# Contents

# Chapter 1
# The General Purpose Machine

**1.1** The PowerPC 601 processor addresses a maximum of $2^{32}$ bytes of memory. What is the maximum number of 64-bit words that can be stored in this memory? **(§1.1)**

**Solution:** A 64-bit word = 8 bytes =$2^3$ bytes. $2^{32}$ bytes of memory is equivalent to $2^{29}$ 64-bit words. In decimal this is about a half billion words.

**1.2** A certain IBM 970 processor has a system clock frequency of 1.2 GHz, what is the clock period? **(§1.1)**

**Solution:** 1.2GHz is $1.2 \times 10^9$ clock cycles per second, so the period is $8.3 \times 10^{-10}$s, or 830 ps.

**1.3** If the cost of RAM is $95 for a 4MB module, what will it cost to install 32 M words in an originally empty machine if the word size is 32 bits? **(§1.1)**

**Solution:** With a word size of 32 bits, one word = 4 bytes. So 32MW = 32M × 4B = 128MB. At $95 for 4MB, this gives 128MB/memory × $95/4MB = $3,040/memory.

**1.4** How many 500MB tapes will be required to back up a 120GB hard drive? How long will the backup process require if one tape can be filled in 5 minutes? (No coffee breaks allowed.) **(§1.1)**

**Solution:** There are $120 \times 2^{30}$ B/disk and $500 \times 2^{20}$ B/tape. Thus one needs about $120 \times 2^{30}/500 \times 2^{20}$ tapes/disk, or 240 tapes. 240 tapes × 5minutes/tape = 1200 minutes = 20 hours.

**1.5** a. A certain machine requires 1.5 μs to process each 64-byte data record in a database. How long will it take to process a database containing $100 \times 10^8$ records?

b. How many 700MB-capacity CD-ROMs will be required to store the database? **(§1.1)**

**Solution:** a. 1.5 μs/record × $100 \times 10^8$records/database = 15000 seconds/database = 250 minutes.

b. If a record takes 16 bytes, then there are $16 \times 10^{10}$bytes/database. At $700 \times 10^6$ B/CD-ROM, this means about 229 CD-ROMs/database. Unreasonable, even for small records!

**1.6** a. What is the percentage relative error in using $2^{10}$ as an approximation for $10^3$?

b. What is the percentage relative error in using $2^{30}$ as an approximation for $10^9$?

c. What is the general formula for the relative error in using $2^{10k}$ as an approximation for

$10^{3k}$? **(§1.1)**

**Solution:** a. The relative error in approximating 1000 is $\frac{1024-1000}{1000} = 2.4\%$.

b. Relative error in using $2^{30}$ for $10^9$ is $\frac{2^{30}-10^9}{10^9} = 7.37\%$ .

c. $\frac{2^{10k}-10^{3k}}{10^{3k}}$ is an answer, but it is uninteresting. Strictly dullsville! The interesting thing that can be seen from parts a) and b) is that the relative error increases. How can we get a formula that helps us understand when the value of $k$ is too large for the approximation to be any good. One way is as follows:
Relative error is

$\frac{2^{10k}}{10^{3k}} - 1$, and since $\ln(\frac{2^{10k}}{10^{3k}}) = k(10\ln 2 - 3\ln 10) = 0.0237k$, the relative error is $e^{0.0237k} - 1$. The first two terms of a power series expansion give an idea of the behavior: relative error $\approx 0.0237k + 0.0003k^2$. From this we see that if $k > 20$, the relative error exceeds 50%—not a good approximation.

**1.7** If one printed character can be stored in 1 byte, approximately how many bytes will be required to store the text of this textbook? Do not include the graphics, and do not count the characters one by one. Show your work and state your assumptions. **(§1.1)**

**Solution:** In manuscript form there are about 96 characters (including spaces and punctuation) in a full line of text. About 3/4 of a page is full lines on the average, and there are 570 pages in the textbook. A full page could contain 50 lines.

50 lines/page × 570 pages/book × 75% × 96 char/line ≈ 2,052,000 char ≈ 1.95 MB.

**1.8** Consider computing the electric field in a box 1.5 cm on a side. The spatial resolution in each dimension is to be 50 μm. Assume that it takes 150 instructions for every point in the 3-D grid to do the calculation. How long does the computation take on a computer that can execute at a rate of 100 MFOPS (millions of floating point instructions per second)? **(§1.1)**

**Solution:** Each side of the box is $1.5 \times 10^{-2}$ m long. The spatial resolution in each dimension is $50 \times 10^{-6}$m. Dividing the length of the box by the spatial resolution gives 300 points/side. Since the box is 3-dimensional, $300^3 = 27 \times 10^6$ points must be calculated. Each point needs 150 instructions, so $27 \times 10^6$ points/3-D grid × 150 instructions/point = $4.05 \times 10^9$ instructions/3-D grid. The processor can execute 100 MFOPS, so dividing $4.05 \times 10^9$ instructions/3-D grid by $100 \times 10^6$ instructions/s = 40.5 s.

**1.9** Describe the tools used by the assembly language programmer. **(§1.3)**

**Solution:** The tools used by the assembly language programmer are editor, assembler, linker, loader, debugger, and development system. The *editor* is used to edit the source code (the assembly language). The *assembler* allows the programmer to generate machine language program from assembly language programs. It translates assembly language statements to their binary equivalents. The *linker* links separately assembled modules together into a single module suitable for loading and execution. The *loader* loads the executable binary code into the memory and changes some logical addresses to appropriate physical addresses. The *debugger* allows the programmer to observe the details of program execution. The *development system* is a collection of hardware and software that is used to support system development.

**1.10** Describe the differences between assembly language and high-level languages as regards type checking. What is the source of these differences? **(§1.3)**

**Solution:** High-level languages usually have several primitive data types that are part of the language definition. In addition, most high-level languages provide some constructs to let the user define complex types by composition of the primitive types of the language. Correct type usage is usually enforced by the compiler during syntactic and semantic check phases. The rich data type structure built into higher level programming languages is missing from most assembly and machine languages. It's all "bits-n-bytes" at the machine level.

Assembly language programmers need to specify the addresses where the program and data should be located and to focus on the machine level implementation. High-level language programmers want the compiler to check the use of language constructs and to check some semantics of programs. High-level language programmers also need short representations of objects that are of interest to them, such as integers, reals, and characters.

**1.11** What is the difference between a programmable calculator and a personal computer? **(§1.4)**

**Solution:** The programming capability of a programmable caculator is limited, and there's only one "language." It can only solve some particular kinds of numerical problems. The I/O capability is very poor.

A personal computer is a general purpose machine. Nearly all kinds of operating systems, programming languages, and application software can run on it. It can solve nearly all kinds of problems, such as mathematical computation, controlling, design, information processing, and artificial intelligence problems. It can drive all sorts of I/O devices—vastly more than a calculator. Different computers can also be connected via a network.

**1.12** How would computers be operated if there were no stored programs? **(§1.4)**

**Solution:** If there were no stored programs, computers could only be operated by button pushing, switching, and rewiring. Furthermore, the operator would have to wait for the result of a step before he or she could enter the next step. It would be the operator's responsibility to determine the next step.

**1.13** What is an ISA, and what are its components? **(§1.3)**

**Solution:** The ISA is the collection of instructions and resources. It includes the instruction set, the machine's memory, and all of the programmer-accessible registers in the CPU and elsewhere in the machine.

**1.14** Using only the instructions in Table 1.3, compile by hand the following C statements into VAX11 assembly language. Assume all variables are integers. **(§1.3)**

a. `V = (W + X)*(Y + Z);`

b. `A = B*C*D + E;`

c. `z = x*y^2;`

d. `U = V; W = U + Y;`

**Solution:** T is a temporary memory location introduced to avoid overwriting operands.

a.
```
ADD W, X, T
ADD Y, Z, V
MPY T, V, V
```

b.
```
MPY B, C, T
MPY D, T, A
ADD E, A, A
```

c.
```
MPY y, y, z
MPY x, z, z
```

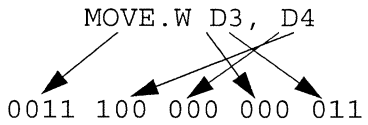
d.
```
MOV V, U
ADD U, Y, W
```

**1.15** Using only the information in Table 1.2, encode the following MC68000 assembly language instructions into machine language. Express your results in binary. **(§1.3)**

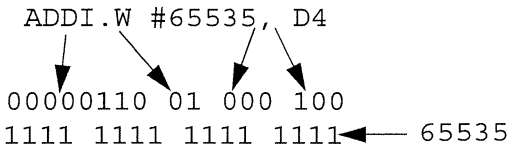
a. `MOVE.W D3, D4`

b. `ADDI.W #65535, D4`

**Solution:** a.




b.




**1.16** Describe the advantages of data typing in high-level language programming. **(§1.3)**

**Solution:** Data typing helps prevent the programmer from making errors due to misuse of language constructs, and also provides guidance to the compiler about the meaning of the program. High-level language data types are designed for ease of representing objects

that are of interest to the programmer.

**1.17** If assembly language is mostly free of data typing, how are data types expressed in assembly languages? **(§1.3)**

**Solution:** They are solely determined by the intent of the programmer, who performs "data typing" by the kind of instruction used.

**1.18** Define the difference between a simulator and an emulator. Which would you prefer to work with, and why? **(§1.4)**

**Solution:** Simulators are software tools that mimic aspects of the system's behavior and allow a certain amount of performance estimation at an early part of the design process. Since simulators mimic hardware performance in software, they are usually slower in operation by orders of magnitude. Emulators can be thought of as hardware-assisted simulators that provide operation speed closer to the speed of the hardware being emulated.

Emulators are preferred because they provide results closer to the behavior of the real machine being emulated.

**1.19** a. Define the term *bus*.

b. Why are buses used in computers?

c. Describe the similarities and differences between the computer bus and the electric power grid.

d. Describe the differences between the computer bus and the water system in your home. **(§1.4)**

**Solution:** a. A bus is a multiplexer interconnecting multiple devices and allowing multiple devices to use it for communication by time sharing. It provides both a data path and a signaling or control path.

b. It can save hardware and thereby reduce the cost. Using buses makes the design relatively simple, and it is not difficult to add more devices to the system.

c. They use a common medium. The electric power grid only delivers power. The computer bus transfers data and control signals. Once the connection is established, the power can flow to all terminals simultaneously. Different devices use the computer bus by time-sharing.

d. They use a common medium. The water system only delivers water. The computer bus transfers data and control signals. Once the connection is established, the water can flow to all terminals simultaneously. Different devices use the computer bus by time-sharing.

**1.20** Provide a diagram similar to Figure 1.5 for the computer you work at most frequently. **(§1.5)**

**Solution:** The diagram should include CPU, memory, I/O, and their interconnection. The name of each bus should be specified. I/O devices may include keyboard, mouse, bit pad,

display, printer, disk drive, CD ROM, tape, network, other computers, and so on.

**1.21** How does the computer architect communicate machine specifications to the logic designer? **(§1.5)**

**Solution:** Natural languages and some informal descriptions are inevitable. They provide a general view of the architecture and a more intuitive "feel" for the meaning of a construct. But they can be confusing, imprecise, or incomplete in their descriptive power. Formal description languages are often used to augment natural language in this communications process. These languages are called register transfer languages, as they are specifically designed to describe information transfer between registers, a feature that is central to the operation of the computer. Formal descriptions provide the means to be precise and exact.

**1.22** What natural separation distinguishes computer logic design from classical logic design? **(§1.5)**

**Solution:** The computer designer does not design an entire digital computer using state machine design techniques. There is a natural separation or partitioning of concerns between the data path and the control path. Whereas the logic designer sees logic gates and flip-flops, the computer designer sees multiplexers, decoders, and register files.

**1.23** Estimate the costs of the components in Figure 1.6 for the computer you work at most frequently. State where you got the component costs. **(§1.5)**

**Solution:** From *PC Magazine,* www.pcmag.com.

| **Component** | **Price** |
|---|---|
| CPU: Intel Pentium 4 motherboard | $120 |
| Cache: IBM 256 KB (94G3141) Cache Memory | $39 |
| Main memory: 512MB 400MHz DDR PC3200 DIMM | $80 |
| Disk memory: Western Digital 120GB (Ultra ATA/100, 7200 RPM) | $92 |
| Tape memory: Exabyte 80/160 GB VXA-2 Tape Drive | $528 |

**1.24** Describe as accurately as you can the implementation domain of the computer proposed by Charles Babbage. **(§1.5, 1.6)**

**Solution:** Gears, shafts, wheels, cranks, and dials.

**1.25** Describe as accurately as you can the implementation domains of the first through the fourth generation of computers. **(§1.5, 1.6)**

**Solution:** The first generation: vacuum tubes, relays, and mercury delay lines. The second generation: discrete transistors. The third generation: small- and medium-scale integrated circuits, TTL chips. The fourth generation: VLSI on silicon, TTL chips, ECL chips, PLAs, and sea-of-gates gate arrays.