Министерство науки и высшего образования Российской Федерации

Томский государственный университет
систем управления и радиоэлектроники

А.А. Бомбизов

ЗНАКОМСТВО СО СРЕДОЙ. БАЗОВАЯ ЛОГИКА

Методические указания к выполнению
лабораторной и самостоятельной работы
по дисциплине «Проектирование систем на кристалле»

Томск
2020

УДК 681.3 (075.32)
ББК 32.973стд1-02
Б 803

Рецензент:
Тренкаль Е.И., доцент кафедры конструирования узлов и деталей радиоэлектронной аппаратуры ТУСУР, канд. техн. наук

Бомбизов, Александр Александрович
Б 803 Знакомство со средой. Базовая логика: методические указания к выполнению лабораторной и самостоятельной работы по дисциплине «Проектирование систем на кристалле» / А.А. Бомбизов. – Томск. гос. ун-т систем упр. и радиоэлектроники, 2020. – 15 с.

Настоящее методическое указание по выполнению лабораторной и самостоятельной работы по дисциплине «Проектирование систем на кристалле».

Методическое пособие содержит краткое описание порядка создания проекта для ПЛИС и основ структурного проектирования на базе реализации комбинационной логической схемы.

Одобрено на заседании каф. КУДР, протокол № 234 от 5 марта 2022 г.

УДК 681.3 (075.32)
ББК 32.973стд1-02

1 Введение

В течение последних лет в мире наблюдается заметный рост числа проектов, использующих программируемые логические интегральные схемы, типа FPGA. Они хоть и уступают полузаказным микросхемам типа ASIC, но при этом обладают характеристиками и возможностями достаточными для разработки сложных систем. В последних поколениях FPGA появились серии, ориентированные на создание «систем-на-кристалле», предлагающие разработчикам такие системные функции, как встроенные процессоры, высокоскоростные последовательные интерфейсы, специализированные арифметические модули большой разрядности, встроенные блоки памяти различной конфигурации и назначения и др.

Целью работы является освоение порядка создания проекта для ПЛИС и основ структурного проектирования на базе реализации комбинационной логической схемы.

2 Краткая теория

Одним из мировых лидеров производства ПЛИС и средств разработки для них является компания Xilinx. В направлении FPGA в настоящий момент актуальными являются семейства Artix-7, Kintex-7 и Virtex-7. В рамках настоящего лабораторного курса используется отладочная плата Basys 3 компании Digilent [1]. Установленный вариант FPGA представляет собой XC7A35TCPG236-1 [2] семейства Artix-7.

При проектировании устройств на базе ПЛИС используются различные уровни абстракции:

1) поведенческий уровень (Behavioral Level) – описание устройства при помощи последовательных алгоритмов, работающих одновременно и согласованно;

2) уровень регистровых передач (RTL – Register-Transfer Level) – описание устройства при помощи заданного набора операций преобразования значений, хранящихся в регистрах;

3) схемный уровень (Gate Level) – описание устройства при помощи схемы (netlist) из функциональных элементов в заданной библиотеке элементов.

Высокоуровневое описание составляется на одном из специализированных компьютерных языков описания аппаратуры, используемых для описания поведения и моделирования электронных схем. Как правило, используются VHDL, Verilog, SystemC и др.

Verilog HDL поддерживает иерархическое описание проекта. На самом верхнем уровне лежит модуль, содержащий экземпляры модулей следующего, более низкого уровня иерархии, а также связи между ними. Модули более низкого уровня в свою очередь могут содержать экземпляры модулей следующего уровня и т. д. Аналогом модуля в классическом языке программирования может быть типичная подпрограмма, например, функция *main* является модулем верхнего уровня.

Объявление модуля заключено между двумя ключевыми словами *module* и *endmodule*. Синтаксис приведен ниже:

module <имя модуля>(<список портов>)
<объекты модуля>
endmodule

Имя модуля является обязательным, это имя, под которым модуль представлен в описании. Список портов является опциональным, он определяет порядок портов при создании экземпляра модуля.

Объекты модуля перечислены ниже:

- параметры;
- входные, выходные, двунаправленные порты;
- цепи;
- регистры;
- переменные;
- экземпляры примитивов;
- экземпляры модулей;
- непрерывные присвоения;
- начальные состояния;
- процесс-блоки;
- и другие.

Общий синтаксис объявления портов модуля приведен ниже:

<направление порта> <[разрядность]> <имя порта 1>, <имя порта 2>,…;

Направление порта определяется ключевыми словами *input*, *output*, *inout*, для того чтобы определить какие порты являются входными, выходными, двунаправленными. Опциональное поле *разрядность* является диапазонным значением и определяется следующим образом: *<msb>:<lsb>*, где *msb* – старший значащий бит (most significant bit), а *lsb* – младший значащий бит (least significant bit).

Как правило, модуль описывается в отдельном файле с расширением .v и названием, совпадающем с названием модуля. Допустим, необходимо описать модуль следующего вида (рисунок 1).

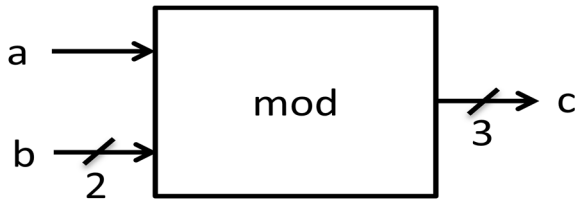


Рисунок 1 – Внешний вид модуля

Пример определения модуля приведен ниже:

```
module mod(input a, input[1:0] b, output[0:2] c);
  // описание
endmodule
```

где порт *a* является одноразрядным (однопроводным), *b* –двухразрядным (двухпроводным), *c* – трехразрядным (трехпроводным) с обратным порядком следования значащих бит.

Если не указаны направления портов при объявлении модуля, то они должны быть определены в списке объявления.

```
module mod(a, b, c);
  input a;
  input [1:0] b;
  output[0:2] c;
  // описание
endmodule
```

Обычно различают два подхода к описанию:

- структурный – явно описываются экземпляры модулей более низкого уровня и связи между ними;
- поведенческий – без явного описания структуры. Задается поведение (реализуемые функции) модулей и связи между ними.

В текущей лабораторной работе будет использоваться только структурный подход.

Для определения модулей и их использования могут потребоваться цепи (линии связи, провода, проводники…) между ними. Цепи являются типами данных и предназначены для связи структурных объектов, например, для соединения между собой логических вентилей, модулей более низкого уровня, встроенных примитивов и др. Цепи не удерживают своего состояния, оно должно непрерывно удерживаться (continuous

assignment) выходом логического элемента или другого модуля. Если к цепи не подключен источник, то цепь переходит в состояние высокого импеданса.

Цепи объявляются следующим образом:

wire <[разрядность]> <имя цепи 1>, <имя цепи 2>, …

Пример объявления:

wire w1;

или с учетом разрядности

wire[2:0] w1;

где *w1* – это имя цепи.

Нужно отметить, что порты модуля по умолчанию являются цепями.

В определении цепей было упомянуто ключевое слово «примитив», под которым понимается отдельная иерархическая единица, аналогичная модулю в том отношении, что на любом уровне иерархии она занимает равные права с модулем. При этом у примитива не может быть более одного выхода, причем одноразрядного.

Общий синтаксис при создании экземпляра примитива следующий:

<имя примитива> <собственное имя>(<список соединений>);

Имя примитива означает уникальное имя ранее определенного или встроенного примитива. Собственное имя – имя экземпляра созданного примитива. При создании экземпляра примитива не обязательно указывать собственное имя. Список соединений состоит из имен цепей, следующих друг за другом в порядке, определенном порядком портов модуля. Имена соединительных цепей разделяются запятыми.

Существуют встроенные примитивы *and*, *nand*, *or*, *nor*, *xor*, *xnor*, *not*, реализующие логические функции в соответствии с их названиями. При создании экземпляров примитивов этой группы, первый порт в списке всегда является выходом. Количество входов может быть больше двух (для *not* всегда один), при этом системой синтеза примитив будет представлен как каскадное включение двухвходовых логических элементов.

Пример использования логического элемента И:

and (out,in1,in2);//без имени экземпляра

and a1(out,in1,in2);//с именем экземпляра

Комбинированные логические схемы могут моделироваться как с использованием логических элементов, так и с использованием непрерывных присвоений. Это означает, что значение переменной будет изменяться сразу же, как только изменится значение хотя бы одного

операнда справа от присвоения. Ниже приведен формальный синтаксис такого присвоения:

assign <список аргументов>;

Список аргументов содержит не просто список переменных, но и список переменных с присвоенными им значениями, которые могут быть константой, выражением, результатом выполнения функции, операцией конкатенации, условным оператором. Аналогичным непрерывному присвоению является инициализация цепи при её описании.

Примеры непрерывного присваивания:

```
//при объявлении
wire w1=1'b1;//присваивается одноразрядное бинарное значение 1
wire w2=1'b0;//присваивается одноразрядное бинарное значение 0
wire w3=w1;//w3 изменится при изменении w1 даже после текущей строки
…
//при реализации
assign w3=w1|w2;
```

После того как модули описаны, должно быть выполнено общее построение проекта, которое состоит из компиляции, синтеза, размещения и трассировки.

Компиляция в представлении ПЛИС – это трансляция пригодного к синтезу HDL-описания на уровне RTL в технологически независимый список соединений, примитивами которого являются общие для всех ПЛИС элементы. В результате компиляции должна быть создана схема примитивов, каждый из которых отражает только функциональное назначение, например логическое И или логическое ИЛИ, но не отражает способ физической реализации данной функции [3].

Затем должен быть выполнен синтез, который представляет собой оптимизацию и размещение полученного на этапе компиляции списка соединений общего вида в технологические ячейки с использование, где это необходимо, специальных возможностей выбранной пользователем архитектуры. Полученный в результате синтеза список соединений передается средствам размещения и трассировки среды разработки. После завершения выполняется генерация файла прошивки (bitstream) для непосредственного программирования ПЛИС.

Разработка проекта для ПЛИС компании Xilinx выполняется в среде проектирования Vivado 2019. Для создания проекта необходимо выбрать пункт Create Project как показано на рисунке 2. Альтернативным вариантом

является выбор пункта меню File->Project->New… В появившемся пошаговом диалоге необходимо нажать кнопку Next до появления этапа Project Name для указания имени проекта и каталога его размещения. В имени проекта и пути расположения не должно быть русских букв и пробелов.

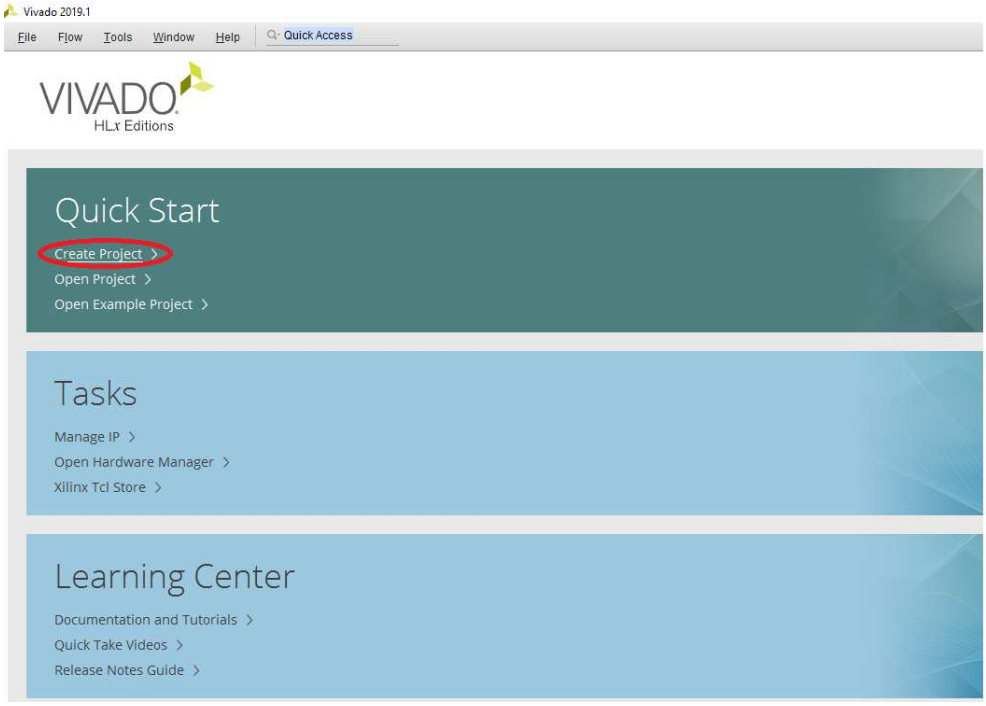


Рисунок 2 – Начальное окно

На следующем этапе необходимо выбрать тип проекта RTL Project без исходных файлов по умолчанию и нажать кнопку Next (рисунок 3).

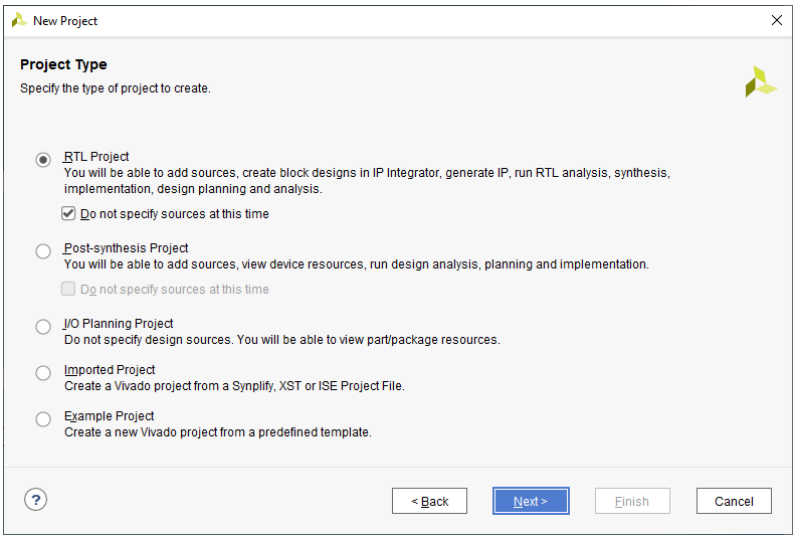
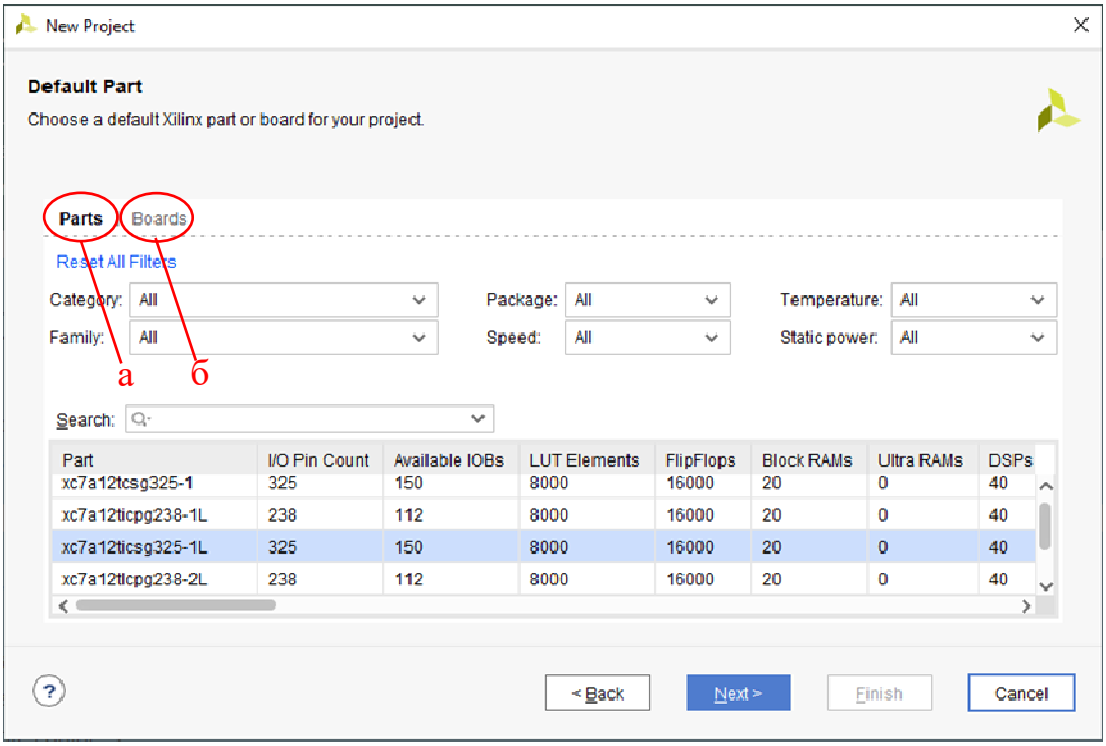


Рисунок 3 – Выбор типа проекта



Далее на этапе Default Part необходимо во вкладке Parts выбрать программируемую логическую интегральную схему согласно маркировке (рисунок 4, выноска а), либо переключиться во вкладку Boards (рисунок 4, выноска б) и выбрать требуемую отладочную плату.

По окончании необходимо нажать на кнопку Finish, после чего система завершит создание базового проекта.

Рисунок 4 – Выбор типа ПЛИС

Создание модуля выполняется путем выбора пункта меню File->Add Sources…, либо путем нажатия одной из кнопок, как показано на рисунке 5.

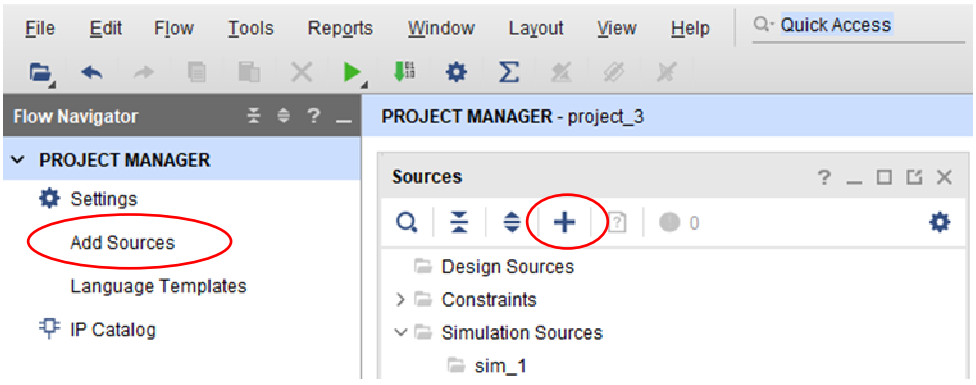


Рисунок 5 – Создание модуля

В появившемся пошаговом диалоге Add Sources необходимо выбрать пункт Add or create design sources. На следующем шаге нажать на кнопку Create File, ввести имя модуля и нажать на кнопку Finish. В следующем диалоге следует определить конфигурацию портов в соответствии с

поставленной задачей. Если порт имеет разрядность отличную от 1, то нужно установить галочку в столбце Bus и указать msb и lsb, как показано на рисунке 6.

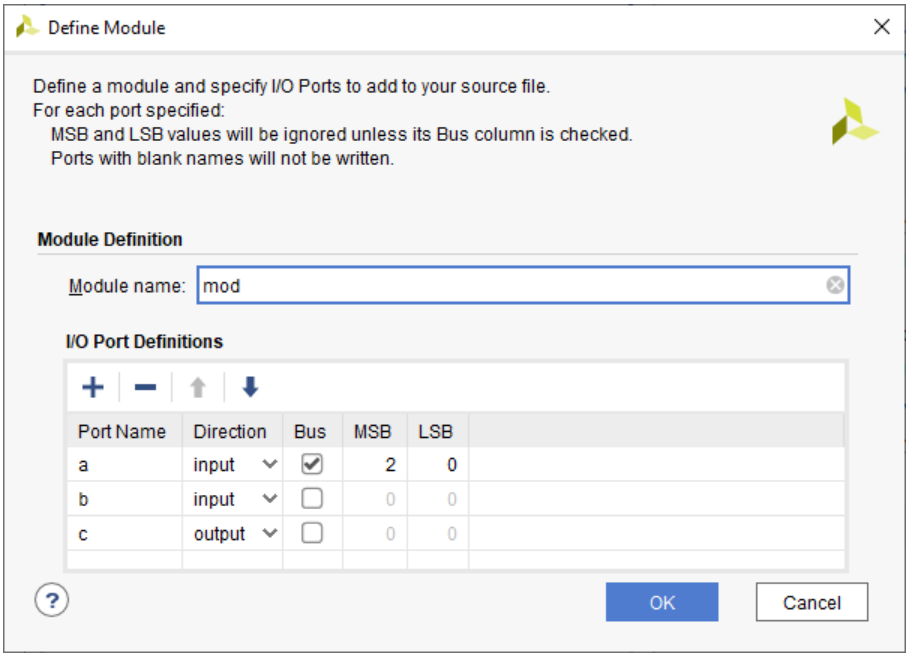


Рисунок 6 – Описание портов модуля

3 Порядок выполнения работы

В ходе данной работы необходимо реализовать комбинационную схему согласно варианту задания, на вход которой должны поступать сигналы с переключателей отладочной платы, а результат работы выводиться на подключенный к выходу схемы светодиод или группу светодиодов, если выходов несколько. Для этого потребуется выполнить следующие действия:

3.1 Изучите предложенный в п. 2 теоретический материал.

3.2 Создайте проект и модуль как описано в теоретическом материале. В модуле необходимо предусмотреть количество и виды портов согласно варианту задания.

3.3 Опишите в модуле выданную комбинационную схему, используя только встроенные примитивы и цепи.

3.4 Выполните компиляцию модуля путем выбора пункта RTL Analysis->Open Elaborated Design-> Schematic.

3.5 Верифицируйте полученный результат, то есть сравните полученную схему с вариантом задания. Если всё соответствует, то перейдите к следующему шагу.

3.6 Выполните синтез проекта путем выбора в разделе SYNTHESIS пункта Run Synthesis (F11). Процесс синтеза отображается в верхнем правом углу среды.

3.7 После выполнения синтеза откроется диалог Synthesis Completed, в котором необходимо выбрать пункт Open Synthesized Design и нажать OK.

3.8 Если появится предупреждение No Constraints file, то нажмите на кнопку Define Target. Это предупреждение о том, что не создан файл связей между внешними портами и выводами ПЛИС. В появившемся диалоге создайте и выберете требуемый файл. После чего должно появиться расположение выводов как показано на рисунке 7.

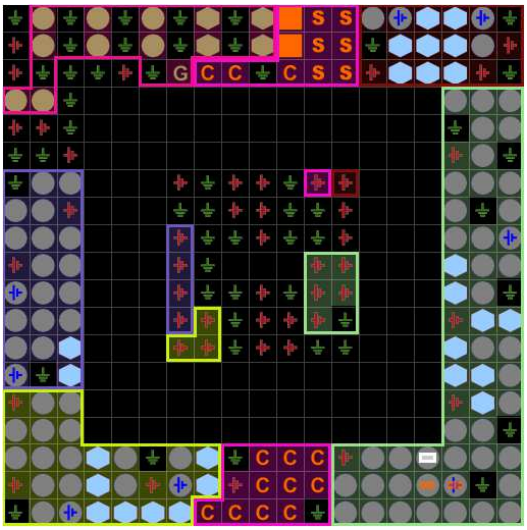


Рисунок 7 – Контакты FPGA

3.9 Определите по внешнему виду платы или по схеме [4] названия выводов, которые будут являться входами и выходами комбинационной схемы.

3.10 Во вкладке I/O Ports напротив каждого внешнего порта выберете в столбце Package Pin наименование вывода микросхемы.

3.11 В столбце I/O Std логику LVCMOS33 для установки напряжения 3,3В для логической единицы.

3.12 Сохраните созданные связи Save Constraints (Ctrl+S).

3.13 Выполните размещение и трассировку синтезированного проекта на кристалле путем выбора в разделе IMPLEMENTATION пункта Run Implementation.

3.14 По окончанию трассировки и размещения будет отображен диалог с возможностью выбора просмотра результата (Open Implementation Design) работы среды разработки.

3.15 Сгенерируйте файл прошивки путем выбора в разделе PROGRAM AND DEBUG пункта Generate Bitstream.

3.16 По окончанию генерации будет предложено открыть менеджер устройства (Open Hardware Manager).

3.17 Подключить отладочную плату к компьютеру с использованием разъема micro USB – USB-A.

3.18 В менеджере устройства подключитесь к устройству путем выбора в разделе Open Hardware Manager пункта Open Target -> Auto Connect.

3.19 Запрограммируйте ПЛИС путем выбора пункта Program Device и указания какой кристалл необходимо запрограммировать.

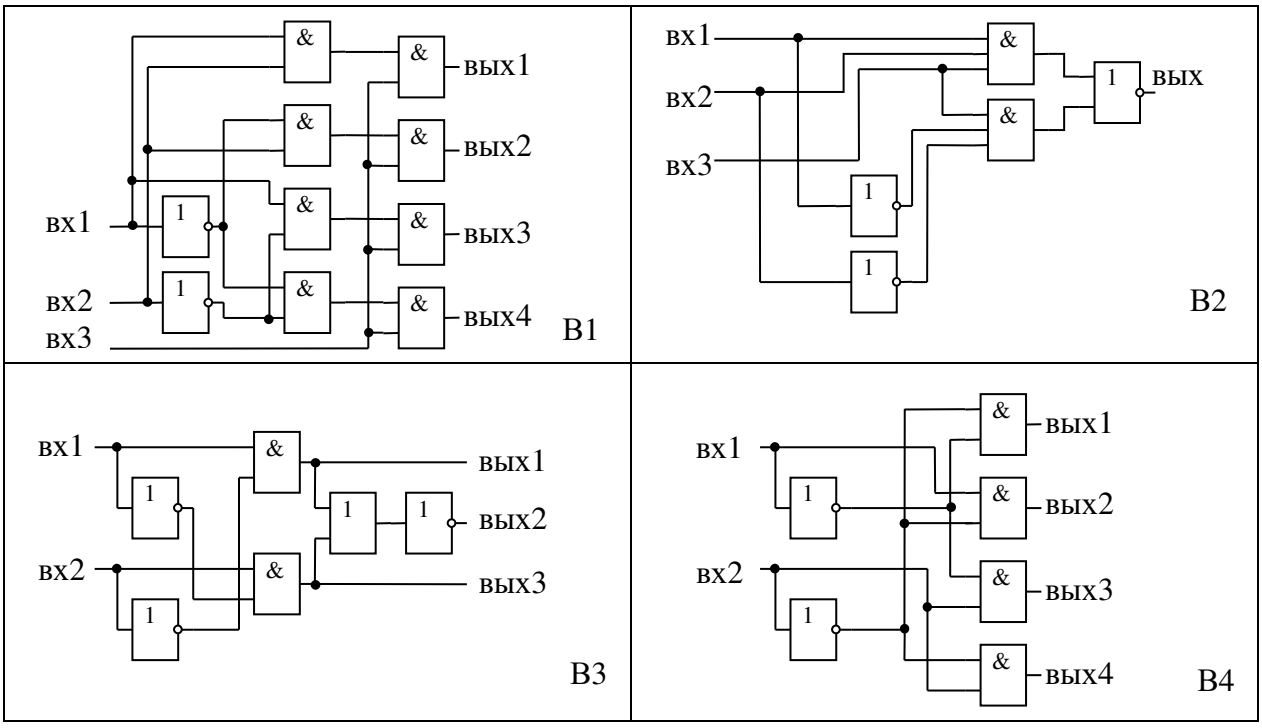
3.20 Изменяя состояние переключателей на плате, по индикации светодиода убедитесь, что схема работоспособна.

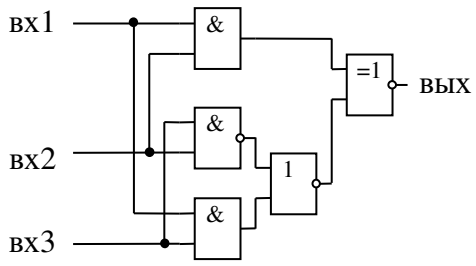
3.21 Повторите работу, используя непрерывное присваивание вместо встроенных примитивов.

3.22 Оформите отчет, содержащий титульный лист, введение, ход выполнения работы, схему логических элементов, таблицу истинности комбинационной схемы, ответы на контрольные вопросы и выводы.

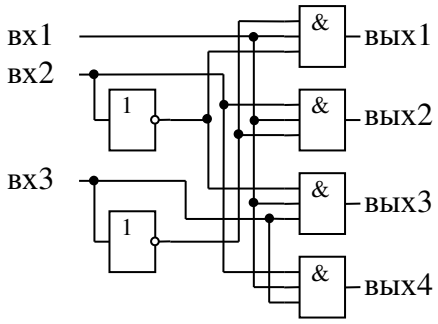
3.23 Защитите отчет у преподавателя.

Варианты

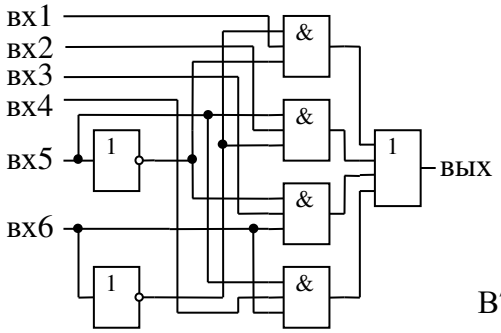




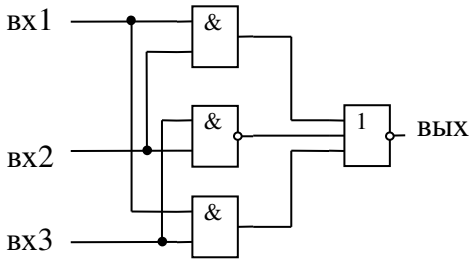
В5



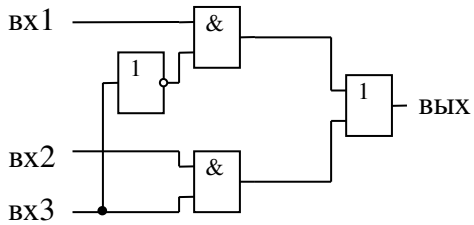
В6



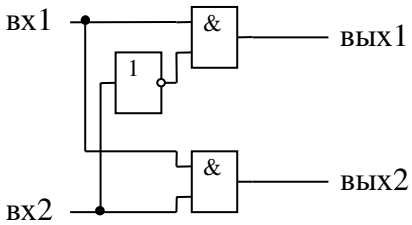
В7



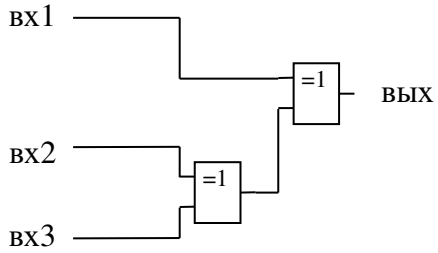
В8



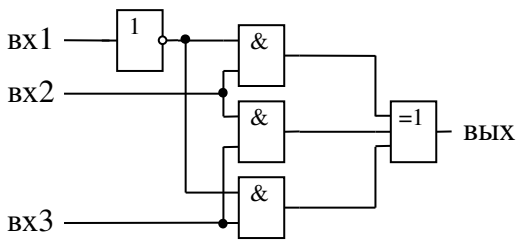
В9



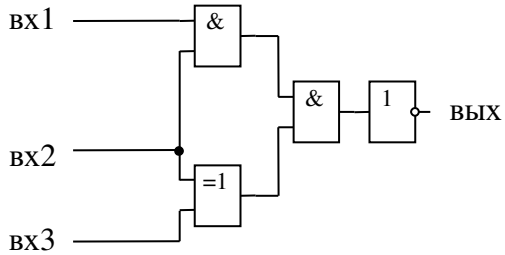
В10



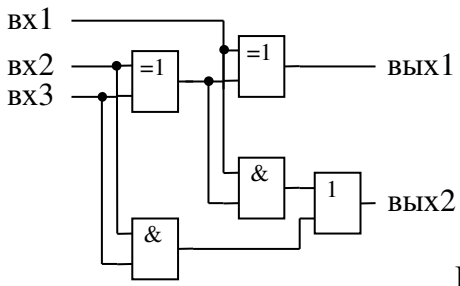
В11



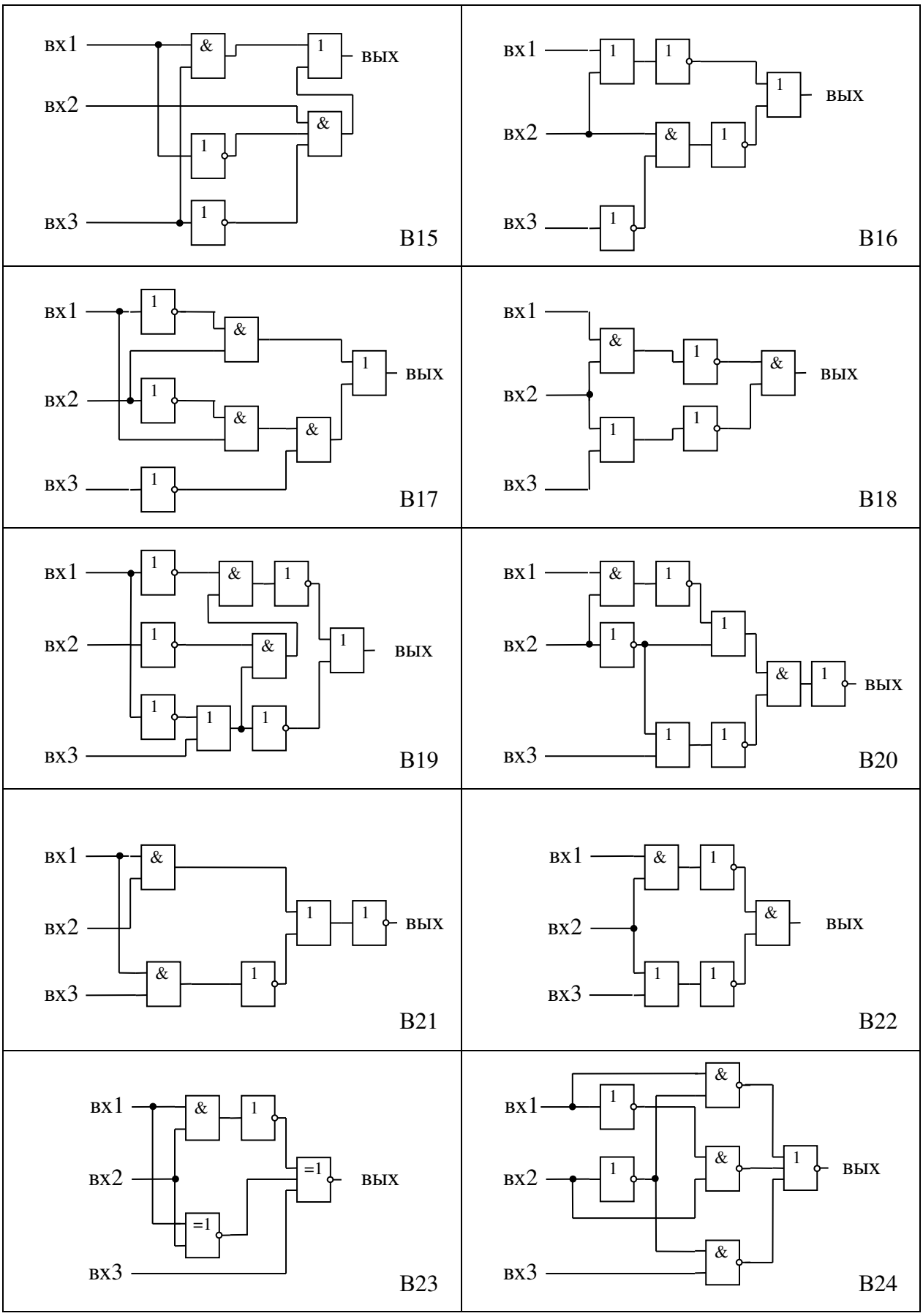
В12



В13



В14


вх1
вх2
вх3
вых
&
1
=1
В15
В16
В17
В18
В19
В20
В21
В22
В23
В24

4 Контрольные вопросы

4.1 Для чего нужна компиляция?

4.2 Что такое комбинационная схема?

4.3 Что такое встроенный примитив и как им пользоваться?

4.4 Какие варианты определения портов модуля?

4.5 Как понимать непрерывное присваивание?

4.6 Что такое разрядность цепи?

4.7 Зачем нужны цепи в ПЛИС?

4.8 Как описать цепь?

4.9 RTL – что это?

Список литературы

1 Basys3™ FPGA Board Reference Manual. URL: https://reference.digilentinc.com/_media/reference/programmable-logic/basys-3/basys3_rm.pdf (дата обращения: 31.01.2020);

2 Product Tables and Product Selection Guide. URL https://www.xilinx.com/support/documentation/selection-guides/cost-optimized-product-selection-guide.pdf#A7 (дата обращения 01.02.2020);

3 Рабоволюк А.Обзор маршрута проектирования ПЛИС FPGA Advantage компании Mentor Graphics / А. Рабоволюк // Компоненты и технологии.– 2006.– №6.– URL: https://www.kit-e.ru/articles/plis/2006_6_126.php (дата обращения 01.02.2020);

4 Basys3™ Schematic. URL: https://reference.digilentinc.com/_media/reference/programmable-logic/basys-3/basys-3_sch.pdf (дата обращения: 31.01.2020).