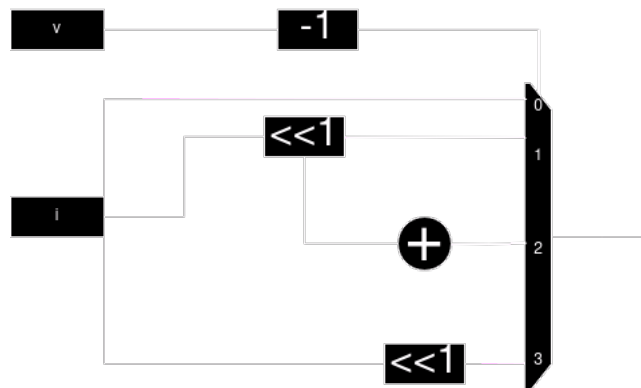


Logika Cyfrowa

Jakub Gałaszewski

April 8, 2024

- 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 onizszy 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
```

UWAGA, output należy jeszcze ustalić na logic, czyli output logic [3:0]o

Problem jest (jeżeli dobrze zrozumiałem) że chcemy aby rozwiązane rozważało wszystkie gałęzie kodu, które są tutaj nierozpatrzone.

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
```

jeżeli mamy bit en zapalony, to jest to dekodery 2 do 4, ale napisane w dosyć zawiłej formie. Wystarczy rozpisać karnaugh'a aby znacząco uprościć ten kod.

```
module zadanie(  
    input [1:0] w,  
    input en,  
    output logic y0, y1, y2, y3  
);  
    always_comb begin  
        y0 = en&~w[1] &~w[0];  
        y1 = en&~w[1]&w[0];  
        y2 = en&w[1] &~w[0];  
        y3 = en&w[1]&w[0];  
    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

```
module bcd(input [3:0]i, output [3:0]o);  
  
    o[3:0] = dec2to4(i[1:0], !i[2]);  
    o[7:4] = dec2to4(i[1:0], i[2]);  
endmodule
```

- 5 Wadą kodowania BCD jest skomplikowane obliczanie dopełnienia cyfry. Jednym z alternatywnych sposobów kodowania cyfr dziesiętnych jest kodowanie 8 4 -2 -1, w którym najmłodsze dwie cyfry binarne mają ujemne wagi. Zaimplementuj w SystemVerilogu konwerter kodowania cyfr z kodowania 8 4 -2 -1 do BCD.

```
module bcd(input [3:0]i, output [3:0]o);  
  
    always_comb  
        o = (i[3:0] & ~3) - i[1:0];  
  
endmodule
```