4. Computer hardware

1. Digital processing and representation of information
2. The binary system of numerals
3. Digital computer architecture
4. I/O and operating systems

Inquiry

• Is a computer an “electronic brain”?
• Does a computer work like human symbolic reasoning?
• Why does my phone boot quickly and my laptop boot slowly?
4. Hardware

**Topic objective**

Explain basic computer operation and storage

**Essential and priority objectives**

4.0a Recall basic hardware concepts*  
4.1a Use the basic terminology of computer hardware**  
4.2b Describe the applications of binary data*  
4.3a Explain how processors execute programs*
1. Digital processing and representation of information

- What computer hardware do you use?
- What do you know about computer hardware?
- How does a phone differ from the brain?

Subtopic objectives

4.1a Recognize and use the basic terminology of computer hardware*

4.1b Distinguish digital from analog data
Why study computer hardware?

Q: How does it help a user to study hardware?

Some answers:

• Hardware changes are enabling new communication devices with new features
• Computers process information differently from humans due to different underlying architecture
• To understand representation of data, e.g., “bit,” “byte,” “gigabyte,” “pixel,” “256 Mb”

IT Hardware categories

• General-purpose programmable devices
  – desktop, laptop computers
  – mobile phones, tablets
  – entertainment devices (video game players)
• I/O devices (printers, scanners, keyboards)
• Embedded devices (modems, fuel injectors, LCD watches)
### Silicon vs. neurons

- The brain stores *analog* information
- Neurons respond to the strength of electrical impulses by *firing* (emitting an impulse)
- Computers process *bits* (binary digits)
- The brain is massively *parallel*; digital computing is *serial* today
- *Emotions* currently play little role in computing systems; play a large role in the brain

#### Architecture of a computer

- Data flows as shown by arrows, under control of processor
- Processor contains *registers*
**History of digital hardware**

- *Mechanical*: Jacquard punch-card loom, Pascal’s calculator, Hollerith punch-card census tabulation
- *Transistor* (1945): second-generation electronic computers

**Transistors**

- Invented 1947, used in a 92-transistor computer ‘53
- 10 million trillion were manufactured in 2010
- 100,000 cost the same as one grain of rice
- An application of quantum theory
- Works like an electronic valve or an electromagnetic relay
- Uses silicon to switch switches, acting sometimes like a wire (on) and sometimes like an insulator (off) to control current in circuits
4. Hardware

Digital representation of information

- In IT devices and processes, all data is represented digitally, by symbols, e.g., 0 and 1
- Analog data is on a continuum (e.g., sound waves)
- Analog data may be digitized, converted to bits, e.g., by a sound card
- Digital output may be converted to analog, e.g., by a sound card
- Information: the meaning of data

Examples of digital and analog data

- Symbols, such as words and numerals, are the form of digital data
- Images and sounds, in wave form, are analog
- Rulers are analog devices; calculators digital
- Output of a DVD reader is a digital signal later converted to analog signal and image
- Output of a digital camera is analog; image is stored digitally
- Output of a computer printing process is a digital bit stream to the printer
4. Hardware

David Keil      Introduction to Information Technology      7/15

Binary data

• All data is stored and communicated in IT as binary digits (bits)
• Bit: a single truth value or on/off switch (0, 1)
• Byte: 8 bits, capable of storing a small number or a character
• Word: 32 or 64 bits, the size value a processor can manipulate in one operation
• Access is sequential (e.g., I/O from ports) or random (e.g., RAM)

Binary physical representation

The following are equivalent ways to represent binary data:
• low voltage / high voltage (wire)
• of / on (switch)
• no pit / pit (optical storage)
• iron oxide upp-down / sideways (hard disk)
• zero / 1
• false / true
Digitizing information

- **Definition**: using discrete symbols to represent continuous information
- **Discrete**: requires detecting presence/absence (off/on; true/false)
- **Example**: numeric digits for phone dialing
- Digitizing must preserve information content as the form of data is altered

All information may be digitally encoded

- **Analog** information can be digitized by sampling or other methods such as OCR
- **Bit patterns** are arbitrary symbols that stand for other things
- **Meaning** of bits depends on interpretation by software
- **Example**: A TCP/IP packet stores *any* data available over Internet
Encoding of color image data

- **Color**: RGB monitor has 3 color elements per pixel, each of some intensity, e.g., 0 to 255 (0 to $2^8 - 1$) for 8-bit
- Computing on a representation: examples are tinting, boosting highlights on a photo
- Printers, cathode-ray-tube (CRT) and liquid-crystal diode (LCD) displays represent colors differently

Encoding of sound

- **Digitizing sound** includes sampling waves to convert analog to digital (bit) form
- **Sound card** converts analog to digital (microphone) or digital to analog (speaker)
- File formats: .wav, mp3, …
- **Compression** is a concern: reduce number of bits by using patterns
2. The binary system of numerals

- What does “543.21” mean?
- What is place value?
- How many different positions does a light switch have?
  - Two light switches?
  - Three switches?
  - n switches?

Subtopic objectives

4.2a Manipulate binary numerals
4.2b Describe the applications of binary data*
The binary system

- Appropriate for two-state devices
- *Binary* is the form in which all information is represented (numeric, text, graphical, sound)
- Uses two digits (1 and 0) rather than ten
- Like decimal, uses *place values*
- We distinguish *numerals* (representations) from *numbers* (abstractions)

Hardware representation of numbers

- Values are stored as sequences of *bits*
- One bit can store either of two values: 0, 1
- Two bits can store 4 different values
- Four bits can store $2^4 = 16$ different values
- One byte is eight bits; a byte can store $2^8 = 256$ different values, e.g., 0 .. 255
- On a 32-bit computer, a register can store up to $2^{32} \approx 4$ billion different values
- In general, $k$ bits of storage may store $2^k$ different values
Binary-to-decimal conversion

- Add the powers of 2 represented by binary digits of the numeral
- The maximum value of an \( n \)-bit binary numeral is \( 2^n - 1 \)
- An algorithm for conversion is pictured:

\[
\begin{array}{c}
11010_2 \\
0 \times 2^0 = 0 \\
1 \times 2^1 = 2 \\
0 \times 2^2 = 0 \\
1 \times 2^3 = 8 \\
1 \times 2^4 = 16 \\
\hline
26_{10}
\end{array}
\]

Binary-to-decimal algorithm

Input \( x \)
\( y \leftarrow 0 \)
\( pv \leftarrow 1 \)
\( i \leftarrow \#\text{bits}(x) \)
While \( i > 0 \)
\hspace{1cm} If \( x_i = 1 \)
\hspace{2cm} \( y \leftarrow y + pv \)
\hspace{2cm} \( pv \leftarrow pv \times 2 \)
\hspace{2cm} \( i \leftarrow i - 1 \)
Display \( y \)

Note \( x_i \) is the \( i \)th bit of string \( x \).
Example: If \( x = '10' \) and \( i = 2 \),
then \( x_i = '0' \)
4. Hardware

Decimal-to-binary conversion

Two methods:

(a) Repeatedly divide decimal numeral by 2, jotting down remainder, right to left

\[
\begin{align*}
25 \div 2 &= 12 \text{ R 1} \\
12 \div 2 &= 6 \text{ R 0} \\
6 \div 2 &= 3 \text{ R 0} \\
3 \div 2 &= 1 \text{ R 1} \\
1 \div 2 &= 0 \text{ R 1}
\end{align*}
\]

\[11001_2\]

(b) Find the set of powers of 2 that add up to the decimal value; record each as a binary 1, left to right

\[
\begin{align*}
16 & \quad 8 \\
1 & \quad 1 \quad 25_{10}
\end{align*}
\]

3. Digital computer architecture

- How many processing units does the brain have?
- A smart phone or laptop?
- How many events can occur at the same time in a computer?
- How does a computer perform its operations?
4. hardware

Subtopic objectives

4.3a Explain how processors execute programs*
4.3b Test an assembler-language program and describe its results†
4.3c Write and test an assembler program†

Logic gates

• Used to manipulate binary data
• 1 or 2 bit input, 1-bit output
• Specified using truth tables
• NOT (negation)
• AND (conjunction)
• OR (disjunction)
• Used as components of more complex circuits: adders, etc.
Gates and truth tables

<table>
<thead>
<tr>
<th>Gate</th>
<th>Schematic</th>
<th>Truth table</th>
</tr>
</thead>
</table>
| NOT  | ![NOT Gate Schematic](image) | \[
| a    | 1 | 0 \\
| 0    | 1 |
| a     | 0 | b       | a or b |
| 0     | 0 | 0       | 0       |
| 0     | 1 | 1       | 1       |
| 1     | 0 | 1       | 1       |
| 1     | 1 | 1       | 1       |

| OR    | ![OR Gate Schematic](image) | \[
| a    | 0 | b | a or b |
| 0     | 0 | 0 | 0       |
| 0     | 1 | 0 | 1       |
| 1     | 0 | 0 | 0       |
| 1     | 1 | 1 | 1       |

| AND   | ![AND Gate Schematic](image) | \[
| a    | 0 | b | a and b |
| 0     | 0 | 0 | 0       |
| 0     | 1 | 0 | 0       |
| 1     | 0 | 0 | 0       |
| 1     | 1 | 1 | 1       |

XOR and NOR circuits

| XOR | ![XOR Gate Schematic](image) | \[
| a    | 0 | b | a xor b |
| 0     | 0 | 0 | 0       |
| 0     | 1 | 1 | 1       |
| 1     | 0 | 0 | 1       |
| 1     | 1 | 1 | 0       |

• Problem: Can you build a NOR gate from NOT, AND, and OR gates?
Combinational circuits

- The XOR and NOR circuits are examples of hardware that combines gates to process data
- An *adder* circuit performs addition on binary numeric data stored in registers of a processor
- A *flip-flop* circuit stores a bit; as in RAM

Hardware units

- *Fundamental units*: Gates, flip-flops
- *Processor* (CPU): composed of control and arithmetic logic units; resides on mother board
- *Memory*: ROM (read-only memory), RAM (random-access: addressable, volatile)
- *Peripherals handle input/output and long-term storage*: monitor; keyboard; mouse; disk; hard disk; DVD; flash memory
- *Ports* (e.g., USB, Ethernet): channels used to communicate with peripherals and network
Data storage

- Registers are within processor; are high-speed
- Memory is
  - Cache (high speed within processor)
  - Read-only (e.g., programs that start the computer) or
  - Random-access (RAM, electronic speed) or
  - Secondary (e.g., hard disks; a mechanical speed, slow)
- Memory sticks are RAM that appears to the user as a hard disk
- RAM is accessible by numeric addresses

General-purpose computers

- In this course, we will use “computer” to refer to general-purpose, or stored-program, devices.
- IPods and cell phones may be programmable, but not in the general-purpose sense
- Program: A series of instructions to enable a computer to perform a task.
- Kinds of computers: desktops, notebooks, handhelds, tablets, mainframes, supercomputers
- Computers may be serial, or parallel (multiprocessor)
4. Hardware

Programs are data

- The idea of a *stored program computer* relies on the idea of storing a program as data
- John von Neumann originated this idea in the 1930s
- Alan Turing, a British logician, developed an equivalent idea at about the same time
- Until this concept, every kind of computation had required its own specialized hardware

The PC and mobile eras

- Based on microprocessor
- App that expanded PC market for business: Visicalc spreadsheet
- PC-DOS, MS-DOS operating systems
- *Macintosh* with graphical user interface and mouse (1984)
- *Internet* opened computing to millions, mid-1990s
Program execution

- Processor runs a *fetch-execute* cycle
  - repeat to obtain instruction from consecutive locations in RAM
  - then carry out instruction, updating counter
- Instruction format: *opcode/mnemonic; operand*
- *Registers* (storage locations on processor):
  - program counter
  - instruction register
  - data registers (e.g., accumulator)
- Instructions may *transform* or *copy* data

Input/output

- I/O is under control of the processor
- I/O *peripherals* interface with RAM
- In our model, keyboard input and screen output are implemented by instructions
- In actual computers
  - I/O instructions are much more complex
  - *Device drivers* are device-specific software interfaces
  - *External storage* (HD, CD, flash):
    low-speed, nonvolatile
A model computer

- RAM (random-access memory) contains programs and data

An assembler-language program

```assembly
// echo.asm; echoes input
input a
output a
stop

Sample input/output:
[Input:] 26
[Output:] 26
```
The fetch-execute cycle

```
PC ← 0
Repeat
    IR ← instruction at MEM(PC)
    PC ← PC + 1
    Execute operation in IR
until IR = STOP
```

- $PC =$ Program Counter register
- $IR =$ Instruction Register
- $← =$ “gets” (variable assignment)

Assembler and machine languages

- A machine language, expressed in binary, is the set of instructions that a processor responds to.
- An assembler language is expressed as mnemonics (abbreviations for instructions) and labels for data and lines of instruction code.
- Machine and assembler are not human-friendly.
- Each processor (Pentium, G3, etc.) is designed with its own machine language.
**The model computer’s language**

- **input**: waits for user to key a number; stores that number at location specified in operand
- **output**: displays on screen the number stored at the operand address
- **load**: copies to ACC the value stored at the operand address
- **store**: copies a value from ACC to a RAM address
- **add (sub)**: adds (subtracts) value at operand address to value in ACC, storing result in ACC

**Variables**

- A *variable* is a named memory location that stores a value
- Variables in programs must be declared:
  
  ```
  sum data 0  
  int sum;  
  ```

  (assembler)  
  (JavaScript)

- Variables are assigned values
  
  ```
  load 2  
  store sum  
  ```

  (assembler)

  (sum ← 2)
4. Hardware

A program to add

\[
\begin{align*}
\text{input } & \text{ input1} \\
\text{input } & \text{ input2} \\
\text{load } & \text{ input1} \\
\text{add } & \text{ input2} \\
\text{store } & \text{ sum} \\
\text{print } & \text{ sum} \\
\text{stop} & \\
\end{align*}
\]

\[
\text{sum } \leftarrow \text{input1 + input2}
\]

`input1 data 0`  
`input2 data 0`  
`sum data 0`

// ADD.ASM: Displays sum of 2 inputs

- `input` copies data from keyboard to RAM
- `load` and `add` alter contents of ACC
- `store` copies from ACC to memory

Processor-simulator software

- See D. Keil for P10 software
- This software enables the user to create and edit programs in a 10-instruction assembler language
- …and to step through them visually on a simulated processor
- Purpose: to help understand the fetch-execute cycle, processing of data in the CPU, and motion of data between CPU and RAM
4. Hardware

**Microprocessor features**

- **Word size**: number of bits processed in a register (in 2006, usually 32 or 64)
- **Clock**: synchronizes system; speed is measured in MHz (millions of cycles/sec)
- **Cache**: high-speed memory in the processor mirroring some addresses in RAM
  
  Registers ↔ Cache ↔ RAM ↔ Hard disk

- **Pipelining**: Overlapping the steps to execute two or more instructions (like conveyor belt)
- **Parallel** processing (e.g., dual core) executes two or more instructions simultaneously on different CPUs

4. I/O and operating systems

- What storage does your laptop have?
- How do computers communicate internally?
- How is information displayed?
- How is information stored?
4.4 Describe how operating systems manage memory and input/output

Operating systems

- The OS is *system software* that runs whenever the computer power is on
- The OS
  - Provides services, such as file management, to other programs and to the user
  - Loads and runs application programs
  - Manages device I/O via *driver* software
  - Manages sharing of memory
Expansion ports

• Support communication between RAM and peripherals under processor control
• Kinds of connection (most are ports, on back):
  – Keyboard
  – Mouse
  – USB (general-purpose)
  – Speaker, microphone
  – Video
  – Ethernet network connector
  – Modem (phone jack)
  – Wireless Ethernet (not a port)

Buses

• A bus is a connection that transfers data between the CPU, RAM, and peripherals
• Local or front-side bus connects CPU and RAM
• Bus clock speed: now about 1 GHz (1 billion cycles/sec)
• Bus width: number of bits transferred, in parallel
Display technologies

- **Kinds:**
  - *LCD* (liquid crystal diode): flat screen, more expensive
  - *Plasma*: manipulates miniature fluorescent lights; expensive
  - *LED* (light emitting diode)

- **Screen resolution:** Number of pixels horizontal by vertical; e.g., $1024 \times 768$

Special input devices

- **Touch screens:** newer versions capture multiple touch locations
- **RFID** (Radio frequency identification): tiny antennas and transmitters embedded in cards or other media
- **Bar-code readers** (ISBN, postal, UPC, QR)
- **Optical mark readers:** for check deposit
- **Optical character recognition** for conversion of images of text to text files
- **Biometric readers:** fingerprint, voice
RFID tags

- Contain antenna that broadcasting information either with battery (active) or with energy from card reader (passive)
- Often are attached to merchandise at a store
- Used in parking-lot and other access
- Enable inventory tracking

Device drivers

- Peripherals’ interactions with CPU and RAM are managed by *device driver* software
- Device drivers come with hardware, are available online from manufacturer, or are standard part of operating system (*Plug and Play*)
4. Hardware

Multi-function I/O devices

- Devices are available for $200 and up to
  - Print, producing hard copy of electronic files
  - Scan images to .jpg bitmap files
  - Copy; i.e. scan and print
  - Fax images via phone lines
- Some such devices may be driven from a network; e.g., sending a document to be printed from an office to a copy room

Hard disk drives

- Hard drives store bits as magnetic orientations of iron oxide particles
- Drive has multiple spinning disks, each accessible by a read/write head
- Disk is divided into tracks (concentric circles of data locations), sectors (pie slices)
- Access time is determined by seek (time for head to reach track) and latency (time to wait for sector to spin into place)
4. Hardware

Flash memory and USB sticks

- USB (Universal Serial Bus) is a standard for manufacture of I/O ports
- One device usable via the USB port is flash memory sticks
- Flash memory provides solid-state storage, as opposed to mechanical or volatile storage
- Capacity in 2013: up to 64GByte
- Flash memory is also used in digital cameras and in smart phones

Kinds of data storage

- Registers in processor enable operations on data
- Cache memory data is easily moved to/from registers
- RAM (random-access memory) is volatile
- ROM (read-only memory) is non-volatile, stores operating system’s BIOS (basic I/O system)
- CMOS requires battery power, stores long-term but updatable specs
- Hard disk enables long-term storage
- Flash memory sticks are portable and non-volatile
- CDROM, DVD, and Blu-ray disks use optical storage technology
Power-saving methods

- *Sleep* (standby): documents being edited stay in RAM; hard disk and monitor turn off
- *Hibernate*: documents are saved to disk while power is down
- To wake up PC, press a key or move mouse
- PC may be set to power off monitor and hard disk after a fixed time of inactivity
- *Windows*: see Control Panel, *Power Options*

Concurrent processing

- *Pipelined execution* enables execution of one instruction in parallel to fetch of next
- CPU may be augmented by *graphical processing unit* (GPU) to handle screen graphics
- *Dual- or multi-core processing* may allocate different processes or threads (e.g., OS, app, and virus scan) to multiple processing units on one or more chips
4. Hardware

Multitasking

• An OS enables multiple programs to execute concurrently
• Processor devotes time slices to each program or process in round robin fashion
• Bar at bottom of Windows screen shows some programs running
• Windows Task Manager shows all processes running at a particular time
• Programs may run in background

Parallel and distributed processing

• Parallel processing with many or thousands of processors is used for weather modeling and other problems that lend themselves to parallel program execution
• Distributed computing coordinates networked computers on a single problem
Cell-phone technology

- Cell phones are IT, have processors, run apps, and are now general-purpose computers
- A cell is a geographic area with a tower and with transmit/receive hardware
- Voice sound (analog data) is converted to digital form by a chip
- Digital signal processor compresses bits to be transmitted in packets via radio waves, decompresses at receiving end for conversion to sound

Meta-data in this topic

- Operand addresses
- Variable names
- File-format descriptors
Users co-create their computing environments

- You configure your own computing system by your buying and other decisions
- To assure usability, the buyer checks features and compatibility of hardware purchased

References


Pinard and Romer. CMPTR. Course Technology, 2012.