<table>
<tr><td>Termin zajęć
Piątek – PARZYSTY – -
GODZ. 8:00</td><td colspan="2">Układy cyfrowe i systemy wbudowane</td></tr>
<tr><td colspan="2">Osoby wykonujące ćwiczenie:
Karol Gąsior 280913
Michał Sadowy 280884</td><td>Grupa nr:
14</td></tr>
<tr><td colspan="2">Tytuł ćwiczenia:
Układy kombinacyjne</td><td>Laboratorium nr:
2</td></tr>
<tr><td>Data wykonania ćwiczenia</td><td>10-10-2025</td><td rowspan="2">Ocena:</td></tr>
<tr><td>Data oddania sprawozdania</td><td>24-10-2025</td></tr>
</table>

1 Zadanie 1

Stworzyć układ kombinacyjny o zadanej funkcji **Y** = (**X**+7) mod 16. **X i Y** to wektory 4b w naturalnym kodzie binarnym. Następnie przeprowadzić symulację behawioralna oraz czasową. Całość zaprezentować na sprzęcie, gdzie za wektor X będą służyć klawisze K(3:0) zaś za Y diody LED(3:0).

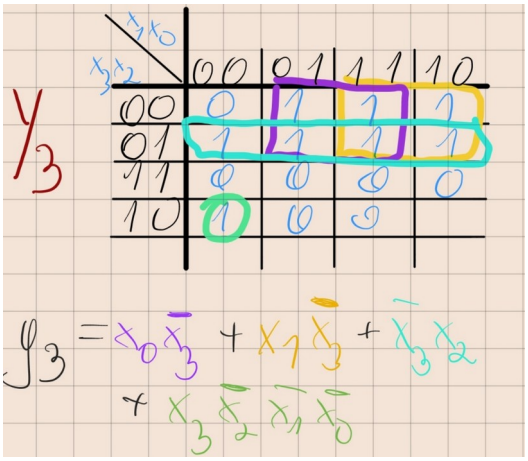
1.1 Stworzenie tabeli prawdy dla zadanej funkcji

Tab. 1: Tabela prawdy funkcji Y = (X+7) mod 16

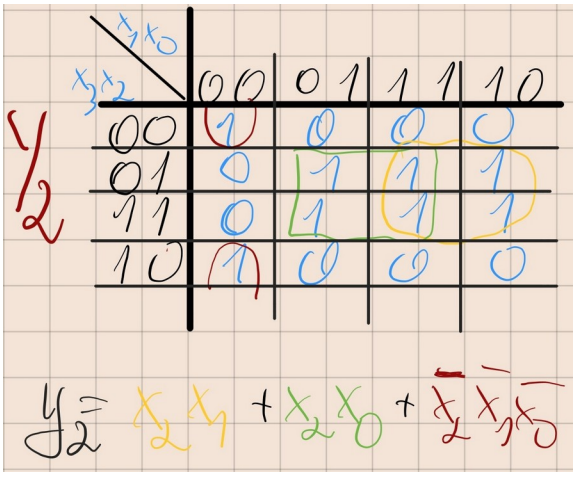
Nr.	x3	x2	x1	x0	y3	y2	y1	y0
0	0	0	0	0	0	1	1	1
1	0	0	0	1	1	0	0	0
2	0	0	1	0	1	0	0	1
3	0	0	1	1	1	0	1	0
4	0	1	0	0	1	0	1	1
5	0	1	0	1	1	1	0	0
6	0	1	1	0	1	1	0	1
7	0	1	1	1	1	1	1	0
8	1	0	0	0	1	1	1	1
9	1	0	0	1	0	0	0	0
10	1	0	1	0	0	0	0	1
11	1	0	1	1	0	0	1	0
12	1	1	0	0	0	0	1	1
13	1	1	0	1	0	1	0	0
14	1	1	1	0	0	1	1	1
15	1	1	1	1	0	1	1	0

1.2 Zaprojektowanie układu kombinacyjnego oraz minimalizacja za pomocą siatek Karnough.

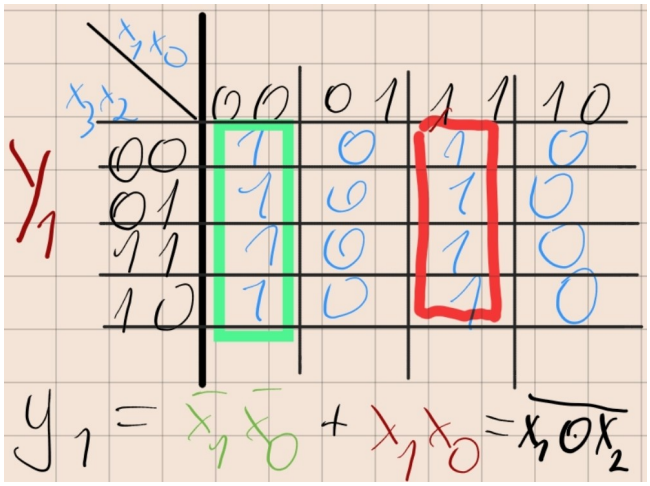
Na podstawie stworzonej tabeli prawdy, stworzono 4 siatki Karnough, odpowiednio dla kolumn y3, y2, y1, y0 tabeli prawdy. Stworzono grupy implicentów a następnie otrzymane funkcje zminimalizowano. Rys.1 do Rys. 4 przedstawiają odpowiedni siatki Karnough dla kolejnych sygnałów wyjściowych oraz minimalizację ich funkcji.



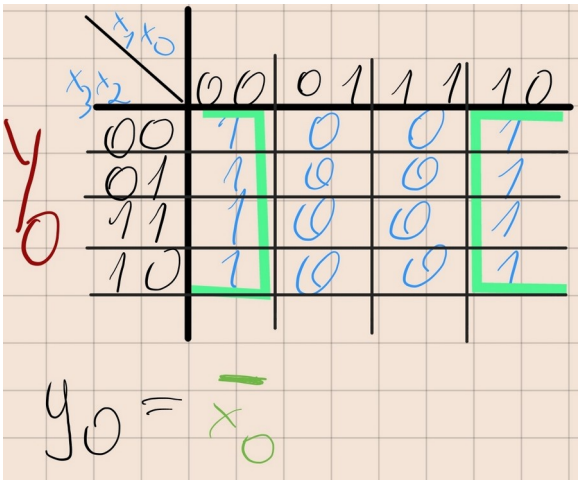
Rys. 1: Siatka Karnough oraz wynik minimalizacji dla y3



Rys. 2: Siatka Karnough oraz wynik minimalizacji dla y2



Rys. 3: Siatka Karnough oraz wynik minimalizacji dla y1

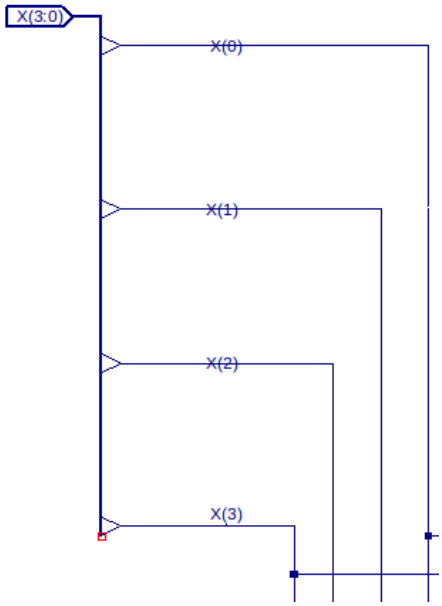


Rys. 4: Siatka Karnough oraz wynik minimalizacji dla y0

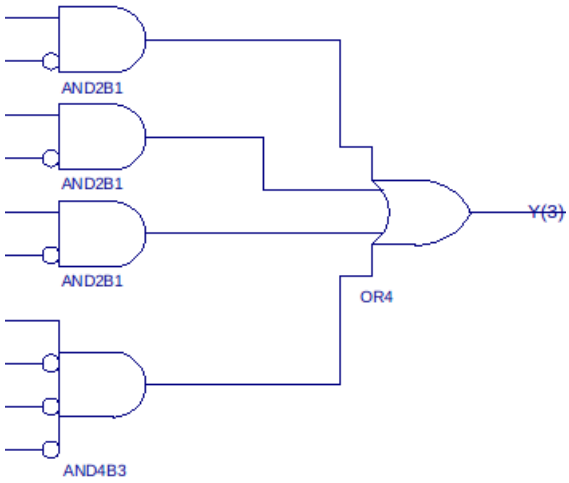
1.3 Stworzenie schematu logicznego w ISE

Na podstawie wyznaczonych funkcji wyjść stworzono schemat logiczny w ISE (plik *.sch). Najpierw stworzono magistralę cztero-bitową którą nazwano **X** (Rys. 5), następnie dodano kolejne bramki oraz połączono je tak, aby stworzone z nich układy logiczne odpowiadały odpowiednim funkcjom wyjścia. Rys. 6 do Rys. 9 przedstawiają schematy tych układów. Na koniec stworzono cztero-bitową magistralę wyjściową i odpowiednio podłączono do niej sygnały wyjściowe z w/w układów reprezentujących funkcję wyjściowe.

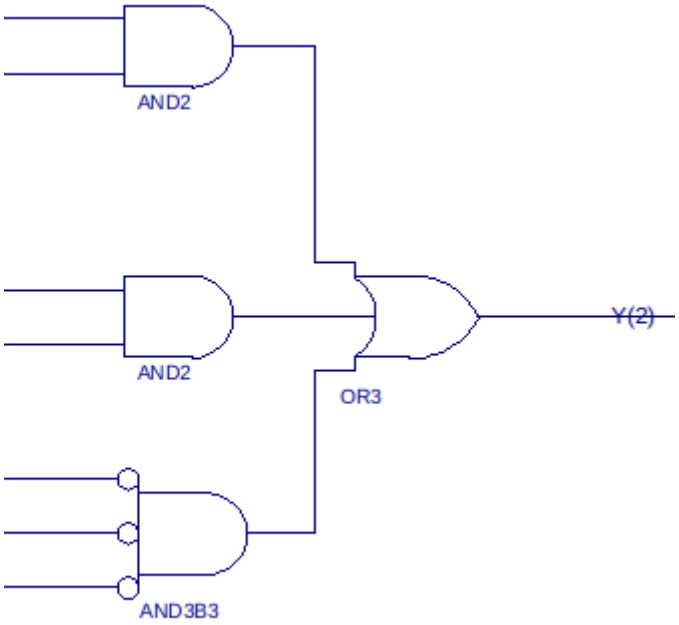
Gotowy schemat logiczny sprawdzono za pomocą przycisku *Check Schematics*.



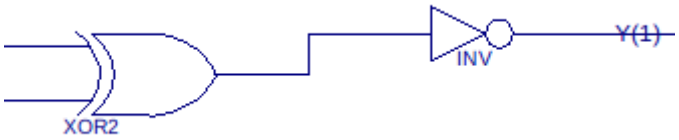
Rys. 5: Cztero-bitowa magistrala



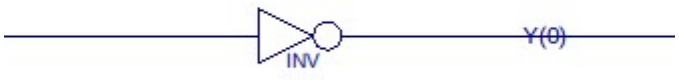
Rys. 6: Układ logiczny odpowiadający funkcji wyjścia dla y3



Rys. 7: Układ logiczny odpowiadający funkcji wyjścia dla y2



Rys. 8: Układ logiczny odpowiadający funkcji wyjścia dla y1



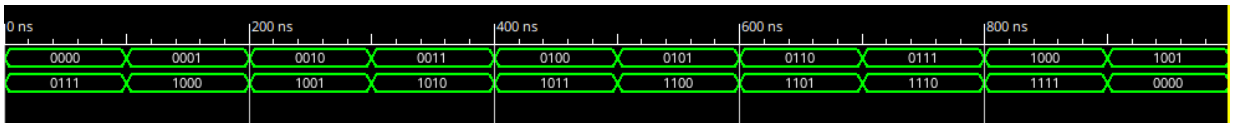
Rys. 9: Układ logiczny odpowiadający funkcji wyjścia dla y0

1.4 **Symulacja behawioralna i czasowa**

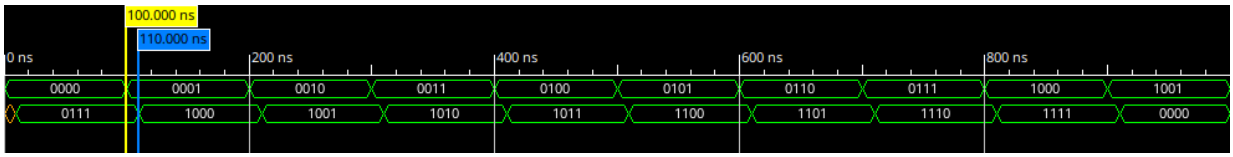
Pierwszym krokiem do przygotowania symulacji behawioralnej jest wygenerowanie pliku VHDL, który będzie opisywał zachowanie sygnałów wejściowych. Na Rys. 10 przedstawiono fragment wygenerowanego plik z wstawionym kodem, który definiuje zachowanie sygnałów wejściowych. Na Rys. 11 przedstawiono wynik symulacji behawioralnej. Na Rys. 12 przedstawiono output z symulacji czasowej. Czas propagacji sygnału wynosi 10 ns.

```
BEGIN

   UUT: Schematlab1 PORT MAP(
      X => X,
      Y => Y
   );

   X <= "0000",
      X"1" after 100 ns,
      X"2" after 200 ns,
      X"3" after 300 ns,
      X"4" after 400 ns,
      X"5" after 500 ns,
      X"6" after 600 ns,
      X"7" after 700 ns,
      X"8" after 800 ns,
      X"9" after 900 ns,
      "1010" after 1000 ns,
      "1011" after 1100 ns,
      "1100" after 1200 ns,
      "1101" after 1300 ns,
      "1110" after 1400 ns,
      "1111" after 1500 ns,
      X"0" after 1600 ns;
```

Rys. 10: Fragment listingu pliku VHDL opisującego zachowanie sygnałów



Rys. 11 Output z symulacji behawioralnej



Rys. 12: Output z symulacji behawioralnej, opóźnienie wynosi 10 ns.

1.5 **Implementacja**

Implementacja stworzonego układu logicznego na płycie testowej wymaga powiązania stworzonych magistral z rzeczywistymi elementami tej płyty. Odbywa się to za pośrednictwem pliku *.ucf. Gotowy plik uzyskano od prowadzącego laboratorium. Rys. 13 przedstawawia zmiany wprowadzone w pliku, polegające na powiązaniu magistrali X z przyciskami, zaś magistrali Y z diodami LED.

```
# Keys
NET "X<0>" LOC = "P42";
NET "X<1>" LOC = "P40";
NET "X<2>" LOC = "P43";
NET "X<3>" LOC = "P38";
#NET "Key<4>" LOC = "P37";
#NET "Key<5>" LOC = "P36";
#NET "Key<6>" LOC = "P24";
#NET "Key<7>" LOC = "P39";

# LEDS
NET "Y<0>"  LOC = "P35";
NET "Y<1>"  LOC = "P29";
NET "Y<2>"  LOC = "P33";
NET "Y<3>"  LOC = "P34";
#NET "LED<4>"  LOC = "P28";
#NET "LED<5>"  LOC = "P27";
#NET "LED<6>"  LOC = "P26";
#NET "LED<7>"  LOC = "P25";
```

*Rys. 13: Fragment pliku *.ucf definiującego powiązania magistrali z fizycznymi elemntami na płycie testowej.*

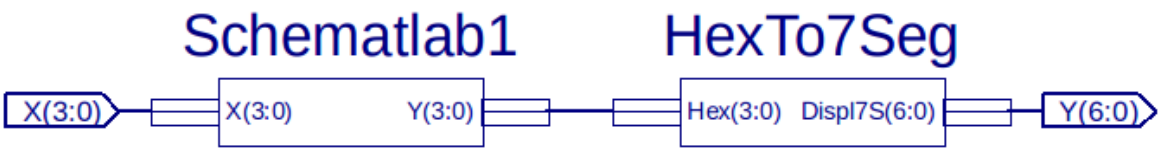
1.6 Programowanie układu, testowanie

Zgodnie z instrukcją dostarczoną przez prowadzącego wygenerowane pliki oraz zaprogramowano płytę ZL-9572, a następnie zadając kolejne kombinacje przetestowano układ. Układ działał odwrotnie niż powinien, a mianowicie diody zapalały się odwrotnie niż zakładano, było spowodowane tym, iż diody aktywowane są staniem niskim, zaś w trakcie projektowania układu, założono, że są one aktywowane stanem wysokim. Aby zmienić działanie układu, należałoby zanegować wyjście każdego układu logicznego reprezentującego funkcję wyjścia.

Przeprowadzone testy potwierdziły, że zaprojektowany układ działał poprawnie, co zostało sprawdzone przez prowadzącego.

2 Zadanie 2, stworzenie projektu hierarchicznego oraz wykorzystanie wyświetlacza segmentowego.

Zgodnie z instrukcją laboratoryjną z układ stworzony w zadaniu 1 przekształcono na submoduł i stworzono jego symbol schematowy. Rys. 13 przedstawia nowo powstały schemat. Rys 14 przedstawia fragment pliku *.ucf wiążący magistralę wyjściową z wyświetlaczem segmentowym.



Rys. 14: Schemat powstały w ramach zadania drugiego

```
# DISPL. 7-SEG
#NET "D7S_D<0>" LOC = "P8" | SLEW = "SLOW";
#NET "D7S_D<1>" LOC = "P6" | SLEW = "SLOW";
#NET "D7S_D<2>" LOC = "P4" | SLEW = "SLOW";
#NET "D7S_D<3>" LOC = "P9" | SLEW = "SLOW";
NET "Y<0>" LOC = "P12"; # Seg. A; shared with LED<10>
NET "Y<1>" LOC = "P13"; # Seg. B; shared with LED<8>
NET "Y<2>" LOC = "P22"; # Seg. C; shared with LED<12>
NET "Y<3>" LOC = "P19"; # Seg. D; shared with LED<14>
NET "Y<4>" LOC = "P14"; # Seg. E; shared with LED<15>
NET "Y<5>" LOC = "P11"; # Seg. F; shared with LED<9>
NET "Y<6>" LOC = "P20"; # Seg. G; shared with LED<13>
#NET "D7S_S<7>" LOC = "P18";  # Seg. DP;shared with LED<11>
```

*Rys. 15: Fragment pliku *.ucf odpowiadający za powiązanie magistraliu Y(3:0) z wyświetlaczem segmentowym*

Również w tym przypadku zaprogramowano płytę testową oraz sprawdzono fizyczne działanie układu. Układ działał poprawnie

3 Podsumowanie

W trakcie laboratorium zapoznano się z podstawami pracy w środowisku ISE 14.7 stworzono prosty układ kombinacyjny następnie przeprowadzono symulację behawioralną oraz czasową. W kolejnym kroku wygenerowano plik dla programatora. Programator poprzez JTAG zaprogramował płytę ZL-9572. Zwieńczeniem laboratorium było przeprowadzanie testów poprawności zaprojektowania, implementacji oraz zaprogramowania płyty. Całość działała poprawnie. Wszystkie zdania laboratoryjne zrealizowano poprawnie.