BUS ARCHITECTURES

Lizy Kurian John
Electrical and Computer Engineering Department, The University of Texas as Austin

Keywords: Bus standards, PCI bus, ISA bus, Bus protocols, Serial Buses, USB, IEEE 1394

Contents

Summary

A bus is a common pathway to connect various subsystems in a computer system. A bus consists of the connection media like wires and connectors, and a bus protocol. Buses can be serial or parallel, synchronous or asynchronous. Depending on these and other features, several bus architectures have been devised in the past. The Universal Serial Bus (USB) and IEEE 1394 are examples of serial buses while the ISA and PCI buses are examples of popular parallel buses. This article first describes fundamental information on bus architectures and bus protocols, and then provides specific information on various industry standard bus architectures from the past and the present, and their advantages and disadvantages. It also describes how different types of bus architectures are used simultaneously in different parts of a modern personal computer.

1. Introduction

A typical computer system is composed of several components such as the Central Processing Unit (CPU), memory chips, and Input/Output (I/O) devices. A bus is a common pathway or a set of wires that interconnect these various subsystems. The bus thus allows the different components to communicate with each other. The concept of a bus is illustrated in Figure 1.

A bus, in computer language, is a channel over which information flows between units or devices. It typically has access points, or places into which a device can tap to become part of the channel. Most buses are bidirectional and devices can send or receive information. A bus is a shared communication link between the different

devices. It allows to add new devices easily and facilitates portability of peripheral devices between different computer systems. However, if too many devices are connected to the same bus, the bandwidth of the bus can become a bottleneck. Typically more than two devices or subsystems are involved in a bus, and channels connecting only two components are some times referred to as ports instead of buses.

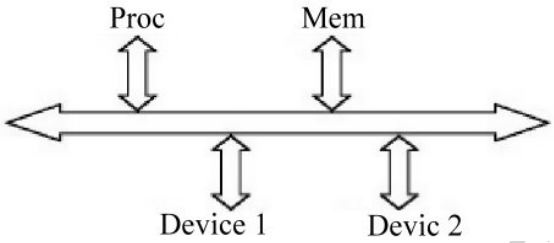

Figure 1: The concept of a bus

Buses often include wires to carry signals for addresses, data, control, status, clock, power and ground as illustrated in Figure 2. The address lines indicate the source or destination of the data on the data lines. Control lines are used to implement the bus protocol. Often there are lines to request bus control, to handle interrupts, etc. Status lines indicate the progress of the current transaction. Clock signals are used in synchronous bus systems to synchronize bus operations.

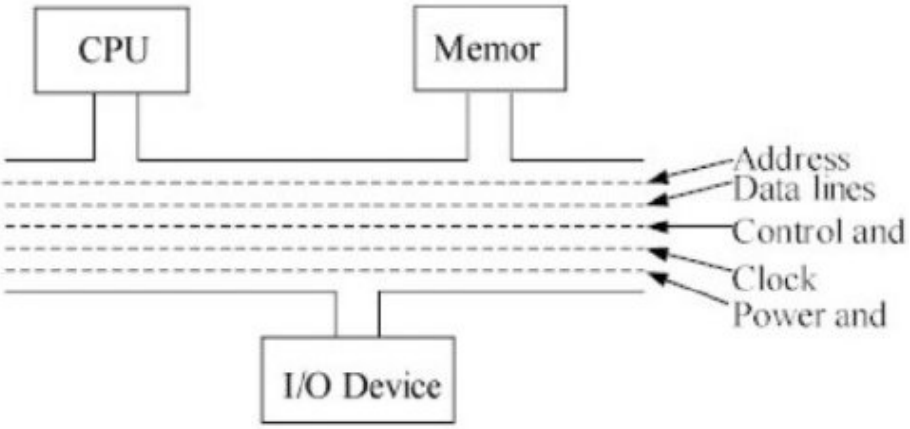

Figure 2: Different kinds of wires in a bus

The communications needs of the different devices in a computer system vary. For instance, fast high bandwidth communication is needed between processors and memory whereas bandwidth requirements are not so high on buses to I/O devices. This has led to the creation of different kinds of buses differing in their width, latency and bandwidth capabilities. Typically there are two kinds of buses, CPU-memory buses and I/O buses. CPU-memory buses are fast and short, and I/O buses are typically long and slow. I/O buses also have many devices connected to them. Some refer to this arrangement as a bus hierarchy with fast buses closer to the processor and slow buses farther away from the processor. Figure 2 illustrates the bus hierarchy in a typical computer that uses the Pentium II processor.

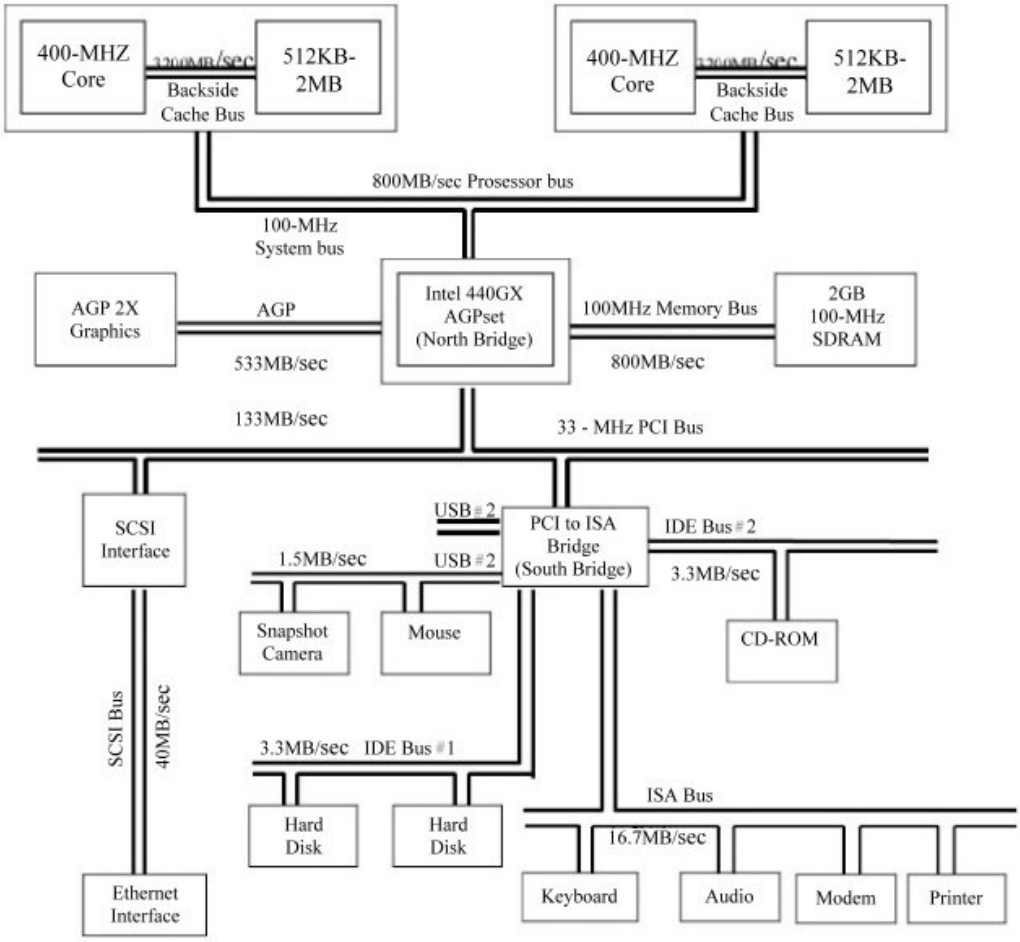

Figure 3: Buses in a typical personal computer system with Intel Pentium Processor

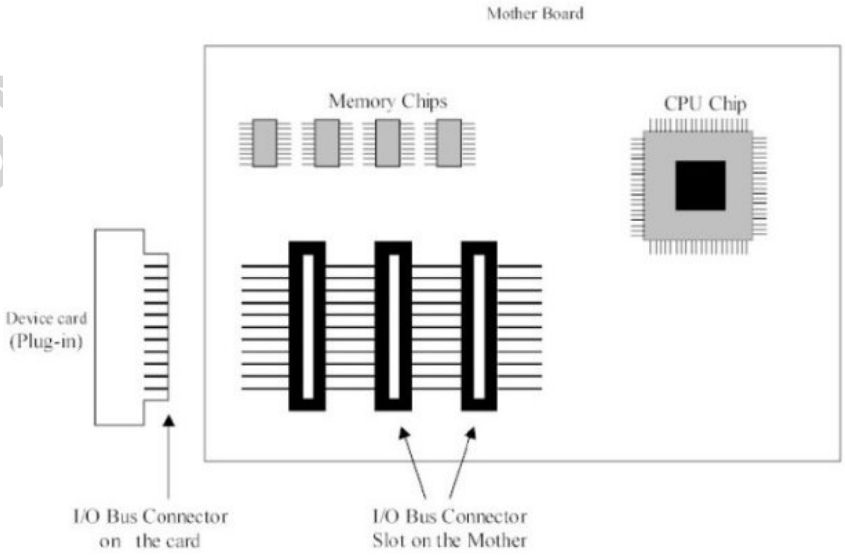

Figure 4. Simplified sketch of I/O device slots on a motherboard

The I/O buses in a personal computer are often etched on to a printed circuit board, which has connector slots to insert the peripheral device card. The I/O device cards have tabs that will fit into the connector, as illustrated in Fig. 4. The tab on the I/O card has metallic strips on each side to make electrical contact with the bus.

2. Bus Protocols

A bus is a communication channel shared by many devices and hence rules need to be established in order for the communication to happen correctly. These rules are called bus protocols. Design of a bus architecture involves several tradeoffs related to the width of the data bus, data transfer size, bus protocols, clocking, etc. Depending on whether the bus transactions are controlled by a clock or not, buses are classified into synchronous and asynchronous buses. Depending on whether the data bits are sent on parallel wires or multiplexed onto one single wire, there are parallel and serial buses. Control of the bus communication in the presence of multiple devices necessitates defined procedures called arbitration schemes. In this section, different kinds of buses and arbitration schemes are described.

2.1. Synchronous and Asynchronous Buses

In a synchronous bus, bus operations are synchronized with reference to a clock signal. The bus clock is generally derived from the computer system clock, however, often it is slower than the master clock. For instance, 66MHz buses are used in systems with a processor clock of over 500MHz. Buses were traditionally slower than processors because memory access times are typically longer than processor clock cycles. A bus transaction often takes several clock cycles, although the cycles are collectively referred to by many as a bus cycle.

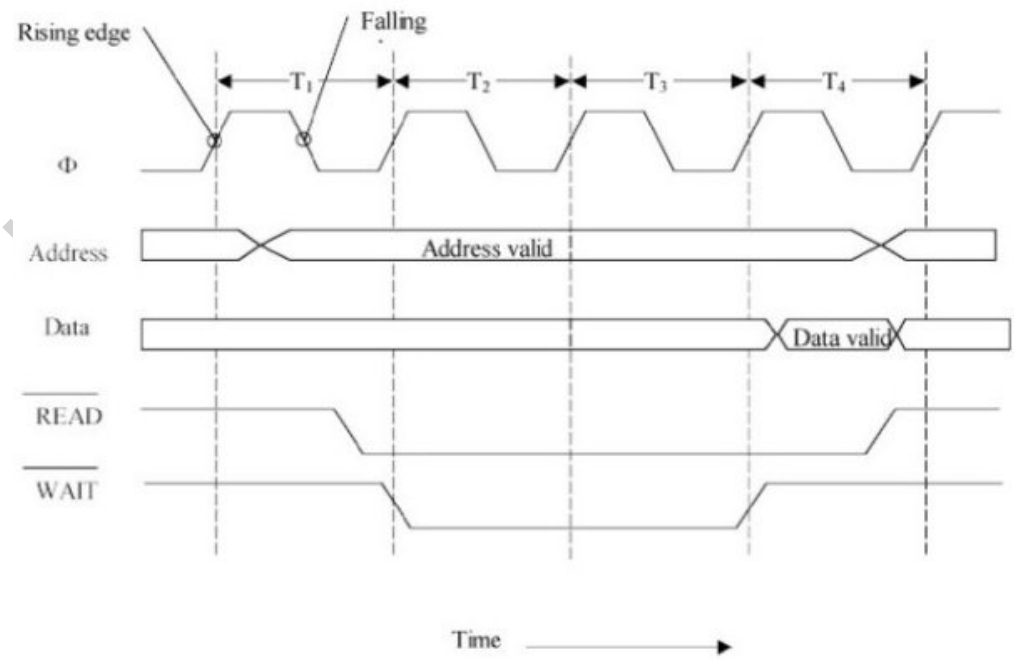

Figure 5. Read Operation on a Synchronous Bus

A memory read transaction on the synchronous bus typically proceeds as illustrated in Fig. 5. During the first clock cycle the CPU places the address of the location it wants to read, on the address lines of the bus. Later during the same clock cycle, once the address lines have stabilized, the READ request is asserted by the CPU. Many times, some of these control signals are active low and asserting the signal means that they are pulled low. A few clock cycles are needed for the memory to perform accessing of the requested location. In a simple non-pipelined bus, these appear as wait states and the data is placed on the bus by the memory after the tow or three wait cycles. The CPU then releases the bus by deasserting the READ control signal. The write transaction is similar except that the processor is the data source and the WRITE signal is the one that is asserted. Different bus architectures synchronize bus operations with respect to the rising edge or falling edge or level of the clock signal.

An asynchronous bus has no system clock. Handshaking is done to properly conduct the transmission of data between the sender and the receiver. The process is illustrated in Fig. 6. For example, in an asynchronous read operation, the bus master puts the address and control signals on the bus and then asserts a synchronization signal. The synchronization signal from the master prompts the slave to get synchronized and once it has accessed the data, it asserts its own synchronization signal. The slave's synchronization signal indicates to the processor that there is valid data on the bus, and it reads the data. The master then deasserts its synchronization signal, which indicates to the slave that the master has read the data. The slave then deasserts its synchronization signal. This method of synchronization is referred to as a full handshake. Note that there is no clock and that starting and ending of the data transfer are indicated by special synchronization signals. An asynchronous communication protocol can be considered as a pair of Finite State machines (FSMs) that operate in such a way that one FSM does not proceed until the other FSM has reached a certain state.

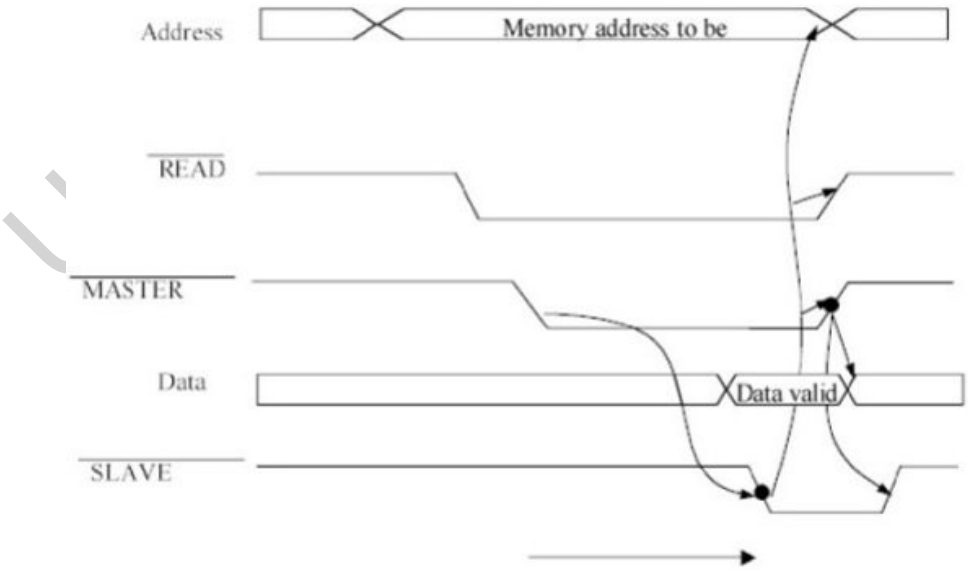

Figure 6. Read Operations on an Asynchronous Bus

Synchronous buses are typically faster than asynchronous buses because there is no overhead to establish a time reference for each transaction. Another reason that helps the synchronous bus to operate fast is that the bus protocol is predetermined and very little logic is involved in implementing the Finite State machine. However, synchronous buses are affected by clock skew and they cannot be very long. But asynchronous buses work well even when they are long because clock skew problems do not affect them. Thus asynchronous buses can handle longer physical distances and higher number of devices. Processor-memory buses are typically synchronous because the devices connected to the bus are fast, are small in number and are located in close proximity. I/O buses are typically asynchronous because many peripherals need only slow data rates and are physically situated far away.

2.2. Serial and Parallel Buses

Buses that transfer several data bits at the same are called parallel buses. It is desirable to have wide buses because large chunks of data can be transferred quickly when multiple lines can be used. Parallel buses typically have 8, 16, 32 or 64 data lines. The ISA, PCI, VESA, and EISA buses are examples of parallel buses. Serial buses use the same line to transfer different data bits of the same byte/word. Typically they have only one data line and the bits are sent one after the other, as a packet. The Universal Standard Bus (USB) and IEEE 1394 bus architecture are examples of serial buses. Serial buses are less expensive than parallel buses, however, parallel buses have higher throughput.

-
-
-

Bibliography

Don Anderson, John Swindle and Tom Shanley (1995), *ISA System Architecture*, MindShare, Addison Wesley Publishing Company, ISBN 0-201-40996-8 [This book is a detailed reference on the ISA architecture]

Don Anderson (1997), *Universal Serial Bus Architecture*, MindShare Inc., Addison Wesley Publishing Company, 1997 [This book provides details on the USB architecture]

Don Anderson (1995), *PCMCIA system Architecture*, Addison Wesley Publishing Company, [This is a detailed reference for the PCMCIA interface]

Don Anderson (1998), *FireWire System Architecture: IEEE 1394a*, Addison Wesley Publishing Company, [This is a detailed reference for the IEEE 1394 serial bus architecture]

John L. Hennessy and David A. Patterson (1996), *Computer Architecture: A Quantitative Approach*, Morgan Kaufman Publishers, San Francisco, California. [Chapter 6 of this book discusses some basic information on buses]

Miles J. Murdocca and Vincent P. Heuring (2000), *Principles of Computer Architecture*, Prentice Hall,

Upper Saddle River, New Jersey, ISBN 0-201-43664-7 [Chapter 8 of this book describes operation of I/O buses]

David A. Patterson and John L. Hennessy (1997), *Computer Organization and Design: The Hardware Software Interface,* Morgan Kaufman Publishers, ISBN 1-55860-428-6, Second Edition [Chapter 8 of this book describes some basic information on operation of buses]

Tom Shanley and Don Anderson (1999), *PCI System Architecture*, MindShare, Addison Wesley Publishing Company, ISBN 0-201-40993-3 [This book describes the PCI bus architecture]

Tom Shanley and Don Anderson (1995), *Plug and Play System Architecture,* Addison Wesley Publishing Company, [This book describes the concepts behind plug and play devices and the interface required for it]

Bruce Shriver and Bennet Smith (1998), *The Anatomy of a High Performance Microprocessor: A Systems Perspective*, IEEE Computer Society, 1998, ISBN 0-8186-8400-3 [This book has descriptions of several bus standards such as PCI, ISA, USB, etc. and it also provides pointers to more information on these bus architectures. The book comes with a CDROM which contains elaborate descriptions on the buses and reprints of articles]

Edward Solari, *ISA and EISA Theory and Operation*, Anna Books, ISBN 0-929392-15-9

[Provides a detailed reference for the ISA and EISA buses]

Edward Solari and George Willse, *PCI Hardware and Software Architecture and Design*,

Anna Books, ISBN 0-929392-59-0 [Provides a detailed reference for the PCI bus]

Andrew S. Tanenbaum (1999), *Structured Computer Organization*,Prentice Hall, Upper Saddle River, New Jersey, ISBN 0-13-095990-1 [Section 3.6 in the book provides fundamental operation of buses]

http://www.techfest.com/hardware/bus.htm [This website has compiled a large amount of information on all bus architectures]

http://www.scsita.org/ [This is the web site of the SCSI Trade Association and provides upto date information on the newest revisions in the SCSI standard]