

## Лабораторная работа 3

### Изучение IP-компонентов и системы моделирования САПР Quartus Prime Lite

**Цель лабораторной работы:** Знакомство с базовыми возможностями схемного редактора (использование IP-компонентов), редактора конечного автомата и системы моделирования САПР Quartus Prime. Создание иерархического проекта.

Работа включает следующие этапы:

- Создание проекта
- Создание преобразователя двоичного кода в 7-сегментный код на базе файла с текстовым описанием
- Создание экземпляров счетчика и мультиплексора на базе библиотеки IP-компонентов
- Создание конечного автомата с помощью редактора конечных автоматов
- Создание дизайн - файла компонента с помощью схематического редактора
- Функциональное моделирование компонента
- Создание иерархического проекта на базе созданного компонента
- Функциональное моделирование проекта
- Назначение выводов ПЛИС
- Создание sdc – файла
- Выполнение полной компиляции проекта
- Временное моделирование проекта
- Изменение параметров схемы проекта и его повторная полная компиляция
- Конфигурирование ПЛИС и проверка работоспособности на отладочной плате.

**Алгоритм работы проекта:** В проекте реализована схема динамической индикации информации на светодиодном 7-сегментном 4-х разрядном индикаторе. На разрядах индикатора отображаются следующая информация:

- на 1-ом разряде число 0, формируемое схемой проекта;
- на 2-ом разряде число, задаваемое переключателями SW[3..0];
- на 3-ом разряде число, задаваемое переключателями SW[7..4];
- на 4-ом разряде число F, формируемое схемой проекта;

#### 1. Создание проекта.

##### 1.1. Запустите пакет проектирования **Quartus Prime Lite**.

**Замечание.** Порядок создания проекта подробно описан в документе «Практическое занятие 1\_v1».

##### 1.2. Для создания проекта выполните следующие действия:

**File -> New Project Wizard.**

В появившемся окне “**Introduction**” выполните «**Next**».

1.3. В окошке “**Directory, Name, Top-Level Entity**” (рис. 1) задайте (или выберите существующую) директорию с проектами и название проекта **lab3\_<fio>** (<fio> – аббревиатура на латинице из первых букв ФИО).

**Важно!**

1. Желательно выделить папку для проектов в корневом каталоге.
2. Не используйте кириллицу и пробелы в определении путей и названий проектов и

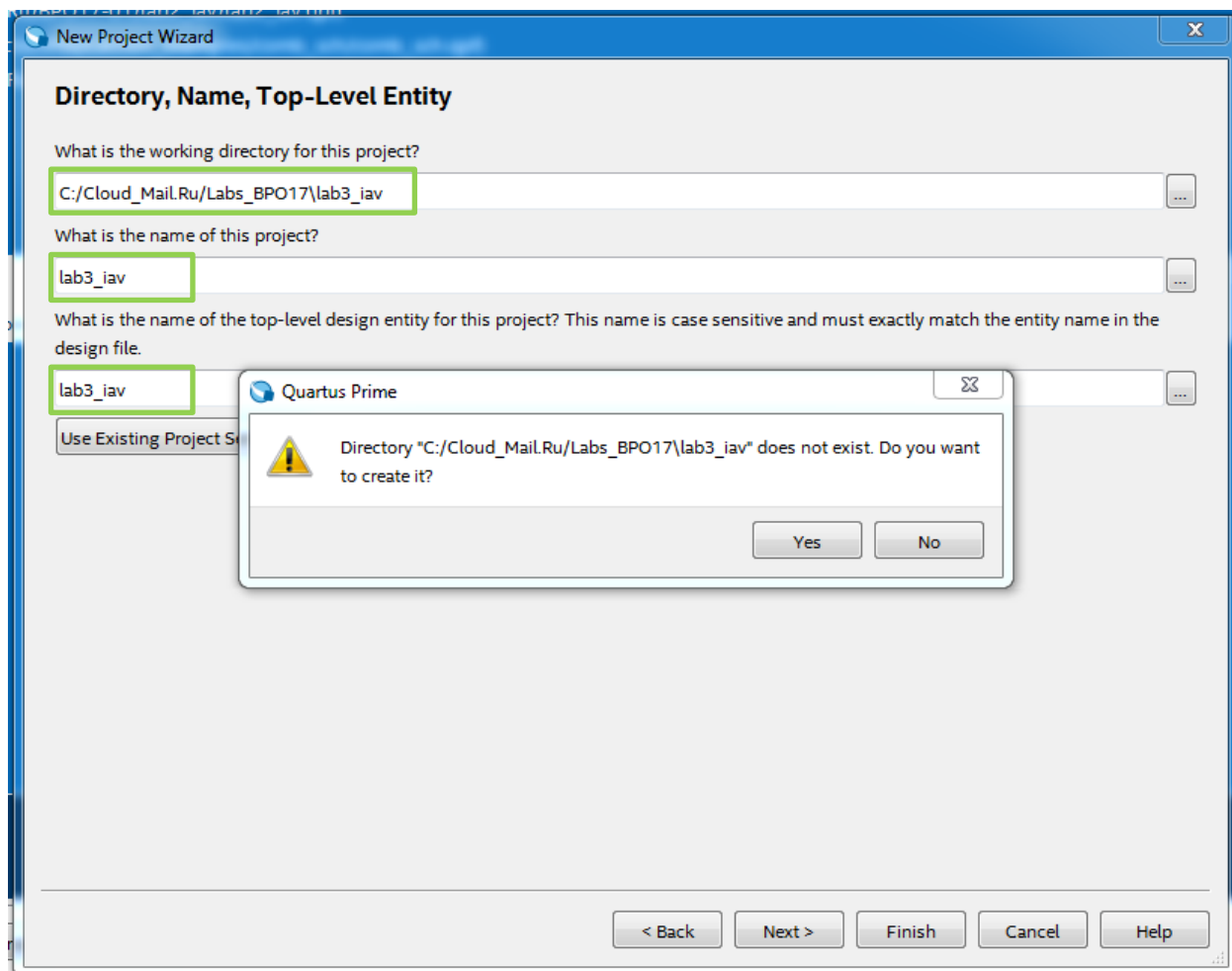


Рис.1 Окно выбора директории для установки проекта

Выполните «**Next**», далее «**Yes**» на предложение создать новую директорию для создаваемого проекта (если директория не создана ранее).

1.4. Появится окно “**Project Type**”, выберите *empty project* (пустой проект) и затем выполните «**Next**».

1.5. Появится окно “**Add Files**”, выполните «**Next**».

1.6. Появится окно “**Family, Device & Board Settings**”. На панели *Device family* выбирается семейство ПЛИС *Cyclone IVE*, а из таблицы *Available devices* нужная микросхема **EP4CE6E22C8**. Для облегчения поиска микросхемы можно конкретизировать ее параметры с помощью выбора параметров «тип корпуса» (*package*), «количество выводов» (*pin count*), «быстродействие ядра» (*core speed grade*) из

соответствующих выпадающих меню панели *Show in 'Available devices' list* (рис. 2). Для проектов практических и лабораторных работ необходимо выбирать микросхему, установленную на отладочную плату, с помощью которой выполняются данные работы. Выполните «**Next**».

1.7. Появится окно “**EDA Tool Settings**”, выполните «**Next**».

1.8. Появится последнее окно создания проекта “**Summary**”, в котором можно проверить все настройки, сделанные на предыдущих шагах. Выполните «**Finish**».

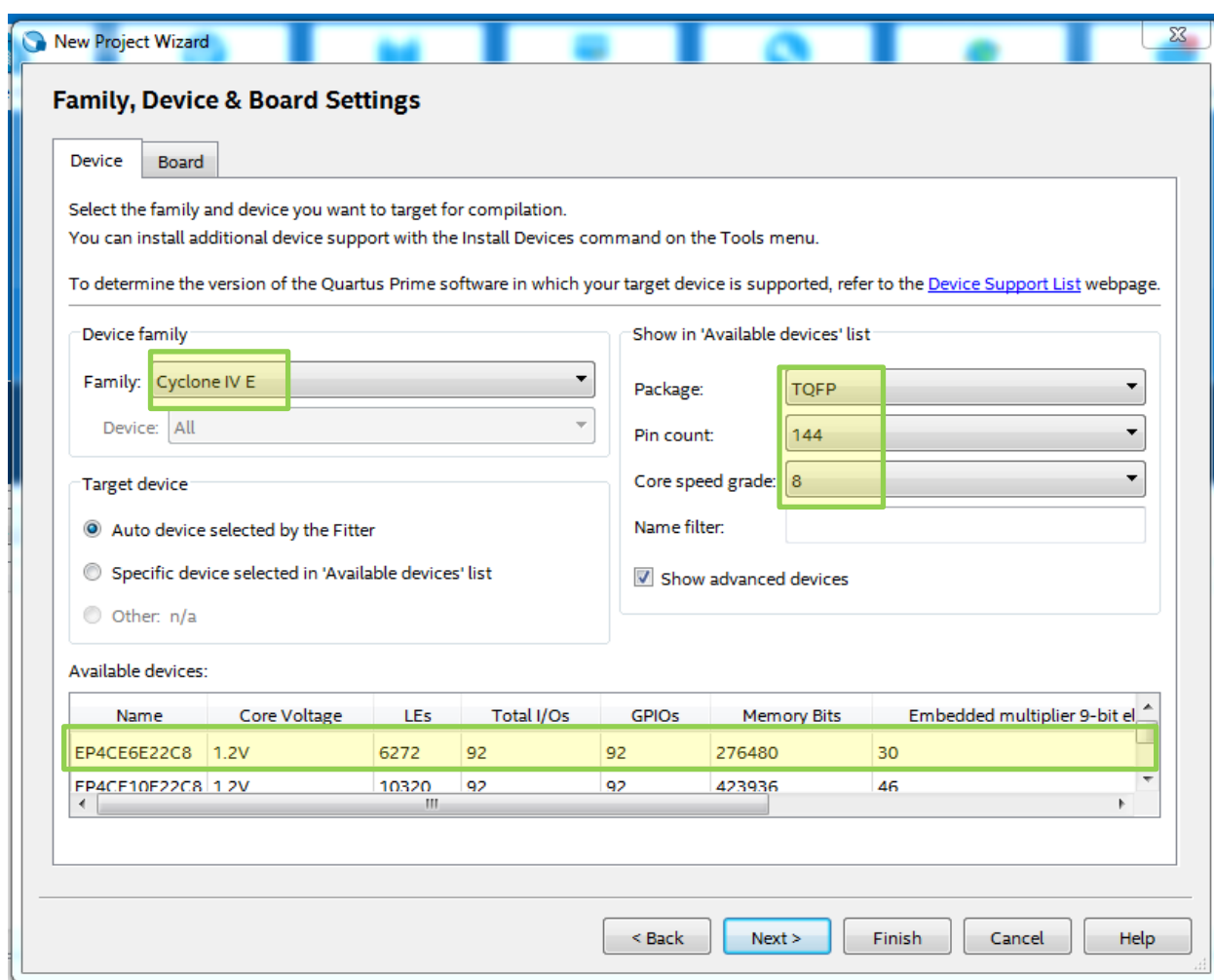
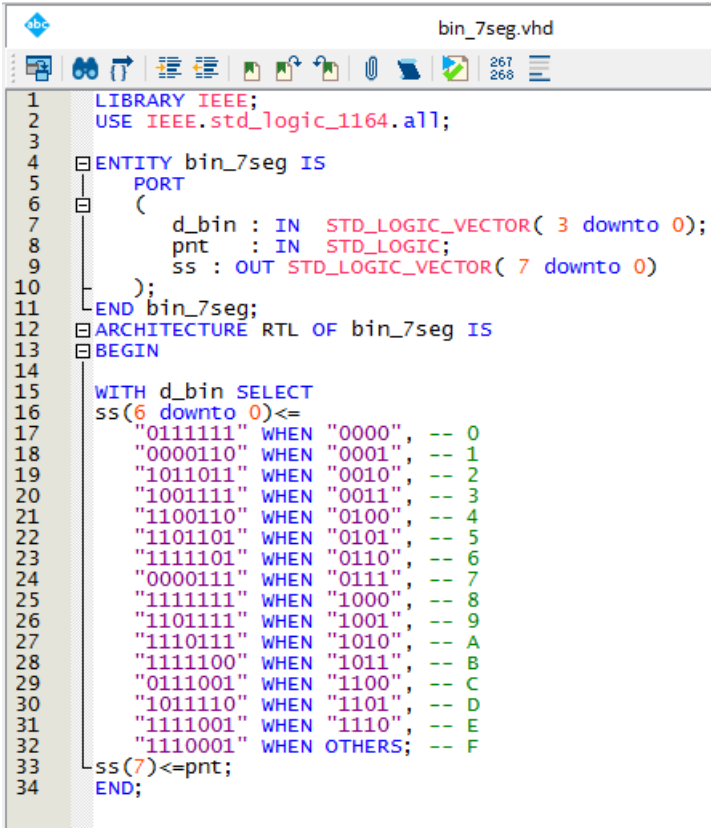


Рис.2 Окно для выбора микросхемы

После этого вновь созданный проект становится текущим в запущенной системе проектирования **Quartus Prime**.

## 2. Создание преобразователя двоичного кода в 7-сегментный код на базе файла с текстовым описанием

2.1. Откройте файл bin\_7seg.vhd из папки проекта (рис. 3). В нем описан компонент, преобразующий входной 4-х разрядный двоичный код в 7-сегментный код на языке описания аппаратуры VHDL.



```
1  LIBRARY IEEE;
2  USE IEEE.std_logic_1164.all;
3
4  ENTITY bin_7seg IS
5  PORT
6  (
7      d_bin : IN  STD_LOGIC_VECTOR( 3 downto 0);
8      pnt   : IN  STD_LOGIC;
9      ss    : OUT STD_LOGIC_VECTOR( 7 downto 0)
10 );
11 END bin_7seg;
12 ARCHITECTURE RTL OF bin_7seg IS
13 BEGIN
14
15     WITH d_bin SELECT
16     ss(6 downto 0) <=
17         "0111111" WHEN "0000", -- 0
18         "0000110" WHEN "0001", -- 1
19         "1011011" WHEN "0010", -- 2
20         "1001111" WHEN "0011", -- 3
21         "1100110" WHEN "0100", -- 4
22         "1101101" WHEN "0101", -- 5
23         "1111101" WHEN "0110", -- 6
24         "0000111" WHEN "0111", -- 7
25         "1111111" WHEN "1000", -- 8
26         "1101111" WHEN "1001", -- 9
27         "1110111" WHEN "1010", -- A
28         "1111100" WHEN "1011", -- B
29         "0111001" WHEN "1100", -- C
30         "1011110" WHEN "1101", -- D
31         "1111001" WHEN "1110", -- E
32         "1110001" WHEN OTHERS; -- F
33
34     ss(7) <= pnt;
35 END;
```

Рис.3 Окно текстового редактора с кодом преобразователя 4-х разрядного двоичного кода в 7-сегментный код на языке описания аппаратуры VHDL

2.2. В окне открытого текстового редактора выполните

**File -> Create/Update -> Create Symbol File for Current File**

Будет создан графический символ bin\_7seg.bsf для текстового описания преобразователя кода, который будет использоваться в качестве компонента схемы проекта (рис. 25).

### 3. Создание экземпляра счетчика на базе библиотеки IP-компонентов

3.1. В окне **IP Catalog** в разделе **Library->Basic Functions->Arithmetic** выберите мегафункцию **LPM\_COUNTER** (рис. 4) и нажмите кнопку «+ADD»,

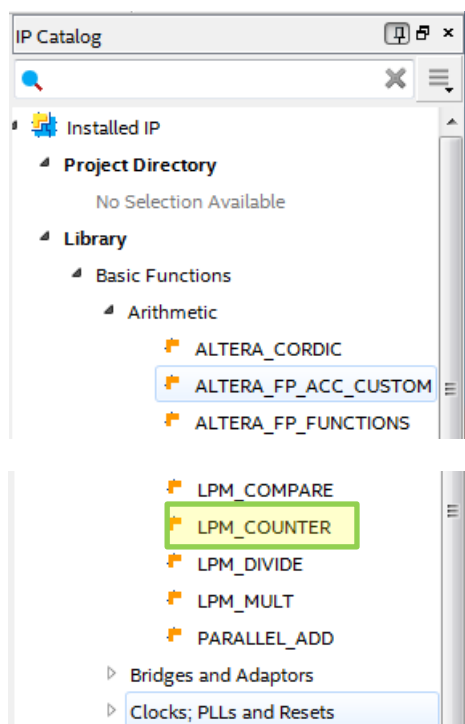


Рис. 4 Окно **IP Catalog**

3.2. В появившемся окне настройки задайте имя экземпляра выбранного компонента (Counter\_12b) и укажите язык описания аппаратуры (VHDL), в коде которого он будет храниться в проекте (рис.5). Выполните «**OK**».

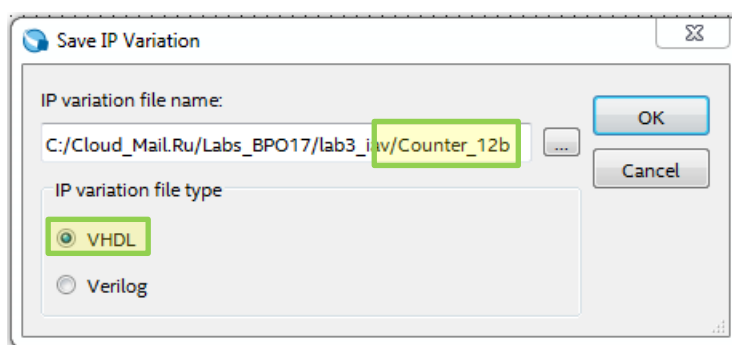


Рис. 5 Окно создания экземпляра выбранного компонента

3.3. В появившемся окне “**MegaWizeded Plug-In Manager [page 1 of 5]**” установите разрядность счетчика 12 бит (рис. 6). Выполните «**Next**».

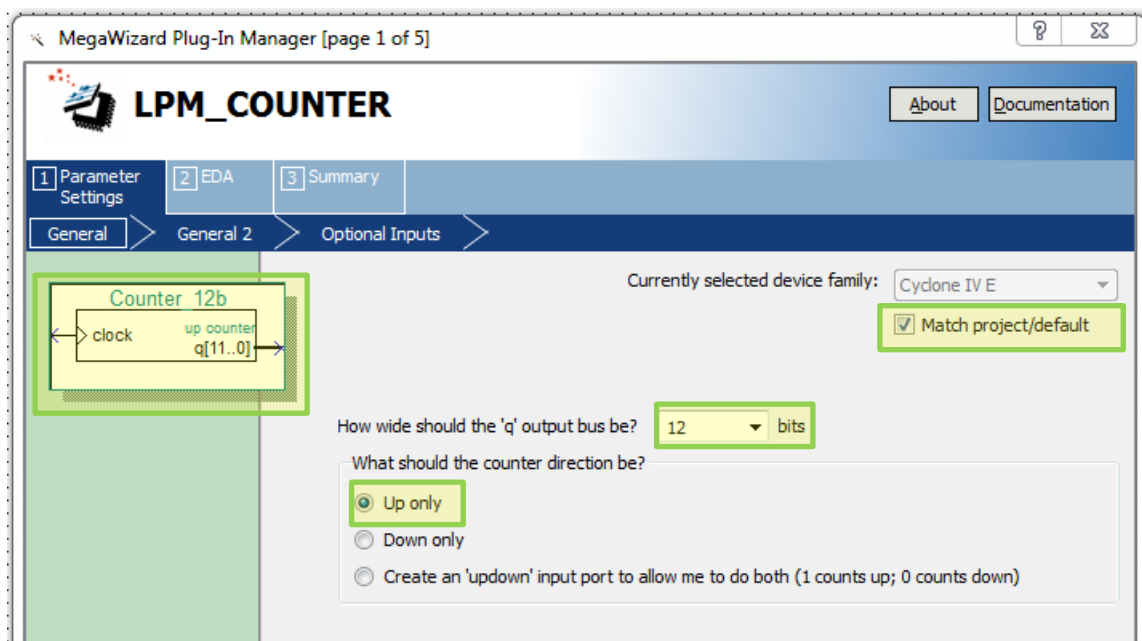


Рис. 6 Окно настройки экземпляра выбранного компонента (page 1)

3.4. В появившемся окне “**MegaWizeded Plug-In Manager [page 2 of 5]**” установите модуль счета **4** и выход переноса *carry-out* (рис. 7), выполните «Next».

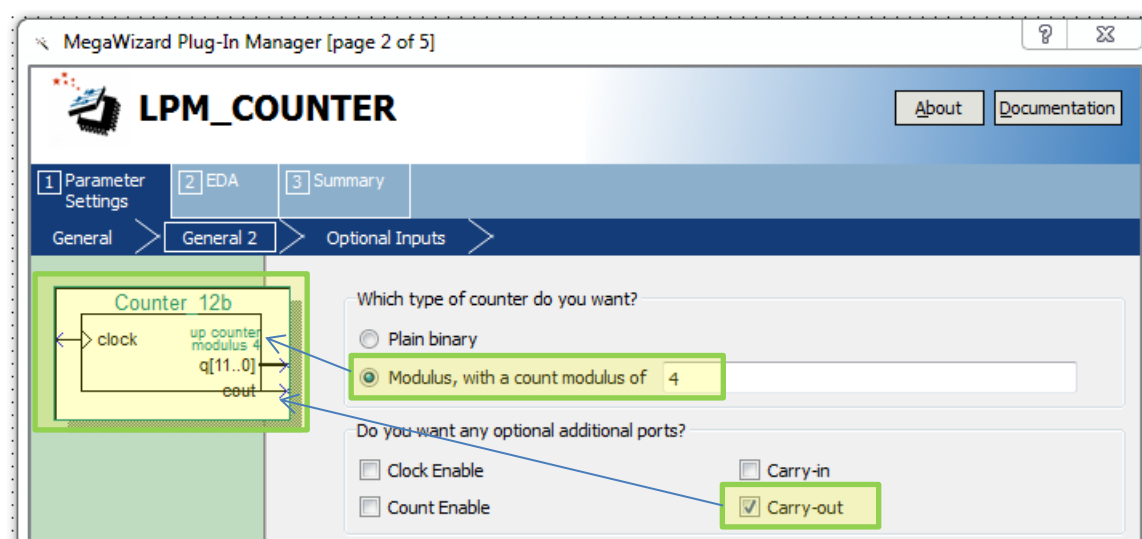


Рис. 7 Окно настройки экземпляра выбранного компонента (page 2)

3.5. В появившемся окне “**MegaWizeded Plug-In Manager [page 3 of 5]**” выполните «Next».

3.6. В появившемся окне “**MegaWizeded Plug-In Manager [page 4 of 5]**” выполните «Next».

3.7. В появившемся окне “**MegaWizeded Plug-In Manager [page 5 of 5]**” выберите создаваемые файлы, как на рис. 8. Далее «Finish». Экземпляр счетчика **Counter\_28b** создан.

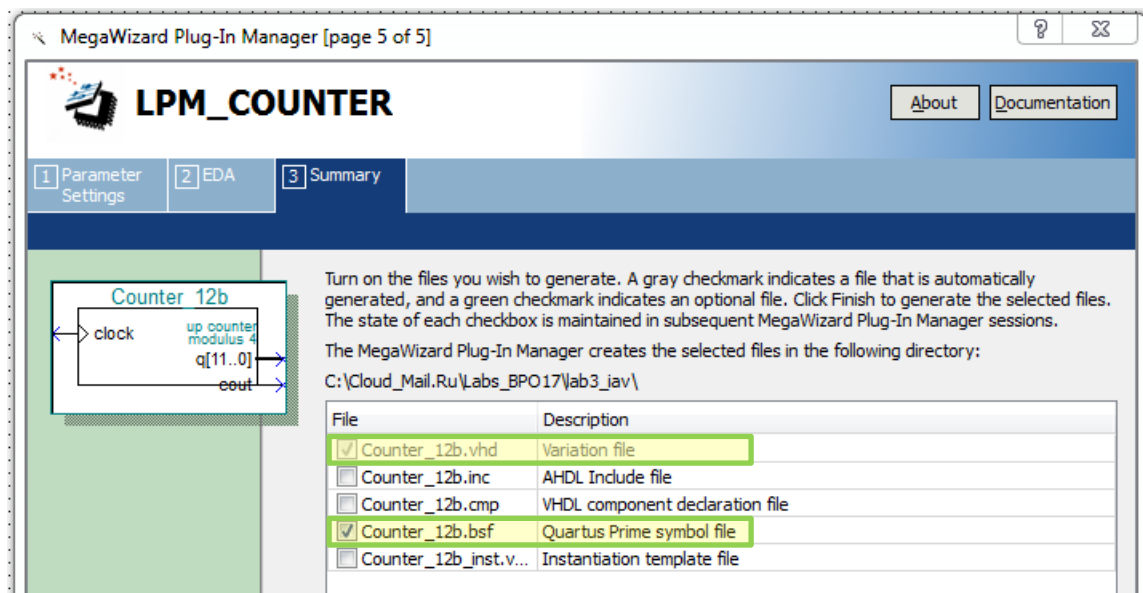


Рис. 8 Окно настройки экземпляра выбранного компонента (page 5)

3.8. В появившемся окне “**QUARTUS Prime IP Files**”, на предложение добавить созданный компонент в проект, выберите «**Yes**».

3.9. В окне **IP Catalog** в разделе **Library->Basic Functions->Miscellaneous** выберите мегафункцию **LPM\_MUX** (рис. 9) и нажмите кнопку «**+ADD**».

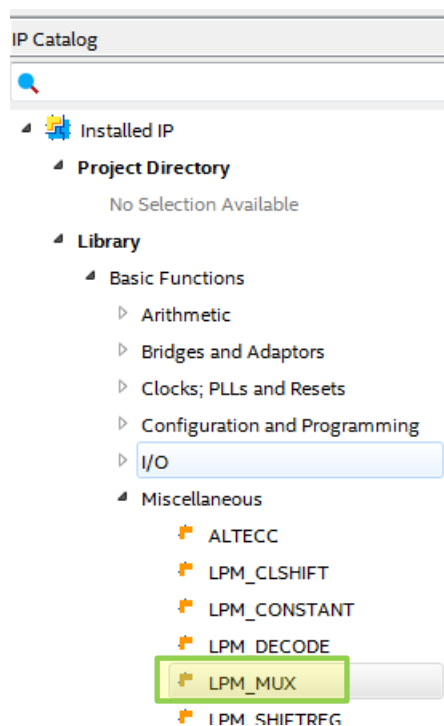


Рис. 9 Окно **IP Catalog**

3.10. В появившемся окне настройки задайте имя экземпляра выбранного компонента (MUX5\_1) и укажите язык описания аппаратуры (VHDL), в коде которого он будет храниться в проекте (рис.10). Выполните «**OK**».

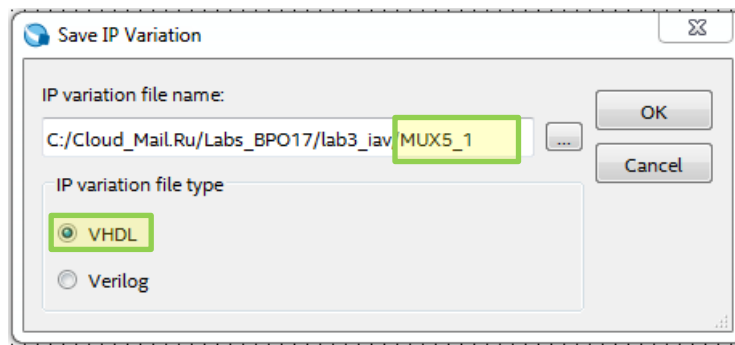


Рис. 10 Окно создания экземпляра выбранного компонента

3.11. В появившемся окне **“MegaWizeded Plug-In Manager [page 1 of 3]”** установите 4 входа разрядностью 5 бит (рис. 11). Выполните **«Next»**.

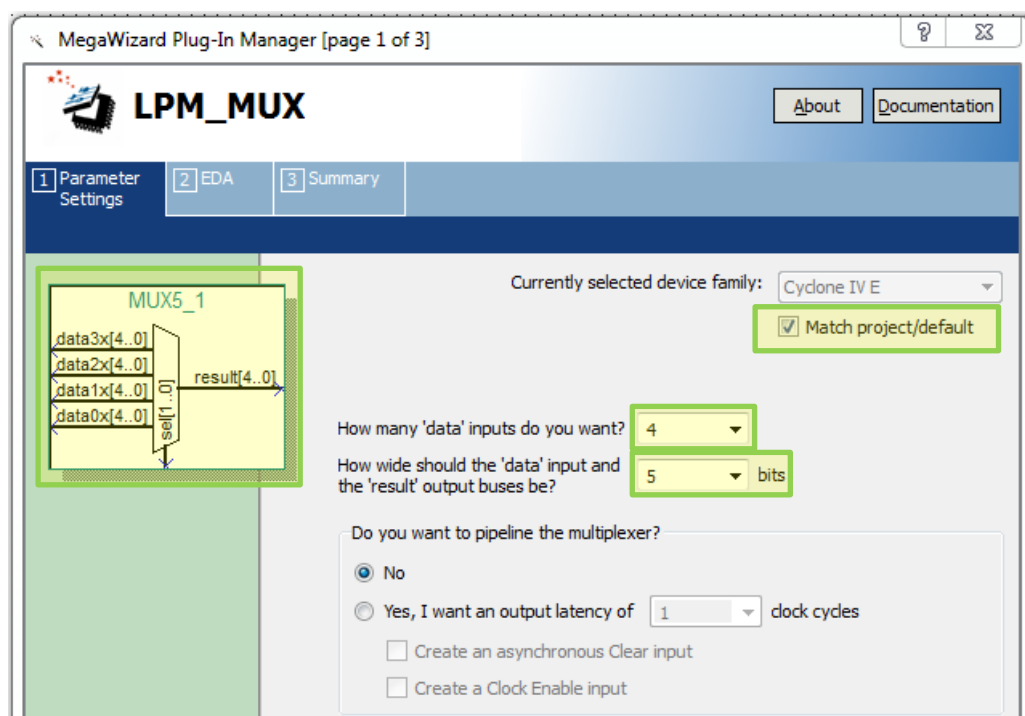


Рис. 11 Окно настройки экземпляра выбранного компонента (page 1)

3.12. В появившемся окне **“MegaWizeded Plug-In Manager [page 2 of 3]”** выполните **«Next»**.

3.13. В появившемся окне **“MegaWizeded Plug-In Manager [page 3 of 3]”** выберите создаваемые файлы как на рис. 12. Далее выполните **«Finish»**. Экземпляр мультиплексора MUX5\_1 создан.



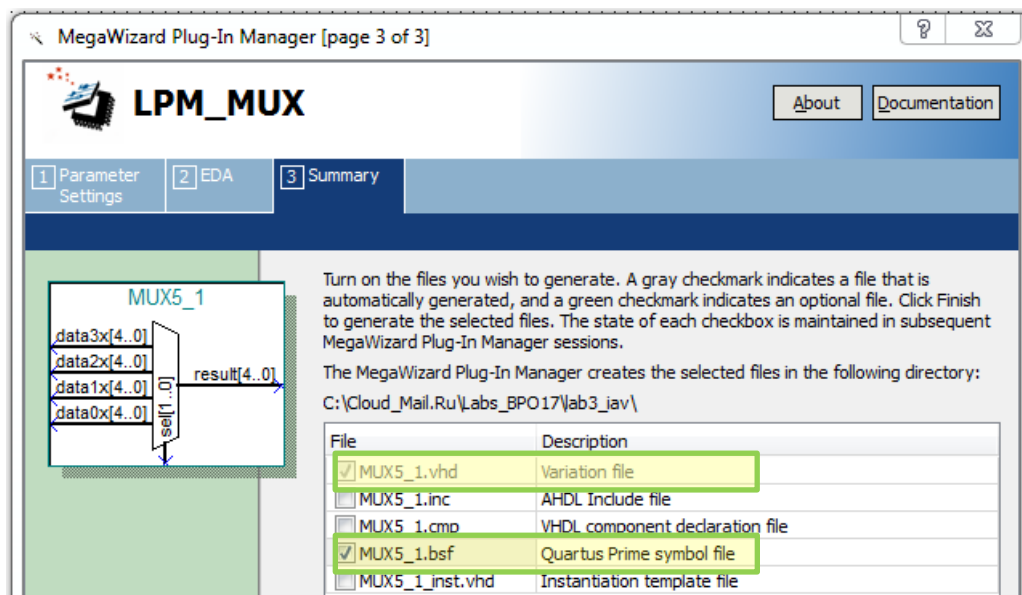


Рис. 12 Окно настройки экземпляра выбранного компонента (page 3)

3.14. В появившемся окне “**QUARTUS Prime IP Files**”, на предложение добавить созданный компонент в проект, выберите «Yes».

#### 4. Создание конечного автомата с помощью редактора конечных автоматов

4.1. Выполните **File -> New** или **Create New Design** в окне **Task** (рис. 13a), откроется окошко (рис.13b) с выбором типов создаваемых файлов.

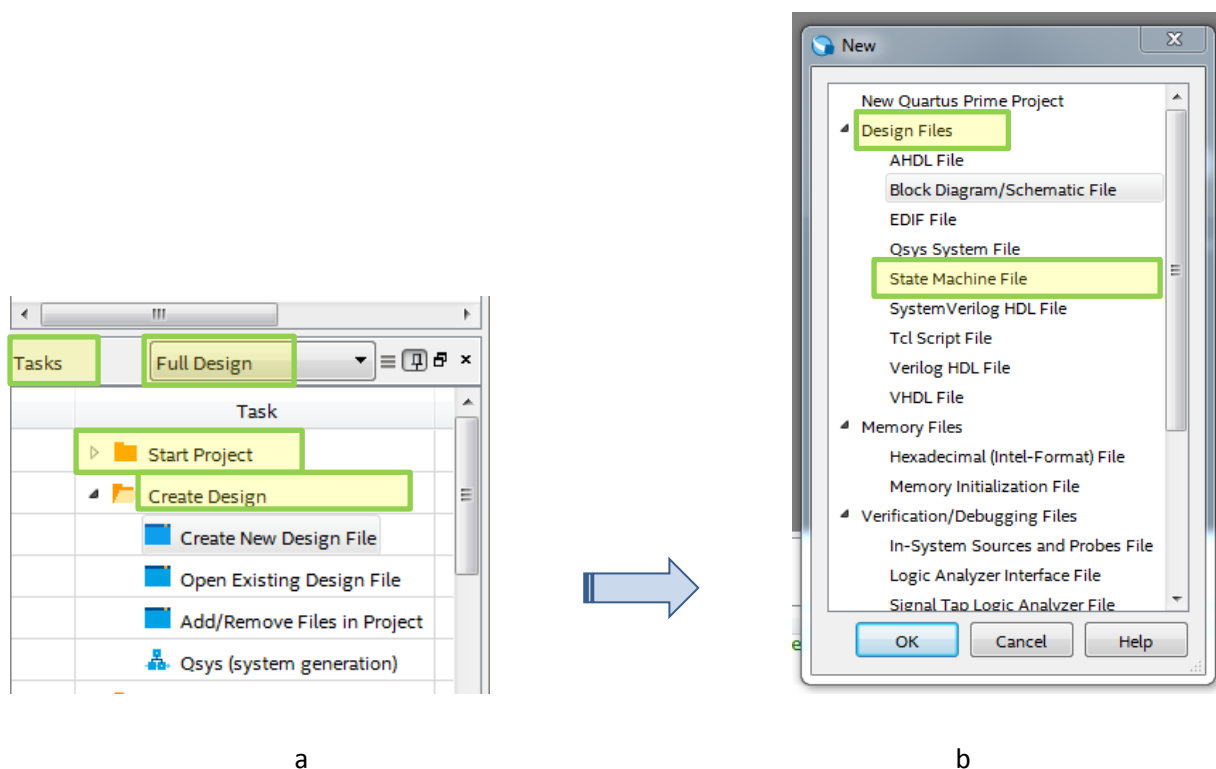


Рис. 13 Окно создания проектного файла (a) и окно выбора типа создаваемого проектного файла(b)

4.2. Выберите **Design Files -> State Machine File**, далее «OK». В результате откроется окошко редактора конечного автомата – **State Machine Editor** (рис. 14).

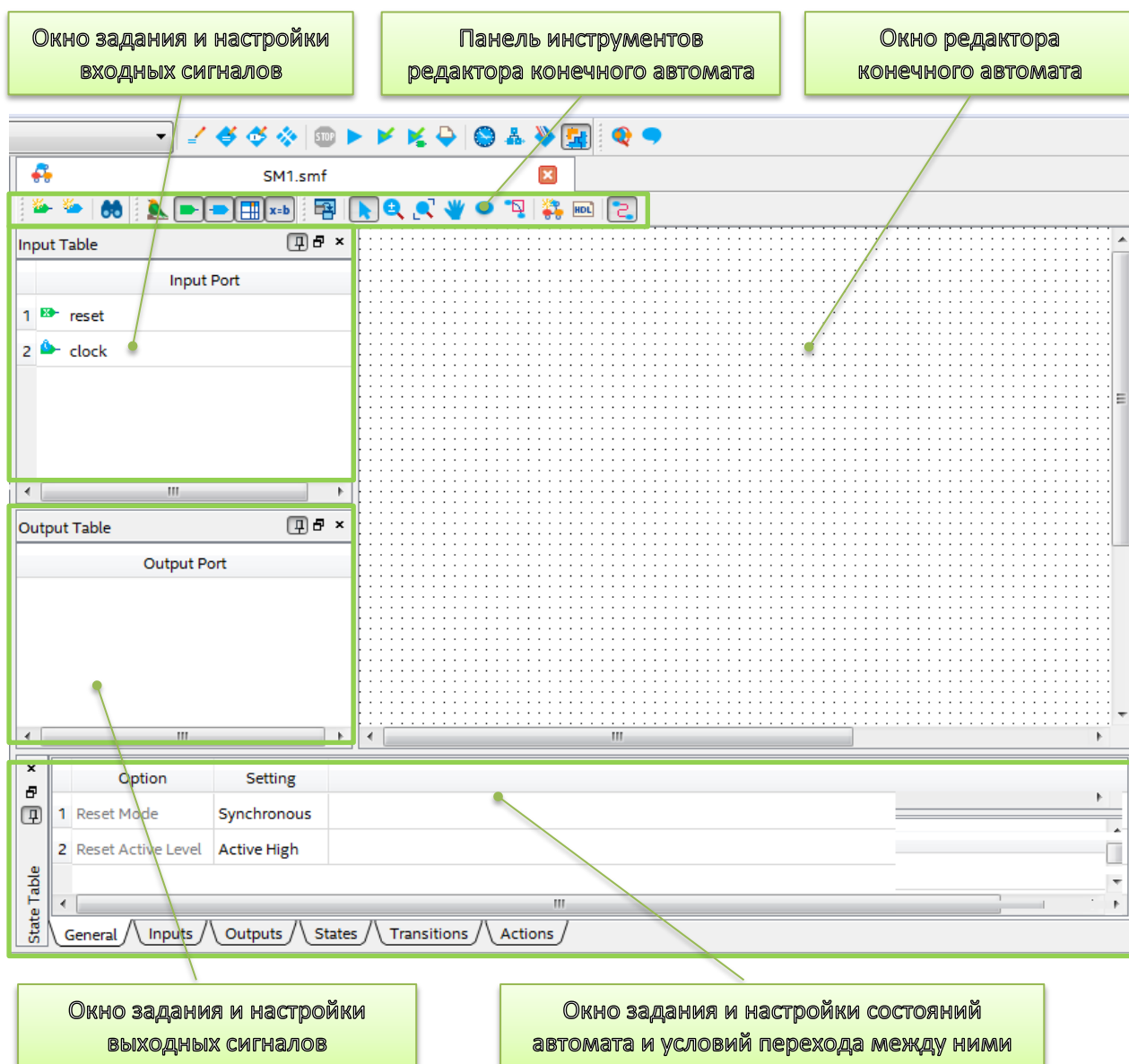





















Рис. 14 Окно редактора конечного автомата

Значения пиктограмм на панели инструментов редактора конечного автомата отображены в таблице 1.

Таблица 1

Пиктограмма	Функция	Комментарий
	Add New Input Port	Добавить новый вход автомата
	Add New Output Port	Добавить новый выход автомата
	Find /Ctrl+F	Поиск объектов
	Bird's Eye View	Просмотр всей схемы (взгляд с высоты птичьего полета)
	Input Port List	Включить/ выключить окно с входными сигналами (Input Table)
	Output Port List	Включить/ выключить окно с выходными сигналами (Output Table)
	State Table	Включить/ выключить окно задания и настройки состояний автомата и условий перехода между ними (State Table)
	Show Transition Equations	Включить/ выключить показ условий перехода между состояниями на схеме
	Detach/Attach Window	Отсоединить/Присоединить State Machine Editor
	Selection Tool	Выбор режима редактирования схемы
	Zoom Tool	Масштабирование схемы
	Magnifying Glass Tool	Увеличение фрагмента схемы
	Hand Tool	Перетаскивание схемы по полю редактора
	State Tool	Установка объекта “состояние автомата” на поле редактора
	Transition Tool	Инструмент для рисования переходов между состояниями
	State Machine Wizard	Запуск создания и редактирования автомата
	Generate HDL File	Создание описания созданного автомата на языке описания аппаратуры
	Use Rubberbanding	Создание соединений между состояниями автомата в виде произвольных кривых

4.3. В данной работе создадим конечный автомат с помощью инструмента **State Machine Wizard**. Нажмите кнопку  на панели инструментов редактора конечных автоматов. Появится окно **State Machine Wizard** (рис. 15), на вкладке **General** которого настройте сигнал *reset* со следующими значениями: *Asynchronous*, *Reset is Active High*.

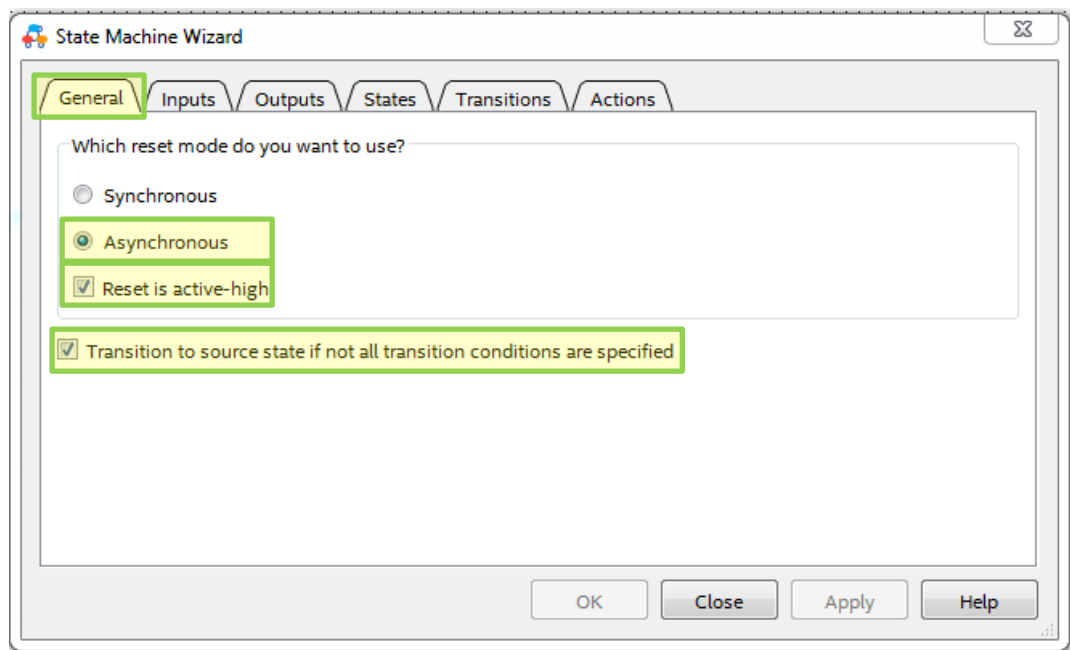


Рис. 15 Задание параметров сигнала **reset** конечного автомата

4.4. На вкладке **Inputs** кликните левой кнопкой мыши по надписи «**New**» в колонке **Input Port** и введите имя **ENA** (рис. 16).

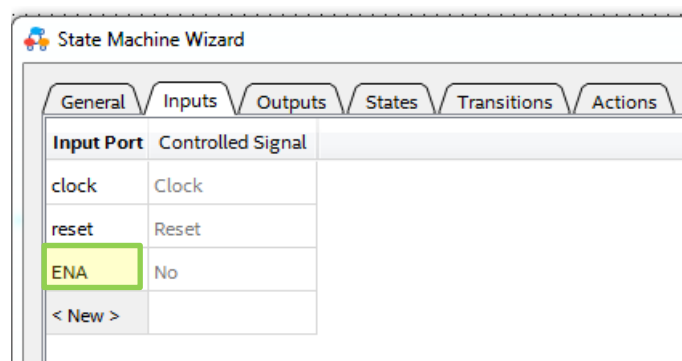


Рис. 16 Задание входных сигналов конечного автомата

4.5. На вкладке **Outputs** кликните левой кнопкой мыши по надписи «**New**» в колонке **Output Port** и введите имена сигналов **DIG[3:0]** и **SEL[1:0]** (рис. 17).

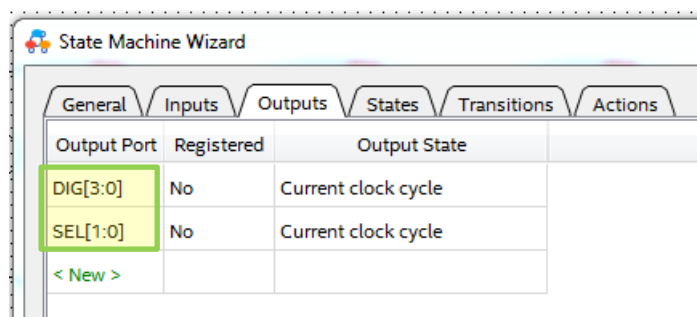


Рис. 17 Задание выходных сигналов конечного автомата

4.6. На вкладке **States** кликните левой кнопкой мыши по надписи «**New**» в колонке **State** и введите имена состояний автомата **state1**, **state2**, **state3**, **state4** (рис. 18). В колонке **Reset** определено, что по активному сигналу **reset** автомат перейдет в состояние **state1**.

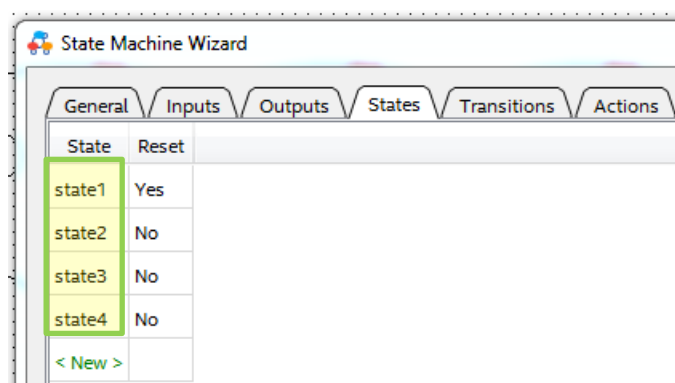


Рис. 18 Задание состояний конечного автомата

4.7. На вкладке **Transitions** кликните левой кнопкой мыши по надписи «**New**» в колонке **Source State** (исходное состояние) и введите имя состояния автомата **state1**. Затем в колонке **Destination State** (состояние назначения) введите состояния автомата **state2**. В колонке **Transition** определите условие перехода автомата из состояния **state1** в состояние **state2**. Аналогично заполните остальные строки таблицы (рис. 19).

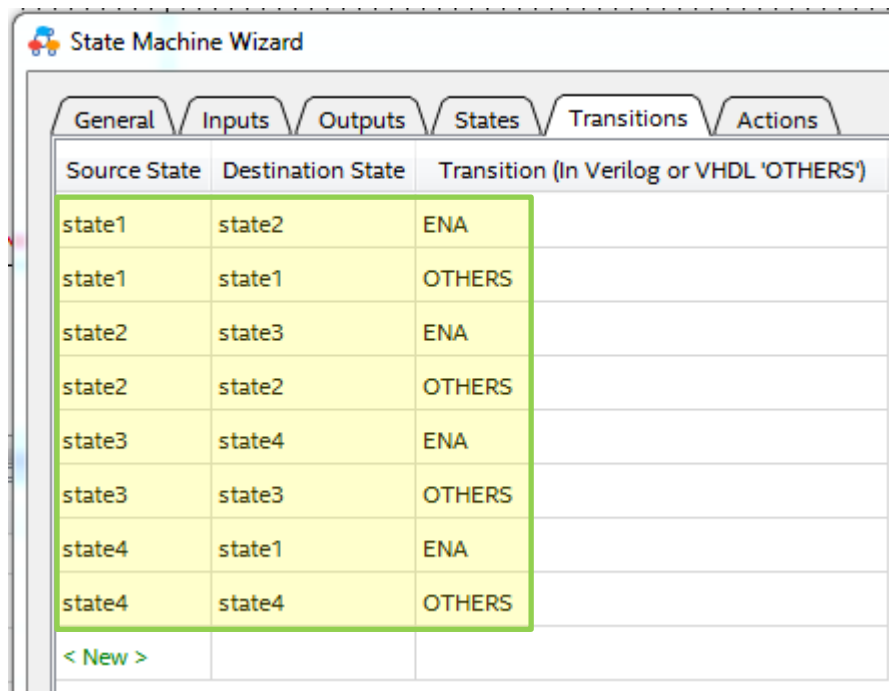


Рис. 19 Задание условий перехода конечного автомата

4.8. На вкладке **Actions** кликните левой кнопкой мыши по надписи «**New**» в колонке **Output Port** и выберите выходной сигнал **DIG[3:0]** из выпадающего списка. Затем, в колонке **Output Value**, задайте его значение для состояния, которое выбирается в колонке **In State**. Аналогично заполните остальные строки таблицы (рис. 20).

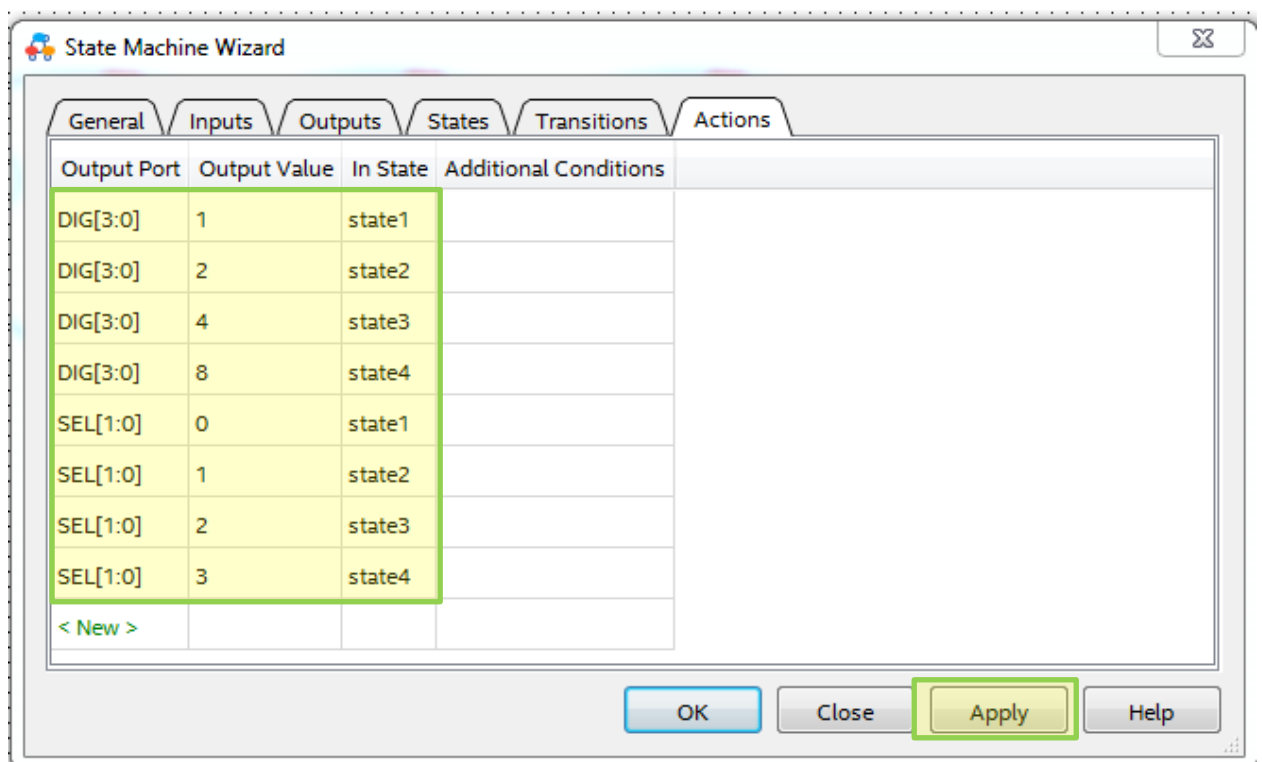


Рис. 20 Задание значений выходных сигналов для состояний конечного автомата

4.9. Выполните «**Apply**», затем «**Close**». На экране редактора конечных автоматов появится схема созданного автомата (рис. 21).

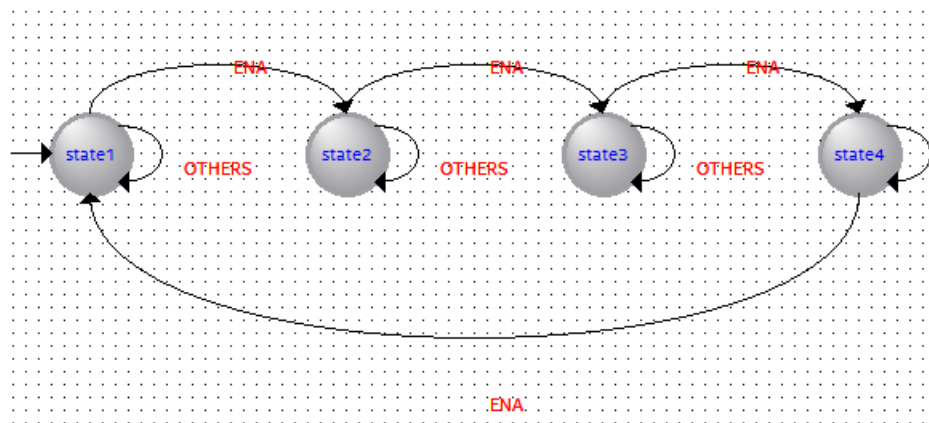



Рис. 21 Схема созданного конечного автомата

4.10. При желании схему можно отредактировать. Для этого надо выбрать инструмент  и выполнить, путем выделения элементов схемы, перестановку надписей, символов состояний и линий связи в желаемый вид (рис. 22).

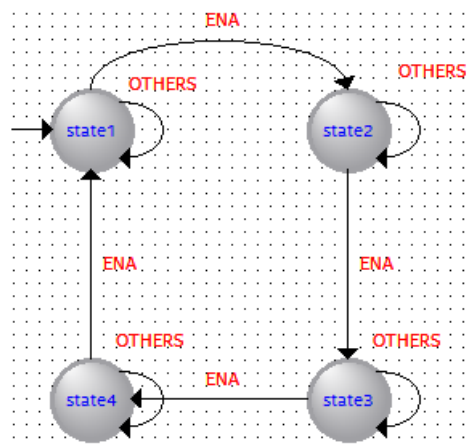



Рис. 22 Отредактированная схема созданного конечного автомата

4.11. Сохраните файл, выполнив **File -> Save As**, под именем *fsm\_gen.smf*.

4.12. Выполните генерацию файла описания созданного автомата на языке HDL с помощью инструмента . Для этого нажмите на нее ЛКМ на панели инструментов и в появившемся окне **Generate HDL File** выберите нужный язык описания (в нашем случае VHDL) (рис. 23). Далее выполните «**OK**». Появится окно текстового редактора **Text Editor** с созданным текстом описания автомата на языке VHDL (рис. 24).

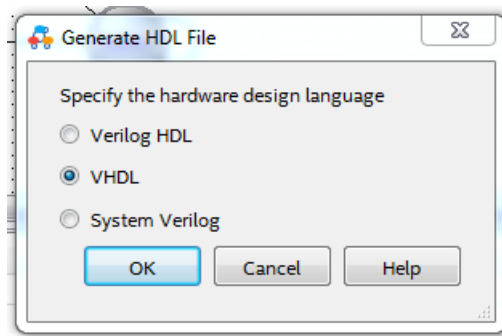


Рис. 23 Генерация файла описания созданного автомата на языке HDL

```

1  -- Copyright (C) 2018 Intel Corporation. All rights reserved.
2  -- Your use of Intel Corporation's design tools, logic functions
3  -- and other software and tools, and its AMPP partner logic
4  -- functions, and any output files from any of the foregoing
5  -- (including device programming or simulation files), and any
6  -- associated documentation or information are expressly subject
7  -- to the terms and conditions of the Intel Program License
8  -- Subscription Agreement, the Intel Quartus Prime License Agreement,
9  -- the Intel FPGA IP License Agreement, or other applicable license
10 -- agreement, including, without limitation, that your use is for
11 -- the sole purpose of programming logic devices manufactured by
12 -- Intel and sold by Intel or its authorized distributors. Please
13 -- refer to the applicable agreement for further details.
14
15 -- Generated by Quartus Prime Version 18.1.0 Build 625 09/12/2018 SJ Lite Edition
16 -- Created on Tue Mar 02 21:37:32 2021
17
18 LIBRARY ieee;
19 USE ieee.std_logic_1164.all;
20
21 ENTITY fsm_gen IS
22 PORT (
23     clock : IN STD_LOGIC;
24     reset : IN STD_LOGIC := '0';
25     ENA : IN STD_LOGIC := '0';
26     DIG : OUT STD_LOGIC_VECTOR(3 DOWNTO 0);
27     SEL : OUT STD_LOGIC_VECTOR(1 DOWNTO 0);
28 );
29 END fsm_gen;
30
31 ARCHITECTURE BEHAVIOR OF fsm_gen IS
32     TYPE type_fstate IS (state1,state2,state3,state4);
33     SIGNAL fstate : type_fstate;
34     SIGNAL reg_fstate : type_fstate;
35 BEGIN
36     PROCESS (clock,reset,reg_fstate)
37     BEGIN
38         IF (reset='1') THEN
39             fstate <= state1;
40         ELSIF (clock='1' AND clock'event) THEN
41             fstate <= reg_fstate;
42         END IF;
43     END PROCESS;
44
45     PROCESS (fstate,ENA)
46     BEGIN
47         DIG <= "0000";

```

Рис. 24 Файл описания созданного автомата на языке HDL

4.13. В окне открытого текстового редактора выполните

**File -> Create/Update -> Create Symbol File for Current File**

Будет создан графический символ *fsm\_gen.bsf* для созданного автомата, который будет использоваться в качестве компонента для схемы проекта (рис. 25).



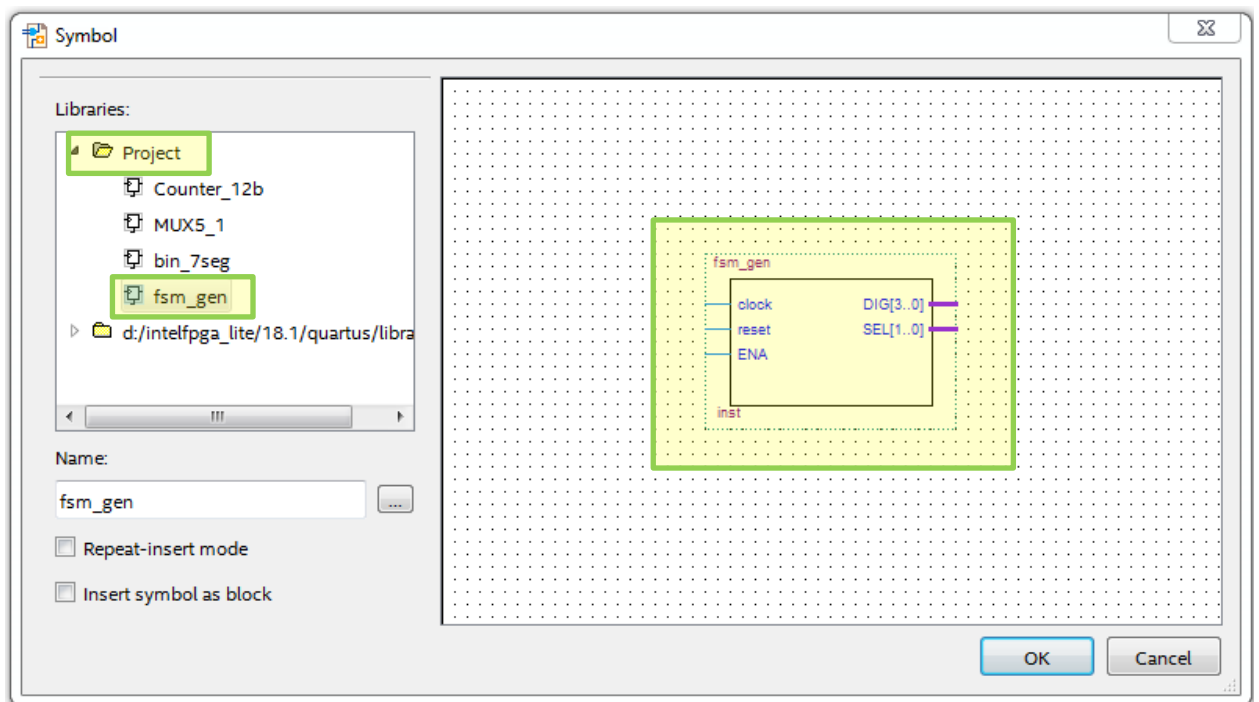


Рис. 25 Символ созданного автомата в библиотеке проекта

## 5. Создание дизайн - файла компонента с помощью схематического редактора

5.1. Выполните **File -> New** или **Create New Design** в окне **Task** (рис. 26a), откроется окошко (рис.26b) с выбором типов создаваемых файлов.

Выберите **Block\_Diagram/Schematic File**, далее «OK». В результате откроется окошко схематического редактора (**Schematic editor**).

5.2. Сохраните файл со схемой **File -> Save As** с именем *ind\_drv.bdf*. Пакет по умолчанию предлагает имя файла с именем проекта и расширением, которое определяется выбранным типом проектного файла. По умолчанию опция '**Add file to current project**' включена и при сохранении созданный файл автоматически подключится к текущему проекту.

5.3. Нарисуйте схему компонента (рис. 27). Создаваемый компонент выполняет функцию драйвера индикатора, который формирует информационные сигналы для индикатора из 16-битного двоичного числа, поступающего на его вход.

Инструменты для создания схемы и работа с ними описаны в документе «Практическое занятие 1\_v1». Доступ к созданным экземплярам компонентов из **IP\_catalog** осуществляется через автоматически создаваемую папку **Project** в библиотеке элементов (рис. 25). Выбирайте верхний регистр при именовании сигнала для лучшей читаемости схемы.

**Пояснения к схеме.** Сигналы **D3\_0[3..0]**, **D4\_7[3..0]**, **D8\_11[3..0]**, **D12\_15[3..0]**, **PNT[3..0]** поступают на цепочки из двух последовательно соединенных триггеров для исключения явлений метастабильности триггеров. Таким образом, асинхронные входные сигналы привязываются к тактовой частоте **CLK**, на которой работает схема и удовлетворяют требованиям правил построения синхронных схем.

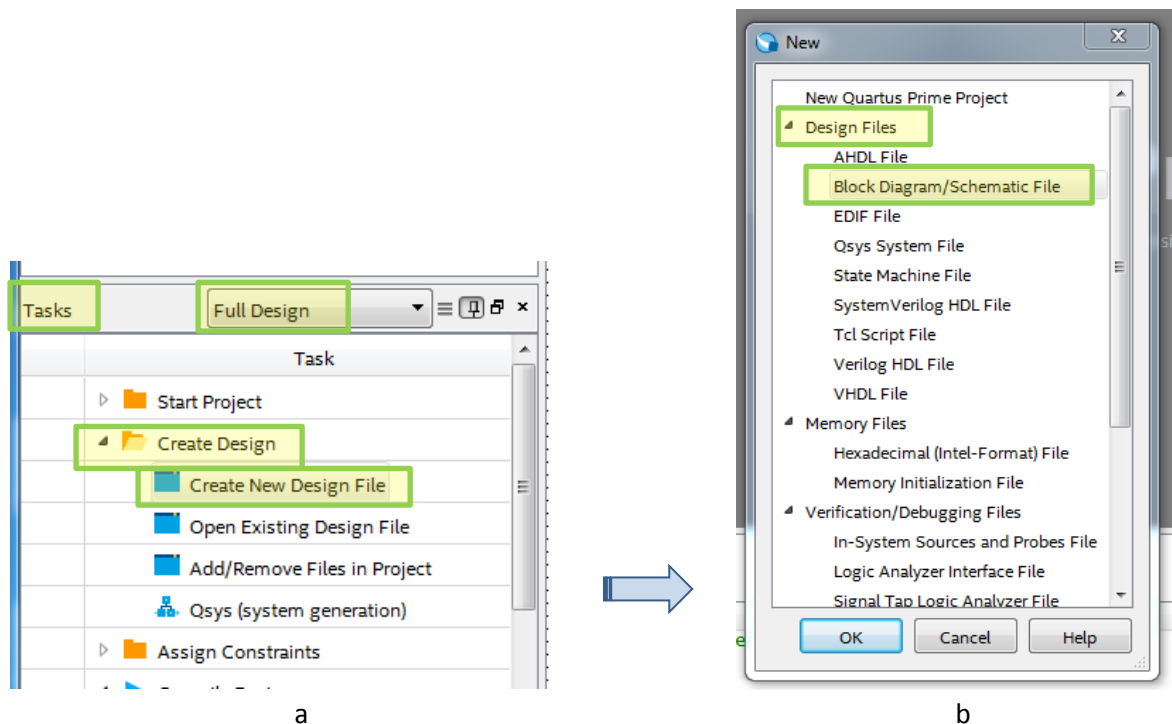



Рис. 26 Окно создания проектного файла (a) и окно выбора типа создаваемого проектного файла(b)

Схему удобней рисовать с включенной сеткой, управление ее видимостью можно выполнить следующим образом: **View-> Show Guidelines** или кликнуть ПКМ на экране редактора, затем выбрать **Show->Show Guidelines**. Если на схеме видны только части имен, настройте параметры монитора на вашем компьютере (может быть установлен не тот масштаб изображения).

При создании схемы обратите особое внимание на индексацию сигналов. Компоненты **WIRE** и **GND** проще искать в библиотеке по именам в окне ввода имени **Name**. С помощью компонента **WIRE** выполнена конкатенация (слияние) одноразрядных сигналов **PNT\_INT[x]** и 4-х разрядных сигналов **Dx\_xINT[3..0]** в 5-ти разрядный сигнал **DATAx[4..0]**. Таким образом к двоичному сигналу, который будет отображаться после преобразования в 7-сегментный код на соответствующем разряде индикатора, добавлен сигнал управления свечением точки на этом разряде индикатора. Входные сигналы перечисляются через запятую, при этом порядок их следования важен и соответствует порядку разрядов на выходе элемента **WIRE** (рис. 28).

Рекомендуется сначала поименовать компоненты **INPUT** и **OUTPUT**. В данной схеме это даст компилятору возможность распознать шинные источники сигналов и при создании схемы рисовать шинные линии. В противном случае это необходимо делать вручную следующим образом: выбрать инструмент  на панели инструментов схемного редактора и нарисовать курсором мыши нужную линию. Либо нарисовать тонкую линию, затем выделить ее, нажать правую кнопку мыши и из выпадающего меню выбрать опцию **Bus Line**. Если поименовать простую (тонкую) линию связи на схеме многобитным именем, компилятор выдаст ошибку несоответствия типа линии и имени сигнала.



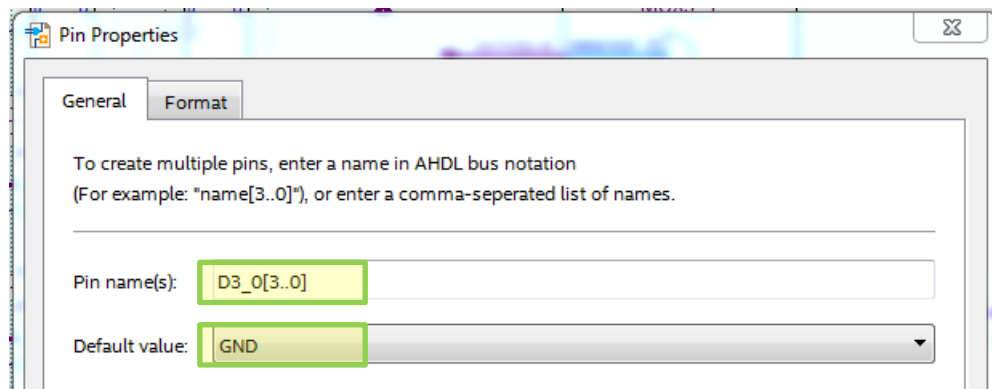



Рис. 29 Установка значений сигналов на входных выводах по умолчанию, когда к ним не подключены внешние сигналы

## 6. Функциональное моделирование компонента

6.1. При открытом окне схемного редактора со схемой компонента **ind\_drv** выполните **Project -> Set as Top-Level Entity**. (Либо выделите имя компонента в папке **Files** окна **Project Navigator**, нажмите правую кнопку мыши и выберите опцию **Set as Top-Level Entity**, либо выполните **Ctrl+Shift+J**). Схема компонента станет верхним уровнем проекта.

6.2. Выберите задачу **Analysis@Synthesis** из окна задач **Tasks** или нажмите кнопку  на панели задач, или выполните **Ctrl+K**, или **Processing->Start->StartAnalysis@Synthesis**. Запустится компиляция, результаты которой можно наблюдать в окне **Compilation Report** (рис. 30). Видно что, проектом верхнего уровня установлен файл **ind\_drv**.

Flow Summary	
<<Filter>>	
Flow Status	Successful - Fri Mar 05 15:52:36 2021
Quartus Prime Version	18.1.0 Build 625 09/12/2018 SJ Lite Edition
Revision Name	lab3_iav
Top-level Entity Name	ind_drv
Family	Cyclone IV E
Device	EP4CE6E22C8
Timing Models	Final
Total logic elements	85
Total registers	68
Total pins	33
Total virtual pins	0
Total memory bits	0
Embedded Multiplier 9-bit elements	0
Total PLLs	0

Рис. 30 Результаты компиляции **Analysis@Synthesis**

Обратите внимание на имя проекта верхнего уровня. Устраните ошибки при необходимости.

6.3. Выполните команду

**File -> Create/Update -> Create Symbol File for Current File**

Будет создан графический символ для созданного компонента **ind\_drv.bsf**, который будет использоваться в качестве элемента в схеме проекта.

6.4. Выполните **File -> New**, откроется окошко (рис.31) с выбором типов создаваемых файлов. Выберите **University Program VWF**, далее «**OK**». В результате откроется окошко редактора эпюр симуляции (**Simulation Waveform Editor**) (рис. 32).

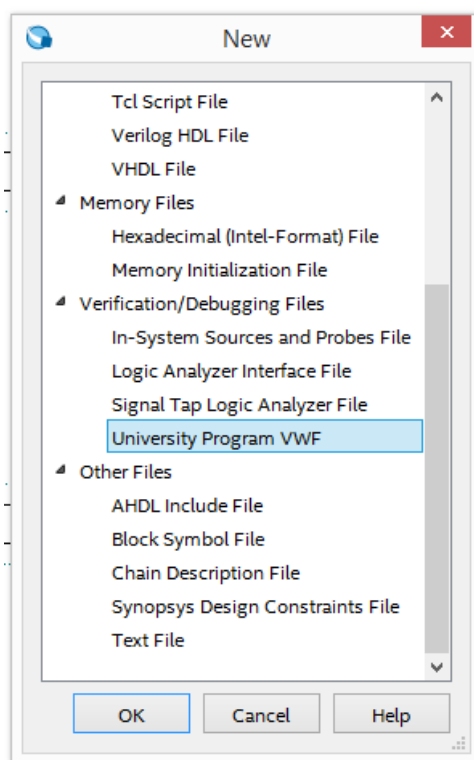


Рис. 31 Окошко выбора типа создаваемого файла симуляции (Verification/Debugging File)

6.5. Сохраните созданный файл под именем **ind\_drv.vwf** (**File->Save\_As**). В окошке редактора сделайте следующие настройки (рис. 33):

- шаг сетки на экране      **10 ns:**      **Edit -> Grid Size;**
- время симуляции      **2000 ns:**      **Edit -> Set End Time;**

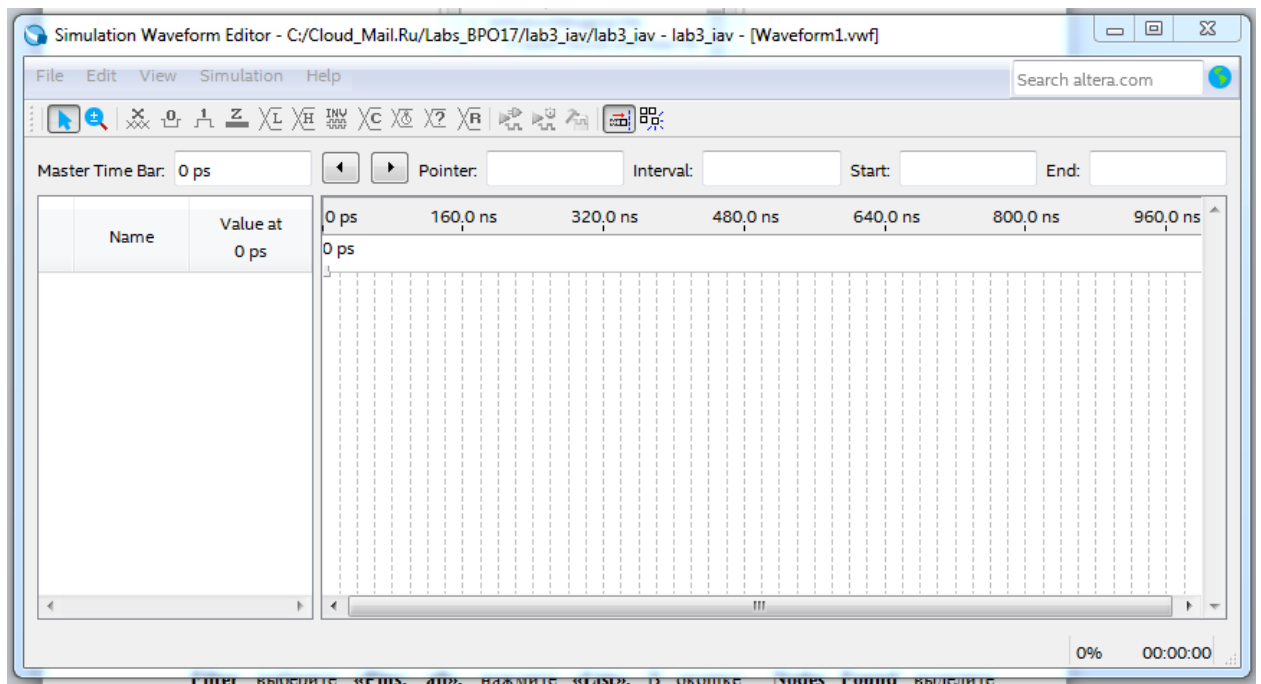


Рис. 32 Окно редактора Simulation Waveform

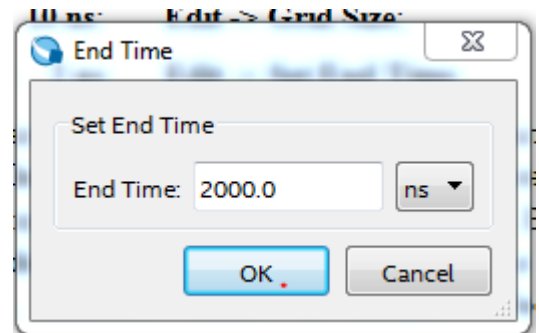
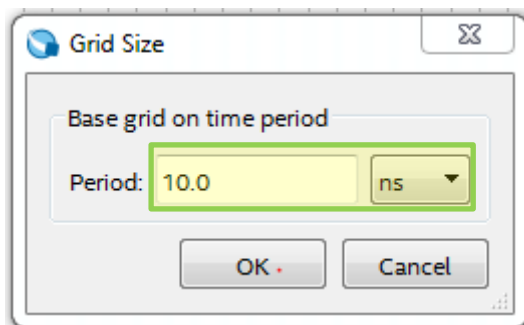




Рис. 33 Задание параметров окна симуляции (шага сетки и времени симуляции)

6.6. Задайте в колонке редактора **Name** имена выводов и сигналов проекта. Для этого выполните: **Edit-> Insert-> Insert Node or Bus** (или в поле колонки «Name», кликните ПКМ и выберите из выпадающего списка «**Insert Node or Bus**»). В появившемся окошке (рис. 34) выберите «**Node Finder...**». Далее в появившемся окошке **Node Finder** в окошке **Filter** выберите «**Pins: all**», нажмите «**List**». В окошке **Nodes Found** выделите необходимые сигналы и перенесите в окошко **Selected Nodes** с помощью кнопки  (по одному сигналу или группу выделенных сигналов, при этом можно сразу упорядочить сигналы, чтобы было удобней наблюдать их при анализе результатов симуляции) или кнопку  (все сигналы) (рис. 35). Попробуйте сразу упорядочить сигналы, это позволит при анализе эюр сигналов лучше понять работу схемы. Многозарядные сигналы целесообразней отображать как группу. Далее дважды выполните «**Ok**».

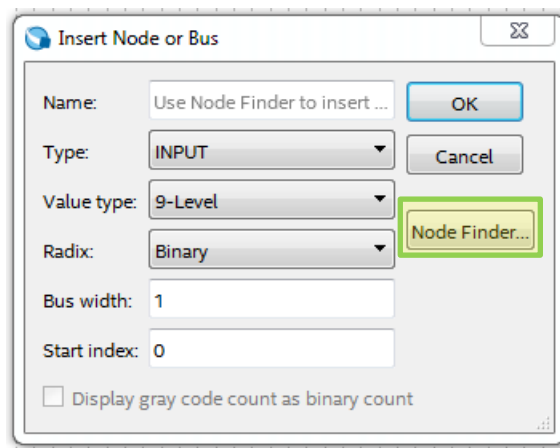




Рис.34 Окно выбора сигналов **Node Finder**

Экран редактора симуляции эюр приобретет вид как на рисунке 36. При этом входные сигналы имеют по умолчанию значение , а выходные сигналы не определены .

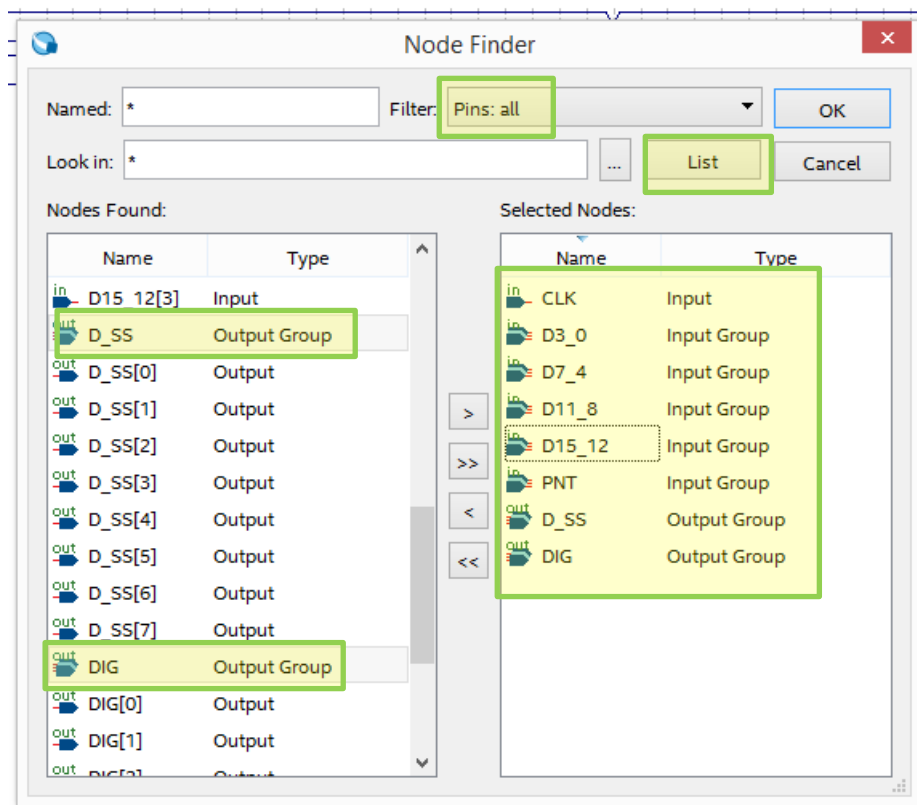


Рис.35 Выбор сигналов для симуляции

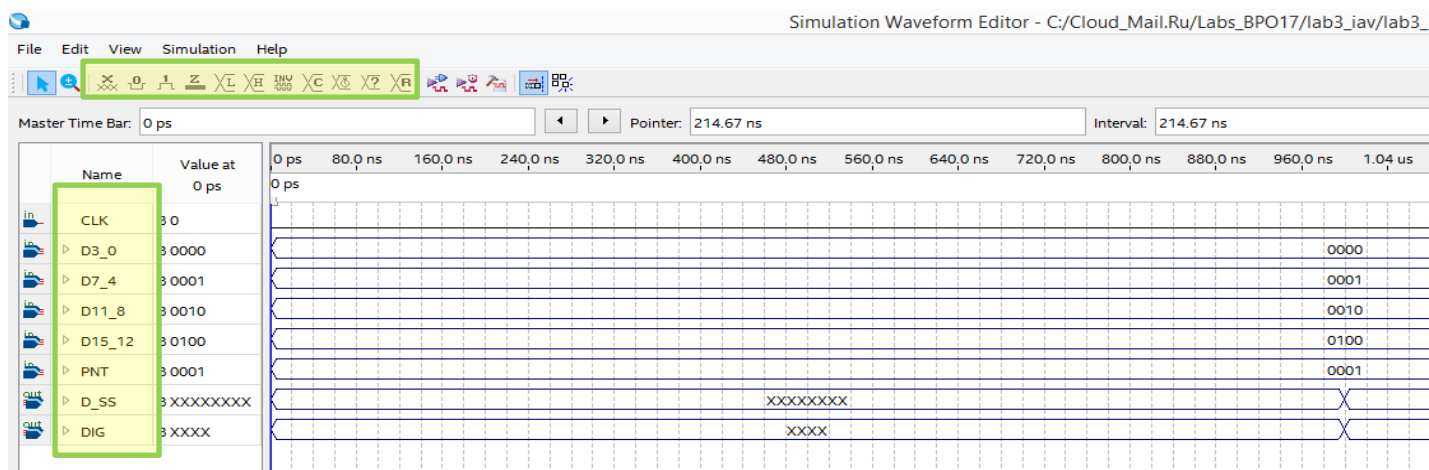
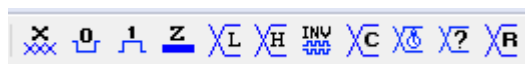


Рис.36 Экран редактора симуляции с именами выбранных для симуляции сигналов

6.7. Цель симуляции состоит в проверке схемы на всем множестве значений входных сигналов. Сигналы задаются с помощью панели значений (value):




Значения пиктограмм отображены в таблице 2.

Таблица 2

Пикто- грамма	Функция	Горячие клавиши	Комментарий
	Forcing Unknown (X)	Ctrl+Alt+X	Сильная неопределенность
	Forcing Low (0)	Ctrl+Alt+0	Сильный логический '0'
	Forcing High (1)	Ctrl+Alt+1	Сильная логическая '1'
	High Impedance (Z)	Ctrl+Alt+Z	Высокий импеданс (высокое выходное сопротивление)
	Weak Low (L)	Ctrl+Alt+L	Слабый логический '0'
	Weak High (H)	Ctrl+Alt+H	Слабая логическая '1'
	Invert	Ctrl+Alt+I	Инверсия сигнала
	Count Value...	Ctrl+Alt+V	Изменение сигнала (время, инкремент)
	Overwrite Clock...	Ctrl+Alt+K	Периодический сигнал (период, смещение и скважность)
	Arbitrary Value...	Ctrl+Alt+B	Заданное пользователем значение
	Random Values...	Ctrl+Alt+R	Случайное значение

Задать значения сигналов для симуляции можно 4-мя способами.



**Первый** способ задания сигналов: выделите фрагмент сигнала в поле эпюр и нажмите на требуемую по смыслу кнопку на панели значений. Для примера зададим сигнал **CLK** в виде периодически изменяющегося сигнала. Выделите все поле сигнала **CLK** двойным кликом ЛКМ на поле эпюр и выберите кнопку . Появится окно настройки сигнала **Clock**, в котором зададим период сигнала значением **40 ns** (рис. 37). Выполните «Ok», в окне редактора эпюр появится периодический сигнал частотой 25Mhz (рис. 38)

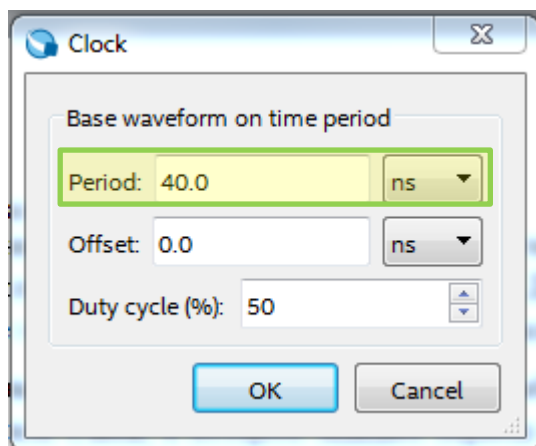


Рис.37 Окно задания сигнала с значением **Overwrite Clock...**

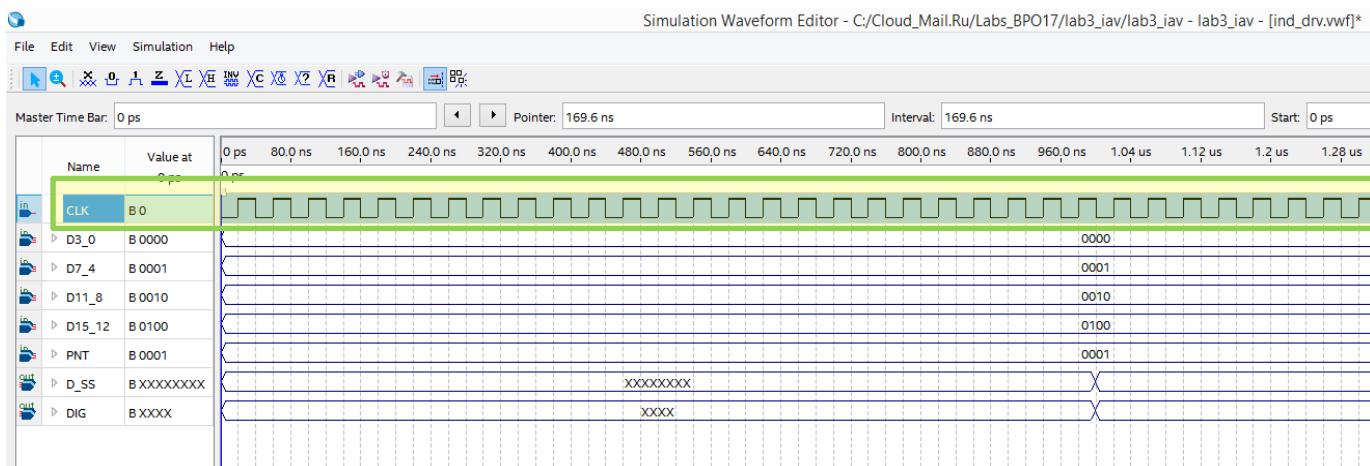


Рис.38 Эпюра сигнала CLK после задания его параметров

**Второй** способ задания сигналов:

1. Выделите всю строку сигнала в поле эпюр двойным кликом ЛКМ или фрагмент сигнала. Для этого нажмите ЛКМ в поле эпюр на строке выбранного сигнала (**D3\_0**) и потяните вправо, формируя при этом прямоугольное окошко нужной длины, отпустите ЛКМ, при этом останется выделенный фрагмент (рис. 39).
2. В выделенном сигнале (или фрагменте) нажмите ПКМ и в выпадающем меню выберите строку **Value**, появится еще одно меню, из которого можно выбрать нужные значения (рис. 40). В данном примере значение **Arbitrary value** (значение задаваемое пользователем). Задайте значение **binary**, “0000” (рис. 41).

**Третий** способ задания сигнала: также как и во втором способе выделите сигнал и задайте его значение через опции основного меню экрана **Edit -> Value ->....**

**И четвертый** способ, это использование комбинации клавиш при выделенном сигнале (см. выше в описании кнопок значений сигналов, таблица 2).

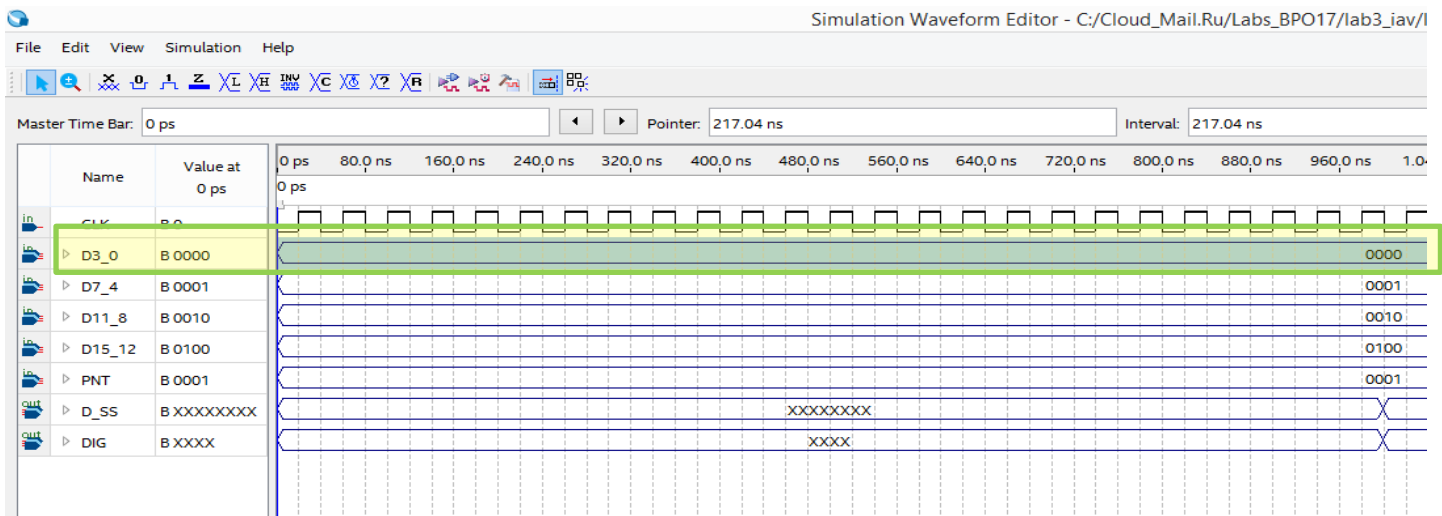


Рис.39 Выделенный сигнал **D3\_0** в поле эпор

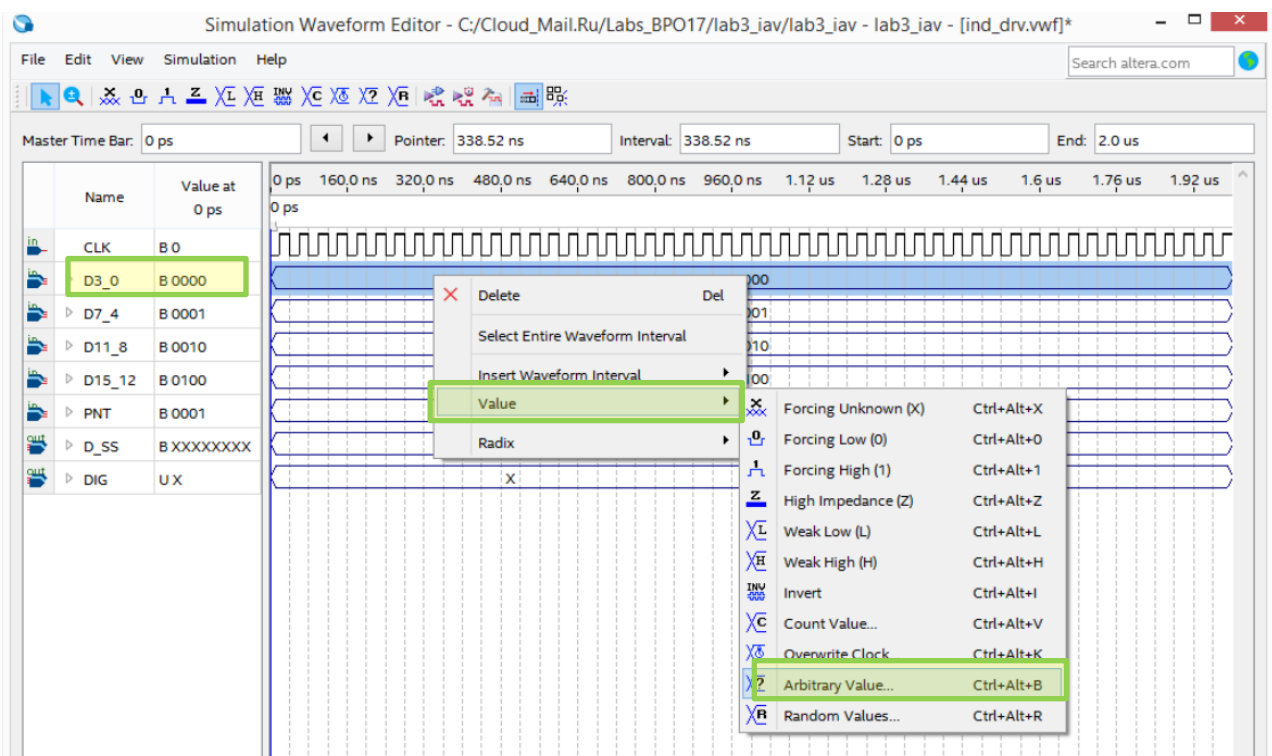


Рис.40 Определение типа значения выбранного сигнала

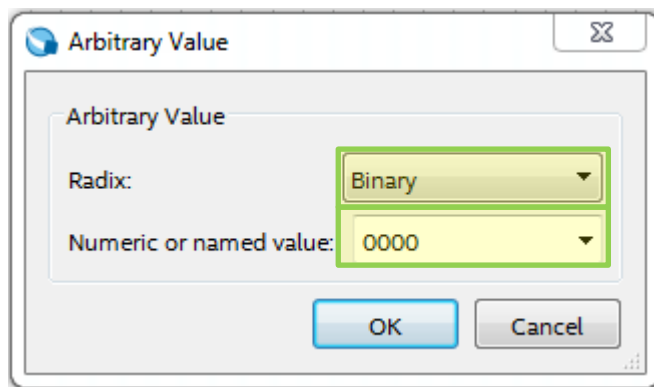


Рис.41 Определение значения сигнала выбранного типа

6.8. Задайте значения для остальных сигналов (**Value -> Arbitrary** или **X?**):

<b>D7_4</b>	<b>binary</b>	<b>0001</b>
<b>D11_8</b>	<b>binary</b>	<b>0010</b>
<b>D15_12</b>	<b>binary</b>	<b>0100</b>
<b>PNT</b>	<b>binary</b>	<b>0001</b>

Окончательно, тест должен иметь вид как на рис. 42.

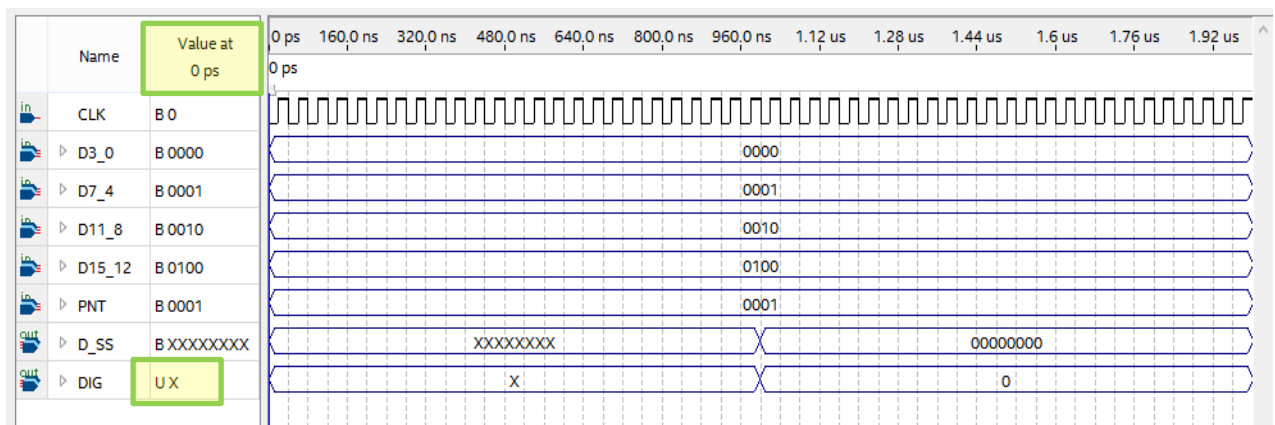


Рис.42 Эпюры тестовых сигналов

В колонке **Value at 0ps** можно выбрать систему счисления сигналов. Для этого нажмите ЛКМ на сигнал в этой колонке, вся строка при этом высветится, затем нажмите ПКМ в этой же колонке и в выпадающем меню выберите строку **Radix**. В выпавшем меню можно выбрать требуемую систему счисления сигнала (рис. 42, 43).

Другой способ задания системы счисления сигнала: также выделить сигнал и выбрать систему счисления через опции основного меню экрана **Edit->Radix->....**

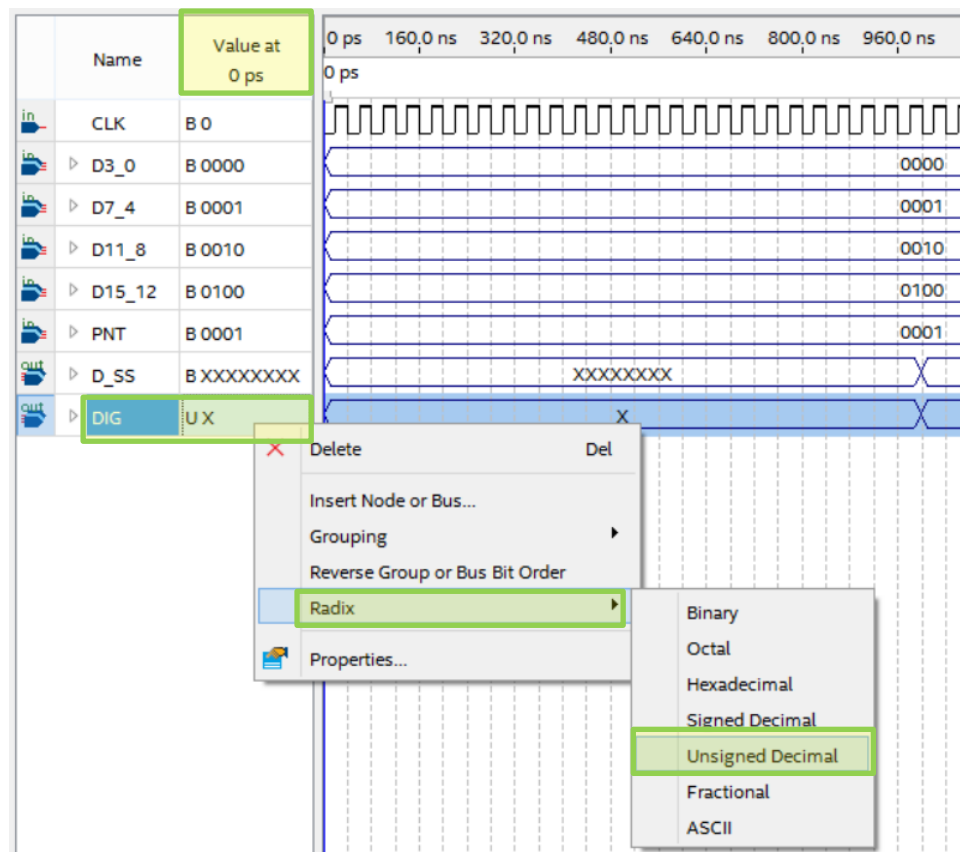


Рис.43 Выбор системы счисления для отображения значений сигналов

6.9. Выполните в окне редактора эюр (рис. 44)

**Simulation -> Simulation Settings -> Restore Defaults-> Save.**

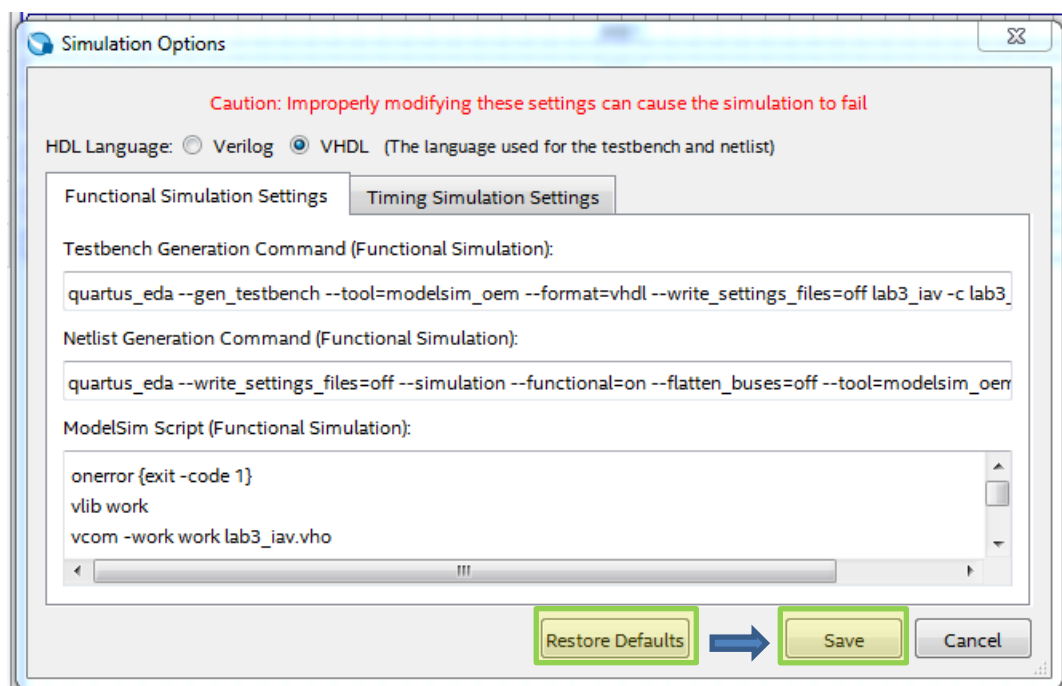



Рис.44 Настройка путей к файлу симуляции

6.10. Запустите симуляцию нажатием на кнопку  (**Run Functional Simulation**) или выполните **Simulation -> Run Functional Simulation**.

На предложение во всплывшем окошке сохранить изменения в файле симуляции нажмите “Ok”. Запустится функциональная симуляция, появится окно **Simulation Flow Progress** с отображением выполнения команд текущего процесса. В случае успешной симуляции окно **Simulation Flow Progress** закроется и откроется новое окно редактора эппур (Read-Only) с результатами симуляции (рис. 45). В этом окне можно только просмотреть результаты симуляции, для модификации эппур надо вернуться в первое окно симулятора, где они создавались. Обратите внимание на значение “1” в старшем бите сигнала **D\_SS[7..0]**, когда выбирается 1-й разряд индикатора. Этот бит отвечает за свечение точки на индикаторе и определяется значением сигнала **PNT[3..0]**. В данном примере, он содержит единицу только для 1-го разряда индикатора. Поменяйте значение сигнала **PNT[3..0]** на “0100”, выполните симуляцию и убедитесь, что теперь значение логической “1” находится в старшем бите 3-го разряда индикатора. Сохраните скриншот для отчета.

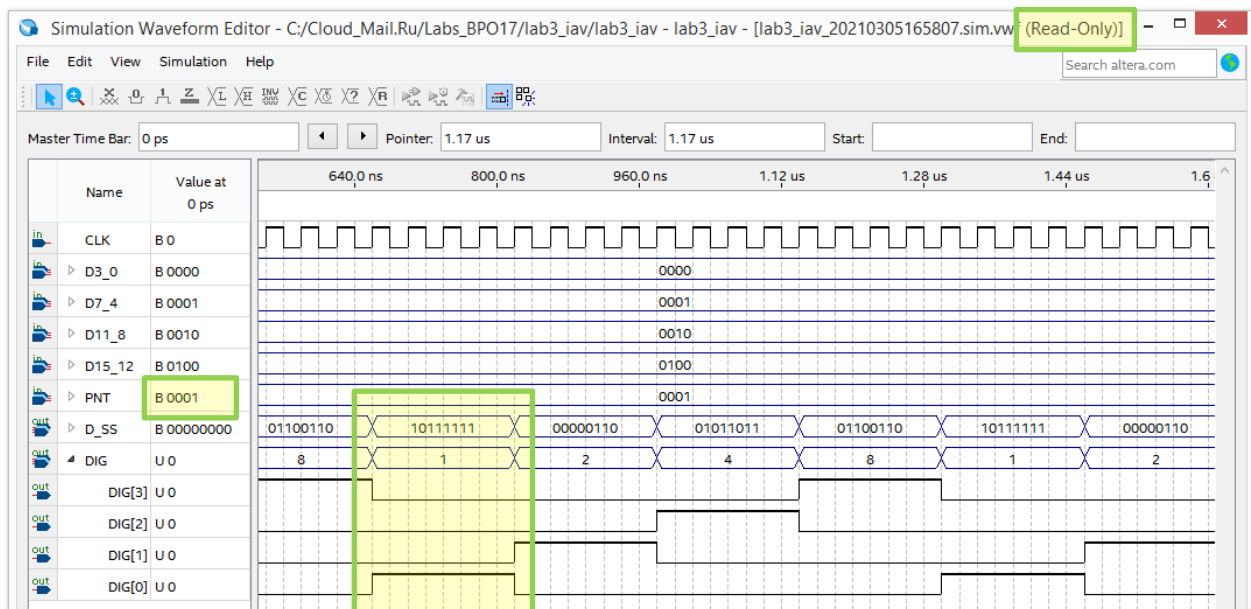


Рис.45 Результат функциональной симуляции компонента

Если окно **Simulation Flow Progress** после симуляции не исчезнет и в нем появятся сообщения об ошибках (рис. 46), выполните следующую процедуру:

- закройте окно **Simulation Flow Progress**
- закройте окно редактора эппур **Simulation Waveform Editor**
- выполните еще раз задачу **Analysis@Synthesis** (проконтролируйте, чтобы проектом верхнего уровня был файл **ind\_drv.bdf**). При этом обновятся все изменения в проекте, связанные с симуляцией.
- после успешной компиляции откройте файл **ind\_drv.vwf** и в открывшемся окне **Simulation Waveform** выполните **Simulation -> Run Functional Simulation**
- убедитесь в правильности результатов симуляции

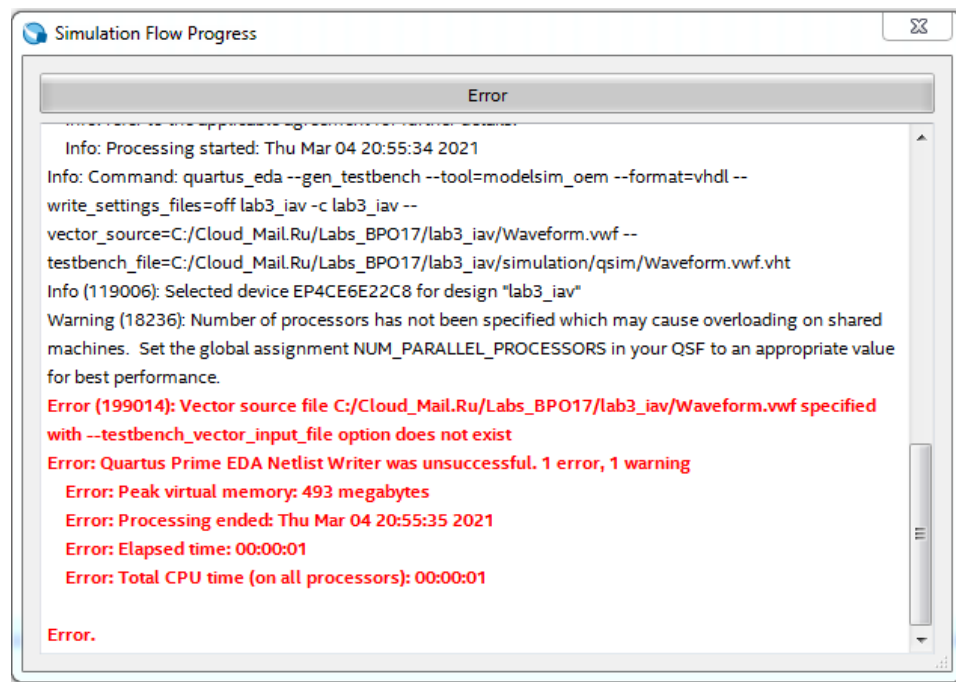


Рис.46 Окно **Simulation Flow Progress** редактора симуляции с сообщением об ошибке

## 7. Создание иерархического проекта и его функциональное моделирование

7.1. Выполните **File -> New** или **Create New Design** в окне **Task**, выберите **Block\_Diagram/Schematic File**, далее «OK». В результате откроется окошко схематического редактора (**Schematic Editor**).

7.2. Нарисуйте схему проекта верхнего уровня на базе созданного компонента **ind\_drv** (рис. 47). Это будет иерархический проект из 2-х уровней.

7.3. Сохраните файл со схемой **File -> Save As** с именем **lab3\_iav.bdf**.

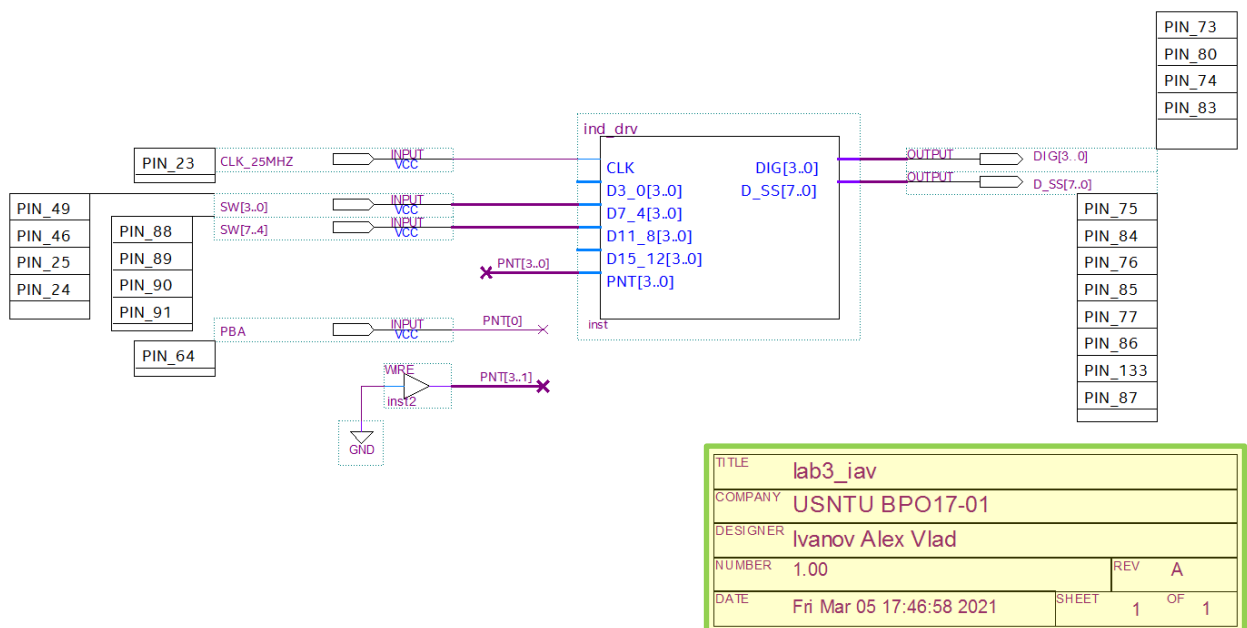


Рис. 47 Схема дизайн-файла проекта **lab3\_iav**

Номера выводов появятся только после назначения номеров выводов микросхемы, к которым подключаются сигналы схемы (см. п.8.2.). Видимость номеров микросхемы на схеме управляется следующим образом: кликните ПКМ на поле редактора схемы, выберите **Show -> Show Location Assignments**.

## 8. Функциональное моделирование проекта

8.1. Установите файл *lab3\_iav.bdf* проектом верхнего уровня:

**Project -> Set as Top-Level Entity.**

8.2. Выполните компиляцию **Analysis@Synthesis**. Обратите внимание на имя проекта верхнего уровня. Устраните ошибки при необходимости.

8.3. Выполните **File -> New**, откроется окошко с выбором типов создаваемых файлов. Выберите **University Program VWF**, далее «OK». В результате откроется окошко редактора эпюр симуляции (**Simulation Waveform Editor**).

8.4. Сохраните созданный файл под именем *lab3\_iav.vwf* (**File->Save\_As**). В окошке редактора сделайте следующие настройки:

- шаг сетки на экране      **10 ns:**      **Edit -> Grid Size;**
- время симуляции      **1 us:**      **Edit -> Set End Time;**

8.5. Задайте в колонке **Name** редактора эпюр имена выводов и сигналов проекта. Для этого выполните: **Edit-> Insert-> Insert Node or Bus** (или в поле колонки «Name», кликните ПКМ и выберите из выпадающего списка «**Insert Node or Bus**»). В появившемся окошке выберите «**Node Finder...**». Далее в появившемся окошке **Node Finder** (рис. 48) в окошке **Filter** выберите «**Pins: all**», нажмите «**List**».

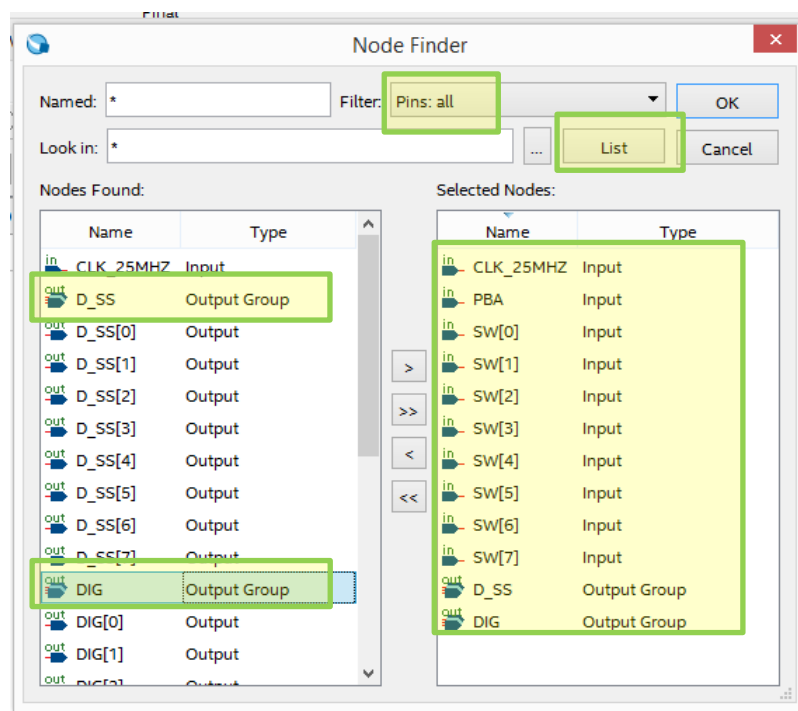






Рис.48 Выбор сигналов для симуляции

В окошке **Nodes Found** выделите необходимые сигналы и перенесите в окошко **Selected Nodes** с помощью кнопки  (по одному сигналу или группу выделенных сигналов, при этом можно сразу упорядочить сигналы, чтобы было удобнее наблюдать их при анализе результатов симуляции) или кнопку  (все сигналы). Попробуйте сразу упорядочить сигналы, это позволит при анализе эюр сигналов лучше понять работу схемы. Многоразрядные выходные сигналы выберите как групповые. Входные выберите поразрядно. Далее дважды выполните «Ok».

Экран редактора симуляции эюр приобретет вид как на рисунке 49. При этом входные сигналы имеют по умолчанию значение , а выходные сигналы не определены .

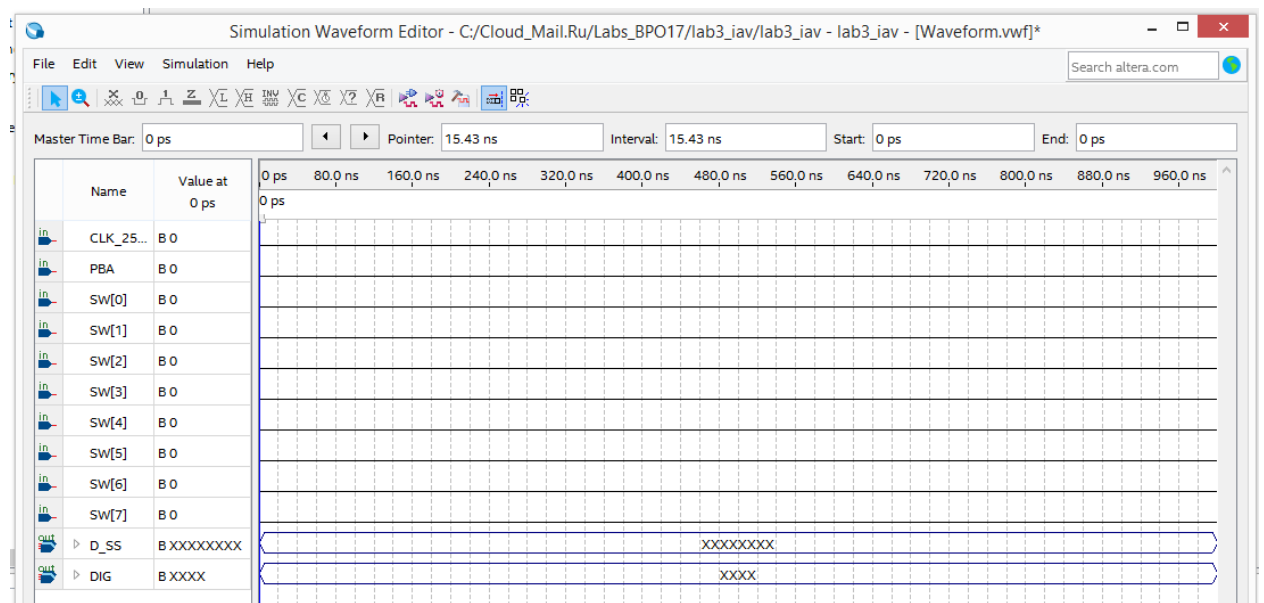


Рис.49 Экран редактора симуляции с именами выбранных для симуляции сигналов

Упорядочьте и сгруппируйте сигналы SW[0] – SW[7] (рис.50- 51).

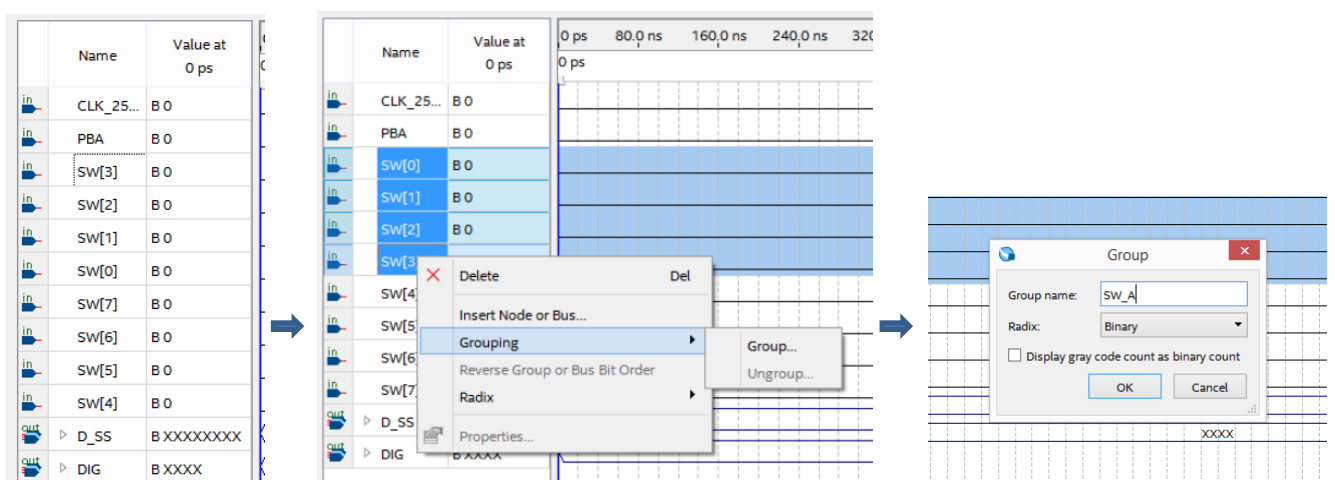


Рис.50 Упорядочивание, группирование и именование сигналов



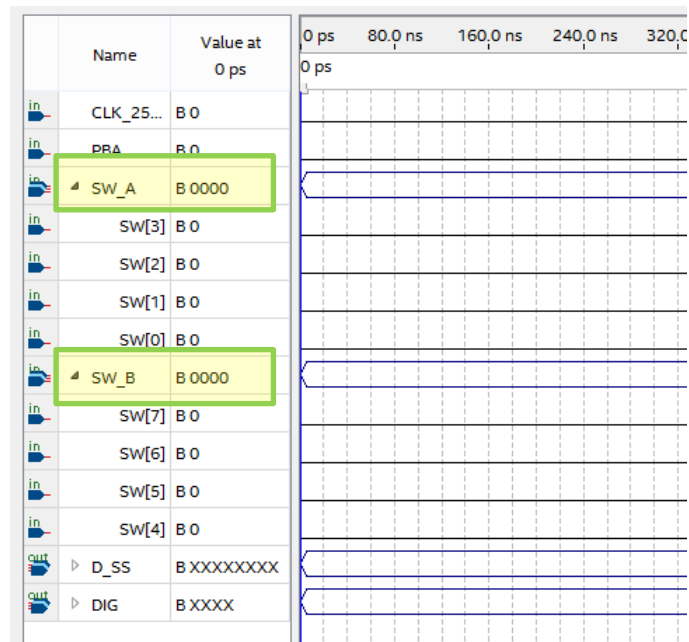



Рис.51 Окончательный вид колонки Name после группирования сигналов

Для этого выделите нужный сигнал ЛКМ и, не отпуская кнопки, перетащите имя сигнала в нужное место, затем отпустите ЛКМ. Далее сгруппируйте сигналы в группы **SW\_A** (сигналы **SW[3]**, **SW[2]**, **SW[1]**, **SW[0]**) и **SW\_B** (сигналы **SW[7]**, **SW[6]**, **SW[5]**, **SW[4]**). Группа создается следующим образом: выделите **Ctrl + ЛКМ** все нужные сигналы группы, затем нажмите **ПКМ**, выберите **Grouping-> Groupe** и в появившемся окне введите имя группы, далее «**Ok**». После создания группы можно закрыть ее, чтобы отображался только общий сигнал группы.

8.6. Задайте сигнал **CLK\_25MHZ** как периодический с периодом 40ns (  ),

Задайте значения групп **SW\_A = "0111"** и **SW\_B="0001"** (выделите поле сигнала нужной группы двойным кликом ЛКМ -> ПКМ-> **Value -> Arbitrary Value**).

Значение **PBA** задайте как на рис. 52. Сохраните созданный тест.

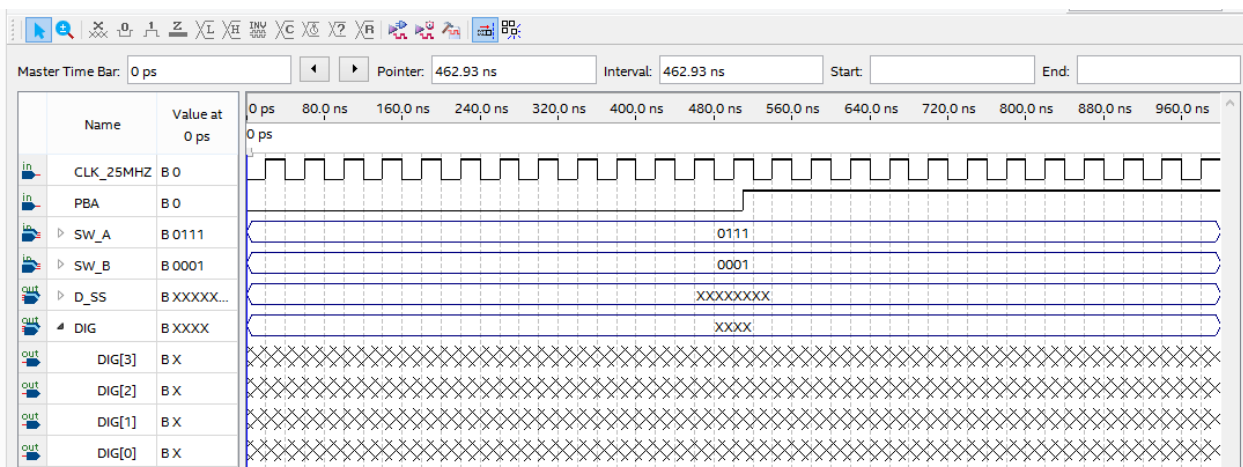



Рис.52 Окончательный вид теста для симуляции

## 8.7. Выполните в окне редактора эппюр

**Simulation -> Simulation Settings -> Restore Defaults-> Save.**

8.8. Запустите симуляцию нажатием на кнопку  (**Run Functional Simulation**) или выполните **Simulation -> Run Functional Simulation**.

На предложение во всплывшем окошке сохранить изменения в файле симуляции нажмите “Ok”. Запустится функциональная симуляция, появится окно **Simulation Flow Progress** с отображением выполнения команд текущего процесса. В случае успешной симуляции окно **Simulation Flow Progress** закроется и откроется новое окно редактора эппюр (Read-Only) с результатами симуляции (рис. 53). В этом окне можно только просмотреть результаты симуляции, для модификации эппюр надо вернуться в первое окно симулятора, где они создавались. Обратите внимание, что когда сигнал PBA устанавливается в значение логической ‘1’, в старшем бите сигнала **D\_SS[7..0]** для первого разряда индикатора также устанавливается логическая ‘1’. Этот бит отвечает за свечение точки на индикаторе и определяется значением сигнала **PNT[3..0]**. В данном примере, сигналы **PNT[3..1]** подключены к источнику логического ‘0’, а сигнал **PNT[0]** подключен к кнопке **PBA** и при эмуляции ее нажатия (на тесте это когда сигнал PBA переключается из ‘0’ в ‘1’), загорается точка на 1 разряде индикатора. Сохраните скриншот для отчета.

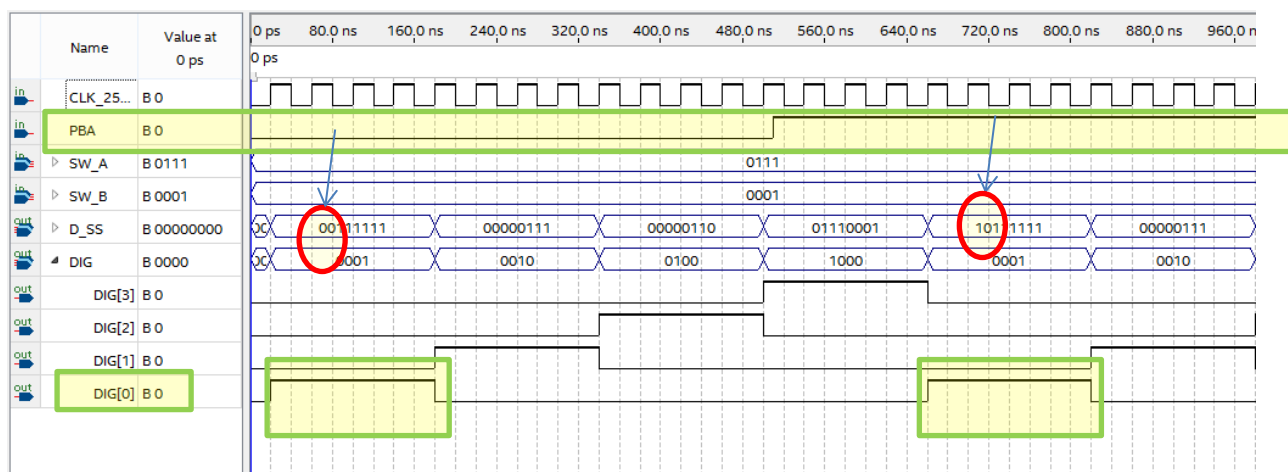



Рис.53 Результат функциональной симуляции схемы проекта

Если окно **Simulation Flow Progress** после симуляции не исчезнет и в нем появятся сообщения об ошибках, выполните следующую процедуру:

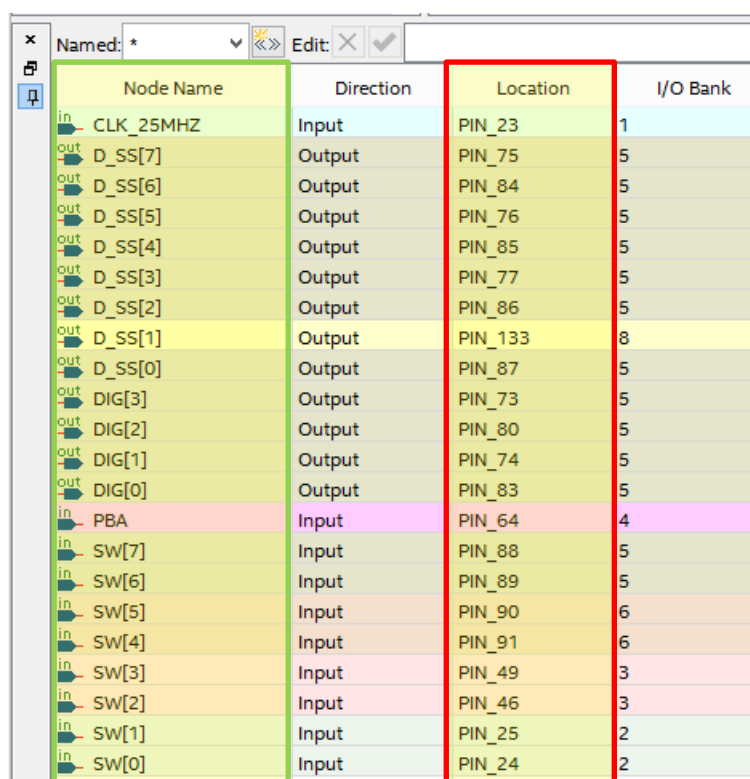
- закройте окно **Simulation Flow Progress**
- закройте окно редактора эппюр **Simulation Waveform Editor**
- выполните еще раз задачу **Analysis@Synthesis** (проконтролируйте, чтобы проектом верхнего уровня был файл **ind\_drv.bdf**). При этом обновятся все изменения в проекте, связанные с симуляцией.
- после успешной компиляции откройте файл **ind\_drv.vwf** и в открывшемся окне **Simulation Waveform** выполните **Simulation -> Run Functional Simulation**
- убедитесь в правильности результатов симуляции

## 9. Назначение выводов ПЛИС

9.1. Откройте редактор **Pin Planner** одним из следующих способов: **Assignments -> Pin Planner** / **Ctrl + Shift + N** / кнопка **Pin Planner**  на панели задач



9.2. В поле **Location** для каждого сигнала необходимо установить номер вывода микросхемы в соответствии со схемой отладочной платы. Для этого в строке выбранного сигнала в поле **Location** кликните ЛКМ и наберите номер вывода как на рис. 54 (набирайте только номер вывода!). Затем выполните **Enter** и выберите следующий сигнал. После того как все сигналы будут привязаны к выводам микросхемы ПЛИС, закройте окно редактора, все изменения автоматически сохранятся в файле проекта **lab3\_iav.qsf** (quartus setting file).



Node Name	Direction	Location	I/O Bank
in CLK_25MHZ	Input	PIN_23	1
out D_SS[7]	Output	PIN_75	5
out D_SS[6]	Output	PIN_84	5
out D_SS[5]	Output	PIN_76	5
out D_SS[4]	Output	PIN_85	5
out D_SS[3]	Output	PIN_77	5
out D_SS[2]	Output	PIN_86	5
out D_SS[1]	Output	PIN_133	8
out D_SS[0]	Output	PIN_87	5
out DIG[3]	Output	PIN_73	5
out DIG[2]	Output	PIN_80	5
out DIG[1]	Output	PIN_74	5
out DIG[0]	Output	PIN_83	5
in PBA	Input	PIN_64	4
in SW[7]	Input	PIN_88	5
in SW[6]	Input	PIN_89	5
in SW[5]	Input	PIN_90	6
in SW[4]	Input	PIN_91	6
in SW[3]	Input	PIN_49	3
in SW[2]	Input	PIN_46	3
in SW[1]	Input	PIN_25	2
in SW[0]	Input	PIN_24	2

Рис. 54 Окно редактора **Pin Planner** с привязкой сигналов проекта к выводам ПЛИС

9.3. Все неиспользованные в проекте выводы микросхемы ПЛИС необходимо установить в режим «**AS INPUT TRI-STATED**». Для этого выполните **Assignments-> Device-> Device and Pin Options**. В появившемся окне **Device and Pin Options** выберите в поле **Category** строку **Unused Pins** и в окошке **Reserve all unused pins** выберите **As input tri-stated with weak pull-up**. После назначения выводов у символов ввода-вывода на схеме появятся номера выводов (**PIN\_nn**) микросхемы, к которым они привязаны (рис. 47).

## 10. Создание sdc – файла

10.1. Создайте файл с информацией о временных требованиях (т.е. на какой частоте проект должен работать гарантированно) к проекту:

**File =>New => Other Files => Synopsys Design Constraints File**

10.2. Сохраните созданный текстовый проектный файл **File =>Save As** с именем **lab3\_iav.sdc** . Минимальный набор команд, который должен быть в нем определен, показан на рисунке 55.

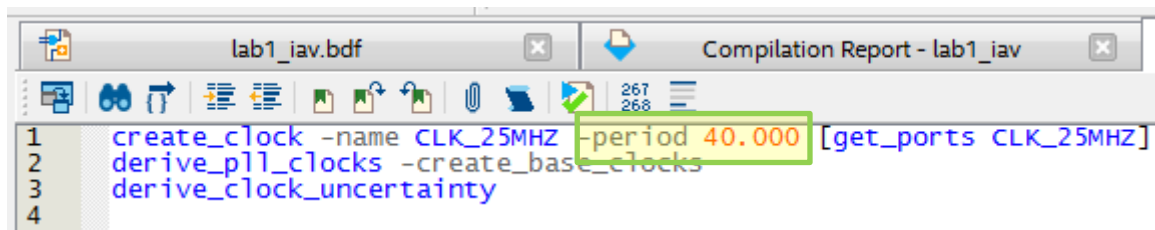


Рис. 55 Задание временных требований к проекту

Скопируйте и внесите в файл следующие строки (чтобы не набирать вручную):

```
create_clock -name CLK_25MHZ -period 40.000 [get_ports CLK_25MHZ]
derive_pll_clocks -create_base_clocks
derive_clock_uncertainty
```

В первой строке задана частота, которой должен удовлетворять проект. Данная частота (25MHz) поступает на вход ПЛИС с выхода генератора прямоугольных сигналов отладочной платы.

### **Подсказка.**

Откройте **lab1\_iav.sdc** из проекта **lab1\_iav** (**File-> Open-> <path project/ lab1\_iav/...>**). Сохраните под именем **lab3\_iav.sdc** в папке проекта **lab3\_iav** и добавьте в текущий проект **lab3\_iav**. Важно, чтобы имена сигналов, для которых определены временные требования, в обоих проектах совпадали. При необходимости, отредактируйте значения временных параметров и имена сигналов.

10.3. Сохраните файл и проследите, чтобы файл появился в окне **Project Navigator** (Выберите вкладку **Files**). В противном случае, при открытом окне с файлом **lab3\_iav.sdc** выполните **Project->Add Current File to Project**. Файл **lab3\_iav.sdc** добавится в проект.

## 11. Выполнение полной компиляции проекта

11.1. После назначения сигналов на выходы микросхемы необходимо выполнить разводку схемы на кристалле ПЛИС (**Fitter**). Для этого выберите задачу **Compile Design**

из окна задач **Tasks** или нажмите кнопку  на панели задач или выполните **Ctrl+L** или

**Processing->Start Compilation.** Запустится компиляция, результаты которой можно наблюдать в окне **Compilation Report** (рис. 56). В окне отображаются основные параметры проекта и затраченные на его реализацию ресурсы микросхемы (количество логических элементов и использованных в них триггеров, количество выводов микросхемы, количество встроенных блоков памяти, умножителей и PLL).

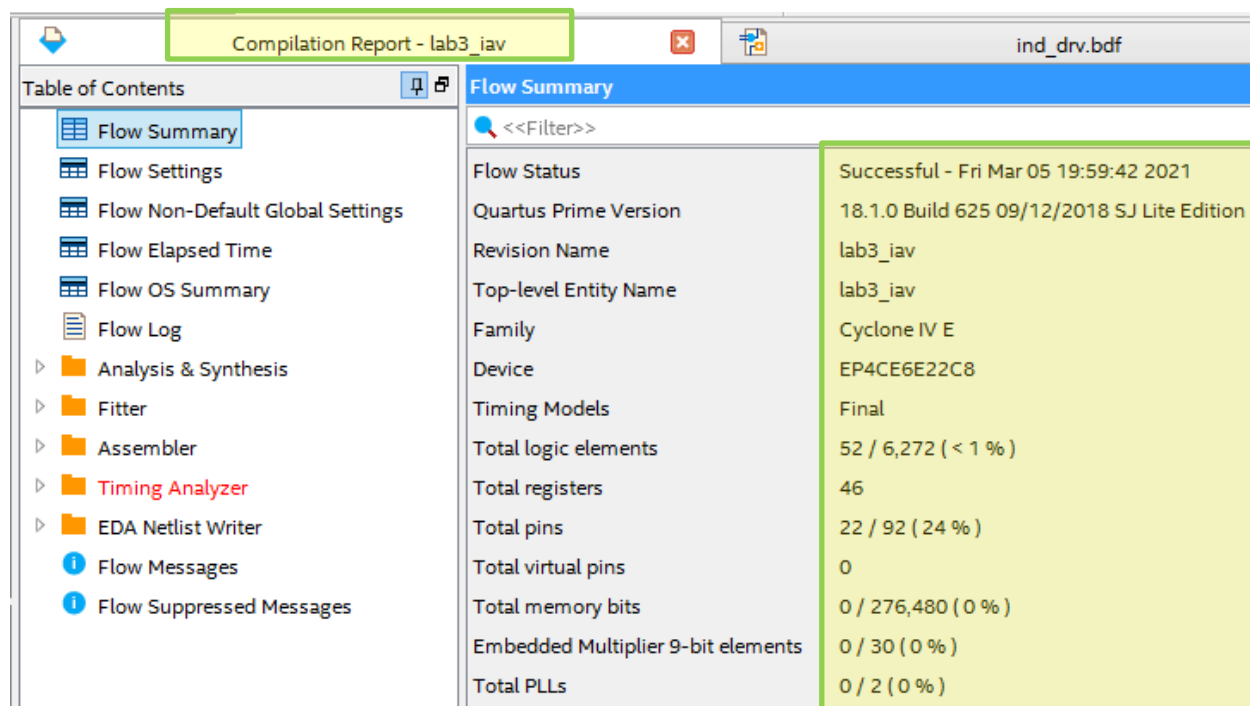


Рис. 56 Окно **Compilation Report** с результатами полной компиляции

11.2. В случае успешной компиляции в окне **Message** появится информация об ее успешном окончании. В противном случае, устраните ошибки.

11.3. В окне **Compilation Report** откройте файл **Fmax Summary** с результатами временного анализа (рис. 57).

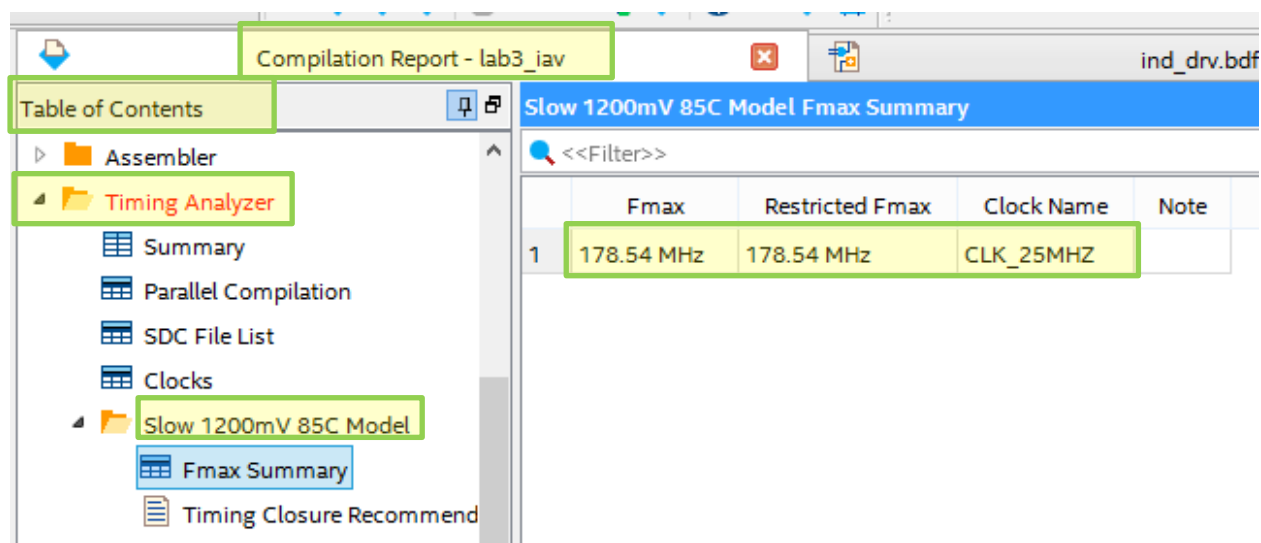



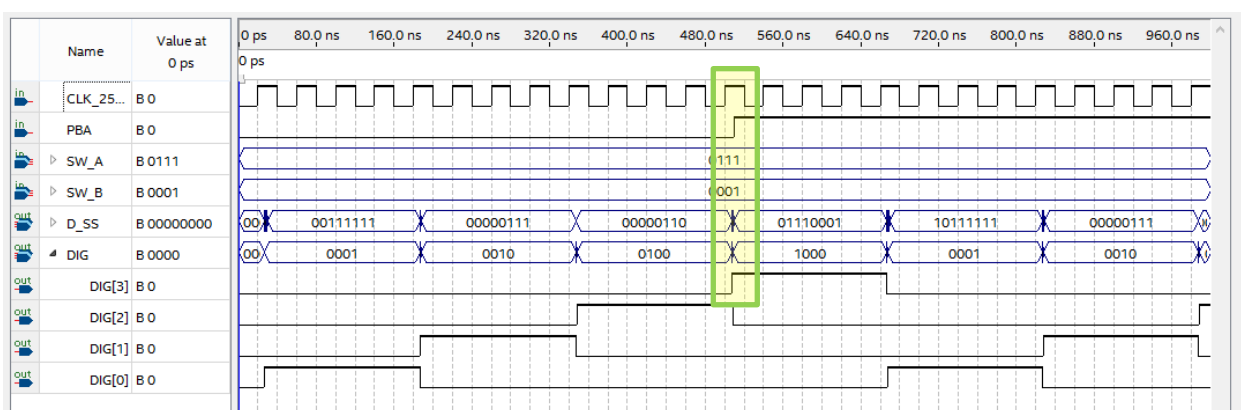
Рис. 57 Окно **Compilation Report** с результатами временного анализа проекта

Убедитесь, что максимально возможная частота работы проекта (Fmax), превышает заданные в файле **lab3\_iav.sdc** временные требования (CLK\_25MHZ).

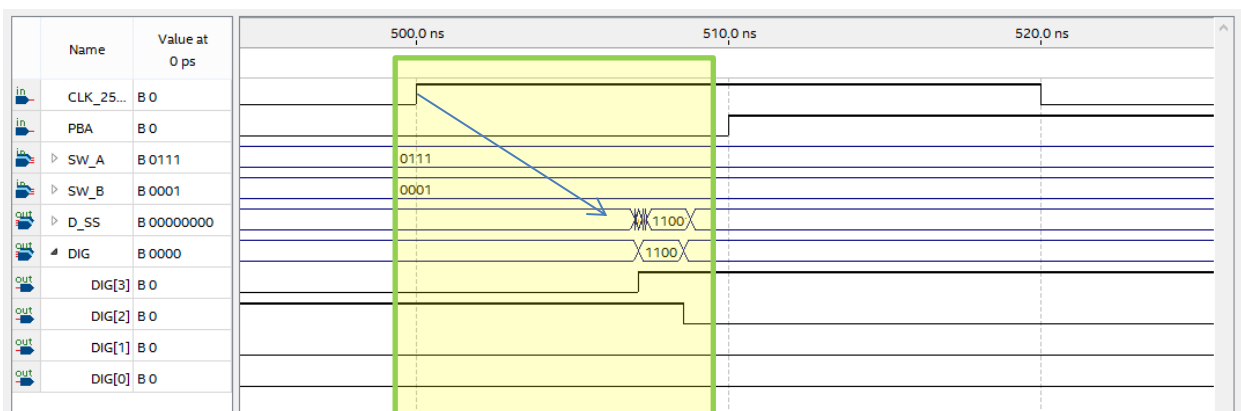
## 12. Временное моделирование проекта

12.1. Откройте файл lab3\_iav.vwf.

12.2. Запустите симуляцию нажатием на кнопку  (**Run Timing Simulation**) или выполните **Simulation -> Run Timing Simulation**. На эякурах с результатами симуляции (рис. 58) видны переходные процессы на выходных сигналах, связанные с разным временем формирования результатов на разрядах сигналов **DIG[3..0]** и **D\_SS[7..0]**. Определите (примерно) время задержки изменения сигналов DIG[3] и DIG[2] относительно переднего фронта сигнала CLK\_25. Занесите результаты в отчет.



a)



b)

Рис.58 Результат временной симуляции схемы проекта в исходном масштабе (a) и в увеличенном масштабе (b)

Если в результате симуляции появится окно **Simulation Flow Progress** с отображением ошибки, выполните **Simulation -> Simulation Settings -> Выберите вкладку Timing Simulation Setting -> Restore Defaults -> Save**. Затем повторите временную симуляцию.

### 13. Изменение параметров схемы проекта и его повторная полная компиляция

13.1. Для возможности симуляции за разумное время, формирование развертки изображения на индикаторе производится в ускоренном режиме. Это обеспечивается формированием высокой частоты сигнала **ENA**, по которому происходит смена состояний автомата по формированию сигналов развертки для индикатора. Эта частота, в свою очередь, формируется счетчиком **Counter\_12b** и зависит от его модуля счета. Для симуляции модуль установлен в значение **4**. Для реальной работы схемы на отладочной плате необходимо частоту сигнала **ENA** снизить, иначе элементы схемы развертки индикатора не будут успевать отрабатывать необходимый алгоритм смены сигналов (не хватит быстродействия переключения транзисторов, например). Поэтому необходимо установить модуль счетчика, формирующего сигнал **ENA**, в большее значение. Для данной схемы приемлимое значение равно **4095**. Модуль счетчика меняется с помощью **MegaWizard Plug-In Manager**.

13.2. Откройте файл со схемой проекта **lab3\_iav.bdf** и дважды кликните ЛКМ на компонент **ind\_drv**. В появившемся окошке **Select File** выберите файл **ind\_drv.bdf** (рис. 59), далее «**Ok**», откроется окно схематического редактора со схемой компонента **ind\_drv**.

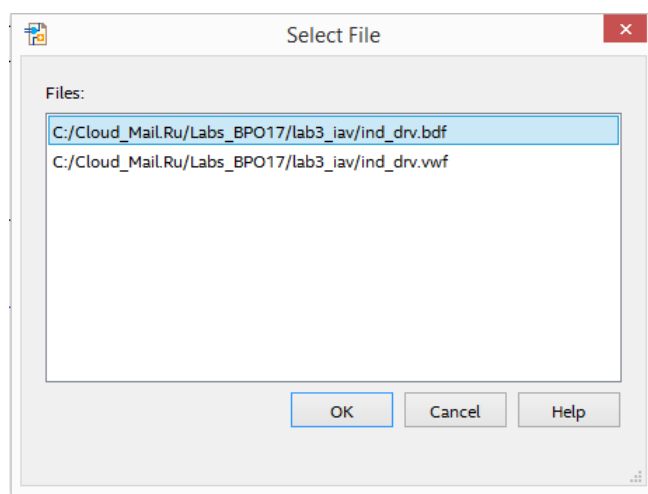


Рис. 59 Выбор файла для изменений

13.3. Дважды кликните на компонент **Counter\_12b**, откроется **MegaWizard Plug-In Manager**. На странице 2 измените значение **modulus** с **4** на 4095 (рис. 60), далее на странице 5 выполните **Finish**. Далее в окне **MegaWizard Plug-In Manager** (рис. 61) выполните «**Ok**», подтвердите **Update** символа **Counter\_12b** (рис. 62а) и в следующем окне Update Symbol or Block выполните «**Ok**» (рис. 62b). На символе компонента появится новое значение модуля: 4095 (рис. 63).

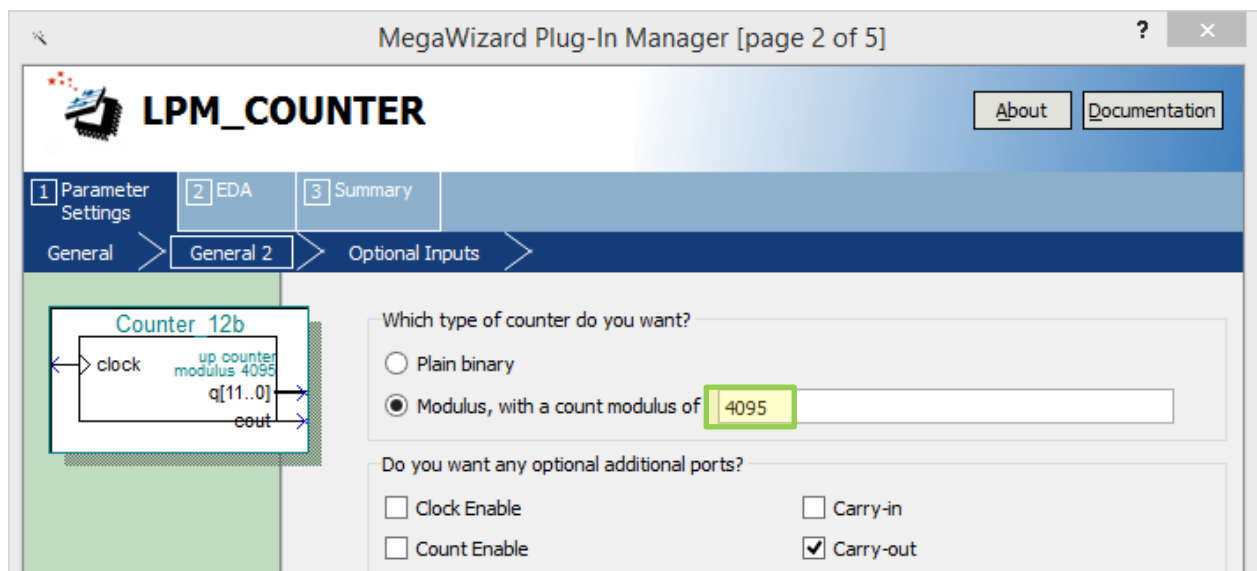


Рис. 60 Изменение модуля счетчика

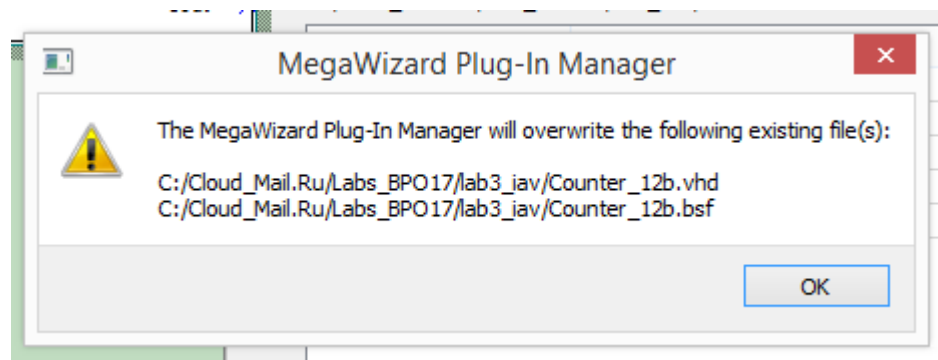
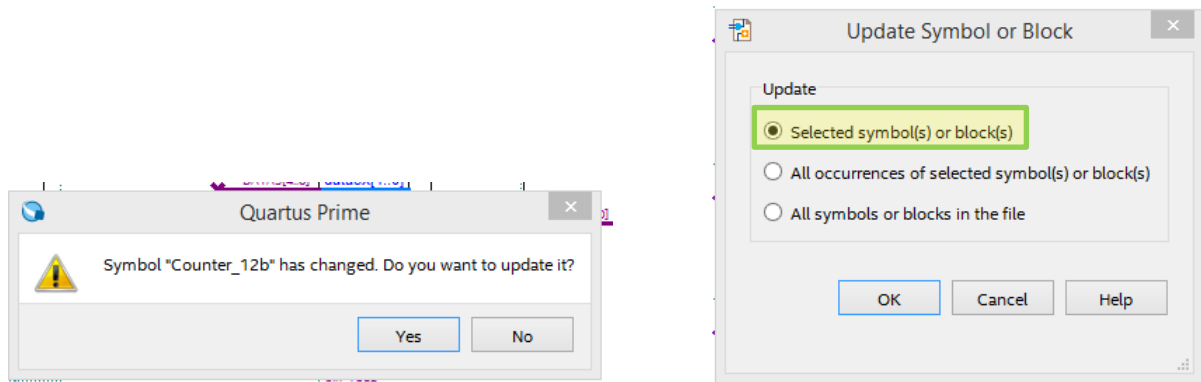


Рис. 61 Обновление компонента Counter\_12b



a

b

Рис. 62 Обновление символа Counter\_12b в схеме



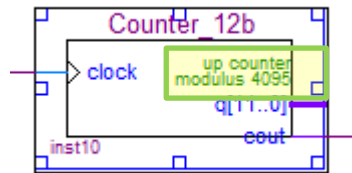


Рис. 63 Обновление символа Counter\_12b в схеме

13.4. Сохраните схему и выполните полную компиляцию.

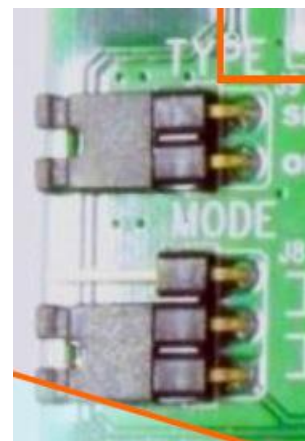
#### 14. Конфигурирование ПЛИС и проверка работоспособности на отладочной плате.

14.1. Подсоедините к отладочной плате модуль 4-х разрядного 7-сегментного светодиодного индикатора.

14.2. Подключите отладочную плату **miniDLab-CIV** к компьютеру с помощью **USB** кабеля (тип A–miniB), и включите питание (рис. 64a). Проверьте правильность установки перемычек (рис. 64b).




а



б

Рис. 64 Переключатель Power(b) и конфигурация разъемов установки режимов работы платы (а)

14.3. Откройте окно программатора **Programmer** из окна **Tasks (Program Device)** или кнопкой  или **Tools->Programmer** (рис. 65).

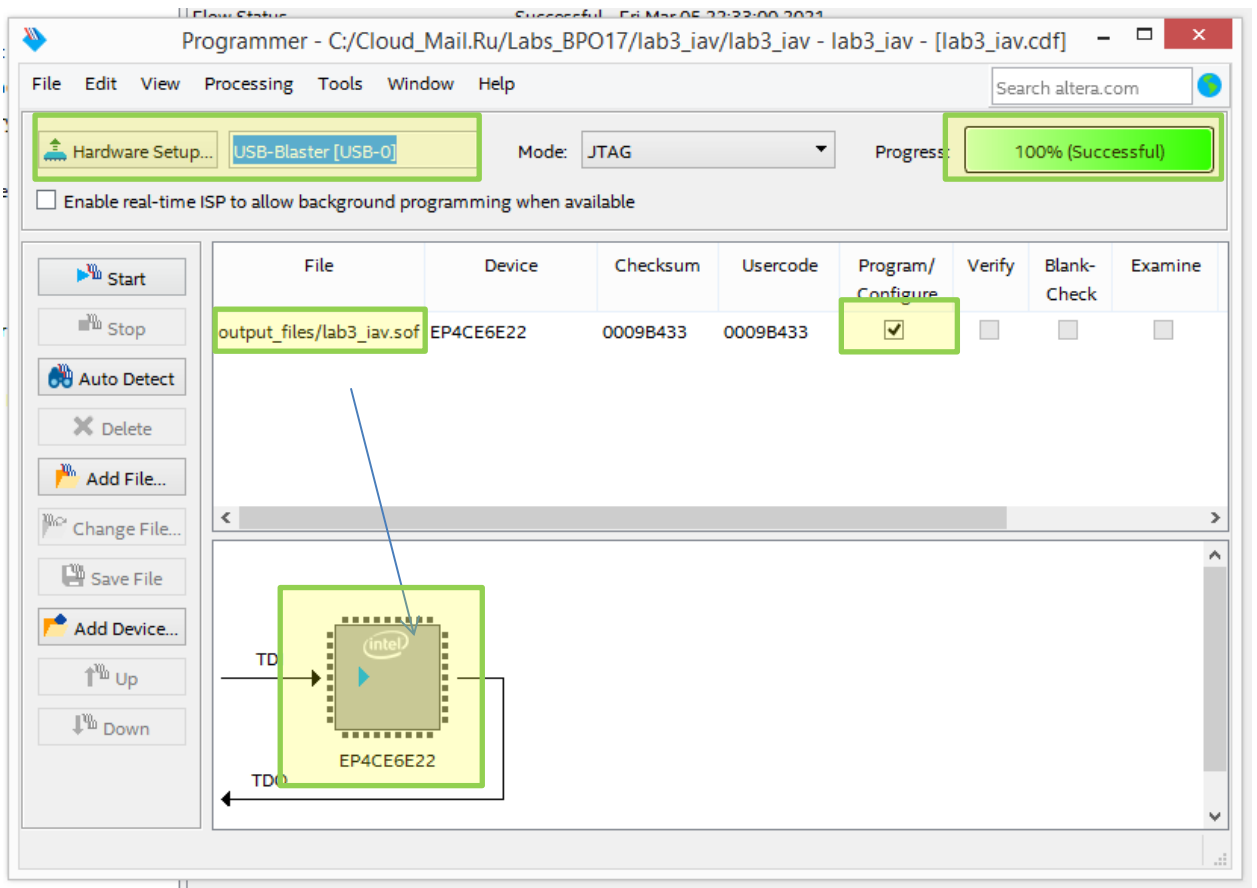


Рис.65 Окно программатора

14.4. Если **USB blaster** в программаторе не виден, выберите его через кнопку **Hardware Setup**.

14.5. Если конфигурационный файл автоматически не загрузился, выберите его с помощью кнопки **Add File** из папки **output\_files** проекта, он хранится в файле с расширением **sof** (SRAM Object File) **lab3\_iav.sof**.

14.6. Выполните **Start**, в окошке **Progress** будет виден текущий статус загрузки.

14.7. После успешной загрузки конфигурации ПЛИС проверьте работу проекта. Схема работает следующим образом:

на 4-м разряде будет высвечиваться символ 'F',

на 3-м разряде будет высвечиваться код с переключателей SW7-SW4 в шестнадцатиричном коде (hexadecimal),

на 2-м разряде будет высвечиваться код с переключателей SW3-SW0 в шестнадцатиричном коде (hexadecimal),

на 1-м разряде будет высвечиваться символ '0', светится точка при не нажатой кнопке РВА и при нажатой кнопке РВА точка гаснет.

14.8. Выключите и отсоедините отладочную плату от компьютера.

Лабораторная работа завершена.

Общие требования к отчету:

1. Отрадите в отчете порядок выполнения лабораторной работы и занесите в него результаты выполнения ключевых этапов в виде рисунков (путем копирования необходимой информации с экрана монитора) и текстовых пояснений.
2. В выводах отразите суть выполненной работы и полученные навыки.
3. Проанализируйте полученные данные, если это требуется в задании, и сделайте по ним выводы.