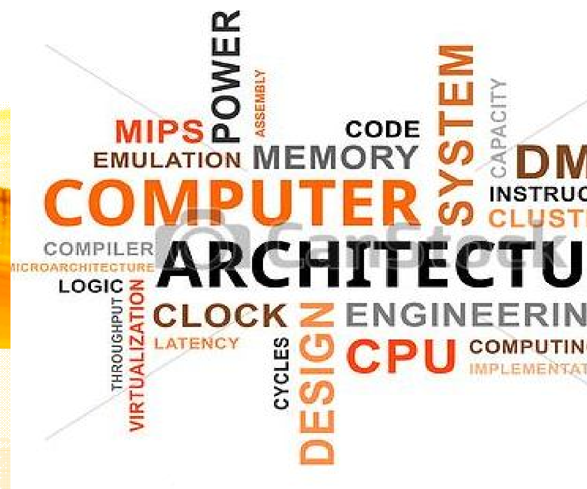
UNIT V
I/O ORGANIZATION AND PARALLELISM

Accessing I/O devices – Interrupts – Direct Memory Access – **Buses–**Interface circuits – Standard I/O Interfaces (PCI, SCSI, USB) –Instruction Level Parallelism : Concepts and Challenges – Introduction to multicore processor – Graphics Processing Unit



Recap the previous Class

Overview

The **primary function** of a bus is to provide a communications path for the transfer of data.

A bus protocol is the set of rules that govern the **behavior of various devices connected to the bus** as to when to place information on the bus, assert control signals, etc.

Three types of bus lines: data, address, control

The bus control signals also carry timing information.

Buses – Common Characteristics

- Multiple devices communicating over a single set of wires
- Only one device can talk at a time or the message is garbled
- Each line or wire of a bus can at any one time contain a single binary digit.
- These lines may and often do send information in parallel
- A computer system may contain a number of different buses

Buses – Structure

Serial versus parallel

Bus lines (parallel)

- Data , Address , Control & Power

Bus lines (serial)

- Data, address, and control are sequentially sent down single wire
- There may be additional control lines
- Power

Buses – Structure (continued)

Data Lines

- Passes data back and forth
- Number of lines **represents width**

Address lines

- Designates location of source or destination
- Width of address bus **specifies maximum memory capacity**
- High order **selects module** and low order **selects a location** within the modul

Bus Structure – Control lines

- Because multiple devices communicate on a line, **control is needed**
- Timing
- Typical lines include:
  - Memory Read or Write , I/O Read or Write
  - Transfer ACK
  - Bus request ,Bus grant
  - Interrupt request , Interrupt acknowledgement
  - Clock , Reset

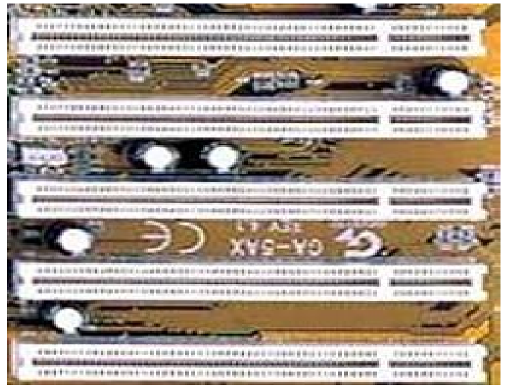
Operation – Sending Data

- Obtain the use of the bus
- Transfer the data via the bus
- Possible acknowledgement

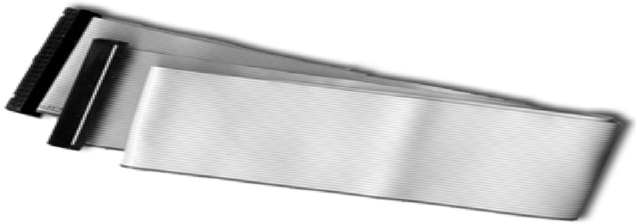
Operation – Requesting Data

- Obtain the use of the bus
- Transfer the data request via the bus
- Wait for other module to send data
- Possible acknowledgement

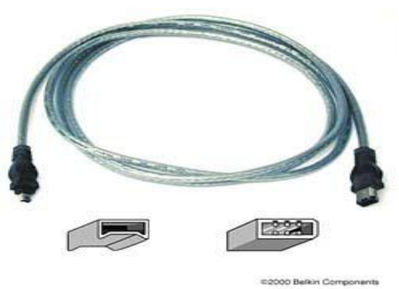
Physical Implementations



- Parallel lines on circuit boards (ISA or PCI)
- Ribbon cables (IDE)



- Strip connectors on mother boards (PC104)
- External cabling (USB or Firewire)



Single Bus Problems

ots of devices on one bus leads to:

Physically long buses

- **Propagation delays** – Long data paths mean that co-ordination of bus use can adversely affect performance
- **Reflections/termination problems**

Aggregate data transfer approaches bus capacity

Slower devices dictate the maximum bus speed

Multiple Buses

Most systems use multiple buses to overcome these problems

Requires bridge to buffer (FIFO) data due to differences in bus speeds

Sometimes I/O devices also contain buffering (FIFO)

Multiple Buses – Benefits

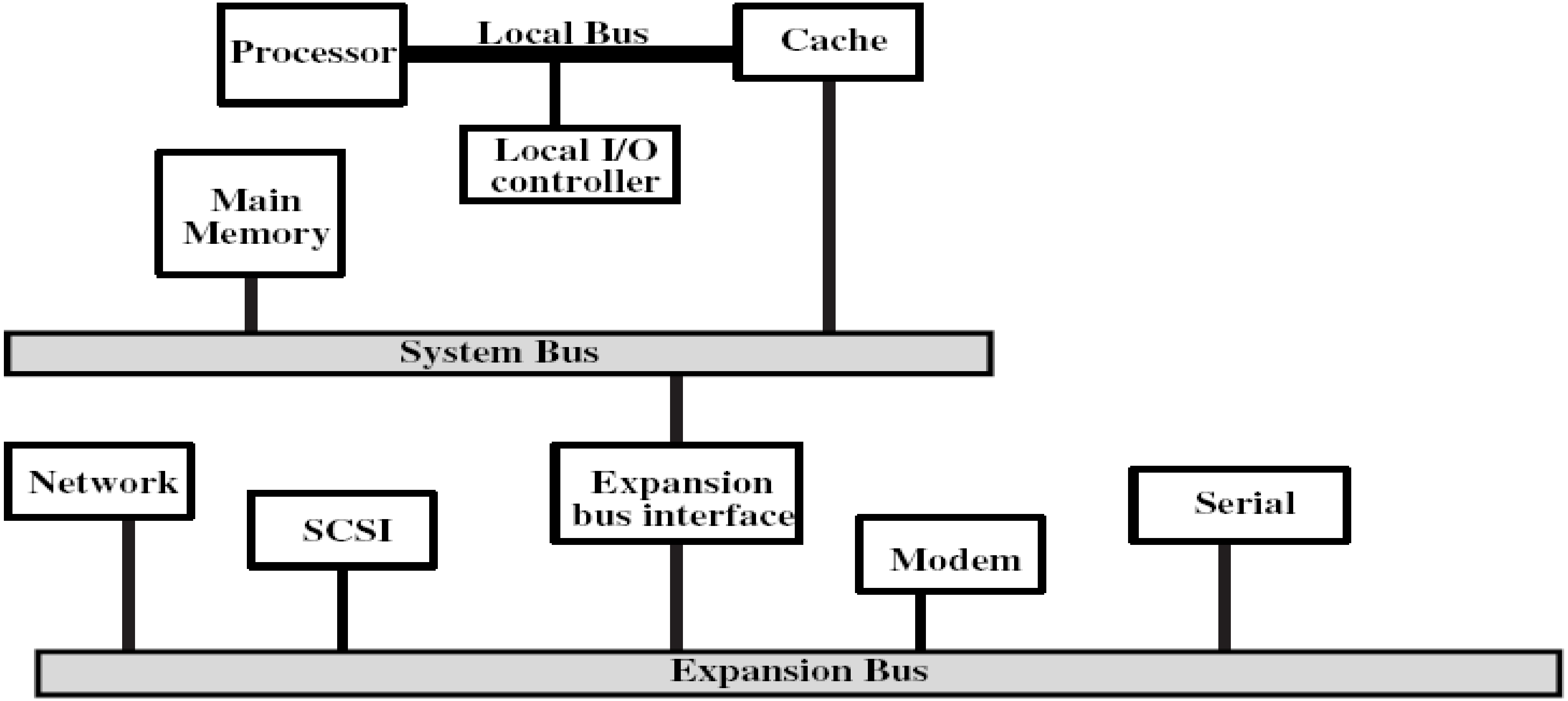
Isolate processor-to-memory traffic from I/O traffic

Support wider variety of interfaces

Processor has bus that connects as **direct interface to chip**, then an expansion bus interface interfaces it **to external devices** (ISA)

Cache (if it exists) may act as the interface to system bus

Expansion Bus Example



(a) Traditional Bus Architecture

Bus Arbitration

The device that is allowed to initiate data transfers on the bus at any given time is called the **bus master.**

Bus arbitration is the process by which the next device to become the bus master is selected and bus mastership is transferred to it.

Need to establish a priority system.

Talking on the bus is a problem – need arbitration to allow more than one module to control the bus at one time

Arbitration may be centralised or distributed

Centralised Arbitration

- Single hardware device controlling bus access – Bus Controller / Arbiter
- May be part of CPU or separate

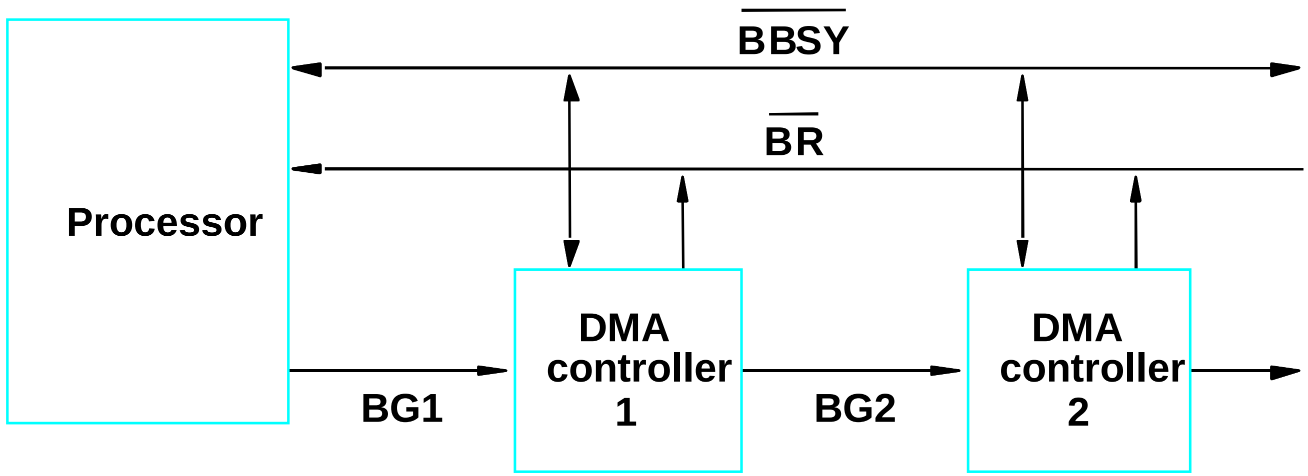


Figure:. A simple arrangement for bus arbitration using a daisy chain.

Distributed Arbitration

- Each module may claim the bus , Access control logic is on all modules
- Modules work together to control bus

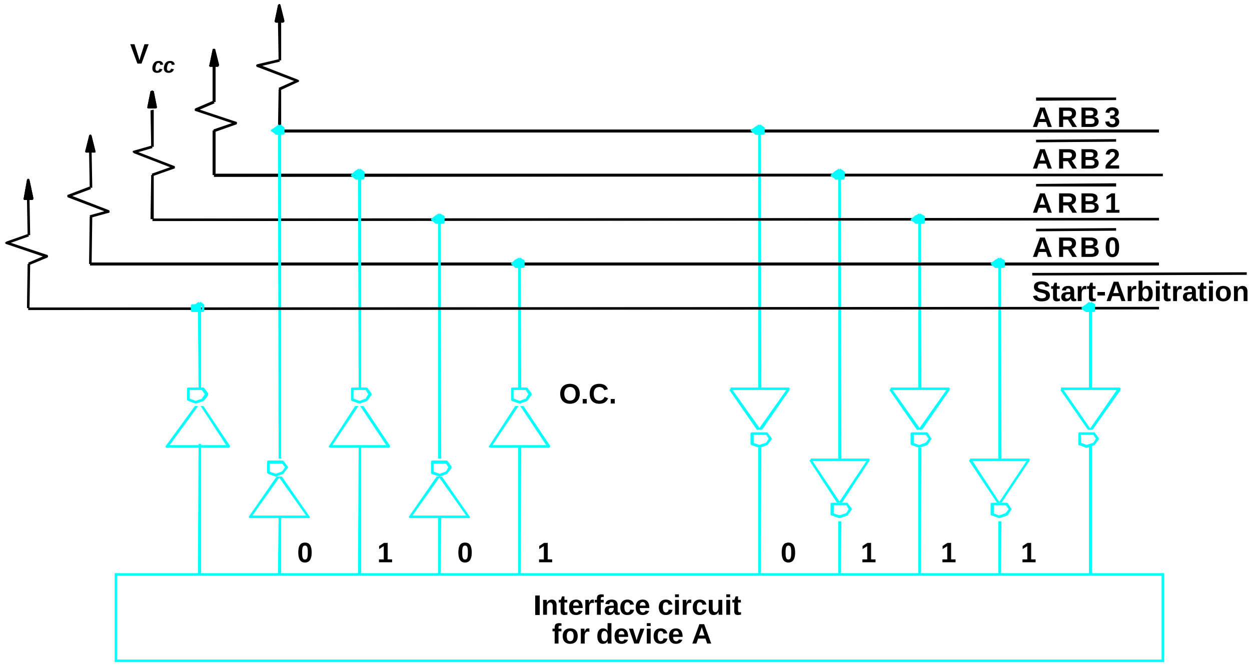


Figure. A distributed arbitration scheme.

Bus Timing

- **Synchronous** – controlled by a clock
- **Asynchronous** – timing is handled by **well-defined specifications**, i.e., a response is delivered within a specified time after a request

Synchronous Bus Timing

Events determined by clock signals

Control Bus includes clock line

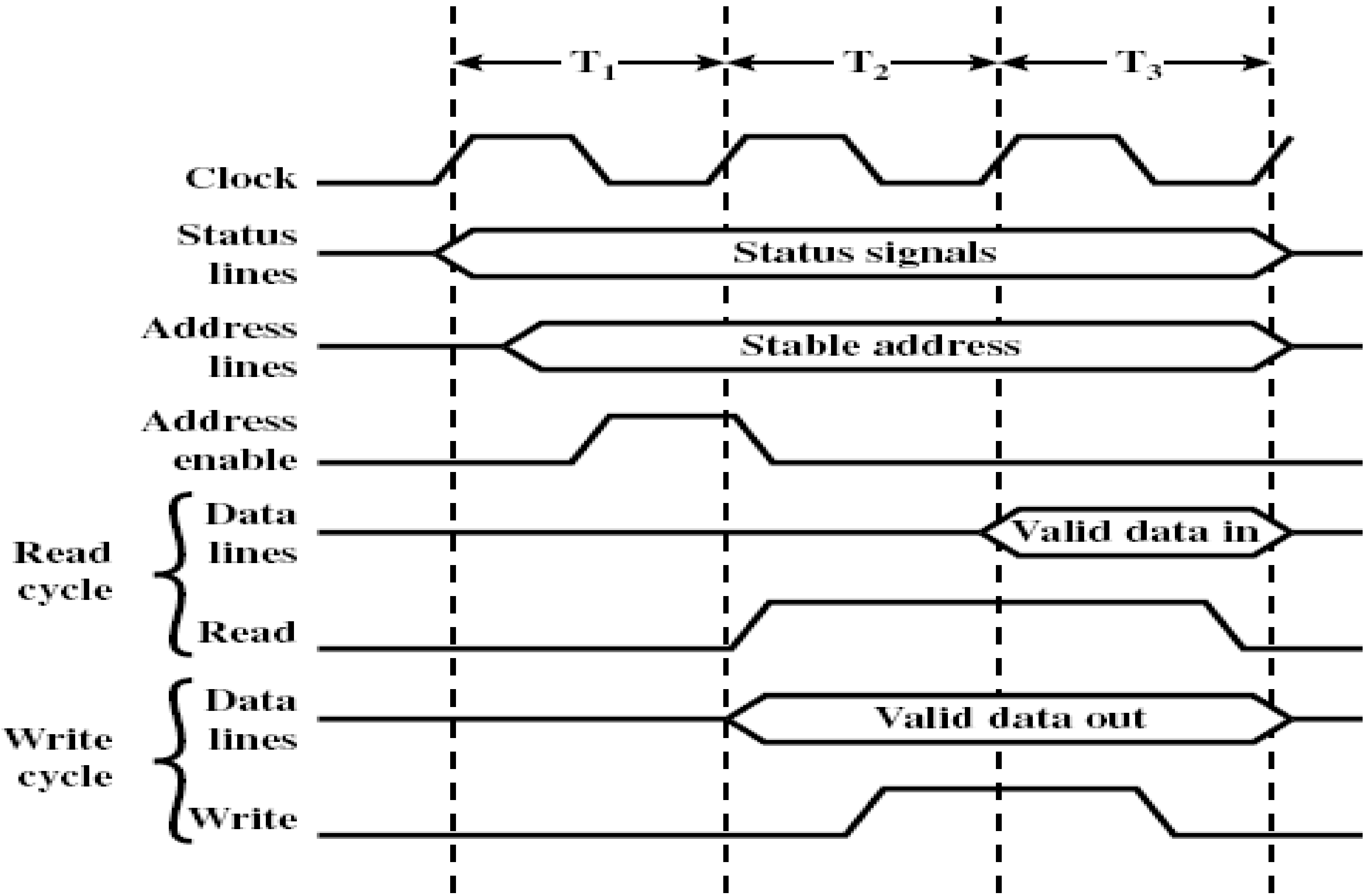
A single 1-0 cycle is a bus cycle

All devices can read clock line

Usually sync on **leading/rising edge** , a **single cycle** for an event

Usually stricter in terms of its timing requirements

Synchronous Bus Timing



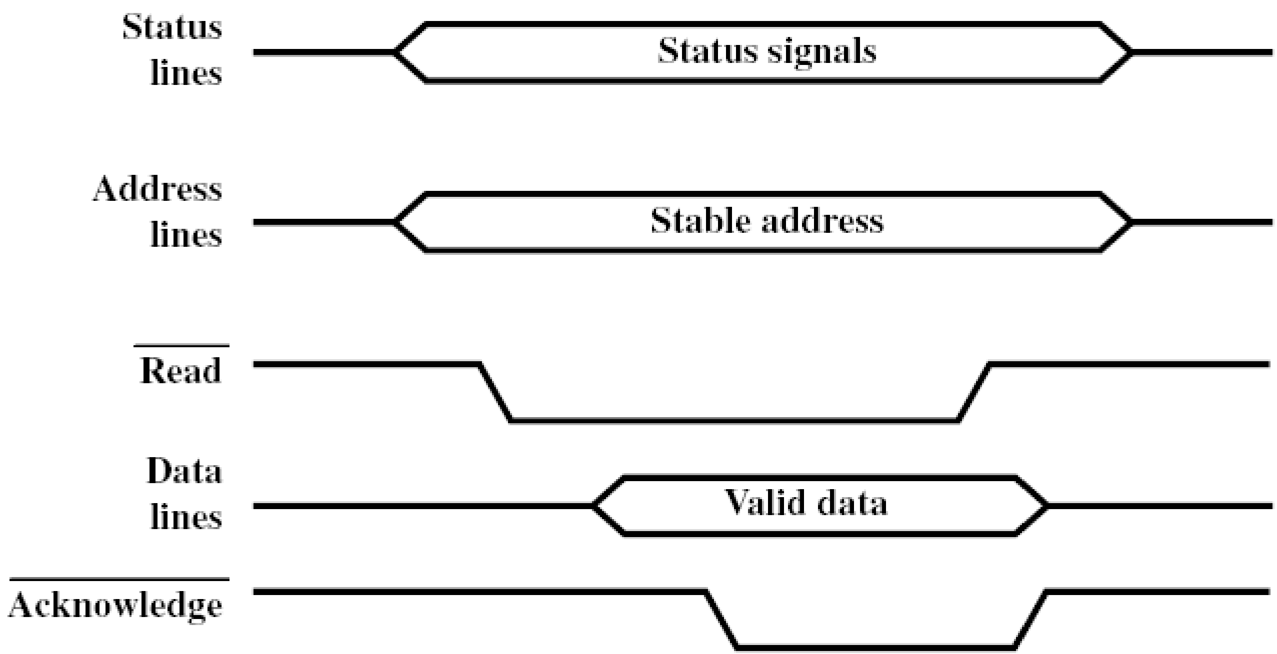
Asynchronous Timing

Devices must have certain tolerances to provide responses to signal stimuli

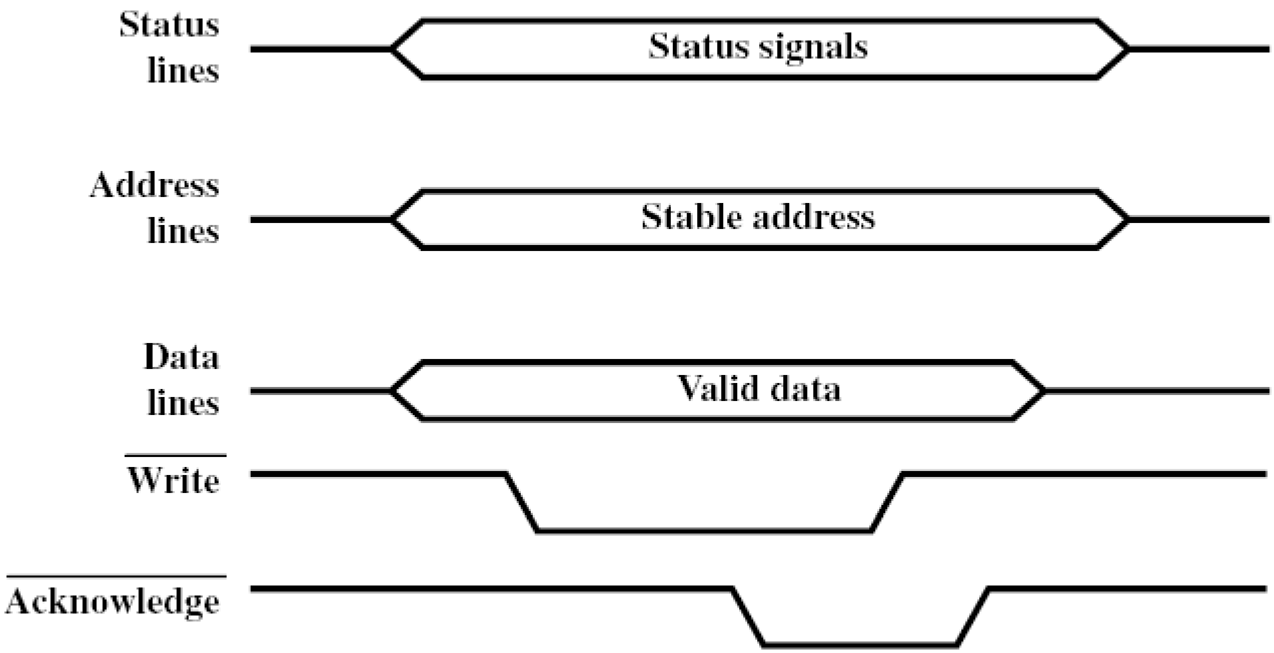
More flexible allowing slower devices to communicate on same bus with faster devices.

Performance of faster devices, however, is limited to speed of bus

Asynchronous Timing – Read



Asynchronous Timing – Write



Bus Width

- Wider the bus the better the data transfer rate or the wider the addressable memory space
- Serial “width” is determined by length/duration of frame

TEXT BOOK

Carl Hamacher, Zvonko Vranesic and Safwat Zaky, “Computer Organization”, McGraw-Hill, 6th Edition 2012.

REFERENCES

1. David A. Patterson and John L. Hennessey, “Computer organization and design”, MorganKauffman ,Elsevier, 5th edition, 2014.
2. William Stallings, “Computer Organization and Architecture designing for Performance”, Pearson Education 8th Edition, 2010
3. John P.Hayes, “Computer Architecture and Organization”, McGraw Hill, 3rd Edition, 2002
4. M. Morris R. Mano “Computer System Architecture” 3rd Edition 2007
5. David A. Patterson “Computer Architecture: A Quantitative Approach”, Morgan Kaufmann; 5th edition 2011

THANK YOU