Exercises in this document are supposed to be done at the lab. You do not need to send us their solutions, we will not grade them anyway.

1 Exam-like questions

• CPU frequencies have stopped increasing around year 2005. Why did this happen? Does it contradict with Moore’s law?

  Answer: It happened because we have bumped into physical constraints that prevent us from reaching higher frequencies. More specifically, there are two such constraints: First, CPU power consumption grows roughly as the square of its frequency, so for high frequencies we have a heat dissemination problem. Second, high frequencies require very small delays of the transistors, and mother nature is setting a hard boundary there.

  This does not contradict Moore’s law, which concerns the number of transistors per chip, not the frequency. E.g., although the frequency remains the same, we still keep packing more transistors per chip, allowing us to make multicore chips, with more cache, etc. Moore’s law is expected to remain valid for the next 10-15 years at least.

• Explain the difference between translation and interpretation. Which computer organization levels are typically translated, and which ones are typically interpreted?

  Answer: Translation (a.k.a. compilation) transforms a program in language $L_n$ into a lower-level language $L_{n-1}$. Once translation has been realized the lower-level program can be executed any number of times without needing any more translation.

  Interpretation replaces each instruction from language $L_n$ into an equivalent sequence of instructions in language $L_{n-1}$ at runtime, that is, as the $L_n$ program is being executed.

  Examples of translated (compiled) languages: Assembly (translated by an assembler program into machine bytecode), Java (translated by the “javac” compiler into JVM bytecode), C (translated by the “cc” or “gcc” compiler into machine bytecode).

  Examples of interpreted languages: Java bytecode (interpreted by the “java” program into machine bytecode), Python (interpreted by the “python” program into machine bytecode), Machine bytecode (interpreted by the CPU into microprogram instructions).
• Explain the difference between an early computer such as ENIAC and Von Neumann’s machine.

Answer: The ENIAC computer could only run a single program because the program was hardcoded in the physical circuits. Von Neumann’s design states that user programs should reside in the computer’s memory, and the computer should be generic enough to be able to execute such programs.

• Early computers used a single bus to connect all devices together. Why was this a bad idea?

Answer: Different devices communicate at different speeds. It is more efficient to have very fast buses for very fast devices (e.g., memory) and slower buses for slower devices (e.g., hard drives). This also allows to support old legacy buses (e.g., ISA) in new computers.

• Typical computers usually have several different buses. For example, in Figure 1, the USB2 bus is connected onto the PCI bus. These two buses use completely different protocols and message formats. How do you think the two buses are connected to each other so that the CPU can communicate with its USB-connected keyboard?

Answer: The two buses are connected thanks to the USB controller. This controller acts as a normal device with respect to the PCI bus, and as the bus controller from the USB bus. It translates messages from one bus protocol to the other.

• The following word has been encoded using the Hamming code. One of these bits is wrong. Which one? (the leftmost bit is \(d_1\) and the rightmost is \(d_{14}\))

\[
11011101010100
\]
Answer: Bit number 10 is wrong. The correct word is therefore:

\[1101110100100\]

2 Hamming codes

For these exercises we will use Hamming codes to detect and correct errors in 16-bit words.

Question 1: For each parity bit (1, 2, 4, 8, 16): write down the list of bits that are verified by this parity bit.

Question 2: Let us consider the following bit string, which is encoded by the Hamming code: 0010 0010 1001 0011 01101. Is there any corrupted bit in this sequence? If yes, which is the corrupted bit?

3 Moore’s law in practice

Question 1: The Intel 8008 processor, produced in 1972, had 3500 transistors. Give an estimate of the number of transistors that a processor produced in 2006 should have, by using Moore’s law. Use the formulation of Moore’s law which states that the number of transistors in an integrated circuit doubles every two years.

Question 2: Compare your estimation with the following processors produced around the year 2006:
Intel Core 2 Duo - 291,000,000 transistors
Cell - 241,000,000 transistors
Dual Core Itanium 2 - 1,700,000,000 transistors
Are these processors above or below the curve of Moore’s law?

Question 3: Cray-1, one of the world’s first supercomputers, was produced around the year 1975 and contained approximately 200,000 logical gates (i.e., approximately 400,000 transistors). How many Cray-1 supercomputers do you think could be built with the number of transistors from your home computer’s CPU?
4 Monitor a computer

In this exercise you will learn a few commands that can be used to monitor a Linux computer. For this, you will have to log into one of the department’s Linux servers: ssh.data.vu.nl, pcoms001.vu.nl or pcoms002.vu.nl. Your username and password are your VUnet-ID and your VU password. You can do this in two ways:

- for terminal access, you can use PuTTY; you will find it among the applications installed on your computer;
- for GUI access, you can use CygwinX; after logging in, you can open a command-line terminal by choosing Applications > Accessories > Terminal.

We recommend PuTTY as it is much faster.

In Linux, the /proc directory contains general information about the system’s hardware, and also real-time information about the resources’ utilization and about the processes that are currently running.

Question 1: Look at the CPU and memory information using the following commands:

```
less /proc/cpuinfo
less /proc/meminfo
```

The top command allows you to see real-time information about the processes running on your computer and about the utilization of its resources. The simplest way to use this command is without any argument:

```
top
```

Question 2: Use the top command to see the processes that are currently consuming most of the CPU on the machine. If you want to have more time to look at the output, you can specify how often it is refreshed (the default is 5 seconds):

```
top -d 20
```

Question 3: With the top command you can also monitor an individual process. Let’s try to monitor the memory consumption of a simple Java program. Download the class MemLeak.java from the course’s site, and copy it into your home directory. Here is the source code of the class:

```java
import java.util.ArrayList;

public class MemLeak
{
    private static ArrayList<byte[]> bigArray =
        new ArrayList<byte[]>();

    public static void main (String args[])
    {
        for (int i=1; ; i++)
        {
            byte[] bytes = new byte[5000];
            bigArray.add(bytes);
        }
    }
}
```
System.out.println("Step " + i);

try {
    Thread.currentThread().sleep(50);
} catch (InterruptedException ie) {
    ie.printStackTrace();
}
}

Compile (javac MemLeak.java) and run (java MemLeak) this class. After starting it, you need to find the process identifier (PID) that the operating system assigned to this process. You can do this by issuing the following command:

```
ps -f -u <your_username_here>
```

In the output of this command, look for your Java process and see the number from the PID column. This is the process identifier. Then, you can use the top command in the following way:

```
top -p <the_pid_of_the_java_process>
```

Look at the RES column (this is the amount of physical memory that the program is using). This value is constantly increasing, because our program has a memory leak.

**Background information**

There are many tools that allow for a more detailed monitoring of the memory consumption in Java programs, and can be used to debug memory leaks. On the other hand, commands like top are easier to use and can help you to quickly detect when such problems occur.

— the end —