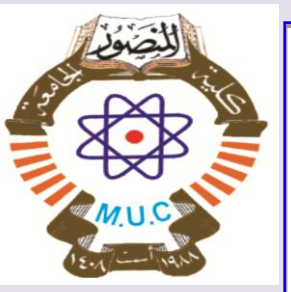
قسم
هندسة تقنيات
(الشبكات & الأتمتة)
المرحلة الثالثة



الجامعة
المنصور
كلية

Real-Time System Design Lectuers

2015 - 2016

Lec.4

الزمن الحقيقي

Lec.
Noor Kareem

500

Basic Interface Devices

Fig. (1) below shows a single CPU attached to the main memory of the system via some kind of **memory bus** or interconnection. Some devices are connected to the system via a general **I/O bus** which in many modern systems would be **PCI** (or one of its many derivatives); finally, even lower down are one or more of what we call a **peripheral bus**, such as **SCSI**, **SATA**, or **USB**. These connect the slowest devices to the system, including **disks**, **mice**, and other similar components.

A hierarchical structure like this will put simply: physics, and cost. The faster a bus is, the shorter it must be; thus, a high-performance memory bus does not have much room to plug devices and such into it. In addition, engineering a bus for high performance is quite costly. Thus, system designers have adopted this hierarchical approach, where components that demand high performance (such as the graphics card) are nearer the CPU.

Lower performance components are further away. The benefits of placing disks and other slow devices on a peripheral bus are manifold; in particular, you can place a large number of devices on it.

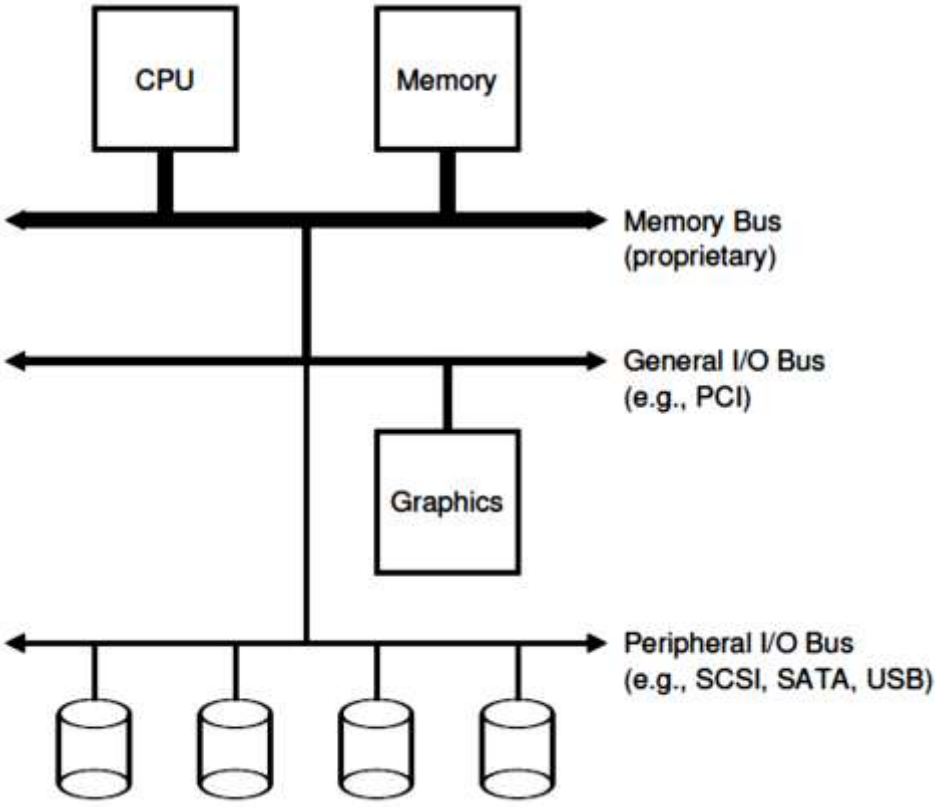


Fig. (1): Prototypical System Architecture.

Bus Interface

A **bus** is a shared communication link, which uses one set of wires to connect multiple subsystems. The two major advantages of the bus organization are *versatility* and *low cost*.

Fig. (2) shows the bus interface.

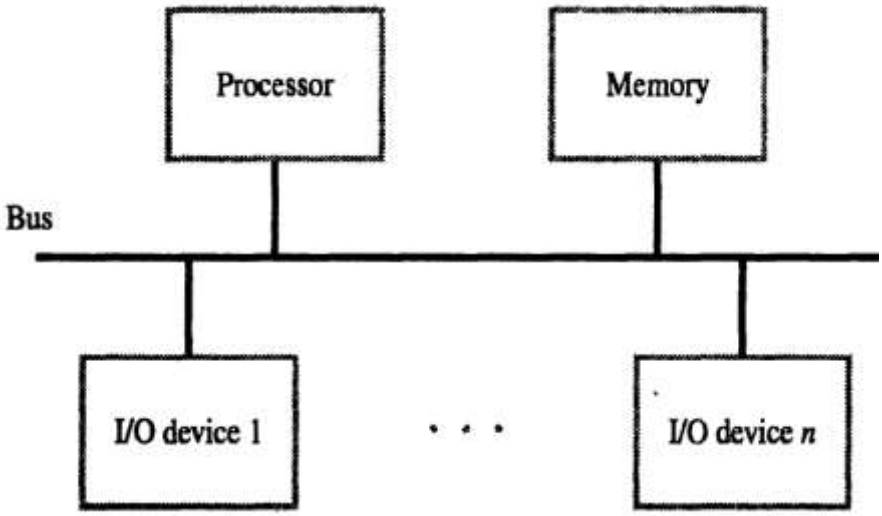


Fig. (2): Bus Interface.

Accessing I/O Devices

- Most modern computers use single bus arrangement for connecting I/O devices to CPU & Memory.
- The bus enables all the devices connected to it to exchange information.
- Bus consists of 3 set of lines: *Address, Data, and Control.*
- Processor places a particular address (unique for an I/O Dev.) on address lines.
- Device which recognizes this address responds to the commands issued on the Control lines.
- Processor requests for either Read / Write.
- The data will be placed on Data lines.

Connect an I/O Device with the Bus

Interface Circuit
– Address Decoder
– Control Circuits
– Data registers
– Status registers
• The Registers in I/O Interface – buffer and control
• Flags in Status Registers, like SIN, SOUT
• Data Registers, like Data-IN, Data-OUT
Fig. (3) shows the connection of an input device to the bus.

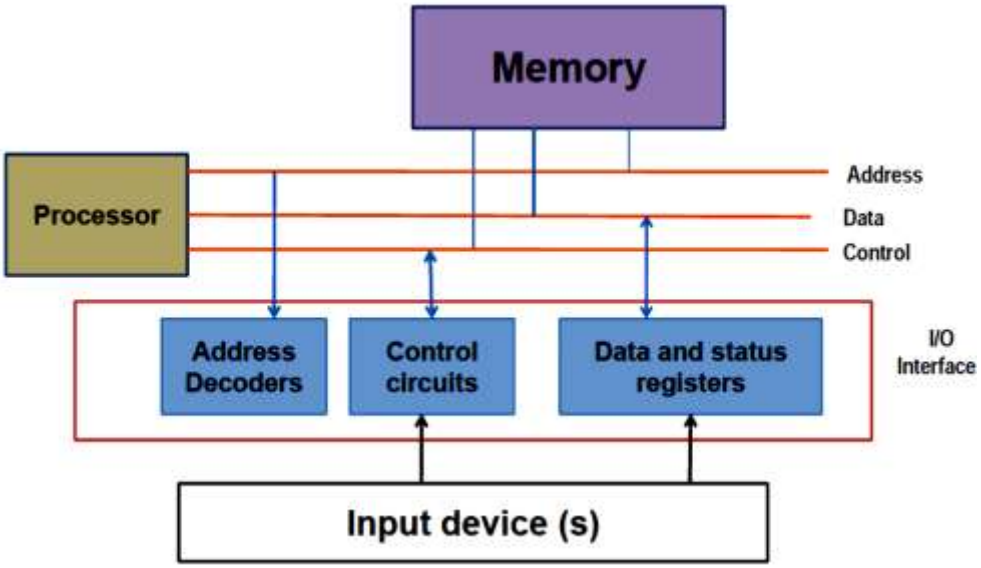


Fig. (3): Input device connection with the bus.

Input/Output Mechanism

- A bus generally contains a set of control lines and a set of data lines.
- The control lines are used to signal requests and acknowledgments, and to indicate what type of information is on the data lines. The control lines are used to indicate what the bus contains and to implement the bus protocol.
- The data lines of the bus carry information between the source and the destination. This information may consist of data, complex commands, or addresses.
- Buses are traditionally classified as processor-memory buses or I/O buses or special purposed buses (Graphics, etc.). Processor memory buses are short, generally high speed, and matched to the memory system so as to maximize memory processor bandwidth.
- I/O buses, by contrast, can be lengthy, can have many types of devices connected to them, and often have a wide range in the data bandwidth of the devices connected to them. I/O buses do not typically interface directly to the memory but use either a processor-memory or a backplane bus to connect to memory.

The major disadvantage of a bus is that it creates a communication bottleneck, possibly limiting the maximum I/O throughput.

When I/O must pass through a single bus, the bandwidth of that bus limits the maximum I/O throughput.

Reason why bus design is so difficult:

The maximum bus speed is largely limited by physical factors: the length of the bus and the number of devices. These physical limits prevent us from running the bus arbitrarily fast.

In addition, the need to support a range of devices with widely varying latencies and data transfer rates also makes bus design challenging.

It becomes difficult to run many parallel wires at high speed due to clock skew and reflection.

The basic schemes of the Bus

The two basic schemes for communication on the bus are **synchronous** and **asynchronous**.

If a bus is **synchronous** (e.g. Processor-memory), it includes a clock in the control lines and a fixed protocol for communicating that is relative to the clock. This type of protocol can be implemented easily in a small finite state machine. Because the protocol is predetermined and involves little logic, the bus can run very fast and the interface logic will be small.

Synchronous buses have two major disadvantages:

1. Every device on the bus must run at the same clock rate.
2. Second, because of clock skew problems, synchronous buses cannot be long if they are fast.

An **asynchronous** bus is not clocked. It can accommodate a wide variety of devices, and the bus can be lengthened without worrying about clock skew or synchronization problems. To coordinate the transmission of data between sender and receiver, an asynchronous bus uses a handshaking protocol.