1. The IEEE single-precision floating-point format is composed of a 1-bit sign, an 8-bit biased exponent with a bias of 127 and a 23-bit fraction. Consider the following two operands X and Y in this format, which are specified in hex:

   \[ X = \text{C1A00000} \quad Y = \text{BE800000} \]

(a) (16 points) Determine the decimal value of each of the operands.

(b) (9 points) Determine the IEEE single-precision floating-point representation (specified in hex) for the product \( X \times Y \). (Perform the multiplication in decimal and then convert the result into the IEEE format.)

2. The following machine language VeSPA program is placed into the file v.out. (Note that some instruction encodings are provided on the last page.)

   58 80 00 0F
   28 44 00 00
   30 02 10 00
   F8 00 00 00

(a) (12 points) Give the corresponding assembly language program.

(b) (18 points) Give the 32-bit contents (specified in hex) of the PC and registers r0 through r3, inclusive, and the values of the condition code bits C, V, Z and N after each instruction has been executed by behavioral.v.

3. On a certain computer, instructions in Class A have a CPI of 1, instructions in Class B have a CPI of 3 and instructions in Class C have a CPI of 4. Consider the following two code sequences: Code sequence 1 contains 3 instructions in Class A, 1 instruction in Class B and 1 instruction in Class C. Code sequence 2 contains 2 instructions in Class A, 3 instructions in Class B and 5 instructions in Class C.

(a) (4 points) Determine the number of instructions executed in each sequence.

(b) (8 points) Determine the number of CPU clock cycles required for each sequence.

(c) (8 points) Determine the CPI for each sequence.
4. (25 points) A file called easy.v contains the following:

module easy;

reg [7:0] a, b, r, s, t, u, v, w;

initial repeat (5) begin
  a = $random;
  b = $random;
  r = {a[6:3], b[4:1]};
  s = ¬r;
  t = r + s;
  u = s^t;
  v = u - r;
  w = t*(v + 1);

  #10 $display($time, " %d", w);
end

endmodule

If the command:

verilog easy.v

is executed, give the output that will be produced by the $display statement. Be sure to clearly show how you obtained your answer by including all of the intermediate calculations.
ADD* - Addition

*This instruction sets the condition code bits.

Assembly code notation
a) ADD rdst, rs1, rs2
b) ADD rdst, rs1, #immed16

Instruction encoding
a)

<table>
<thead>
<tr>
<th>31 ... 27</th>
<th>26 ... 22</th>
<th>21 ... 17</th>
<th>16</th>
<th>15 ... 11</th>
<th>10 ... 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>00001</td>
<td>rdst</td>
<td>rs1</td>
<td>0</td>
<td>rs2</td>
<td>000 0000 0000</td>
</tr>
</tbody>
</table>

b)

<table>
<thead>
<tr>
<th>31 ... 27</th>
<th>26 ... 22</th>
<th>21 ... 17</th>
<th>16</th>
<th>15 ... 0</th>
<th>immed16</th>
</tr>
</thead>
<tbody>
<tr>
<td>00001</td>
<td>rdst</td>
<td>rs1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

HLT - Halt

Assembly code notation
HLT

Instruction encoding

<table>
<thead>
<tr>
<th>31 ... 27</th>
<th>26 ... 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>11111</td>
<td>000 0000 0000 0000 0000 0000 0000</td>
</tr>
</tbody>
</table>

LDI - Load immediate

Assembly code notation
LDI rdst, #value

Instruction encoding

<table>
<thead>
<tr>
<th>31 ... 27</th>
<th>26 ... 22</th>
<th>21 ... 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>01011</td>
<td>rdst</td>
<td>immed22</td>
</tr>
</tbody>
</table>

NOT - Bit-wise logical complement

Assembly code notation
a) NOT rdst, rs1

Instruction encoding

<table>
<thead>
<tr>
<th>31 ... 27</th>
<th>26 ... 22</th>
<th>21 ... 17</th>
<th>16 ... 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>00101</td>
<td>rdst</td>
<td>rs1</td>
<td>000000 0000 0000 0000 0000</td>
</tr>
</tbody>
</table>

XOR - Bit-wise logical exclusive-OR

Assembly code notation
a) XOR rdst, rs1, rs2
b) XOR rdst, rs1, #immed16

Instruction encoding

a)

<table>
<thead>
<tr>
<th>31 ... 27</th>
<th>26 ... 22</th>
<th>21 ... 17</th>
<th>16</th>
<th>15 ... 11</th>
<th>10 ... 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>00110</td>
<td>rdst</td>
<td>rs1</td>
<td>0</td>
<td>rs2</td>
<td>000 0000 0000</td>
</tr>
</tbody>
</table>

b)

<table>
<thead>
<tr>
<th>31 ... 27</th>
<th>26 ... 22</th>
<th>21 ... 17</th>
<th>16</th>
<th>15 ... 0</th>
<th>immed16</th>
</tr>
</thead>
<tbody>
<tr>
<td>00110</td>
<td>rdst</td>
<td>rs1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>