PC Fundamentals for Design Professionals

An Online Continuing Education Course for Engineers

Course Number: PD-1001

Credit: 1 Hour / 1 PDH / 1 CPD

PC Fundamentals for Design Professionals
Jack H Warner Jr, PE

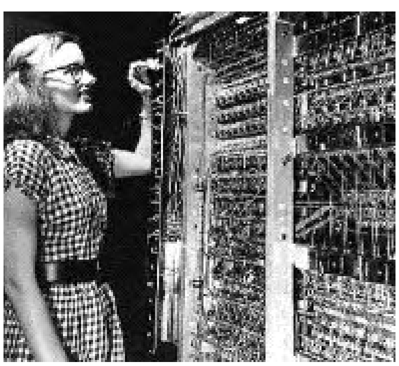
Introduction

As a design professional in the 21st century, your PC (Personal Computer) has become one of your most valuable tools. This was not true just a decade or two ago; what an incredible change! You depend on your computer for calculations, drawings, presentations, and correspondence. But how much do you know about how it works? Some people can be content to treat it like a "black box", not needing to know how it does what it does. But as a design professional that depends upon its operation and output for your success, wouldn't you like to understand a bit about its inner workings and how it came such a long way in such a short time?

I thought so… so let's get started!

History of the Personal Computer

IBM celebrated its 90th anniversary in 2001! That makes the PC industry, which has just been around since just 1981, sound like a relative infant. IBM was started in 1911 when Herman Hollerith's Tabulating Machine Co. merged with two other companies to become the Computing Tabulating Company, or CTR. In 1924 they changed their name to IBM. IBM officially stands for International Business Machines, but many other possibilities have been proposed, such as "I've Been Moved", referring to the fact that employees were frequently transferred to another office.



The first machines that would be really recognized as electronic computers were created in the 1940's. These behemoths were as big as a small house. They used punched cards and paper tapes, had lots of blinking lights and switches, were full of relays and/or vacuum tubes, and took several engineers to run. They were used mainly in university and government work. Computers first became available for commercial use in 1951, but you had to have about $1,000,000 to spare

Computers have continued ever since to get faster, more powerful, easier to use, and less expensive. As they became more powerful, more durable, and less expensive, they became common in every large business, and programs were written to handle many tasks.

In the 1970's, simple computer components became inexpensive enough to allow hobbyists to create or purchase small computers. The early models were very difficult to use, requiring input and output via switches and lights, but gradually features we take for granted were added, such as keyboards and computer screens (though these were often your TV!).

Some models became popular enough that you might have heard of them: the Commodore PET, the Apple II, and the Radio Shack TRS-80. These machines had keyboards, screens, floppy disks, tape storage, and user's manuals.

But the turning point, the real beginning of the PC boom, was the introduction of the IBM PC (Personal Computer) in 1981. Several factors led to this machine's huge success. First, it was made by IBM. Everyone knew who IBM was, and they were a large, long established company. This resulted in the general feeling that this was not simply a hobbyist machine, but a *business* machine. Second, the IBM PC was created primarily from hardware and software from other vendors (CPU from Intel, Operating System from Microsoft). This helped speed development, but also meant that the "architecture" of this machine was not completely proprietary. Other companies could create products to add to this machine. The products and software that rapidly came to market to enhance the PC increased the usefulness and appeal of this machine.

Ever since, IBM has continued to improve its PC's, as have many makers of "clones". The clones were possible because others besides IBM could purchase the same components, and create computers with a similar architecture. This encouraged innovation and competitive prices. Personal Computers have pretty much doubled in power every 18 months since their inception, and will probably continue to do so.

As they became useful, they found their way into the offices of design and construction professionals, where they increased productivity in many ways. They began to be used for construction drawings (CAD), engineering calculations, project scheduling, financial planning, and numerous other tasks.

Now most businesses would be completely dead-in-the-water without their PC's. They've made that much of a difference in most businesses in just a few decades.

Basic Computer Components

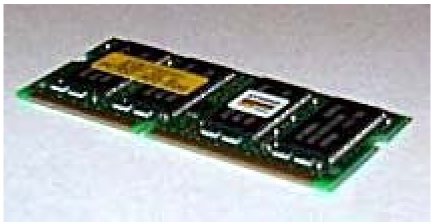
Throughout this continual evolution that the computer has undergone, its basic underpinnings have remained constant. They are composed of the same basic components… it's just that those components have shrunk by leaps and bounds, and have evolved in their own ways. Right now we'll describe these basic building blocks that make up the computer.

Central Processing Unit:

The central processing unit (CPU for short) is the workhorse of the computer. This is where each and every instruction in a computer program gets executed. It contains millions of microscopic transistors connected into circuitry that execute program logic, do integer mathematics, and even do floating-point mathematics. I mention floating-point mathematics separately because in the early days of PC's, you had to purchase an optional math co-processor if you wanted the computer to handle floating-point math with any competence.

In the days of the Intel 386 processor, this could be a $1000 option! Today, all PC CPU microprocessors have floating-point math circuitry built in.

Memory:



Memory is where the computer keeps currently running programs and data. Older computers used all sorts of fancy contraptions for memory, but today it is always millions of transistors piled into chips. There are several types of memory inside a PC, but the primary type is Random Access Memory (RAM). RAM "loses its memory" when it loses power, so it is only useful for holding data while the computer is on. For holding programs and data longer-term (power on OR off), you need some sort of storage.

Storage:

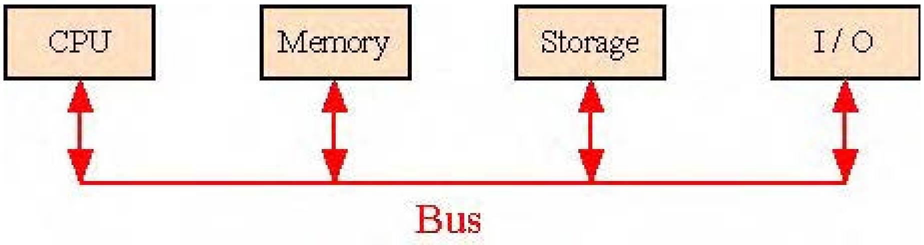
A storage device is one of many types of components that can store programs and data with no power required. Floppy disks, hard drives, CD/DVD drives, and magnetic tape are the most common storage devices in use today. Some computers even have ALL of these devices attached. Which device is used depends upon the quantity of data, the speed needed to get to the data, and whether it needs to be removed from the computer for sharing or storage.

Input and Output:

This is a very broad category, and defines how we interface with the computer. Input and Output devices (I/O for short) include your keyboard and your monitor (computer screen). I/O also includes mice, sound capabilities, scanners, printers… anything that gets data to or from the user.

Bus:

The computer's bus is the "wiring" that connects all of the other components together, and handles the communication between them.



Data travels to and from all the computer's components via this "data highway". Though the bus seldom gets much attention when shopping for computers, it has a tremendous impact on the performance of the computer. Just like CPU's, bus designs have become more complex, and capable of operating at higher speeds.

Software (BIOS and OS)

We're not going to delve into software in this lesson, except to point out that without at least some rudimentary software, these components won't even boot up. This is the software that I'll mention briefly. Even before you start your programs for CAD, or word processing, or whatever, two basic pieces of software have already done their work.

The first piece of software is called BIOS, which stands for Basic Input Output System. This bit of software is usually stored in a "flash" memory chip in the computer, which is a unique type of memory that can retain its data even when the power is off. This software controls the initial boot-up process. It runs a test on the system, called the Power-On Self Test (POST). It also contains low-level code to allow the I/O devices (especially the keyboard and screen) to work, as well as the hard drive.

Once this code has been loaded, the BIOS can then load the Operating System (OS). The operating system is on the hard drive; therefore it could not be loaded until the BIOS was in place and the hard drive was accessible. The operating system is responsible for then getting the rest of the devices working (such as the mouse). It then manages the computer's memory, CPU usage, and software applications. It also provides the interface that the user sees when the computer is done booting up, as well as defines the interface of any programs that are then run. Windows is by far the most popular PC operating system, but there are also some PC's running other OS's such as Linux or DOS.

Bits and Bytes

This topic is going to be pretty fundamental, but it will help lay the groundwork for the next couple of sections. Now we'll start discussing the key to how these components work, and how they talk to each other.

If anyone ever told you a computer was smart, they were full of beans; the truth is, they are not very good at counting higher than 1! At the heart of every digital circuit, each tiny transistor can only remember one of two things... "on" or "off". That's great for desk lamps, but our computer needs require a little more... we need to be able to count higher than "one". So, electrical engineers combine many of these transistors to handle more complicated data. They combine them, and treat the "on"s and "off"s like "1"s and "0"s in binary math. And what is binary math? Well, what most of us are used to is "base-10" math. That's where when you get past 9, you have to go to double digits (carry to the next decimal place). Binary math is "base-2" math, so when you go past 1, you have to go to double digits. That would be pretty tedious for humans to use, but it works perfectly in electrical circuits where there is only "on" and "off". Here is how you count in binary:

Base-10	Binary	
0	**0**	
1	**1**	(so far so good)
2	**10**	(OK, now it's starting to get weird… there is no "2", so we carry)
3	**11**	
4	**100**	
5	**101**	
6	**110**	
7	**111**	

8	**1000**
9	**1001**
10	**1010**
11	**1011**
12	**1100**

Don't worry if you are not immediately doing binary multiplication problems in your head... only computers, programmers and technicians really have to know this. But if you see that every time you need to go above 1, you have to carry over into the next column, and that the columns appear to be powers of 2 (1,2,4,8,etc) rather than powers of 10 (1,10,100,1000, etc), then you're getting the idea.

Guess what computer people call these **B**inary dig**ITS**? You guessed it, **BITS**! A bit is simply a "1" or a "0", and if you get a whole lot of these bits together, you can make really useful numbers. When I say a whole lot, I think I can show you that I really mean it. As you can see above, it took 4 bits to describe the number "8". It takes 6 bits (100111) to describe the age at which I stopped counting (39). It takes 9 bits (111000010) to describe the number of cents you might have to pay for bottle of water at a fine hotel (450... who would pay that?).

Since one bit all by itself is pretty useless, it seems only natural that we would group them together most of the time, and perhaps give this common grouping a name. That's where we got the name **Byte**. A Byte is 8 bits. Why 8? Because some of the early usable processors were designed to handle 8 bits at a time. With 8 bits you can count to 255. That is a useful number. The earlier 4 bit processors could only count to 15, so 8 bits was a huge leap in usability. We now have 32 and 64 bit processors, but the grouping of 8 bits per Byte has stuck.

So, if you've got a number of bits and want to know how many Bytes that is, you divide by 8. If you've got a number of Bytes and want to know how many bits that is, you multiply by 8.

Bits and Bytes often are used to express data transfer speeds as well. An analog modem might transmit data at 56 kbits/sec. That's 56,000 bits per second. A 10baseT LAN connection might transfer data at 10 Mbits/sec. That's 10,000,000 bits per second (or 1,250,000 Bytes per second).

Before we leave this topic, I can't help but share a "geek" joke:

"There are only 10 kinds of people in this world... those who understand binary, and those who don't!"

If you understand this one, then you've been paying attention in this section. If not, go to the table above and review what the binary number "10" represents in normal base-10 numbers.

A Word on Binary Precision

How a program handles floating-point numbers is important to design professionals. Every computer program that deals with floating-point calculations (such as CAD, or engineering programs) has to store and calculate with its data in binary form. Most programs store floating-point data as either 4 bytes (single precision), or 8 bytes (double precision).

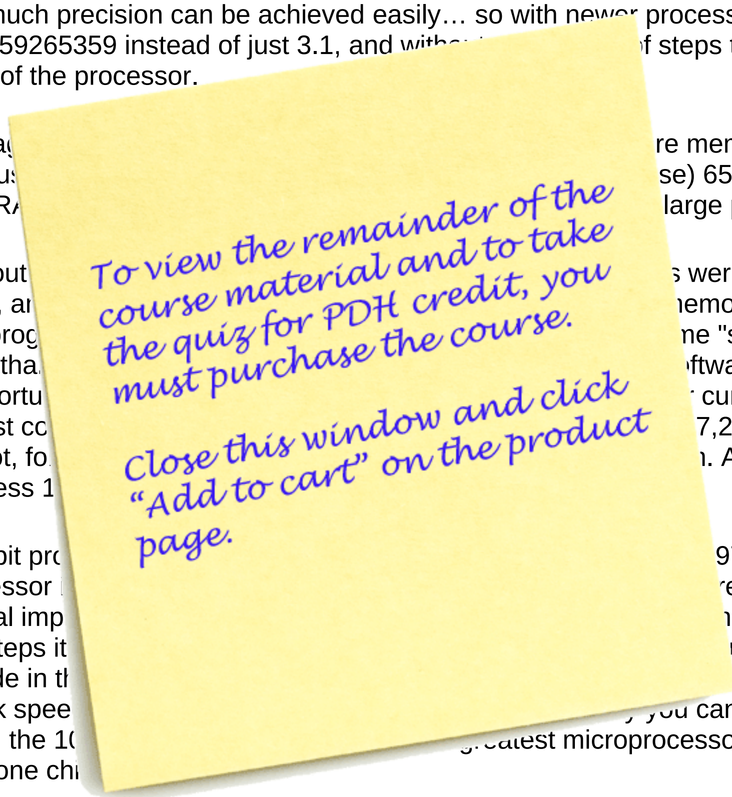
The advantage to storing numbers as 4 bytes is that they can take up less room in memory and on disk. Data stored in this manner results in 6-7 significant digits (base 10), so it's close enough for doing simple calculations. These numbers are a bit limited in their range as well. The smallest number that can be represented is about $1x10^{-38}$, and the largest number is about $3x10^{38}$.

Storing numbers as 8 bytes is generally preferable for precise calculations. This results in 15-16 significant digits, and calculations are just as fast on today's PC's as they are with single precision. This precision is really noticeable when doing iterative calculations, such as finite element analysis. Double precision can handle numbers as small as about 2×10^{-308}, and numbers as large as about 2×10^{308}.

Bits and Processors

Now we'll talk about microprocessors. One reason this topic is interesting right now is that Intel and AMD have recently begun shipping computers with 64-bit processors. So, let's start with "what does 64-bit mean?" Remember that a "bit" is a Binary digIT. It can be either a "0" or a "1". A transistor or wire in a digital circuit either has an electrical charge applied ("1"), or no charge applied ("0"). Also remember that it takes quite a few bits to make useful numbers. Here are the two main reasons that more bits are better when describing processors.

The first microprocessor, the Intel 4004, was a 4-bit wonder (this was in 1971). If you could limit your calculations to numbers less than 16 (which would make them 4 bits or less), you could calculate very efficiently with a 4-bit processor. If you are working with lots of numbers greater than 15, however, having a processor with a wider data path than 4 bits is very helpful. With these larger numbers, they could only be sent to and worked on by the processor in 4-bit chunks. Processors with more bits (in this context it would mean a processor with 8, 16, 32, or 64 bits) can work with larger numbers more easily, This advantage also applies to handling floating-point numbers and how much precision can be achieved easily… so with newer processors we can calculate with 3.14159265359 instead of just 3.1, and with
f steps to get the
numbers in and out of the processor.



The second advanta
re memory. A 16-bit
processor, such as u
se) 65,536 bytes
(64kB) of memory (R
large programs.

The IBM PC came out
s were being
shipped with 128kB, a
emory. Through
some complicated prog
ne "switcheroo"
games to use more tha
ftware capabilities
and performance. Fortu
current 32-bit
processors (the most c
7,296 bytes (4
GB). Believe it or not, fo
. A 64-bit
processor can address 1

Intel released an 8-bit pr
978, called the
8086, a 32-bit processor
e is, of course,
more to the continual imp
dle. The clock
speed (how many steps it
mprovements
have also been made in t
IBM PC, by
the way, had a clock spee
you can buy machines
with clock speeds in the 1
atest microprocessor chips even
bundle 2 CPU's on one ch

Strengths of the Computer