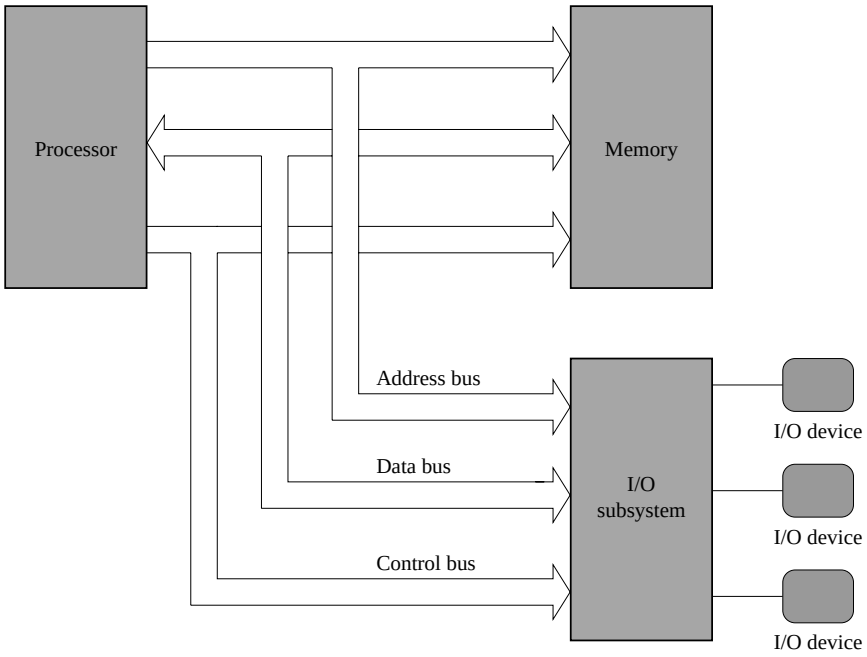
COMP 2003 Computer Organization

Assignment 1

September 28, 2005

1. The system bus consists of three major components: an address bus, a data bus, and a control bus (see the figure below).



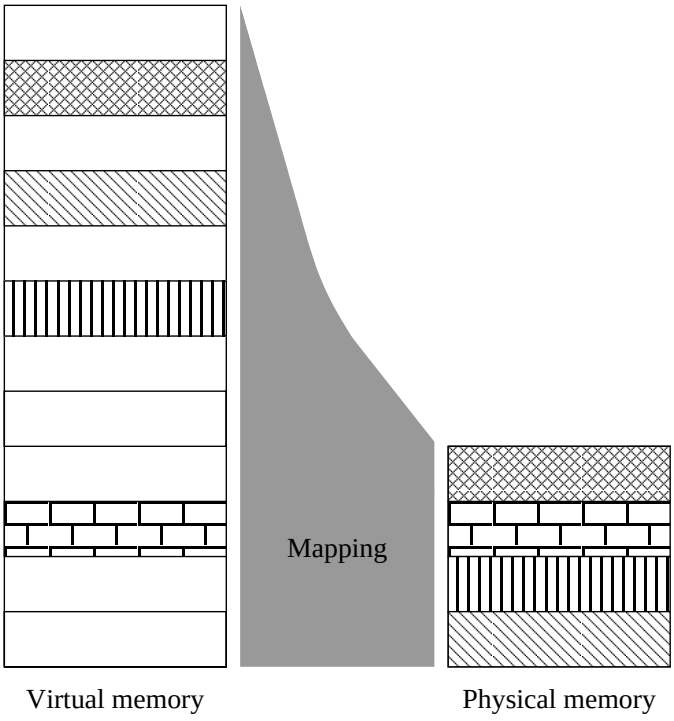
The address bus width determines the amount of physical memory addressable by the processor. The data bus width indicates the size of the data transferred between the processor and memory or an I/O device. For example, the Pentium processor has 32 address lines and 64 data lines. Thus, the Pentium can address up to 2^{32}, or 4 GB of memory. Furthermore, each data transfer can move 64 bits of data. The Intel Itanium processor uses address and data buses that are twice the size of the Pentium buses (i.e., 64-bit address bus and 128-bit data bus). The Itanium, therefore, can address up to 2^{64} bytes of memory and each data transfer can move 128 bits.

The control bus consists of a set of control signals. Typical control signals include memory read, memory write, I/O read, I/O write, interrupt, interrupt acknowledge, bus request, and bus grant. These control signals indicate the type of action taking place on the system bus. For example, when the processor is writing data into the memory, the memory write signal is generated. Similarly, when the processor is reading from an I/O device, it generates the I/O read signal.

2. Even though processors have a large memory address space, only a fraction of this address space actually contains memory. For example, even though the Pentium has 4 GB of address space, most PCs now have between 128 MB and 256 MB of memory. Furthermore, this memory is shared between the system and application software. Thus, the amount of memory available to run a program is considerably smaller. In addition, if you run more programs simultaneously, each application gets an even smaller amount of memory. You might have experienced the result of this: terrible performance.

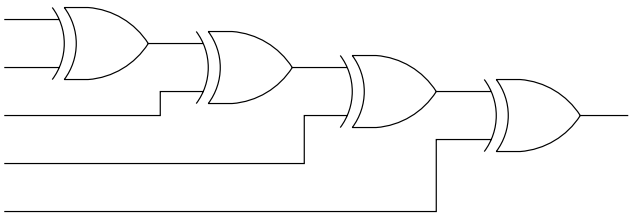
   Apart from the performance issue, this scenario also causes another more important problem: What if your application does not fit into its allotted memory space? How do you run such an application program? This is the motivation for proposing virtual memory.

   Virtual memory was developed to eliminate the physical memory size restriction mentioned before. There are some similarities between the cache memory and virtual memory. Just as with the cache memory, we would like to use the relatively small main memory and create the illusion (to the programmer) of a much larger memory, as shown in the following figure. The programmer is concerned only with the virtual address space. Programs use virtual addresses and when these programs are run, their virtual addresses are mapped to physical addresses at run time.



   The illusion of larger address space is realized by using much slower disk storage. Virtual memory can be implemented by devising an appropriate mapping function between the virtual and physical address spaces.

3. The logic circuit is shown below:



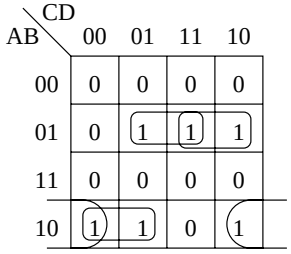
4.

$$\begin{aligned}
\overline{B}\,\overline{D} + A\,\overline{C}\,D + A\,B\,D &= \overline{B}\,\overline{D}\,(1 + A\,\overline{C}) + A\,\overline{C}\,D\,(B + \overline{B}) + A\,B\,D \\
&= \overline{B}\,\overline{D} + \overline{B}\,\overline{D}\,A\,\overline{C} + A\,B\,\overline{C}\,D + A\,\overline{B}\,\overline{C}\,D + A\,B\,D \\
&= \overline{B}\,\overline{D} + A\,\overline{B}\,\overline{C}\,(D + \overline{D}) + A\,B\,D\,(1 + \overline{C}) \\
&= \overline{B}\,\overline{D} + A\,\overline{B}\,\overline{C} + A\,B\,D
\end{aligned}$$

5. The truth table is shown below:

A	B	C	D	F
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0

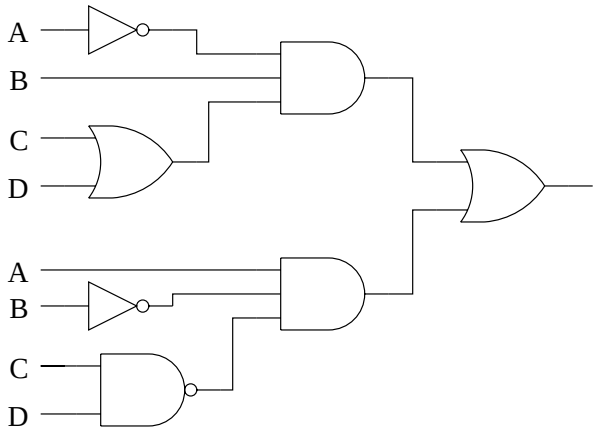
From the following Karnaugh map



we derive the simplified expression as

$$A\,\overline{B}\,\overline{D} + A\,\overline{B}\,\overline{C} + \overline{A}\,B\,D + \overline{A}\,B\,C = A\,\overline{B}\,(\overline{C} + \overline{D}) + \overline{A}\,B\,(C + D)$$

The following circuit implements this logic expression:


A
B
C
D
A
B
C
D