Układy cyfrowe i systemy wbudowane

SPRAWOZDANIE NR 2

Układy sekwencyjne

PROWADZĄCY:

DR INŻ. JACEK MAZURKIEWICZ

AUTORZY:

MATEUSZ GRUSZKA 249448

BARTOSZ RUDNIK 248893

Grupa: C

TERMIN ZAJĘĆ:

WT 9:15 TP

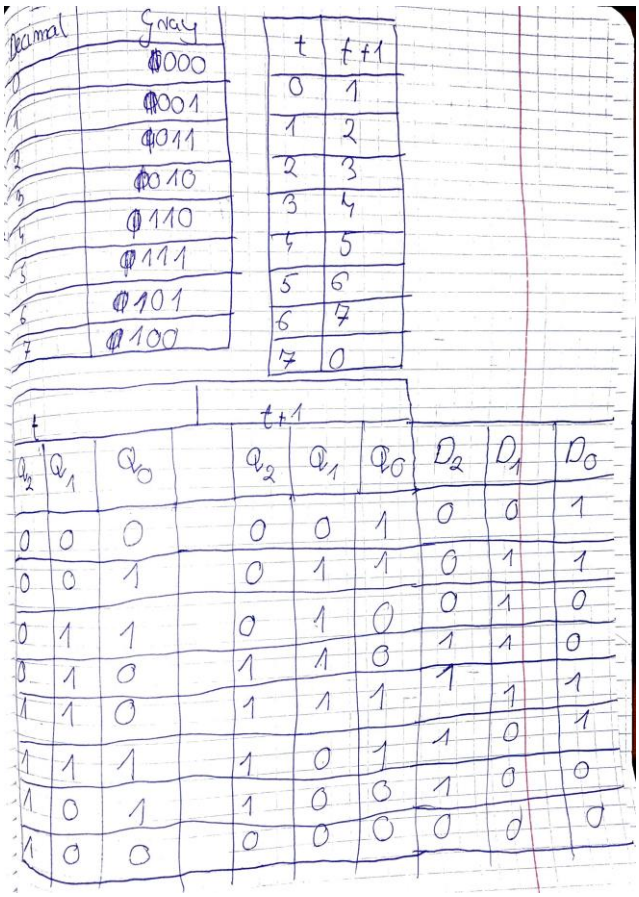
WROCŁAW, 10.11.2020 r.

1. Cel ćwiczenia

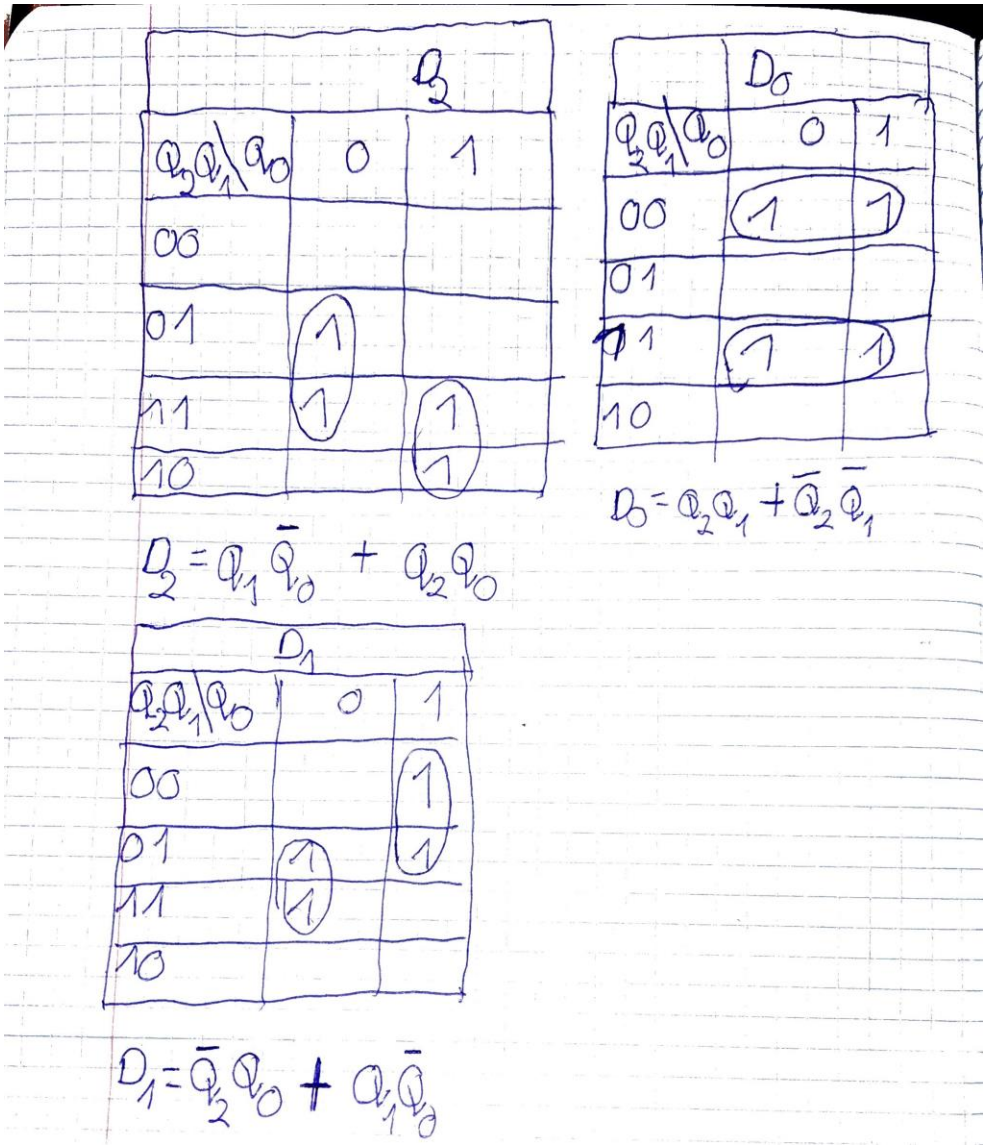
Celem ćwiczenia jest doskonalenie się w obsłudze zintegrowanego środowiska Xilinx ISE służącym do wykonywania wszystkich operacji związanych z opracowaniem projektu układu cyfrowego oraz jego implementacji w układzie CPLD lub FPGA. W tym celu na zajęciach mieliśmy zrealizować dwa układy: licznik synchroniczny oraz detektor sekwencji bitowej.

2. Licznik synchroniczny

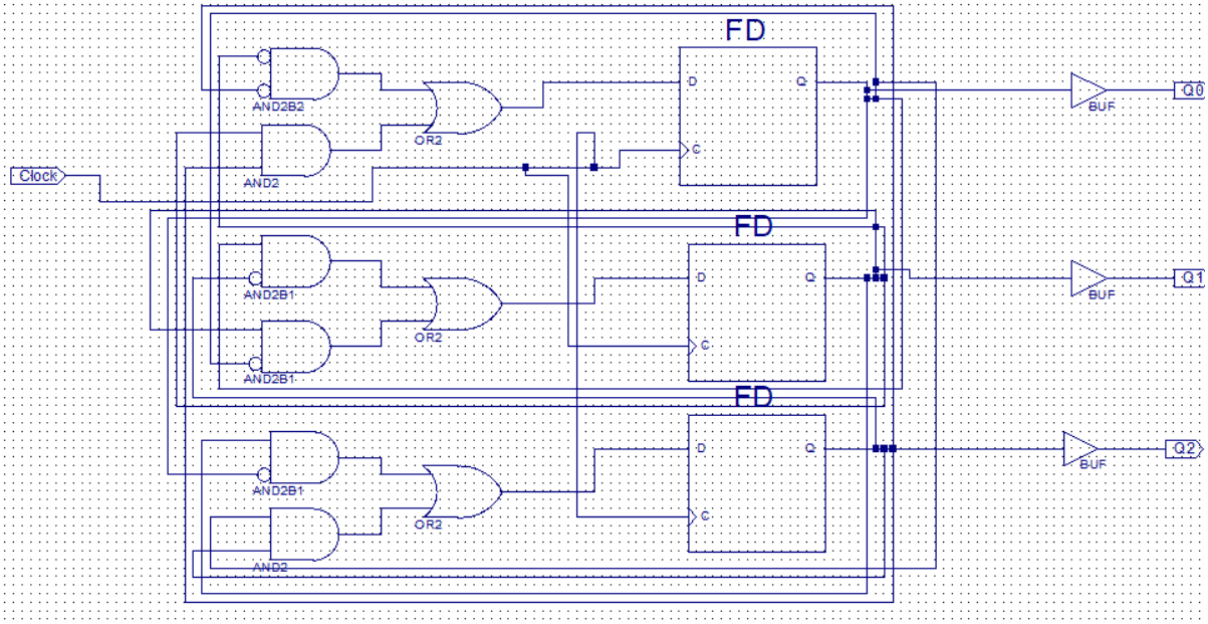
Pierwszym zadaniem było stworzenie licznika synchronicznego, mod7, pozytywnego w kodzie Gray'a. Licznik synchroniczny jest to licznik, który zmienia swój stan jednocześnie z taktem zegarowym. Mod7 oznacza, iż będzie to licznik podający kolejne cyfry od 0-7. Pozytywny jest słowem oznaczającym iż liczyć będziemy w górę. W momencie otrzymania cyfry 7 licznik przechodzi do cyfry 0. Natomiast kod Gray'a jest dwójkowym, bez wagowym, nie pozycyjnym kodem, którego dwa kolejne słowa kodowe różnią się wyłącznie stanem jednego bitu. Kod Gray'a jest także kodem cyklicznym ponieważ jego pierwszy i ostatni wyraz również różnią się wyłącznie stanem jednego bita.



Rysunek 1 tabela prawdy licznika synchronicznego



Rysunek 2 Minimalizacja metodą siatek Karnaugh'a przy wykorzystaniu przerzutników typu D

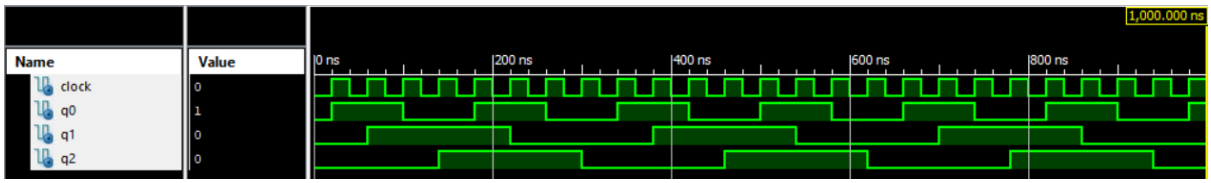


Rysunek 3 Schemat projektu w środowisku Xilinx ISE

Poniżej przedstawiony jest kod źródłowy testowego pliku VHDL. W pliku tym opisane są sygnały wejściowe oraz wyjściowe, a także określone są wartości jakie mają przyjąć sygnały wejściowe w określonym przez nas czasie.

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;
LIBRARY UNISIM;
USE UNISIM.Vcomponents.ALL;
ENTITY schematLicznik_schematLicznik_sch_tb IS
END schematLicznik_schematLicznik_sch_tb;
ARCHITECTURE behavioral OF schematLicznik_schematLicznik_sch_tb IS

   COMPONENT schematLicznik
   PORT( Clock     :     IN     STD_LOGIC;
         Q0     :     OUT     STD_LOGIC;
         Q1     :     OUT     STD_LOGIC;
         Q2     :     OUT     STD_LOGIC);
   END COMPONENT;

   SIGNAL Clock     :     STD_LOGIC := '0';
   SIGNAL Q0     :     STD_LOGIC;
   SIGNAL Q1     :     STD_LOGIC;
   SIGNAL Q2     :     STD_LOGIC;

BEGIN

   UUT: schematLicznik PORT MAP(
         Clock => Clock,
         Q0 => Q0,
         Q1 => Q1,
         Q2 => Q2
   );

Clock <= not Clock after 20 ns;

END;
```

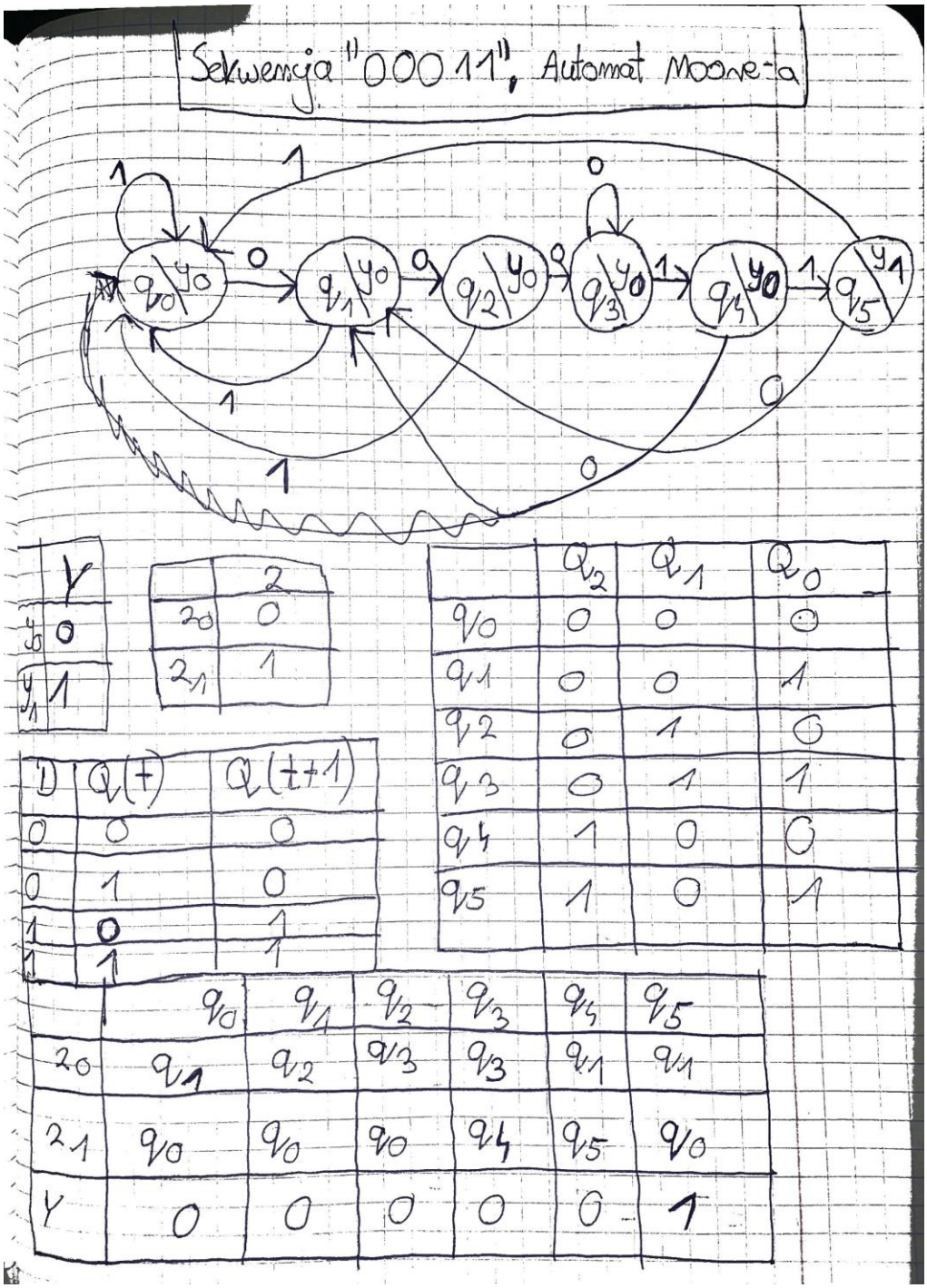
Na załączonym poniżej obrazku możemy zaobserwować wyniki symulacji przeprowadzone dla licznika synchronicznego. Poniższy rysunek jest zgodny z tabelą prawdy, co oznacza, że zadanie zostało wykonane poprawnie.



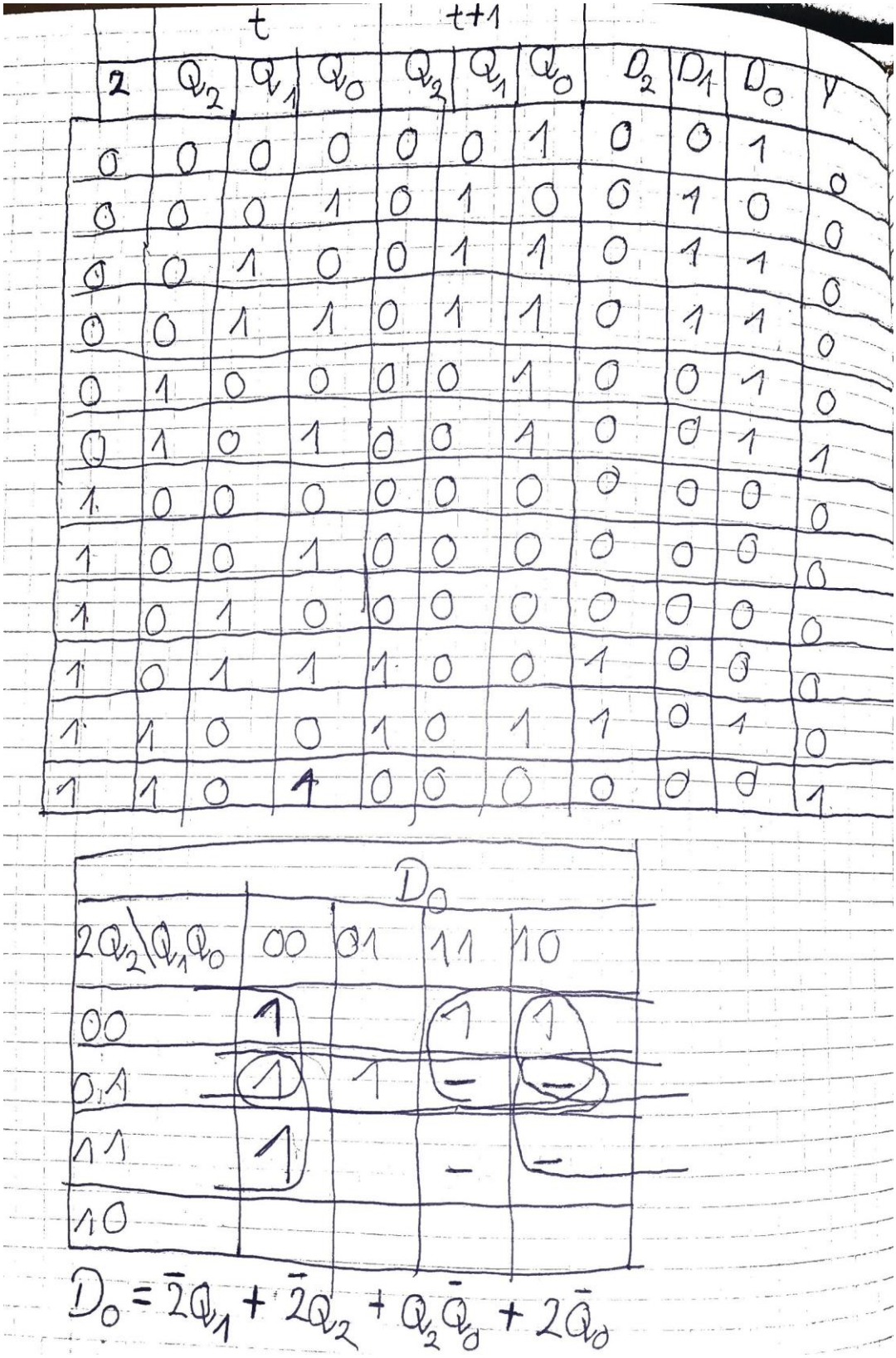
Rysunek 4 Wyniki działania symulacji dla licznika synchronicznego

3. Detektor sekwencji bitowej

Następnym zadaniem było stworzenie detektora sekwencji bitowej opisanego wskazanym typem automatu, posiadającym jedno wejście, jedno wyjście, brak resetu, z sekwencja prawidłową 5-bitową. W naszym przypadku jest to automat Moore'a odczytujący sekwencje "00011"



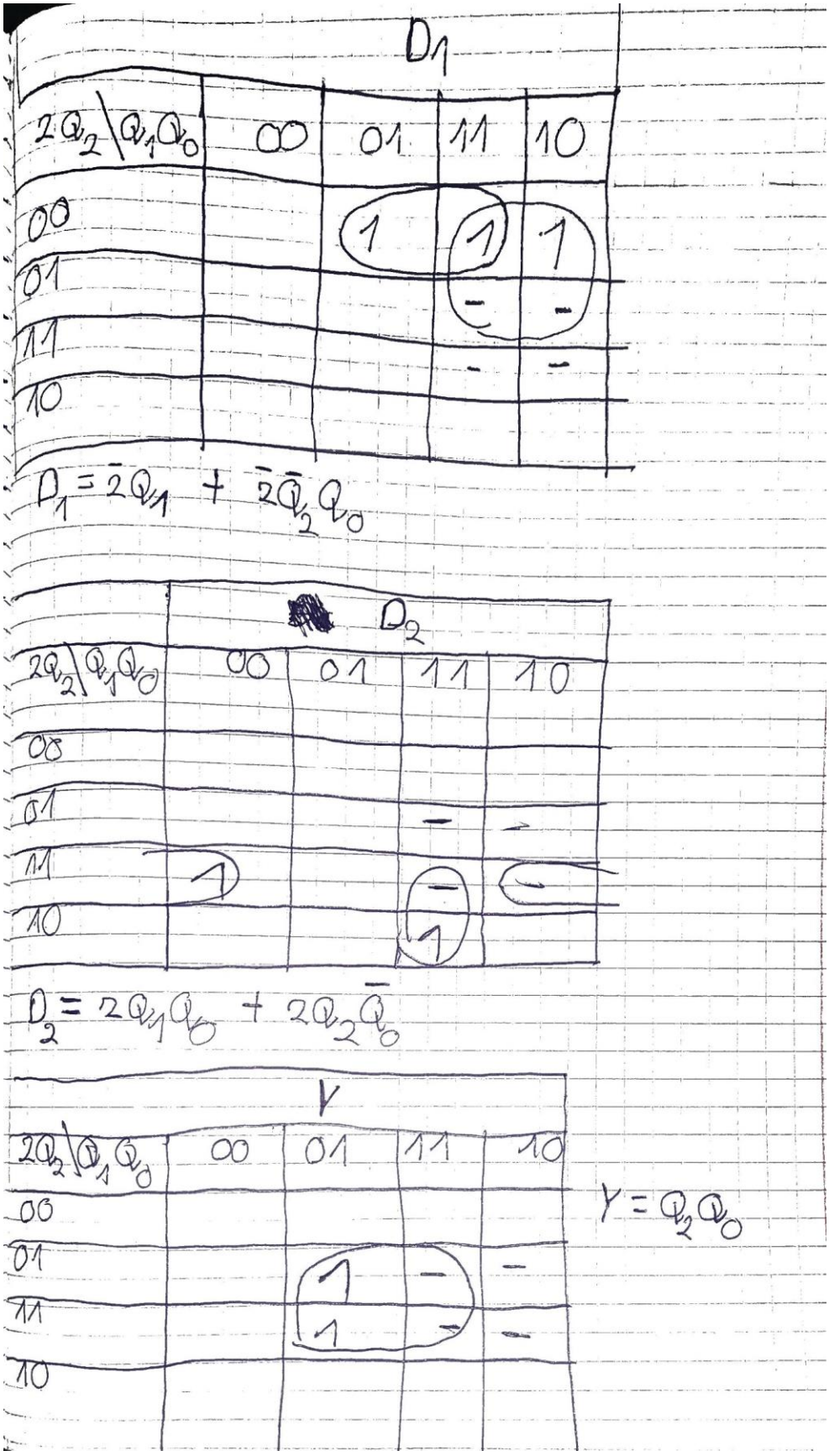
Rysunek 5 Tabela wejść, wyjść, stanów, reprezentacja graficzna automatu



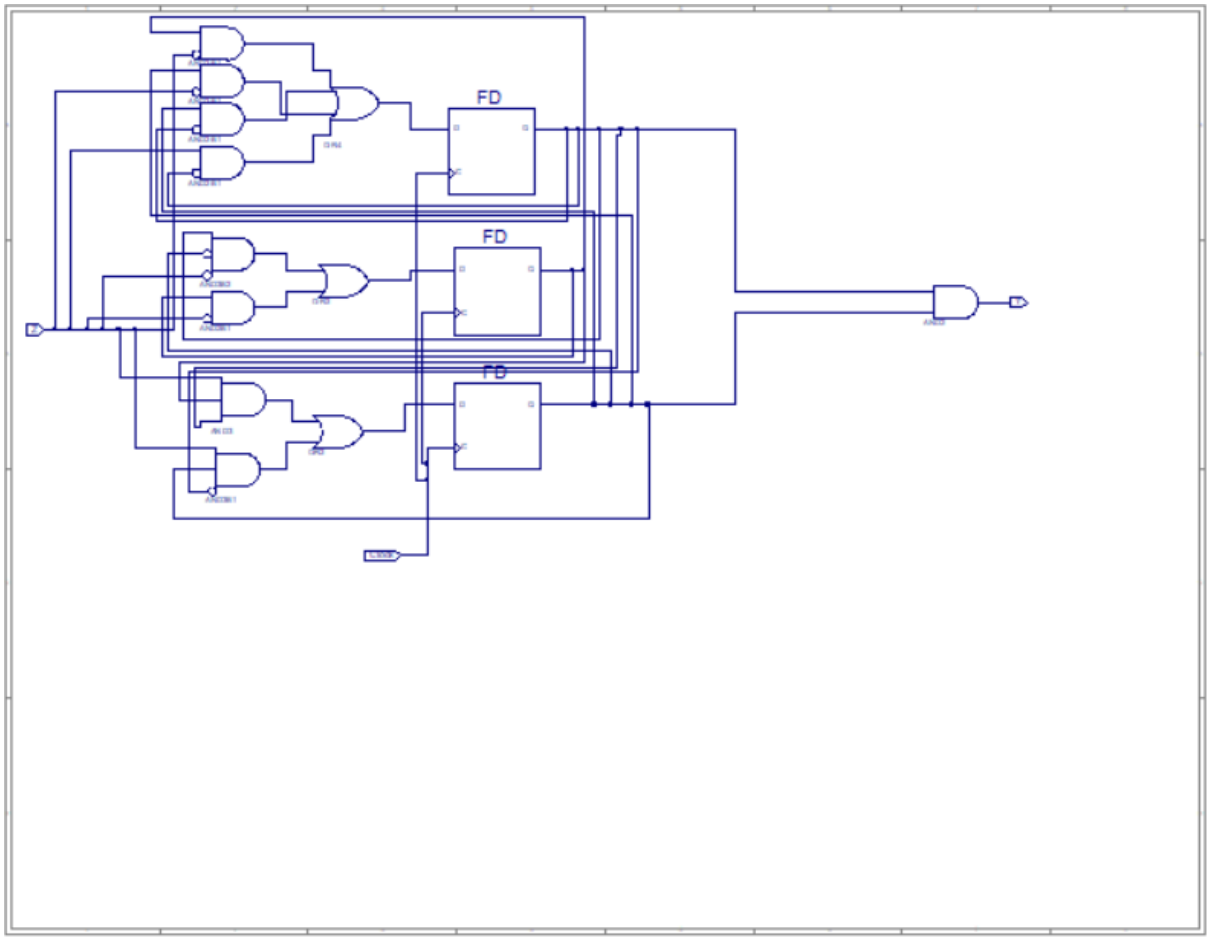
	t			t+1						
2	Q_2	Q_1	Q_0	Q_2	Q_1	Q_0	D_2	D_1	D_0	Y
0	0	0	0	0	0	1	0	0	1	0
0	0	0	1	0	1	0	0	1	0	0
0	0	1	0	0	1	1	0	1	1	0
0	0	1	1	0	1	1	0	1	1	0
0	1	0	0	0	0	1	0	0	1	0
0	1	0	1	0	0	1	0	0	1	1
1	0	0	0	0	0	0	0	0	0	0
1	0	0	1	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0	0	0
1	0	1	1	1	0	0	1	0	0	0
1	1	0	0	1	0	1	1	0	1	0
1	1	0	1	0	0	0	0	0	0	1

$$D_0 = \bar{2}Q_1 + \bar{2}Q_2 + Q_2\bar{Q}_0 + 2\bar{Q}_0$$

Rysunek 6 Tabela prawdy zadanego automatu



Rysunek 7 Minimalizacja metodą siatek Karnaugh'a



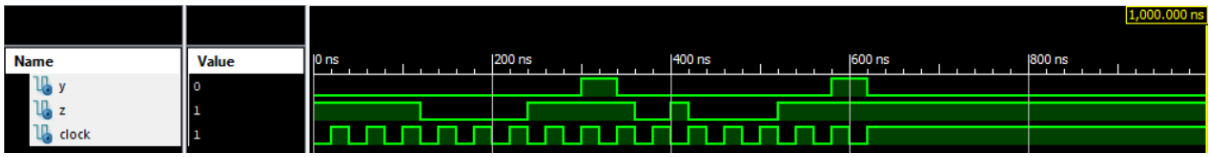
Rysunek 8 Schemat projektu w środowisku Xilinx ISE

W poniższym testowym pliku VHDL w celu przetestowanie w symulacji wszystkich możliwych kombinacji sygnałów wejściowych opisane zostały odpowiednie pobudzenia sygnałów.

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;
LIBRARY UNISIM;
USE UNISIM.Vcomponents.ALL;
ENTITY detektorSchemat_detektorSchemat_sch_tb IS
END detektorSchemat_detektorSchemat_sch_tb;
ARCHITECTURE behavioral OF detektorSchemat_detektorSchemat_sch_tb IS

   COMPONENT detektorSchemat
   PORT( Y    :    OUT    STD_LOGIC;
         Z    :    IN    STD_LOGIC;
         Clock    :    IN    STD_LOGIC);
   END COMPONENT;

   SIGNAL Y    :    STD_LOGIC;
   SIGNAL Z    :    STD_LOGIC;
   SIGNAL Clock    :    STD_LOGIC;

BEGIN

   UUT: detektorSchemat PORT MAP(
       Y => Y,
       Z => Z,
       Clock => Clock
   );

    Z <= '1', '0' after 120 ns, '1' after 240 ns, '0' after 360 ns, '1'
after 400 ns, '0' after 420 ns, '1' after 520 ns;

    Clock <= '0', '1' after 20 ns, '0' after 40 ns, '1' after 60 ns, '0'
after 80 ns, '1' after 100 ns, '0' after 120 ns, '1' after 140 ns, '0'
after 160 ns, '1' after 180 ns, '0' after 200 ns, '1' after 220 ns, '0'
after 240 ns, '1' after 260 ns, '0' after 280 ns, '1' after 300 ns, '0'
after 320 ns, '1' after 340 ns, '0' after 360 ns, '1' after 380 ns, '0'
after 400 ns, '1' after 420 ns, '0' after 440 ns, '1' after 460 ns, '0'
```

```
after 480 ns, '1' after 500 ns, '0' after 520 ns, '1' after 540 ns, '0'
after 560 ns, '1' after 580 ns, '0' after 600 ns, '1' after 620 ns;


END;
```

Na załączonym poniżej obrazku możemy zobaczyć wynik symulacji dla detektora sekwencji “00011”. Po ponownym zweryfikowaniu wyników z tabelą prawdy możemy upewnić się, że nasz system działa poprawnie.



Rysunek 9 Wyniki działania symulacji dla detektora sekwencji

4. Wnioski

Ćwiczenia te w dalszym stopniu oswajały nas ze środowiskiem Xilinx ISE oraz pozwalały obyć się z możliwościami, jakie daje nam ten program. Podczas zajęć wykonaliśmy dwa sprawnie działające układy. Przy pracach nad ćwiczeniem bardzo ważną rolę odegrała wiedza nabyta przez nas w trakcie 3 semestru na zajęciach z Logiki Układów Cyfrowych, wiedza ta pozwoliła nam sprawnie wykonać przedstawione zadania.