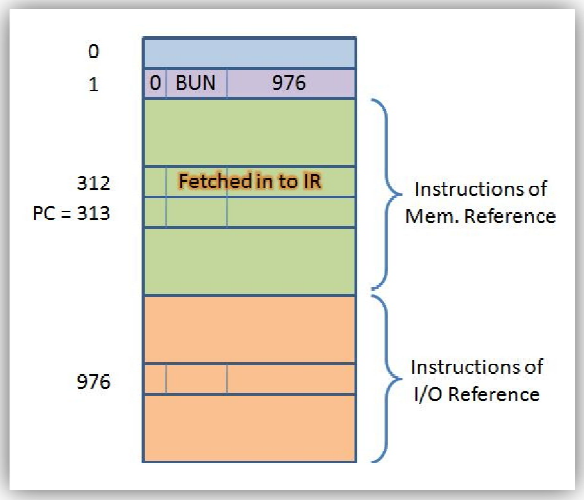
Interrupt I/O: A good alternative to Programmed I/O is to let the device inform the control that some I/O transfer is needed. The CPU will halt it's ongoing task temporarily and branch to the requested I/O transfer. After the control is finished with I/O, it returns to the halted task. Here letting the device intervene CPU is called Interrupt. A third flip-flop IEN is used to denote the forecast of I/O transfer.

Interrupt Cycle: Interrupt cycle is very similar to the instruction cycle. At the very start, the status of flip-flop R is checked. If it is 0 there is no interrupt and CPU can continue its ongoing tasks. But when R=1, it denotes that the ongoing process should halt because an interrupt has occurred.

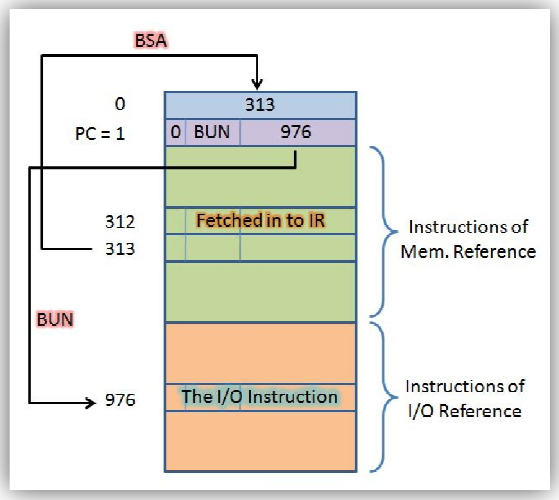
When R=0, CPU continues its tasks checking the status of IEN in parallel. If it is 1, FGI and FGO are checked in a hierarchy. If any of these flip-flops are found set, R is immediately set by 1.

When R=1, the content in PC (address of next instruction in memory) is saved at M [0] and then PC is set by 1 enabling it to point the BUN operation. The instruction at M[1] is a BUN instruction that leads the control to appropriate I/O ref. Instruction stored at some other location in the memory. Now separate Fetch, Decode and Execute phases are practised to entertain the I/O ref. instruction.

Once the I/O ref. instruction is executed completely, PC is loaded with 0 where it finds the saved RETURN address. The entire workout is diagrammed as follows:



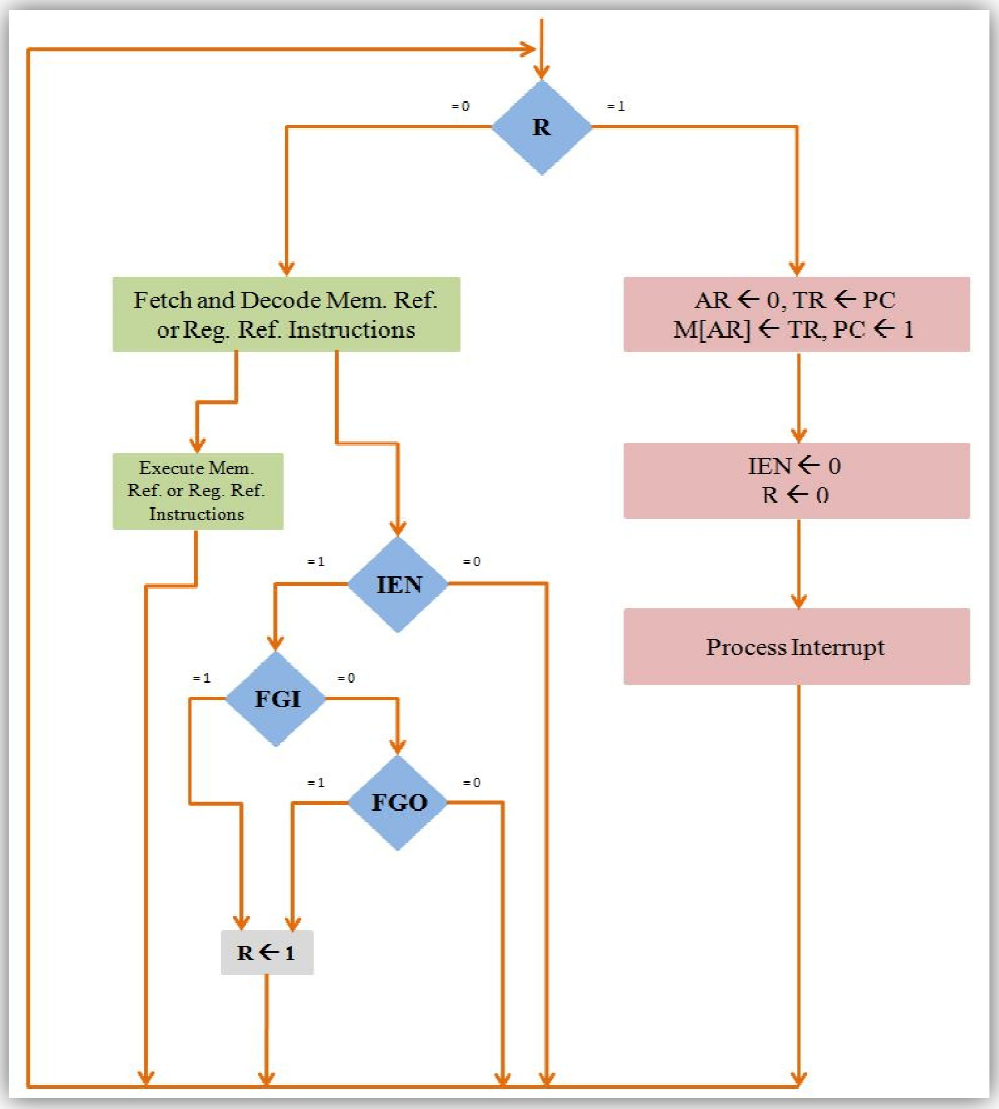
Before Interrupt



During Interrupt

© Shashi Kant Mani Tripathi

The complete interrupt cycle is diagrammed as follows:



Assignment:

1. Differentiate between Programmed I/O and Interrupt I/O.
2. In clear words, demonstrate the handling of interrupts.

© Shashi Kant Mani Tripathi