Raport z laboratorium nr 1
Układy kombinacyjne

Układy cyfrowe i systemy wbudowane 1

Autorzy:	Radosław Zimoch 263963, Kamil Gondek 263916
Prowadzący:	Dr. inż. Jarosław Sugier
Termin:	Poniedziałek, 11:00, Tydzień parzysty
Data:	27 października 2023

1 Wstęp

Podczas tych zajęć mieliśmy dwa kluczowe cele do osiągnięcia. Pierwszym z nich było zapoznanie się z narzędziami środowiska Xilinx ISE oraz ISim, aby zrozumieć ich funkcje i możliwości. Drugim celem było projektowanie układów kombinacyjnych. W ramach tych działań mieliśmy zadanie stworzenia 4-bitowego układu, który realizował matematyczne równanie Y = (2 - X) mod 16.

2 Zadania do wykonania

- 4-bit układ kombinacyjny
- Projekt hierarchiczny, Z1 jako podschemat + moduł HexTo7Seg

3 Realizacja zadań

3.1 Synteza układu Y=(2-X)mod16 - metoda siatek Karnaughta

Tabela 1: Tabela prawdy

x_3	x_2	x_1	x_0	y_3	y_2	y_1	y_0
0	0	0	0	0	0	1	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	0	0
0	0	1	1	1	1	1	1
0	1	0	0	1	1	1	0
0	1	0	1	1	1	0	1
0	1	1	0	1	1	0	0
0	1	1	1	1	0	1	1
1	0	0	0	1	0	1	0
1	0	0	1	1	0	0	1
1	0	1	0	1	0	0	0
1	0	1	1	0	1	1	1
1	1	0	0	0	1	1	0
1	1	0	1	0	1	0	1
1	1	1	0	0	1	0	0
1	1	1	1	0	0	1	1
1	0	0	0	1	0	1	0
1	0	0	1	1	0	0	1
1	0	1	0	1	0	0	0
1	0	1	1	0	1	1	1
1	1	0	0	0	1	1	0
1	1	0	1	0	1	0	1
1	1	1	0	0	1	0	0
1	1	1	1	0	0	1	1

Tabela 2: Siatka Karnaught dla y_0

$x3$x2 \ $x1$x0	00	01	11	10
00	0	1	1	0
01	0	1	1	0
11	0	1	1	0
10	0	1	1	0

Tabela 3: Siatka Karnaught dla y_1

$x3x2$ \ $x1x0$	00	01	11	10
00	1	0	1	0
01	1	0	1	0
11	1	0	1	0
10	1	0	1	0

Tabela 4: Siatka Karnaught dla y_2

$x3x2$ \ $x1x0$	00	01	11	10
00	0	0	1	0
01	1	1	0	1
11	1	1	0	1
10	0	0	1	0

Tabela 5: Siatka Karnaught dla y_3

$x3x2$ \ $x1x0$	00	01	11	10
00	0	0	1	0
01	1	1	1	1
11	0	0	0	0
10	1	1	0	1

Na podstawie powyższych siatek otrzymaliśmy następujące funkcje logiczne opisujące kolejne wyjścia:

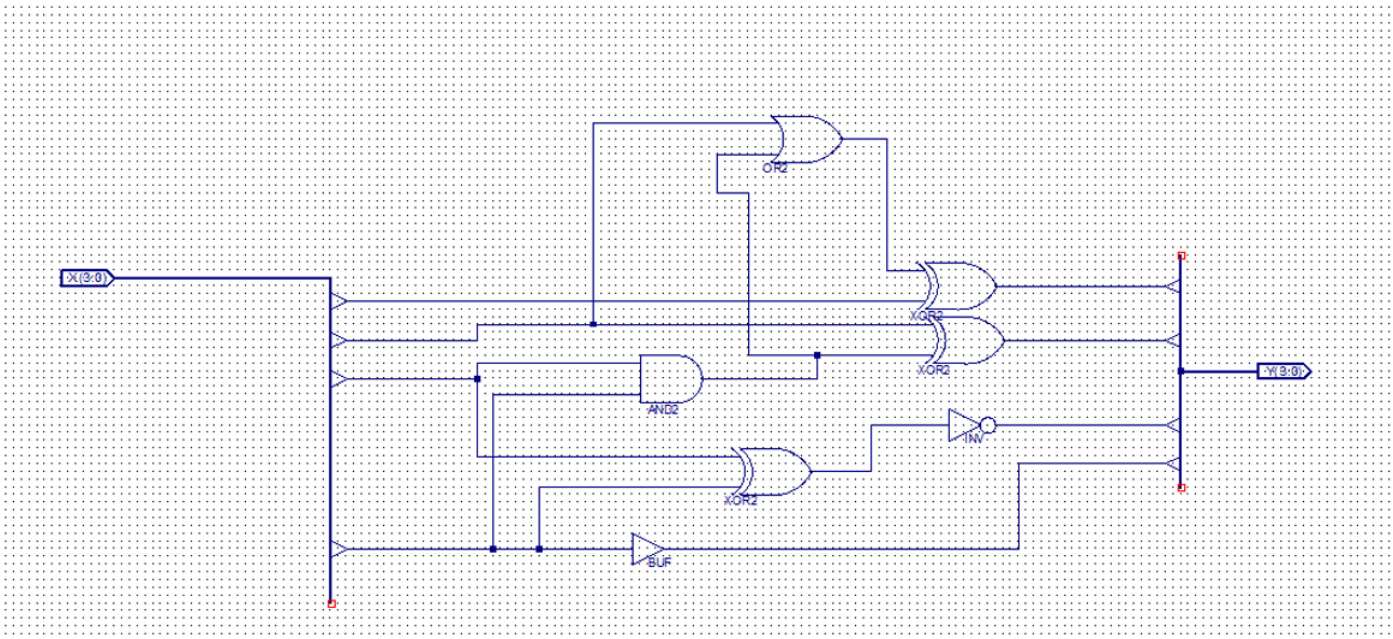
$$y_3 = \overline{x}_3 x_2 + x_3\overline{x_2 x_1} + x_3\overline{x_2 x_0} + \overline{x_3} x_1 x_0 = \overline{x}_3(x_2 + x_1 x_0) + x_3\overline{(x_2 + x_1 x_0)} = x_3 \oplus (x_2 + x_1 x_0)$$

$$y_2 = x_2(\overline{x}_1 + \overline{x}_0) + \overline{x}_2 x_1 x_0 = x_2 \oplus x_1 x_0$$

$$y_1 = \overline{x}_1 \cdot \overline{x}_0 + x_1 \cdot x_0 = \overline{x_1 \oplus x_0}$$

$$y_0 = x_0$$

Następnie zbudowaliśmy nasz układ za pomocą bramek.



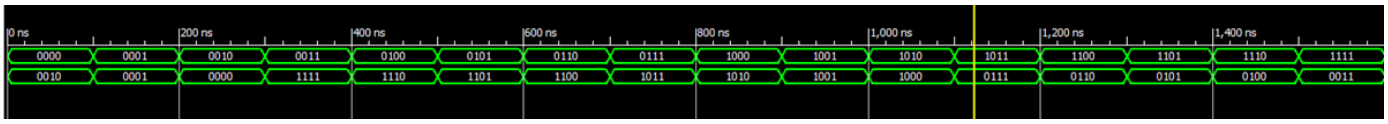
Rysunek 1: Schemat układu z wykorzystaniem bramek

W celu przetestowania stworzonego przez nas schematu wykorzystaliśmy symulację behawioralną. Modyfikując wygenerowany przez środowisko plik vhdl, na wejście podaliśmy wszystkie 16 kombinacji ze zmianami co 100ns.

```
1   LIBRARY ieee;
2   USE ieee.std_logic_1164.ALL;
3   USE ieee.numeric_std.ALL;
4   LIBRARY UNISIM;
5   USE UNISIM.Vcomponents.ALL;
6   ENTITY Z1_Z1_sch_tb IS
7   END Z1_Z1_sch_tb;
8   ARCHITECTURE behavioral OF Z1_Z1_sch_tb IS
9
10     COMPONENT Z1
11     PORT( X  :   IN  STD_LOGIC_VECTOR (3 DOWNTO 0);
12           Y :   OUT STD_LOGIC_VECTOR (3 DOWNTO 0));
13     END COMPONENT;
14
15     SIGNAL X :   STD_LOGIC_VECTOR (3 DOWNTO 0);
16     SIGNAL Y :   STD_LOGIC_VECTOR (3 DOWNTO 0);
17
18  BEGIN
19
20     UUT: Z1 PORT MAP(
21          X => X,
22          Y => Y
23     );
24
25      X <= "0000", "0001" after 100ns, "0010" after 200ns, "0011" after 300ns, "0100" after 400ns,
26     "0101" after 500ns, "0110" after 600ns, "0111" after 700ns, "1000" after 800ns, "1001" after 900ns,
27     "1010" after 1000ns, "1011" after 1100ns, "1100" after 1200ns, "1101" after 1300ns, "1110" after 1400ns,
28     "1111" after 1500ns;
29  END;
```

Rysunek 2: Kod do test benchu

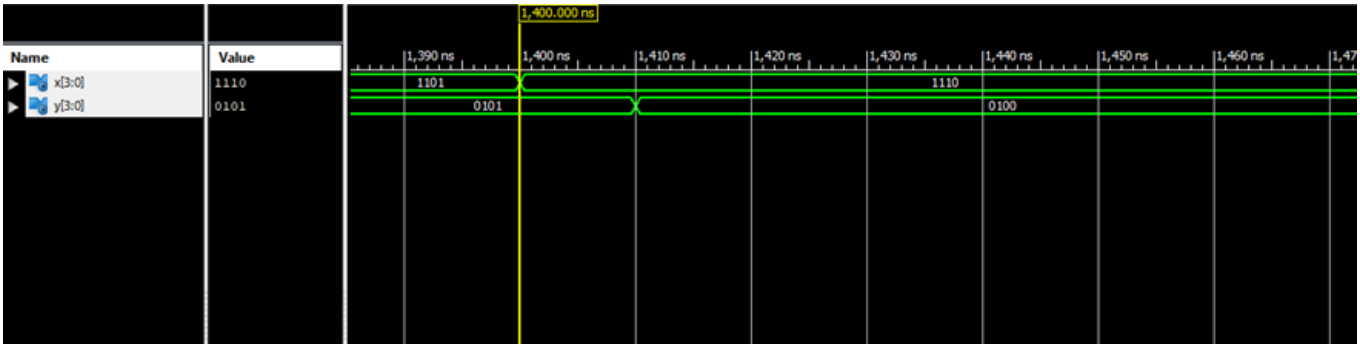
Symulacja behawioralna pokazała, że zaprojektowany schemat daje oczekiwanie wyniki.



Rysunek 3: Wynik symulacji behawioralnej

Rysunek 4: Poszczególne sygnały dla symulacji behawioralnej

Następnie w symulacji czasowej zbadaliśmy czas propagacji. Wynosił on 10ns.

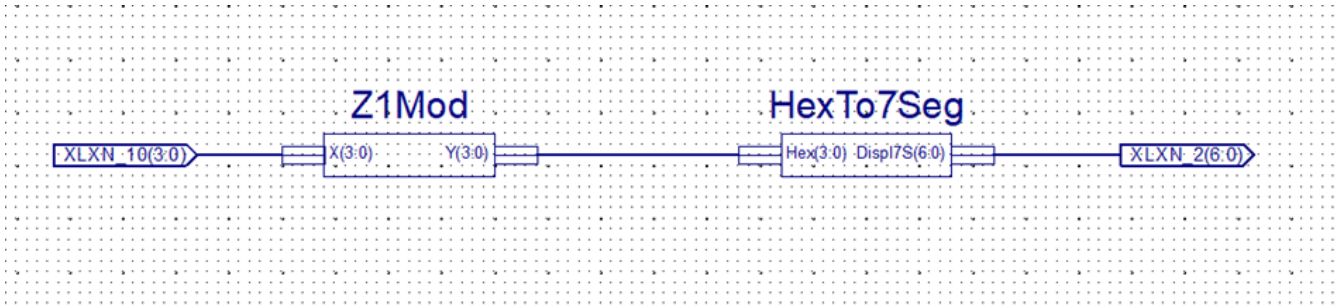


Rysunek 5: Wynik symulacji Post-Fit

Na koniec uruchomiliśmy zgodnie z instrukcją układ CPLD zgodnie z przygotowanym projektem i sprawdziliśmy jego poprawność na płycie laboratoryjnej. Wszystkie wyniki zgadzały się z oczekiwaniami

3.2 Projekt hierarchiczny

Następnym zadaniem było stworzenie z pierwszego układu podschematu i połączenie go z modułem HexTo7Seg pobranym ze wskazanego źródła oraz przeprowadzenie testów. Podczas zajęć udało nam się jedynie połączyć oba schematy i na tym zakończyliśmy pracę na laboratorium.



Rysunek 6: Połączone schematy

4 Wnioski

Rozpoczynając pracę z narzędziem Xilinx ISE, nabyliśmy umiejętności w projektowaniu i testowaniu układów logicznych. Pomimo początkowych trudności staliśmy się coraz bardziej biegli w obszarze tego środowiska. Układ został zaprojektowany i przetestowany w pełni poprawnie co zostało udowodnione podczas symulacji i testów.