Basics of Computer Systems

Peter Manshausen

Sommerakademie in Leysin, August 2016

Abstract

Computers are central to our modern day life and come in a variety of forms. Therefore it is important to understand how they work and what they actually do. We develop two models of how to think about computers and organize their parts into categories, thus showing what the basic building blocks are, every computer needs, most importantly input/output devices, memory and a processor. We describe how these hardware components work and interact on a basic level, giving an example of a typical computer system in the form of the Apple iPad 2. Thereafter, we concern ourselves with the hierarchical organization of software which makes it possible for humans to interact with a machine using the binary language ones and zeroes. In the end, we make a short excursus to understand the booting of a computer.

Contents

1. Introduction: Significance of Computer Systems

Since the first developments in information technology, computers have completely changed all aspects of our lives: From their (first) use in cryptography and the military over business applications to the entertainment industry, today everything relies on computers. Because of this wide field of purposes what we call a computer can take almost every shape and size: from modern Warehouse-Scale-Computers to the tiny embedded systems controlling a traffic light. Also, new concepts of computing are emerging, e. g. cloud services in place of single smaller servers and personal mobile devices (PMDs) like smartphones or tablets, replacing older technologies.

2. Models for understanding computers

This large amount of different forms computers can take makes ist necessary to develop models for what a computer is and how it works in general, which are applicable to all computers we encounter and help us to understand and categorize their parts.

2.1. The Calculator Model

The most naive way to think about what a computer does is the so called Calculator Model[1], in which there is an input of data and instructions into the computer - the numbers and the operation we want the computer to perform on them - and an output of the operation's results.

2.2. The File Clerk Model

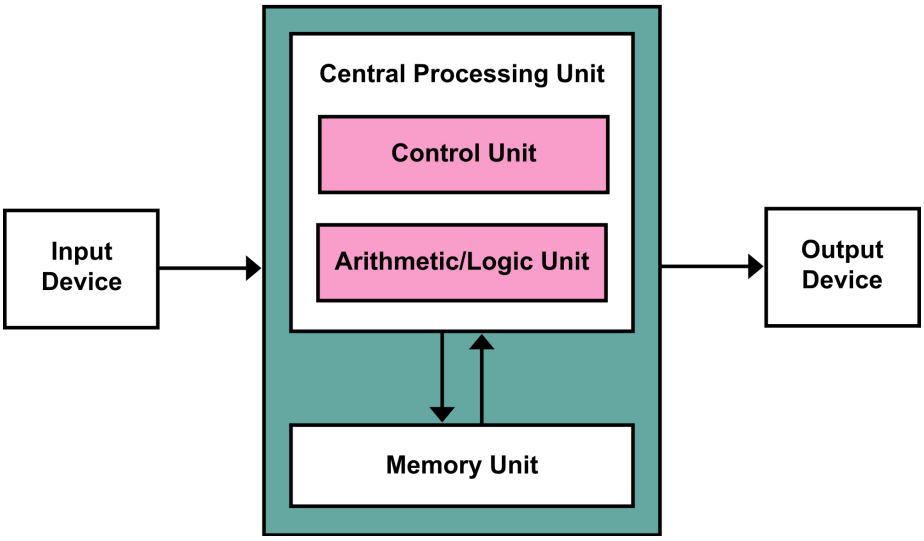
Another, maybe more useful model is the File Clerk Model of computers, as proposed by Richard Feynman.[2] In this model, a computer is just a reader, modifier and writer of data, like an old-fashioned filing clerk in a business. The difference between the filing clerk and a computer is the clerk's ability to make calculations like addition and multiplication directly and to read and understand instructions written in human language. A computer can do the same things, except that it's instructions and data have to first be translated into the machine language of ones and zeroes that is called binary. A single operation takes a computer many more steps than a filing clerk as it follows simpler instructions, which you need more of to reach the same result. But compared to the filing clerk, the computer is much faster. As opposed to the calculator model, this way of thinking emphasizes the data organization over the calculation process.

---

[1] cf. Stokes, John: Inside the Machine

[2] cf. Feynman, Richard: Computer Heuristics Lecture

3. Hardware: Basic Building Blocks of Computers



THE VON NEUMANN SCHEME FOR THINKING ABOUT COMPUTERS (FROM: HTTPS://COMMONS.WIKIMEDIA.ORG/WIKI/FILE:VON_NEUMANN_ARCHITECTURE.SVG)

The scheme that is most widely used to categorize a computer's building blocks is the one John von Neumann described already in 1945 in his „First Draft of a Report on the EDVAC". Therefore it is called the von-Neumann-architecture. It orders the parts of a computer into the categories of input, output, memory, arithmetic logic unit and control unit with the last two often summarized under the term processor.

3.1. Input/Output

I/O are all the system parts which empower human users to interact with the computer, input being the assemblies that allow to feed in data, output those that show the results in some way. Examples for input devices include mouse and keyboard, buttons, microphones, cameras, and devices that measure position and movement (gyroscopes, accelerometers, GPS). Displays, thought of first as output devices can also serve as input devices: for PMDs with touchscreens, they are even the main input device. Other examples for output assemblies include speakers and the vibration modules in some PMDs.

3.2. Processor

The Processor, in the von Neumann architecture, is something like the computer's most vital part. In the file clerk model, arguably, the processor is the file clerk itself, with the other computer parts being represented by the office the clerk works in. The

CPU includes the Control Unit, which is responsible for orchestrating the execution of instructions, especially organizing the data flow, loading and storing data.

The datapath on the other hand is itself comprised of at least two parts, namely the arithmetic logic unit (ALU) and the registers. The ALU is where the real calculations like addition and multiplication together with logical operations like negation (not) and conjunction (and) are done. The registers are a kind of small but very fast memory inside the CPU and thus physically close to the ALU, where the data is loaded that thereafter is fed into the ALU and operations are performed upon next. The results are again stored in registers, before they are written to main memory. Because the data doesn't need to be accessed directly from memory, the processor speed is to some extent decoupled from the memory speed.

3.3. Memory

Because registers can only store a small amount of data that is not nearly enough for the applications computers are developed for, there has to be another kind of larger storage. It is not possible to just make the registers larger, as obviously, there is only so much space inside the CPU. So main memory has to be physically farther away from the ALU and is already for that reason slower than the register storage. Also it needs to hold more data at a lower cost and therefore uses different storage technique which is also slower than the one used in registers: Main memory (also primary memory) is almost always some kind of RAM (random access memory as opposed to sequential access like on video tapes) and volatile, which means, that the stored data is lost, when power is switched off. Secondary storage though, which is still slower and farther away, is mostly non volatile and uses flash or hard discs.

3.4. Buses

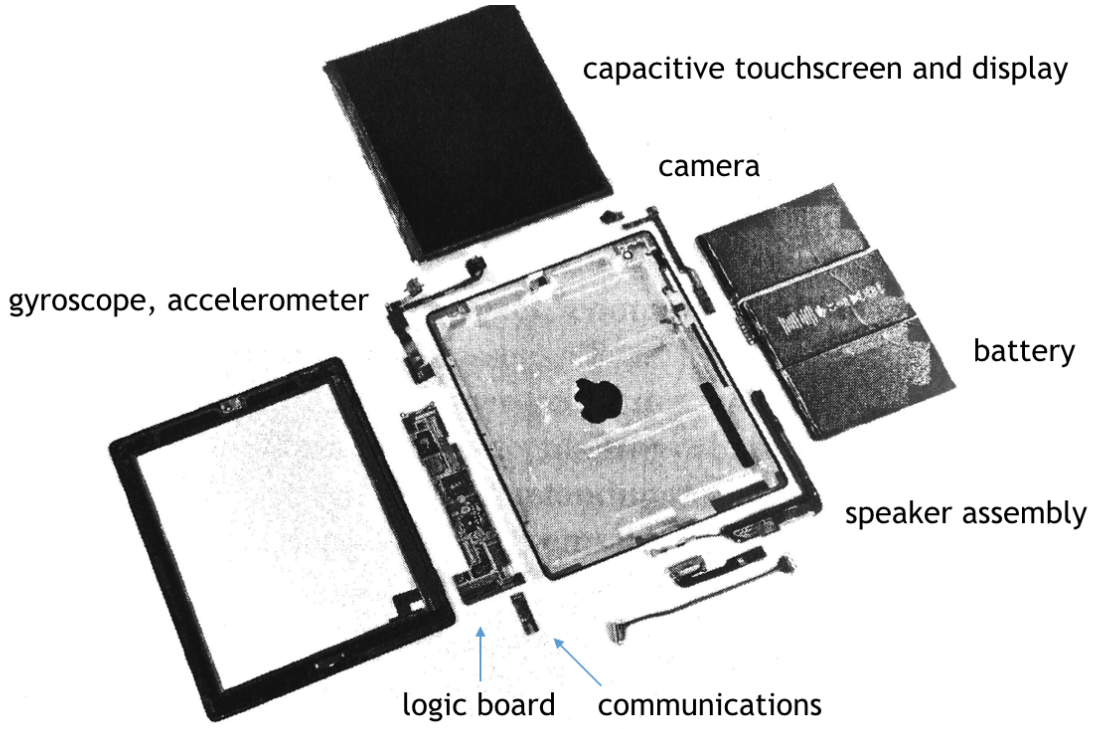
Buses are the building blocks of a computer, which transmit information between the different parts. In a first approach, one can imagine buses as something like a more sophisticated wire that transmits electrical signals. Depending on their responsibility, we distinguish between the data bus, which transmits the data that is operated on, the address bus, which transmits the place in memory the data is stored at, and the control bus, which orchestrates the shared usage of the bus system by multiple parts. Because the data flow can only go in one direction at every given time, buses were one of the first bottlenecks in computer systems, setting a limitation to computing speed.

3.5. Integrated Circuits

Almost all the inner parts of computers we have discussed so far were, on some level, made of transistors. So called integrated circuits, or chips of short, pack large amounts of transistors on a single wafer made from a semiconductor. This semiconductor, mostly silicon, can be made a electrical conductor, insulator or switch in a chemical process. Gordon Moore, one of the founders of Intel, suggested in 1965, that the transistor density in computer systems would double roughly every 18 to 24 months, which has held true for almost every period of time we look at since.

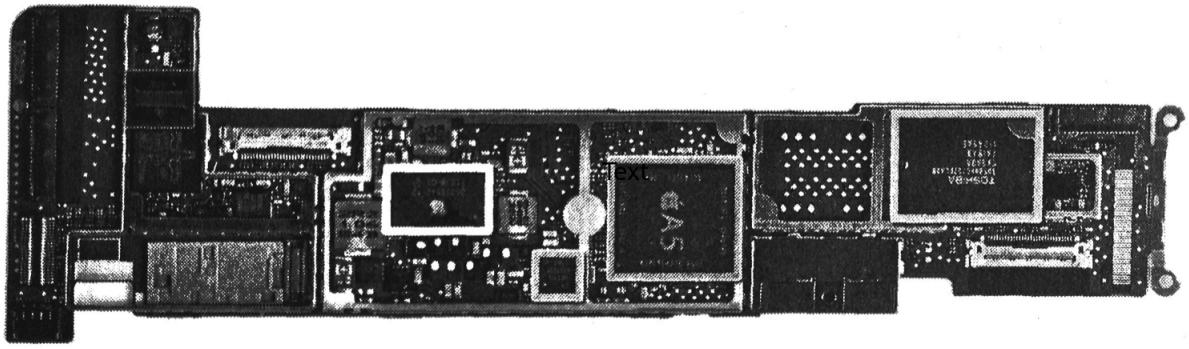
3.6. Example: Apple iPad 2

Three successively more detailed views of the Apple iPad giving an overview of hardware in real life:[3]



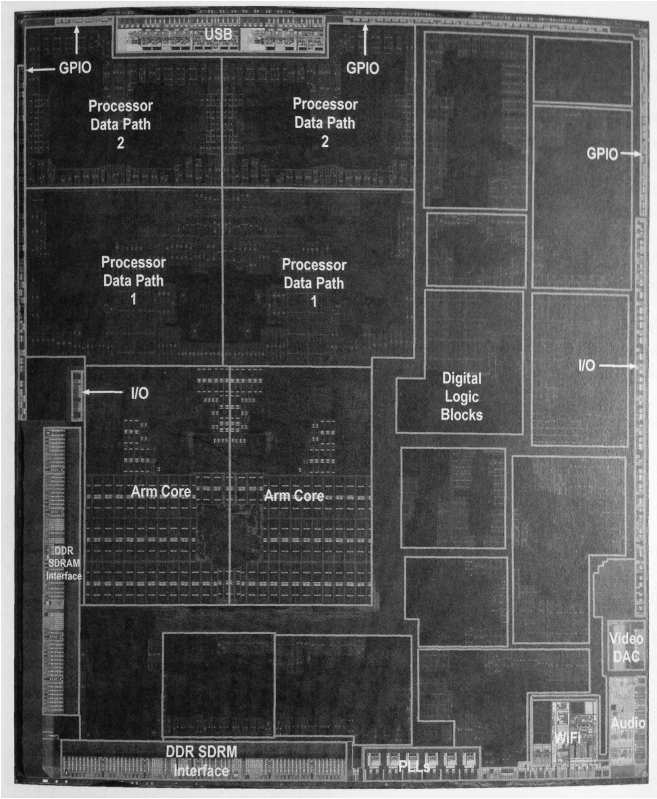
THE LARGER COMPONENTS OF THE IPAD 2: ALL OF THEM BUT THE BATTERY AND THE LOGIC BOARD BEING I/O DEVICES

[3] Hennessy, Patterson: Computer Organization and Design, The Hardware/Software Interface, 5th Edition, p. 20 f.



secondary storage processor and main memory

THE IPAD'S LOGIC BOARD WITH SECONDARY STORAGE AND THE A5 CHIP CONTAINING CPU AND MAIN MEMORY



THE PROCESSOR INTEGRATED CIRCUIT PART OF THE A5 CHIP

3.7. Computer Speed and Clock

The computer speed is highly dependent on the rate at which the parts change their respective states, because the computer always works in discrete steps. This rate is called clock cycle and given by the „clock generator“, a device that uses the well defined resonance frequency of a piezoelectric quartz crystal in a circuit to give out a signal of constant frequency to all other parts of the system. Because the speed of processors has grown significantly more than the speed of other building blocks, some parts of the hardware work at a whole number multiple of the clock cycle time. Typical clock rates of modern PCs and tablets are in the order of magnitude of 1 GHz.

4. Software: Fundamental Computing Concepts

4.1. Stored Program Computer

What we think of as a modern computer is a general purpose computer that allows for the usage of myriads of different applications without a change to the computer hardware. That is made possible by the storage of program instructions in the computer’s read-write-memory. This, together with the code being in the same binary machine language as the data the instructions are executed on, makes all those different applications possible, without having to input the instructions in real time manually with punchcards or switches.

4.2. Below the program: Program Hierarchy

When we look at the computer, we might ask ourselves, how the translation of a program written in a high-level programming language such as Java or C++ (a language we can understand) gets translated to simple instructions in binary (a language the computer can understand). Also, we might wonder, how a program written on and for one computer, can be interpreted correctly on a different architecture, in short, how a language can be „portable“.

As we will see, these questions are answered by the way software is organized. Basically, it is just multiple layers of software working on top of each other: On the surface is the application software the user interacts with. Underneath we find the systems software, consisting firstly of the compiler, which translates the programming language into the simpler instruction language. A high-level instruction is brought down to the basic arithmetic and logical operations the ALU can perform. The operating system handles the I/O devices, allocates storage and manages the shared use of the computer by multiple applications. The part of the system software that is right above the hardware level is called the assembler. An assembler translates the

instruction given out by the compiler in assembly language into binary, which in turn can be processed by the underlying hardware.

4.3. Booting

If we look at the way, programs are executed by the computer, we will notice that everything is going well and ordered when the computer has been running for a while: Memory is filled with the instructions and data and instructions about where to get more instructions from. But what happens when we start the computer? Main memory, as it is volatile, will not contain any instructions the CPU could execute. This problem is somewhat like pulling oneself up at one's bootstraps, why it is called booting. It is solved with a first instruction, the processor is hardwired to execute as soon as it is started, namely, to address a specific ROM (read-only-memory) storage containing the BIOS(-program). This program runs basic tests of system functionality and then directs to the bootloader(-program), which in turn loads the operating system. With the operating system up and running, the computer is ready for interaction with the user.

Bibliography

Stokes, John: Inside the Machine: An Illustrated Introduction to Microprocessors and Computer Architecture, 2006

Hennessy, Patterson: Computer Organization and Design, The Hardware/Software Interface, 5th Edition, 2014, Oxford

Feynman, Richard: Computer Heuristics Lecture, part of Idiosyncratic Thinking workshops, recorded on September 26th, 1985, accessed under: https://www.youtube.com/watch?v=EKWGGDXe5MA on August 16th, 2016

von Neumann, John: First Draft of a Report on the EDVAC, 1945, Princeton