Input/Output Operations

The transfer of data to or from port can be done in two ways.

One way is to execute an instruction that causes a single byte or word to be transferred.

Other way is to execute a sequence of instructions that causes a special system component (DMA Controller) associated with the interface to transfer a sequence of bytes or words to or from a predesignated block of memory locations. This is referred to as Block transfer or Direct Memory Access

.....

In case of byte or word transfer if data are o be moved between memory and a port , they must be transferred via CPU register. To transfer a stream of bytes or words being sent to memory from an I/O, the program would need to successively, one by one, input the byte or words to the CPU and put them in consecutive memory locations.This could be done with program loop.

.....

In case of block transfer, it is supervised by the interface and the DMA controller and each byte is moved directly from port to the memory as soon as it arrives at the port. The program need only give the required commands to the interface and DMA controller to initiate the transfer.

Similarly, the interface and its DMA controller can take consecutive bytes or words from memory and send them to external device.

Input/Output Operations(Cont..) Programmed I/O

Various methods may be used to control the peripheral adaptor and pass data to and from the attached peripheral.

Programmed I/O

Memory Mapped I/O

Interrupted I/O

DMA

....

In this case all transfers between the CPU and the peripheral adaptor are completely under program control.The instruction set of processor contains instructions such as:

-input data to the CPU from the peripheral

-output data from the CPU to the peripheral

-set individual bits in the peripheral adaptor's control register

-set individual bits in the peripheral adaptor's status register

....

- A program waits for the peripheral to be ready with data (or ready to receive data) and then explicitly transfers the data.
- The CPU addresses the adapter and tests the status register, if the relevant bit in the register shows that the peripheral is ready, then the CPU puts the data into the interface unit transmit data register.

  This will automatically set a flag to indicate that data is available for the peripheral.

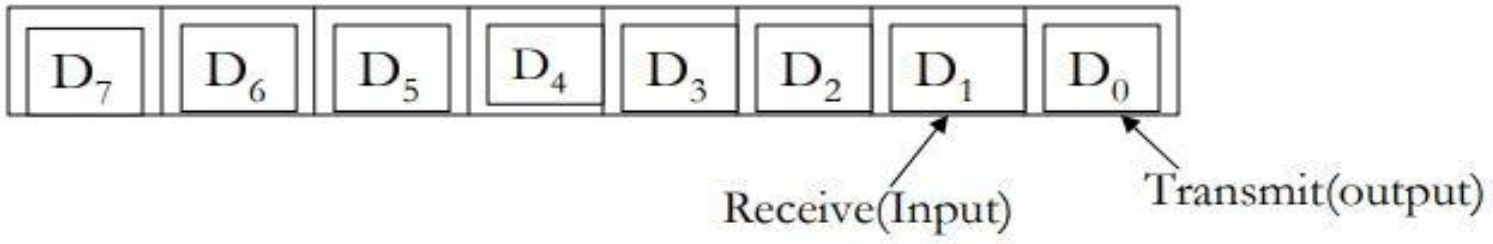
- If the peripheral is not ready, the program must wait and retry

.....

- Example

Sending a character via Intel 8251 Interface circuit.

- The 8251 contains a status register whose least significant two bits indicate the readiness of the transmit and receive sections of the device.
- The program first reads the status register and test the appropriate bit,
  - ✓ If the device is not ready it repeats the test until it is
  - ✓ If the device is ready , a datum is sent and the program proceeds.



Interrupt I/O

Interrupts allows the peripherals to signal its readiness by interrupting the processor .Thus it improves on programmed I/O by not requiring the program to sit idle while waiting for a peripheral to become ready.This allows more effective use of the processor, including running programs at the same time as I/O is being performed.But still it requires action from the CPU for each data to be transferred.

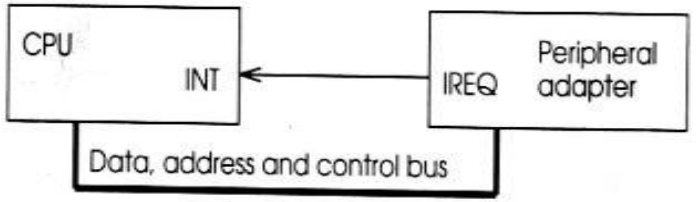


Figure 2.5 *Interrupt connection*

Interrupt I/O

- An interrupt is an event that causes the CPU to initiate a fixed sequence, known as an interrupt sequence.
- When a peripheral is able to transfer data, it sets its ready flag in its status register and also asserts a control line (IREQ) connected to the CPU.

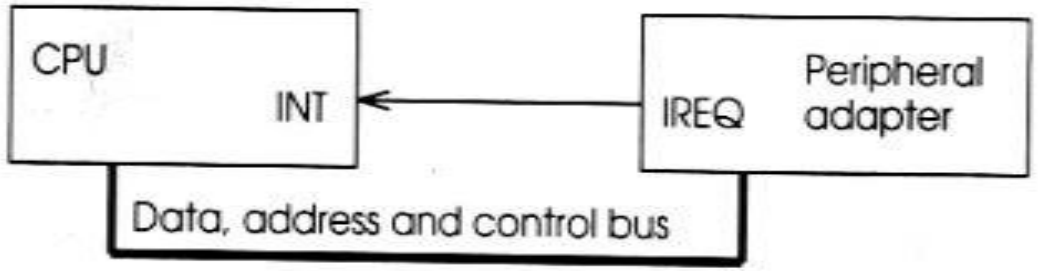


Figure 2.5 *Interrupt connection*

Priority Interrupts

Multiple interrupt requests can be resolved by using a priority interrupt scheme in which a hardware priority encoder arbitrates between requests and sends a single value that represents the highest priority device to the CPU. The interrupt request is formed by performing a logical OR of the peripheral's IREQ lines.

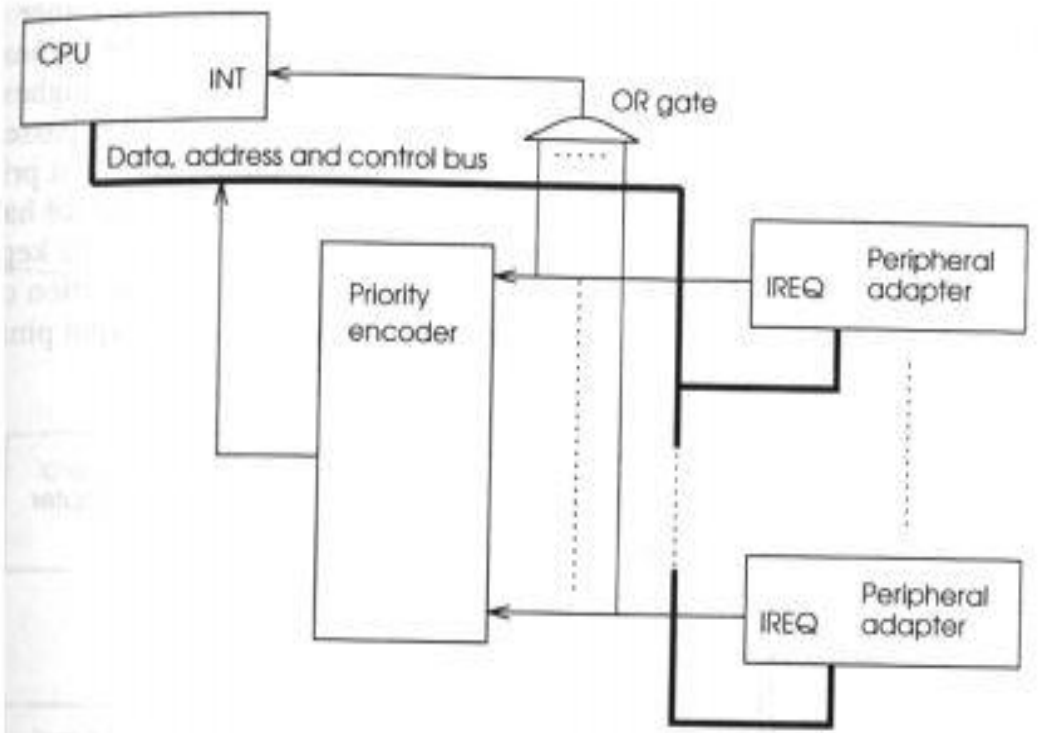


Figure 2.6 *Priority interrupts using a priority encoder*

Interrupt Acknowledgement

- Interrupt requests are assumed to remain asserted until reset by instructions in the service routine. But this is not the most efficient technique.
- Until a request is de-asserted it is not possible for another request to be seen.This may result in data from a fast peripheral being lost while service routine is getting around to clearing a low priority interrupt.
- It could be better if the request could be cleared quickly after the request is noticed.
- To assists in this most computers have a signal (Interrupt Acknowledgement, IACK) generated by the CPU that is returned to the peripheral as soon as the interrupt is detected

Interrupt Acknowledgement

- This clears the interrupt request from that device and allows other devices to use the interrupt line.

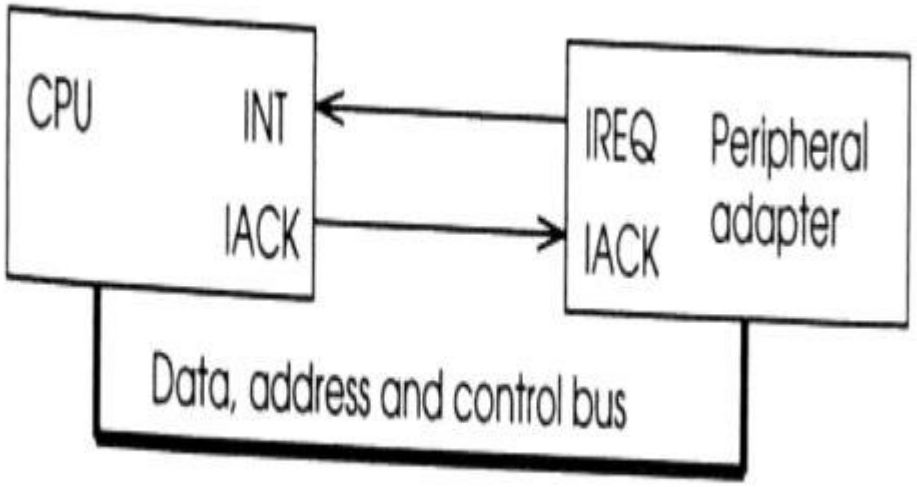


Figure 2.7 *Interrupts with acknowledge*

Priority interrupt using daisy chain..

daisy chain fashion. All the interrupt request lines are OR'ed together, typically with circuitry that allows them simply to be connected together – known as *wired OR*. The CPU IACK is connected directly to the highest priority device so that if more than one request has been made the highest priority device sees it first. If that device has not made a request it passes the IACK along to the next device. This continues down to the lowest priority device which will receive an acknowledge only if no other device has made a request. The only problem with this is that the chain must be kept intact when peripheral adaptors are not inserted, usually by the insertion of a dummy circuit board which simply links the IACK input and output pins.

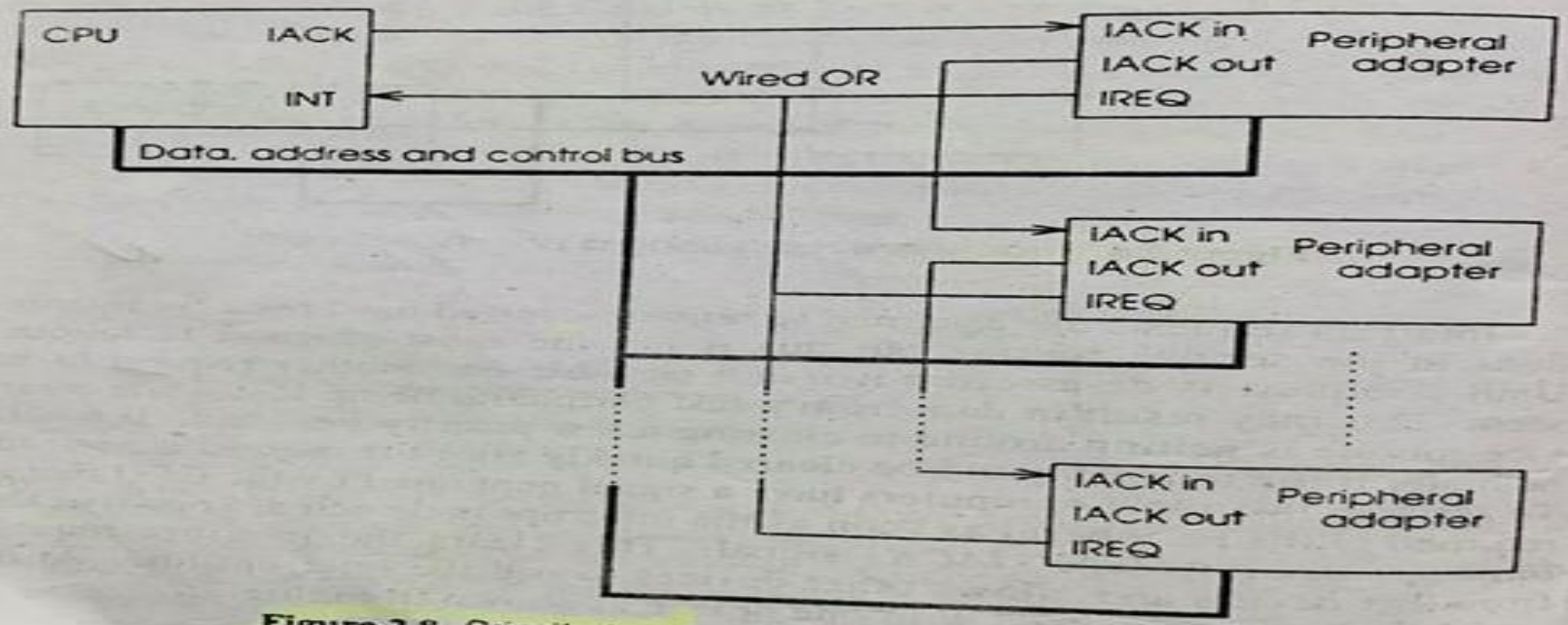


Figure 2.8 *Priority interrupts using a daisy chain*

In order to ensure a fast

Block Data Transfer

- If data transfer rate to or from an I/O is relatively slow , then communication is possible using programmed I/O or interrupt I/O.But some devices (such as A/D converter) may operate at high data rate that can not be handled by byte or word transfer.For this cases, block transfers are required.
- Blocks of data are transferred at a time by special instructions.
- Register contents need not to be saved, so data can be moved faster than programmed or interrupt driven I/O.
- The CPU synchronizes the block movement to or from the peripheral.

Block Data Transfer

- While the block movement is in progress, CPU is unable to perform any other function.
- It is only suited to fast transfers where the CPU and peripheral speeds are reasonably well matched.
- It is not commonly used.

DMA

- Avoids the use of CPU completely for I/O transfers.
- The function of the CPU is to generate correct addresses for the peripheral adaptor and memory and to issue signals to effect transfer. For this it executes one or more instructions only to initiate data transfer operation in between memory and the peripheral.
- For this CPU executes one or more instructions.
- But it is possible to add hardware to implement these functions. Even in some computers, special hardware is used to initiate the data transfer operation.

DMA

- Use of such hardware is known as Direct Memory Access (DMA)
- When a peripheral indicates that it is ready for a transfer, the DMA unit gains the control of the bus, places appropriate address and control signals on it to make the transfer and then releases the bus.
- This action of taking over the bus for a period and executing a memory access cycle instead of the CPU doing so is known as Cycle Stealing.

END..