САНКТ-ПЕТЕРБУРГСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

ПЕТРА ВЕЛИКОГО

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Институт компьютерных наук и технологий

Высшая школа интеллектуальных систем и суперкомпьютерных технологий

Отчет

по заданию lab6

Дисциплина

«Языки описания аппаратных средств вычислительных систем»

выполнила:

Фам Ба Нам

группа:

3530901/90201

преподаватель:

Федотов А. А.

Санкт-Петербург

2021

Оглавление

[1 Задание lab6\_1 4](#_Toc85579168)

[1.1 Задание 4](#_Toc85579169)

[1.2 Описание на языке Verilog 4](#_Toc85579172)

[1.3 Результат синтеза (RTL) 6](#_Toc85579173)

[1.4 Моделирование 6](#_Toc85579174)

[1.5 Тестирование на плате miniDiLaB- CIV 7](#_Toc85579175)

[1.6 Выводы 7](#_Toc85579176)

[2 Задание lab6\_2 8](#_Toc85579177)

[2.1 Задание 8](#_Toc85579178)

[2.2 Описание на языке Verilog 8](#_Toc85579179)

[2.3 Результат синтеза (RTL) 10](#_Toc85579180)

[2.4 Моделирование 10](#_Toc85579182)

[2.5 Тестирование на плате miniDiLaB-CIV 11](#_Toc85579184)

[2.6 Выводы 11](#_Toc85579185)

[3 Задание lab6\_3 12](#_Toc85579186)

[3.1 Задание 12](#_Toc85579187)

[3.2 Описание на языке Verilog 12](#_Toc85579188)

[3.3 Результат синтеза (RTL) 14](#_Toc85579189)

[3.4 Моделирование 14](#_Toc85579191)

[3.5 Тестирование на плате miniDiLaB-CIV 15](#_Toc85579193)

[3.6 Выводы 15](#_Toc85579194)

[4 Задание lab6\_4 12](#_Toc85579186)

[3.1 Задание 12](#_Toc85579187)

[3.2 Описание на языке Verilog 12](#_Toc85579188)

[3.3 Результат синтеза (RTL) 14](#_Toc85579189)

[3.4 Моделирование 14](#_Toc85579191)

[3.5 Тестирование на плате miniDiLaB-CIV 15](#_Toc85579193)

[3.6 Выводы 15](#_Toc85579194)

[4 Выводы 16](#_Toc85579195)

Список иллюстраций

[Рис. 1‑1 Описание на языке Verilog 4](#_Toc83927050)

[Рис. 1‑2 Синтезированная схема 4](#_Toc83927051)

[Рис. 1‑3 Результат моделирования средствами QII 6](#_Toc83927052)

[Рис. 1‑4 Обзор выводов в приложении Pin Planner 6](#_Toc83927053)

[Рис. 2‑1 Описание на языке Verilog 8](#_Toc83927054)

[Рис. 2‑2 Синтезированная схема 9](#_Toc83927055)

[Рис. 2‑3 Результат моделирования средствами QII 10](#_Toc83927056)

[Рис. 2‑4 Обзор выводов в приложении Pin Planner 11](#_Toc83927057)

[Рис. 3‑1 Описание на языке Verilog 12](#_Toc83927058)

[Рис. 3‑2 Синтезированная схема 12](#_Toc83927059)

[Рис. 3‑3 Результат моделирования средствами QII 14](#_Toc83927060)

[Рис.3‑4 Обзор выводов в приложении Pin Planner 15](#_Toc83927061)

# Задание lab6\_1

## Задание

На языке Verilog создайте:

Функцию, обеспечивающую

преобразование двоичного кода в код Грея и обратное преобразование кода Грея в

двоичный код:

Особенности:

– Разрядность преобразуемого кода задается

параметром WIDTH

– Направление преобразования задается

параметром DIR

– 0: двоичный код в код Грея

– 1: код Грея в двоичный

– необходимо использовать циклы

Модуль

параметризированный, параметр W -

разрядность преобразуемого кода, содержащий два экземпляра разработанной

функции:

– Один экземпляр функции преобразует входной

двоичный код в код Грея (разрядность задается

параметром W)

– Второй экземпляр функции преобразует код Грея в

выходной двоичный код (разрядность задается

параметром W)

## Описание на языке Verilog

|  |
| --- |
| module lab6\_1 (d\_in, to\_gray, to\_bin);  parameter W = 4;  (\*altera\_attribute = "-name IO\_STANDARD\"3.3-V LVCMOS\"", chip\_pin = "49, 46, 25, 24"\*)  input [W-1:0] d\_in;  (\*altera\_attribute = "-name IO\_STANDARD\"3.3-V LVCMOS\"", chip\_pin = "65, 66, 67, 68"\*)  output [W-1:0] to\_gray;  (\*altera\_attribute = "-name IO\_STANDARD\"3.3-V LVCMOS\"", chip\_pin = "69, 70, 71, 72"\*)  output [W-1:0] to\_bin;  function [W-1:0] convert;  input dir;  input [W-1:0] a;  reg [W-1:0] q;  reg [W-1:0] current;  integer i;  begin  if(dir)  for(i=0; i<=W-1; i=i+1)  q[i]=^(a>>i);  else  for(i=W-1; i>=0; i=i-1) begin  current = a>>i;  q[i]=current[0]^current[1];end  convert=q;  end  endfunction  assign to\_gray = convert(0,d\_in);  assign to\_bin = convert(1,to\_gray);  endmodule |

Рис.1.2 Описание на языке SystemVerilog

## Результат синтеза (RTL)

Diagram, schematic

Description automatically generated

Рис.1.3 Синтезированная схема

## Моделирование

A picture containing shape

Description automatically generated

Рис.1.4 Результаты моделирования

## Тестирование на плате miniDiLaB- CIV

Для тестирования проекта на плате использовались тесты, описанные в разделе 2.4

Проведенное на плате miniDiLaB-CIV тестирование разработанного устройства показало: результаты совпадают с ожидаемыми, устройство работает в соответствии с заданием.

## Выводы

В ходе выполнения работы был спроектировано устройство, выполняющее преобразование двоичного кода в код Грея и обратное преобразование кода Грея в двоичный код. Выводы СБИС были назначены с использованием атрибутов.

# Задание lab6\_2

## Задание

Выводы устройства:

Входы:

Переключатель sw[7:6] – операнд A

Переключатель sw[5:4] – операнд B

Переключатель sw[3:2] – операнд C

Переключатель sw[1:0] – операнд D

Clk – тактовый вход

Rst (кнопка pbA) – вход синхронного сброса

Выходы – операнды, отсортированные по убыванию или по возрастанию

светодиоды led[7:6]

светодиоды led[5:4]

светодиоды led[3:2]

светодиоды led[1:0]Описание на языке Verilog

|  |
| --- |
| module lab6\_2 (  (\*altera\_attribute = "-name IO\_STANDARD \"3.3-V LVCMOS\"",  chip\_pin = "25, 24"\*) input reg [1:0] a,  (\*altera\_attribute = "-name IO\_STANDARD \"3.3-V LVCMOS\"",  chip\_pin = "49, 46"\*) input reg [1:0] b,  (\*altera\_attribute = "-name IO\_STANDARD \"3.3-V LVCMOS\"",  chip\_pin = "90, 91"\*) input reg [1:0] c,  (\*altera\_attribute = "-name IO\_STANDARD \"3.3-V LVCMOS\"",  chip\_pin = "88, 89"\*) input reg [1:0] d,  (\*altera\_attribute = "-name IO\_STANDARD \"2.5 V\"",  chip\_pin = "71, 72"\*) output reg [1:0] max,  (\*altera\_attribute = "-name IO\_STANDARD \"2.5 V\"",  chip\_pin = "69, 70"\*) output reg [1:0] max\_min,  (\*altera\_attribute = "-name IO\_STANDARD \"2.5 V\"",  chip\_pin = "67, 68"\*) output reg [1:0] min\_max,  (\*altera\_attribute = "-name IO\_STANDARD \"2.5 V\"",  chip\_pin = "65, 66"\*) output reg [1:0] min,  (\*altera\_attribute = "-name IO\_STANDARD \"3.3-V LVCMOS\"",  chip\_pin = "23"\*) input clk,  (\*altera\_attribute = "-name IO\_STANDARD \"2.5 V\"",  chip\_pin = "58"\*) input pba);  reg [1:0] min1, min2, max1, max2, mid1, mid2;  task sort;  output reg [1:0] maximum, minimum;  input reg [1:0] a, b;  begin  maximum <= (a > b) ? a : b;  minimum <= (a > b) ? b : a;  end  endtask  always @ (posedge clk, negedge pba) begin  if (!pba) begin  max <= 2'b0;  max\_min <= 2'b0;  min\_max <= 2'b0;  min <= 2'b0;  end else begin  sort(max1, min1, a, b);  sort(max2, min2, c, d);  sort(max, mid1, max1, max2);  sort(mid2, min, min1, min2);  sort(max\_min, min\_max, mid1, mid2);  end  end  endmodule |

Рис.2.2 Описание на языке SystemVerilog

## Результат синтеза (RTL)

## Diagram Description automatically generated

Рис. 2.3 Синтезированная схема

## Моделирование

## Diagram Description automatically generated

Рис.2.4 Результаты моделирования

## Тестирование на плате miniDiLaB-CIV

Для тестирования проекта на плате использовались тесты, описанные в разделе 2.4

Проведенное на плате miniDiLaB-CIV тестирование разработанного устройства показало: результаты совпадают с ожидаемыми, устройство работает в соответствии с заданием.

## Выводы

В ходе выполнения работы было спроектировано устройство, сортирующее входные числа по возрастанию. Выводы СБИС были назначены с использованием атрибутов.

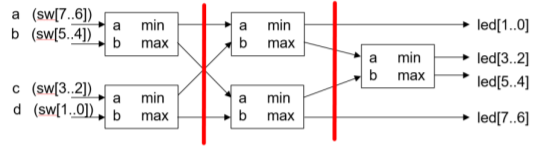
# Задание lab6\_3

## Задание

Осуществите конвейеризацию устройства

из части Lab6\_2

На рисунке показано место для регистров

конвейеризации. 

Проведите исследование:

– Для исходной (lab6\_2) реализации зафиксируйте

– с помощью RTL Viewer структуру.

– Аппаратные затраты

– Максимальную тактовую частоту

– Для конвейерной реализации зафиксируйте

результат

– с помощью RTL Viewer структуру.

– Аппаратные затраты

– Максимальную тактовую частоту

– Осуществите сравнение

Для конвейерной реализации осуществите

моделирование.

## Описание на языке Verilog

|  |
| --- |
| module lab6\_3 (  (\*altera\_attribute = "-name IO\_STANDARD \"3.3-V LVCMOS\"",  chip\_pin = "25, 24"\*) input reg [1:0] a,  (\*altera\_attribute = "-name IO\_STANDARD \"3.3-V LVCMOS\"",  chip\_pin = "49, 46"\*) input reg [1:0] b,  (\*altera\_attribute = "-name IO\_STANDARD \"3.3-V LVCMOS\"",  chip\_pin = "90, 91"\*) input reg [1:0] c,  (\*altera\_attribute = "-name IO\_STANDARD \"3.3-V LVCMOS\"",  chip\_pin = "88, 89"\*) input reg [1:0] d,  (\*altera\_attribute = "-name IO\_STANDARD \"2.5 V\"",  chip\_pin = "71, 72"\*) output reg [1:0] max,  (\*altera\_attribute = "-name IO\_STANDARD \"2.5 V\"",  chip\_pin = "69, 70"\*) output reg [1:0] max\_min,  (\*altera\_attribute = "-name IO\_STANDARD \"2.5 V\"",  chip\_pin = "67, 68"\*) output reg [1:0] min\_max,  (\*altera\_attribute = "-name IO\_STANDARD \"2.5 V\"",  chip\_pin = "65, 66"\*) output reg [1:0] min,  (\*altera\_attribute = "-name IO\_STANDARD \"3.3-V LVCMOS\"",  chip\_pin = "23"\*) input clk,  (\*altera\_attribute = "-name IO\_STANDARD \"2.5 V\"",  chip\_pin = "58"\*) input pba);  reg [1:0] min1, min2, max1, max2, mid1, mid2, minRg, maxRg;  task sort;  output reg [1:0] maximum, minimum;  input reg [1:0] a, b;  begin  maximum <= (a > b) ? a : b;  minimum <= (a > b) ? b : a;  end  endtask  always @ (posedge clk, negedge pba) begin  if (!pba) begin  max <= 2'b0;  max\_min <= 2'b0;  min\_max <= 2'b0;  min <= 2'b0;  min1 <= 2'b0;  min2 <= 2'b0;  max1 <= 2'b0;  max2 <= 2'b0;  mid1 <= 2'b0;  mid2 <= 2'b0;  minRg <= 2'b0;  maxRg <= 2'b0;  end else begin  sort(max1, min1, a, b);  sort(max2, min2, c, d);  sort(maxRg, mid1, max1, max2);  sort(mid2, minRg, min1, min2);  sort(max\_min, min\_max, mid1, mid2);  max <= maxRg;  min <= minRg;  end  end  endmodule |

Рис.2.1 Описание на SystemVerilog

## Результат синтеза (RTL)

## 

Рис. 3.3 Синтезированная схема

## Моделирование

## 

Рис.3.4 Результаты моделирования

## Тестирование на плате miniDiLaB-CIV

Для тестирования проекта на плате использовались тесты, описанные в разделе 3.4

Проведенное на плате miniDiLaB-CIV тестирование разработанного устройства показало: результаты совпадают с ожидаемыми, устройство работает в соответствии с заданием.

## Выводы

В ходе выполнения работы была осуществлена конвейеризация устройства сортировки входных чисел по возрастанию. СБИС были назначены с использованием атрибутов.

# Задание lab6\_4

## Задание

На языке Verilog введите представленный

ниже текст

Изображение выглядит как текст

Автоматически созданное описание

Осуществите анализ и синтез, посмотрите:

- Предупреждения в процессоре сообщений и проверить наличие предупреждений о триггерах защелках (Latch)

-синтезированную пакетом Q схему (RTL Viewer), найти триггеры защелки

- Оцените аппаратные затраты (запомните кол-во

использованных логических элементов)

- Исправьте описание так, чтобы не нарушая логику работы исключить возможность появления триггеров-защелок.

- Повторите анализ и синтез, убедитесь, что

триггеры-защелки отсутствуют (нет предупреждений в процессоре сообщений;

результаты синтеза в RTL Viewer)

- Оцените аппаратные затраты (сравните с результатами, полученными ранее)

Изображение выглядит как текст

Автоматически созданное описание

Рис. 3.3 Предупреждения

Изображение выглядит как текст

Автоматически созданное описание

Рис. 3.3 Аппаратурные затраты

Теперь изменим схему.

## Описание на языке Verilog

|  |
| --- |
| module lab6\_4 (sel, x, y, z, q1, q2, q3);  input [2:0] sel;  input x, y, z;  output reg q1, q2, q3;  always @\* begin  q1 = 1'b0;  q2 = 1'b0;  q3 = 1'b0;  if (sel == 3'h1) q1 = x;  if (sel == 3'h2) q2 = y;  if (sel == 3'h3) q3 = z;    end  endmodule |

Рис.2.1 Описание на SystemVerilog

Приведенные выше предупреждения были устранены.

Изображение выглядит как текст

Автоматически созданное описание

Рис. 3.3 Аппаратурные затраты

## Результат синтеза (RTL)

## 

Рис. 3.3 Синтезированная схема

## Выводы

В ходе выполнения работы были проанализированы предупреждения в процессоре сообщений о триггерах защелках (Latch), оценены аппаратные затраты. Схема была изменна таким образом, чтобы уменьшить аппаратные затраты и избавиться от предупреждений.

# Выводы

В ходе выполнения работы было спроектировано устройство, выполняющее преобразование двоичного кода в код Грея и обратное преобразование кода Грея в двоичный код, устройство сортировки по возрастанию, также была проведена его конвейеризация. Кроме того, был видоизменен исходный код, в результате чего удалось избавиться от предупреждений и уменьшить аппаратные затраты.