Binary to gray code converter

```
// Code your testbench here
// or browse Examples
module tb_binary_to_gray;
  reg [3:0] bin;
  wire [3:0] gray;

  binary_to_gray uut (.bin(bin),
.gray(gray));

  initial begin
    $monitor("Time=%0t | Binary=%b |
Gray=%b", $time, bin, gray);
    bin=4'b0000;#10;
    bin=4'b0001;#10;
    bin=4'b0010;#10;
    bin=4'b0100;#10;
    bin=4'b1111;#10;
    $finish;
  end
endmodule
```

```
// Code your design here
module binary_to_gray (
  input [3:0]bin,
  output [3:0]gray
);
  assign gray[3] = bin[3];
  assign gray[2] = bin[3] ^bin[2];
  assign gray[1] = bin[2] ^bin[1];
  assign gray[0] = bin[1] ^bin[0];
endmodule
```

Output:

Time=0 | Binary=0000 | Gray=0000
Time=10 | Binary=0001 | Gray=0001
Time=20 | Binary=0010 | Gray=0011
Time=30 | Binary=0100 | Gray=0110
Time=40 | Binary=1111 | Gray=1000

Gray to binary code converter

```
// Code your testbench here
// or browse Examples
module tb_gray_to_binary;
  reg [3:0] gray;
  wire [3:0] bin;

  gray_to_binary uut (.gray(gray),
.bin(bin));

  initial begin
    $monitor("Time=%0t | Gray=%b |
Binary=%b", $time, gray, bin);
    gray=4'b0000;#10;
    gray=4'b0001;#10;
    gray=4'b0010;#10;
    gray=4'b0011;#10;
    gray=4'b0100;#10;
    gray=4'b0101;#10;
    gray=4'b0110;#10;
    gray=4'b1111;#10;
    $finish;
  end
endmodule
```

```
// Code your design here
module gray_to_binary (
  input [3:0]gray,
  output [3:0]bin
);
  assign bin[3] = gray[3];
  assign bin[2] = gray[3] ^ gray[2];
  assign bin[1] = bin[2] ^ gray[1];
  assign bin[0] = bin[1] ^ gray[0];
endmodule
```

Output:

Time=0 | Gray=0000 | Binary=0000
Time=10 | Gray=0001 | Binary=0001
Time=20 | Gray=0010 | Binary=0011
Time=30 | Gray=0011 | Binary=0010
Time=40 | Gray=0100 | Binary=0111
Time=50 | Gray=0101 | Binary=0110
Time=60 | Gray=0110 | Binary=0100
Time=70 | Gray=1111 | Binary=1010

Binary to BCD code converter

```
// Code your testbench here
// or browse Examples
module tb_binary_to_bcd;
  reg [3:0] bin;
  wire [3:0] ones,tens;

  binary_to_bcd_gate uut(.bin(bin),
.ones(ones),.tens(tens));

  initial begin
    $monitor("bin=%b => tens=%d ones=%d",
bin, tens, ones);
    bin = 4'd0;#10;
    bin = 4'd5;#10;
    bin = 4'd7;#10;
    bin = 4'd9;#10;
    bin = 4'd10;#10;
    bin = 4'd12;#10;
    bin = 4'd15;#10;
    $finish;
  end
endmodule
```

```
// Code your design here
module binary_to_bcd_gate(input[3:0]bin,
output [3:0]ones,output[3:0] tens);
  wire [3:0]b;
  assign b=bin;

  assign tens[0] =(b[3] & b[2]) | (b[3] &
b[1]);
  assign tens[1] =(b[3] & b[2] & ~b[1]) |
(b[3] & b[1] & b[0]);
  assign tens[2] =0;
  assign tens[3] =0;

  assign ones= b-(tens*4'd10);
endmodule
```

Output:

bin=0000 => tens= 0 ones= 0

bin=0101 => tens= 0 ones= 5

bin=0111 => tens= 0 ones= 7

bin=1001 => tens= 0 ones= 9

bin=1010 => tens= 1 ones= 0

bin=1100 => tens= 3 ones=14

bin=1111 => tens= 3 ones= 1

BCD to binary code converter

```
// Code your testbench here
// or browse Examples
module tb_bcd_to_binary;
  reg [7:0]bcd;
  wire [6:0]bin;

  bcd_to_binary uut(.bcd(bcd), .bin(bin));

  initial begin
    $monitor("BCD=%b -> Binary=%d", bcd,
bin);
    bcd = 8'b0000_0101;#10;
    bcd = 8'b0001_0010;#10;
    bcd = 8'b0010_0101;#10;
    bcd = 8'b0101_1001;#10;
    bcd = 8'b1001_1001;#10;
    bcd = 8'b1010_1010;#10;

    $finish;
  end
endmodule
```

```
// Code your design here
module bcd_to_binary(input [7:0]bcd,
output [6:0] bin);

  assign bin =(bcd[7:4]*4'd10)+ bcd[3:0];
endmodule
```

Output:

BCD=00000101 -> Binary= 5

BCD=00010010 -> Binary= 12

BCD=00100101 -> Binary= 25

BCD=01011001 -> Binary= 59

BCD=10011001 -> Binary= 99

BCD=10101010 -> Binary=110

Excess 3 to BCD converter

```
// Code your testbench here
// or browse Examples
module tb_excess3_to_bcd;
  reg [3:0]ex3;
  wire [3:0]bcd;

  excess3_to_bcd uut(.ex3(ex3),
.bcd(bcd));

  initial begin
    $monitor("Excess-3 = %b, BCD = %b",
ex3, bcd);
ex3 = 4'b0011;#10;
ex3 = 4'b0010;#10;
ex3 = 4'b0001;#10;
ex3 = 4'b0100;#10;
    ex3 = 4'b0101;#10;
    ex3 = 4'b0110;#10;
    ex3 = 4'b0111;#10;
    $finish;
  end
endmodule
```

```
// Code your design here
module excess3_to_bcd(input[3:0]ex3,
output [3:0]bcd);
  assign bcd =ex3-4'd3;
endmodule
```

Output:

Excess-3 = 0011, BCD = 0000

Excess-3 = 0010, BCD = 1111

Excess-3 = 0001, BCD = 1110

Excess-3 = 0100, BCD = 0001

Excess-3 = 0101, BCD = 0010

Excess-3 = 0110, BCD = 0011

Excess-3 = 0111, BCD = 0100

BCD to excess 3 converter

```
// Code your testbench here
// or browse Examples
module tb_bcd_to_excess3;
  reg [3:0]bcd;
  wire [3:0]ex3;

  bcd_to_excess3 uut(.bcd(bcd),
.ex3(ex3));

  initial begin
    $monitor("BCD = %b, Excess-3 = %b",
bcd, ex3);
    bcd = 4'b0000;#10;
    bcd = 4'b0001;#10;
    bcd = 4'b0010;#10;
    bcd = 4'b1001;#10;
bcd = 4'b1010;#10;
bcd = 4'b1011;#10;
bcd = 4'b1101;#10;
bcd = 4'b1110;#10;

    $finish;
  end
endmodule
```

```
// Code your design here
module bcd_to_excess3(input [3:0]bcd,
output [3:0]ex3);
  assign ex3 = bcd+4'd3;
endmodule
```

Output:

BCD = 0000, Excess-3 = 0011
BCD = 0001, Excess-3 = 0100
BCD = 0010, Excess-3 = 0101
BCD = 1001, Excess-3 = 1100
BCD = 1010, Excess-3 = 1101
BCD = 1011, Excess-3 = 1110
BCD = 1101, Excess-3 = 0000
BCD = 1110, Excess-3 = 0001

ASCII to binary code converter

```
// Code your testbench here
// or browse Examples
module tb_ascii_to_binary;
  reg [7:0]ascii;
  wire [3:0]binary;

  ascii_to_binary uut(.ascii(ascii),
.binary(binary));

  initial begin
    $monitor("ASCII = %d (%c), Binary =
%d", ascii, ascii, binary);
    ascii = 8'd48;#10;
    ascii = 8'd49;#10;
    ascii = 8'd50;#10;
    ascii = 8'd57;#10;
ascii = 8'd52;#10;
ascii = 8'd53;#10;

    $finish;
  end
endmodule
```

```
// Code your design here
module ascii_to_binary(input [7:0] ascii,
output [3:0] binary);
  assign binary =ascii-8'd48;
endmodule
```

Output:

ASCII = 48 (0), Binary = 0
ASCII = 49 (1), Binary = 1
ASCII = 50 (2), Binary = 2
ASCII = 57 (9), Binary = 9
ASCII = 52 (4), Binary = 4
ASCII = 53 (5), Binary = 5