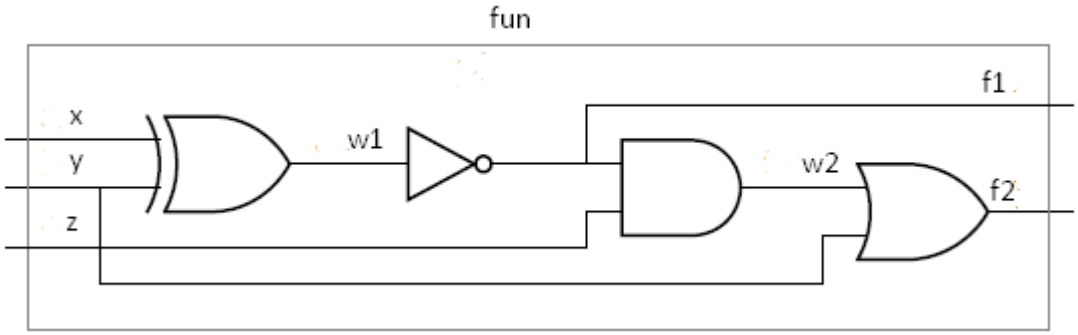
Practice Quiz 2 + Solution

Q1.
Draw the logic circuit described by the Verilog code below.

```
module fun (x, y, z, f1, f2);

      //module fun (x, y, a, b, c);
      /* This was the provided list of port. You can spot the
      error. I intentionally included this mistake. I wanted to
      test if you carefully read the problem before attempting
      to solve it. Once you fix the error, your task was to
      draw the logic diagram of the synthesized hardware, as
      shown in the solution below. */

      input x, y, z;
      output f1, f2;

      wire w1, w2;

      xor x1 (w1, x, y);
      not n1 (f1, w1);
      and a1 (w2, f1, z);
      or o1 (f2, w2, y);

endmodule
```

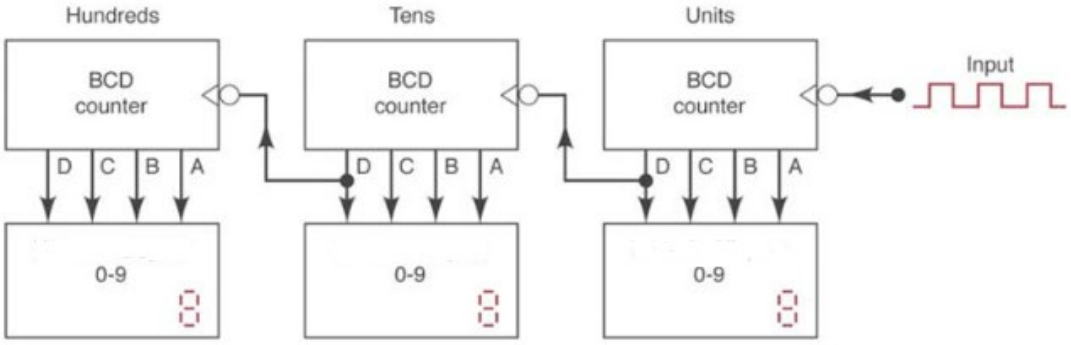
****Solution****



//See HW1's Problem 1

Q2.

BCD counters, also known as decade or MOD-10 counters, are often used whenever pulses are to be counted and the results displayed in decimal. A single BCD counter counts from 0 to 9 and then recycles to 0. To count a larger number than 9, we should cascade a multiple of BCD counters, one for each decade. For example, to construct a BCD counter operation that counts from 000 to 999 we need a three-BCD counter and should proceed with the following design:



1. Initially all counters are reset to 0.
2. Each input pulse advances the first counter once.
3. The 10th input pulse causes the first counter to recycle, which advances the second counter.
4. This continues until the second counter recycles, which advances the third counter.
5. The cycle repeats until 999 is reached and all three counters start again at zero.

In this problem you are required to design a **three_BCD counter** using as building block the BCD counter given below.

```
module BCD (CLOCK, CLR, Q);

     input CLOCK, CLR; //Clock and Reset
     output [3:0] Q;  //BCD output
     reg [3:0] Q;

     always @(negedge CLOCK)
          if (CLR | Q==4'd9)
               Q <= 4'd0;
          else
               Q <= Q + 1;

endmodule
```

Use the following module template for your **three_BCD**:

```
module three_BCD (CLOCK, CLR, BCDu, BCDt, BCDh);

      input CLOCK, CLR;
      output [3:0] BCDu, BCDt, BCDh; //u=1, t=10, h=100
      //WRITE YOUR CODE HERE



endmodule
```

****Solution****

```
module three_BCD (CLOCK, CLR, BCDu, BCDt, BCDh);

      input CLOCK, CLR;
      output [3:0] BCDu, BCDt, BCDh; //u=1, t=10, h=100
      //WRITE YOUR CODE HERE

      BCD c1 (CLOCK, CLR, BCDu[3:0]);
      BCD c2 (BCDu[3], CLR, BCDt[3:0]);
      BCD c3 (BCDt[3], CLR, BCDh[3:0]);

endmodule

//Similar to HW3's Problems 2 & 8
```