7. Suppose that a 16M x 16 main memory is built using 512K × 8 RAM chips and memory is word-addressable.

a. How many RAM chips are necessary?
b. How many RAM chips are there per memory word?
c. How many address bits are needed for each RAM chip?
d. How many banks will this memory have?
e. How many address bits are needed for all of memory?
f. If high-order interleaving is used, where would address 14 (which is E in hex) be located?
g. Repeat Exercise 7f for low-order interleaving.

8. A digital computer has a memory unit with 24 bits per word. The instruction set consists of 150 different operations. All instructions have an operation code part (opcode) and an address part (allowing for only one address). Each instruction is stored in one word of memory.

a. How many bits are needed for the opcode?
b. How many bits are left for the address part of the instruction?
c. What is the maximum allowable size for memory?
d. What is the largest unsigned binary number that can be accommodated in one word of memory?

10. Given a memory of 2048 bytes consisting of several 64 x 8 RAM chips. Assuming byte-addressable memory, which of the following seven diagrams indicates the correct way to use the address bits? Explain your answer.

a. 
2 bits for chip select  8 bits for address on chip  10-bit address

b. 
16 bits for chip select  48 bits for address on chip  64-bit address

c. 
6 bits for chip select  5 bits for address on chip  11-bit address

d. 
1 bit for chip select  5 bits for address on chip  6-bit address

e. 
5 bits for chip select  6 bits for address on chip  11-bit address

f. 
4 bits for chip select  6 bits for address on chip  10-bit address

g. 
8 bits for chip select  56 bits for address on chip  64-bit address
13. List the hexadecimal code for the following program (hand assemble it).

**Hex**

<table>
<thead>
<tr>
<th>Address</th>
<th>Label</th>
<th>Instruction</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td></td>
<td>Load A</td>
</tr>
<tr>
<td>101</td>
<td></td>
<td>Add one</td>
</tr>
<tr>
<td>102</td>
<td></td>
<td>Jump S1</td>
</tr>
<tr>
<td>103</td>
<td>S2,</td>
<td>Add One</td>
</tr>
<tr>
<td>104</td>
<td></td>
<td>Store A</td>
</tr>
<tr>
<td>105</td>
<td></td>
<td>Halt</td>
</tr>
<tr>
<td>106</td>
<td>S1,</td>
<td>Add A</td>
</tr>
<tr>
<td>107</td>
<td></td>
<td>Jump S2</td>
</tr>
<tr>
<td>108</td>
<td>A,</td>
<td>HEX 0023</td>
</tr>
<tr>
<td>109</td>
<td>One,</td>
<td>HEX 0001</td>
</tr>
</tbody>
</table>

18. Write the following code segment in MARIE assembly language. (Hint: Turn the for loop into a while loop):

```plaintext
Sum = 0;
for X = 1 to 10 do
    Sum = Sum + X;
```

25. Suppose we add the following instruction to MARIE's ISA:

**IncSZ**  **Operand**

This instruction increments the value with effective address "Operand," and if this newly incremented value is equal to 0, the program counter is incremented by 1. Basically, we are incrementing the operand, and if this new value is equal to 0, we skip the next instruction. Show how this instruction would be written using RTN.

26. Draw the connection of MARIE's PC to the datapath using the format shown in Figure 4.15 (below).
28. Draw the combinational logic for signal controls of MARIE’s Load instruction using Figure 4.18 (below) as an example.