**Memory Address Map**

**The designer of a computer system must calculate the amount of memory required for the particular application and assignment it to either RAM or ROM. The interconnection between memory and processor is then established from knowledge of the size of memory needed and type of RAM and ROM chips available.**

**The addressing of memory can be established by means of a table that specifies the memory address assigned to each chip. The table, called a *memory address map*, is a pictorial representation of assigned address space for each chip in the system.**

**For example, assume that a computer system needs 512 bytes of RAM and 512 bytes of ROM.**

****

**Here the capacity of each RAM Chip is 128 bytes and capacity of ROM Chip is 512 bytes.**

**The component column specifies whether a RAM or a ROM chip is used. The hexadecimal address column assigns a range of hexadecimal equivalent address for each chip. The address bus lines are listed in the third column.**

**Although there are 16 lines in the address bus, the table shows only 10 lines because the other 6 are not used in this example and are assumed to be zero. The small x’s under the address bus lines designate those lines that must be connected to the address inputs in each chip.**

**The RAM chips have 128 bytes and need seven address lines. (Because 128= 27). The ROM Chip has 512 bytes and needs 9 address lines. (Because 512= 29). The x’s are always assigned to the lower-order bus lines i.e. lines 1 through 7 for RAM and lines 1 through 9 for the ROM.**

**Lines 8 and 9 represent four distinct binary combinations to distinguish between four RAM Chips. Bus line 10 is used to distinction between a RAM and ROM address. When line 10 is 0, the CPU selects a RAM and when line 10 is 1 CPU selects ROM.**

**The equivalent hexadecimal address for each chip is obtained from the information under the address bus assignment. The address bus lines are divided into groups of four bits each. Each group can be represented with a hexadecimal digit.**