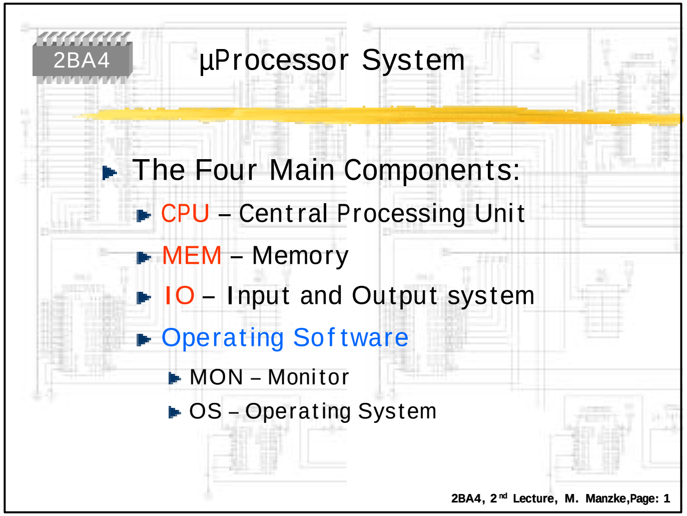
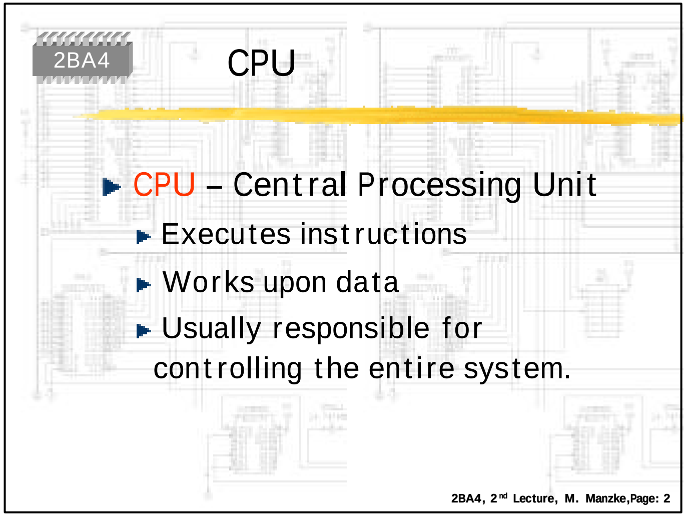


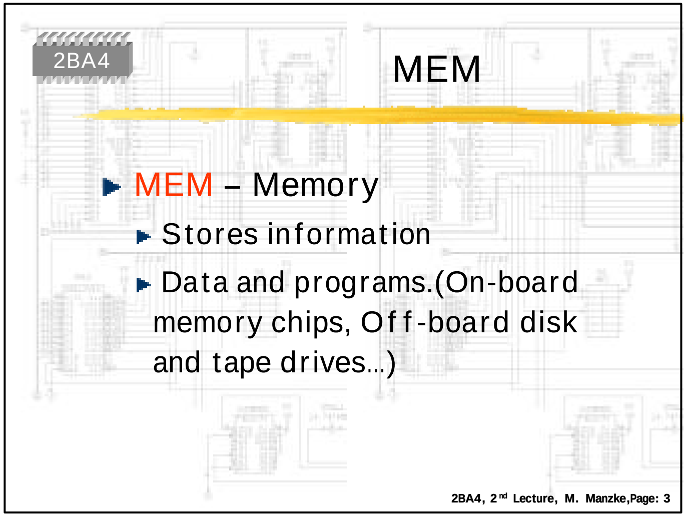
µProcessor System

- The Four Main Components:
  - CPU – Central Processing Unit
  - MEM – Memory
  - IO – Input and Output system
  - Operating Software
    - MON – Monitor
    - OS – Operating System



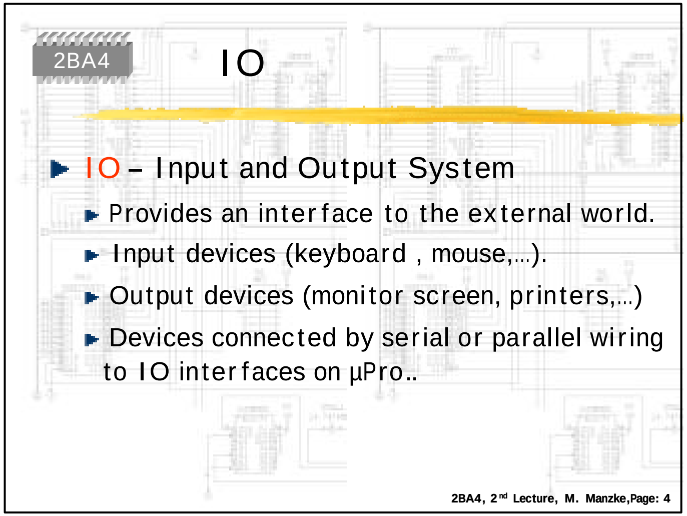
CPU

- CPU – Central Processing Unit
  - Executes instructions
  - Works upon data
  - Usually responsible for controlling the entire system.



MEM

- MEM – Memory
  - Stores information
  - Data and programs.(On-board memory chips, Off-board disk and tape drives...)



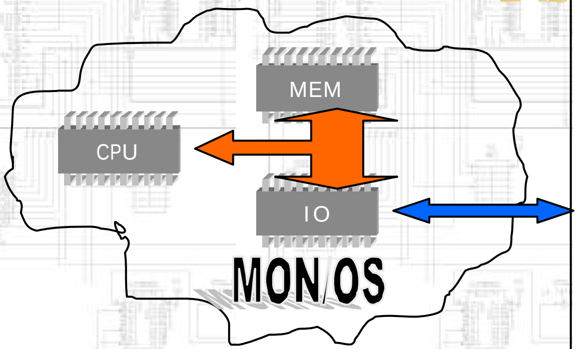
IO

- IO – Input and Output System
  - Provides an interface to the external world.
  - Input devices (keyboard , mouse,...).
  - Output devices (monitor screen, printers,...)
  - Devices connected by serial or parallel wiring to IO interfaces on µPro..

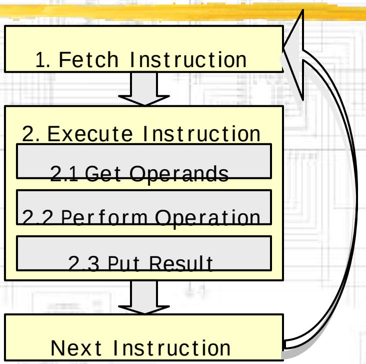
OS

- OS – Operating Software
  - Makes the system useable.
  - Particularly at startup.
  - Provides instructions that are executed by the processor.
  - Manages user interface.
  - Starting and stopping application programs.
  - Maintains system robustness and security.

Microprocessor System Architecture



CPU Operation (one)



1. Fetch Instruction

2. Execute Instruction

2.1 Get Operands

2.2 Perform Operation

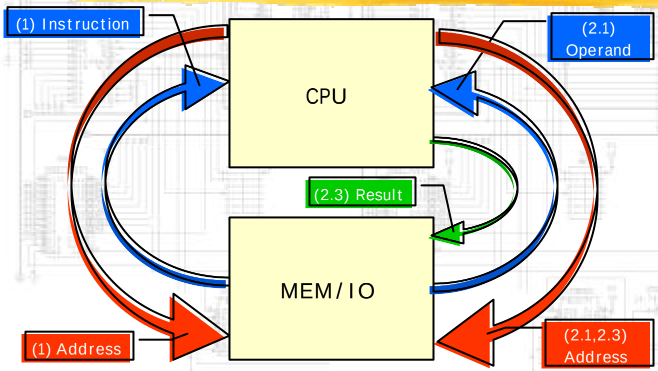
2.3 Put Result

Next Instruction

CPU Operation (two)

- Fetch Instruction
  - Read from on-board memory chip
- Execute Instruction
  - Get Operands
    - Fetched from internal location in CPU, on-board memory (MEM) or an IO interface device (IO).
  - Perform Operation
    - Internal processing by CPU.
  - Put Results
    - Results stored either internally in CPU, or out in on-board memory or an IO interface device.

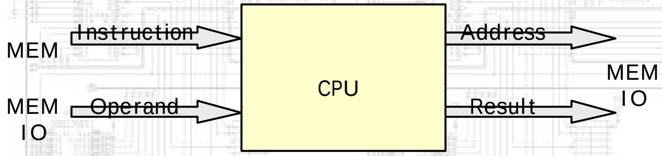
CPU Operation (three)



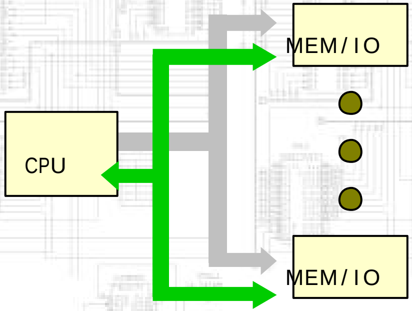
Communication Requirements

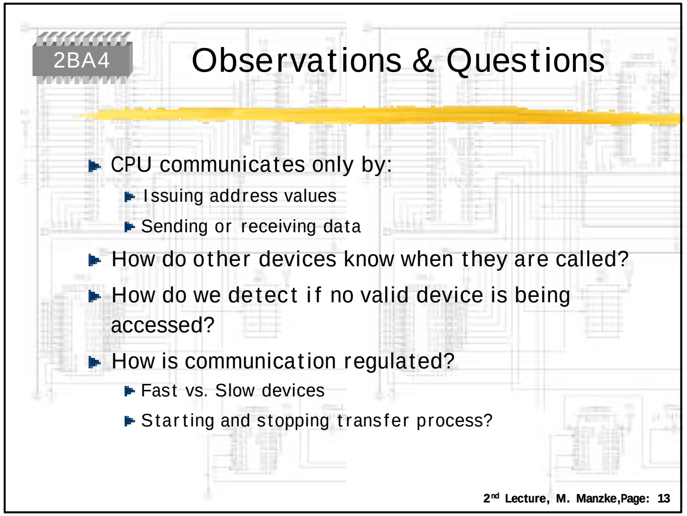
- Instructions (MEM -> CPU)
- Operands (MEM,IO -> CPU)
- Results (CPU -> MEM,IO)
- Usually 1 CPU, but many MEM and IO
- MEM, IO Address Information

CPU Information Flow



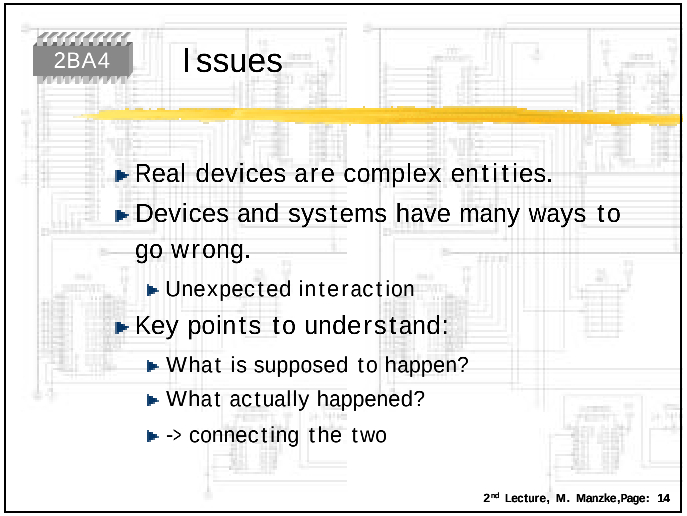
Many Memory and IO Interfaces





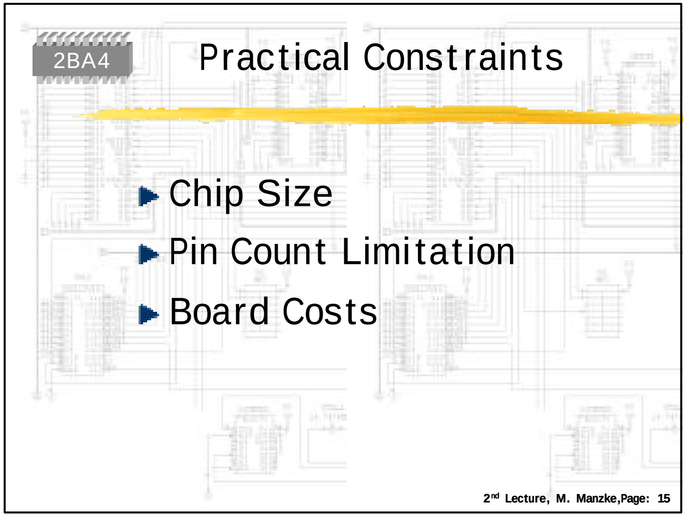
Observations & Questions

- CPU communicates only by:
  - Issuing address values
  - Sending or receiving data
- How do other devices know when they are called?
- How do we detect if no valid device is being accessed?
- How is communication regulated?
  - Fast vs. Slow devices
  - Starting and stopping transfer process?



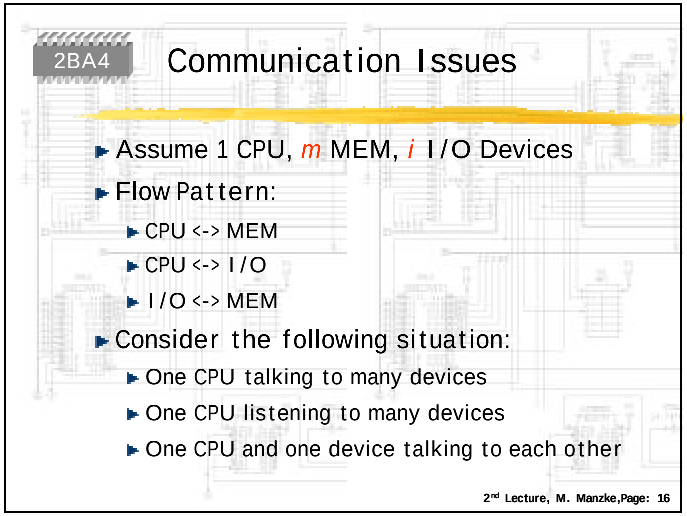
Issues

- Real devices are complex entities.
- Devices and systems have many ways to go wrong.
  - Unexpected interaction
- Key points to understand:
  - What is supposed to happen?
  - What actually happened?
  - -> connecting the two



Practical Constraints

- Chip Size
- Pin Count Limitation
- Board Costs



Communication Issues

- Assume 1 CPU, *m* MEM, *i* I/O Devices
- Flow Pattern:
  - CPU <-> MEM
  - CPU <-> I/O
  - I/O <-> MEM
- Consider the following situation:
  - One CPU talking to many devices
  - One CPU listening to many devices
  - One CPU and one device talking to each other

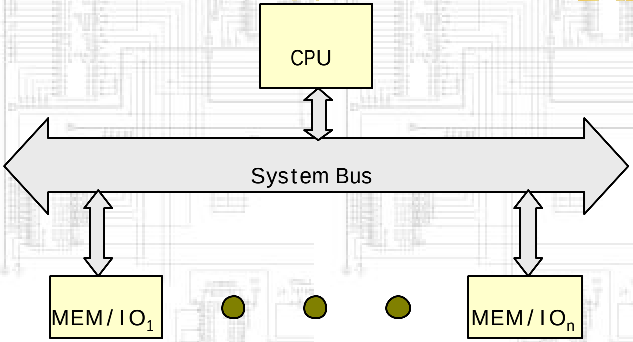
Solution 1 – System Bus

- Bus = Group of wires with common function
    - Address Bus (CPU -> MEM, IO)
    - Data Busses (CPU <-> MEM,IO)
  - Or
    - Address Bus (CPU -> MEM, IO)
    - Data Busses (CPU <-> MEM)
    - IO Busses (CPU <-> IO)

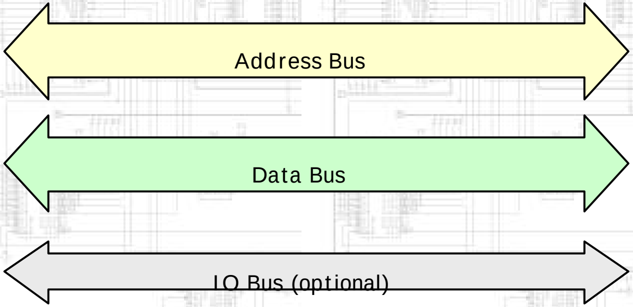
BUS

- Address Bus
  - CPU puts address on the bus, all other other devices read of that bus
- Data Busses
  - Data: Instructions and Operand information.
  - One bus takes data from CPU to MEM and IO.
  - Other bus brings data from IO and MEM to CPU.
  - Problem – many MEM and IO putting data onto one bus – conflict?

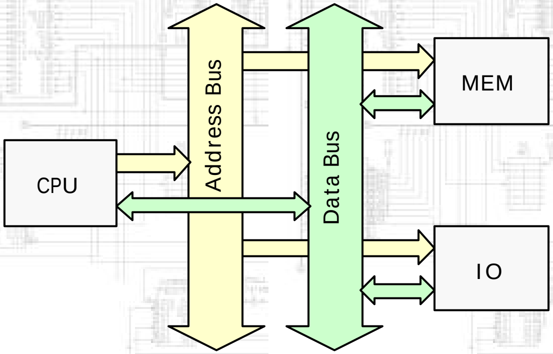
System Bus



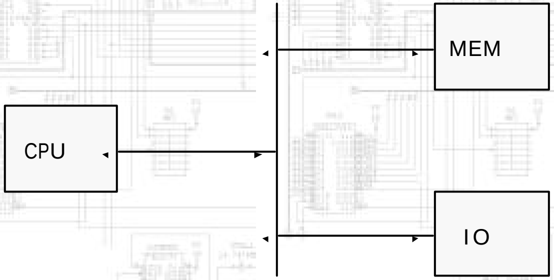
System Bus is close-up



Bus System



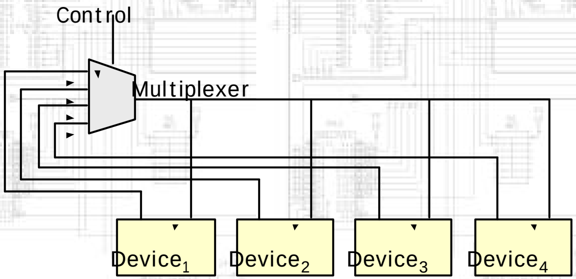
Zooming In
Single Bit Data Bus



Design Rules?

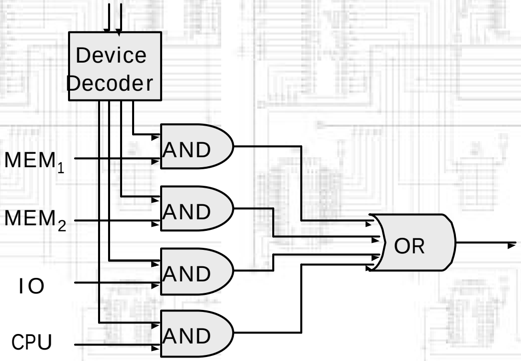
- Output Pins
  - Connect one to many inputs
  - Never Connect two outputs together
- How to do Bi-directional?

How do we implement a bus?



2BA4

Multiplexer Implementation


Device Decoder
MEM$_1$
MEM$_2$
IO
CPU
AND
AND
AND
AND
OR