

Verilog-HDL ゼミ 第2週・練習問題「assign 文とテストベンチ」

問 1.

$A \cdot B$ と $\overline{A + B}$ のうち、信号 $sel=0$ のときには $A \cdot B$ を、 $sel=1$ のときには $\overline{A + B}$ を出力するモジュール `sel2to1`(プログラムリスト 1)を条件演算子を使用して完成してみてください。

また、以下のテストベンチ `sel2to1_tb.v`(プログラムリスト 2)を完成させ、2to1 セレクタの出力を確認してください。その際、表 1 の真理値表の結果となるはずです。

表 1. 2to1 セレクタの真理値表

A	B	sel	out
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	0

```
module sel2to1 ( a, b, sel, out ); // 2to1 selector
// 入力ポートの宣言
input  ...;
// 出力ポートの宣言
output [1:0]  ...;

// 以下より assign 文による 2to1 セレクタの出力の記述
...;

endmodule
```

プログラムリスト 1. `sel2to1.v`

```

`timescale 1 ps / 1 ps

module sel2to1_tb;
// 2to1 セクタへの入力
reg    a,b,sel;
// 2to1 セクタからの出力
wire   [1:0] out;

parameter STEP = 1000;

// 2to1 セクタを呼び出す
...;

initial begin
    $dumpfile("sel2to1_tb.vcd");
    $dumpvars(0, sel2to1_tb);
    $monitor( $stime, " a=%b b=%b sel=%b out=%b", a, b, sel, out);

    // STEP 毎に入力パターンを変化させて全入力パターンを網羅
    #STEP      a = 0; b = 0; sel = 0;
    #STEP      a = 0; b = 0; sel = 1;
    // 以下より残りの入力パターンを与える
    ...;
    #STEP $finish;      // テストベンチの終了
end

endmodule

```

プログラムリスト 2. sel2to1_tb.v

コマンド例) > iverilog -o sel2to1_tb -s sel2to1_tb sel2to1_tb.v sel2to1.v

コンパイル

vvp sel2to1_tb # monitor による出力の確認

gtkwave sel2to1_tb.vcd # 波形の確認

or

cver sel2to1_tb.v sel2to1.v # コンパイル

gtkwave sel2to1_tb.vcd # 波形の確認

```

C:\Users\pnzoz\Documents\etc\zemi\2021\week02>vvp sel2to1_tb
VCD info: dumpfile sel2to1_tb.vcd opened for output.
      0 a=x b=x sel=x out=x
    1000 a=0 b=0 sel=0 out=0
    2000 a=0 b=0 sel=1 out=1
    3000 a=0 b=1 sel=0 out=0
    4000 a=0 b=1 sel=1 out=0
    5000 a=1 b=0 sel=0 out=0
    6000 a=1 b=0 sel=1 out=0
    7000 a=1 b=1 sel=0 out=1
    8000 a=1 b=1 sel=1 out=0

```

図 1. 出力結果の例(vvp)

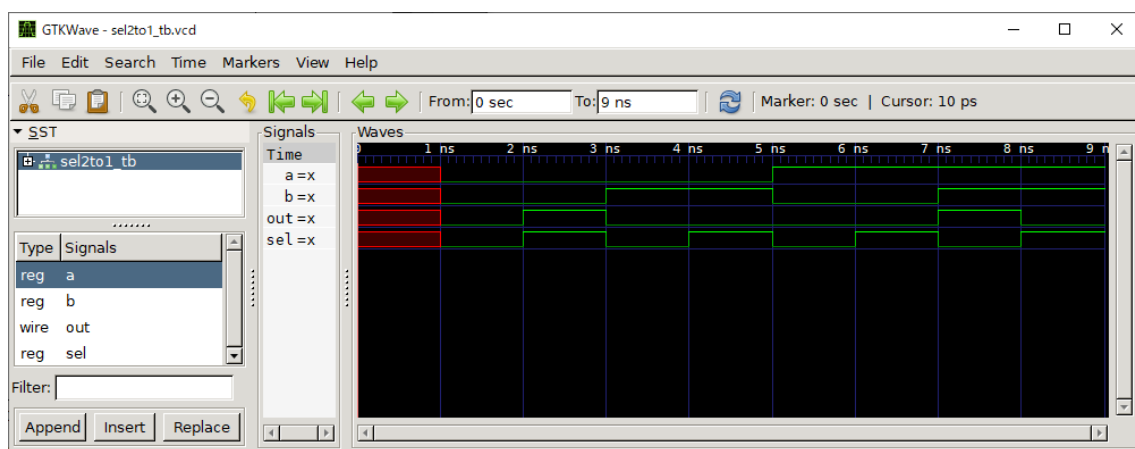


図 2. 波形表示例(gtkwave)

図 1 に示すような、2 ビットの入力(D_1, D_0)を 4 ビットの 10 進コード(Q_3, Q_2, Q_1, Q_0)に変換するデコーダを考えます。10 進コードは、デコーダの真理値表に示すように、各ビットが 10 進コードに対応します。ここでは、3 つのパターンの記述を考えてみます。

プログラムリスト 3 は、等号演算の結果を用いる方法です。この場合は、出力の各ビットに対して、等号演算の結果を代入します。プログラムリスト 4、プログラムリスト 5 に示すコードは、各々 if 文、case 文によりデコーダを記述したものです。両者とも、入力値によって分岐を行い、各分岐では各々の入力値のデコード結果をデコーダの出力として代入します。ここで注意すべき点は、if 文や case 文はステートメントと呼ばれ、initial や always、function 等の各構文の内側にのみ記述できるということです。

case 文による記述の場合、真理値表をそのまま記述するような形になるため、3 通りの記述の中では最も可読性に優れているといえます。

プログラムリスト 6 にデコーダの入出力が正しいかを確認するためのテストベンチ decoder_tb.v を示す。

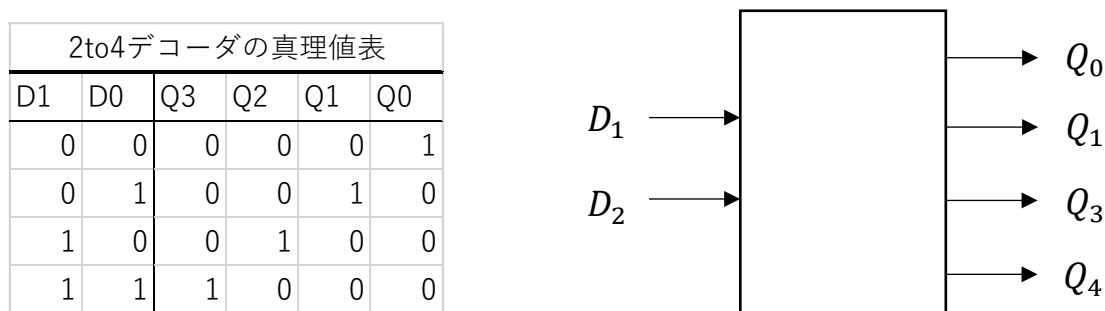


図 3. 2to4 デコーダ

```

module decoder_cond( d, q );
input  [1:0] d;
output [3:0] q;

assign q[0] = (d==2'b00);
assign q[1] = ...; // 入力 d が 2'b01 のとき 1 を出力
assign q[2] = ...; // 入力 d が 2'b10 のとき 1 を出力
assign q[3] = ...; // 入力 d が 2'b11 のとき 1 を出力

endmodule

```

プログラムリスト 3. 等号演算によるデコーダ

```

module decoder_if( d, q );
input  [1:0] d;
outbut [3:0] q;

function [3:0] dec;
input  [1:0] d;
    if ( d==2'd0 )
        dec = 4'b0001;
    // 入力が 1 のときのデコード結果を代入
    else if ( d==2'd1 )
        dec = ...;
    // 入力が 2 のときのデコード結果を代入
    else if ( d==2'd2 )
        dec = ...;
    // 入力が 3 のときのデコード結果を代入
    else
        dec = ...;
endfunction

// デコード結果をデコーダの出力に代入する
assign q = ...;

endmodule

```

プログラムリスト 4. if 文によるデコーダ

```

module decoder_case( d, q );
input  [1:0] d;
outbut [3:0] q;

function [3:0] dec;
input  [1:0] d;
    case ( d )
        2'b00:      dec = 4'b0001;
        2'b01:      dec = ...; // 入力が 1 のときのデコード結果を代入
        2'b10:      dec = ...; // 入力が 2 のときのデコード結果を代入
        2'b11:      dec = ...; // 入力が 3 のときのデコード結果を代入
        default:    dec = ...; // 上記入力に合致しない場合不定値出力
    endcase
endfunction

// デコード結果をデコーダの出力に代入する
assign q = ...;

endmodule

```

プログラムリスト 5. case 文によるデコーダ

```

`timescale 1 ps / 1 ps

module decoder_tb;
reg    [1:0]  d; // 入力を reg 宣言
wire   [3:0]  q; // 出力を wire 宣言

parameter STEP = 1000;

//デコーダの 3 パターンのうちどれかを呼び出す↓
decoder_cond decoder( d, q );
//decoder_if decoder( d, q );
//decoder_case decoder( d, q );

initial begin
    $dumpfile("decoder_tb.vcd");
    $dumpvars(0, decoder_tb);
    $monitor( $stime, " d=%b q=%b", d, q);

                d = 2'b00;
    #STEP        d = 2'b01;
    #STEP        d = 2'b10;
    #STEP        d = 2'b11;
    #STEP        $finish;
end

endmodule

```

プログラムリスト 6. decoder_tb.v

問 2.

プログラムリスト 3~6 の...部分を埋め、プログラムを完成させてください。また、シミュレーション結果が真理値表に一致するかを、コンパイル時の標準出力と、波形表示により確認してみてください。

コマンド例) > iverilog -o decoder -s decoder_tb decoder_tb.v decoder_if.v
vvp decoder
gtkwave decoder_tb.vcd

or

cver decoder_tb.v decoder_if.v (出力されるログで結果を確認)
gtkwave decoder_tb.vcd

```
C:\Users\pnzoz\Documents\etc\zemi\資料\week01\ans>vvp decoder_tb
VCD info: dumpfile decoder_tb.vcd opened for output.
    0 d=00 q=0001
 100000 d=01 q=0010
 200000 d=10 q=0100
 300000 d=11 q=1000
```

図 3. 出力結果例(vvp)

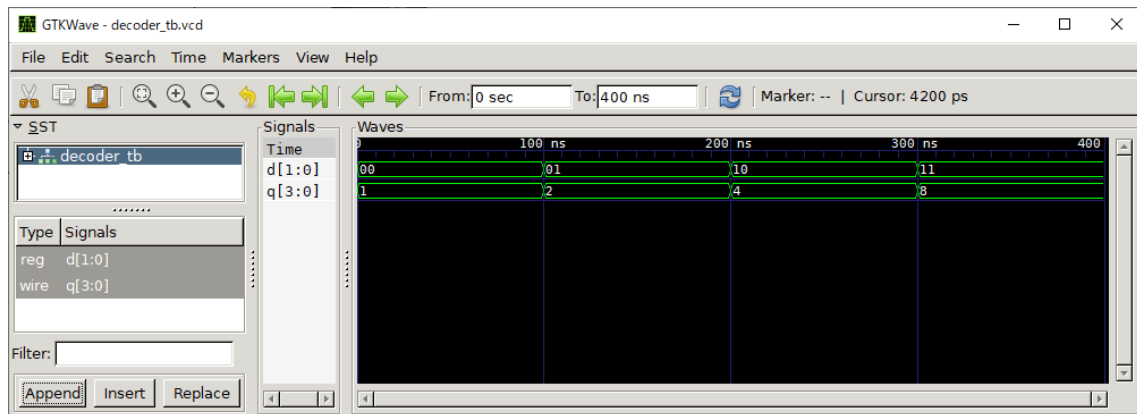


図 4. 波形表示例(gtkwave)

次に、図 2 に示すような、4 ビットの 10 進コード(D_3, D_2, D_1, D_0)を 2 ビットの 2 進数(Q_1, Q_0)に変換するエンコーダを考えます。10 進コードは、デコーダの真理値表に示すように、各ビットが 10 進コードに対応します。ここでは、入力信号の上位ビットに優先順位をつけ、複数の入力が 1 になった場合でも出力が確定するようなプライオリティエンコーダを考えます。これにより、例えば(D_3, D_2, D_1, D_0) = (1,1,0,0)のように D_3 以外のビットに 1 があっても、上位ビットの D_3 が優先され、 D_3 が 1 であれば出力は常に(Q_1, Q_0) = (1,1)となります。

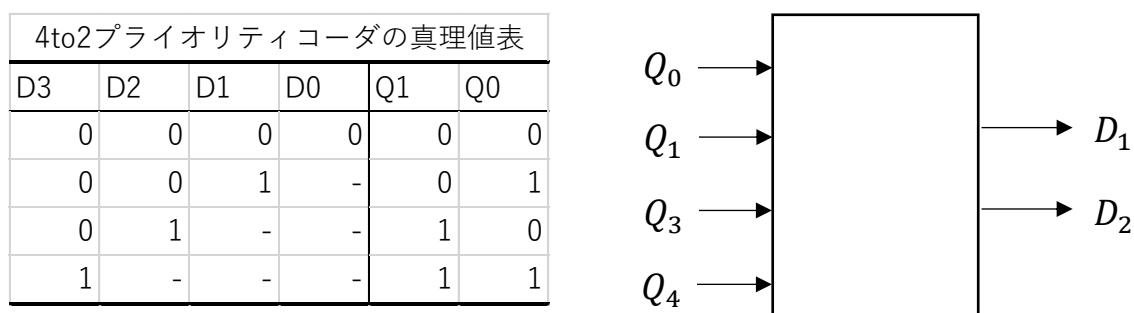


図 5. 4to2 エンコーダ

プログラムリスト 7 に示すコードは、if 文によりエンコーダを実装する方法です。この場合は、if-else 文により、入力信号を上位ビットから見ていき、上位ビットが 1 であれば値を確定します。上位ビットが 1 である場合、それ以降の下位ビットの値は見ません。

プログラムリスト 8 の方は、case 文によるエンコーダの実装です。case 文は、x をドントケアとして扱うので、x のついたビットの値を考慮しません。それにより、例えば入力 d の最上位ビットである d[3]が 1 であれば、他のビットの値に関係なく常に enc に 2'b11 が代入されることになります。これにより、上位ビットに優先順位をつけるようなエンコーダの記述をすることができます。

最後に、プライオリティエンコーダの入出力が正しいかを確認するためのテストベンチ encoder_tb.v をプログラムリスト 9 に示します。

```

module encoder_if ( d, q );
input  [3:0] d;
outbut [1:0] q;

function [1:0] enc;
input [3:0] d;
    if ( d[3] )
        enc = 2'b11;
    else if ( d[2] )
        enc = ...;      // 2'b0100 のエンコード結果を代入
    else if ( d[1] )
        enc = ...;      // 2'b0010 のエンコード結果を代入
    else
        enc = ...;      // 2'b0001 のエンコード結果を代入
endfunction

// エンコード結果をエンコーダの出力に代入する
assign q = ...;

endmodule

```

プログラムリスト 7. if 文によるプライオリティエンコーダ

```

module encoder_casex( d, q );
input  [3:0] d;
outbut [1:0] q;

function [1:0] enc;
input  [3:0] d;
    casex( d )
        4'b1xxx:    enc = 2'b11;
        4'b01xx:    enc = ...;    // 2'b0100 のエンコード結果を代入
        4'b001x:    enc = ...;    // 2'b0010 のエンコード結果を代入
        4'b000x:    enc = ...;    // 2'b0001 のエンコード結果を代入
    endcase
endfunction

// エンコード結果をエンコーダの出力に代入する
assign q = ...;

endmodule

```

プログラムリスト 8. casex 文によるプライオリティエンコーダ

```

`timescale 1 ps / 1 ps

module encoder_tb;
reg          ck, res;
reg  [3:0]   d;    // 入力を reg 宣言
wire  [1:0]  q;    // 出力を wire 宣言

parameter STEP = 100000;

//エンコーダの 2 パターンのうちどちらか呼び出す
encoder_if encoder( d, q );
//encoder_casex( d, q );

always begin
    ck = 0; #(STEP / 2);
    ck = 1; #(STEP / 2);
end

initial begin
    $dumpfile("encoder_tb.vcd");
    $dumpvars(0, encoder_tb);
    $monitor( $stime, " ck=%b res=%b d=%b q=%b", ck, res, d, q);

                                d = 4'b0001;
#STEP                          d = 4'b0010;
#STEP                          d = 4'b0011;
#STEP                          d = 4'b0100;
#STEP                          d = 4'b0101;
#STEP                          d = 4'b0110;
#STEP                          d = 4'b0111;
#STEP                          d = 4'b1000;
#STEP                          d = 4'b1001;
#STEP                          d = 4'b1010;
#STEP                          d = 4'b1011;
#STEP                          d = 4'b1100;
#STEP                          d = 4'b1101;

```

```

        #STEP      d = 4'b1110;
        #STEP      d = 4'b1111;
        #STEP      $finish;
end

endmodule

```

プログラムリスト 9. encoder_tb.v

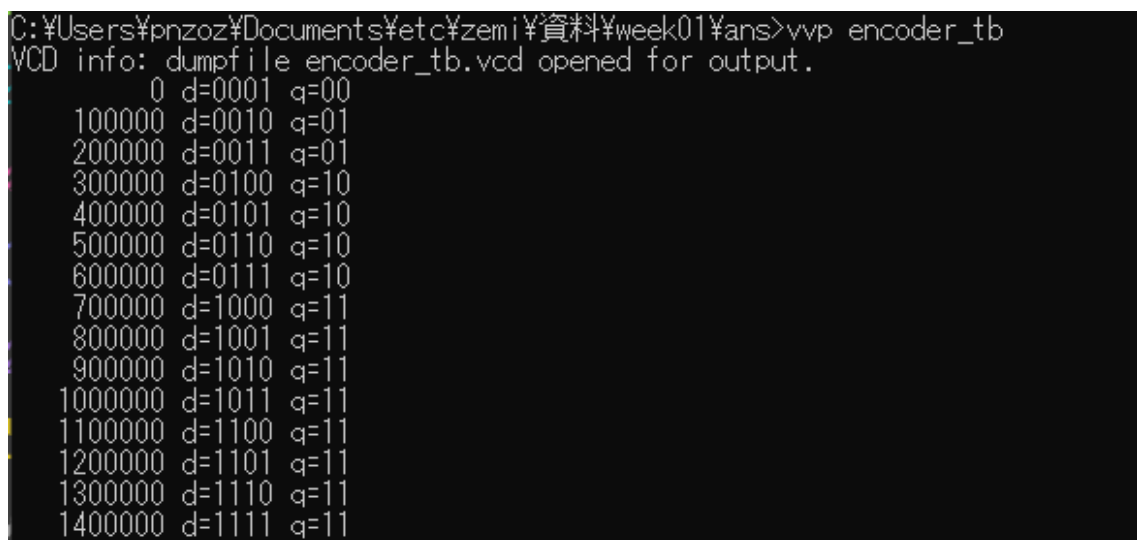
問 3.

プログラムリスト 7~9 の...部分を埋め、プログラムを完成させてください。また、シミュレーション結果が真理値表と一致することを、コンパイル時の標準出力と、波形表示により確認してみてください。

コマンド例) > iverilog -o encoder -s encoder_tb encoder_tb.v encoder_if.v
 vvp encoder
 gtkwave encoder_tb.vcd

or

cver encoder_tb.v encoder_if.v (出力されるログで結果を確認)
 gtkwave encoder_tb.vcd



```

C:\Users\pnzoz\Documents\etc\zemi\資料\week01\ans>vvp encoder_tb
VCD info: dumpfile encoder_tb.vcd opened for output.
  0 d=0001 q=00
100000 d=0010 q=01
200000 d=0011 q=01
300000 d=0100 q=10
400000 d=0101 q=10
500000 d=0110 q=10
600000 d=0111 q=10
700000 d=1000 q=11
800000 d=1001 q=11
900000 d=1010 q=11
1000000 d=1011 q=11
1100000 d=1100 q=11
1200000 d=1101 q=11
1300000 d=1110 q=11
1400000 d=1111 q=11

```

図 6. 出力結果例(vvp)

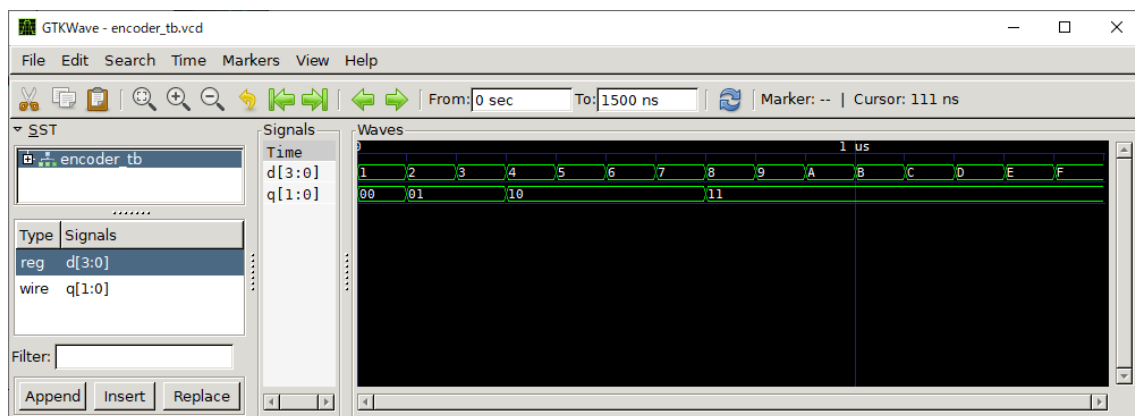


图 7. 波形表示例(gtkwave)