Computer Organization and Architecture

1 Basic Computer Networks NLU-FIT

Chapter 07

INPUT/OUTPUT

KEY POINTS

2 Basic Computer Networks NLU-FIT

- The computer system's I/O architecture is its interface to the outside world.
- The are three principal I/O techniques:
  - **Programmed I/O,** in which I/O occurs under the direct and continuous control of the program requesting the I/O operation.
  - **Interrupt-driven I/O,** in which a program issues an I/O command and then continues to execute, until it is interrupted by the I/O hardware to signal the end of the I/O operation.
  - **Direct memory access (DMA),** in which a specialized I/O processor takes over control of an I/O operation to move a large block of data.

Why I/O Modules and Function?

3 Basic Computer Networks NLU-FIT

- In addition to the processor and a set of memory modules, the third key element of a computer system is a set of I/O modules.
  - There are a wide variety of peripherals with various methods of operation. It would be impractical to incorporate the necessary logic within the processor to control a range of devices.
  - The data transfer rate of peripherals is often much slower than that of the memory or processor. Thus, it is impractical to use the high-speed system bus to communicate directly with a peripheral.
  - On the other hand, the data transfer rate of some peripherals is faster than that of the memory or processor. Again, the mismatch would lead to inefficiencies if not managed properly.
  - Peripherals often use different data formats and word lengths than the computer to which they are attached.

Why I/O Modules and Function?

4 Basic Computer Networks NLU-FIT

- Thus, an I/O module is required. This module has two major functions
  - Interface to the processor and memory via the system bus or central switch
  - Interface to one or more peripheral devices by tailored data links

Why I/O Modules and Function?

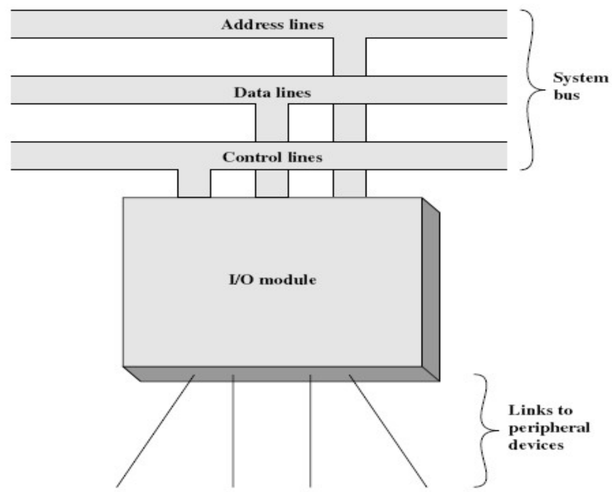


Figure 7.1. Generic Model of an I/O Module

7.1. External Devices

- I/O operations are accomplished through a wide assortment of external devices that provide a means of exchanging data between the external environment and the computer.
- An external device attaches to the computer by a link to an I/O module Figure 7.1.
- The link is used to exchange control, status, and data between the I/O module and the external device.
- An external device connected to an I/O module is often referred to as a *peripheral device or, simply, a peripheral.*

7.1. External Devices

- We can broadly classify external devices into three categories:
  - Human readable: Suitable for communicating with the computer user
    - ✓Screen, printer, keyboard
  - Machine readable: Suitable for communicating with equipment
    - ✓Monitoring and control
    - ✓Magnetic disk, tape systems and sensors
  - Communication: Suitable for communicating with remote devices
    - ✓Modem
    - ✓Network Interface Card (NIC)

7.1. External Devices

- In very general terms, the nature of an external device is indicated in Figure 7.2. The interface to the I/O module is in the form of control, data, and status signals.

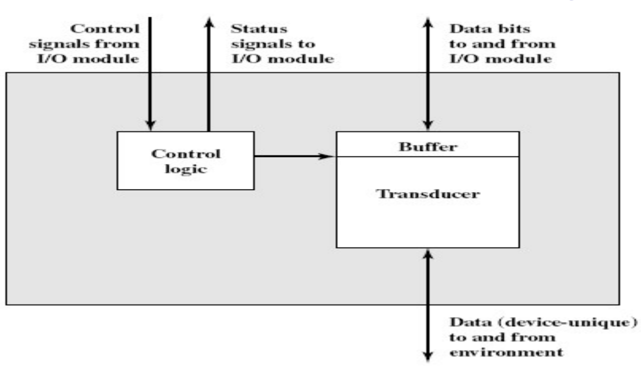


Figure 7.2 Block Diagram of an External Device

7.1. External Devices

- Control signals determine the function that the device will perform, such as
  - Send data to the I/O module (INPUT or READ)
  - accept data from the I/O module (OUTPUT or WRITE), report status
  - or perform some control function particular to the device (e.g., position a disk head).
- Data are in the form of a set of bits to be sent to or received from the I/O module.
- Status signals indicate the state of the device.
  - Examples are READY/NOT-READY to show whether the device is ready for data transfer.

7.1. External Devices

- Control logic associated with the device controls the device's operation in response to direction from the I/O module.
- The transducer converts data from electrical to other forms of energy during output and from other forms to electrical during input.
- Typically, a buffer is associated with the transducer to temporarily hold data being transferred between the I/O module and the external environment a buffer size of 8 to 16 bits is common.

7.1. External Devices

11 Basic Computer Networks NLU-FIT

- **Keyboard/Monitor**
  - The most common means of computer/user interaction is a keyboard/monitor arrangement.
  - The user provides input through the keyboard. This input is then transmitted to the computer and may also be displayed on the monitor.
  - In addition, the monitor displays data provided by the computer.
  - The basic unit of exchange is the character.

7.1. External Devices

12 Basic Computer Networks NLU-FIT

- **Disk Drive**
  - A disk drive contains electronics for exchanging data, control, and status signals with an I/O module plus the electronics for controlling the disk read/write mechanism.
  - In a fixed-head disk, the transducer is capable of converting between the magnetic patterns on the moving disk surface and bits in the device's buffer
  - A moving-head disk must also be able to cause the disk arm to move radially in and out across the disk's surface.

7.2. I/O Module

13 Basic Computer Networks NLU-FIT

- The major functions or requirements for an I/O module fall into the following categories:
  - Control and timing
  - Processor communication
  - Device communication
  - Data buffering
  - Error detection

7.2.1. Control and timing

14 Basic Computer Networks NLU-FIT

- During any period of time, the processor may communicate with one or more external devices in unpredictable patterns, depending on the program's need for I/O.
- The internal resources, such as main memory and the system bus, must be shared among a number of activities, including data I/O.
- Thus, the I/O function includes a control and timing requirement, to coordinate the flow of traffic between internal resources and external devices.
- For example, the control of the transfer of data from an external device to the processor might involve the following sequence of steps:

7.2.1. Control and timing

- 1. The processor interrogates the I/O module to check the status of the attached device.
- 2. The I/O module returns the device status.
- 3. If the device is operational and ready to transmit, the processor requests the transfer of data, by means of a command to the I/O module.
- 4. The I/O module obtains a unit of data from the external device.
- 5. The data are transferred from the I/O module to the processor.

7.2.2. Processor communication

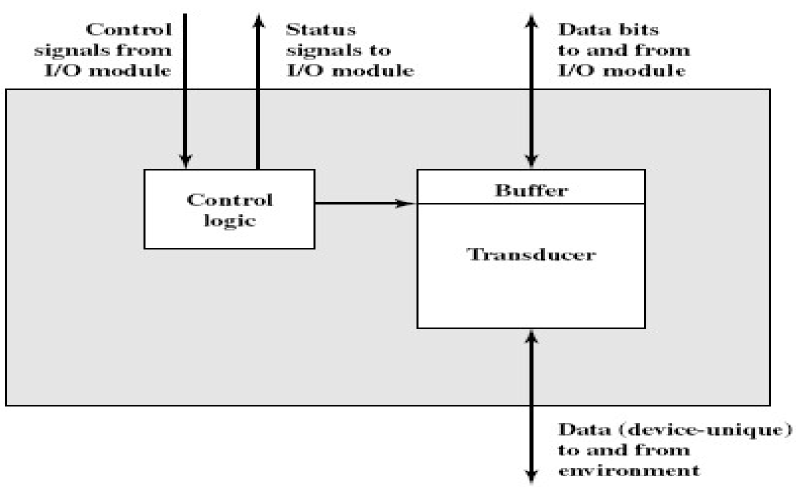
- Processor communication involves the following:
  - **Command decoding**: The I/O module accepts commands from the processor, typically sent as signals on the control bus.
    - ✓For example, an I/O module for a disk drive might accept the following commands: READ SECTOR, WRITE SECTOR, SEEK…
  - **Data**: Data are exchanged between the processor and the I/O module over the data bus

7.2.2. Processor communication

- **Status reporting**: Because peripherals are so slow, it is important to know the status of the I/O module.
  - ✓For example, if an I/O module is asked to send data to the processor (read), it may not be ready to do so because it is still working on the previous I/O command. This fact can be reported with a status signal.
  - ✓Common status signals are BUSY and READY. There may also be signals to report various error conditions.
- **Address recognition**: Just as each word of memory has an address, so does each I/O device. Thus, an I/O module must recognize one unique address for each peripheral it controls.

7.2.3. Device communication

- This communication involves commands, status information, and data as Figure 7.2.



7.2.4. Data buffering

- An essential task of an I/O module is **data buffering.**
  - Whereas the transfer rate into and out of main memory or the processor is quite high, the rate is orders of magnitude lower for many peripheral devices and covers a wide range.
  - Data coming from main memory are sent to an I/O module in a rapid burst. The data are buffered in the I/O module and then sent to the peripheral device at its data rate.

7.2.4. Data buffering

Basic Computer Networks
NLU-FIT

- In the opposite direction, data are buffered so as not to tie up the memory in a slow transfer operation. Thus, the I/O module must be able to operate at both device and memory speeds.
- Similarly, if the I/O device operates at a rate higher than the memory access rate, then the I/O module performs the needed buffering operation

7.2.5. Error detection

- Finally, an I/O module is often responsible for **error detection and for subsequently** reporting errors to the processor.
  - One class of errors includes mechanical and electrical malfunctions reported by the device (e.g., paper jam, bad disk track).
  - Another class consists of unintentional changes to the bit pattern as it is transmitted from device to I/O module.
- Some form of error-detecting code is often used to detect transmission errors.

7.2.6. I/O Module Structure

- I/O modules vary considerably in complexity and the number of external devices that they control.

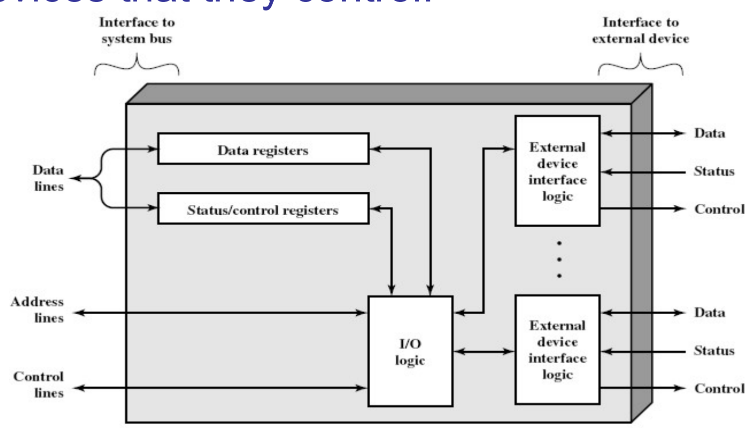


Figure 7.3. Block Diagram of an I/O Module

7.2.6. I/O Module Structure

23 Basic Computer Networks NLU-FIT

- The module connects to the rest of the computer through a set of signal lines (e.g., system bus lines).
- Data transferred to and from the module are buffered in one or more data registers.
- There may also be one or more status registers that provide current status information.
- A status register may also function as a control register, to accept detailed control information from the processor.
- The logic within the module interacts with the processor via a set of control lines.

7.2.6. I/O Module Structure

24 Basic Computer Networks NLU-FIT

- The processor uses the control lines to issue commands to the I/O module.
- Some of the control lines may be used by the I/O module (e.g., for arbitration and status signals).
- The module must also be able to recognize and generate addresses associated with the devices it controls.
- Each I/O module has a unique address or, if it controls more than one external device, a unique set of addresses.
- Finally, the I/O module contains logic specific to the interface with each device that it controls

7.2.6. I/O Module Structure

25 Basic Computer Networks NLU-FIT

- An I/O module that takes on most of the detailed processing burden, presenting a high-level interface to the processor, is usually referred to as an I/O channel or I/O processor.
- An I/O module that is quite primitive and requires detailed control is usually referred to as an ***I/O controller or device controller***. I/O controllers are commonly seen on microcomputers, whereas I/O channels are used on mainframes.

7.3. Programmed I/O

26 Basic Computer Networks NLU-FIT

- Three techniques are possible for I/O operations.
- **With programmed I/O**
  - Data *are* exchanged between the processor and the I/O module.
  - The processor executes a program that gives it direct control of the I/O operation, including sensing device status, sending a read or write command, and transferring the data.
  - When the processor issues a command to the I/O module, it must wait until the I/O operation is complete.
  - If the processor is faster than the I/O module, this is wasteful of processor time.

7.3. Programmed I/O

27 | Basic Computer Networks | NLU-FIT

- **With interrupt-driven I/O**
  - The processor issues an I/O command, continues to execute other instructions, and is interrupted by the I/O module when the latter has completed its work.
  - With both programmed and interrupt I/O, the processor is responsible for extracting data from main memory for output and storing data in main memory for input.
- **With direct memory access (DMA)**
  - The I/O module and main memory exchange data directly, without processor involvement.

7.3. Programmed I/O

28 | Basic Computer Networks | NLU-FIT

I/O Commands

- To execute an I/O-related instruction, the processor issues an address, specifying the particular I/O module and external device, and an I/O command.
- There are four types of I/O commands that an I/O module may receive when it is addressed by a processor:
  - **Control**: Used to activate a peripheral and tell it what to do.
    - ✓For example, a magnetic-tape unit may be instructed to rewind or to move forward one record.

7.3. Programmed I/O

29 Basic Computer Networks NLU-FIT

- **Test**: Used to test various status conditions associated with an I/O module and its peripherals.
  - ✓The processor will want to know that the peripheral of interest is powered on and available for use.
  - ✓It will also want to know if the most recent I/O operation is completed and if any errors occurred.
- **Read:** Causes the I/O module to obtain an item of data from the peripheral and place it in an internal buffer.
  - ✓The processor can then obtain the data item by requesting that the I/O module place it on the data bus.
- **Write:** Causes the I/O module to take an item of data (byte or word) from the data bus and subsequently transmit that data item to the peripheral.

7.3. Programmed I/O

30 Basic Computer Networks NLU-FIT

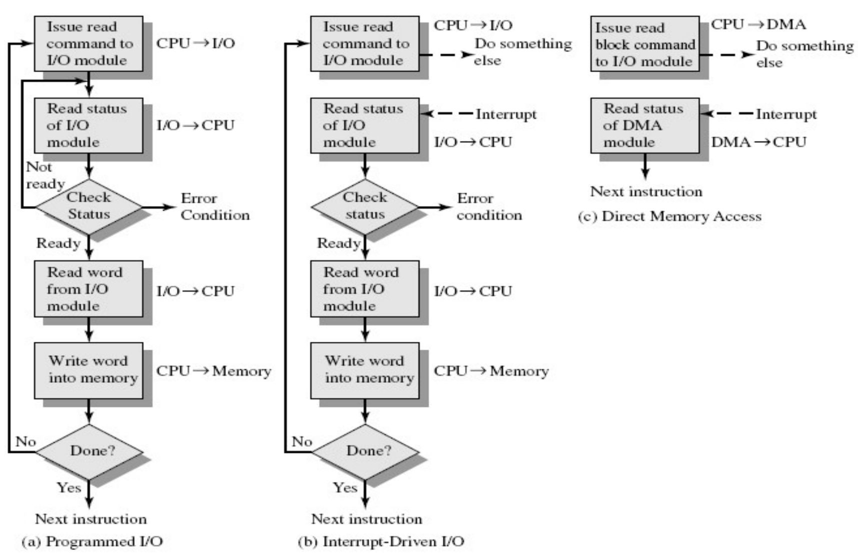


Figure 7.4. Three Techniques for Input of a Block of Data

7.3. Programmed I/O

- Figure 7.4a gives an example of the use of programmed I/O to read in a block of data from a peripheral device (e.g., a record from tape) into memory.
  - Data are read in one word at a time.
  - For each word that is read in, the processor must remain in a status-checking cycle until it determines that the word is available in the I/O module's data register.
  - This flow chart highlights the main disadvantage of this technique: it is a time-consuming process that keeps the processor busy needlessly.

7.4. Interrupt-Driven I/O

- The problem with programmed I/O is that the processor has to wait a long time for the I/O module of concern to be ready for either reception or transmission of data.
  - The processor, while waiting, must repeatedly interrogate the status of the I/O module.As a result, the level of the performance of the entire system is severely degraded.
- An alternative is for the processor to issue an I/O command to a module and then go on to do some other useful work.
  - The I/O module will then interrupt the processor to request service when it is ready to exchange data with the processor.
  - The processor then executes the data transfer, as before, and then resumes its former processing.

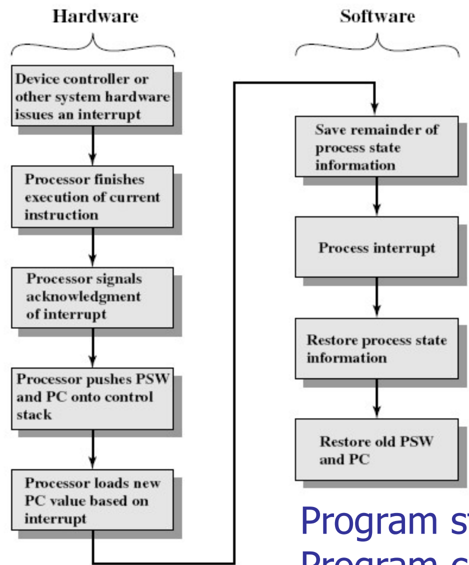
7.4. Interrupt-Driven I/O

- Example
  - Processor issues a READ command. It then goes off and does something else (e.g., the processor may be working on several different programs at the same time).
  - At the end of each instruction cycle, the processor checks for interrupts.
    - ✓When the interrupt from the I/O module occurs, the processor saves the context (e.g., program counter and processor registers) of the current program and processes the interrupt.
    - ✓In this case, the processor reads the word of data from the I/O module and stores it in memory.
  - It then restores the context of the program it was working on (or some other program) and resumes execution.

7.4. Interrupt-Driven I/O

- Figure 7.4b shows the use of interrupt I/O for reading in a block of data.
- Interrupt I/O is more efficient than programmed I/O because it eliminates needless waiting.
- However, interrupt I/O still consumes a lot of processor time
  - because every word of data that goes from memory to I/O module or from I/O module to memory must pass through the processor.

7.4. Interrupt-Driven I/O



Program status word (PSW)
Program counter (PC)

Figure 7.6 Simple Interrupt Processing

7.5. Direct Memory Access

- Interrupt-driven I/O, though more efficient than simple programmed I/O, still requires the active intervention of the processor to transfer data between memory and an I/O module, and any data transfer must traverse a path through the processor.
- Thus, both these forms of I/O suffer from two inherent drawbacks:
  - 1. The I/O transfer rate is limited by the speed with which the processor can test and service a device.
  - 2. The processor is tied up in managing an I/O transfer; a number of instructions must be executed for each I/O transfer

7.5. Direct Memory Access

- DMA involves an additional module on the system bus.
- The DMA module Figure 7.11 is capable of mimicking the processor and, indeed, of taking over control of the system from the processor.
- It needs to do this to transfer data to and from memory over the system bus. For this purpose,
  - The DMA module must use the bus only when the processor does not need it,
  - or it must force the processor to suspend operation temporarily.

7.5. Direct Memory Access

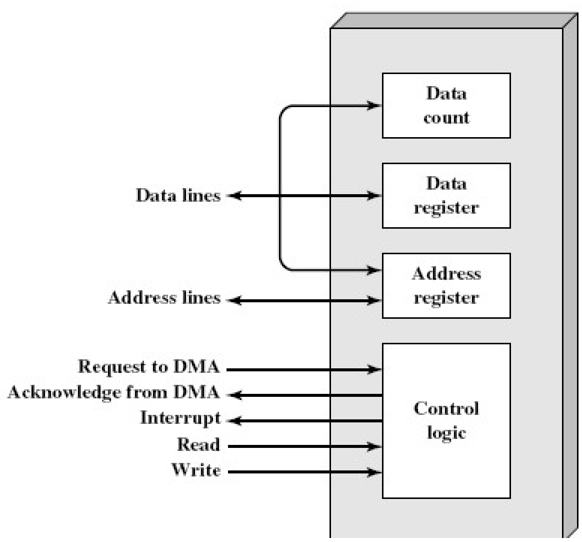


Figure 7.11 Typical DMA Block Diagram

7.5. Direct Memory Access

39 | Basic Computer Networks | NLU-FIT

- When the processor wishes to read or write a block of data, it issues a command to the DMA module, by sending to the DMA module the following information:
  - Whether a read or write is requested, using the read or write control line
  - The address of the I/O device involved
  - The starting location in memory to read from or write to, stored by the DMA module in its address register
  - The number of words to be read or written, again communicated via the data lines and stored in the data count register

7.5. Direct Memory Access

40 | Basic Computer Networks | NLU-FIT

- The processor then continues with other work. It has delegated this I/O operation to the DMA module.
- The DMA module transfers the entire block of data, one word at a time, directly to or from memory, without going through the processor.
- When the transfer is complete, the DMA module sends an interrupt signal to the processor.
- Thus, the processor is involved only at the beginning and end of the transfer (Figure 7.4c).

7.5. Direct Memory Access

41

Basic Computer Networks

NLU-FIT

- The DMA mechanism can be configured in a variety of ways.
- Some possibilities are shown in Figure 7.13.
  - In the first example
    - ✓all modules share the same system bus.
    - ✓The DMA module, acting as a surrogate processor, uses programmed I/O to exchange data between memory and an I/O module through the DMA
    - ✓This configuration, while it may be inexpensive, is clearly inefficient. As with processor-controlled programmed I/O, each transfer of a word consumes two bus cycles.

7.5. Direct Memory Access

42

Basic Computer Networks

NLU-FIT

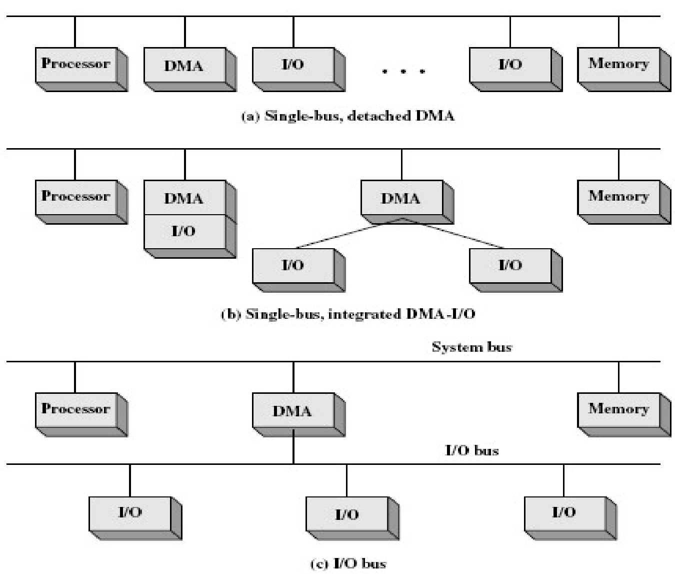


Figure 7.13. Alternative DMA Configurations

7.5. Direct Memory Access

43

Basic Computer Networks

NLU-FIT

- In both of these cases (Figures 7.13b and c),
  - The system bus that the DMA module shares with the processor and memory is used by the DMA module only to exchange data with memory.
  - The exchange of data between the DMA and I/O modules takes place off the system bus.

Computer Organization and Architecture

44

Basic Computer Networks

NLU-FIT

7.6. I/O Channels and Processor (Reference)

7.7. The External Interface (Reference)

- ***Reference: Computer Organization and Architecture Designing for Performance (8th Edition), William Stallings, Prentice Hall, Upper Saddle River, NJ 07458.***