```
1   module RAM32x4TopModule(SW, KEY, HEX0, HEX2, HEX4, HEX5);
2      input [9:0] SW;
3      input [0:0] KEY;
4      output [6:0] HEX0, HEX2, HEX4, HEX5;
5      wire [3:0] q;
6      reg [3:0] dataIn;
7      reg [4:0] address;
8      reg clock, writeEn;
9      reg [3:0] upperaddress;
10
11     always@(*)
12     begin
13        dataIn = SW[3:0];
14        address = SW[8:4];
15        clock = KEY[0];
16        writeEn = SW[9];
17        upperaddress[3]=0;
18        upperaddress[2]=0;
19        upperaddress[1]=0;
20        upperaddress[0]=SW[8];
21     end
22
23     ram32x4 c1(SW[8:4],KEY[0], SW[3:0], SW[9], q);
24     HexDecoder h0(q, HEX0);
25     HexDecoder h2(SW[3:0], HEX2);
26     HexDecoder h4(SW[7:4], HEX4);
27     HexDecoder h5(upperaddress, HEX5);
28  endmodule
29
30  module HexDecoder (In, HEX);
31     input [3:0] In;
32     output [6:0] HEX;
33
34     assign c0 = In[0];
35     assign c1 = In[1];
36     assign c2 = In[2];
37     assign c3 = In[3];
38
39     assign HEX[0] = (~c3 & ~c2 & ~c1 & c0) + (~c3 & c2 & ~c1 & ~c0) + (c3 & ~c2 & c1 & c0) +
    (c3 & c2 & ~c1 & c0);
40     assign HEX[1] = (~c3 & c2 & ~c1 & c0) + (~c3 & c2 & c1 & ~c0) + (c3 & ~c2 & c1 & c0) + (
    c3 & c2 & ~c1 & ~c0) + (c3 & c2 & c1 & ~c0) + (c3 & c2 & c1 & c0);
41     assign HEX[2] = (~c3 & ~c2 & c1 & ~c0) + (c3 & c2 & ~c1 & ~c0) + (c3 & c2 & c1 & ~c0) + (
    c3 & c2 & c1 & c0);
42     assign HEX[3] = (~c3 & ~c2 & ~c1 & c0) + (~c3 & c2 & ~c1 & ~c0) + (~c3 & c2 & c1 & c0) +
    (c3 & ~c2 & ~c1 & c0) + (c3 & ~c2 & c1 & ~c0) + (c3 & c2 & c1 & c0);
43     assign HEX[4] = (~c3 & ~c2 & ~c1 & c0) + (~c3 & ~c2 & c1 & c0) + (~c3 & c2 & ~c1 & ~c0) +
     (~c3 & c2 & ~c1 & c0) + (~c3 & c2 & c1 & c0) + (c3 & ~c2 & ~c1 & c0);
44     assign HEX[5] = (~c3 & ~c2 & ~c1 & c0) + (~c3 & ~c2 & c1 & ~c0) + (~c3 & ~c2 & c1 & c0) +
     (~c3 & c2 & c1 & c0) + (c3 & c2 & ~c1 & c0);
45     assign HEX[6] = (~c3 & ~c2 & ~c1 & ~c0) + (~c3 & ~c2 & ~c1 & c0) + (~c3 & c2 & c1 & c0) +
     (c3 & c2 & ~c1 & ~c0);
46  endmodule
```