ГУАП

КАФЕДРА № 44

ОТЧЕТ
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

Доц., канд. техн. наук, доц.		О.О. Жаринов
должность, уч. степень, звание	подпись, дата	инициалы, фамилия

ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №2

Разработка преобразователей кодов на основе типовых функциональных узлов комбинационной логики с использованием языков описания аппаратуры

по курсу: СХЕМОТЕХНИКА

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ ГР. № 4142		К.С. Некрасов
	подпись, дата	инициалы, фамилия

Санкт-Петербург 2024

Цель работы

Изучить принципы работы типовых функциональных узлов комбинационной логики: шифраторов, дешифраторов, мультиплексоров. Разработать проект преобразователя кодов на их основе, с использованием языков описания аппаратуры. Основной целью работы является формирование навыков использования модулей на языке описания аппаратуры.

Индивидуальное задание. Вариант 3

Индивидуальное задание:

Состояния входных сигналов			3	
x2	x1	x0	y1	y0
0	0	0	1	1
0	0	1	0	0
0	1	0	1	1
0	1	1	0	1
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	0	1

Решение

С мультиплексором

Для разработки мультиплексора был создан модуль multiplexer, который получает селектор, выходы и возвращает подходящий селектору выход.

Мы используем этот модуль вызывая его с первым и вторым битом входного сигнала возвращая значение зависящее от нулевого бита входного сигнала и кладём результат в первый выходной бит. Аналогично делаем для нулевого выходного бита.

Листинг кода

```
module multiplexer(input [1:0] selector, input [3:0] outputs, output reg out);
  always @*
  begin
    case(selector)
      2'b00: out = outputs[0];
      2'b01: out = outputs[1];
      2'b10: out = outputs[2];
      2'b11: out = outputs[3];
      default: out = 1'b0;
```

```
    endcase
  end
endmodule

module second(input [2:0] x, output [1:0] y);
  multiplexer multiplexer1({x[2], x[1]}, {~x[0], ~x[0], x[0], ~x[0]}, y[1]);
  multiplexer multiplexer2({x[2], x[1]}, {~x[0], 1'b1, ~x[0], x[0]}, y[0]);
endmodule
```

Временная диаграмма

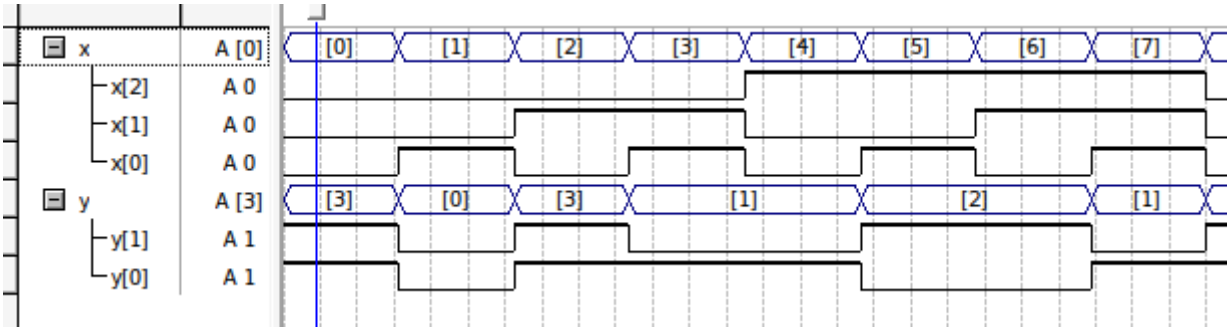

Рисунок 1 – Временная диаграмма

Схема подключения ПЛИС

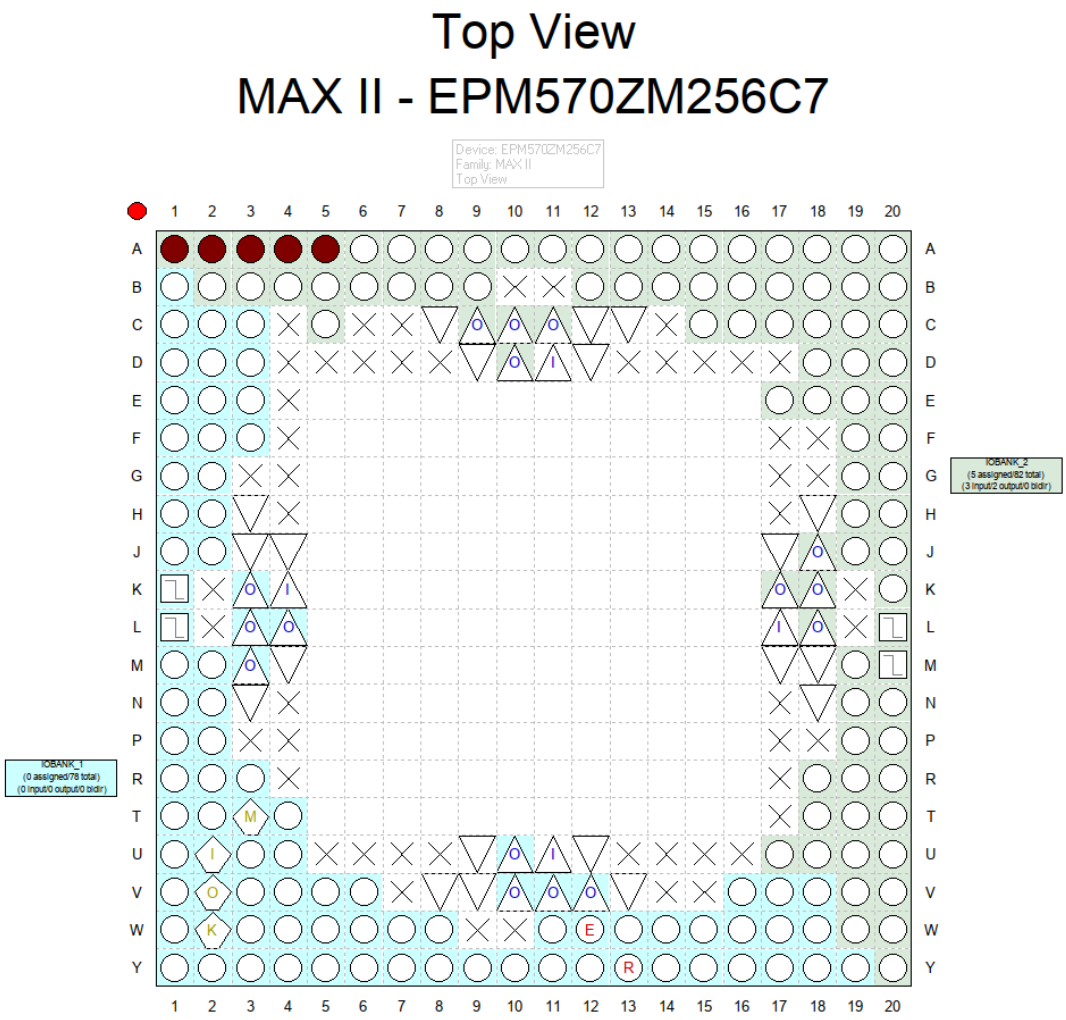

Рисунок 2 – Схема подключения ПЛИС

Шифратор Дешифратор

Дешифратор:

Входной вектор x преобразуется в выходной вектор decoder, используя оператор case. Каждое значение входного вектора соответствует определенному значению выходного вектора.

Шифратор:

Значения выходного вектора decoder используются для определения значений выходного вектора encoder, который является результатом логических операций над битами decoder.

Дешифрация:

Вектор encoder обратно преобразуется в выходной вектор y с помощью оператора case, который сопоставляет значения вектора encoder с соответствующими значениями выходного вектора y в соответствии с заданной логикой дешифрации.

Листинг кода

```
module second1(
 input wire [2:0] x,
 output reg [1:0] y
);
 reg [7:0] decoder;
 wire [3:0] encoder;
 always @*
 begin
  case(x)
   3'd0: decoder = 8'b00000001;
   3'd1: decoder = 8'b00000010;
   3'd2: decoder = 8'b00000100;
   3'd3: decoder = 8'b00001000;
   3'd4: decoder = 8'b00010000;
   3'd5: decoder = 8'b00100000;
   3'd6: decoder = 8'b01000000;
   3'd7: decoder = 8'b10000000;
  endcase
 end
 assign encoder[0] = decoder[1];
 assign encoder[1] = decoder[3] | decoder[4] | decoder[7];
 assign encoder[2] = decoder[6] | decoder[5];
 assign encoder[3] = decoder[0] | decoder[2];
 always @(decoder)
 begin
  case (encoder)
   4'b0001: y = 2'b00;
   4'b0010: y = 2'b01;
   4'b0100: y = 2'b10;
   4'b1000: y = 2'b11;
  endcase
 end
endmodule
```

Временная диаграмма

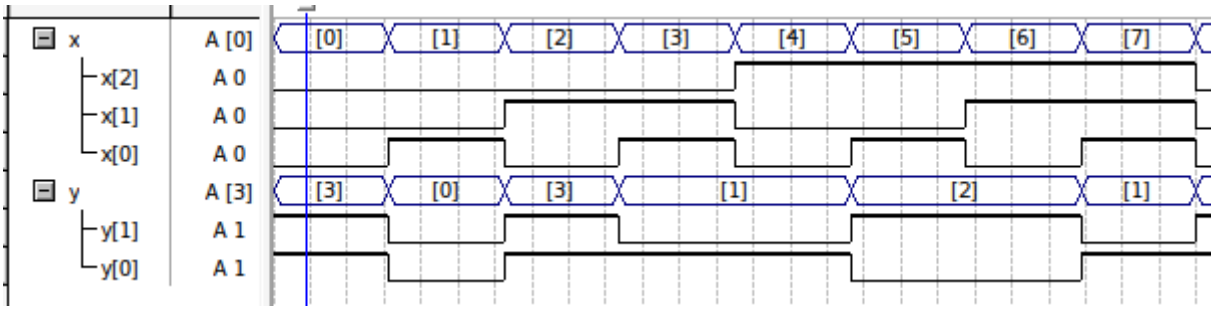

Рисунок 3 – Временная диаграмма

Схема подключения ПЛИС

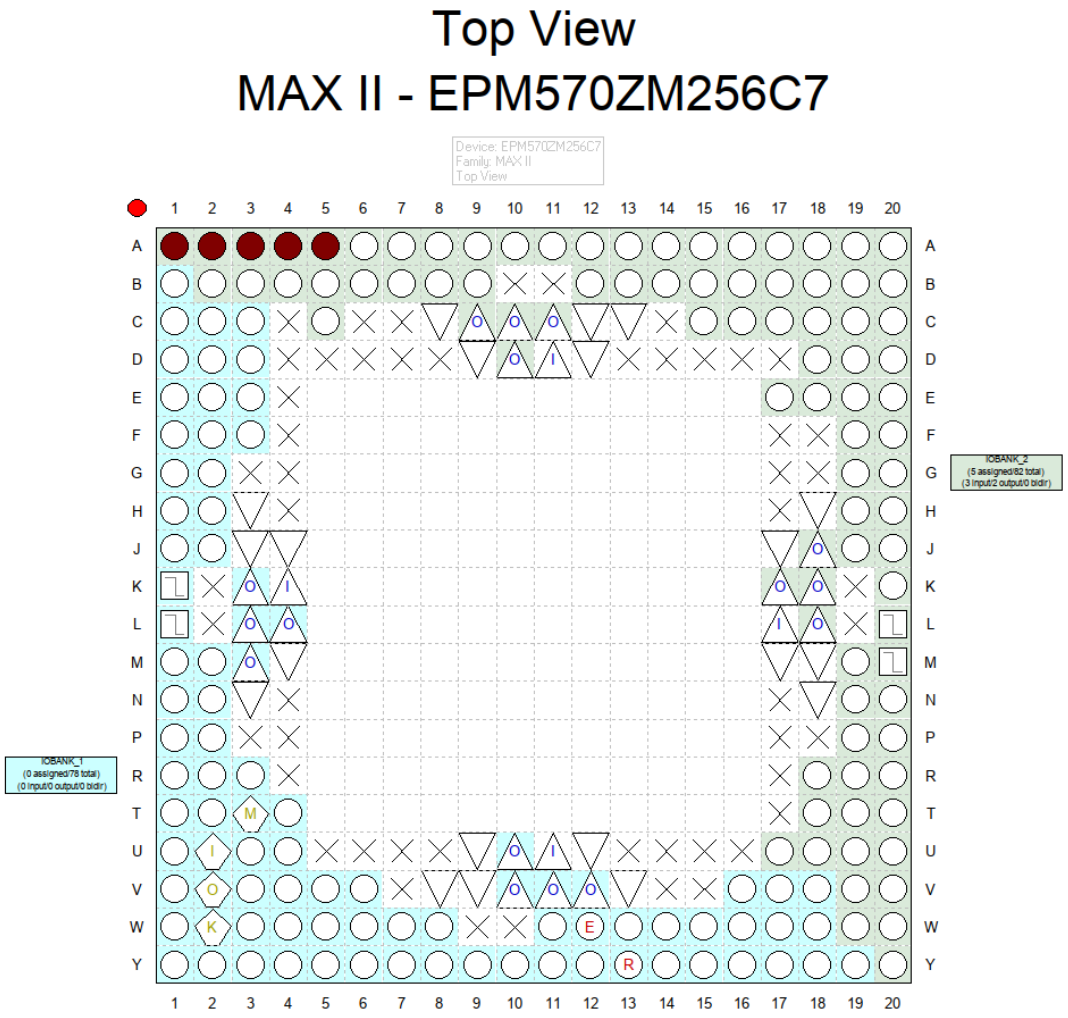

Рисунок 4 – Схема подключения ПЛИС

Вывод

Изучены принципы работы типовых функциональных узлов комбинационной логики: шифраторов, дешифраторов, мультиплексоров. Разработан проект преобразователя кодов на их основе, с использованием языков описания аппаратуры. Получены навыки использования модулей на языке описания аппаратуры Verilog.