

Logika cyfrowa

Lista zadań nr 5

Termin: 3 i 8 kwietnia 2024

Uwaga! Podczas zajęć należy znać pojęcia zapisane **wytluszczoną czcionką**. W przypadku braku znajomości tych pojęć student może być ukarany punktami ujemnymi.

1. Zaprojektuj obwód, który mnoży ośmiobitową liczbę przez 1, 2, 3 lub 4, w zależności od wartości dodatkowego wejścia wyboru. Możesz wykorzystywać znane z wykładu podstawowe układy kombinacyjne (ale nie multiplikator).
2. Poniższy kod w SystemVerilogu miał implementować dekodery 2 do 4 z dodatkowym wejściem aktywującym, ale posiada błąd. Na czym on polega?

```
module dec2to4(  
    input [1:0] i,  
    input en,  
    output [3:0] o  
);  
    integer k;  
    always_comb  
        for (k = 0; k <= 3; k = k + 1)  
            if (i == k)  
                o[k] = en;  
endmodule
```

3. Jaki obwód jest implementowany przez poniższy kod? Czy użyty styl implementacji jest tu dobrym wyborem?

```
module zadanie(  
    input [1:0] w,  
    input en,  
    output logic y0, y1, y2, y3  
);  
    always_comb begin  
        y0 = 0;  
        y1 = 0;  
        y2 = 0;  
        y3 = 0;  
        if (en)  
            if (w == 0) y0 = 1;  
            else if (w == 1) y1 = 1;  
            else if (w == 2) y2 = 1;  
            else y3 = 1;  
    end  
endmodule
```

4. Pokaż, jak zaimplementować w SystemVerilogu dekodery 3 do 8 wykorzystując dwie instancje dekodera 2 do 4 z wejściem aktywującym i dodatkową logikę.
5. Wadą kodowania BCD jest skomplikowane obliczanie dopełnienia cyfry. Jednym z alternatywnych sposobów kodowania cyfr dziesiętnych jest kodowanie $84-2-1$, w którym najmłodsze dwie cyfry binarne mają ujemne wagi. Zaimplementuj w SystemVerilogu konwerter kodowania cyfr z kodowania $84-2-1$ do BCD.