

|  |
| --- |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **"МИРЭА** - **Российский технологический университет"**  **РТУ МИРЭА** |

**Институт** Информационных Технологий

**Кафедра** Вычислительной Техники

**ИТОГОВЫЙ ОТЧЁТ**

**по дисциплине**

**«Архитектура ВМиС»**

Студент группы: **ИКБО-36-22** Утенков Ю. Ю. *(Фамилия студента)*

Преподаватель \_\_\_Гололобов А.А.\_\_\_

*(Фамилия преподавателя)*

Москва 2023

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 3](#_Toc153032867)

[ПРАКТИЧЕСКАЯ РАБОТА №1 4](#_Toc153032868)

[ПРАКТИЧЕСКАЯ РАБОТА №2 7](#_Toc153032869)

[ПРАКТИЧЕСКАЯ РАБОТА №3 10](#_Toc153032870)

[ПРАКТИЧЕСКАЯ РАБОТА №4 15](#_Toc153032871)

[ПРАКТИЧЕСКАЯ РАБОТА №5 20](#_Toc153032872)

[ПРАКТИЧЕСКАЯ РАБОТА №6 22](#_Toc153032873)

[ПРАКТИЧЕСКАЯ РАБОТА №7 27](#_Toc153032874)

[ПРАКТИЧЕСКАЯ РАБОТА №8 31](#_Toc153032875)

[ЗАКЛЮЧЕНИЕ 37](#_Toc153032876)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 38](#_Toc153032877)

# ВВЕДЕНИЕ

Широкое внедрение автоматики во все сферы человеческой деятельности, наблюдаемое в настоящее время, предъявляет жесткие требования к изделиям электронной техники. Это связано, с одной стороны, с возрастанием важности и сложности решаемых задач, а с другой стороны, необходимостью улучшения качественных характеристик, таких как: быстродействие, надежность, потребляемая мощность, габариты, стоимость и др. Одним из путей решения данной проблемы является использование новой элементной базы - программируемых логических интегральных схем (ПЛИС - Programmable Logic Device - PLD).

ПЛИС представляют собой интегральные схемы, обладающие

гибкостью заказных БИС (больших интегральных схем) и доступностью традиционной "жесткой" логики. По существу, разработка устройств на основе ПЛИС представляет собой новую технологию проектирования

электронных схем, включая их изготовление и сопровождение.

CAПP QUARTUS II представляет собой интегрированную среду для

разработки цифровых устройств на базе программируемых логических

интегральных схем (ПЛИС) фирмы ALTERA и обеспечивает выполнение

всех этапов, необходимых для выпуска готовых изделий:

• создание проектов устройств;

• синтез структур и трассировку внутренних связей ПЛИС;

• подготовку данных для программирования или конфигурирования ПЛИС (компиляцию):

• верификацию проектов (функциональное моделирование временной анализ):

• программирование или конфигурирование ПЛИС.

# ПРАКТИЧЕСКАЯ РАБОТА №1

## 1 Постановка задачи

1. Изучить правила построения, принцип работы логических схем.
2. Синтезировать электрическую принципиальную схему логического устройства, описанного заданным преподавателем уравнением в алгебраической форме.
3. Нарисовать синтезированную схему в графическом редакторе САПР QUARTUS II.
4. Произвести симуляцию работы схемы. Зарисовать диаграммы работы и по ее результатам заполнить таблицу истинности смоделированной схемы.
5. Ответить на контрольные вопросы, оформить отчет о выполненной работе.

Вариант 5: *Y =* (A+D)~B\*C

## 2 Теоретическое введение

Схемы, реализующие логические функции, называются логическими элементами. Основные логические элементы имеют, как правило, один выход (Y) и несколько входов, число которых равно числу аргументов. На электрических схемах логические элементы рисуют в виде прямоугольников с выводами для входных (слева) и выходных (справа) переменных. В средине прямоугольника изображается символ, обозначающий функциональное назначение элемента.

Для синтезирования электрической принципиальной схемы логического устройства, описанного уравнением в алгебраической форме, были использованы логические элементы AND, OR, XOR.

Таблица истинности для функции *Y =* (A+D)~B\*C:

*Таблица 1 – Таблица истинности*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | C | D | Y |
| 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 0 |
| 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 |

## 3 Логическая схема и временная диаграмма

На Рисунке 1 и Рисунке 2 представлены логическая схема и временная диаграмма, выполненные для соответствующего варианта.

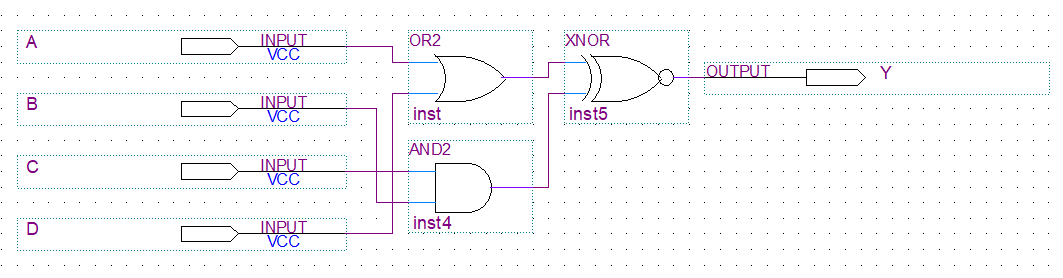


Рисунок 1 – Логическая схема

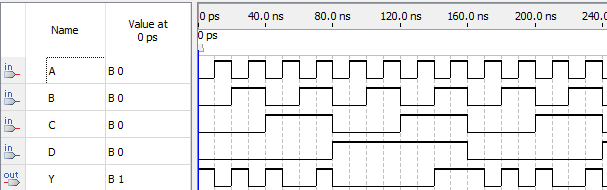


Рисунок 2 – Временная диаграмма

# 

# ПРАКТИЧЕСКАЯ РАБОТА №2

## 1 Постановка задачи

1. Изучить основные элементы языка AHDL и правила описания логических схем.
2. Сделать описание электрической схемы, заданной в предыдущей работе при помощи текстового редактора САПР QUARTUS II.
3. Произвести симуляцию работы схемы. Зарисовать диаграммы работы и по ее результатам заполнить таблицу истинности смоделированной схемы.
4. Сравнить результаты, полученные в ходе выполнения лабораторной работы с результатами, полученными в работе №1.
5. Ответить на контрольные вопросы, оформить отчет о выполненной работе.

Задание варианта 5:*Y =* (A+D)~B\*C

## 2 Теоретическое введение

Язык описания аппаратуры AHDL разработан фирмой Altera и предназначен для описания комбинационных и последовательностных логических устройств, групповых операций, цифровых автоматов (state machine) и таблиц истинности с учетом архитектурных особенностей ПЛИС фирмы Altera. Он полностью интегрируется с системой автоматизированного проектирования ПЛИС QUARTUS II. Файлы описания аппаратуры, написанные на языке AHDL, имеют расширение \* TDF (Text design file). Для создания DF файла можно использовать как текстовый редактор системы QUARTUS II, так и любой другой. Проект, выполненный в виде TDF-файла, компилируется, отлаживается и используется для формирования файла программирования или загрузки ПЛИС фирмы Altera.

Для описания цифровой схемы с помощью языка описания аппаратуры AHDL (текстового редактора CAII OUARTUS II), были использованы логические элементы AND (&), OR (#), XOR ($). SUBDESIGN - это имена, которые пользователь определил для файлов проекта более низкого уровня. Для подключения сигнала A, B, C, D нужно добавить в данный файл проекта строку "A, B, C, D: INPUT;" в секцию SUBDESIGN. И для получения вывода мы должны добавить в данный файл строку “U : OUTPUT;”, в секцию SUBDESIGN.

Таблица истинности для функции *Y =* (A+D)~B\*C:

*Таблица 2 – Таблица истинности*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | C | D | Y |
| 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 0 |
| 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 |

## 3 Код и временная диаграмма

На Рисунке 2 и Рисунке 3 представлены AHDL-код и временная диаграмма, выполненные для соответствующего варианта.

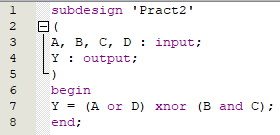


Рисунок 3 – AHDL-Код

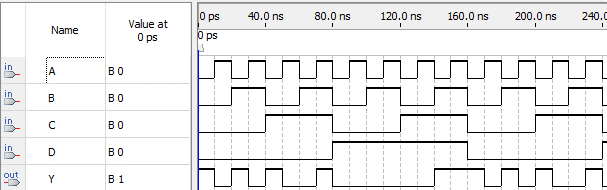


Рисунок 4 – Временная диаграмма

## 4 Сравнение результатов

На Рисунке 5 и Рисунке 6 представлены две временные диаграммы, первая диаграмма построенной логической схеме, а вторая - AHDL-коду.

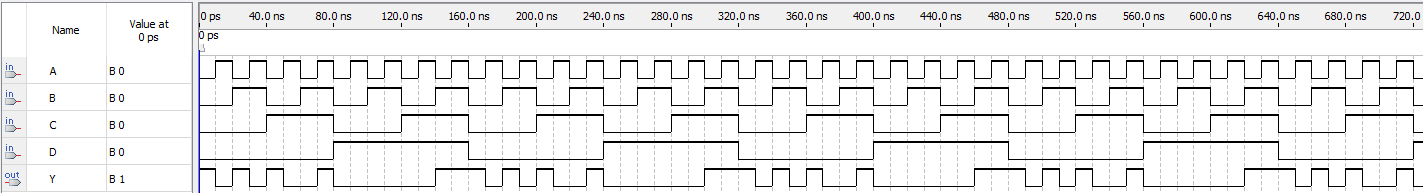


Рисунок 5 – Временная диаграмма практической работы №1

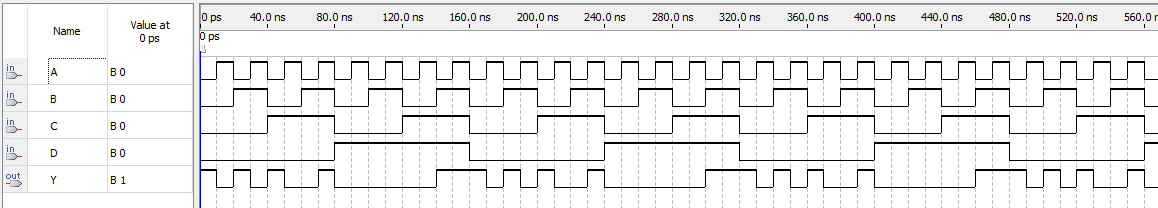


Рисунок 6 – Временная диаграмма практической работы №2

Сравнение показало, что реализация заданного логического уравнения в алгебраической форме принципиальной логической схемой и реализация AHDL-кодом дают одинаковые результаты (временные диаграммы).

# ПРАКТИЧЕСКАЯ РАБОТА №3

## 1 Постановка задачи

1. Изучить правила построения и принцип работы комбинационных схем, и построение их логических схем.
2. Нарисовать электрическую схему по указанию преподавателя при помощи графического редактора САПР QUARTUS II.
3. Произвести симуляцию работы схемы. Зарисовать диаграммы работы и по ее результатам заполнить таблицу истинности смоделированной схемы.
4. Ответить на контрольные вопросы, оформить отчет о выполненной работе.

Задание варианта: DMUX 2-4

## 2 Теоретическое введение

В комбинационных схемах (КС), называемых также автоматами без памяти, совокупность выходных сигналов (выходное слово Y) в дискретный момент времени t однозначно определяется входными сигналами (входным словом Х), поступившим на входы в тот же дискретный момент времени.

Реализуемый в этих схемах способ обработки информации называется комбинационным, Т.К. результат обработки информации зависит от комбинации входных сигналов и вырабатывается сразу после подачи на входы входной информации.

Примерами комбинационных схем являются: шифратор, дешифратор, мультиплексор, демультиплексор, сумматор, компаратор.

Демультиплексор 2-4 выполняет функцию по сути обычного дешифратора но с информационным входом, т.е можно сказать, что в зависимости от адресных входов, сигнал D передаётся на определённый выход демультиплексора.

Работа демультиплексора основана на комбинационном способе обработки информации, где выходы формируются на основе логических элементов, каждый выход соответствует номеру адресных входов непосредственно. В данном случае, демультиплексор 2-4 имеет два адресных входа (двухразрядный сигнал управления), 1 информационный и 4 однобитовых выходов, выбор между которыми осуществляется в соответствии с комбинацией сигналов управления.

Примером комбинационной схемы является демультиплексор 2-4, который, подобно другим устройствам этого типа, таким как шифратор, дешифратор, сумматор и компаратор, выполняет операцию без учета предыдущего состояния.

Таким образом, демультиплексор 2-4 представляет собой устройство, способное выбирать один из восьми входных сигналов на основе трех сигналов управления, и его принципиальная схема реализована с использованием логических элементов, таких как логические элементы AND.

Таблица истинности DMUX 2-4:

*Таблица 3 - Таблица истинности*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| A1 | A0 | E | Y0 | Y1 | Y2 | Y3 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | D | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | D | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 | 0 | D | 0 |
| 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 0 | 0 | 0 | D |

**3 Логическая схема и временная диаграмма**

На Рисунке 7 и Рисунке 8 представлены логическая схема и временная диаграмма, выполненные для соответствующего варианта (DMUX 2-4).

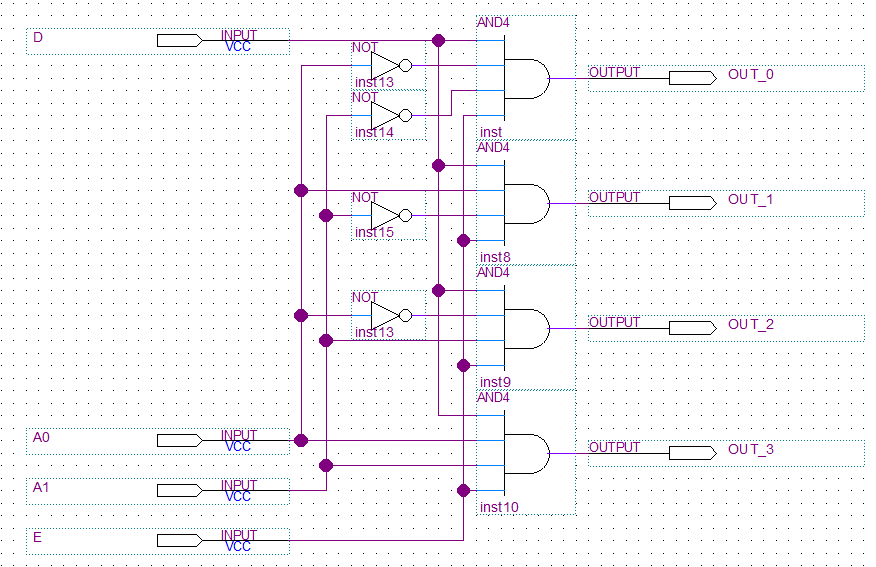


Рисунок 7 – Логическая схема

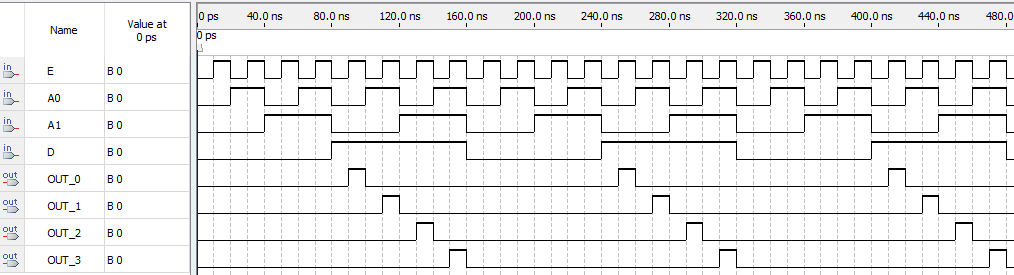


Рисунок 8 – Временная диаграмма

# ПРАКТИЧЕСКАЯ РАБОТА №4

## 1 Постановка задачи

1. Изучить основные элементы языка AHDL и правила описания логических схем.
2. Сделать описание электрической комбинационной схемы, заданной в предыдущей работе при помощи текстового редактора САПР QUARTUS II.
3. Произвести симуляцию работы схемы. Зарисовать диаграммы работы и по ее результатам заполнить таблицу истинности смоделированной схемы.
4. Сравнить результаты, полученные в ходе выполнения лабораторной работы с результатами, полученными в работе №3.
5. Ответить на контрольные вопросы, оформить отчет о выполненной работе.

Задание варианта: DMUX 2-4

## 2 Теоретическое введение

DMUX 2-4 - устройство, имеющее 2 адресных входа, 1 информационный, и 4 выхода, осуществляет пропуск сигнала D на определённый выход данной комбинационной схемы. Можно сказать, что демультиплексор преобразует параллельный код в последовательный. Для описания цифровой схемы DMUX 2-4 с помощью языка описания аппаратуры AHDL (текстового редактора САПР QUARTUS II) были использованы логические элементы AND (&), NOT (!) и OR (#).

Таблица истинности DMUX 2-4:

*Таблица 4 - Таблица истинности*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| A1 | A0 | E | Y0 | Y1 | Y2 | Y3 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | D | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | D | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 | 0 | D | 0 |
| 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 0 | 0 | 0 | D |

## 3 Код и временная диаграмма

На Рисунке 9 и Рисунке 10 представлены AHDL-код и временная диаграмма, выполненные для соответствующего варианта.

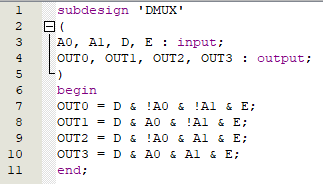


Рисунок 9 – AHDL-Код

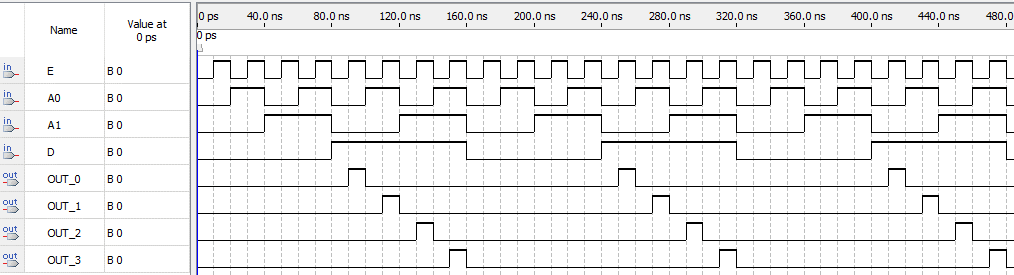


Рисунок 10 – Временная диаграмма

## 4 Сравнение результатов

На Рисунке 11 и Рисунке 12 представлены две временные диаграммы, первая диаграмма построенной логической схеме, а вторая - AHDL-коду.

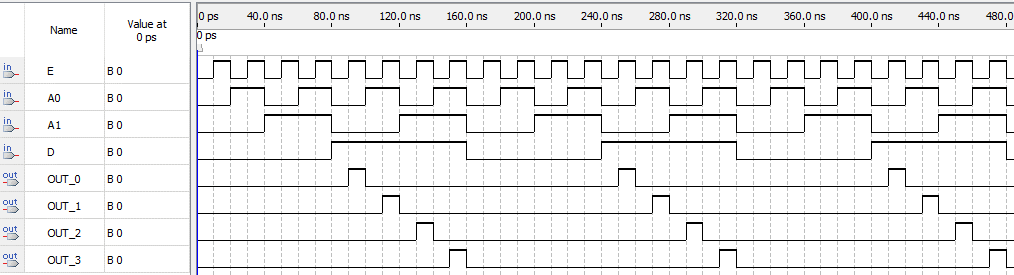


Рисунок 11 – Временная диаграмма практической работы №3

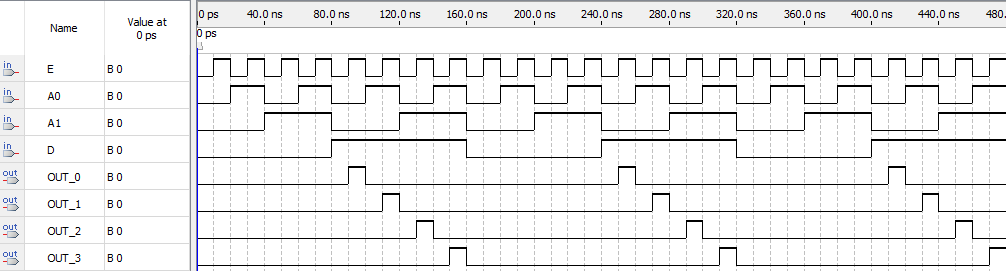


Рисунок 12 – Временная диаграмма практической работы №4

Сравнение показало, что реализация DMUX 2-4 логической схемой и реализация AHDL-кодом дают одинаковые результаты (временные диаграммы) при одинаковых значениях на входах.

# ПРАКТИЧЕСКАЯ РАБОТА №5

## Постановка задачи

1. Изучить правила построения и принцип работы компараторов, и построение их логических схем.
2. Нарисовать электрическую схему по указанию преподавателя при помощи графического редактора САПР QUARTUS II.
3. Произвести симуляцию работы схемы. Зарисовать диаграммы работы и по ее результатам заполнить таблицу истинности смоделированной схемы.
4. Ответить на контрольные вопросы, оформить отчет о выполненной работе.

Задание варианта: 3xcompare A>=B

## 2 Теоретическое введение

Компаратор - это электронное устройство, выполняющее арифметическое сравнение двух многоразрядных кодов А и В и сообщает является ли A>B, А=В или А<В. Компараторы относятся к арифметическим устройствам и, как правило, входят в состав арифметико-логических устройств.

На соответствующем выходе компаратора появляется логическая 1, когда сравниваемые входные двухразрядные числа находятся в нужном соотношении. Можно реализовывать многоразрядные компараторы, выдающие выходной сигнал одной из функций сравнения или всех функций сравнения.

3xcompare A>=B - это трёхразрядный компаратор, осуществляющий сравнение двух трёхразрядных кодов на условие "A равно B" и "A больше B". При выполнении данного сравнения, выход логического элемента "равно" подает логическую единицу, если трёхразрядный код A равен трёхразрядному коду B, аналогично для выхода "больше". Выходы всех элементов "равно" сводятся на входы логического элемента "И". На выходе логического элемента "И" логическая единица генерируется, указывая на истинность условия "A равно B" для всех разрядов.

Таблица истинности одноразрядного компаратора:

*Таблица 5 – Таблица истинности*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | A=B | A>B | A<B |
| 0 | 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 1 | 1 | 0 | 0 |

Для синтезирования электрической принципиальной схемы компаратора были использованы логические элементы AND, XNOR.

**3 Логическая схема и временная диаграмма**

На Рисунке 13 и Рисунке 14 представлены логическая схема и временная диаграмма, выполненные для соответствующего варианта (3xcompare A>=B).

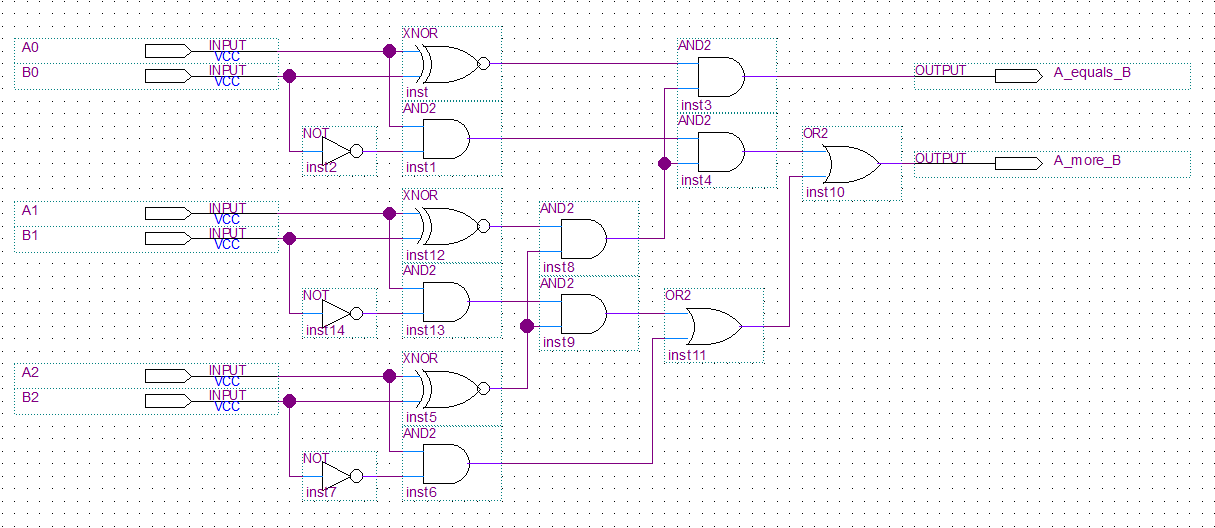
**

Рисунок 13 – Логическая схема

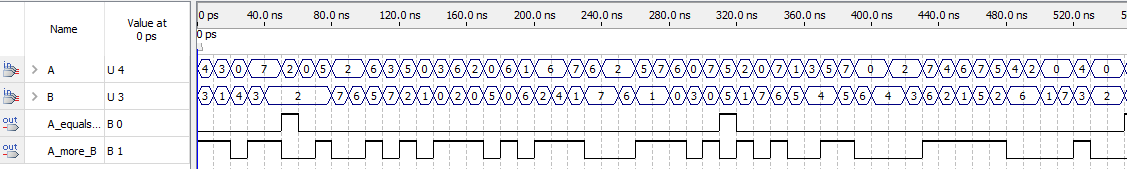
**

Рисунок 14 – Временная диаграмма

# ПРАКТИЧЕСКАЯ РАБОТА №6

## Постановка задачи

1. Согласно своему варианту графа состояний автомата разработать функциональную электрическую схему цифрового программируемого устройства преобразования кодов.
2. Включить ЭВМ и запустить CAIP QUARTUS II.
3. Создать проект, ввести разработанную схему, откомпилировать и смоделировать её.
4. Проверить полученные результаты, сверив их с таблицей истинности устройства.

Задание варианта 27:

*Таблица 6 – Варианты состояний исходного графа*

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 6 | 7 | 8 | 14 | 2 | 9 | 1 | 12 | 11 | 3 | 0 | 5 | 15 | 10 | 13 | 4 |

## 2 Теоретическое введение

Устройство, называемое счетчиком, предназначено для подсчета числа поступающих на вход сигналов (импульсов) в произвольной системе счисления. Двоичные счетчики строятся на основе триггеров, работающих в счетном режиме (Т - триггер или счетный триггер).

Счетный триггер может быть получен из универсального D – триггера путем соединения его инверсного выхода Q со входом D. У счетного триггера состояние выхода изменяется на противоположное при поступлении на вход С каждого очередного счетного импульса.

Необходимо построить логическую схему счётчика с произвольным модулем счёта согласно таблице состояний варианта. Построим таблицу перекодировки состояний автомата, граф по этой таблицы и таблицу истинности автомата.

*Таблица 7 – Таблица перекодировки состояний автомата и их двоичный код*

|  |  |  |
| --- | --- | --- |
| № состояния | № состояния из табл.1 | Двоичный код  q3,q2,q1,q0 |
| 0 | 6 | 0110 |
| 1 | 7 | 0111 |
| 2 | 8 | 1000 |
| 3 | 14 | 1110 |
| 4 | 2 | 0010 |
| 5 | 9 | 1001 |
| 6 | 1 | 0001 |
| 7 | 12 | 1100 |
| 8 | 11 | 1011 |
| 9 | 3 | 0011 |
| 10 | 0 | 0000 |
| 11 | 5 | 0101 |
| 12 | 15 | 1111 |
| 13 | 10 | 1010 |
| 14 | 13 | 1101 |
| 15 | 4 | 0100 |

На Рисунке 15 представлен граф, полученный с учетом таблицы перекодировки.

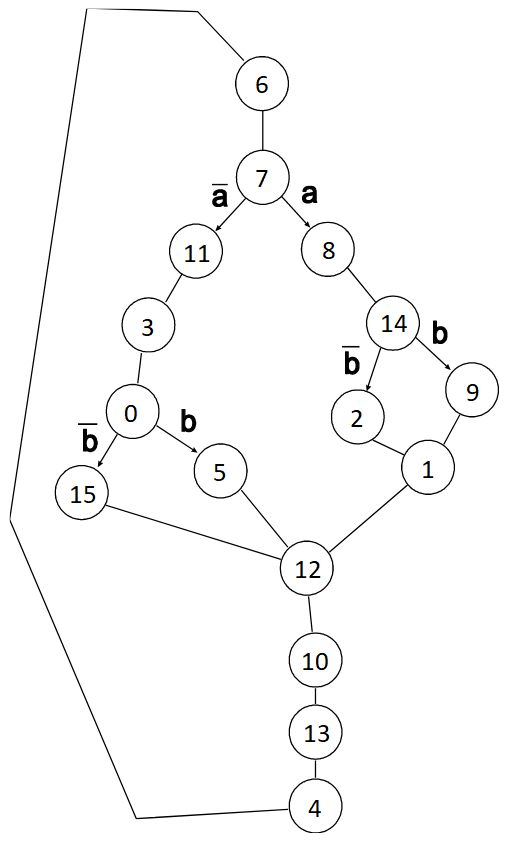


Рисунок 15 – Граф, полученный с учетом таблицы перекодировки

*Таблица 8 – Таблица истинности автомата*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| старое состояние | | условие | новое состояние | |
| № | код |  | № | Код |
| 6 | 0110 | - | 7 | 0111 |
| 7 | 0111 | A=0 | 11 | 1011 |
| 7 | 0111 | A=1 | 8 | 1000 |
| 11 | 1011 | - | 3 | 0011 |
| 3 | 0011 | - | 0 | 0000 |
| 0 | 0000 | B=0 | 15 | 1111 |
| 0 | 0000 | B=1 | 5 | 0101 |
| 15 | 1111 | - | 12 | 1100 |
| 12 | 1100 | - | 10 | 1010 |
| 10 | 1010 | - | 13 | 1101 |
| 13 | 1101 | - | 4 | 0100 |
| 4 | 0100 | - | 6 | 0110 |
| 8 | 1000 | - | 14 | 1110 |
| 14 | 1110 | B=0 | 2 | 0010 |
| 14 | 1110 | B=1 | 9 | 1001 |
| 2 | 0010 | - | 1 | 0001 |
| 1 | 0001 | - | 12 | 1100 |
| 5 | 0101 | - | 12 | 1100 |
| 9 | 1001 | - | 1 | 0001 |

Для синтезирования электрической принципиальной схемы счётчика с произвольным модулем счёта были использованы логические элементы AND, OR, NOT, а также D-триггеры.

**3 Логическая схема и временная диаграмма**

На Рисунке 16 и Рисунке 17 представлены логическая схема и временная диаграмма, выполненные для соответствующего варианта счетчика.

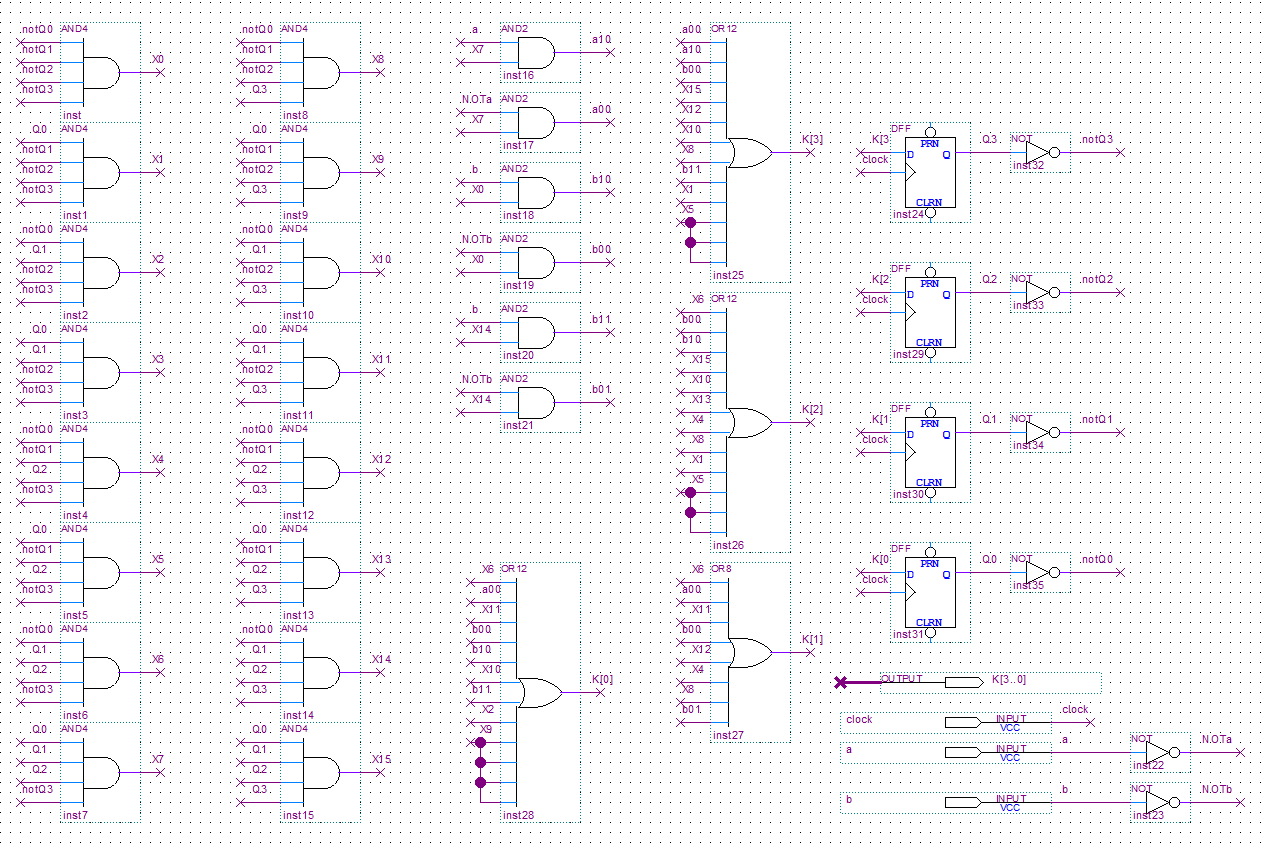
**

Рисунок 16 – Логическая схема

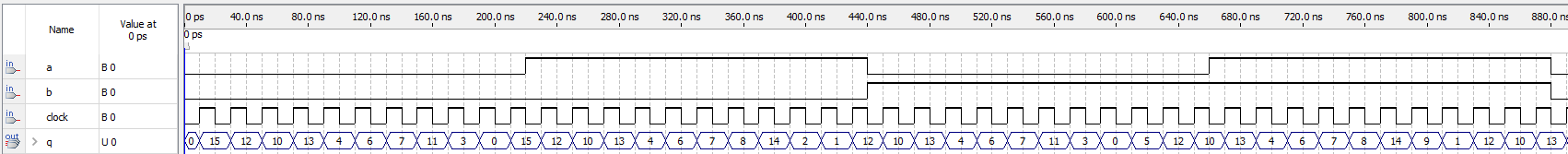
**

Рисунок 17 – Временная диаграмма

# ПРАКТИЧЕСКАЯ РАБОТА №7

1. **Постановка задачи**
2. Изучить основные элементы языка AHDL и правила описания логических схем.
3. Сделать описание электрической схемы, заданной в предыдущей работе при помощи текстового редактора САПР QUARTUS II.
4. Произвести симуляцию работы схемы. Зарисовать диаграммы работы и по ее результатам заполнить таблицу истинности смоделированной схемы.
5. Сравнить результаты, полученные в ходе выполнения лабораторной работы с результатами, полученными в работе №1.
6. Ответить на контрольные вопросы, оформить отчет о выполненной работе.

Задание варианта 27:

*Таблица 9 – Варианты состояний исходного графа*

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 6 | 7 | 8 | 14 | 2 | 9 | 1 | 12 | 11 | 3 | 0 | 5 | 15 | 10 | 13 | 4 |

**2 Теоретическое введение**

Необходимо построить логическую схему на языке AHDL счётчика с произвольным модулем счёта согласно таблице состояний варианта.

На Рисунке 18 представлен граф, полученный с учетом таблицы перекодировки.

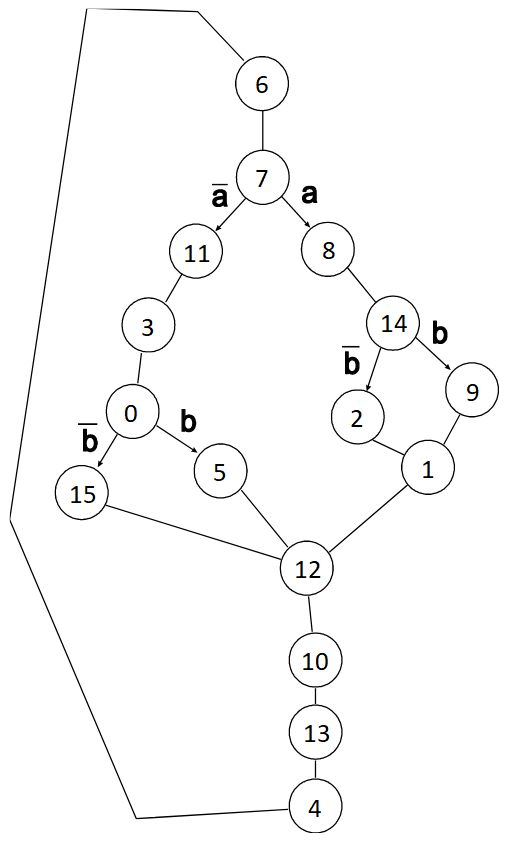


Рисунок 18 – Граф, полученный с учетом таблицы перекодировки

Для синтезирования электрической принципиальной схемы счётчика с произвольным модулем счёта были использованы логические элементы AND, OR, NOT, а также D-триггеры. Для описания цифровой схемы счётчика с произвольным модулем счёта с помощью языка описания аппаратуры AHDL (текстового редактора САПР QUARTUS II) были использованы логические элементы AND (&), OR (#), NOT (!), а также внутренний структурный элемент ПЛИС - D-триггер.

**3 Код и временная диаграмма**

На Рисунке 19 и Рисунке 20 представлены логическая схема и временная диаграмма, выполненные для соответствующего варианта счетчика.

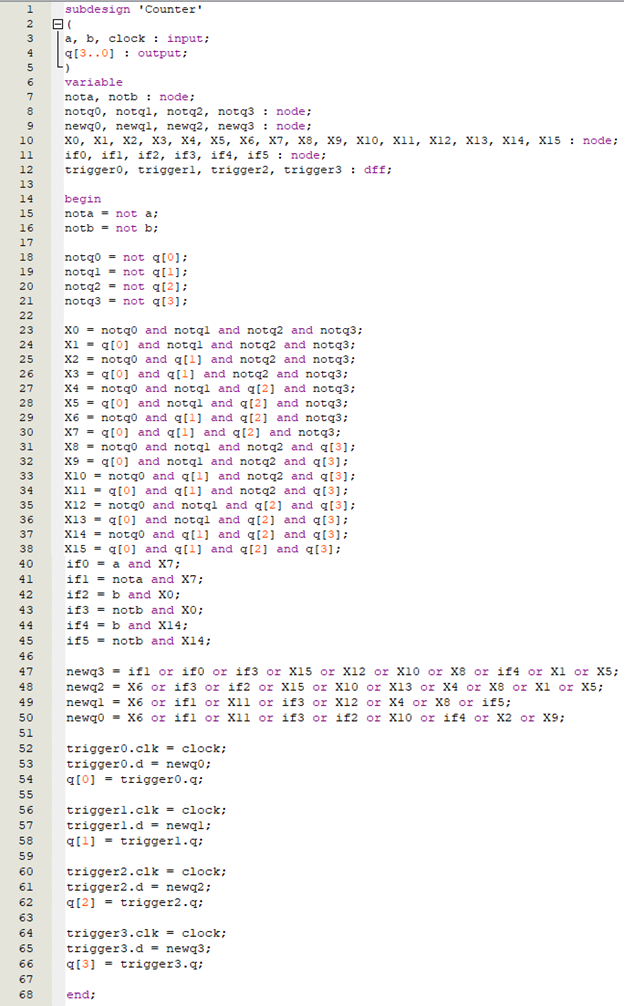


Рисунок 19 – Код

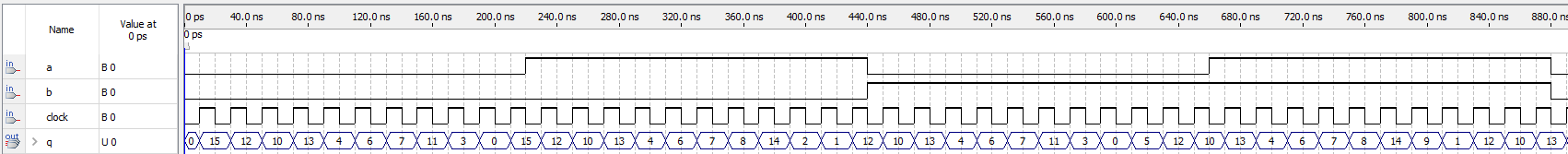
**

Рисунок 20 – Временная диаграмма

## 4 Сравнение результатов

На Рисунке 21 и Рисунке 22 представлены две временные диаграммы, первая диаграмма соответствует построенной логической схеме, а вторая - AHDL-коду.

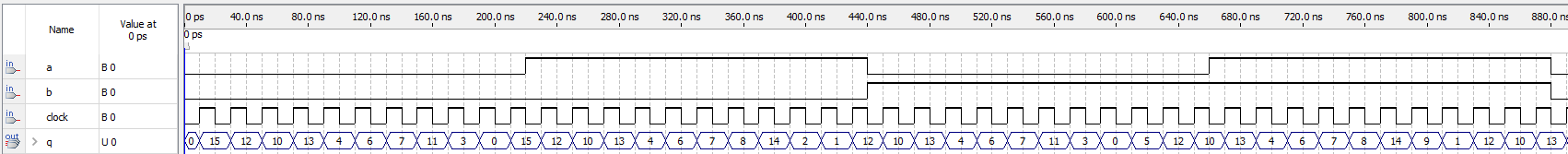
**

Рисунок 21 – Временная диаграмма практической работы №6

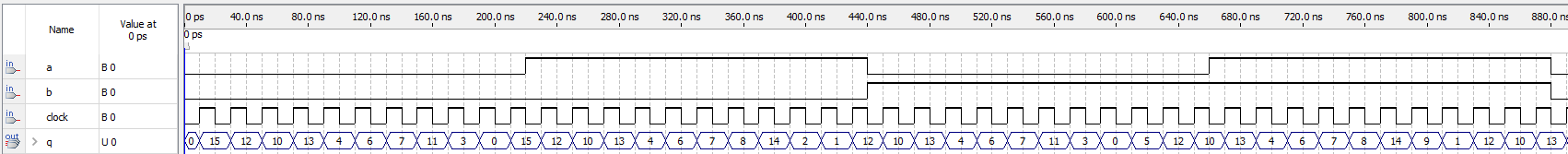
**

Рисунок 22 – Временная диаграмма практической работы №7

Сравнение показало, что реализация счётчика с произвольным модулем счёта логической схемой и реализация AHDL-кодом дают одинаковые результаты (временные диаграммы).

# ПРАКТИЧЕСКАЯ РАБОТА №8

## Выполнение лабораторной работы: вариант 10

В командной строке создаем родительский каталог. В качестве имени каталога задаем свою фамилию (Utenkov). Внутри нашего каталога создаем еше 2 каталога с именами database и temp. Внутри каталога temp создаем каталог report. Переходим в каталог temp (Рисунок 23).

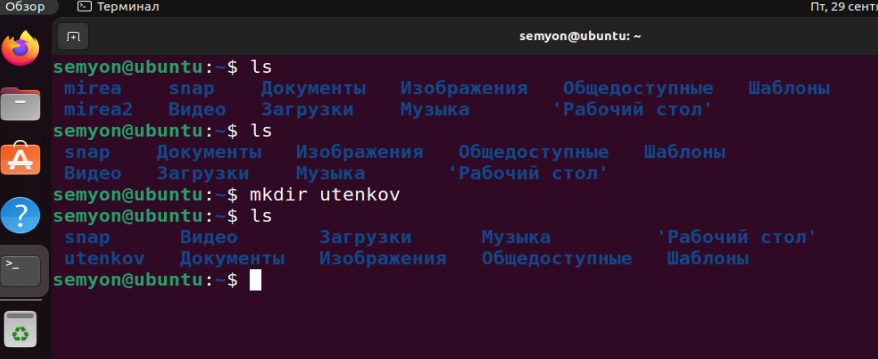


Рисунок 23 – Создание каталогов

Выведем на экран содержимое текущего каталога и убедимся, что все созданные каталоги созданы без ошибок. Для отображения используем утилиту tree (Рисунок 24).

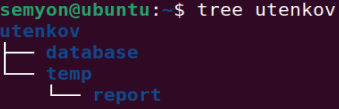


Рисунок 24 – Вывод каталога в виде дерева

Перейдем в каталог temp. Убедимся, что он является текущим. Выведем на экран содержимое каталога (Рисунок 25).

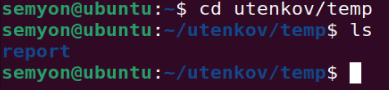


Рисунок 25 – Проверка на то, что находимся в текущем каталоге

С помощью текстового редактора внутри каталога temp создать файл базы данных dataset1.txt. Заполнить файл данными в соответствии с номером варианта задания (10 вариант: Экспорт (наименование товара, объем поставки, стоимость единицы продукции, страна экспорта). Поиск по наименованию товара. Сортировка по объемам поставки. (Рисунок 26).

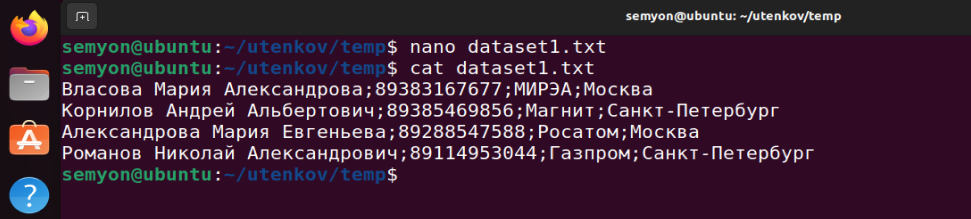


Рисунок 26 – Создание и заполнение dataset1.txt

С помощью конвеера команд внутри каталога temp создать файл базы данных dataset2.txt (Рисунок 27). Заполнить файл данными в соответствии с номером варианта задания.

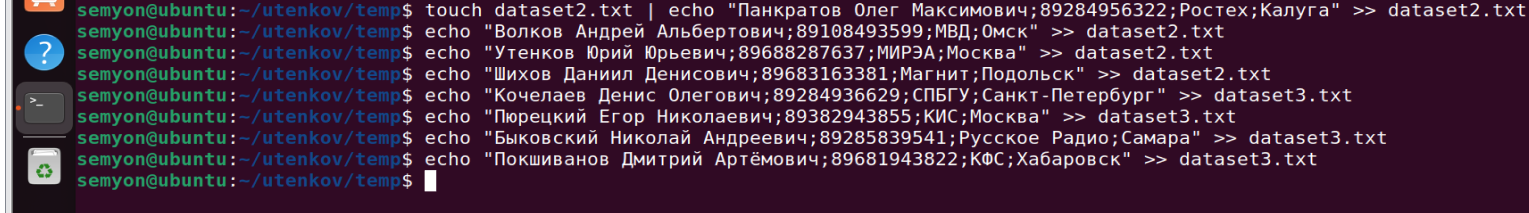


Рисунок 27 – Заполнение и заполнение dataset2.txt

Выводим на экран содержимое всех созданных файлов базы данных (Рисунок 28).

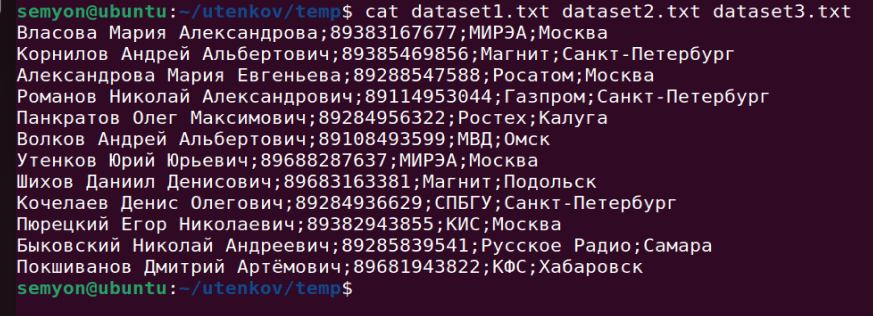


Рисунок 28 – Вывод на экран содержимого всех созданных файлов

Далее мы объединяем содержимое всех созданных файлов базы данных в один файл data.txt и помещаем его в каталог database. При помощи команды cat мы объединяем 3 файла в один и перемещаем их с помощи “>” (Рисунок 29).

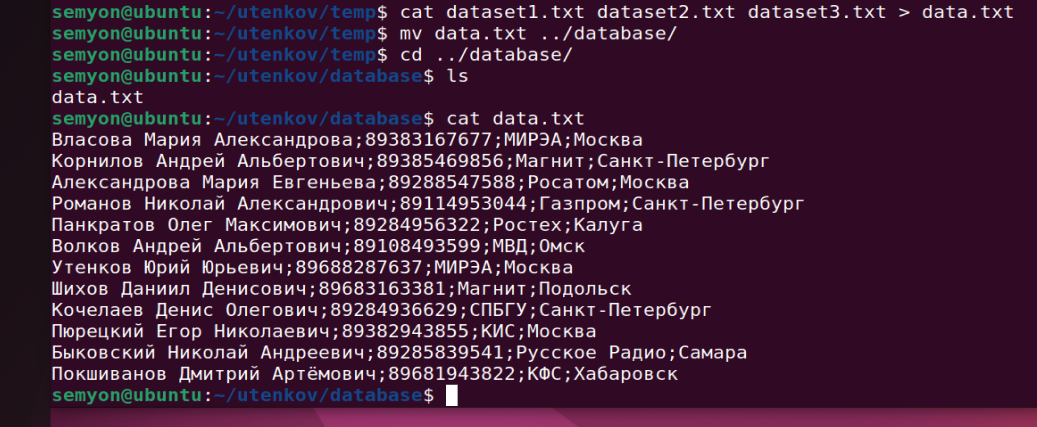


Рисунок 29 – Объединение в один файл и перенос этого файла в другой каталог

Далее при помощи команды wc считаем количество строк в data.txt и выводим эти данные в output.txt (Рисунок 30), расположенный в каталоге report.

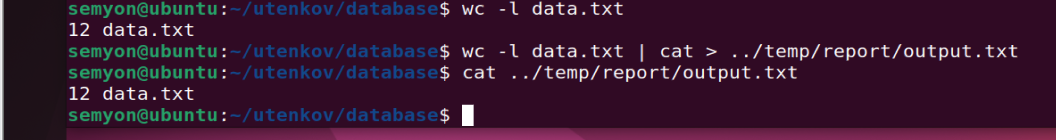


Рисунок 30 – Вывод количества строк в output.txt

Затем с помощью любого из использованных выше способов (команда echo) дополняем файл data.txt двумя строками данных в соответствии с номером варианта задания. Убеждаемся, что файл data.txt содержит все необходимые данные. (Рисунок 31)

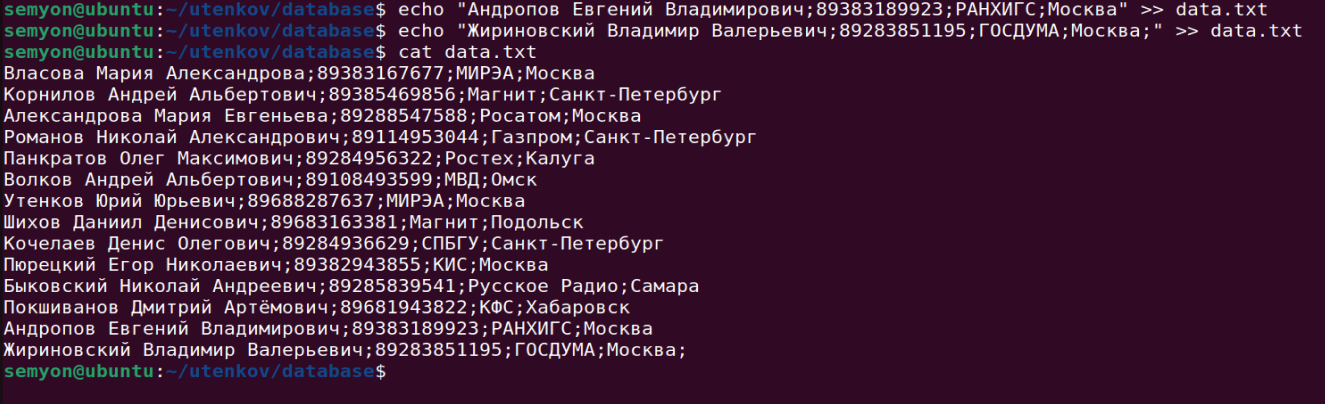


Рисунок 31 – Дополнение файла data.txt двумя строками данных

Далее посчитаем количество строк файла data.txt с помощью wc Результат подсчета выводим на экран и дописываем в конец файла отчета output.txt, расположенного в каталоге report (Рисунок 32).

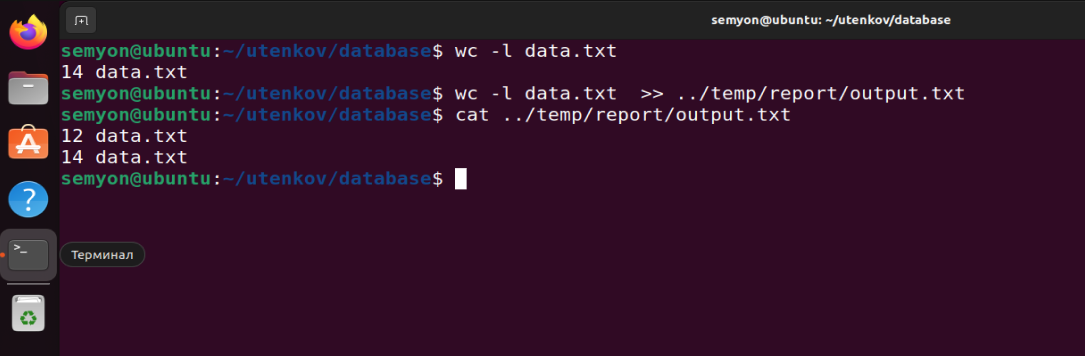


Рисунок 32 – Подсчет количества строк в data.txt после добавления новых строк, запись в output.txt

Осуществляем фильтрацию данных файла data.txt в соответствии с нашим номером варианта задания. Результат фильтрации выводим на экран и в файл отчета filtered.txt (Рисунок 33), расположенный в каталоге героrt.

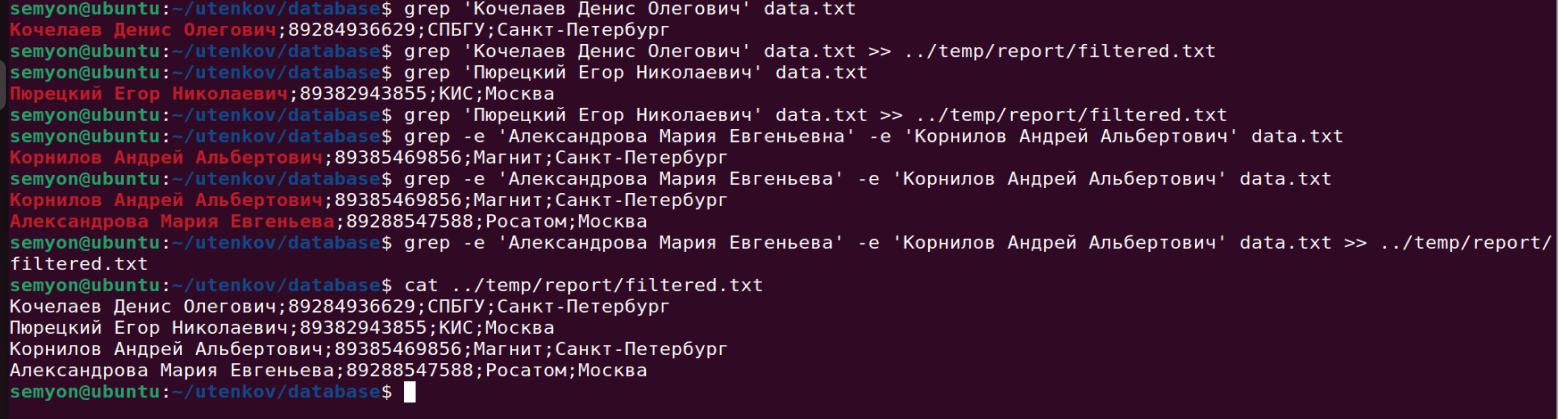


Рисунок 33 - Вывод результата фильтрации

Выполняем сортировку содержимого файла data.txt в соответствии с нашим номером вариантом задания. Результат сортировки выводим на экран и в файл отчета sorted.txt, расположенный в каталоге героrt. (Рисунок 34)

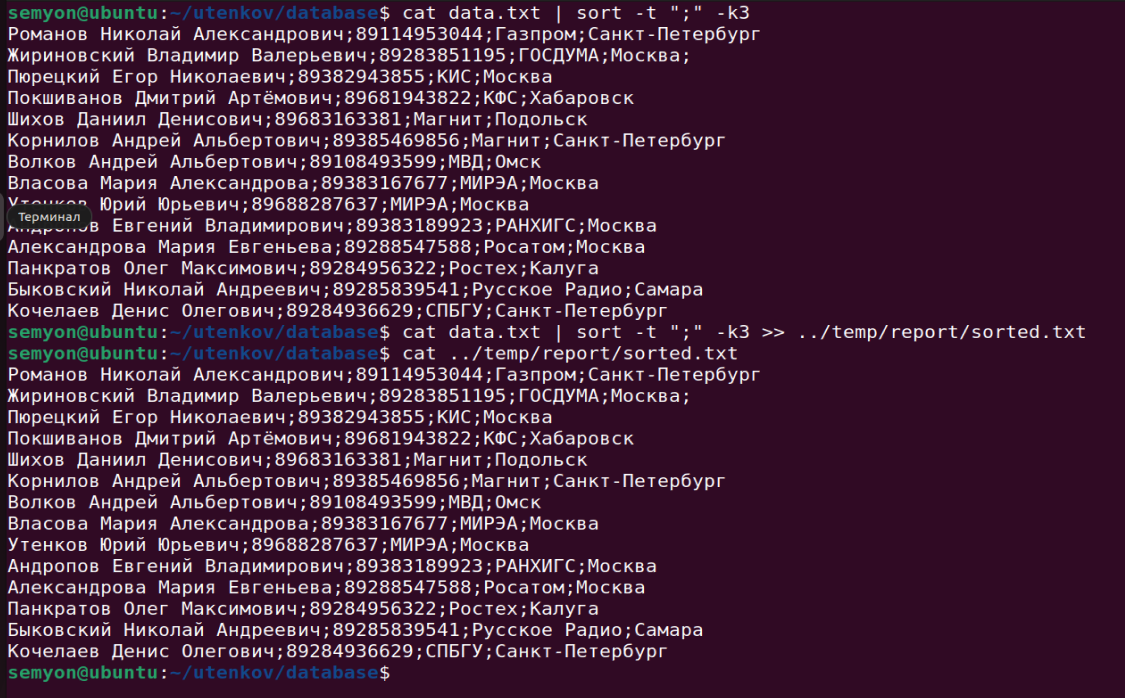


Рисунок 34 - Вывод на экран результата сортировки

Выполняем фильтрацию содержимого файла data.txt с сортировкой результата фильтрации. Фильтрацию и сортировку выполняем в соответствии с номером варианта задания. Результат выводим на экран и в файл отчета filtefedsorted.txt, расположенный в каталоге героrt (Рисунок 35).

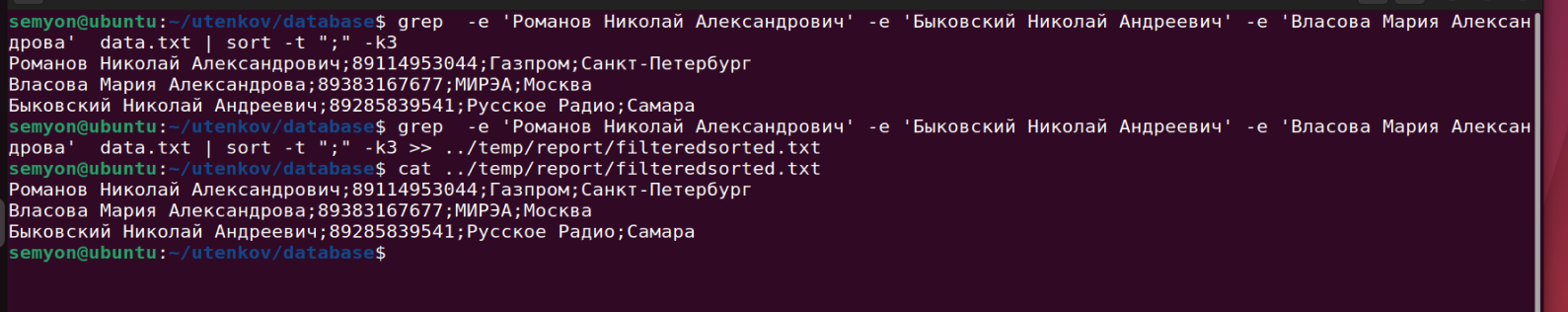


Рисунок 35 - Вывод результата сортировки и фильтрации на экран, в файл filteredsorted.txt

Выполним команду вывода календаря на экран (Рисунок 36) и любым известным способом записать значение в файл calendar.txt, находящийся в каталоге /database. Результат вывести на экран (Рисунок 36).

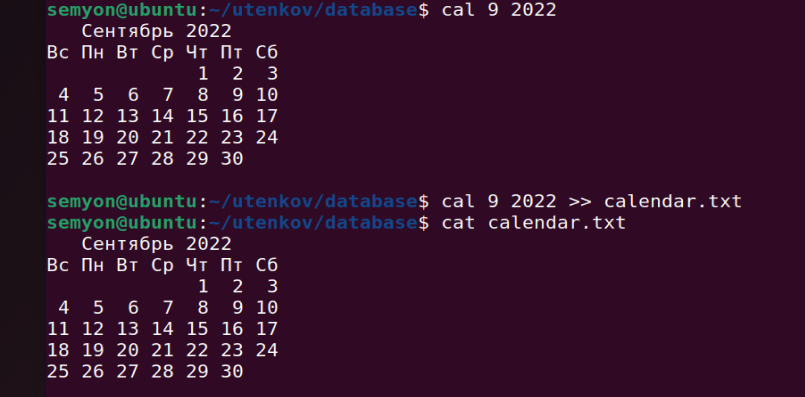


Рисунок 36 – Запись и вывод календаря в файл

# ЗАКЛЮЧЕНИЕ

В ходе выполнения практических работ №l-7 были исследованы способы построения логических схем и написания AHDL-кода в САПР Quartus. Были построены таблицы истинности для логической функции, для DMUX 2-4, для компаратора 3хcompare А>=В. На основе этих таблиц были построены логические схемы и написан код на языке AHDL. Тестирование показало, что таблицы истинности построены корректно, код и схемы работают правильно, временные диаграммы для кода и схемы совпадают.

Также был реализован счётчик с произвольным модулем счёта двумя способами (логическая схема и код на языке AHDL), для которого была построена таблица перекодировки состояний автомата, по этой таблице был построен соответствующий граф, была построена таблица истинности автомата. Тестирование показало, что таблица истинности построена корректно, код и схема работают правильно, временные диаграммы для кода и схемы совпадают.

В ходе выполнения практической работы №8 (Linux) были приобретены навыки установки Ubuntu, был проведен процесс создания виртуальной машины. Также была выполнена установка и начальная настройка операционной системы Kali. Были приобретены навыки работы с файлами и каталогами, и изучены некоторые команды манипулирования данными на примере текстовой базы данных.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Мусихин А.Г. Архитектура вычислительных машин и систем [Электронный ресурс]: Методические рекомендации к контрольным работам / Мусихин А.Г., Смирнов Н.А. - М.: МИРЭА - Российский технологический университет, 2020. - 1 электрон. опт. диск (CD-ROM)
2. Головков А., Пивоваров И., Кузнецов И. Компьютерное моделирование и проектирование радиоэлектронных средств. Учебник для вузов. Стандарт третьего поколения.:- СПб.: 2015. - 208 с.
3. Соловьев В.В., Климович А. Логическое проектирование цифровых систем на основе программируемых логических интегральных схем. - М.: Горячая линия - Телеком, 20011. - 376 с.
4. Стешенко В. ПЛИС фирмы ALTERA: элементная база, система проектирования и языки описания аппаратуры - М.: Додека, 2010.- 576 c.
5. Антонов А.П. Язык описания цифровых устройств AlteraHDL: Практический курс. - М.: ИП «Радиософт», 2013. - 224 с.
6. Ефремов Н.В. Введение в систему автоматизированного проектирования Quartus II. Учебное пособие. - М.: ГОУ ВПО МГУЛ, 2011. - 147 с.