B38DF
Computer Architecture and Embedded Systems

Alexander Belyaev

Heriot-Watt University
School of Engineering & Physical Sciences
Electrical, Electronic and Computer Engineering

E-mail: a.belyaev@hw.ac.uk
Office: EM2.29

Based on the slides prepared by Dr. Mustafa Suphi Erden

Aim of Course

- **To understand how computers (PCs, laptops, microprocessors, embedded system computing) work.**
  - Computer architecture,
  - Processors,
  - Memory,
  - Instructions, etc.
  - Programming of a simple processor with several instructions

Course Assessment

- **Coursework (100%)**:
  - 50% - 2 Take Home Assignments
  - 50% - 3 Lab Assessments

Tentative Program of the 1st Part (weeks 1-6)

- Basic components of computer systems: Hardware, Software, CPU, Memory, Input-output devices
- Basic concepts of computer systems: Program, Language, Architecture, Interpreter, Instruction
- Processors – CPU organization, Instruction execution
- Parallelism, Latency, Throughput
- Performance Analysis, Amdahl's Law
- Primary/Internal Memory – Memory addresses, Big endian and little endian memory
- Primary/Internal Memory – Parity bits, Cache memory
- Secondary/External Memory – Memory hierarchy, Magnetic disks, CD-ROMs, DVD, Flash memory, EEPROM, Ferroelectric RAM
- Programmable Logic Devices (PLDs), Memory addressing
- Instructions – Instruction Formats, Addressing
- Three-instruction programmable processor
- Input/Output Device Connection – Buses, Memory mapping

Main Textbooks

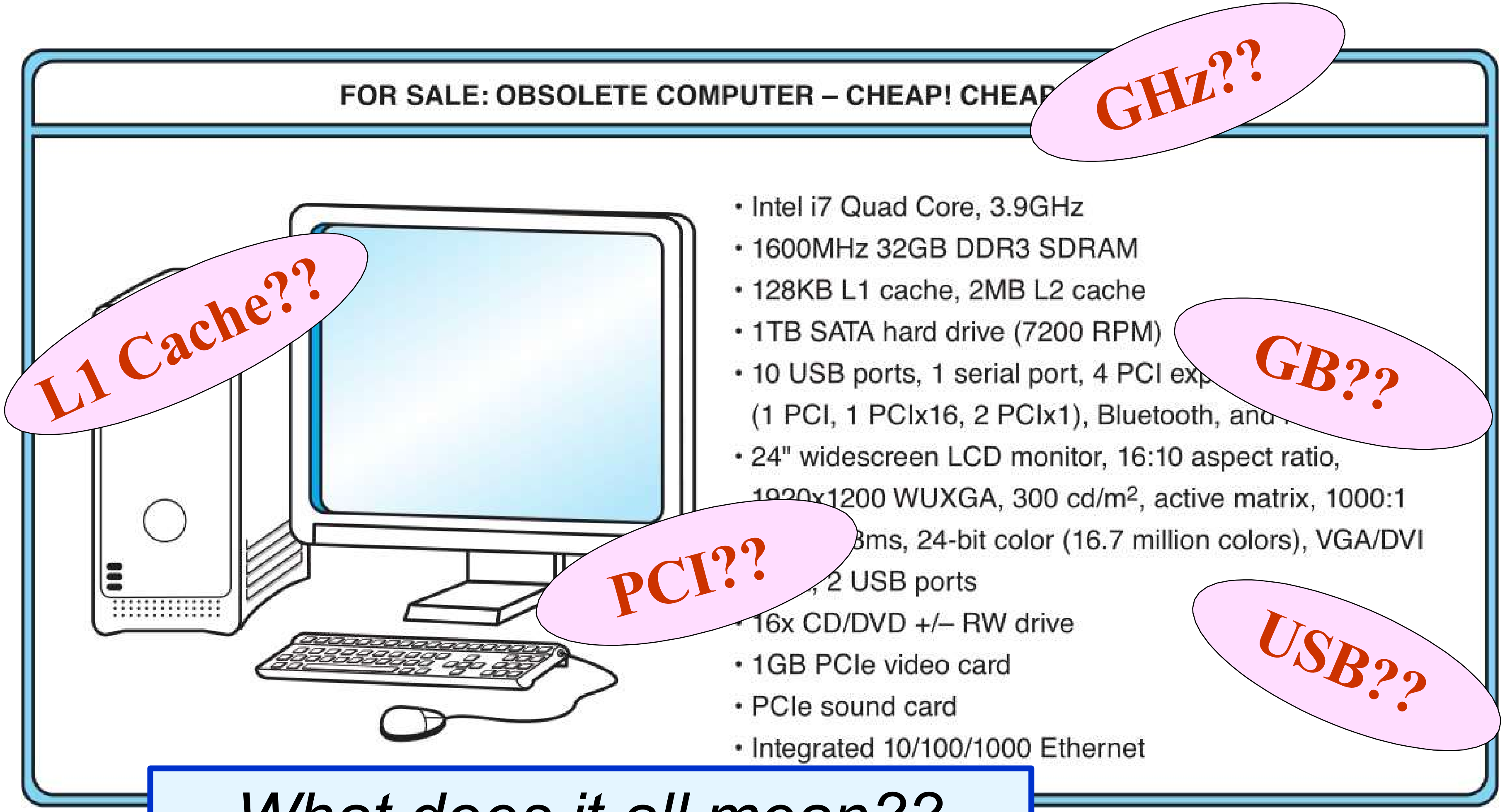
- Andrew S. Tanenbaum, *Structured Computer Organization*, 5th edition, Prentice Hall, 2006.
- Frank Vahid, *Digital Design*, 2nd edition, Wiley, 2011 (Chapter 8).

Other useful textbooks

- William Stallings, *Computer Organization & Architecture*, 6th edition, Prentice Hall, 2003.
- John L. Hennessy and David A. Patterson, *Computer Architecture*, 3rd edition, Morgan Kaufmann, 2003.
- Linda Null and Julia Lobur, *The Essentials of Computer Organization and Architecture*, Jones and Bartlett Publishers, 2003.

What we deal with in this course: An Example System

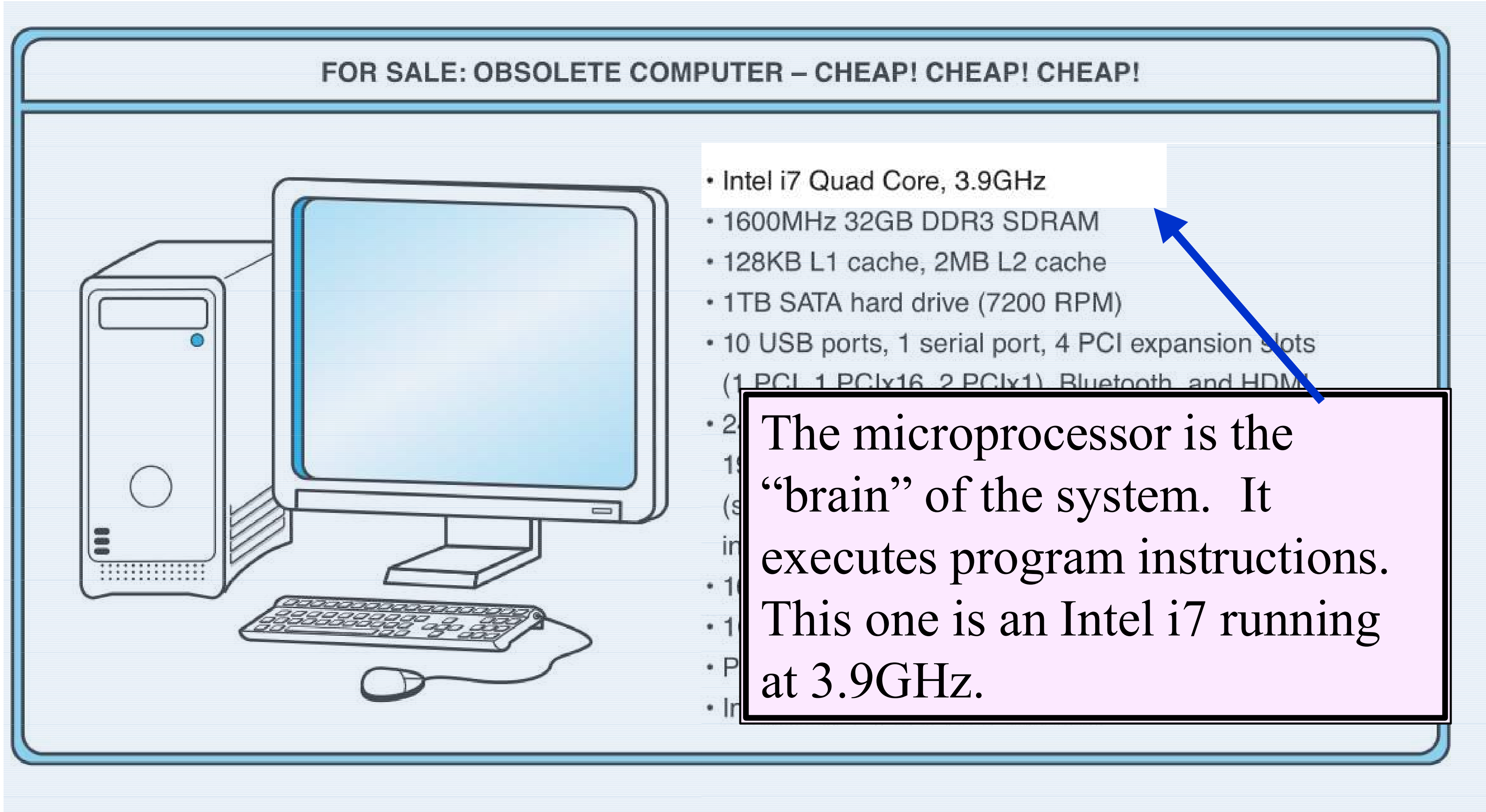
Consider this advertisement:



What does it all mean??

Linda Null and Julia Lobur, The Essentials of Computer Organization and Architecture, (c) Jones and Bartlett Publishers, 2003, 2015.

An Example System



The microprocessor is the "brain" of the system. It executes program instructions. This one is an Intel i7 running at 3.9GHz.

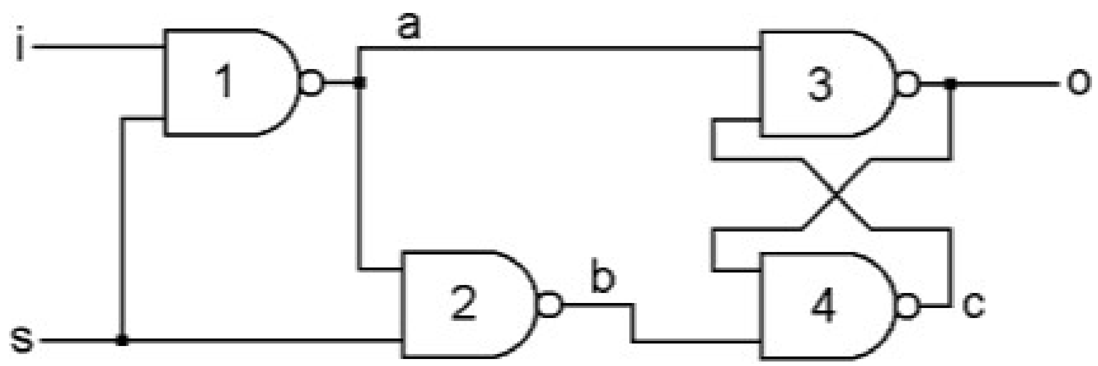
Linda Null and Julia Lobur, The Essentials of Computer Organization and Architecture, (c) Jones and Bartlett Publishers, 2003, 2015.

An Example System

- Computers with large main memory capacity can run larger programs with greater speed than computers having small memories.
- RAM is an acronym for *random access memory*. Random access means that memory contents can be accessed directly if you know its location.
- Cache is a type of temporary memory that can be accessed faster than RAM.

Linda Null and Julia Lobur, The Essentials of Computer Organization and Architecture, (c) Jones and Bartlett Publishers, 2003, 2015.

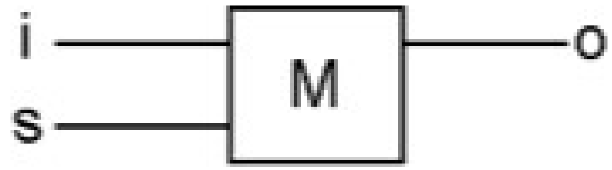
A computer must have memory. How RAM works



One bit of computer memory made of four NAND gates (using a latch)

‘**i**’ where we input the bit we want to remember and ‘**o**’ is the output of the remembered bit. ‘**s**’ is an input that tells these gates when to set the memory.

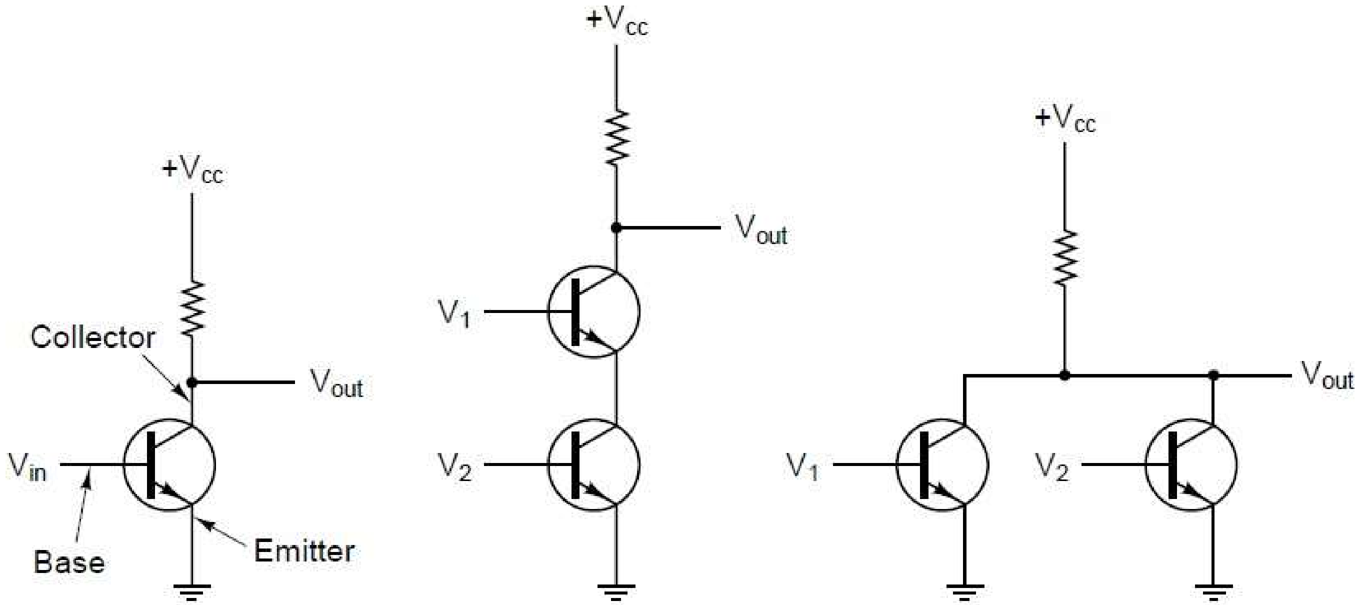
With ‘**s**’ on, ‘**o**’ ends up with the same as ‘**i**’.



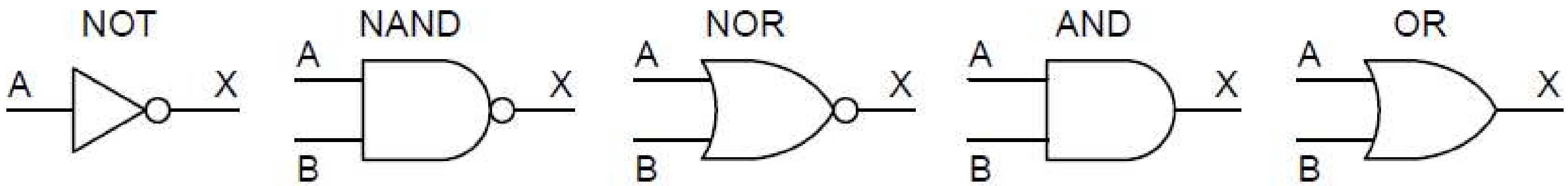
Let us now turn ‘**s**’ off. If ‘**i**’ and ‘**o**’ were on before ‘**s**’ got turned off, then ‘**o**’ stays on. If ‘**i**’ and ‘**o**’ were off before ‘**s**’ got turned off, then ‘**o**’ stays off.

J. Clark Scott, But How Do It Know? The Basic Principles of Computers for Everyone, (c) 2009 by J. Clark Scott.

Gates and Boolean Algebra



(a) (b) (c)



A	X
0	1
1	0

(a)

A	B	X
0	0	1
0	1	1
1	0	1
1	1	0

(b)

A	B	X
0	0	1
0	1	0
1	0	0
1	1	0

(c)

A	B	X
0	0	0
0	1	0
1	0	0
1	1	1

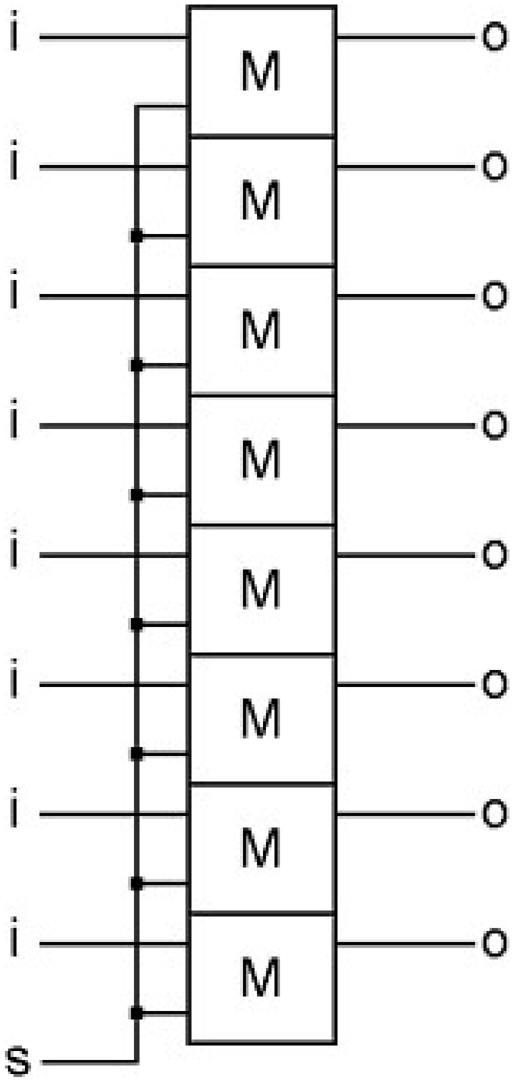
(d)

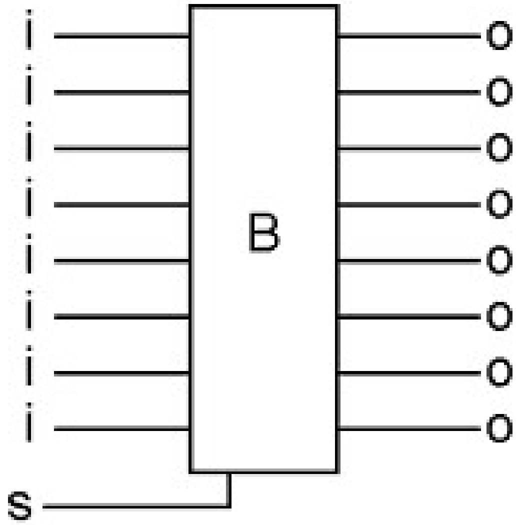
A	B	X
0	0	0
0	1	1
1	0	1
1	1	1

(e)

J. Clark Scott, But How Do It Know? The Basic Principles of Computers for Everyone, (c) 2009 by J. Clark Scott.

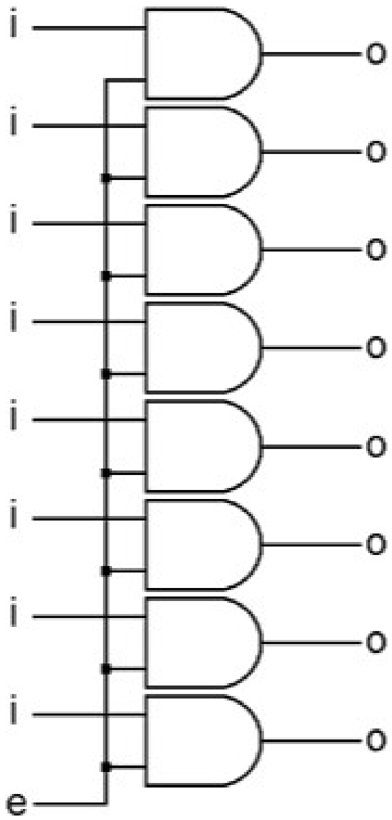
A memory byte

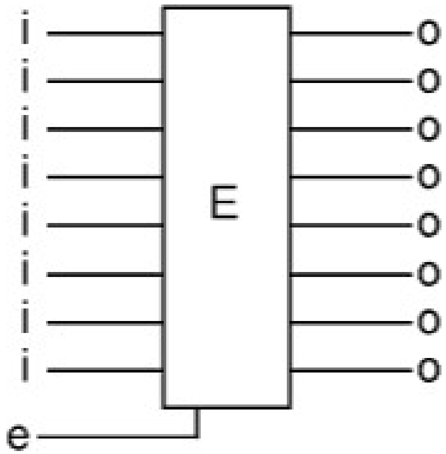




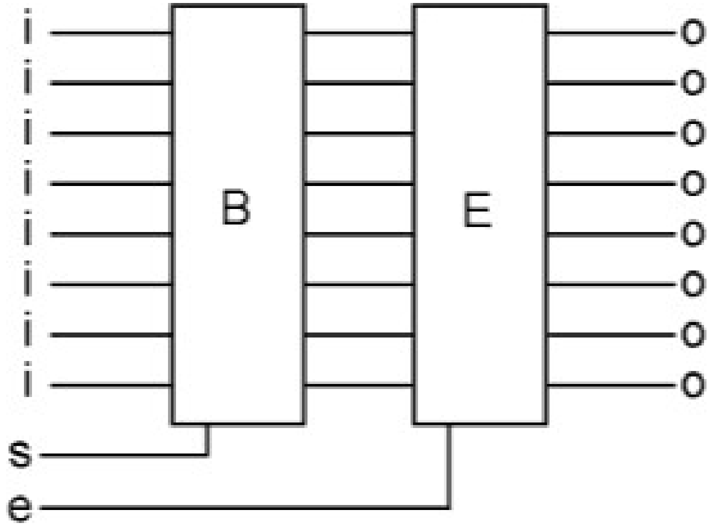
J. Clark Scott, But How Do It Know? The Basic Principles of Computers for Everyone, (c) 2009 by J. Clark Scott.

Register



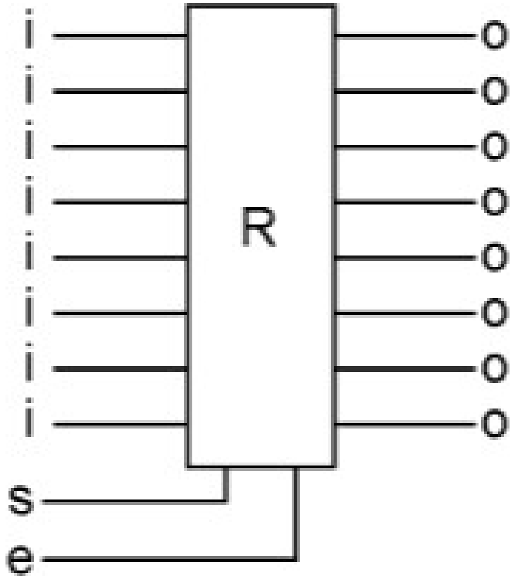


An Enabler allows a byte through when ‘**e**’ is 1 and stops the byte when ‘**e**’ is 0.



Register is a place to record some kind of information.

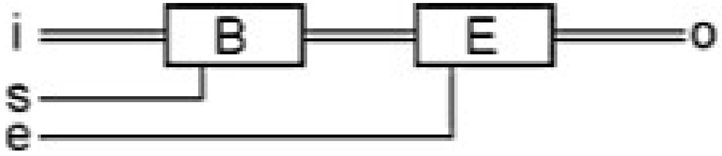
Similar to a hotel register

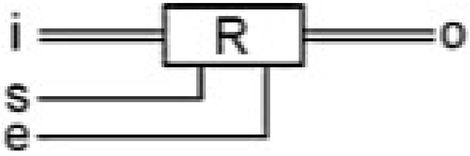


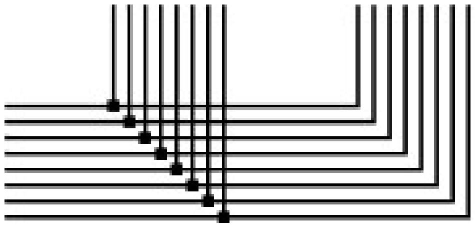
J. Clark Scott, But How Do It Know? The Basic Principles of Computers for Everyone, (c) 2009 by J. Clark Scott.

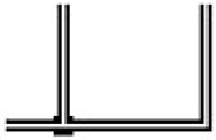
Bus

In the world of computers, a bus is simply a set of wires (usually **8**, **16** or **32**-bits wide) that goes to various places inside a computer.









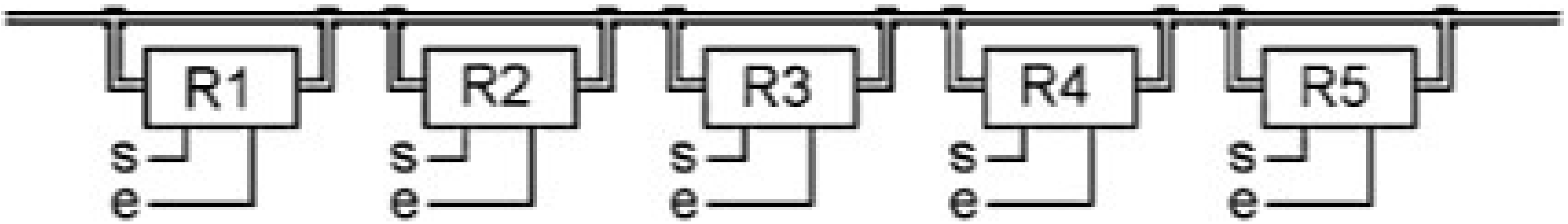
Why is it called a bus?

An old electrical term 'buss' means a bar of metal used as a very large wire in places like power generating plants.

But there is also an interesting similarity to the kind of bus that people use for transportation.

J. Clark Scott, But How Do It Know? The Basic Principles of Computers for Everyone, (c) 2009 by J. Clark Scott.

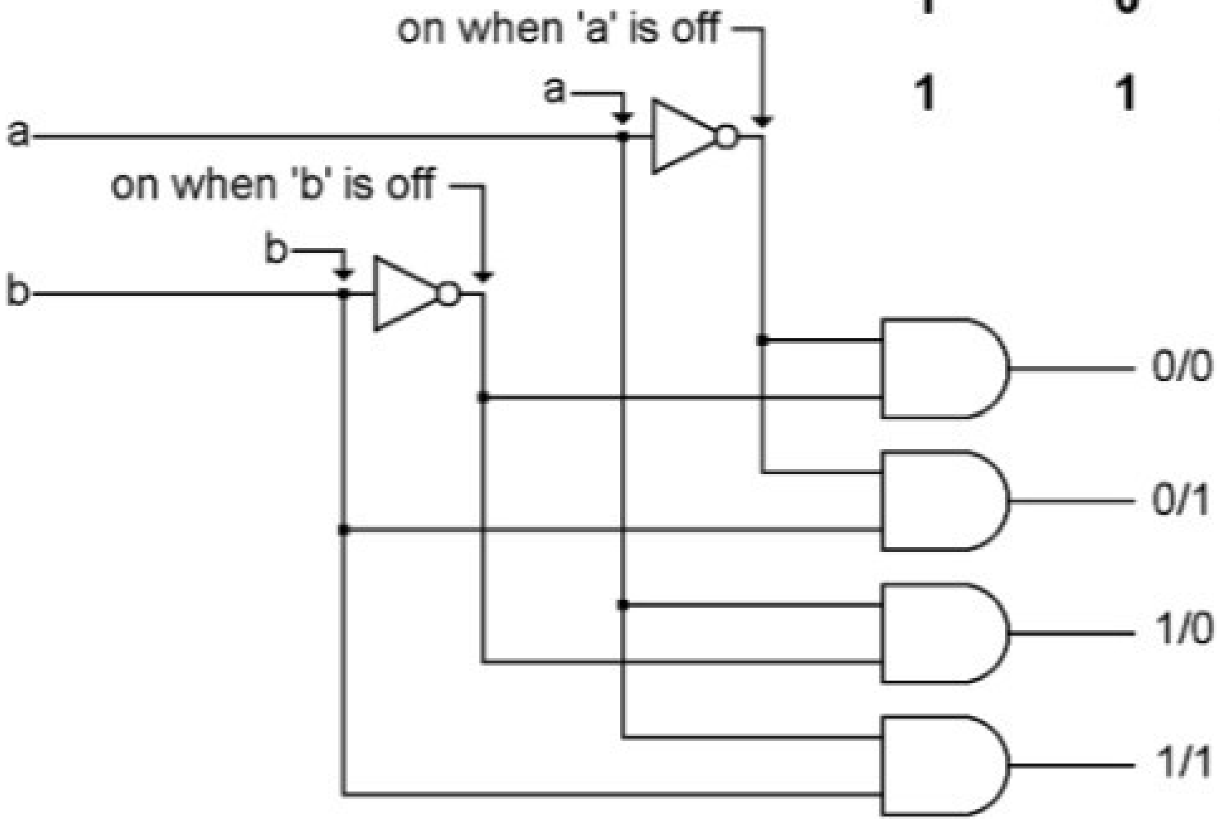
A bus and registers

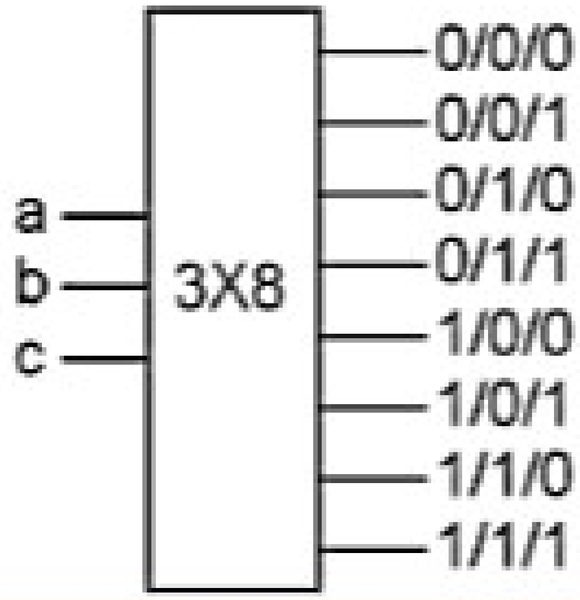


J. Clark Scott, But How Do It Know? The Basic Principles of Computers for Everyone, (c) 2009 by J. Clark Scott.

A binary decoder

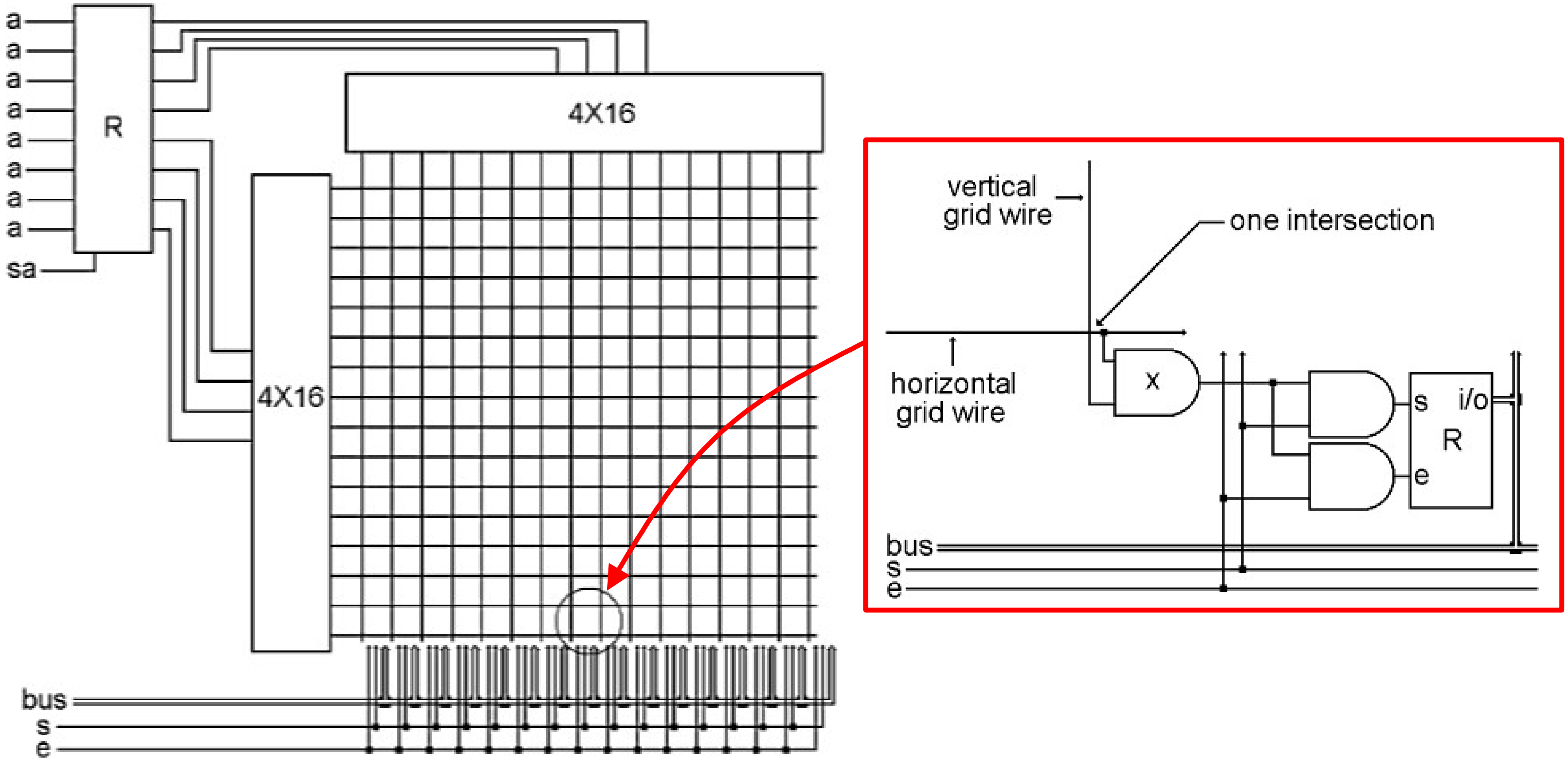
a	b	0/0	0/1	1/0	1/1
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1





J. Clark Scott, But How Do It Know? The Basic Principles of Computers for Everyone, (c) 2009 by J. Clark Scott.

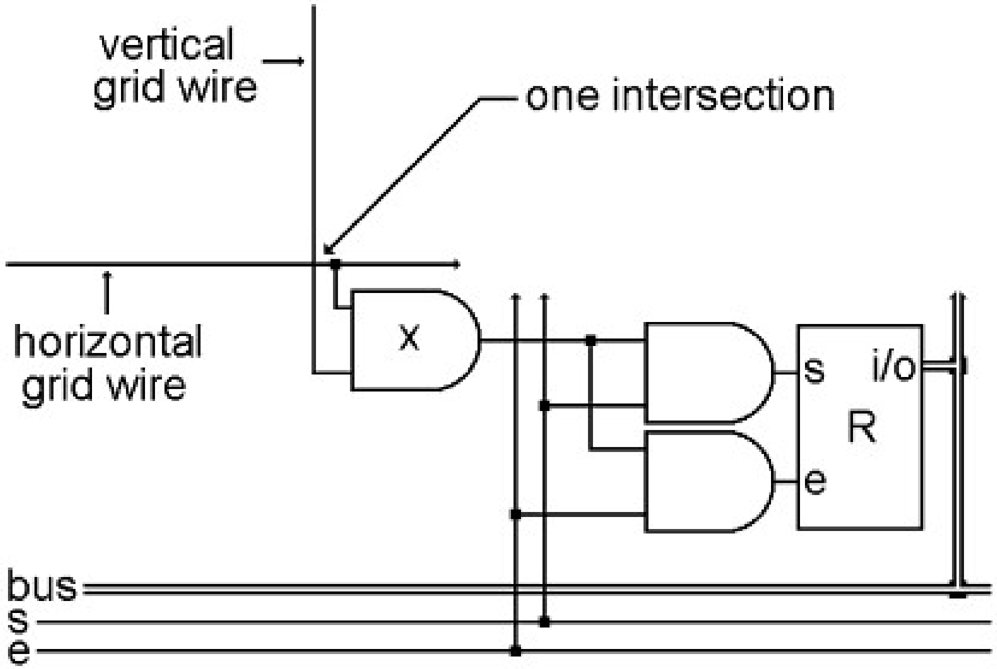
Random Access Memory (RAM)

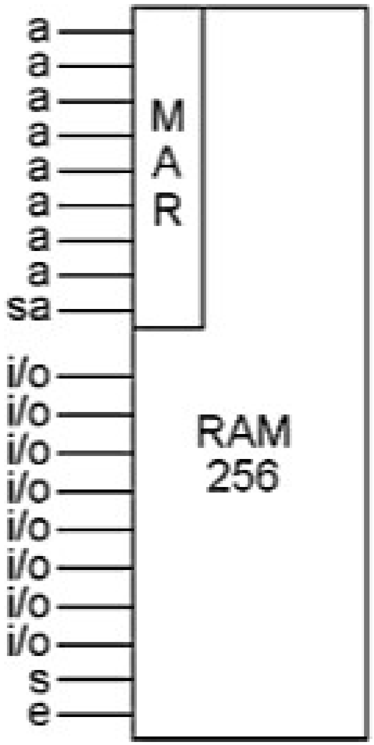


In this example we have 257 registers: 256 of them are memory storage locations and one register is used to select one of the storage locations and is called "Memory Address Register" or "MAR" for short.

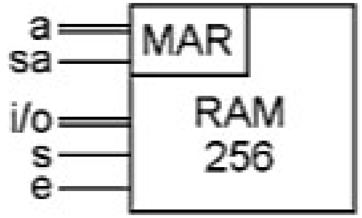
J. Clark Scott, But How Do It Know? The Basic Principles of Computers for Everyone, (c) 2009 by J. Clark Scott.

Random Access Memory (RAM)





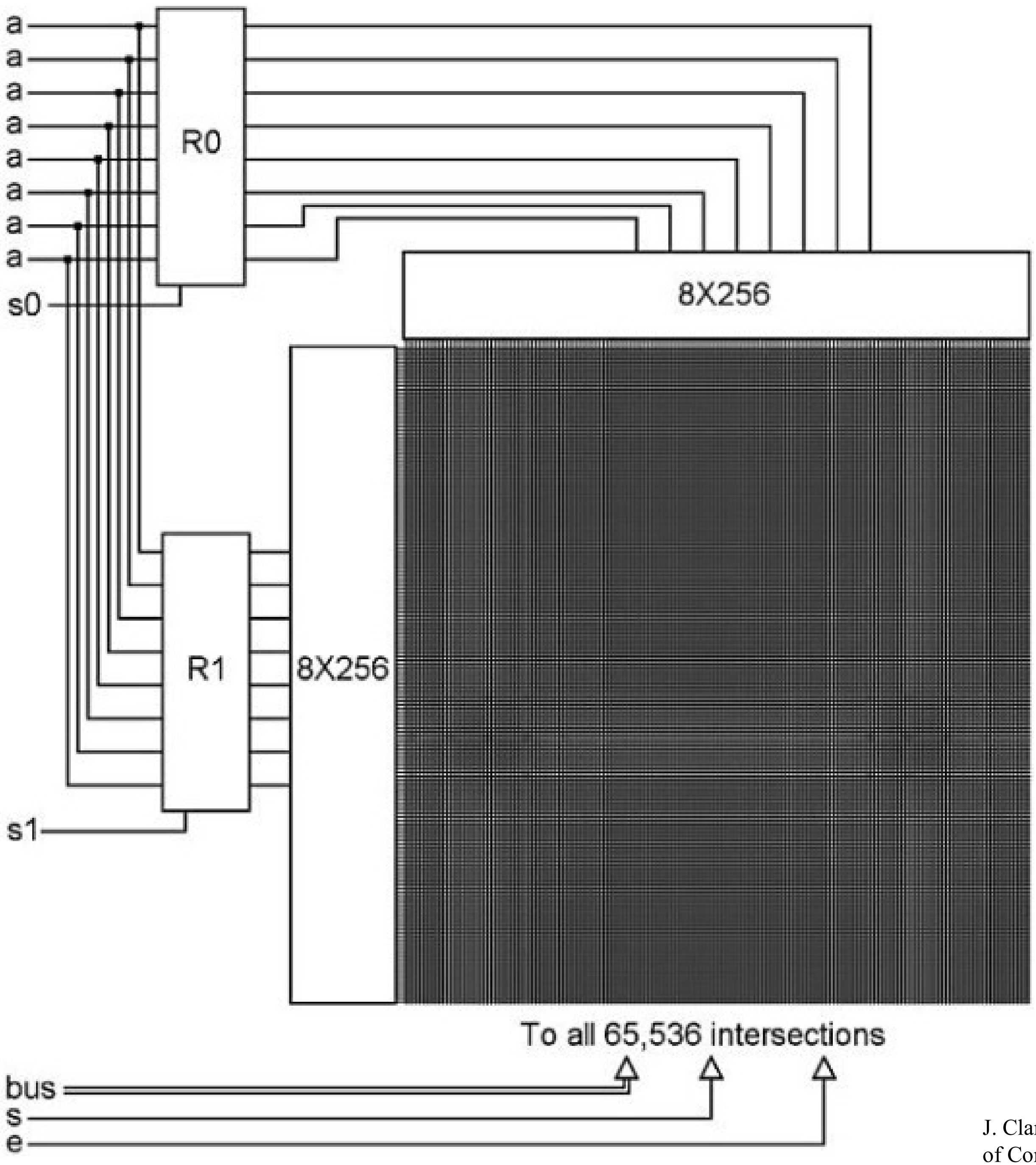
If ‘x’ gate is on, its register can be set from the bus or its contents can be enabled onto the the bus and sent elsewhere by using ‘s’ and ‘e’ bits.



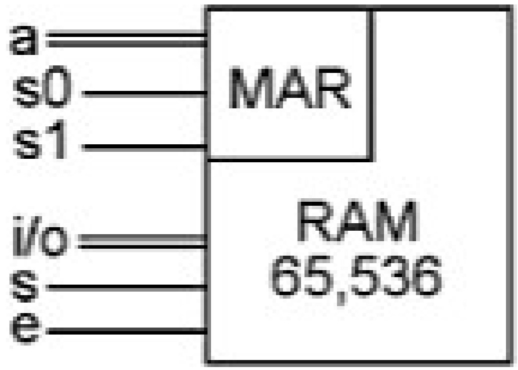
257 registers: 256 of them are memory storage locations and one register is used to select one of the storage locations and is called “Memory Address Register” or “MAR” for short.

J. Clark Scott, But How Do It Know? The Basic Principles of Computers for Everyone, (c) 2009 by J. Clark Scott.

RAM

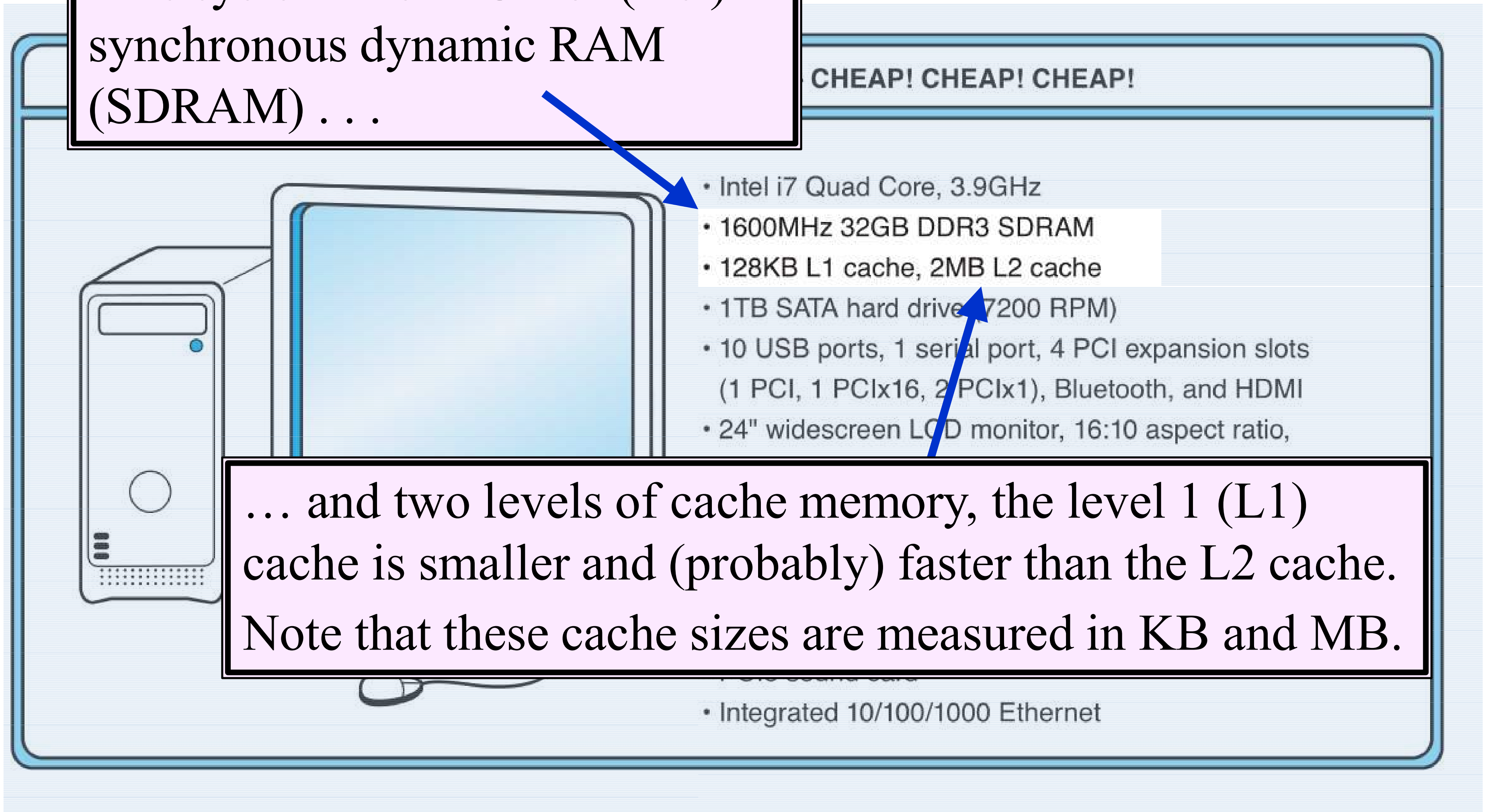


A larger RAM can be built by providing two registers that are used to select a memory storage location



An Example System

This system has 32GB of (fast) synchronous dynamic RAM (SDRAM) . . .

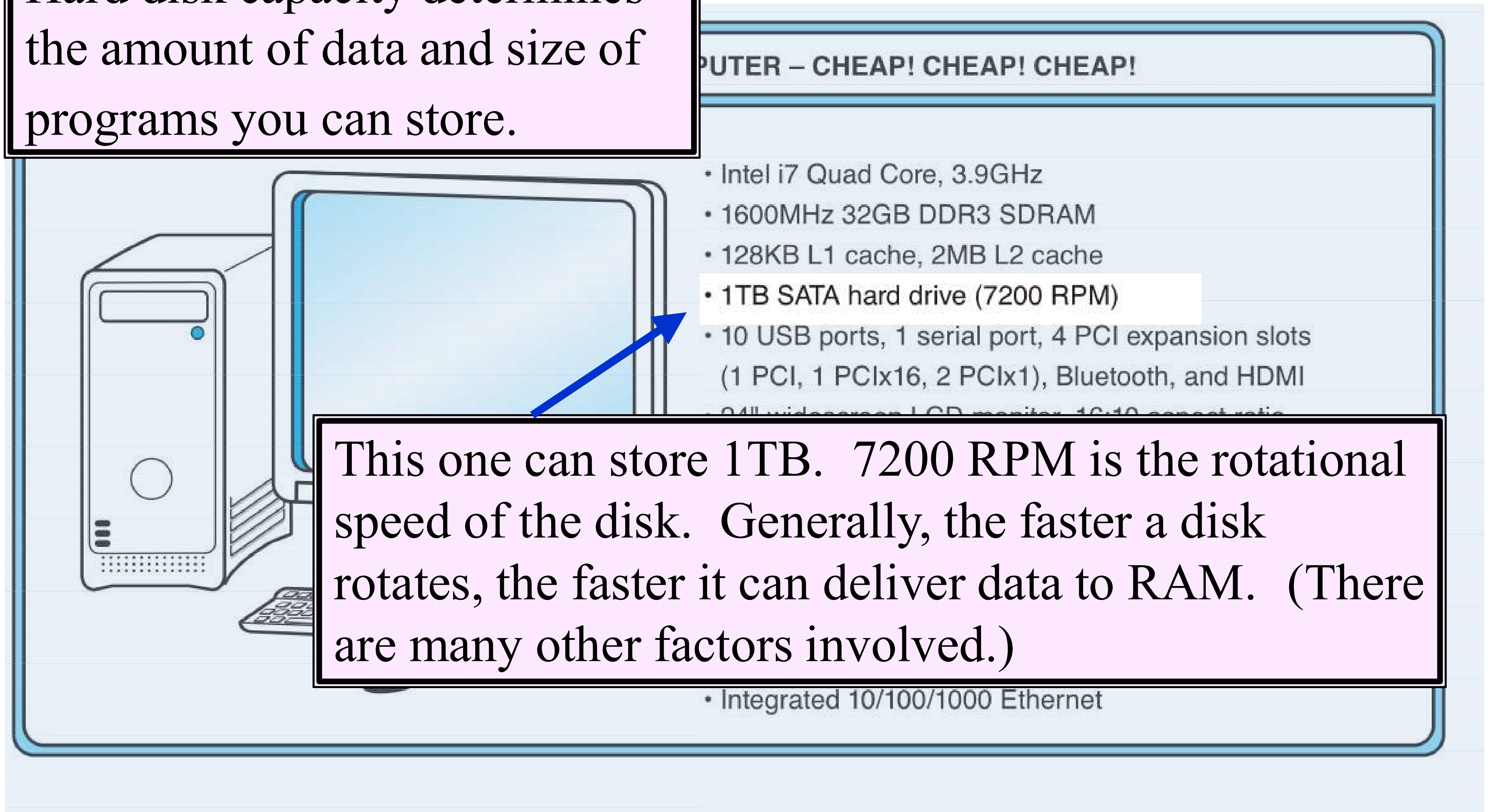


… and two levels of cache memory, the level 1 (L1) cache is smaller and (probably) faster than the L2 cache. Note that these cache sizes are measured in KB and MB.

Linda Null and Julia Lobur, The Essentials of Computer Organization and Architecture, (c) Jones and Bartlett Publishers, 2003, 2015.

An Example System

Hard disk capacity determines the amount of data and size of programs you can store.



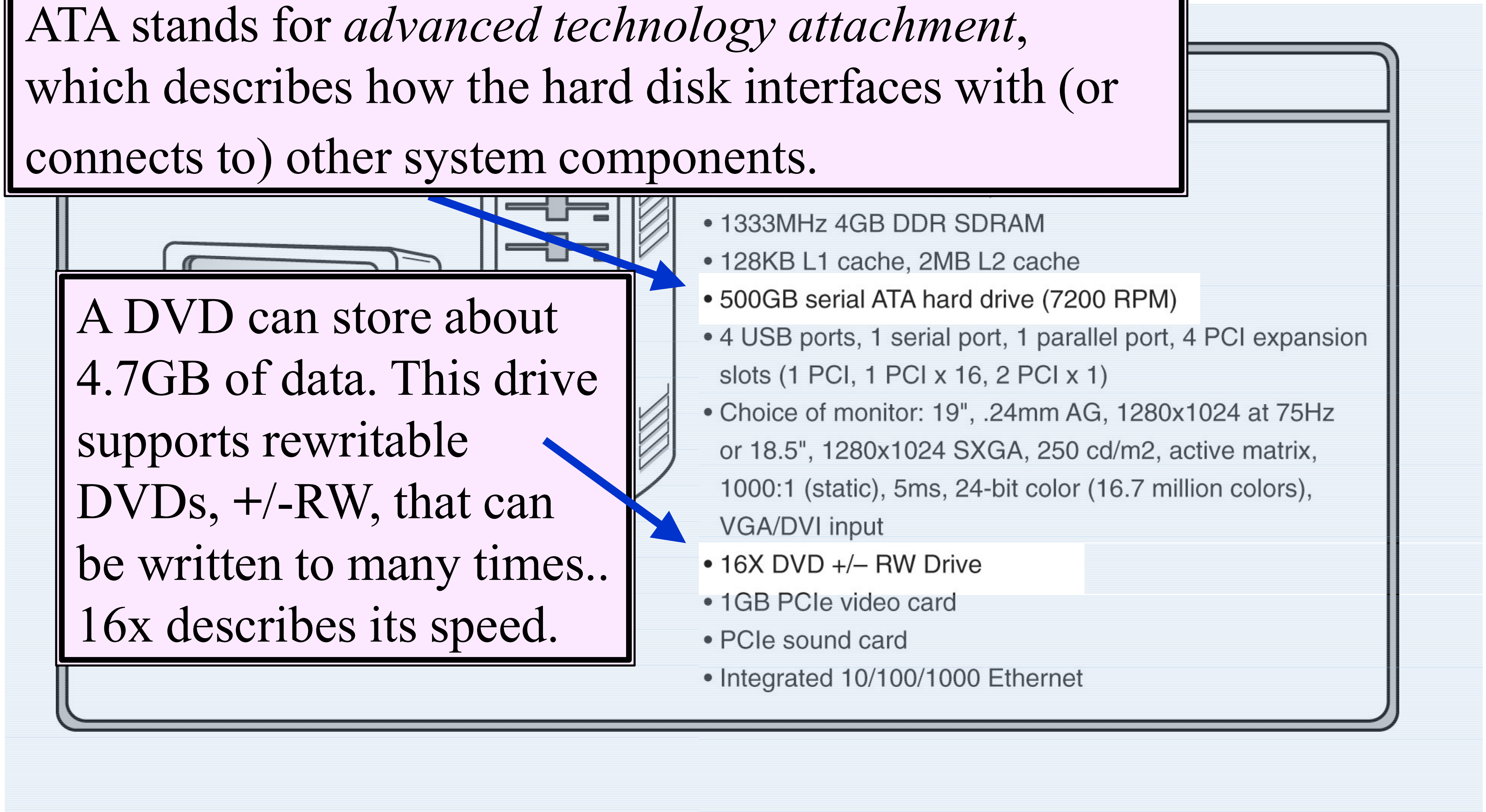
This one can store 1TB. 7200 RPM is the rotational speed of the disk. Generally, the faster a disk rotates, the faster it can deliver data to RAM. (There are many other factors involved.)

Linda Null and Julia Lobur, The Essentials of Computer Organization and Architecture, (c) Jones and Bartlett Publishers, 2003, 2015.

An Example System

ATA stands for *advanced technology attachment*, which describes how the hard disk interfaces with (or connects to) other system components.

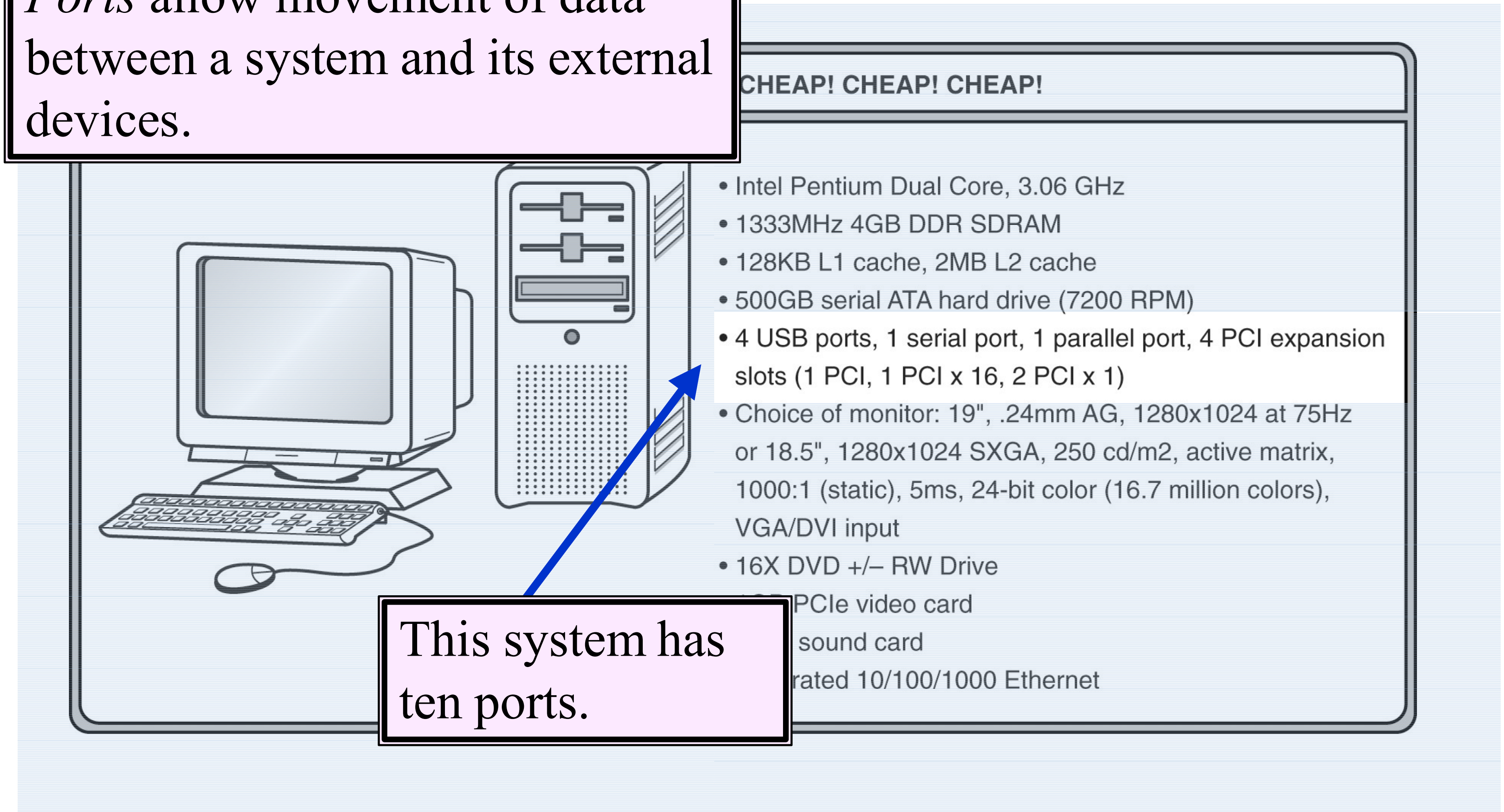
A DVD can store about 4.7GB of data. This drive supports rewritable DVDs, +/-RW, that can be written to many times.. 16x describes its speed.



Linda Null and Julia Lobur, The Essentials of Computer Organization and Architecture, (c) Jones and Bartlett Publishers, 2003, 2015.

An Example System

Ports allow movement of data between a system and its external devices.



This system has ten ports.

Linda Null and Julia Lobur, The Essentials of Computer Organization and Architecture, (c) Jones and Bartlett Publishers, 2003, 2015.

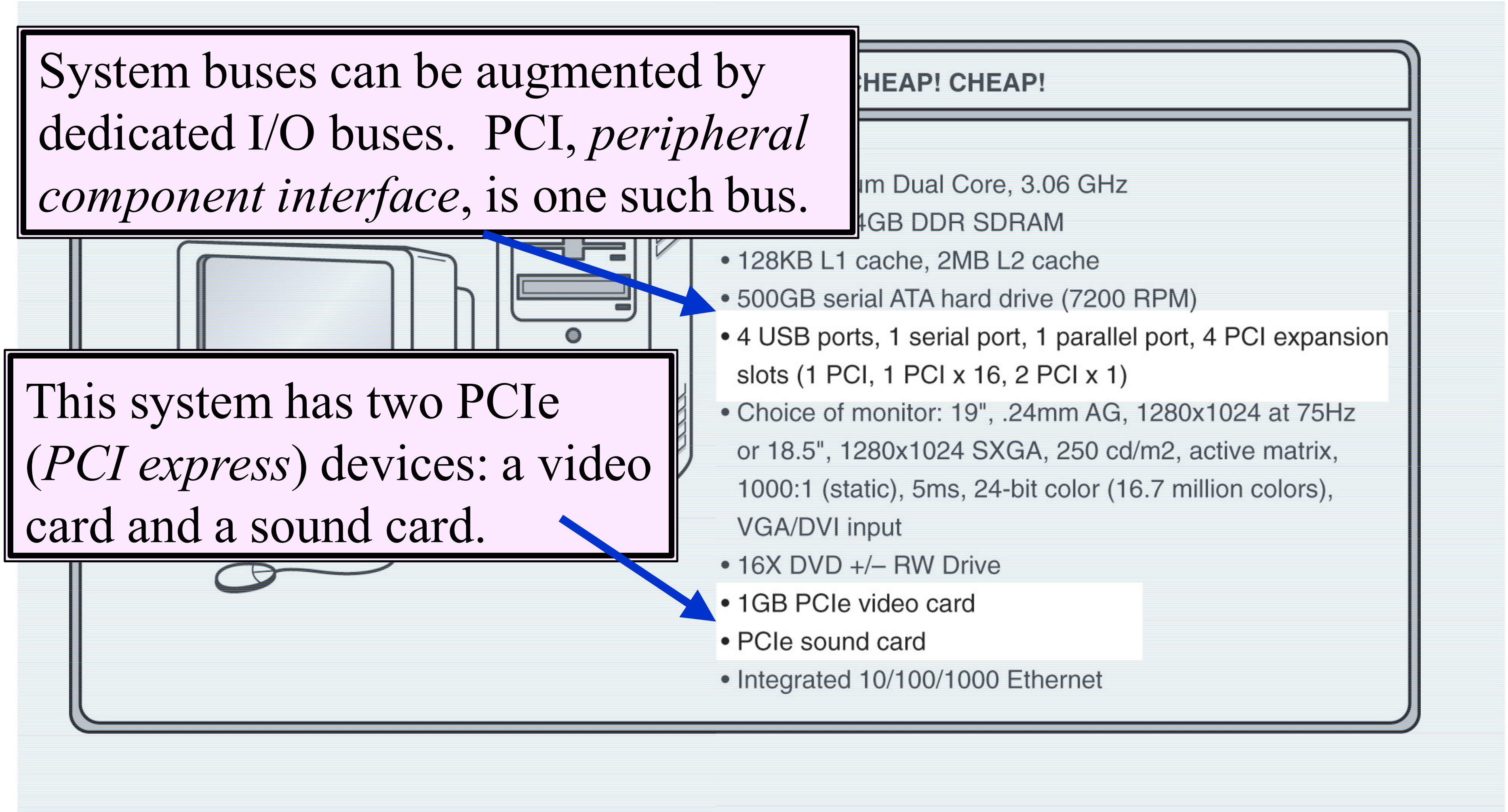
An Example System

- Serial ports send data as a series of pulses along one or two data lines.
- Parallel ports send data as a single pulse along at least eight data lines.
- USB, Universal Serial Bus, is an intelligent serial interface that is self-configuring. (It supports "plug and play.")

An Example System

System buses can be augmented by dedicated I/O buses. PCI, *peripheral component interface*, is one such bus.

This system has two PCIe (*PCI express*) devices: a video card and a sound card.

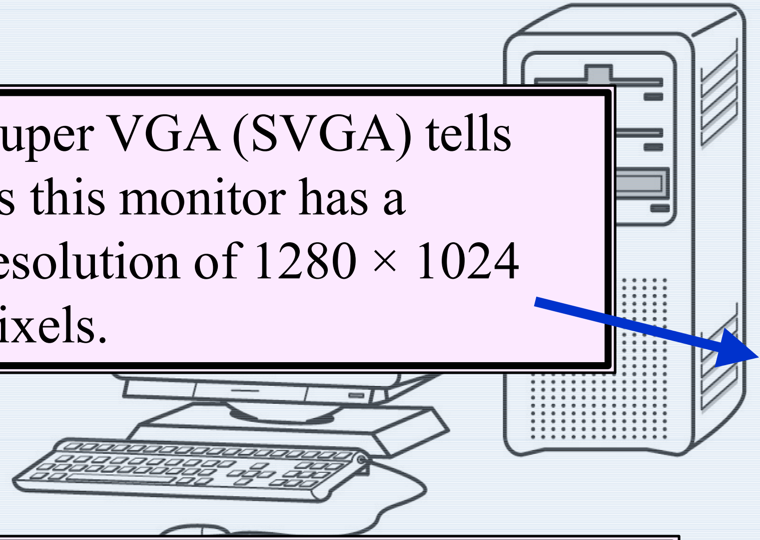


Linda Null and Julia Lobur, The Essentials of Computer Organization and Architecture, (c) Jones and Bartlett Publishers, 2003, 2015.

An Example System

Active matrix technology uses one transistor per picture element (*pixel*). The *resolution* of a monitor determines the amount of text and graphics that the monitor can display.

Super VGA (SVGA) tells us this monitor has a resolution of 1280 × 1024 pixels.

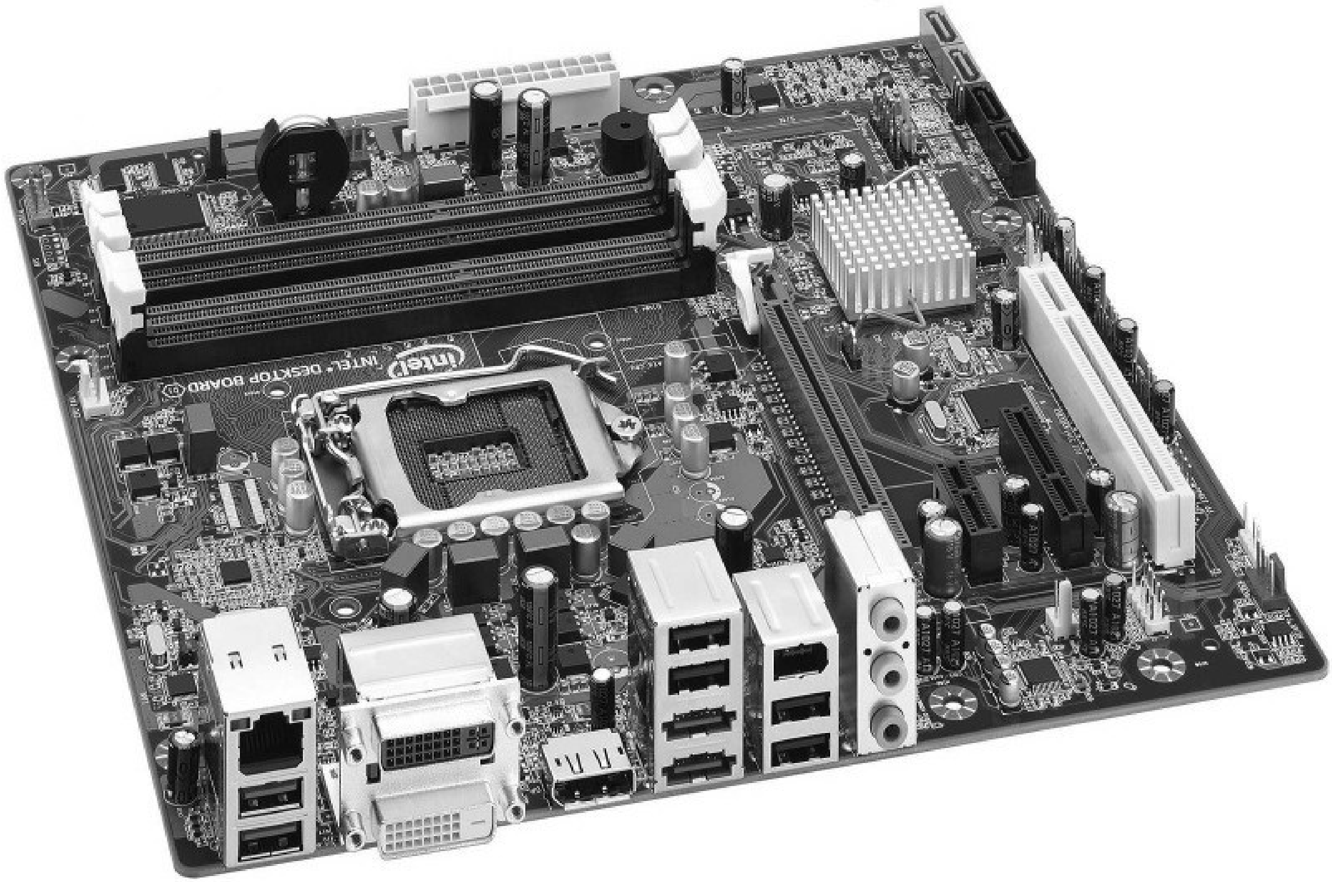


- Intel Pentium Dual Core, 3.06 GHz
- 1333MHz 4GB DDR SDRAM
- 128KB L1 cache, 2MB L2 cache
- 500GB serial ATA hard drive (7200 RPM)
- 4 USB ports, 1 serial port, 1 parallel port, 4 PCI expansion slots (1 PCI, 1 PCI x 16, 2 PCI x 1)
- Choice of monitor: 19", .24mm AG, 1280x1024 at 75Hz or 18.5", 1280x1024 SXGA, 250 cd/m2, active matrix, 1000:1 (static), 5ms, 24-bit color (16.7 million colors), VGA/DVI input
- 16X DVD +/– RW Drive
- 1GB PCIe video card
- PCIe sound card
- Integrated 10/100/1000 Ethernet

The video card contains memory and programs that support the monitor.

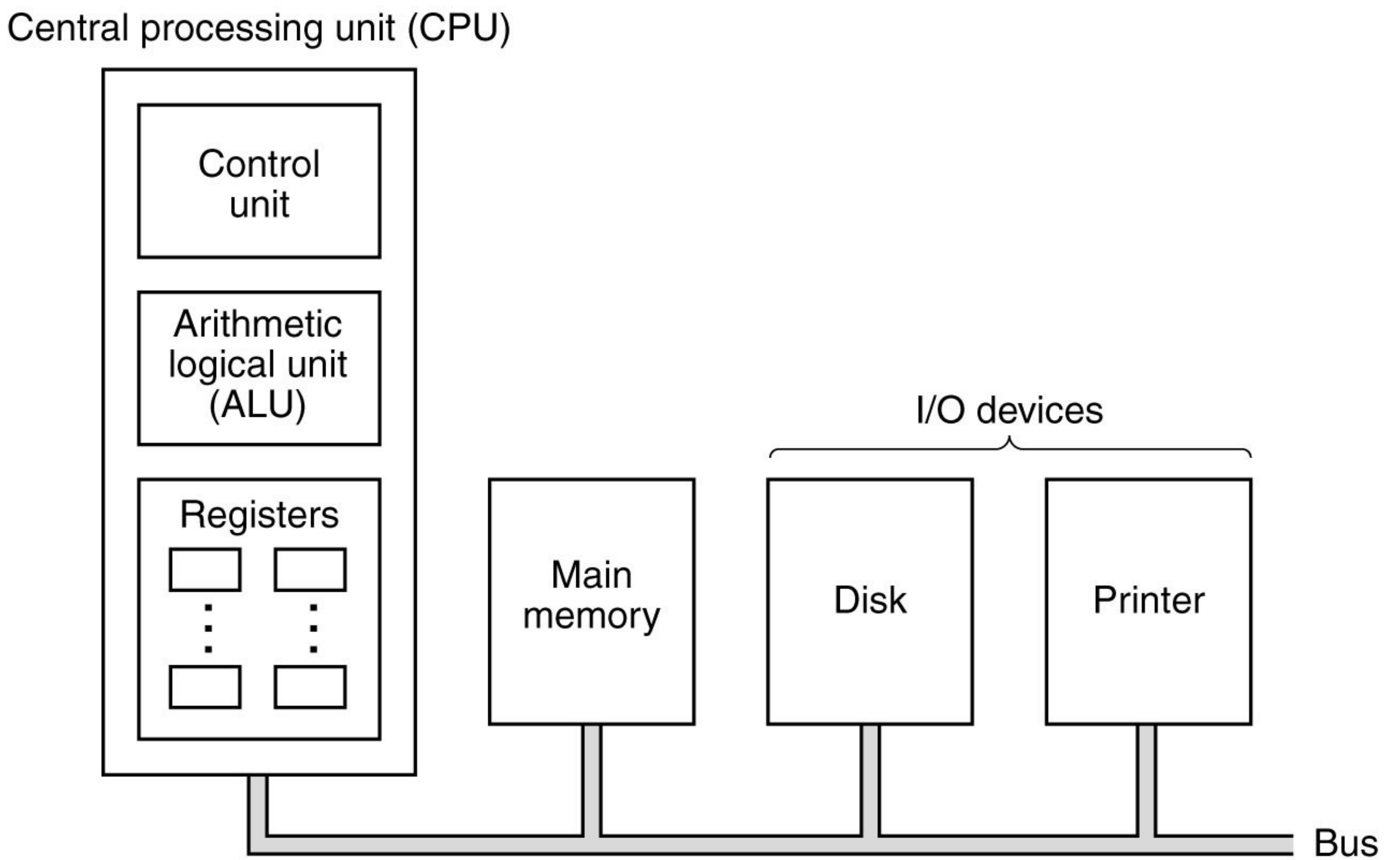
Linda Null and Julia Lobur, The Essentials of Computer Organization and Architecture, (c) Jones and Bartlett Publishers, 2003, 2015.

What are the Parts of a Computer?



Tanenbaum, Structured Computer Organization, Fifth Edition, (c) 2006 Pearson Education, Inc.

What are the Parts of a Computer?



The organization of a simple computer with one CPU and two I/O devices

Tanenbaum, Structured Computer Organization, Fifth Edition, (c) 2006 Pearson Education, Inc.

What is a Computer?

- A computer is a general-purpose
  - **electronic device** that
  - **can be programmed** to carry out a set of
  - **arithmetic or logical operations** automatically.
- A computer consists of at least
  - one **processing element**, typically a central processing unit (**CPU**),
  - some form of **memory**, and
  - **input/output** devices.
- The **CPU**
  - carries out <u>arithmetic and logic operations</u>, and
  - a <u>control unit</u> can change the order of operations in response to stored information.

Source: WIKIPEDIA

What is Hardware and Software?

<u>What is Hardware?</u>

- Collection of <u>physical elements</u> that constitutes a computer system.
- Physical parts or components of a computer, such as the monitor, mouse, keyboard, computer data storage, hard disk drive (HDD), graphic cards, sound cards, memory, motherboard, and so on, all of which are physical objects that are tangible

<u>What is Software?</u>

- <u>Instructions</u> that can be stored and run by hardware.
- Set of machine-readable instructions that directs a computer's processor to perform specific operations.

A combination of hardware and software forms a usable computing system.

Source: WIKIPEDIA

What is Architecture?

Original sense:

- Taking a range of **building materials, putting together in desirable ways** to achieve a building suited to its purpose

In Computer and Embedded Systems Design:

- Similar:

how parts are put together to achieve some overall goal

- Examples:

The architecture of a chip, of the internet, of an enterprise database system, an email system, a cable TV distribution system.

Source: WIKIPEDIA

What is Computer Architecture?

- **<u>Designing</u>** the structure & behaviour of computers

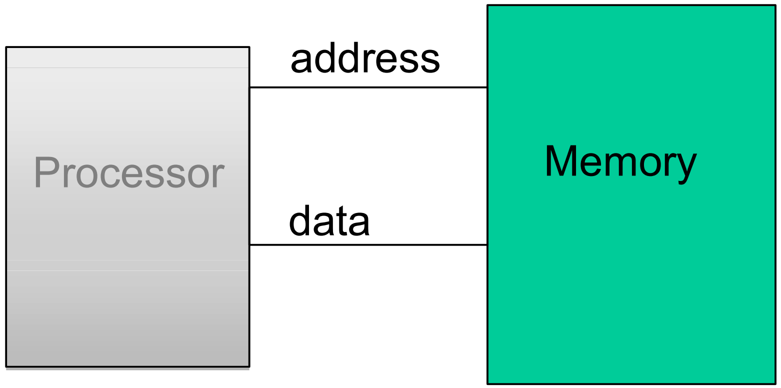
by **<u>selecting and interconnecting hardware</u>** components

to create systems that meet **<u>functional, performance and cost goals.</u>**

- <u>The study of how to design those parts of a computer system that are visible to the programmers.</u>

Main Computer Architectures

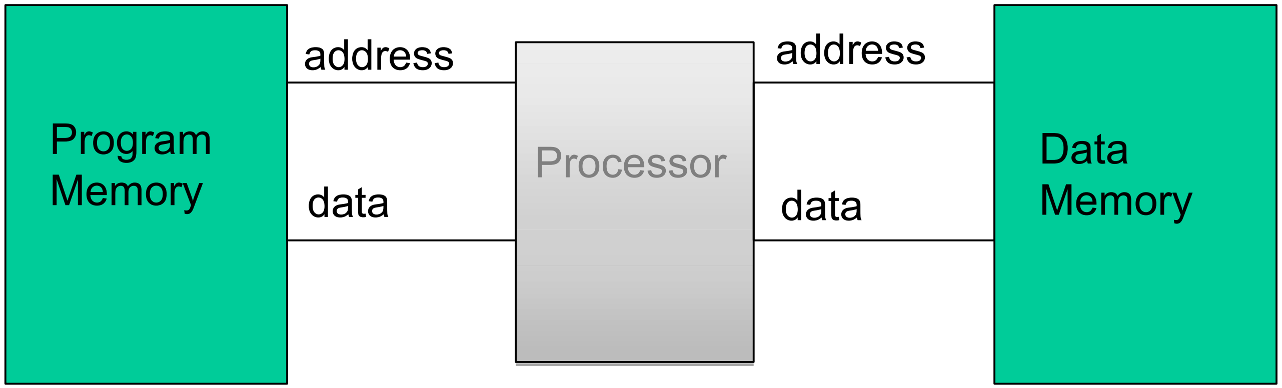
von Neumann



Same memory holds *data* and *instructions*

A von Neumann architecture has only one bus which is used for both data transfers and instruction fetches, and therefore data transfers and instruction fetches must be scheduled – they can not be performed at the same time.

Harvard



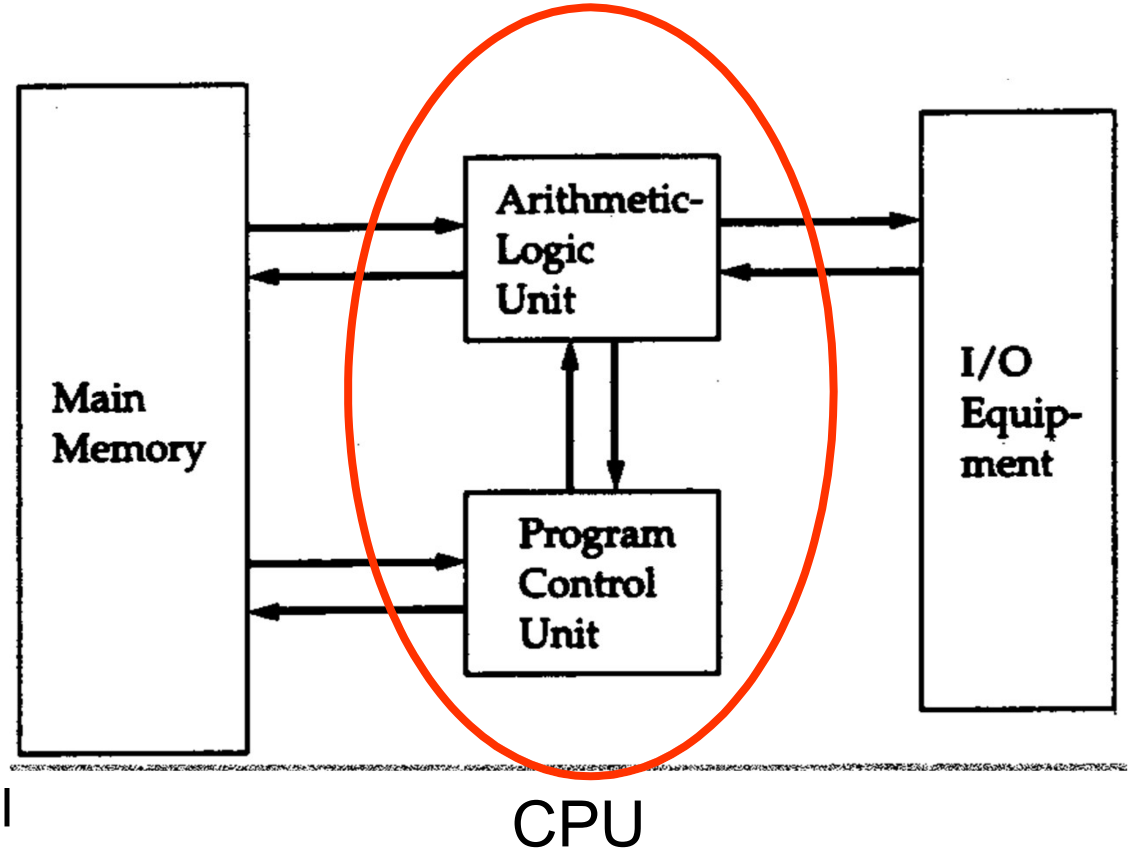
Separate memory for *data* and *instructions*

Von Neumann versus Harvard

- Two general approaches:
  - **<u>von Neumann architecture</u>**
    - **Same memory** holds *data* and *instructions*
    - Single set of address/data busses between CPU and memory
      - Most modern computers use von Neumann architecture
  - **<u>Harvard architecture</u>**
    - **Separate memory** for *data* and *instructions*
    - Two sets of address/data buses between CPU and memory
      - Harvard allows 2 simultaneous memory fetches
      - Harvard architecture is used primary for small embedded computers, microcontrollers and DSP (Digital Signal Processing)

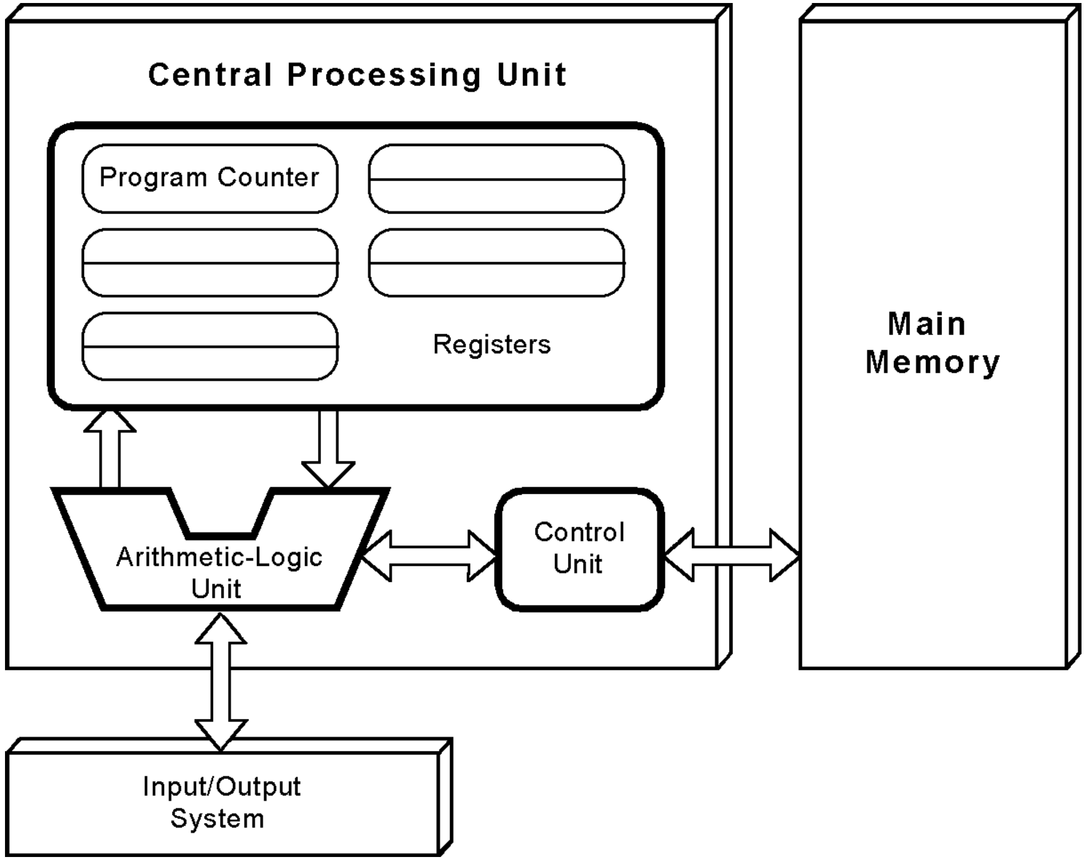
Von Neumann Architecture

- <u>Principal elements</u>
  - Central processing unit (CPU)
  - Main memory
  - I/O subsystems
  - Means of interconnecting all these components



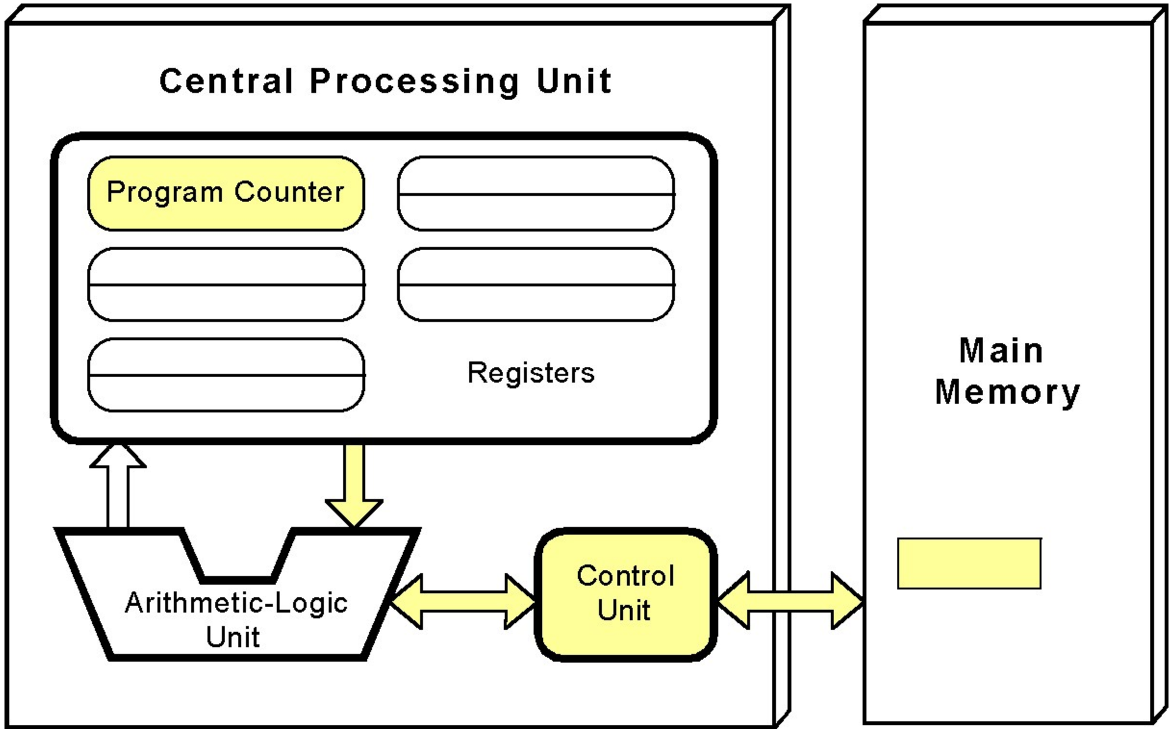
The von Neumann Model

- This is a general depiction of a von Neumann system:
- These computers employ a fetch-decode-execute cycle to run programs as follows . . .



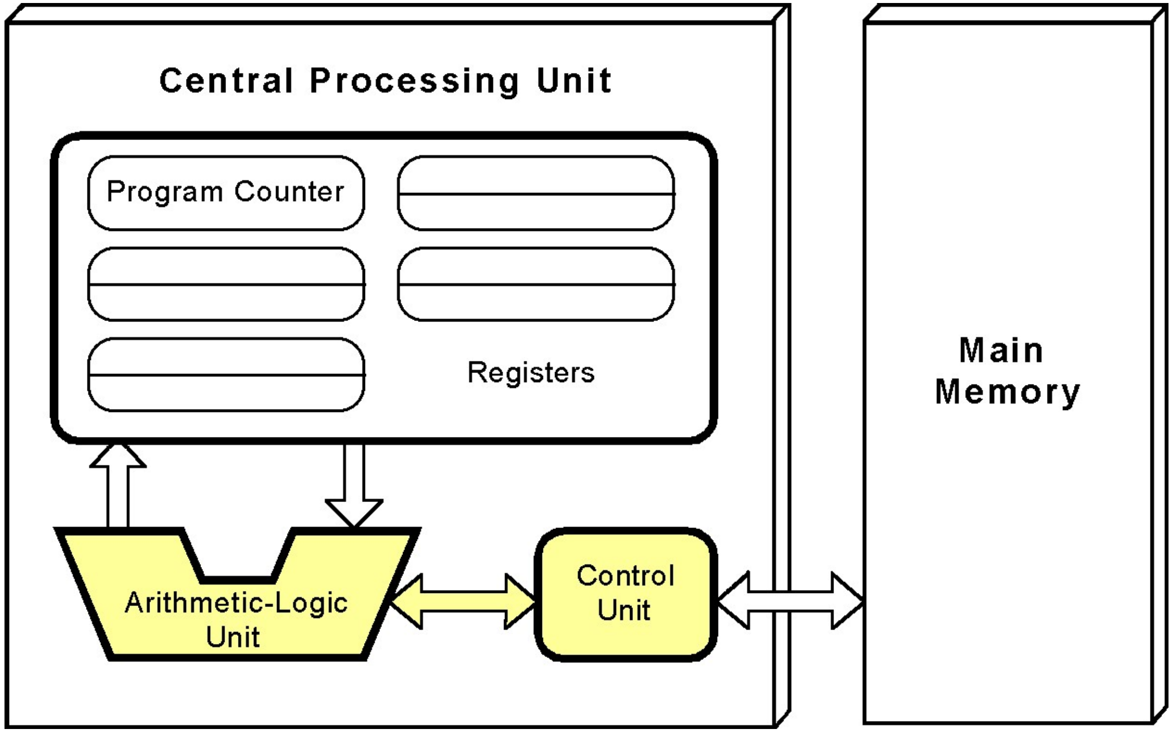
The von Neumann Model

The control unit fetches the next instruction from memory using the program counter to determine where the instruction is located.



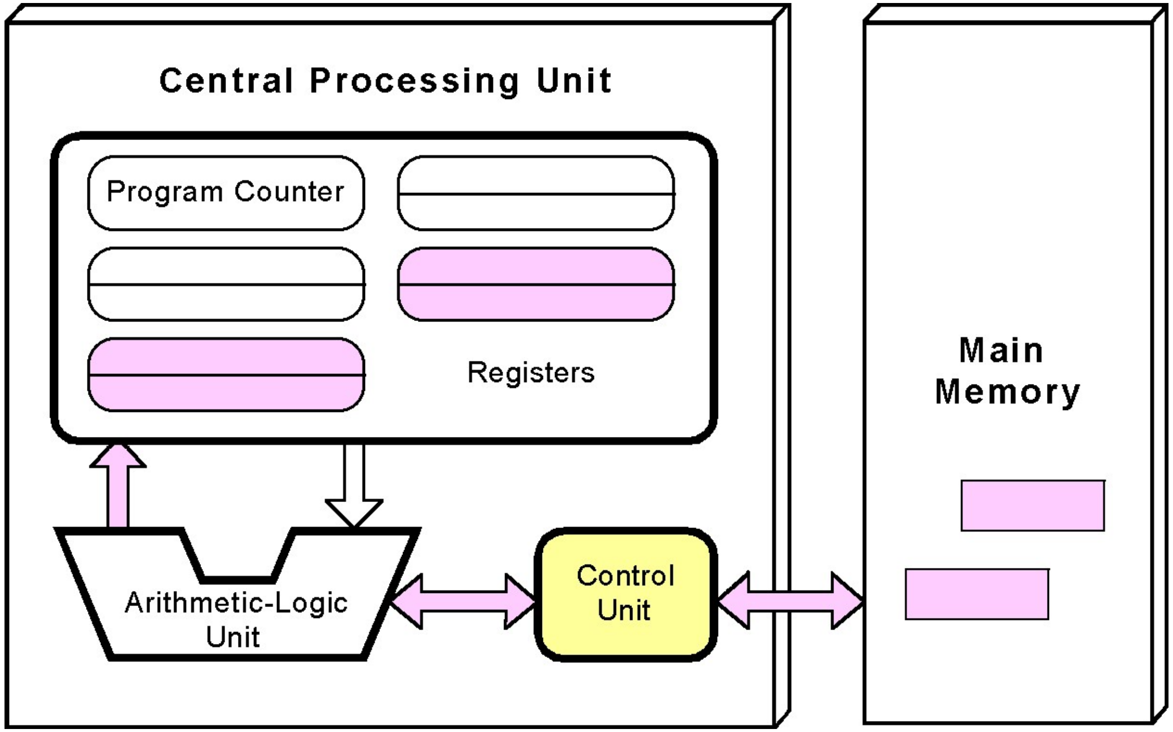
The von Neumann Model

The instruction is decoded into a language that the ALU can understand.



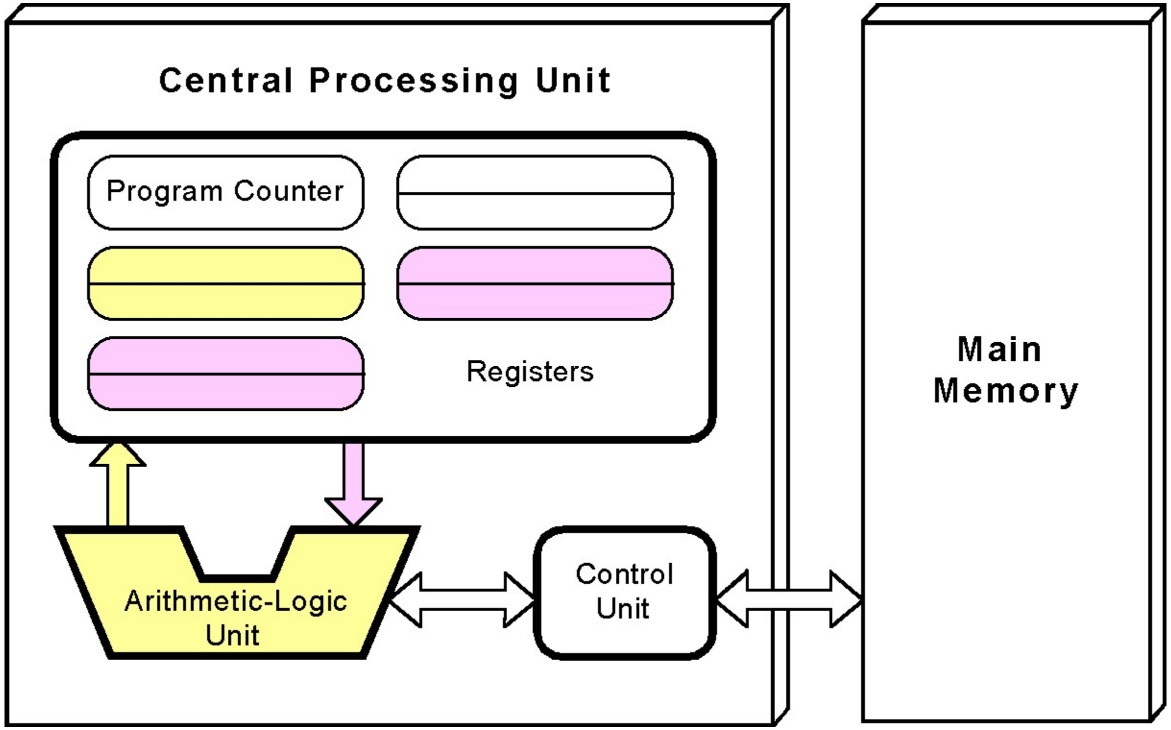
The von Neumann Model

Any data operands required to execute the instruction are fetched from memory and placed into registers within the CPU.



The von Neumann Model

The ALU executes the instruction and places results in registers or memory.



What is a Program, What is a Computer Language?

- **What is a Program?**

A sequence of instructions describing how to perform a certain task

- **What is a Computer Language?**

A computer's primitive instructions form a language in which people can communicate with the computer.

- **Levels of languages**

There is a large gap between what is convenient for people and what is convenient for computers; therefore, there are different levels of languages.

Levels of Languages for Computers

High Level Language Program (e.g., C)

Compiler

Assembly Language Program

Assembler

Machine Language Program

Machine Interpretation

Hardware Architecture Description

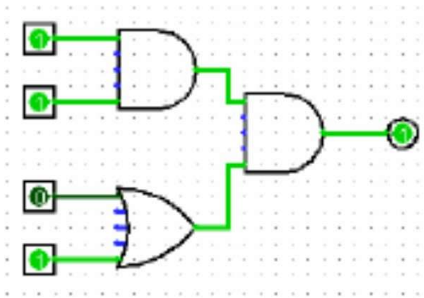
Architecture Implementation

Logic Circuit Description

```
temp = v[k];
v[k] = v[k+1];
v[k+1] = temp;
```

```
lw  $to,   0($2)
lw  $t1,   4($2)
sw  $t1,   0($2)
sw  $t0,   4($2)
```

```
0000 1001 1100 0110 1010 1111 0101 1000
1010 1111 0101 1000 0000 1001 1100 0110
1100 0110 1010 1111 0101 1000 0000 1001
0101 1000 0000 1001 1100 0110 1010 1111
```



Translators, Interpreters, and Virtual Machines

There is a large gap between what is convenient for people and what is convenient for computers. People want to do *X*, but computers can only do *Y*.

The problem can be attacked in two ways: both involve designing a new set of instructions (language L1) that is more convenient for people to use than the set of built-in machine instructions (language L0).

Translation: Replacing each instruction in L1 by an equivalent sequence of instructions in L0. The resulting program consists entirely of L0 instructions.

Interpretation: Writing a program in L0 that takes programs in L1 as input data and carries them out by examining each instruction in turn and executing the equivalent sequence of L0 instructions directly.

Both methods, and increasingly, a combination of them, are widely used.

Rather than thinking in terms of translation or interpretation, it is often simpler to imagine the existence of a hypothetical computer or **virtual machine** whose machine language is L1.

Translators and Interpreters (e.g. from level L1 to L0)

- **Translator**

Each instruction of a program in Level *k* is first replaced by an equivalent sequence of instructions in Level *k-1*.

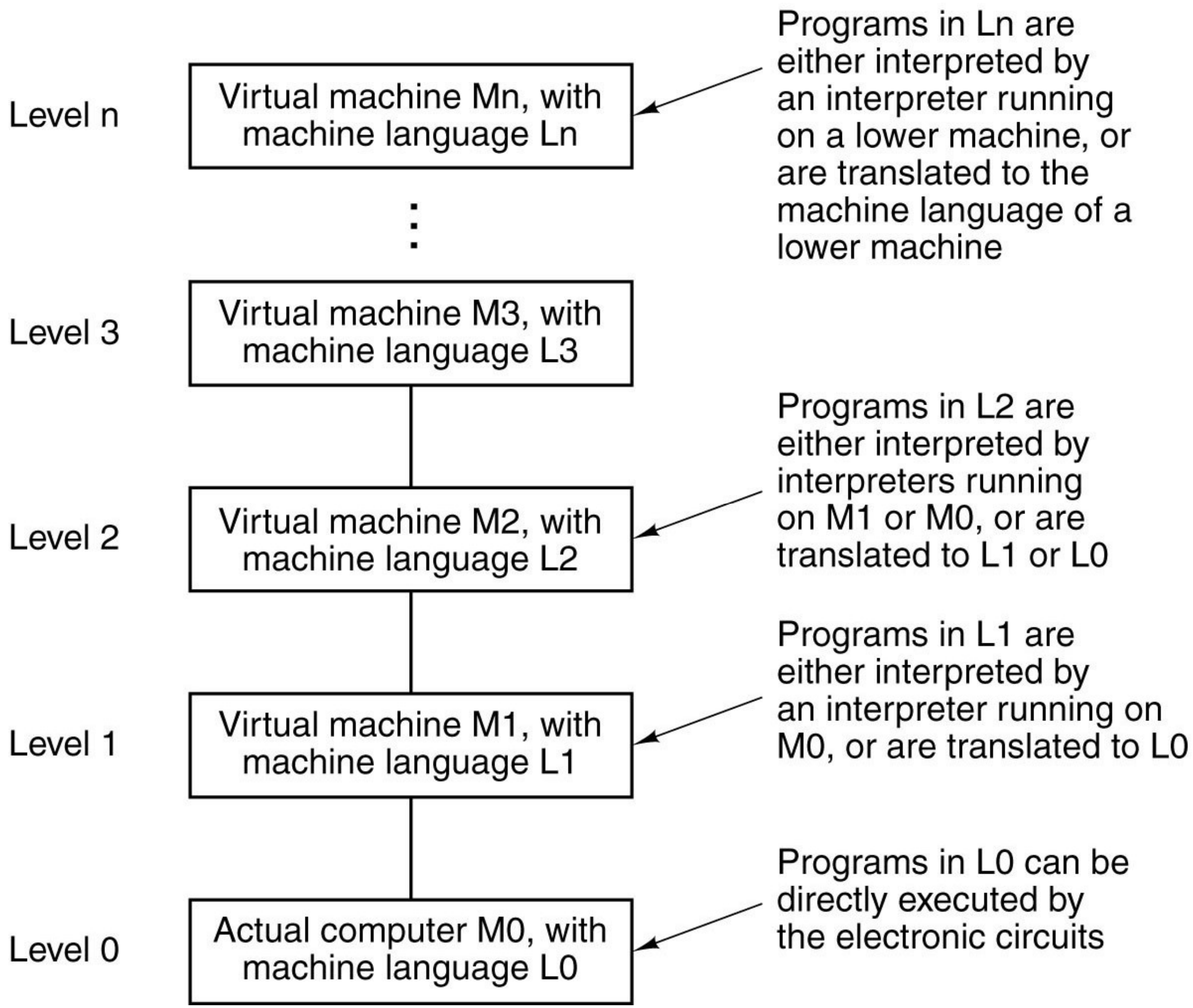
- **Interpreter**

A program in Level *k-1* takes programs in Level *k* as input data and carries them out by examining each instruction in turn and executing the equivalent sequence of Level 0 instructions directly.

- **Virtual Machine**

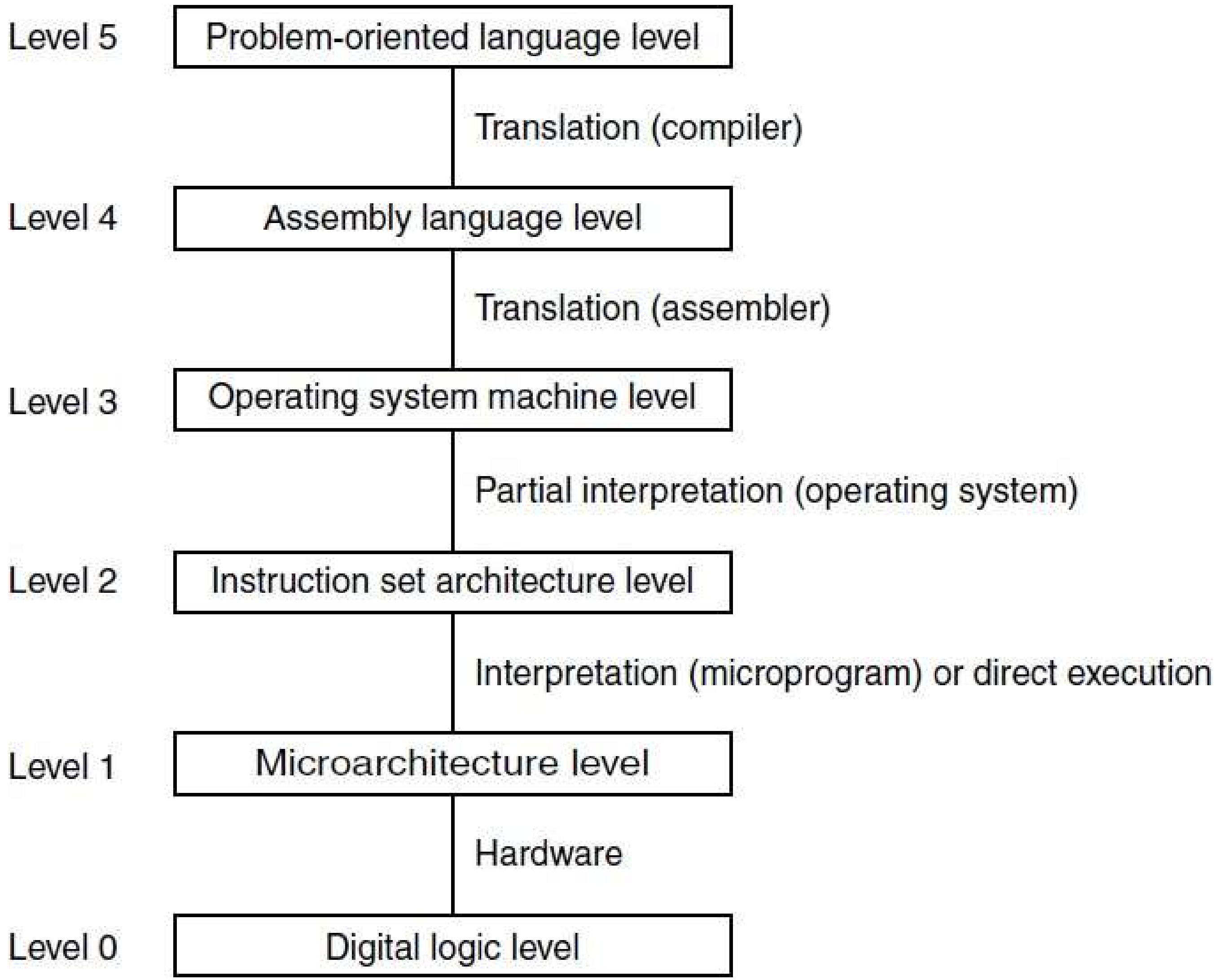
A hypothetical computer whose machine language is Level *k* and which performs automatically the translation or interpretation operations, without the user of Level *k* noticing or bothering whether it is translation or interpretation.

A Multilevel Computer



Tanenbaum, Structured Computer Organization, Fifth Edition, (c) 2006 Pearson Education, Inc.

A Six-level Computer



Tanenbaum, Structured Computer Organization, Fifth Edition, (c) 2006 Pearson Education, Inc.

High-Level Languages and Compilers

- **High-Level Languages**

Languages designed to be used by applications programmers with problems to solve, such as Python, C, C++, Java, LISP, and Prolog.

- **Compilers**

Compilers are the translators (occasionally interpreters) that translate (interpret) the high-level languages (Level 5) to lower level languages (Level 3 or Level 4).