

William Stallings Computer Organization and Architecture

Chapters 1 & 3 System Architecture

-
- Le persone che scoprono la potenza e la bellezza di idee di alto livello di astrazione spesso commettono l'errore di credere che le idee concrete a livelli inferiori di astrazione sono tutto sommato inutili e possono essere dimenticate.
 - Al contrario, i migliori informatici sono sempre saldamente radicati nei concetti basilari che governano il funzionamento dei calcolatori, ed in verità **l'essenza dell'informatica è l'abilità di comprendere e governare molti livelli di astrazione contemporaneamente.**
 - Donald Knuth, Keynote Address at the 8th Annual Conference on Innovation and Technology in Computer Science Education (ITiCSE-03)

From hardware to software

- Hardwired systems are inflexible
 - Changing function requires changing the wiring
 - e.g.: sum 2 numbers with k digits, sum k numbers
- But general purpose hardware can do different tasks, given correct control signals
- Supply a new set of control signals as needed under the control of a “program”

What is a program?

- A sequence of steps
- For each step, an arithmetic, logical, control or data movement operation is done
- For each operation, a different set of control signals is needed

Execution of the program

- For each operation a unique code is provided
 - e.g. ADD, MOVE
- A hardware circuit interprets the code and issues the control signals
- We have a computer!

one of the foremost mathematicians of the 20th century

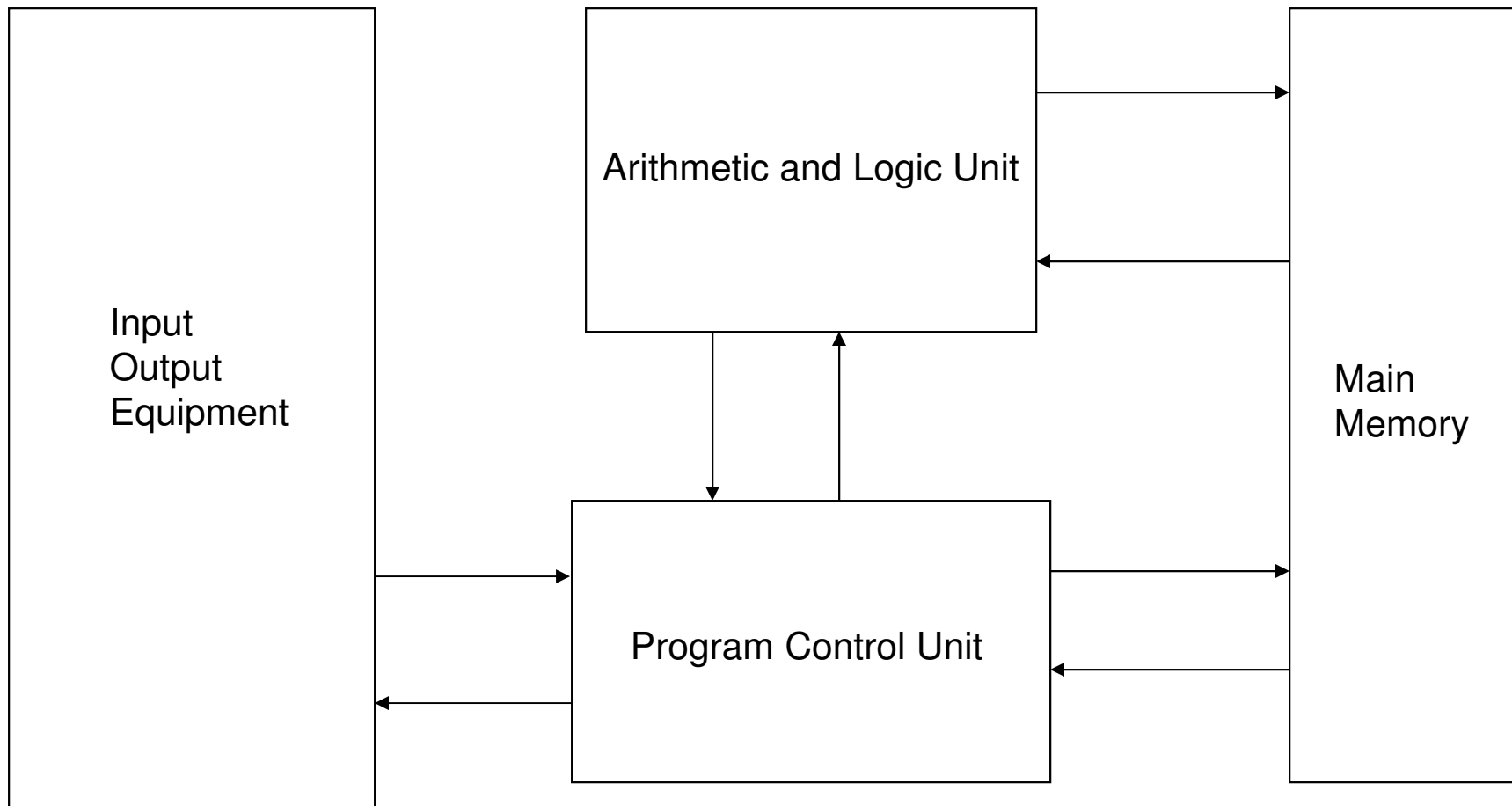


John von Neumann
(1903-1957)

*First Draft of a Report on the
Edvac (1945)*

*Edvac:
Electronic Discrete Variables
Automatic Computer
(1952)*

The von Neumann computer



von Neumann/Turing architecture

- Binary representation for data and program
- Main memory storing data AND PROGRAMS
- Control unit interpreting instructions from memory and executing
- Normal control flow is sequential
- Specialized device (ALU) to operate on data
- Memory accessed by means of address
- Input and output equipment operated by control unit

Architecture & Organization 1

- Architecture expresses those attributes visible to the programmer (i.e.: functions, commands)
 - Instruction set, number of bits used for data representation, I/O mechanisms, addressing techniques.
 - e.g. Is there a multiply instruction?
- Organization is how features are **implemented**
 - Control signals, interfaces, memory technology.
 - e.g. Is there a hardware multiply unit or is it done by repeated addition?
- Architecture = Specification
- Organization = Implementation

Architecture & Organization 2

- All Intel x86 family share the same basic architecture
- The IBM System/370 family share the same basic architecture
- This gives code compatibility
 - At least backwards
- Organization differs between different versions

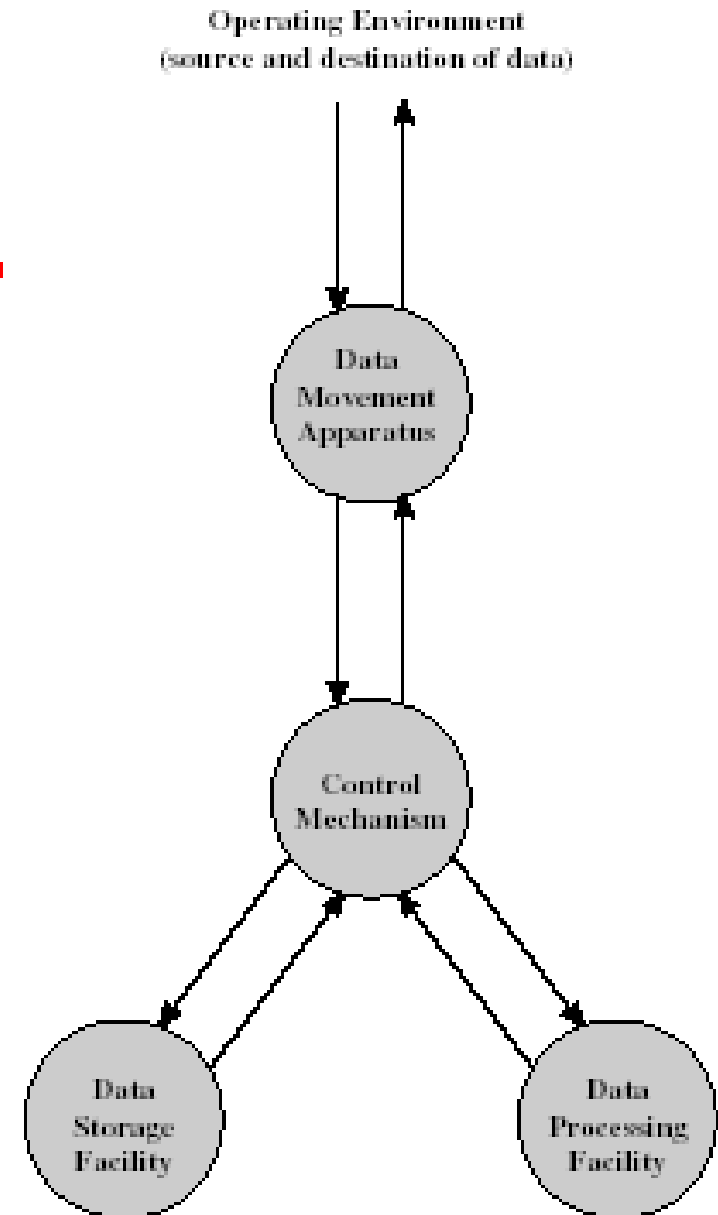
Structure & Function

- Structure is the way in which components relate to each other
- Function is the operation of individual components as part of the structure

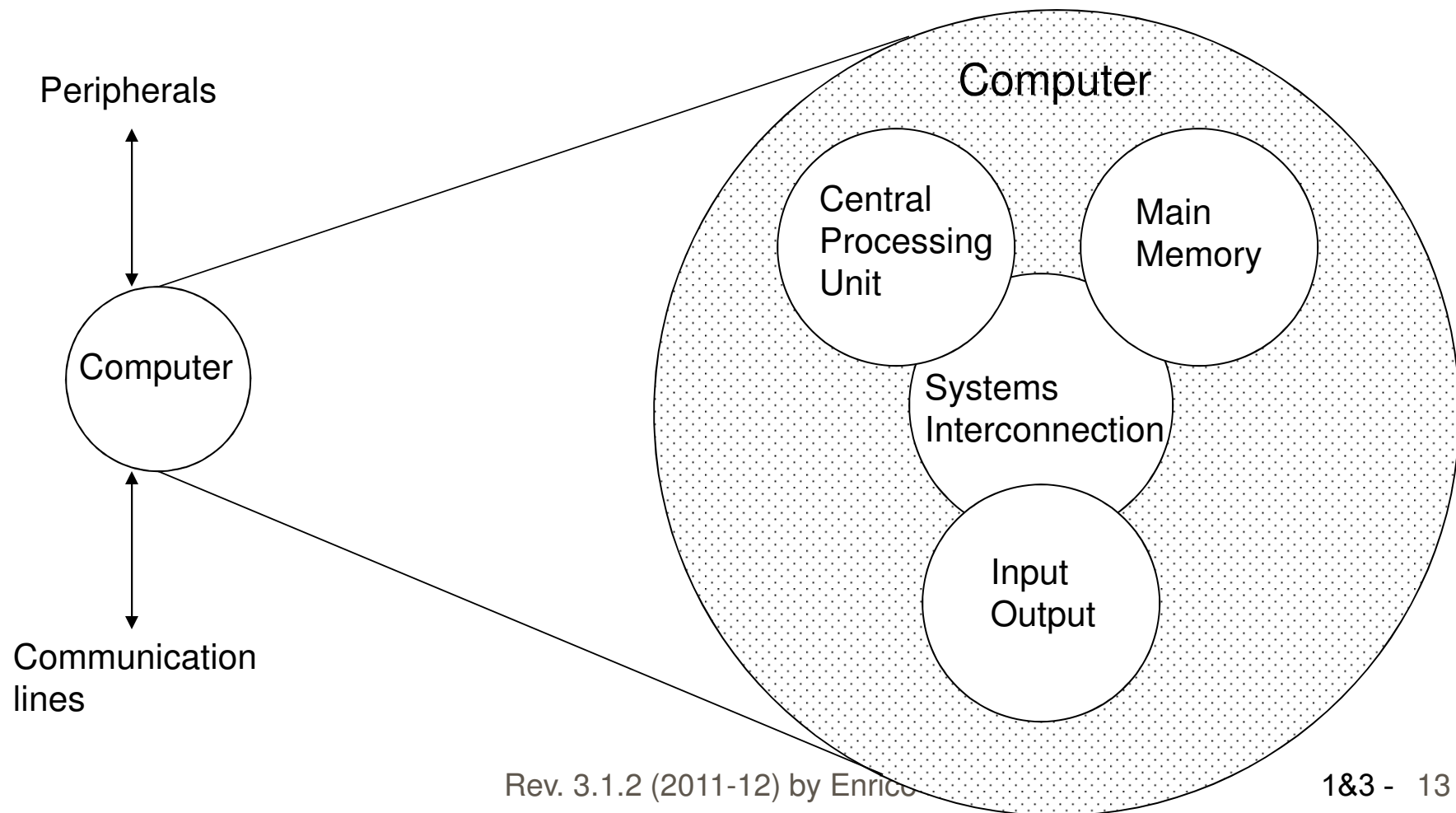
- Structure = static relations among components
- Function = dynamic behaviour of each component

Function

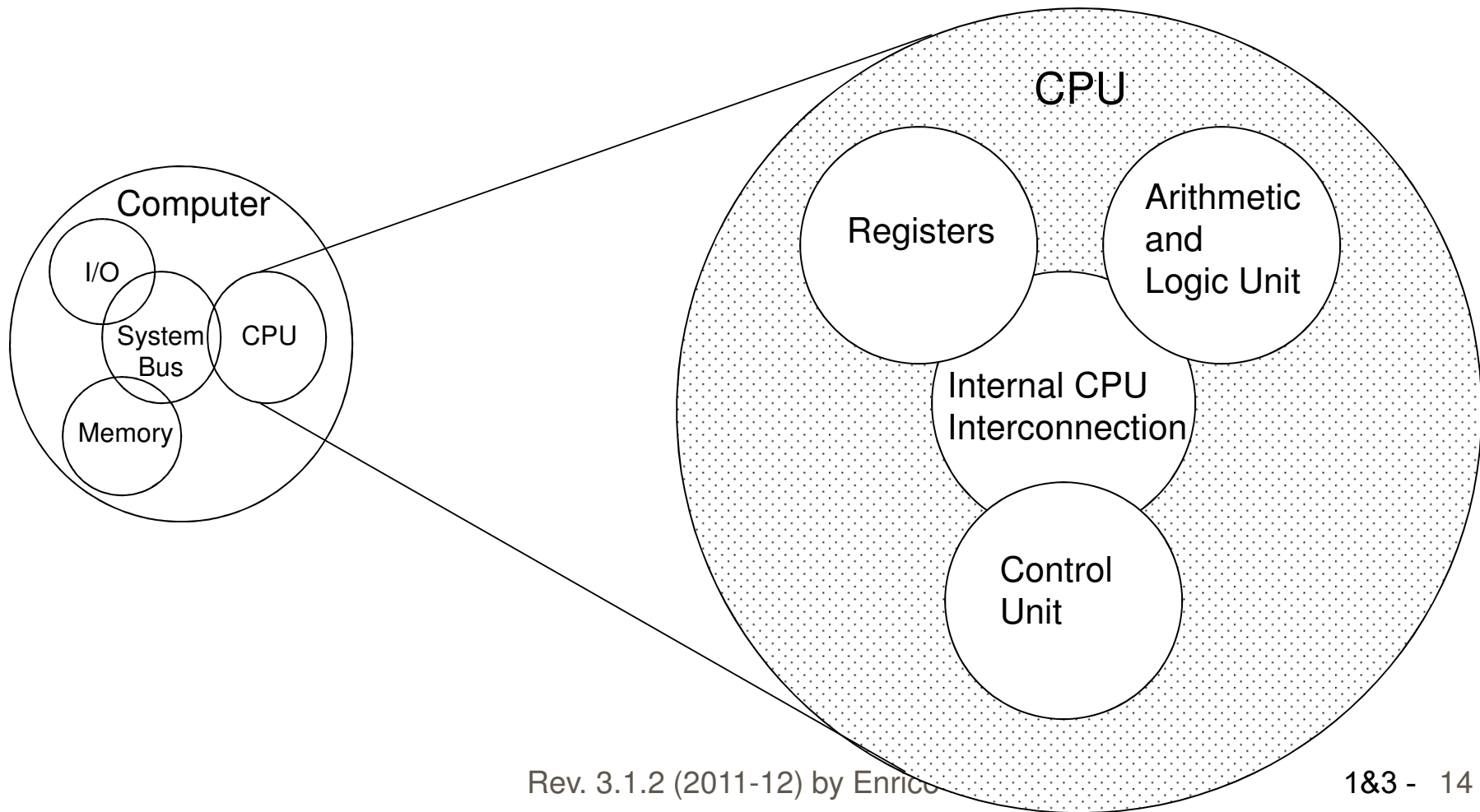
- All computer functions are:
 - Data processing
 - Data storage
 - Data movement
 - Control



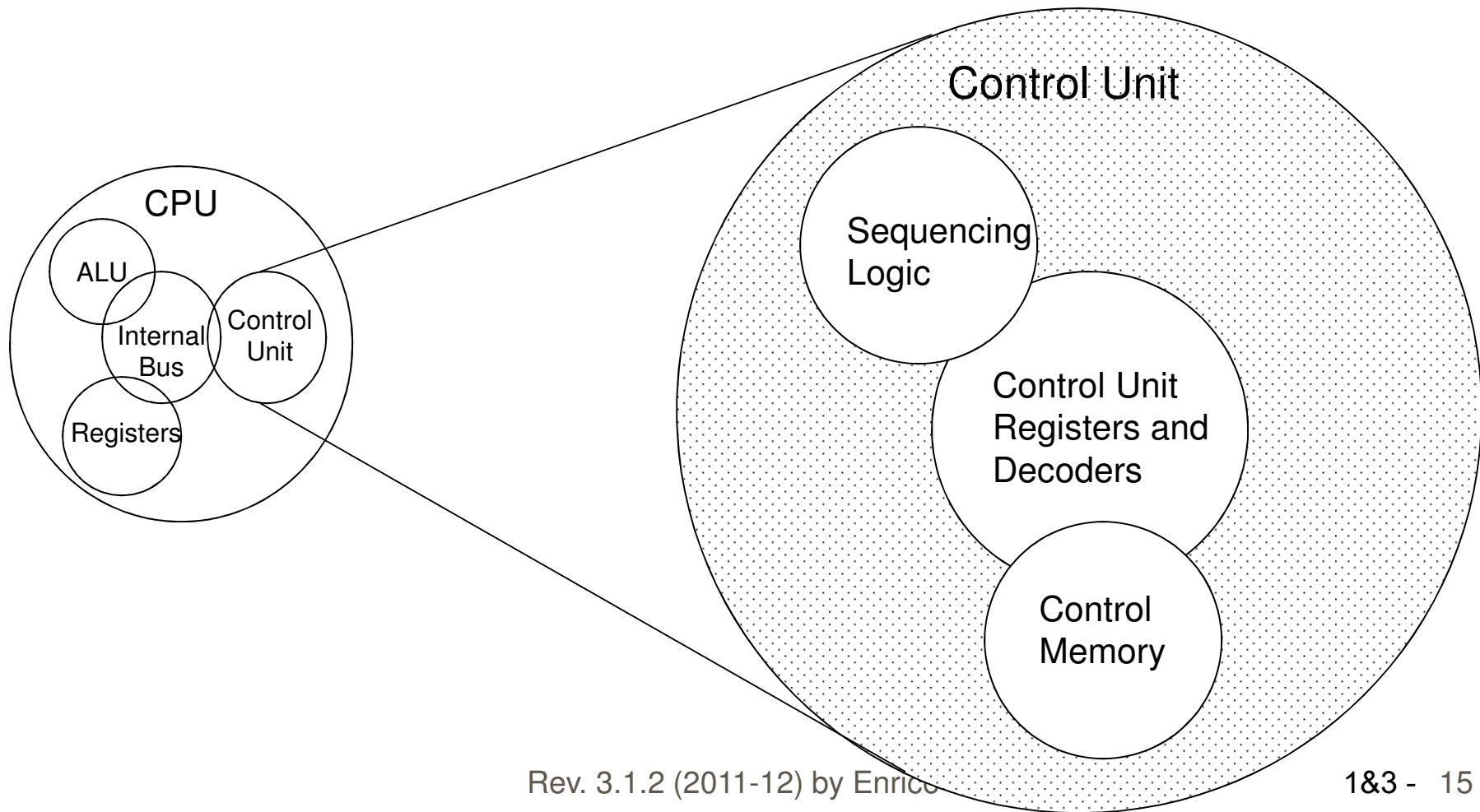
Structure - Top Level



Structure - The CPU

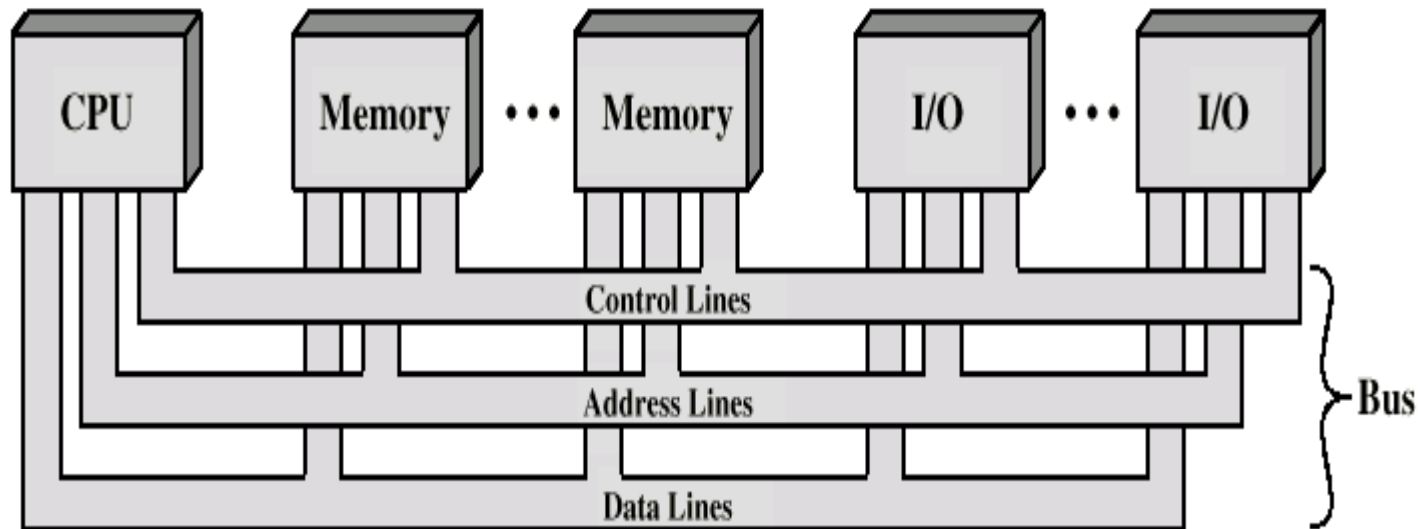


Structure - The Control Unit



Computer Components: Top Level View

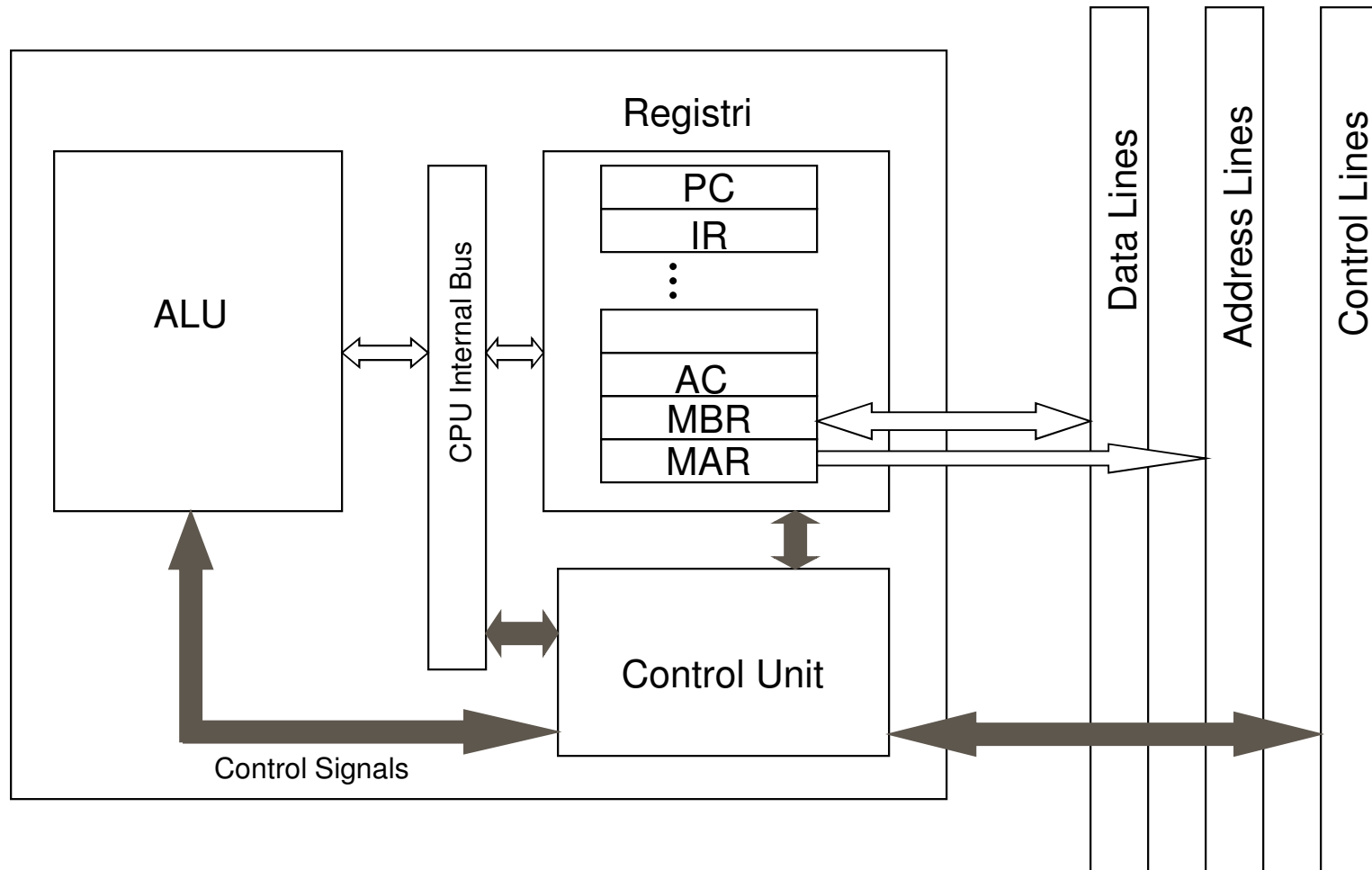
- Central Processing Unit (CPU)
- Memory
- I/O
- System Interconnections (usually a "bus")



Components

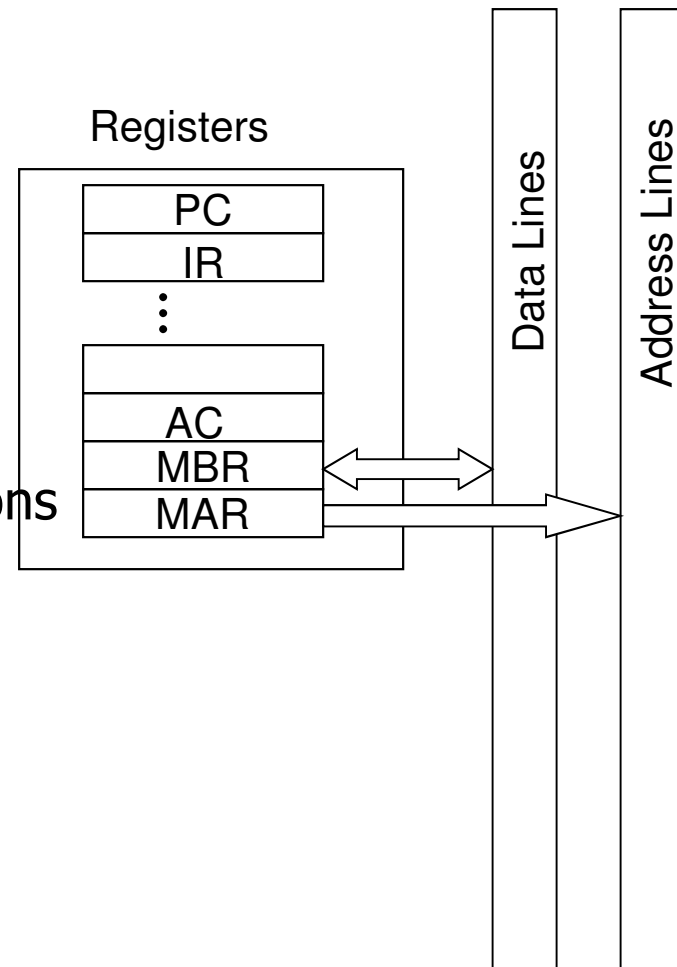
- Central Processing Unit (CPU):
 - Control Unit
 - Arithmetic and Logic Unit
 - Internal Registers
- Input/output
 - Data and instructions need to get into the system and results out
- Main memory
 - Temporary storage of code and results is needed

CPU Components: Top Level View



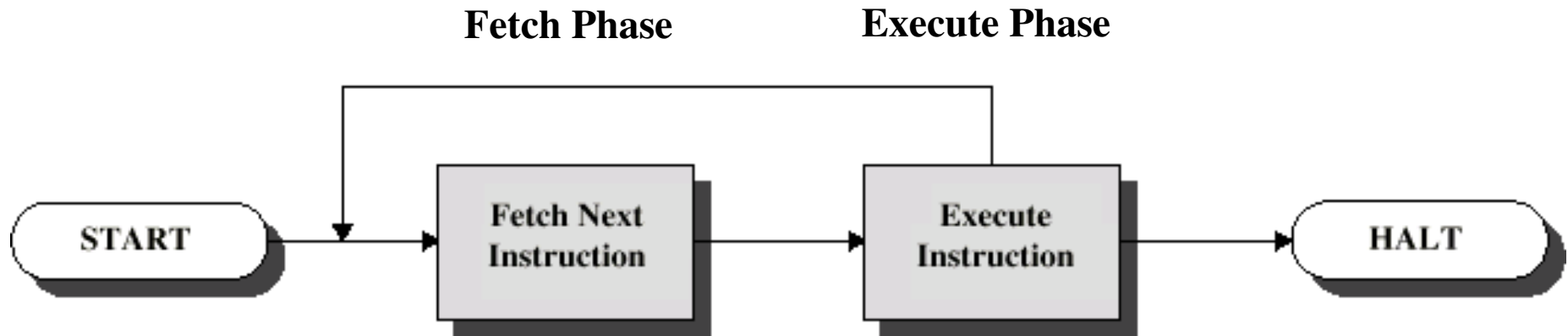
CPU: Function of registers

- Program Counter
 - Address of the next instruction
- Instruction Register
 - Code of instruction to execute
- Accumulator
 - Temporary storage for ALU operations
- Memory Address Register
 - Memory address where to R/W
- Memory Buffer Register
 - Data read/written from/to memory



Instruction Cycle

- Two phases:
 - Fetch
 - Execute



Fetch Phase

- Program Counter (PC) holds address of next instruction to fetch
- Processor fetches instruction from memory location pointed to by PC
- Instruction loaded into Instruction Register (IR)
- Increment PC (but PC may be changed later...)

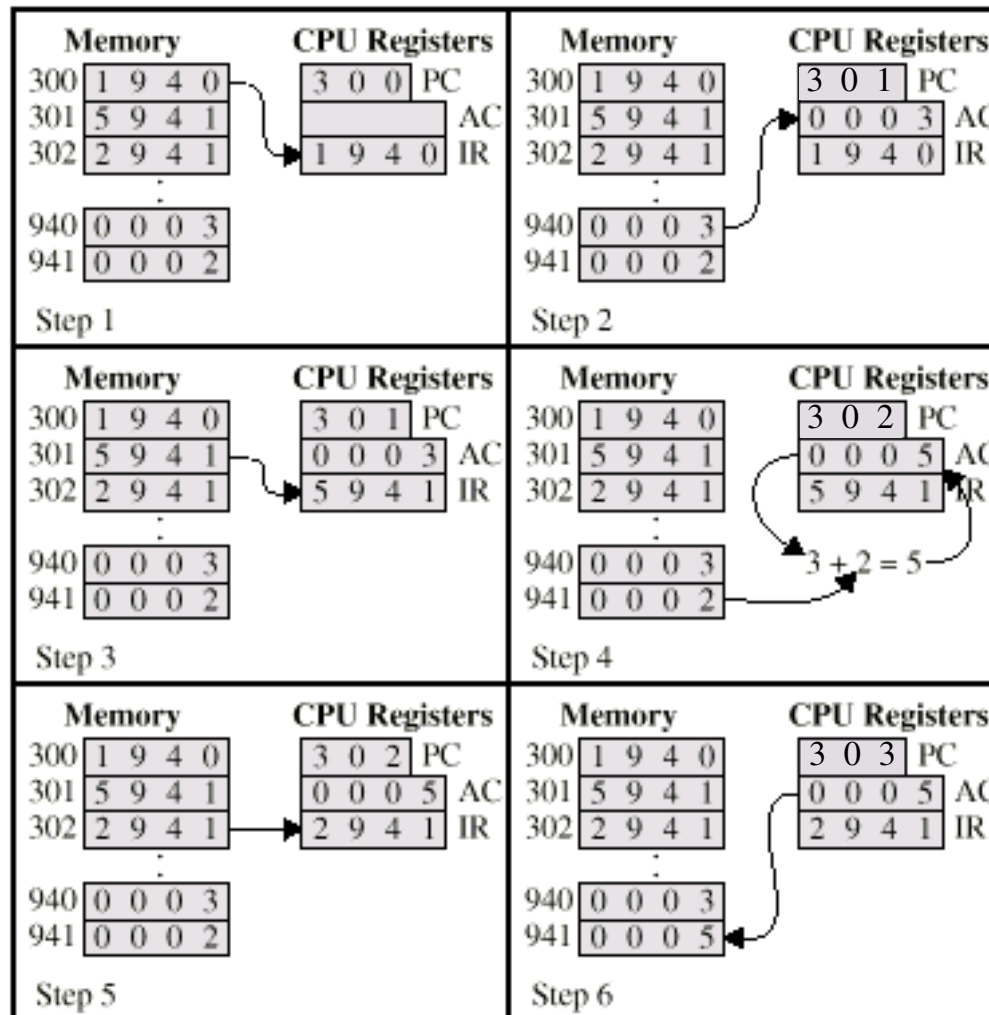
Execute Phase

- Processor decodes instruction and set-up circuits to perform required actions
- Actual execution of operation:
 - Processor-memory
 - data transfer between CPU and main memory
 - Processor-I/O
 - Data transfer between CPU and I/O module
 - Data processing
 - Some arithmetic or logical operation on data
 - Control
 - Alteration of sequence of operations
 - e.g. jump
 - Combination of above

A very simple processor

- INSTRUCTION: OP_CODE + ADDRESS
- OP_CODES:
 - 1 (address) -> Accumulator
 - 2 Accumulator -> address
 - 5 Accumulator+(address) -> Accumulator

Example of Program Execution



Connecting

- All the units must be connected
- Different type of connection for different type of unit
 - Memory
 - Input/Output
 - CPU

Memory Connection

- Receives and sends data
- Receives addresses (of locations)
- Receives control signals
 - Read
 - Write
 - Timing

Input/Output Connection(1)

- Similar to memory from computer's viewpoint
- Data (during output operations)
 - Receive data from computer
 - Send data to peripheral
- Data (during input operations)
 - Receive data from peripheral
 - Send data to computer

Input/Output Connection(2)

- Receive control signals from computer
- Send control signals to peripherals
 - e.g. spin disk
- Receive addresses from computer
 - e.g. port number to identify peripheral
- Send interrupt signals (control)

CPU Connection

- Reads instruction and data
- Writes out data (after processing)
- Sends control signals to other units
- Receives (& acts on) interrupts

Buses

- There are a number of possible interconnection systems
- Single and multiple BUS structures are most common
- e.g. Control/Address/Data bus (PC)
- e.g. Unibus (DEC-PDP)

What is a Bus?

- A communication pathway connecting two or more devices
- Usually broadcast
- Often grouped
 - A number of channels in one bus
 - e.g. 32 bit data bus is 32 separate single bit channels
- Power lines may not be shown

Data Bus

- Carries data
 - Remember that there is no difference between “data” and “instruction” at this level
- Width is a key determinant of performance
 - 8, 16, 32, 64 bit

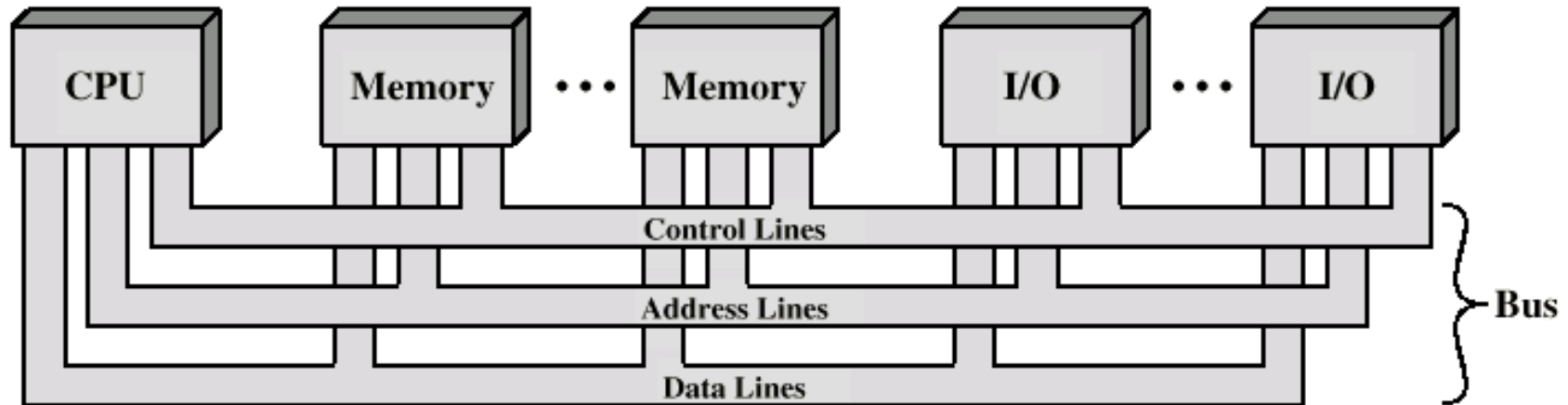
Address bus

- Identify the source or destination of data
- e.g. CPU needs to read an instruction (data) from a given location in memory
- Bus width determines maximum memory capacity of system
 - e.g. 8080 has 16 bit address bus giving 64k address space

Control Bus

- Control and timing information
 - Memory read/write signal
 - Interrupt request
 - Clock signals

Bus Interconnection Scheme



- Every device is attached to the bus:
 - its use needs to be coordinated

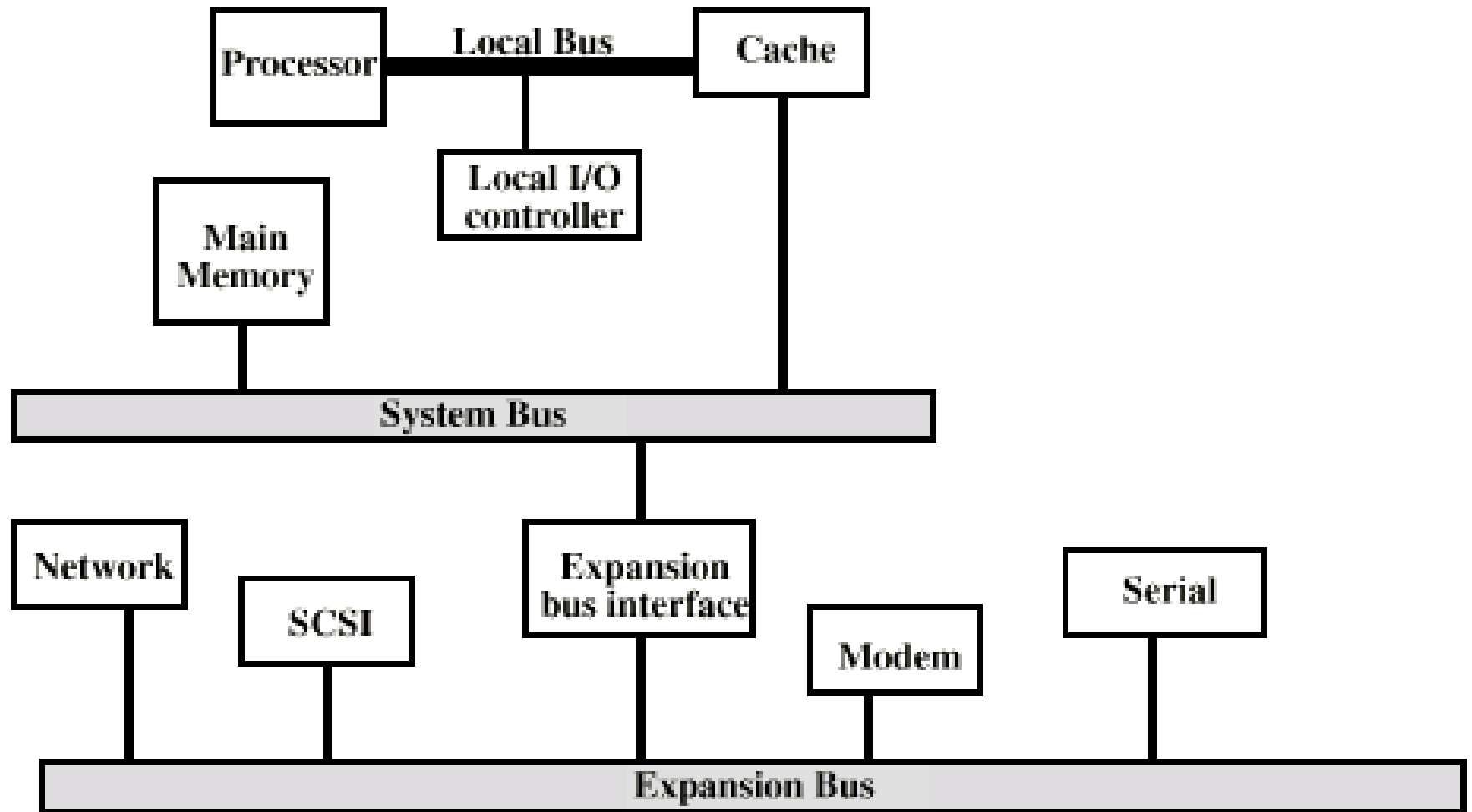
Big and Yellow?

- What do buses look like?
 - Parallel lines on circuit boards
 - Ribbon cables
 - Strip connectors on mother boards
 - e.g. PCI
 - Sets of wires

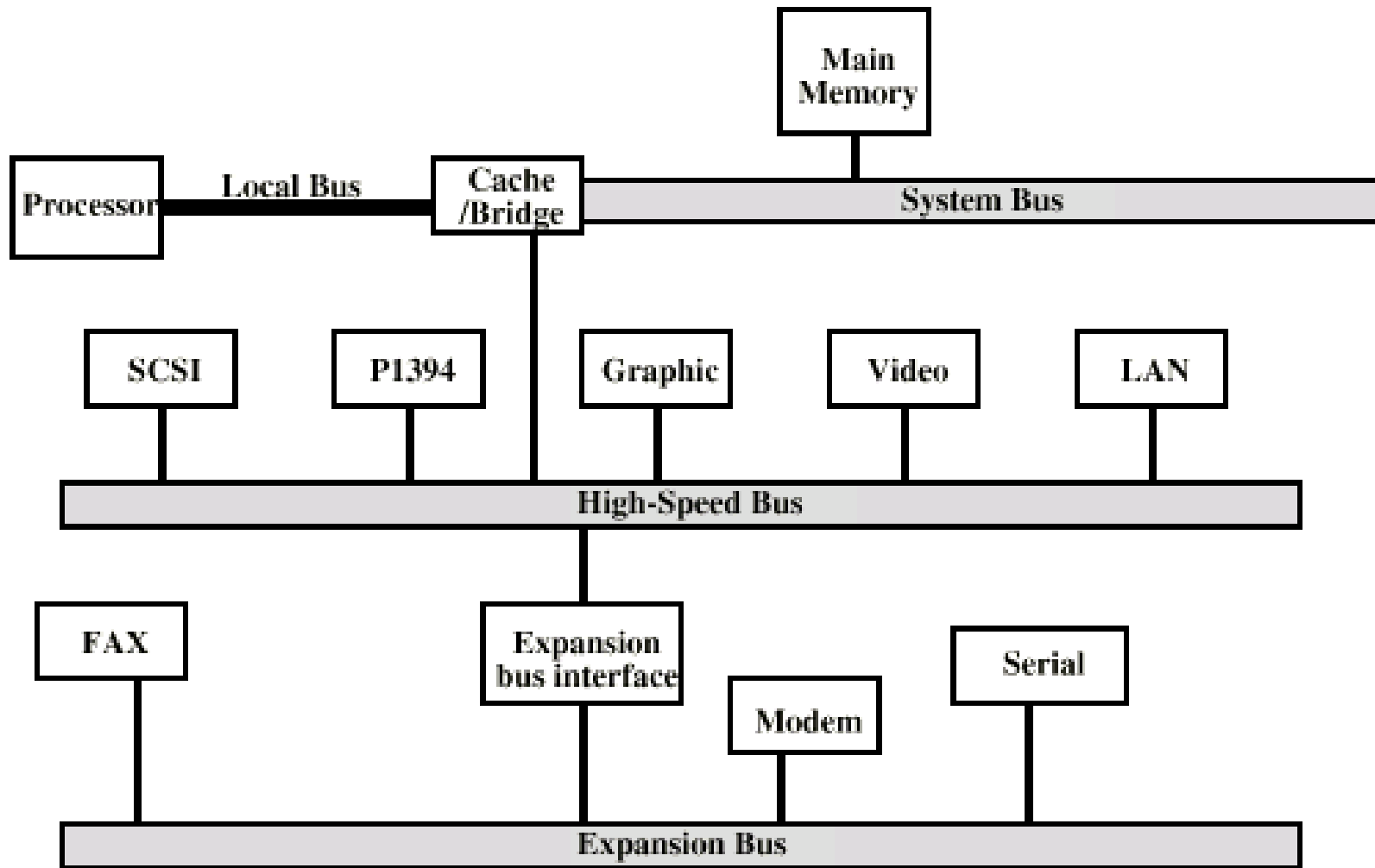
Single Bus Problems

- Lots of devices on one bus leads to:
 - Propagation delays
 - Long data paths mean that co-ordination of bus use can adversely affect performance
 - If aggregate data transfer approaches bus capacity
- Most systems use multiple buses to overcome these problems

Traditional (ISA) (with cache)



High Performance Bus



Bus Types

- Dedicated
 - Separate data & address lines
- Multiplexed
 - Shared lines
 - Address valid or data valid control line
 - Advantage - fewer lines
 - Disadvantages
 - More complex control
 - Ultimate performance