ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«САНКТ-ПЕТЕРБУРГСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ПЕТРА ВЕЛИКОГО»**

Институт компьютерных наук и технологий

Высшая школа интеллектуальных систем и суперкомпьютерных технологий

ЛАБОРАТОРНАЯ РАБОТА №1

**Знакомство с пакетом Vivado**

по дисциплине «Технологии проектирования аппаратных средств компьютерных систем»

Выполнил:

студент группы 3540901/02001

Дроздов Н.Д.

Проверил:

А. П. Антонов

Санкт-Петербург 2020

# **Содержание**

[**Список иллюстраций** 3](#_Toc56568058)

[**Введение** 5](#_Toc56568059)

[**Цель лабораторной работы** 5](#_Toc56568060)

[**Ход работы** 6](#_Toc56568061)

[Этап 1. Создание проекта в пакете Vivado 6](#_Toc56568062)

[Этап 2. Анализ файлов с описанием проекта 6](#_Toc56568063)

[Этап 3. Поведенческое моделирование проекта 8](#_Toc56568064)

[Этап 4. Анализ структуры RTL описания проекта 13](#_Toc56568065)

[Этап 5. Синтез проекта и анализ результатов 14](#_Toc56568066)

[Этап 6. Реализация проекта и анализ результатов 17](#_Toc56568067)

[Этап 7. Моделирование реализованного проекта 23](#_Toc56568068)

[**Вывод** 26](#_Toc56568069)

# **Список иллюстраций**

[*Рисунок 1 - созданный проект* 7](#_Toc56568004)

[Рисунок 2 - файл lab1 8](#_Toc56568005)

[*Рисунок 3 - Файл lab1\_nexys4\_ddr.xdc* 8](#_Toc56568006)

[*Рисунок 4 - Папка Simulation Sources* 9](#_Toc56568007)

[*Рисунок 5 - Файл lab1\_tb.v* 10](#_Toc56568008)

[*Рисунок 6 - Установка значения 200ns для параметра xsim.simulate.runtime* 10](#_Toc56568009)

[*Рисунок 7 - Результаты поведенческого моделирования* 11](#_Toc56568010)

[*Рисунок 8 - Добавление на временную диаграмму сигналов, принадлежащих тестируемому модулю DUT* 12](#_Toc56568011)

[*Рисунок 9 - Временная диаграмма, после перезапуска процесса моделирования* 13](#_Toc56568012)

[Рисунок 10 - Процесс моделирования на всю длину теста 13](#_Toc56568013)

[Рисунок 11 - Панель Tcl Console 14](#_Toc56568014)

[Рисунок 12 - Структурная схема RTL описания 15](#_Toc56568015)

[Рисунок 13 - Project Summary(1) 16](#_Toc56568016)

[Рисунок 14 - Project Summary(2) 16](#_Toc56568017)

[Рисунок 15 - Project Summary(3) 17](#_Toc56568018)

[Рисунок 16 - Схемное представление результатов синтеза 17](#_Toc56568019)

[Рисунок 17 - Детальный отчет об использованных ресурсах 18](#_Toc56568020)

[Рисунок 18 - Примитивы, использованные для отображения созданной после синтеза структурной схемы 18](#_Toc56568021)

[Рисунок 19 - Implementation Completed 19](#_Toc56568022)

[Рисунок 20 - Implemented Design 19](#_Toc56568023)

[Рисунок 21 - Трассировка одной из цепей 20](#_Toc56568024)

[Рисунок 22 - Табличная и графическая форма представления оценок 21](#_Toc56568025)

[Рисунок 23 - Reports 21](#_Toc56568026)

[Рисунок 24 – Reports 22](#_Toc56568027)

[Рисунок 25 – Reports 22](#_Toc56568028)

[Рисунок 26 - Reports 22](#_Toc56568029)

[Рисунок 27 – Reports 23](#_Toc56568030)

[Рисунок 28 – Reports 23](#_Toc56568031)

[Рисунок 29 - Reports 23](#_Toc56568032)

[Рисунок 30 - Отчеты по выполненным этапам Синтеза и Реализации проекта 24](#_Toc56568033)

[*Рисунок 31 - Временная диаграмма* 26](#_Toc56568034)

# **Введение**

Эта лабораторная работа демонстрирует процедуру использования пакета Vivado:

* от создания проекта, для описания которого использован язык описания аппаратуры (HDL),
* до конфигурации FPGA и проверки проекта на плате.

Проект ориентирован на использование микросхем семейства Artix7 и предназначен для реализации на платах Basys3 или Nexys4 DDR.

В описании лабораторной работы:

* {sources} означает: C:\xilinx\_trn\FPGA\_Vivado\_Int\labs\_src
* {labs} означает: C:\xilinx\_trn\FPGA\_Vivado\_Int\labs
* {boards} означает: C:\xilinx\_trn\FPGA\_Vivado\_Int\boards

# **Цель лабораторной работы**

* создать проект, основанный на HDL описании;
* выбрать целевую платформу (как и конкретную FPGA, так и плату с этой микросхемой);
* использовать созданный заранее файла Xilinx Design Constraint (XDC) - файл для задания выводов микросхемы;
* осуществить моделирование проекта (поведенческое и с временными параметрами);
* синтезировать проект;
* создать конфигурационный файл;
* конфигурировать FPGA на плате и осуществлять проверку проекта.

# **Ход работы**

## Этап 1. Создание проекта в пакете Vivado

1. Запустите пакет **Vivado 2020.1** (или Vivado WebPack 2020.1)
2. Создайте новый проект:

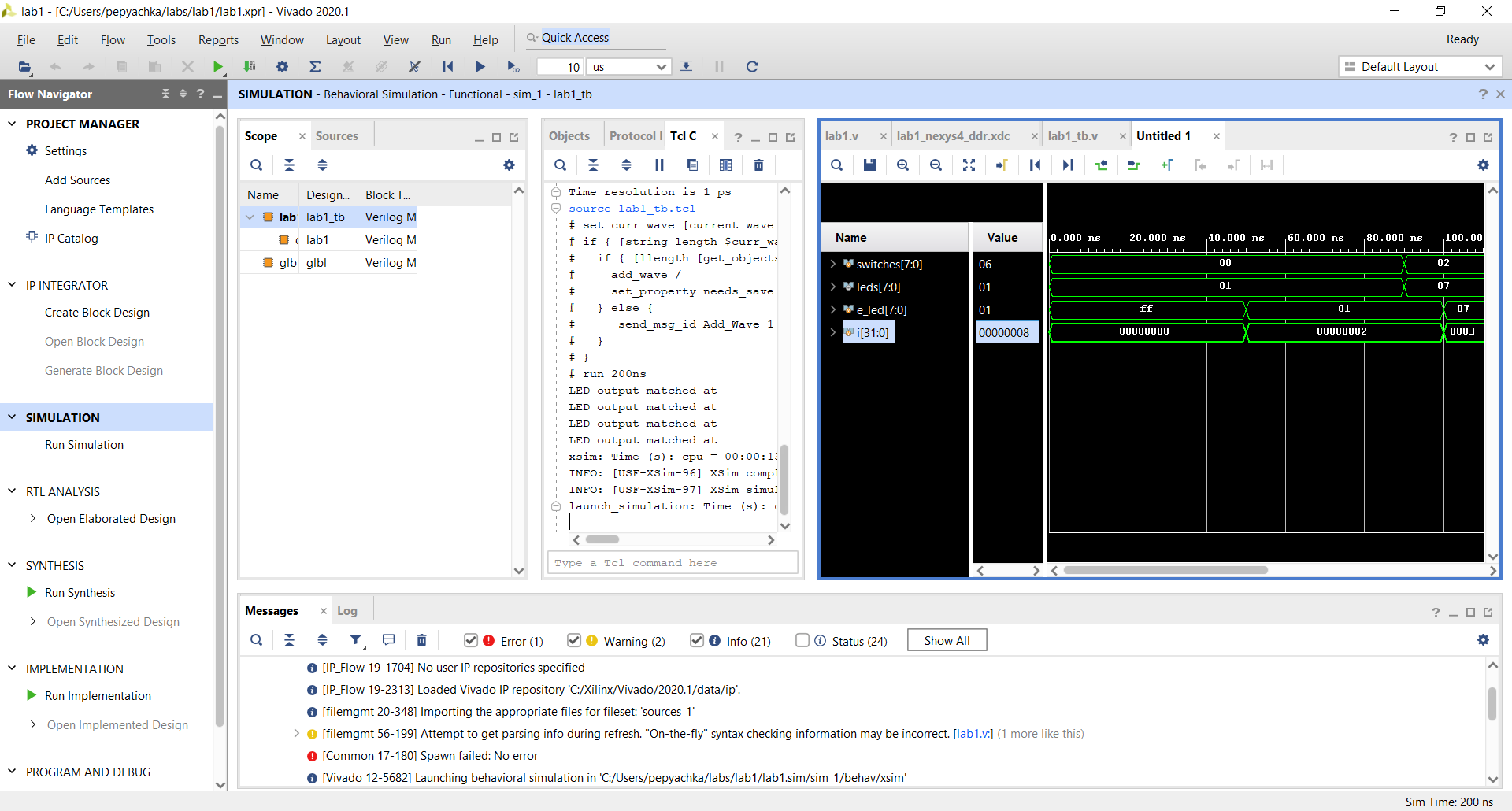
* Папка для проекта {labs}.
* Имя проекта lab1.
* Для описания проекта и теста используется язык Verilog

1. Из папки ***{sources}\lab1*** добавьте к проекту:

* Файл lab1.v с описанием проекта.
* Файл lab1\_nexys4\_ddr.xdc с описанием назначений выводов для платы Nexys4 DDR.

1. Следует выбрать конкретную микросхему FPGA для реализации проекта. Тип микросхемы:

* XC7A100TCSG324-1 (если используется плата Nexys4 DDR)
* *XC7A35TCPG236-1 (если используется плата Basys3).*



*Рисунок 1 - созданный проект*

## Этап 2. Анализ файлов с описанием проекта

1. В разделе **Sources**, в папке **Design Sources**, двойным щелчком по модулю (файлу) **lab1** (**lab1.v**) запустите текстовый редактор пакета Vivado, в котором будет отображено содержимое файла.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 2 - файл lab1

1. В разделе **Sources**, в папке **Constraints** двойным щелчком по файлу **lab1\_nexys4\_ddr.xdc** (по файлу lab1\_basys3.xdc если в лабораторной работе используется плата Вasys3) запустите текстовый редактор пакета Vivado, в котором будет отображено содержимое файла.

Изображение выглядит как текст

Автоматически созданное описание

*Рисунок 3 - Файл lab1\_nexys4\_ddr.xdc*

## Этап 3. Поведенческое моделирование проекта

1. Перейдите к папке **{sources}\lab1**, в которой выберите файл **lab1\_tb.v** (это тест на языке Verilog для проверки алгоритма работы проекта).
2. В разделе **Sources** откройте папку **Simulation** **Sources**: в подпапке sim\_1 находится добавленный файл lab1\_tb.v с тестом, в котором в качестве тестируемого компонента (Design Under Test – DUT) указан файл lab1.v с описанием проекта (модуля lab1).

Изображение выглядит как текст

Автоматически созданное описание

*Рисунок 4 - Папка Simulation Sources*

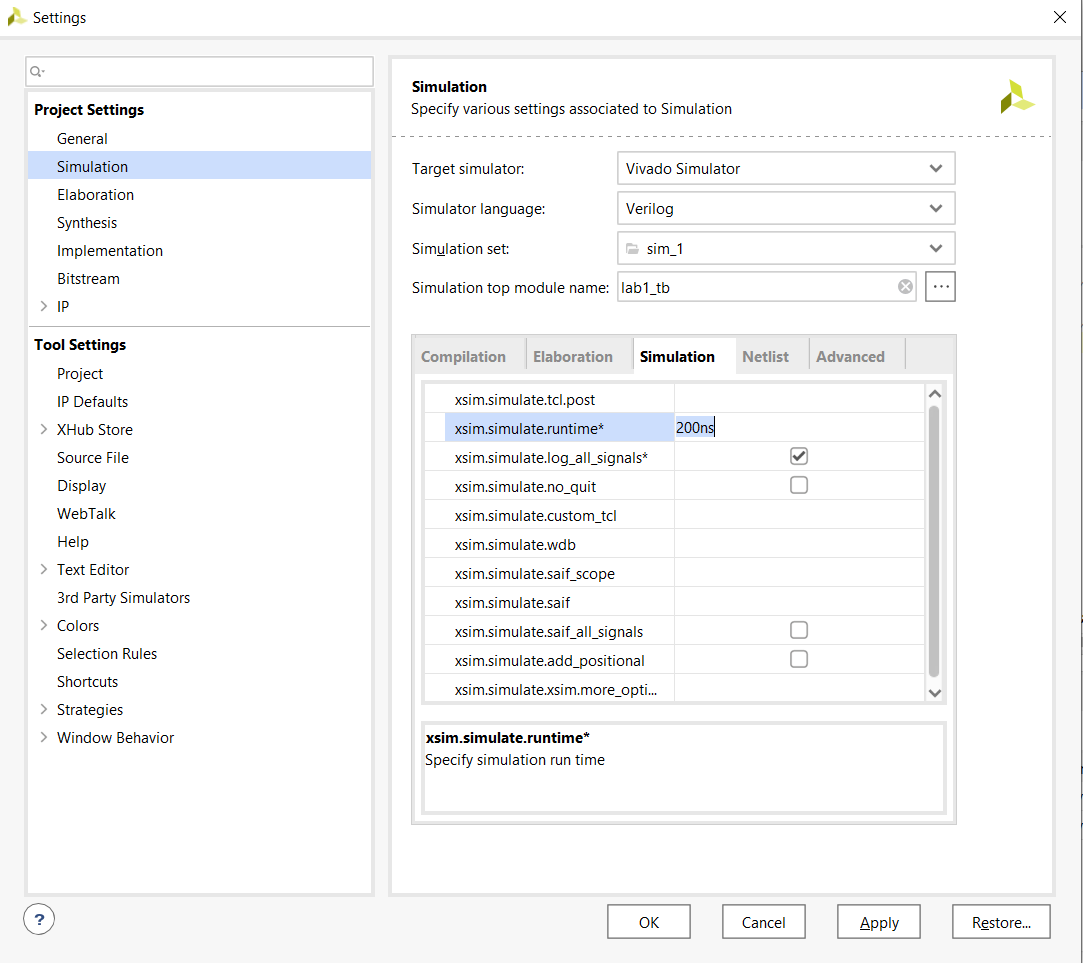
1. В разделе **Sources**, в папке **Simulation** **Sources** => **sim\_1**, двойным щелчком по файлу **lab1\_tb.v** запустите текстовый редактор пакета Vivado, в котором будет отображено содержимое файла.

Изображение выглядит как текст

Автоматически созданное описание

*Рисунок 5 - Файл lab1\_tb.v*

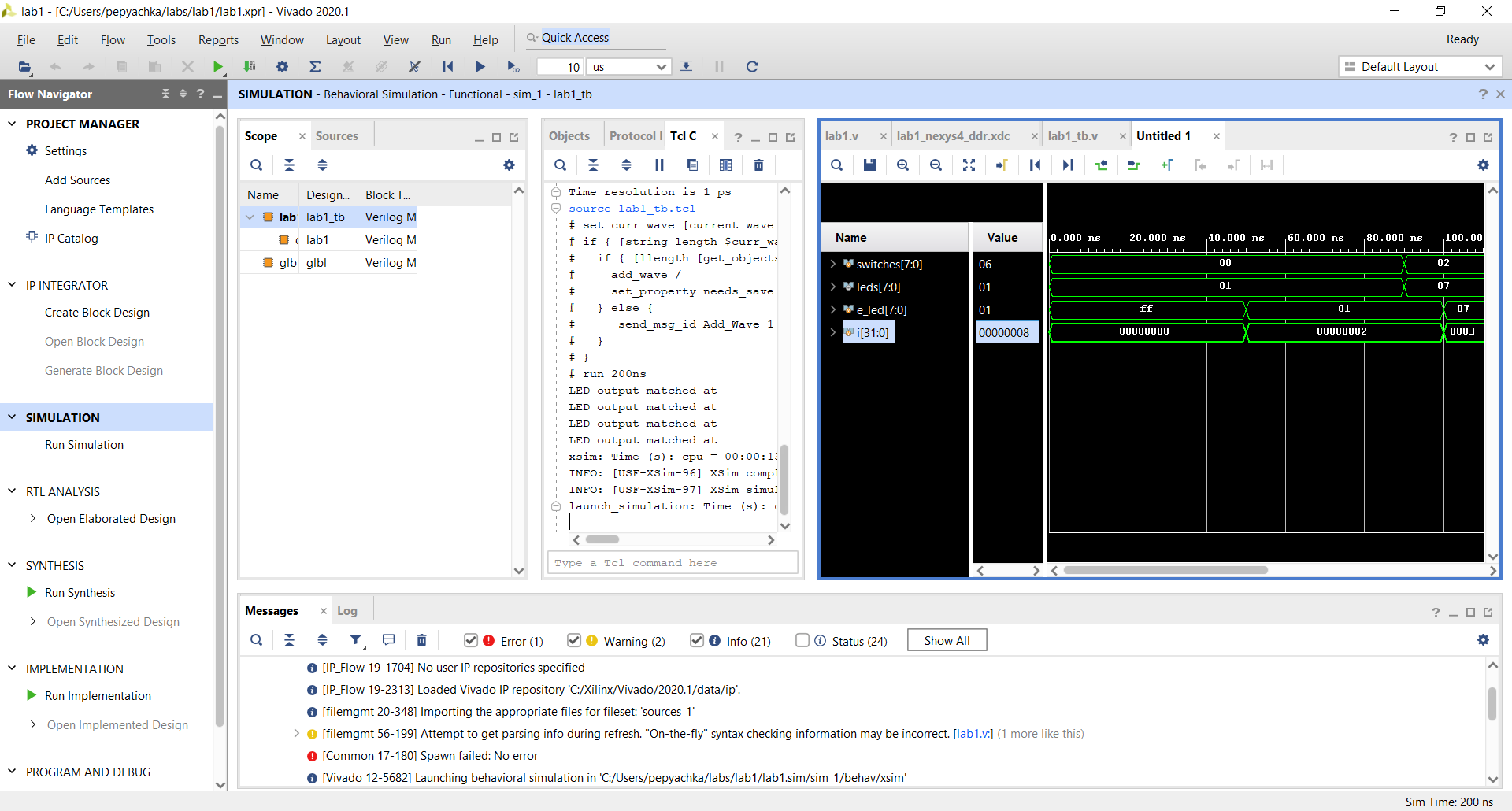
1. В разделе **Flow Navigator** щелчком запустите команду **Settings**.



*Рисунок 6 - Установка значения 200ns для параметра xsim.simulate.runtime*

1. В разделе **Flow Navigator**, в папке **Simulation**, щелчком выберите команду **Run Simulation** и запустите команду **Run Behavioral Simulation** (поведенческое моделирование).

*Пакет Vivado осуществит компиляцию (Elaboration) файла с тестом (testbench) и файла с описанием тестируемого модуля (source). Затем будет запущена процедура моделирования и после моделирования 200ns (модельное время) в окне Simulation будут представлены результаты моделирования, приведенные ниже.*



*Рисунок 7 - Результаты поведенческого моделирования*

На приведенном выше рисунке представлены четыре основные панели системы моделирования:

* **Scopes** – панель, в которой отображается иерархия теста (в данном случае это файл верхнего уровня с тестом и тестируемый модуль DUT)
* **Objects** – панель, в которой отображаются сигналы (входы, выходы и внутренние цепи, и переменные) модуля, выбранного в панели Scopes,
* Окно с временной диаграммой (на приведенном выше рисунке она названа Untitled 1),
* **Tcl Console** – панель, в которой отображаются сообщения и результаты моделирования. Так как используемый в лабораторной работе тест – это тест с самопроверкой, то в панели отображены сообщения о результатах выполнения отдельных Этапов теста (в тесте сообщения формируются командами $display).

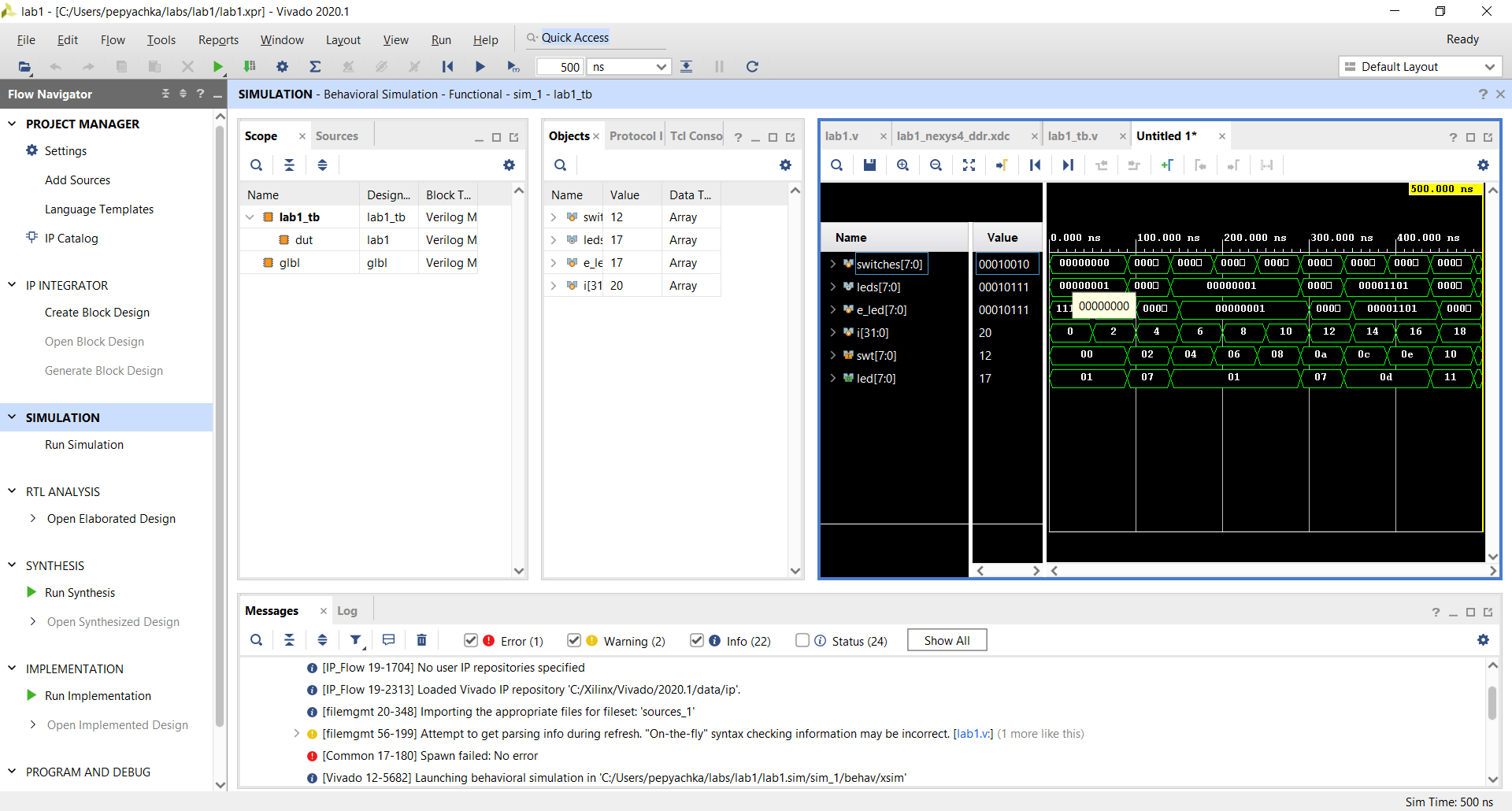
Промежуточные данные и результаты моделирования находятся в созданной пакетом папке **{labs}\lab1\lab1.sim\sim\_1\behav** (behav означает behavioral т. к. было осуществлено поведенческое моделирование).

1. Измените формат отображения данных на временной диаграмме.

* Щелчком в столбце **Name** временной диаграммы выделите шину **i[31:0].**
* Нажмите правую клавишу мыши и в появившемся списке выберите **Radix**=> **Unsigned** **Decimal** (индексы цикла будут отображены в формате целых без знаковых чисел).
* Аналогично измените отображение данных для шин **switches[7:0], leds[7:0]** и **e\_led[7:0]** на *Binary*.

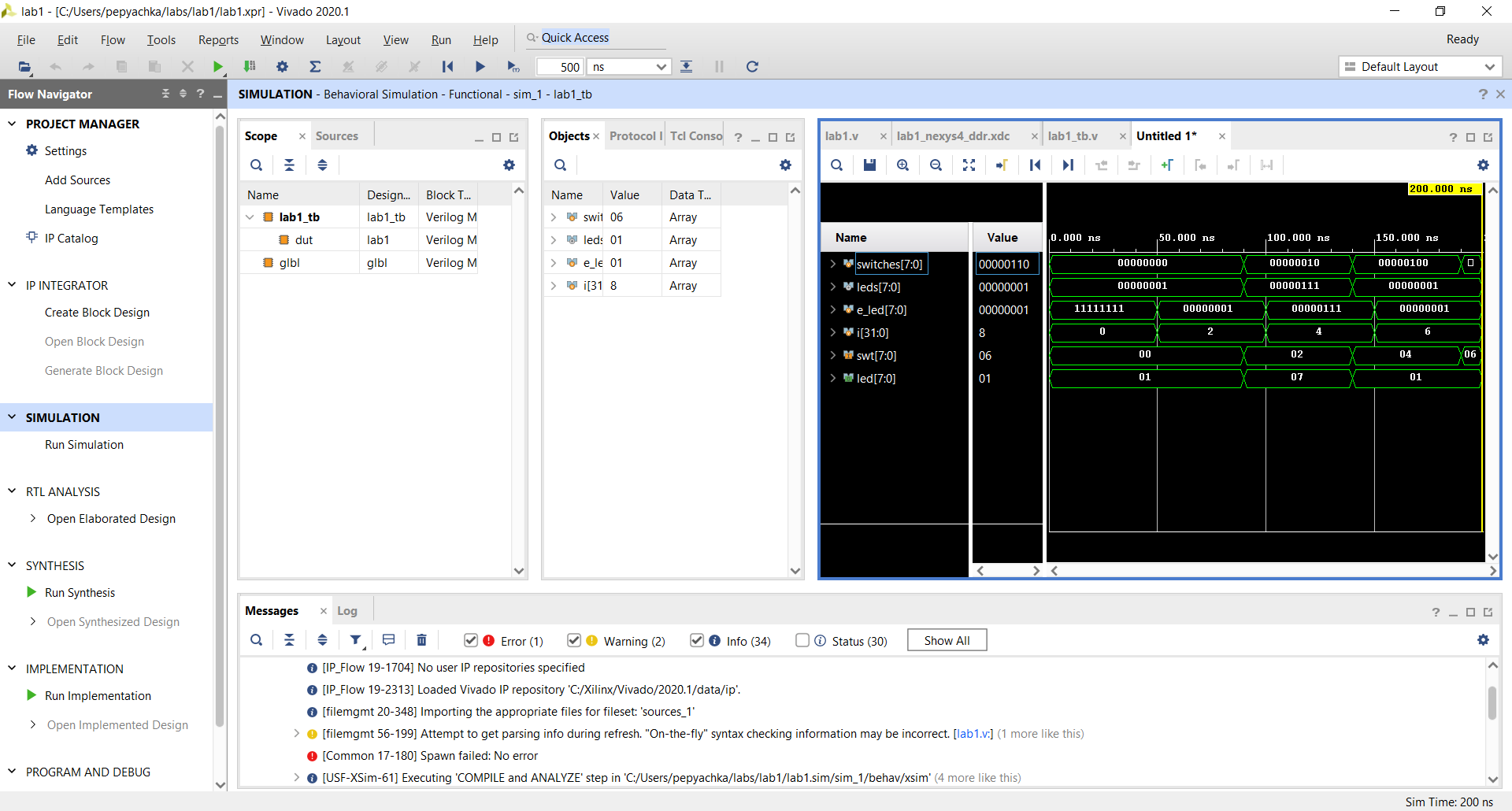
1. Добавьте на временную диаграмму сигналы, принадлежащие тестируемому модулю DUT (аналогичный подход может быть использован для добавления сигналов, принадлежащих любому модулю в иерархии теста).

* В панели **Scope** раскройте иерархию теста **lab1\_tb** и щелчком выберите тестируемый модуль (**dut**).
* Сигналы **swt[7:0]** и **led[7:0]**, принадлежащие тестируемому модулю, появятся в панели Objects.
* В панели **Objects** поочередно щелчком выберите шины **swt[7:0]** и **led[7:0]** и перетащите их в поле **Name** временной диаграммы.



*Рисунок 8 - Добавление на временную диаграмму сигналов, принадлежащих тестируемому модулю DUT*

1. Сбросьте результаты моделирования и осуществите моделирование продолжительностью **500 ns.**
2. Перезапустите процесс моделирования – полученная временная диаграмма должна быть похожа на временную диаграмму, приведенную на рисунке ниже.



*Рисунок 9 - Временная диаграмма, после перезапуска процесса моделирования*

1. Запустите процесс моделирования на всю длину теста – полученная временная диаграмма должна быть похожа на временную диаграмму, приведенную на рисунке ниже.

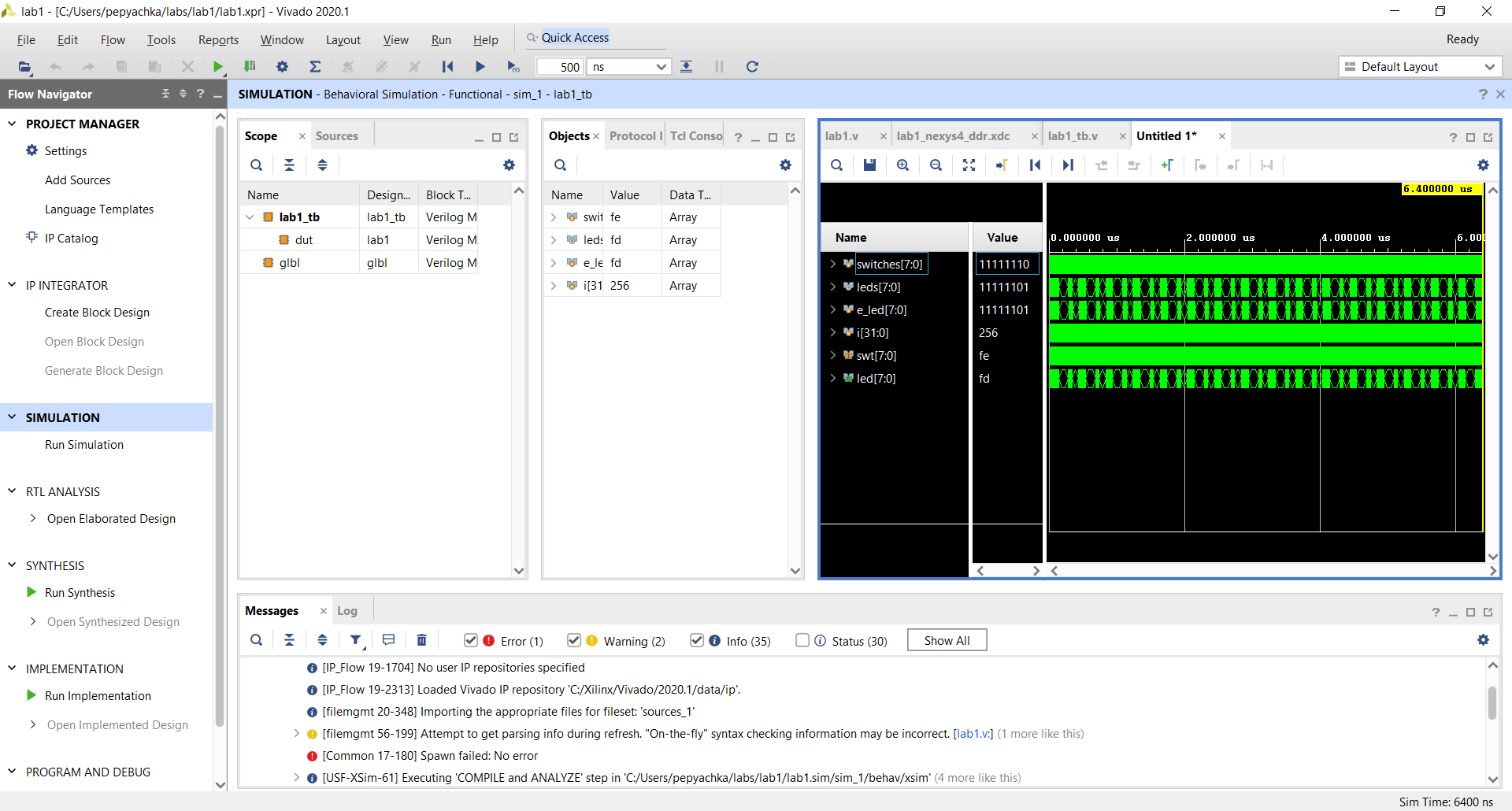


Рисунок 10 - Процесс моделирования на всю длину теста

Обратите внимание на панель Tcl Console, в которой отображены выполненные команды и результаты моделирования, сформированные в тесте командой $display.

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 11 - Панель Tcl Console

1. Закройте систему моделирования:
   * Выполните команду **File** > **Close Simulation**.
   * В появившемся окне щелкните **OK**
   * В появившемся окне щелкните **Discard** (что бы закрыть систему моделирования без сохранения результатов).

## Этап 4. Анализ структуры RTL описания проекта

1. В разделе **Flow** **Navigator**, в папке **RTL** **Analysis** откройте папку **Open** **Elaborated** **Design** и щелкните по строке **Schematic**. В появившемся окне щелкните кнопку **ОК**.

*Пакет Vivado автоматически осуществит компиляцию (Elaboration) RTL описания проекта и отобразит структурную схему RTL описания (так, как ее «понял» пакет).*

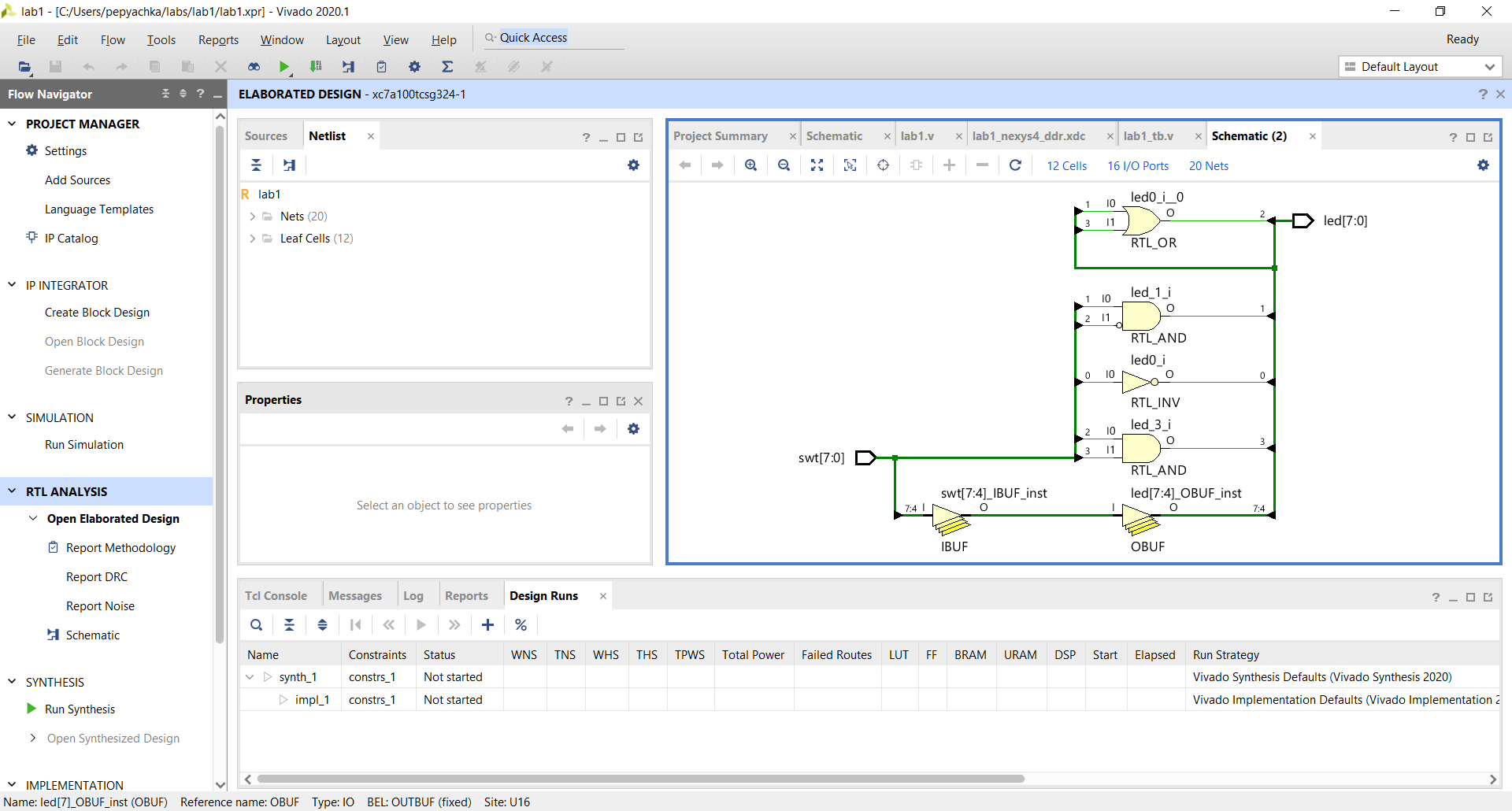


Рисунок 12 - Структурная схема RTL описания

## Этап 5. Синтез проекта и анализ результатов

1. Запустите процедуру Синтеза:

* В разделе **Flow Navigator**, в папке **Synthesis** щелчком запустите команду **Run Synthesis**.
* В появившемся окне **Launch** **Runs** (если его появление не было запрещено установками пакета) оставьте все установки по умолчанию и щелкните кнопку **ОК**.

*Пакет Vivado начнет процедуру синтеза описания проекта, приведенного в файле lab1.v (в выполняемой лабораторной работе lab1.v — это единственный файл, и он является файлом верхнего уровня в иерархии описания. Если в описании проекта имеется несколько файлов, то один из них – файл верхнего уровня иерархии описания, а все остальные нижнего, или нижних уровней. Пакет автоматически находит все файлы в иерархии описания проекта при реализации процедуры синтеза).*

1. После окончания процедуры синтеза появится окно **Synthesis Completed**

* Выберите опцию Open Synthesized Design (так как мы хотим провести анализ результатов синтеза до начала процедуры реализации проекта) щелкните по кнопке OK.

1. Откроется раздел *Synthesized Design:*

* В разделе **Synthesized** **Design** выберите закладку **Project** **Summary** и далее закладку **Overview**.

*Если Вы не видите закладку* ***Project******Summary*** *выполните команду* ***Layout******>******Default******Layout****.*

* Отсоедините закладку **Project** **Summary** от окна раздела **Synthesized** **Design** - щелкните кнопку **Float** и затем разверните появившееся окно на весь экран.

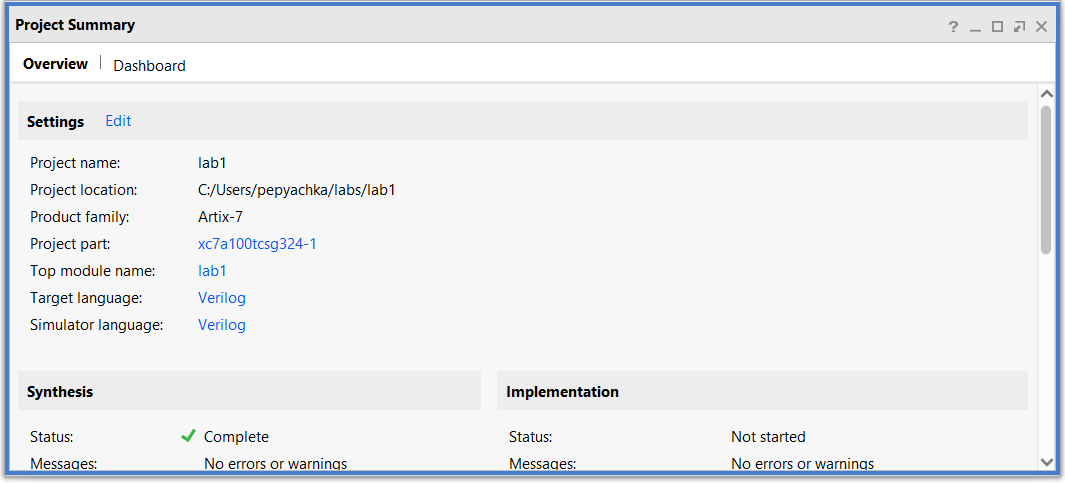


Рисунок 13 - Project Summary(1)

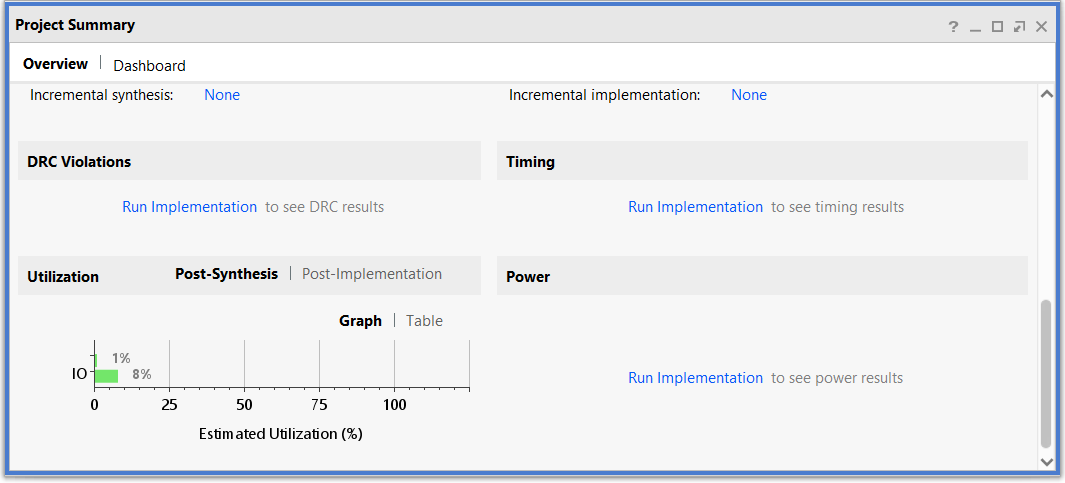


Рисунок 14 - Project Summary(2)

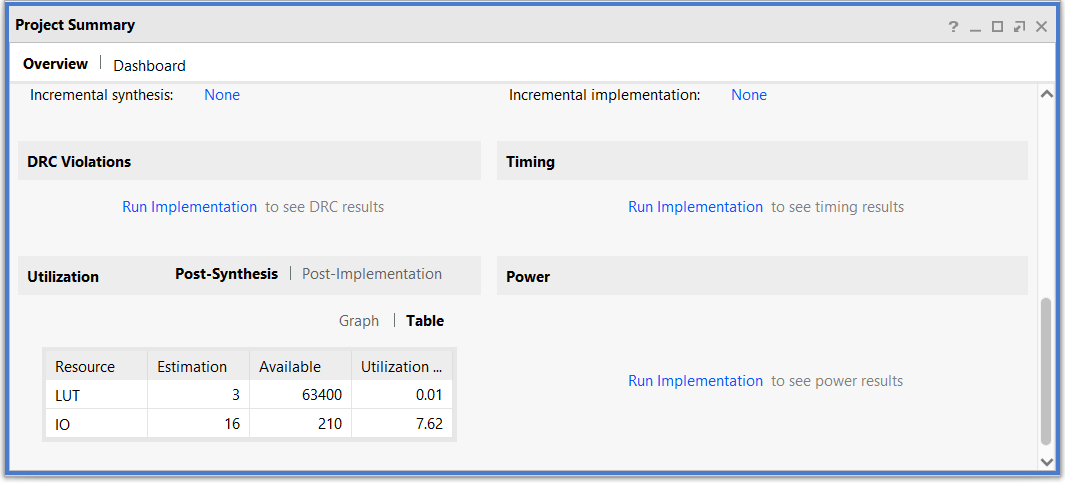


Рисунок 15 - Project Summary(3)

1. Откройте окно со схемным представлением результатов синтеза: в разделе **Flow Navigator**, в папке **Synthesis** разверните подпапку **Open Synthesized** **Design** и щелчком запустите команду **Schematic**

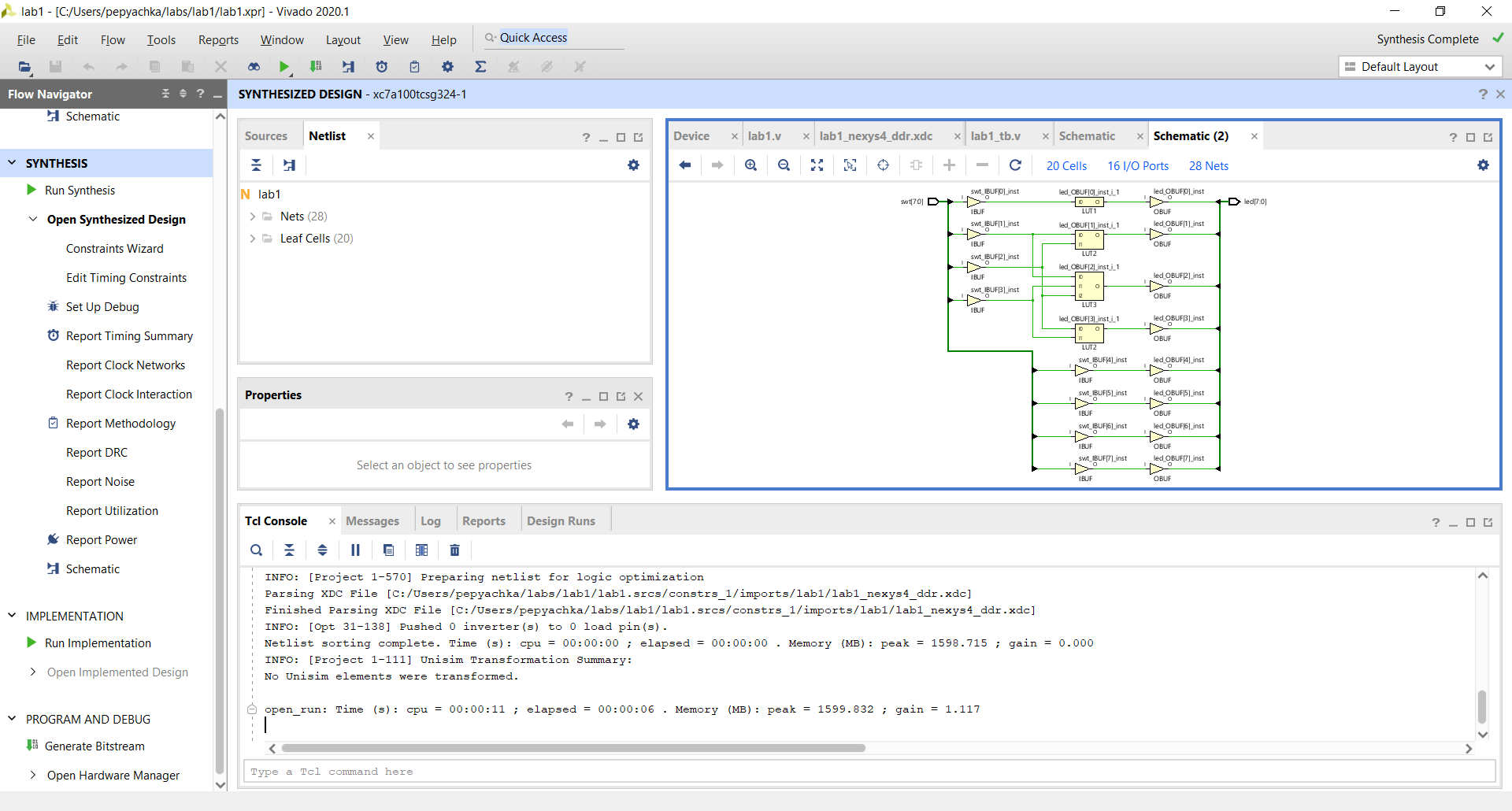


Рисунок 16 - Схемное представление результатов синтеза

1. Откройте детальный отчет об использованных ресурсах: в разделе **Flow** **Navigator**, в папке **Synthesis** разверните подпапку **Open Synthesized Design** и щелчком запустите команду **Report Utilization**. В появившемся окне щелкните кнопку **ОК**.

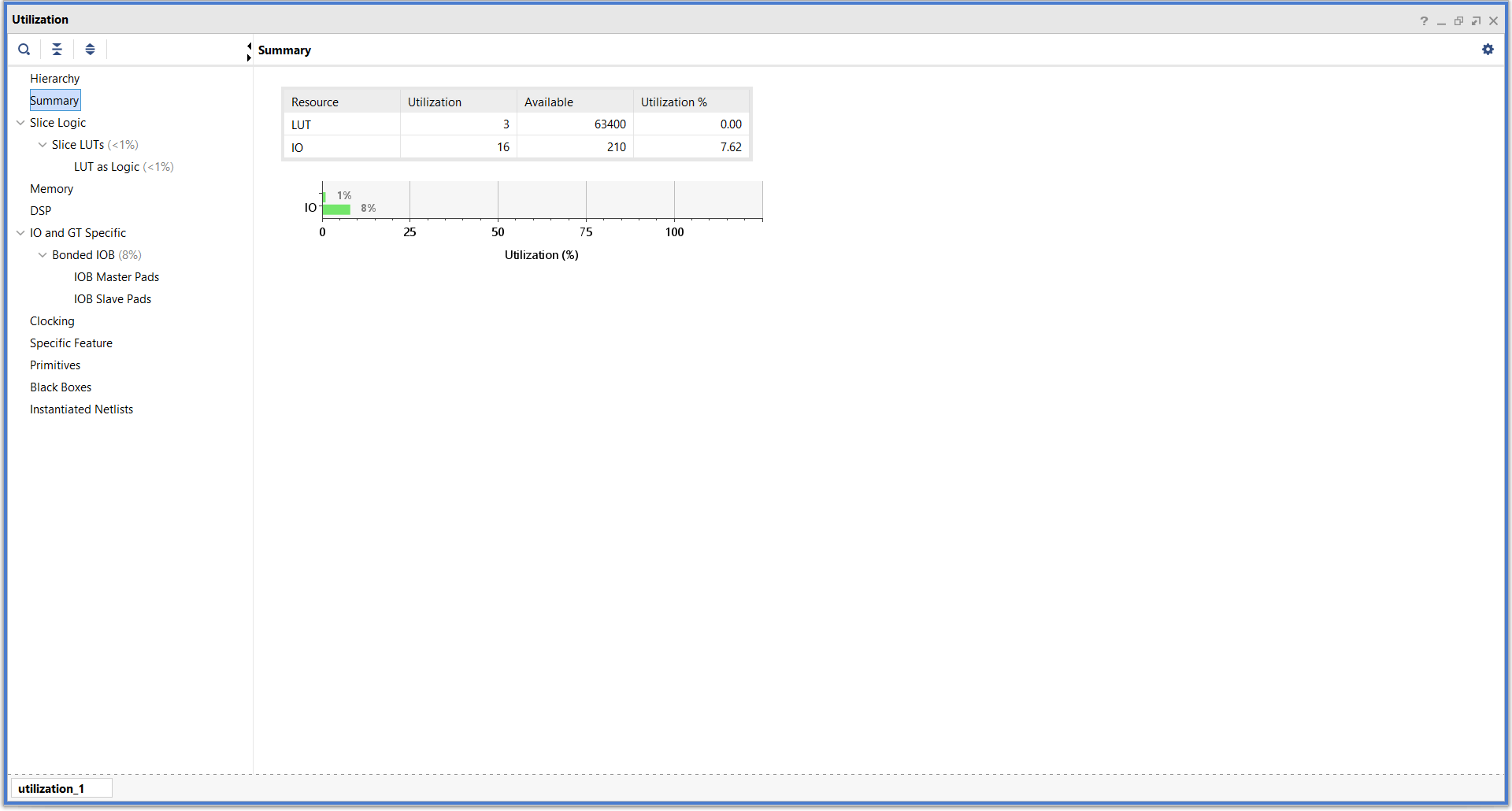


Рисунок 17 - Детальный отчет об использованных ресурсах

*В* *разделе Summary приведена табличная и графическая формы представления оценок (после этапа синтеза) затраченных логических ресурсов и элементов ввода вывода для реализации проекта.*

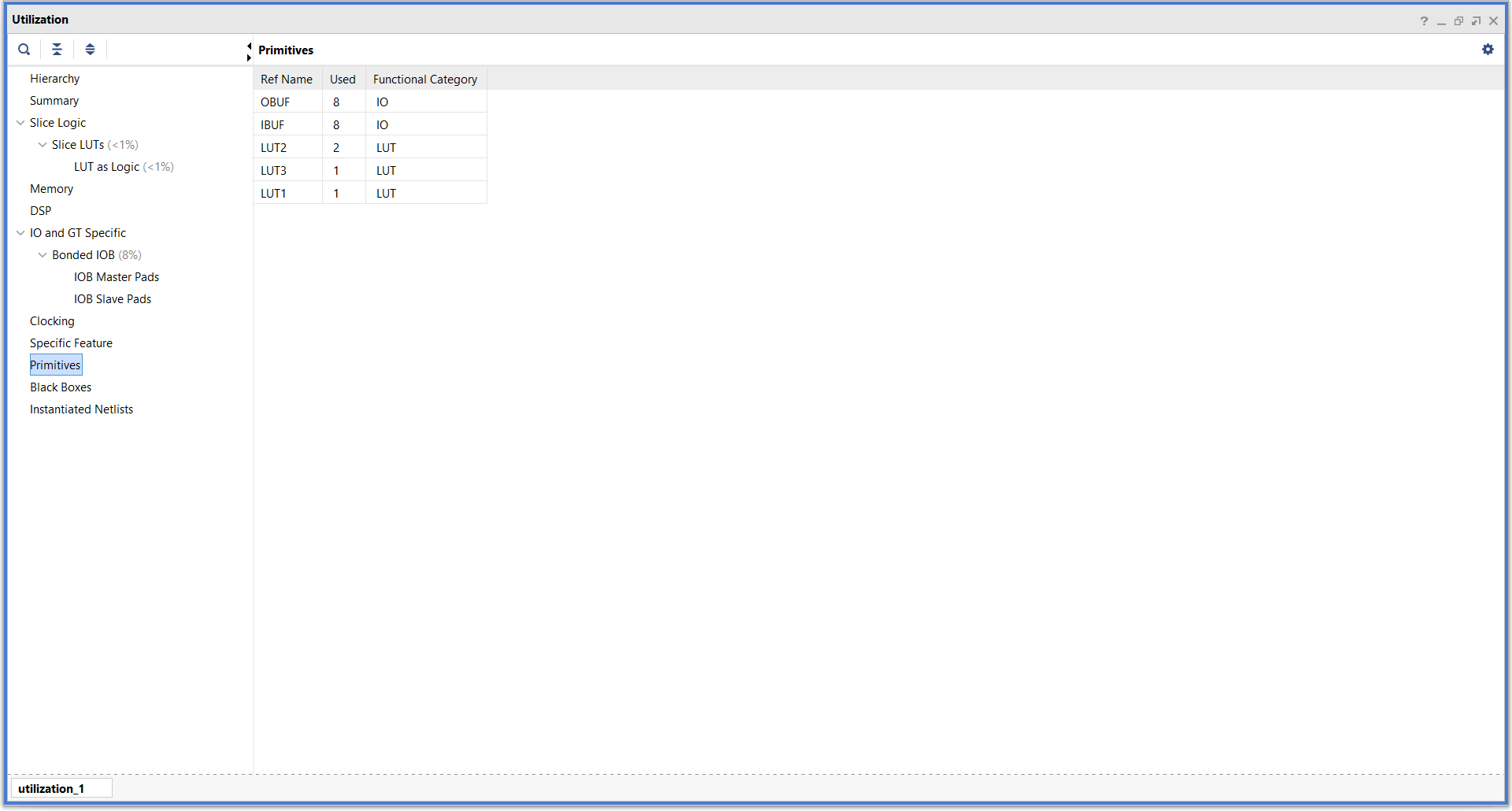


Рисунок 18 - Примитивы, использованные для отображения созданной после синтеза структурной схемы

## Этап 6. Реализация проекта и анализ результатов

1. В разделе **Flow Navigator**, в папке **Implementation** щелчком запустите команду **Run Implementation**. В появившемся окне (если его появление не было запрещено настройками) оставьте все назначения без изменения и щелкните кнопку **ОК**.
2. После окончания процедуры реализации проекта появится окно **Implementation Completed**.

* Выберите опцию **Open** **implemented** **design** и щелкните кнопку **OK**.

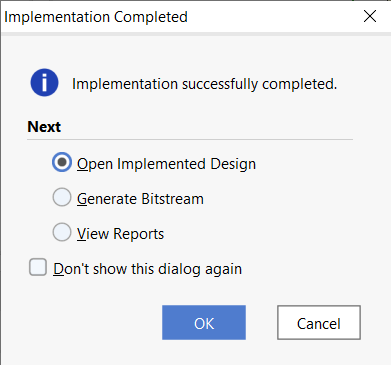


Рисунок 19 - Implementation Completed

1. Появится окно **Implemented** **Design**

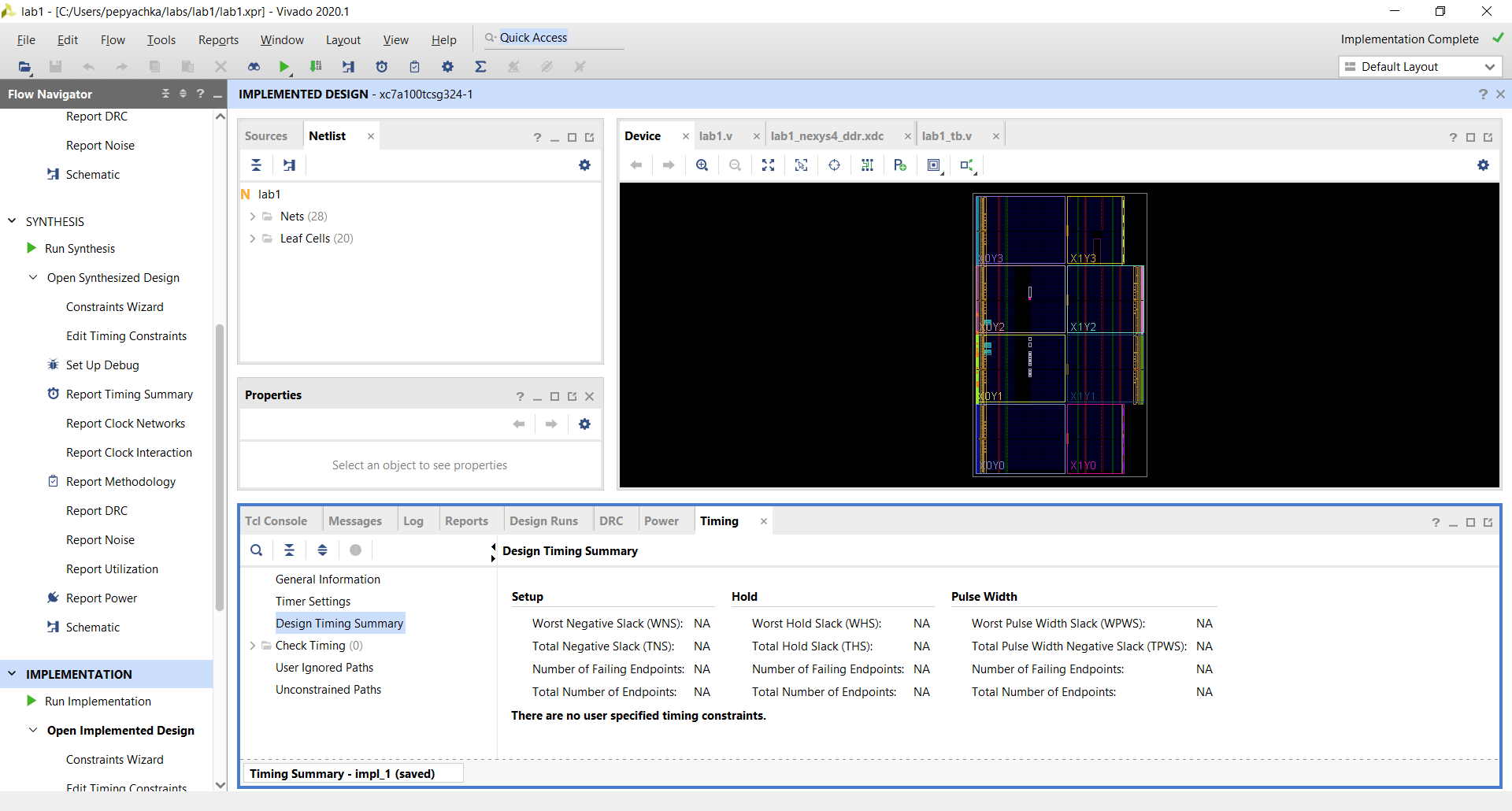


Рисунок 20 - Implemented Design

* В окне **Implemented** **Design** на закладке **Device** найдите одну из цепей (например, **swt\_IBUF[0]**) и отобразите ее трассировку (на рисунке отображена трассировка для FPGA платы Nexys4 DDR).

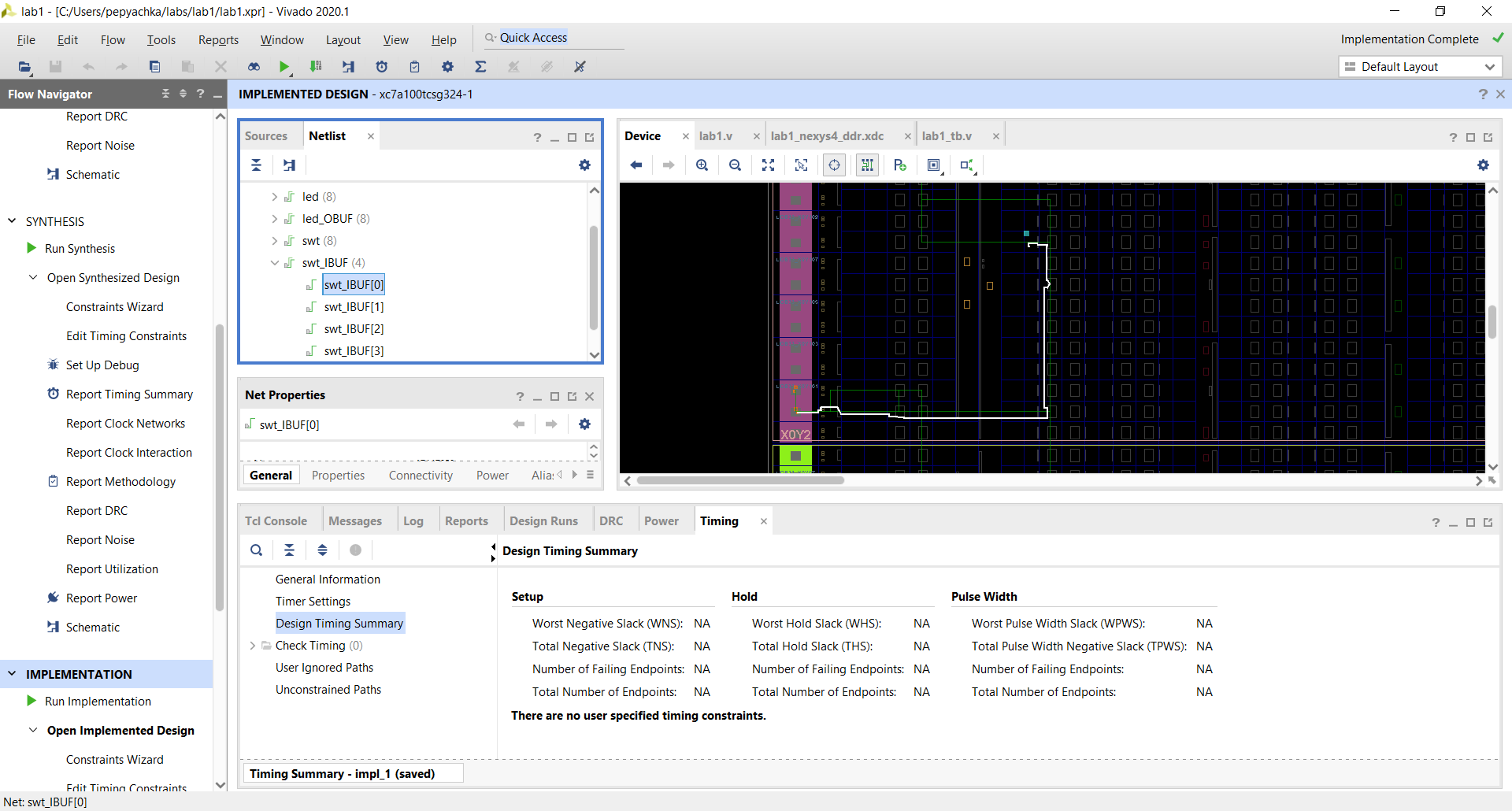


Рисунок 21 - Трассировка одной из цепей

1. Откройте детальный отчет об использованных ресурсах: в разделе **Flow** **Navigator**, в папке **Implementation** разверните подпапку **Open Implemented** **Design** и щелчком запустите команду **Report Utilization**. В появившемся окне щелкните кнопку **ОК**.

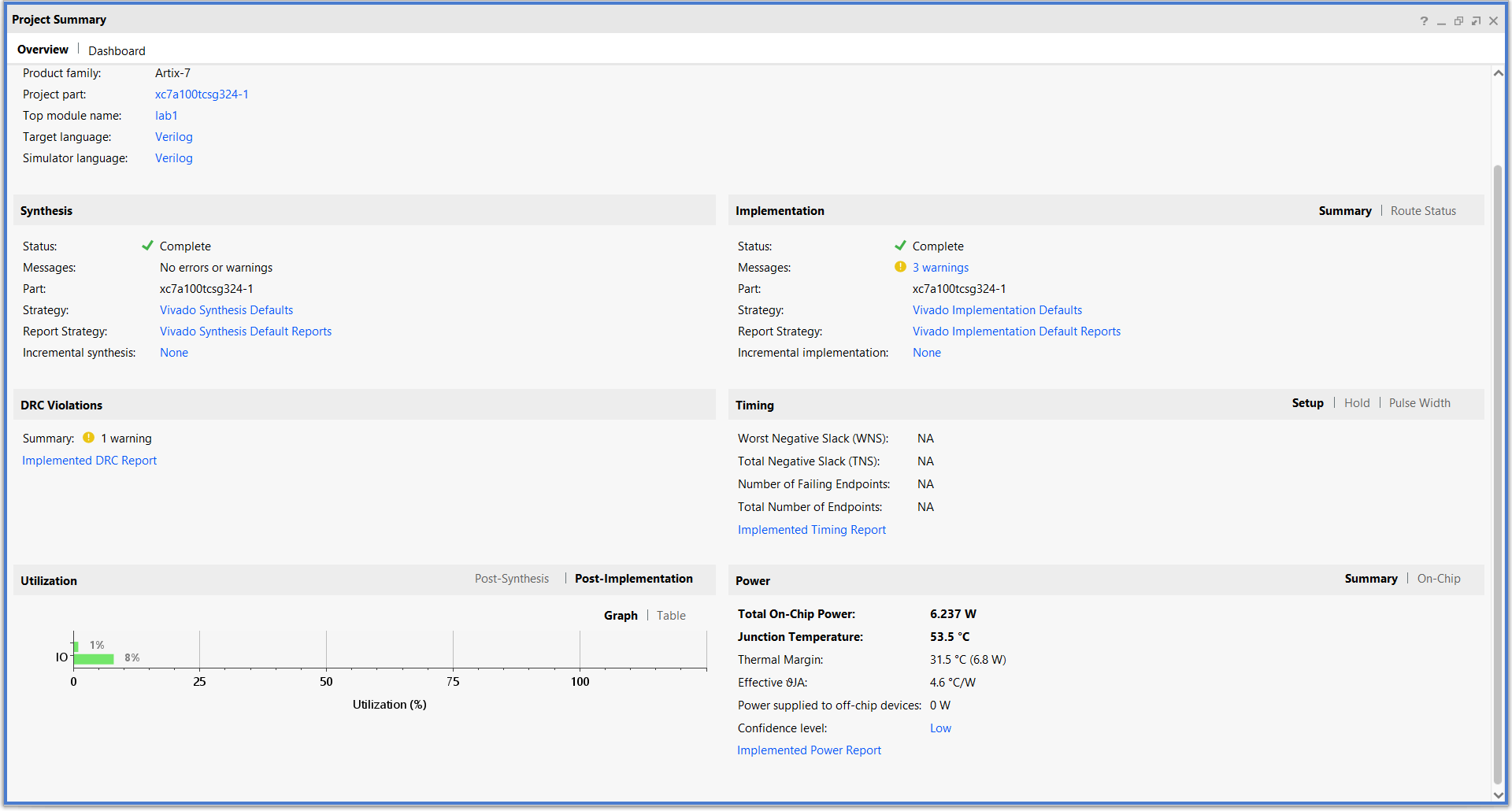


Рисунок 22 - Табличная и графическая форма представления оценок

*В разделе Summary приведена табличная и графическая формы представления оценок (после этапа реализации) затраченных логических ресурсов и элементов ввода вывода.*

1. В окне раздела **Implemented** **Design** выберите (нижняя панель раздела) закладку **Reports**. Если закладка **Reports** не отображается, то выполните команду *Window (панель инструментов пакета Vivado) => Reports*.

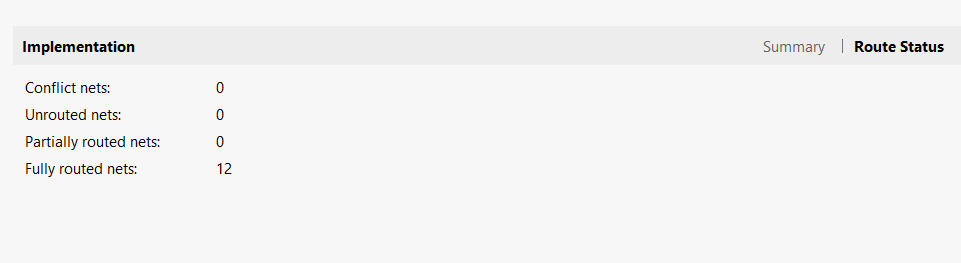


Рисунок 23 - Reports

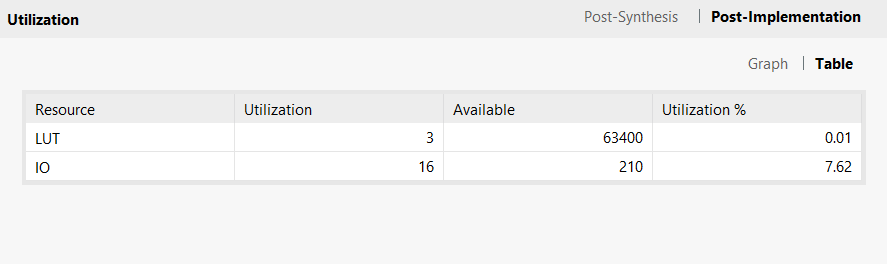


Рисунок 24 – Reports



Рисунок 25 – Reports

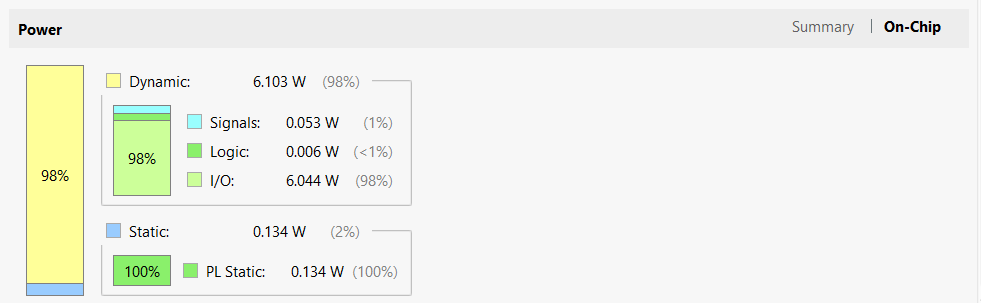


Рисунок 26 - Reports

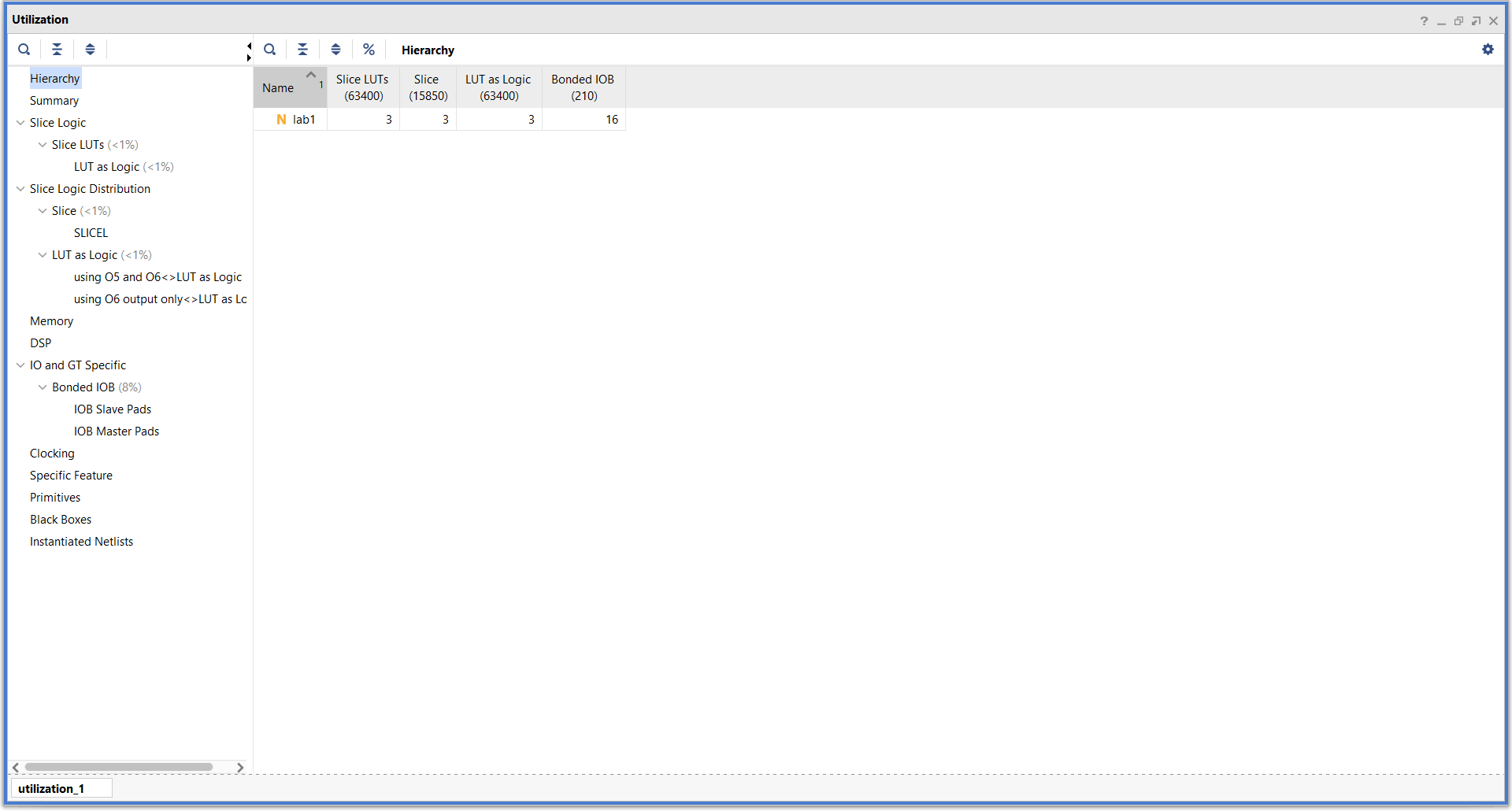


Рисунок 27 – Reports

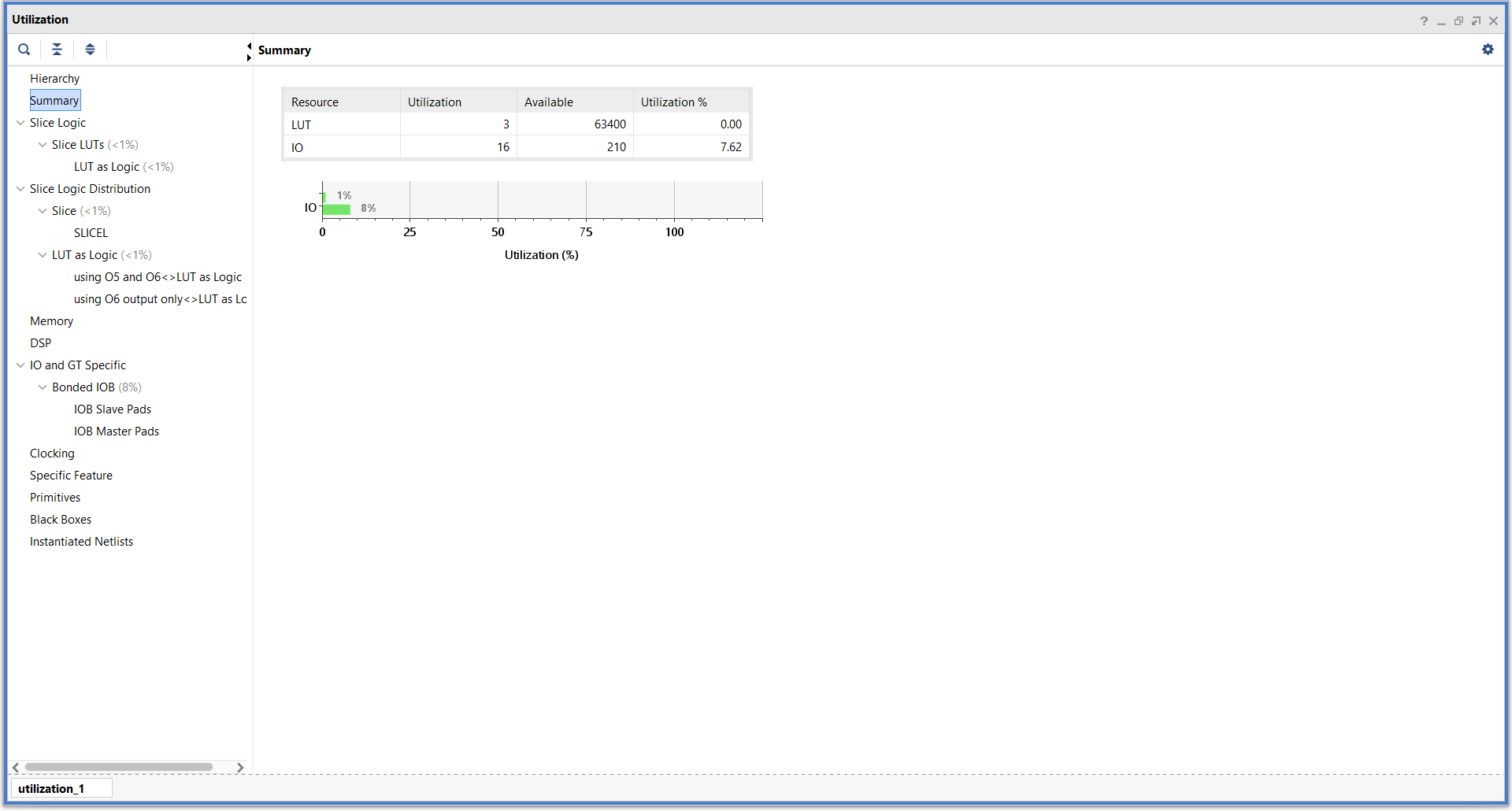


Рисунок 28 – Reports

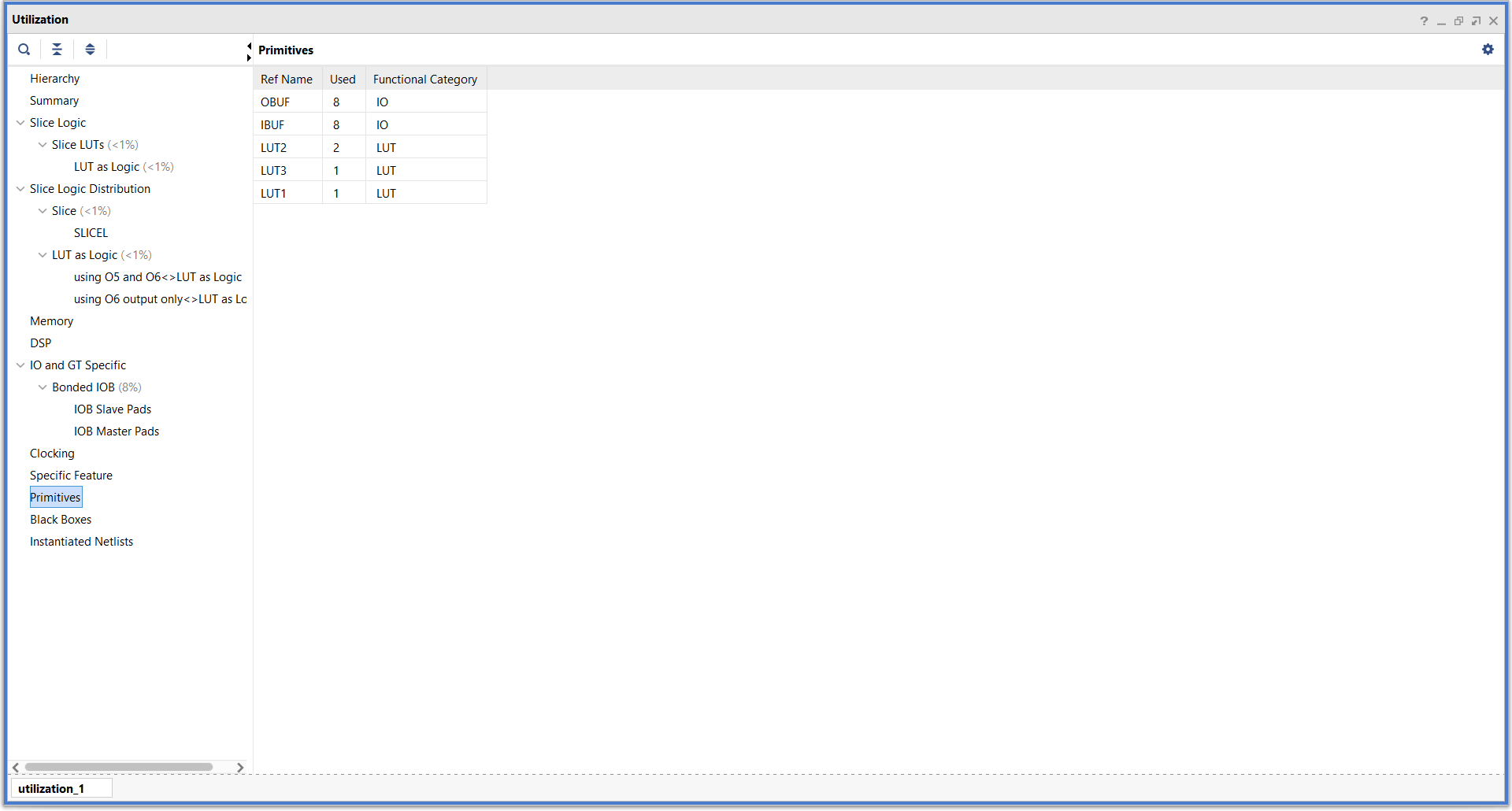


Рисунок 29 - Reports

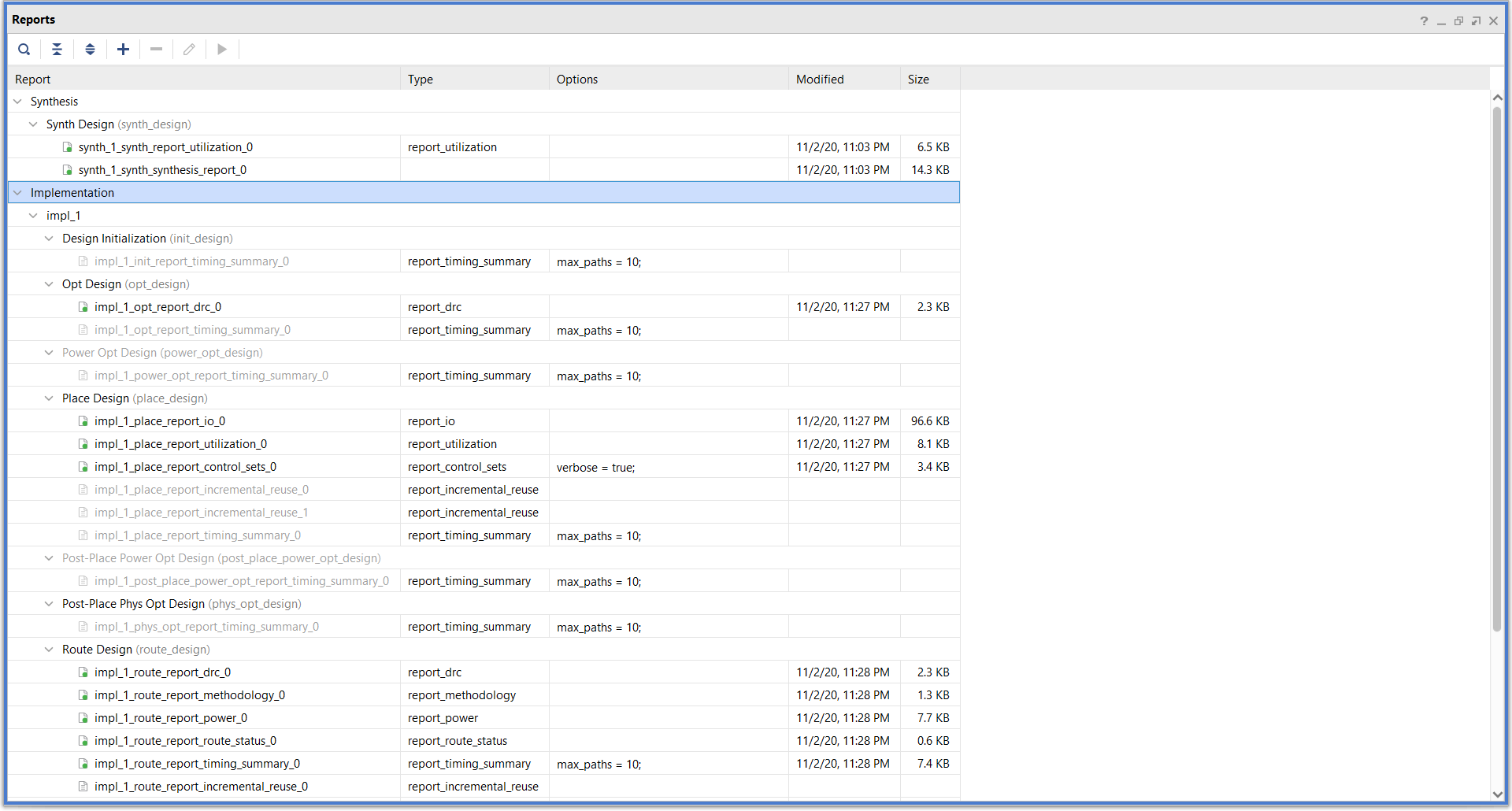


Рисунок 30 - Отчеты по выполненным этапам Синтеза и Реализации проекта

*В окне будут представлены отчеты по выполненным этапам Синтеза и Реализации проекта. Часть отчетов, выделенных серым цветом – недоступны, так как соответствуют под этапами, которые, в соответствии с настройками компилятора, были пропущены. Отчеты открываются двойным щелчком левой клавиши мыши.*

## Этап 7. Моделирование реализованного проекта

1. В разделе *Flow Navigator*, в папке *Simulation*, щелчком выберите команду **Run Simulation**
2. Запустите процедуру **Run Post-Implementation Timing Simulation** *(Временное моделирование после этапа Реализации).*

*Пакет Vivado запустит процедура моделирования и после моделирования 200ns (модельное время) в окне Simulation Post-Implementation Simulation Timing будут представлены результаты моделирования.*

* В окне ***Simulation*** ***- Post-Implementation Simulation - Timing*** щелчком выберите окно с временной диаграммой.
* Щелчком по кнопке **Float** отсоедините его от окна **Simulation**, а затем разверните на весь экран.
* Наведите указатель мыши на курсор и нажмите правую клавишу мыши. В появившемся меню выберите команду **Go To.**
* В появившемся поле **Go To Time** введите 90ns.
* На клавиатуре нажмите клавишу Enter. Курсор переместится на 90ns.
* Используя команды Zoom Out и Zoom In измените масштаб временной диаграммы так, чтобы на экране был виден промежуток времени от 90ns до 100 ns.
* Создайте маркер положения курсора на отметке 90ns: выполните команду Add Marker. Под курсором появится отметка маркера.
* Щелчком выделите шину leds[7:0] выполните команду Next Transition. Курсор перейдет к моменту изменения сигнала на шине.
* Создайте маркер положения курсора: выполните команду Add Marker. Под курсором появится отметка маркера.
* Щелчком выделите шину leds[7:0] выполните команду Next Transition. Курсор перейдет к следующему моменту изменения сигнала на шине.
* Создайте маркер положения курсора: выполните команду Add Marker. Под курсором появится отметка маркера.
* Щелчком выделите шину e\_led[7:0] выполните команду Next Transition. Курсор перейдет к моменту изменения сигнала на шине – к отметке 100ns.
* Щелчком выберите самый левый маркер. Это будет маркер отсчета времени. Внизу окна временной диаграммы появится шкала измерения интервалов времени (Floating Ruler). На ней будут отмечены задержки маркеров и курсора относительно маркера отсчета времени.
* Щелчком разверните содержимое шины leds[7:0].
* В результате должна получиться временная диаграмма похожая на временную диаграмму, приведенную на рисунке ниже.

Изображение выглядит как ноутбук, компьютер, монитор, сидит

Автоматически созданное описание

*Рисунок 31 - Временная диаграмма*

1. Закройте систему моделирования

* Выполните команду **File > Close Simulation**.
* В появившемся окне щелкните кнопку **OK**.
* В появившемся окне щелкните **Discard** (что бы закрыть систему моделирования без сохранения результатов).

# **Вывод**

В ходе выполнения лабораторной работы мы узнали, как:

• создавать проект, основанный на HDL описании,

• выбирать целевую платформу (как и конкретную FPGA, так и плату с этой микросхемой),

• использовать созданный заранее файла Xilinx Design Constraint (XDC) - файл для задания выводов микросхемы,

• осуществлять моделирование проекта (поведенческое и с временными параметрами),

• синтезировать проект,

• создавать конфигурационный файл,

• конфигурировать FPGA на плате и осуществлять проверку проекта.