

Упражнение 5

Цель упражнения:

Знакомство с возможностями пакета QuartusII по заданию установок, назначений, анализу реализации проекта и использованию базовых элементов СБИС Cyclone IV.

Алгоритм работы проекта:

- Проект обеспечивает отображение на 7-сегментном индикаторе 3-разрядных десятичных чисел - результат умножения двух 4-разрядных двоичных чисел, задаваемых с переключателей:
 - Число В, задаваемое переключателями sw[3..0]
 - Число А, задаваемое переключателями sw[7..4]
- Режим отображения – динамический.

Часть 1 – Создание проекта

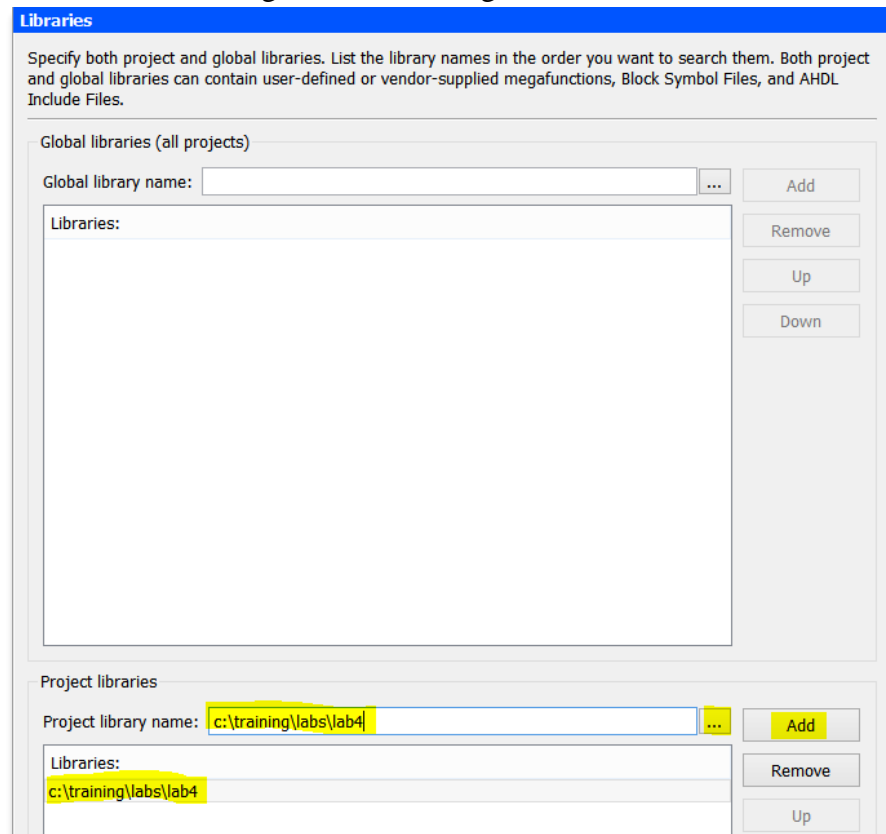
1. Проект:

- Рабочая папка – ... \ lab5;
- Имя проекта – lab5;
- Имя модуля верхнего уровня в иерархии проекта –lab5;
- СБИС ПЛ – *EP4C6E22C8*;

Часть 2 – Подключение внешней библиотеки

1. В проекте будет использован модуль ss_cntr, созданный в упражнении 4. Поэтому папку упражнения 4 необходимо подключить к проекту как библиотечную папку:

- Выполните команд: Assignments=>Settings=>Libraries



- В разделе Project libraries с помощью браузера найдите рабочую папку упражнения 4 и добавьте ее к библиотекам проекта (кнопка add)
Внешнюю библиотеку можно было подключить при создании проекта.

Часть 2 - Создание элементов (используя помощник)

1. Создайте умножитель.

- Имя - mult
- Мегафункция – LPM_MULT (раздел Basic Function=> Arithmetic)
- Разрядность входов – 4 бита (page 1)
- Включить опцию – Unsigned (page 2)
- Включить опцию – Use dedicated multiplier circuitry (page 2)
- Конвейеризация – I want output latency - 2 такта (page 3)
- Включить опцию создания bsf символа (page 5)

2. Создайте mif файл:

- Команда File=>New=>Memory Initialization file

- Задайте количество слов модуля памяти -256, разрядность слов -16
 - Установите систему счисления данных – unsigned decimal; отобразите массив памяти в виде 1 колонки (1 Cell per Row).
 - Откройте bin_bd.xlsx файл. Файл содержит таблицу преобразования 8-разрядного двоичного кода в 3-тетрадный двоично-десятичный код. Скопируйте содержимое колонки A в созданный mif файл
 - Сохраните файл под именем bin_bd.mif .
3. Создайте компонент ROM и свяжите его с файлом инициализации памяти:
- Модуль ROM: 1-PORT (раздел Basic Function=>On Chip Memory)
 - Имя – ROM.
 - Задайте число слов (256), разрядность слов (16) – page 1
 - Выберите опцию – выход синхронный - Which ports should be registered? ‘q’ output port - page 2
 - Файл инициализации памяти – bin_bd.mif (page 3)
 - Включите опцию создания bsf символа (page 5)
4. Создайте экземпляра умножителя тактовой частоты
- Имя модуля – pll_100
 - Модуль – **ALTPLL** (раздел Basic Function=> Clocks; PLLs and Reset=>PLL)
 - Установите параметры умножителя тактовых сигналов на следующих страницах помощника (*на всех остальных страницах оставьте значения, заданные по умолчанию*):

- page 1 of 12:

General

Which device speed grade will you be using? 8

☐ Use military temperature range devices only

What is the frequency of the inclk0 input? 25.000 MHz

☐ Set up PLL in LVDS mode Data rate: Not Available Mbps

PLL Type

Which PLL type will you be using?

☐ Fast PLL ☐ Enhanced PLL ☒ Select the PLL type automatically

Operation Mode

How will the PLL outputs be generated?

☒ Use the feedback path inside the PLL

☒ In normal mode

☐ In source-synchronous compensation Mode

☐ In zero delay buffer mode

☐ Connect the fbmimic port (bidirectional)

☐ With no compensation

☐ Create an 'fbin' input for an external feedback (External Feedback Mode)

Which output clock will be compensated for? c0

- page 2 of 12:

Optional Inputs

☐ Create an 'pllena' input to selectively enable the PLL

☐ Create an 'areset' input to asynchronously reset the PLL

☐ Create an 'pfdena' input to selectively enable the phase/frequency detector

Lock Output

☒ Create 'locked' output

☐ Enable self-reset on loss lock

Advanced Parameters

Using these parameters is recommended for advanced users only

☐ Create output file(s) using the 'Advanced' PLL parameters

- Configurations with output clock(s) that use cascade counters are not supported

- page 6 of 12:

c0 - Core/External Output Clock

Ability to implement the requested PLL

☒ Use this clock

Clock Tap Settings

☐ Enter output clock frequency:

☒ Enter output clock parameters:

Clock multiplication factor: 4

Clock division factor: 1

Clock phase shift: 0.00 deg

Clock duty cycle (%): 50.00

Requested Settings

100.00000000 MHz

Actual Settings

100.000000

4

1

0.00

50.00

Description

Primary clock VCO frequency (MHz): 60

Modulus for M counter: 24

Per Clock Feasibility Indicators

c0 c1 c2 c3 c4

- Включите опцию создания bsf символа (page 12)
5. Создайте экземпляр блока управления глобальным тактовым сигналом (блок обеспечит выдачу тактового сигнала в глобальную цепь распространения тактового сигнала только после того, как умножитель тактовой частоты закончит настройку – сигнал locked станет равным 1)
- Имя модуля – clk_cntr
 - Модуль – ALTCLKCTRL (раздел Basic Function=> Clocks; PLLs and Reset)
 - на странице page 1 задайте:

Altclkctrl

Altclkctrl represents clock buffers that drive the Global Clock Network, the Regional Clock Network, and the dedicated External Clock path.

How do you want to use the altclkctrl ? For global clock

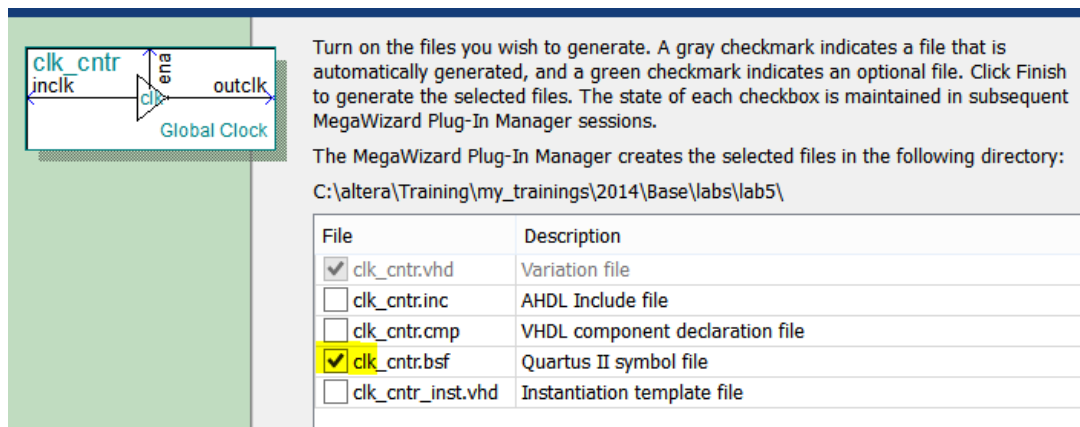
Global clock network allows a clock signal to reach all parts of the chip with the same amount of skew. Input port 'clkselect' can be used to switch between four clock inputs.

How many clock inputs would you like ? 1

☒ Create 'ena' port to enable or disable the clock network driven by this buffer

☐ Ensure glitch-free switchover implementation

- на странице page 3 задайте:



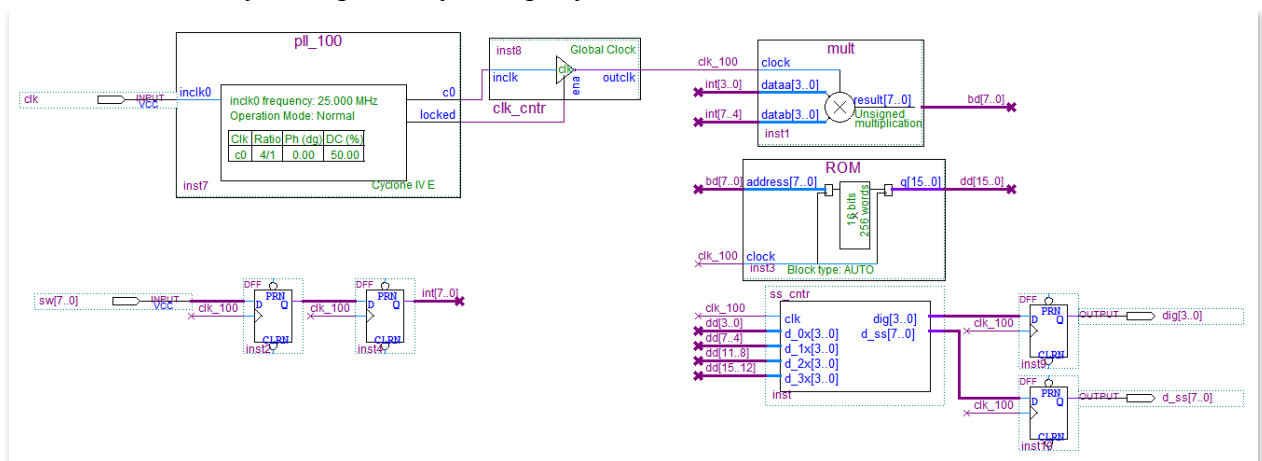
Turn on the files you wish to generate. A gray checkmark indicates a file that is automatically generated, and a green checkmark indicates an optional file. Click Finish to generate the selected files. The state of each checkbox is maintained in subsequent MegaWizard Plug-In Manager sessions.

The MegaWizard Plug-In Manager creates the selected files in the following directory:
C:\altera\Training\my_trainings\2014\Base\labs\lab5\

File	Description
<input checked="" type="checkbox"/> clk_cntr.vhd	Variation file
<input type="checkbox"/> clk_cntr.inc	AHDL Include file
<input type="checkbox"/> clk_cntr.cmp	VHDL component declaration file
<input checked="" type="checkbox"/> clk_cntr.bsf	Quartus II symbol file
<input type="checkbox"/> clk_cntr_inst.vhd	Instantiation template file

Часть 4 - Создание схемы

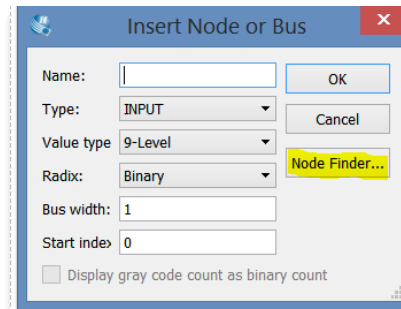
1. Создайте схему, изображенную на рисунке ниже.




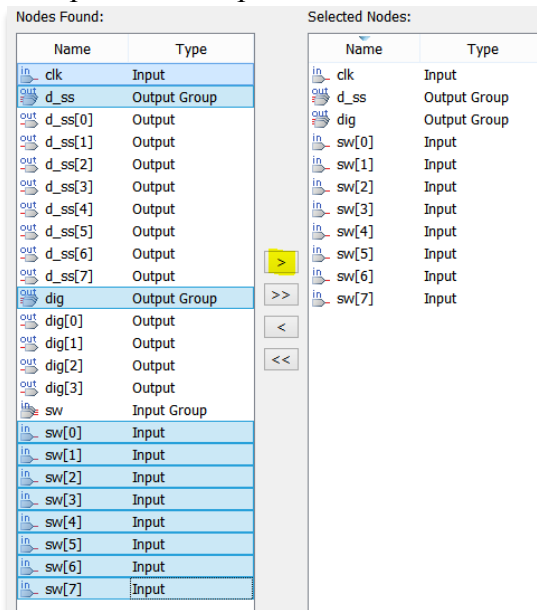
2. Сохраните схему под именем – **lab5.bdf**.
3. Для дальнейшего функционального моделирования проекта следует изменить модуль счета счетчика cnt_div_ss (входит в компонент ss_cntr), обеспечивающего деление входной частоты 25 МГц.
4. В файле lab5.gdf, двойным щелчком по компоненту ss_cntr перейдите на нижний уровень иерархии описания – схему компонента.
5. Двойным щелчком в зоне символа счетчика cnt_div_ss запустите помощник MegaWizard Plug-in Manager
6. В окне настроек счетчика перейдите на страницу 2 и задайте модуль счета равным 4
7. Нажмите кнопку Finish, затем нажмите ее еще раз.
8. В появившемся окне нажмите ОК
9. В следующем окне нажмите кнопку YES
10. Далее укажите: обновить выбранный символ
11. Проверьте в схемном редакторе, что символ счетчика обновлен и все выходы подключены правильно.
12. Сохраните схему.
13. Выполните команду: меню **Processing=>Start=>Start Analysis and Synthesis**.

Часть 5 - Функциональное моделирование

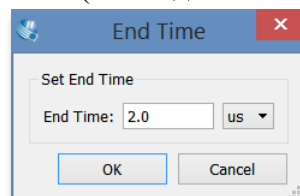
1. Выполните команду **File=>New => University Program VWF**
2. Откроется окно редактора тестовых воздействий
3. Для выбора выводов проекта выполните команду **Edit=>Insert=> Insert Node or Bus** и в появившемся окне запустите **Node Finder**



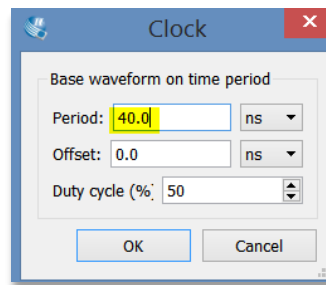
4. Откроется окно **Node Finder**, в котором следует установить фильтр (Filter) – **Pins:all** (все выводы) и нажать кнопку **List**
5. В разделе **Nodes Found** выделить интересующие сигналы и шины
6. Нажать символ  - перенести выбранные сигналы в окно **selected Nodes**



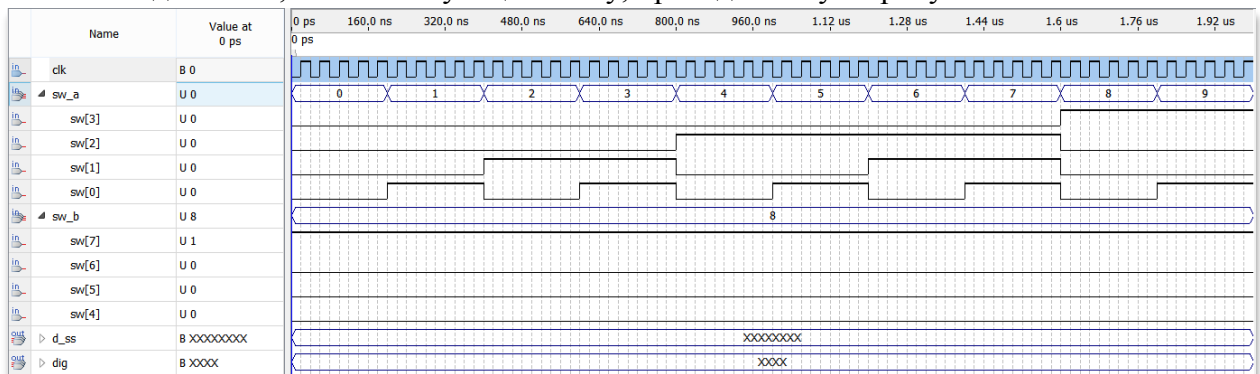
7. Нажмите кнопку **OK**, затем нажмите кнопку **OK** еще раз.
8. Задайте длину теста равной 2000 ns (команда **Edit=>Set End Time**)



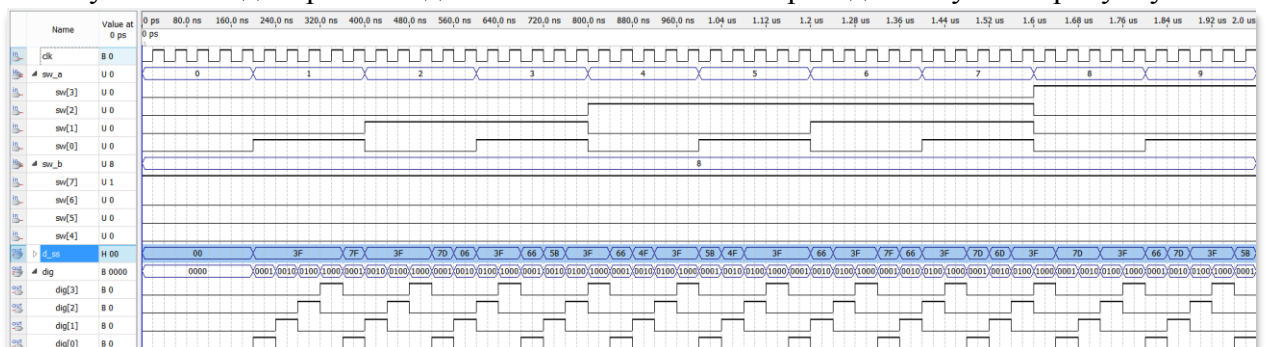
9. Введите тестовые воздействия:
 - Тактовому сигналу (выделите тактовый сигнал и выполните команду ):



- Введите тест, соответствующий тесту, приведенному на рисунке ниже



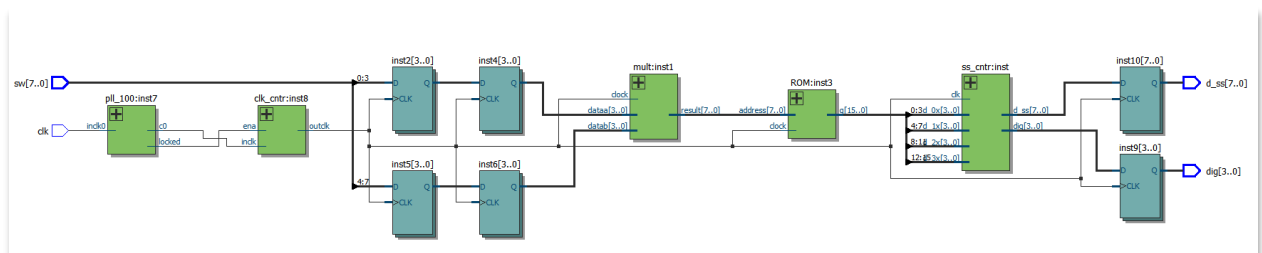
- Сохраните тест: File=>Save (или Save as) под именем **lab5.vwf**.
- Запустите функциональное моделирование: Simulation => Run Functional Simulation.
Временная диаграмма с результатами моделирования откроется в отдельном окне
- Результаты моделирования должны соответствовать приведенному ниже рисунку.



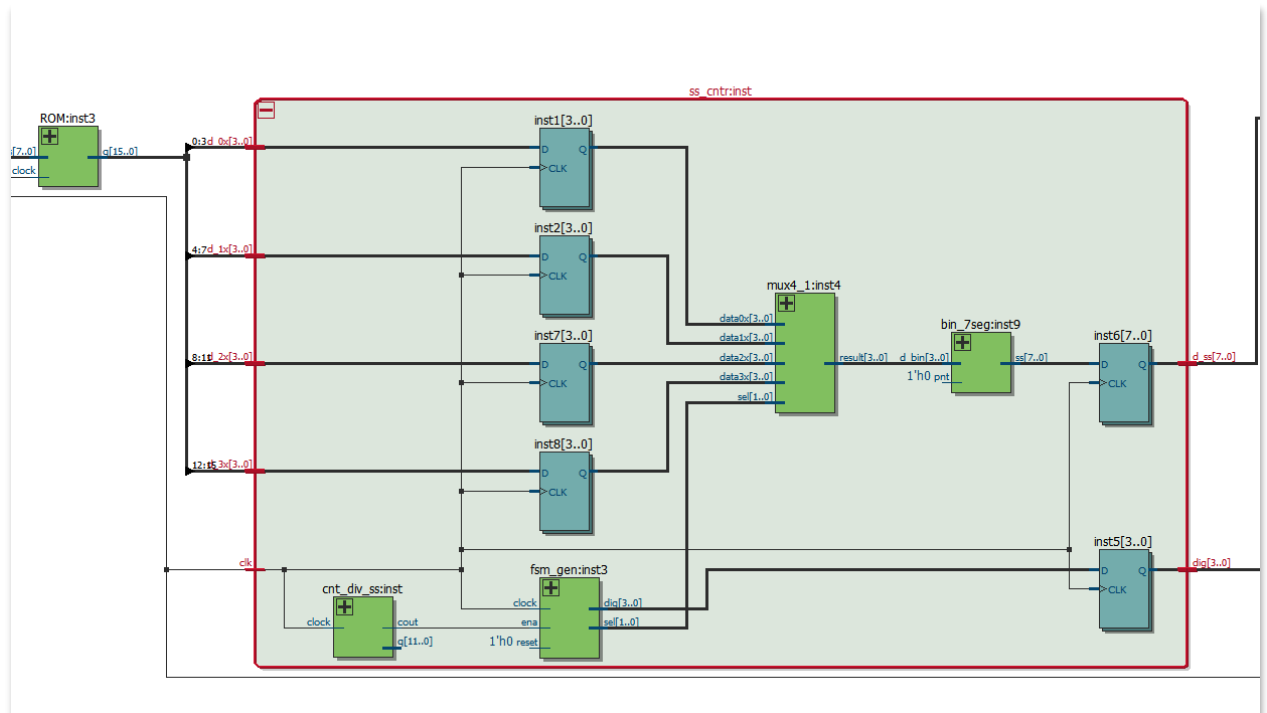
- Объясните полученные результаты моделирования.
Моделирование компонента закончено.

Часть 6 – Анализ проекта с помощью RTL Viewer

- Откройте RTL Viewer: Tools=>Netlist Viewers=>RTL Viewer



2. «Откройте» содержимое модуля `ss_contr`



3. Используя возможности перекрестных ссылок между приложениями пакета, сопоставьте модуль bin_7seg в RTL Viewer исходной схеме:
 - Выберите модуль, нажмите правую клавишу мыши, и выполните команду Locate=>Locate in Design File
4. Повторите процедуру сопоставления (используя возможности перекрестных ссылок) для других модулей (например: pll_100, входа clk ...)

Часть 7 – Назначение контактов СБИС

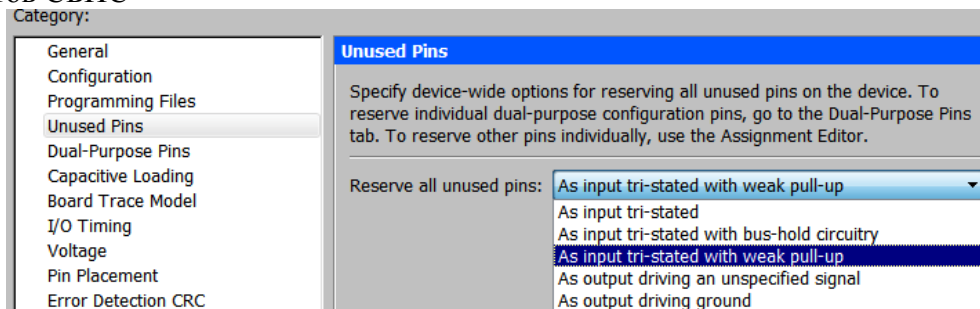
1. Назначьте контакты СБИС выводам проекта и используемый стандарт сигналов в соответствии с приведенной ниже таблицей

Node Name	Direction	Location	I/O Bank	VREF Group	I/O Standard	Current Strength	Slew Rate
clk	Input	PIN_23	1	B1_N0	3.3-V LVCMOS	2mA (default)	
d_ss[7]	Output	PIN_75	5	B5_N0	3.3-V LVCMOS	2mA (default)	2 (default)
d_ss[6]	Output	PIN_84	5	B5_N0	3.3-V LVCMOS	2mA (default)	2 (default)
d_ss[5]	Output	PIN_76	5	B5_N0	3.3-V LVCMOS	2mA (default)	2 (default)
d_ss[4]	Output	PIN_85	5	B5_N0	3.3-V LVCMOS	2mA (default)	2 (default)
d_ss[3]	Output	PIN_77	5	B5_N0	3.3-V LVCMOS	2mA (default)	2 (default)
d_ss[2]	Output	PIN_86	5	B5_N0	3.3-V LVCMOS	2mA (default)	2 (default)
d_ss[1]	Output	PIN_133	8	B8_N0	3.3-V LVCMOS	2mA (default)	2 (default)
d_ss[0]	Output	PIN_87	5	B5_N0	3.3-V LVCMOS	2mA (default)	2 (default)
diq[3]	Output	PIN_73	5	B5_N0	3.3-V LVCMOS	2mA (default)	2 (default)
diq[2]	Output	PIN_80	5	B5_N0	3.3-V LVCMOS	2mA (default)	2 (default)
diq[1]	Output	PIN_74	5	B5_N0	3.3-V LVCMOS	2mA (default)	2 (default)
diq[0]	Output	PIN_83	5	B5_N0	3.3-V LVCMOS	2mA (default)	2 (default)
sw[7]	Input	PIN_88	5	B5_N0	3.3-V LVTTTL	8mA (default)	
sw[6]	Input	PIN_89	5	B5_N0	3.3-V LVTTTL	8mA (default)	
sw[5]	Input	PIN_90	6	B6_N0	3.3-V LVTTTL	8mA (default)	
sw[4]	Input	PIN_91	6	B6_N0	3.3-V LVTTTL	8mA (default)	
sw[3]	Input	PIN_49	3	B3_N0	3.3-V LVTTTL	8mA (default)	
sw[2]	Input	PIN_46	3	B3_N0	3.3-V LVTTTL	8mA (default)	
sw[1]	Input	PIN_25	2	B2_N0	3.3-V LVTTTL	8mA (default)	
sw[0]	Input	PIN_24	2	B2_N0	3.3-V LVTTTL	8mA (default)	

2. Закройте редактор назначения контактов.

Часть 8 – Задание режима работы неиспользуемых контактов СБИС

1. выберите режим **As input tri-stated with weak pull-up** для всех не использованных контактов СБИС



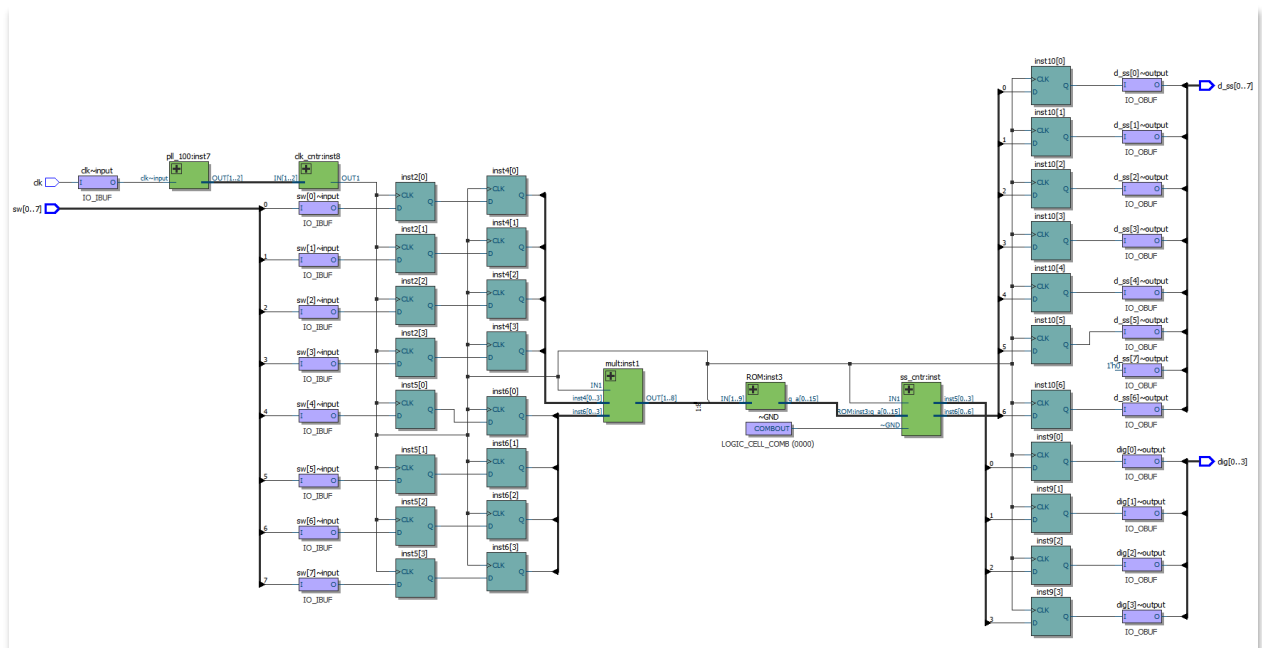
2. Нажмите кнопку OK. В окне Device нажмите кнопку OK еще раз.

Часть 9 – Компиляция проекта

1. В окне задач (Tasks) выберите процедуру Full Design и двойным щелчком левой клавиши мыши по команде Compile Design запустите полную компиляцию проекта.

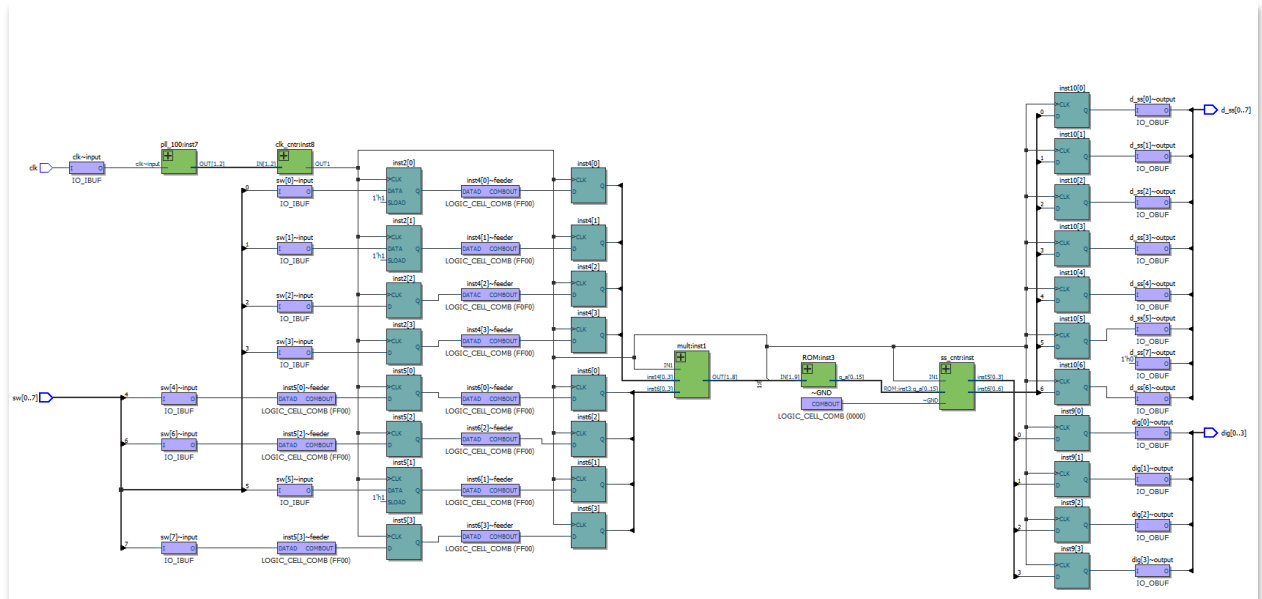
Часть 10 – Анализ проекта с помощью Technology Map Viewer

1. Откройте Technology Map Viewer: Tools=> Netlist Viewers=>Technology Map Viewer (Post-Mapping)



Обратите внимание на то, что появились элементы ввода-вывода IO_IBUF, IO_OBUF, регистры представлены отдельными триггерами ...)

2. Пользуясь возможностями перехода по иерархии и перекрестных ссылок между приложениями, найдите модуль bin_7seg и сопоставьте его с RTL Viewer и исходной схемой.
3. Откройте Technology Map Viewer: Tools=>Netlist Viewers=>Technology Map Viewer (Post-Fitting)

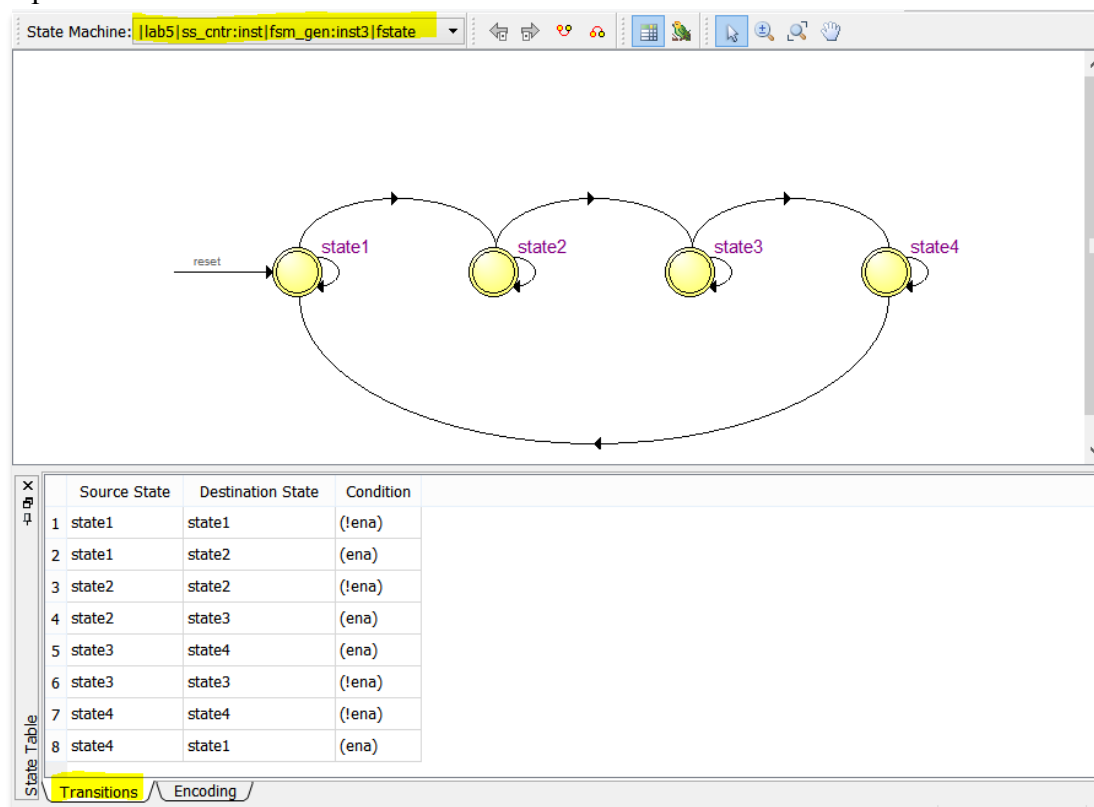


Обратите внимание на то, что появились логические элементы **LOGIC_CELL_COMB**

- Пользуясь возможностями перехода по иерархии и перекрестных ссылок между приложениями, найдите модуль **bin_7seg**, откройте его и посмотрите, как он реализован. Сопоставьте модуль с RTL Viewer и исходной схемой.
- Пользуясь возможностями перехода по иерархии и перекрестных ссылок между приложениями, найдите модули **pll_100** и **clk_cntr**, откройте их и посмотрите, как они реализованы.

Часть 11 – Анализ проекта с помощью State Machine Viewer

- Откройте State Machine Viewer: Tools=>Netlist Viewers=>State Machine Viewer



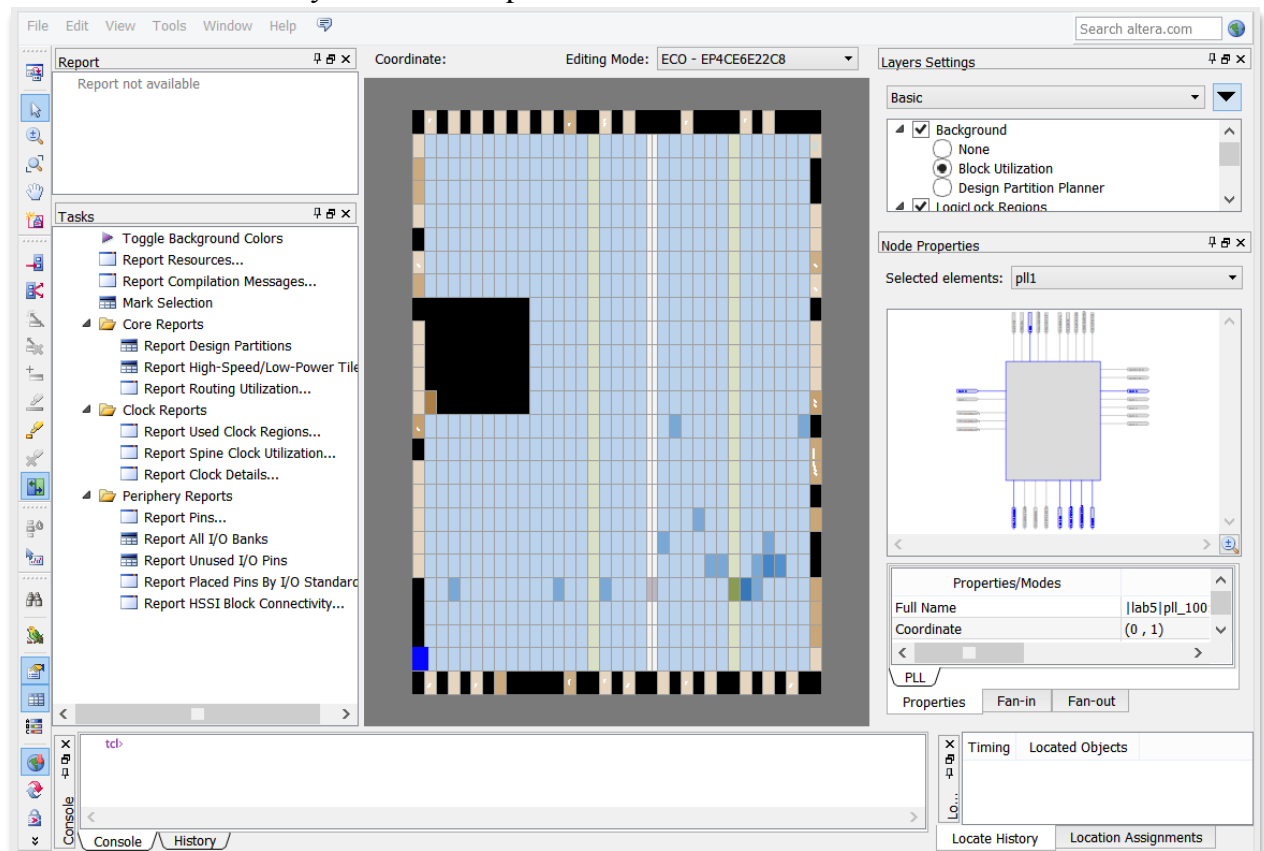
2. Если в проекте есть распознанные компилятором КА (а в данном проекте есть один КА), то они будут отображены (каждый на своей закладке, на которой указан «путь» по иерархии проекта к КА).
 - На закладке Transitions отображается таблица переходов
 - На закладке Encoding отображаются выбранные компилятором коды для состояний автомата

	Name	state4	state3	state2	state1
1	state1	0	0	0	0
2	state2	0	0	1	1
3	state3	0	1	0	1
4	state4	1	0	0	1

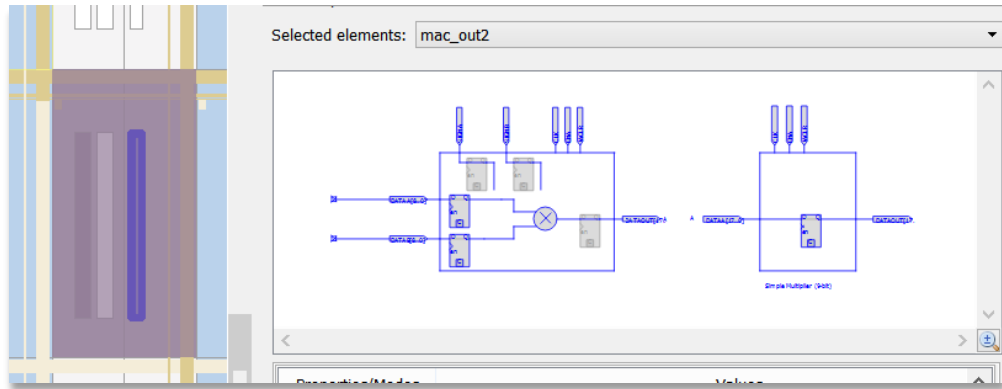
3. Выбрав состояние можно перейти, пользуясь возможностями перекрестных ссылок между приложениями пакета, в исходный файл с описанием КА


Часть 12 – Анализ проекта с помощью Chip Planner

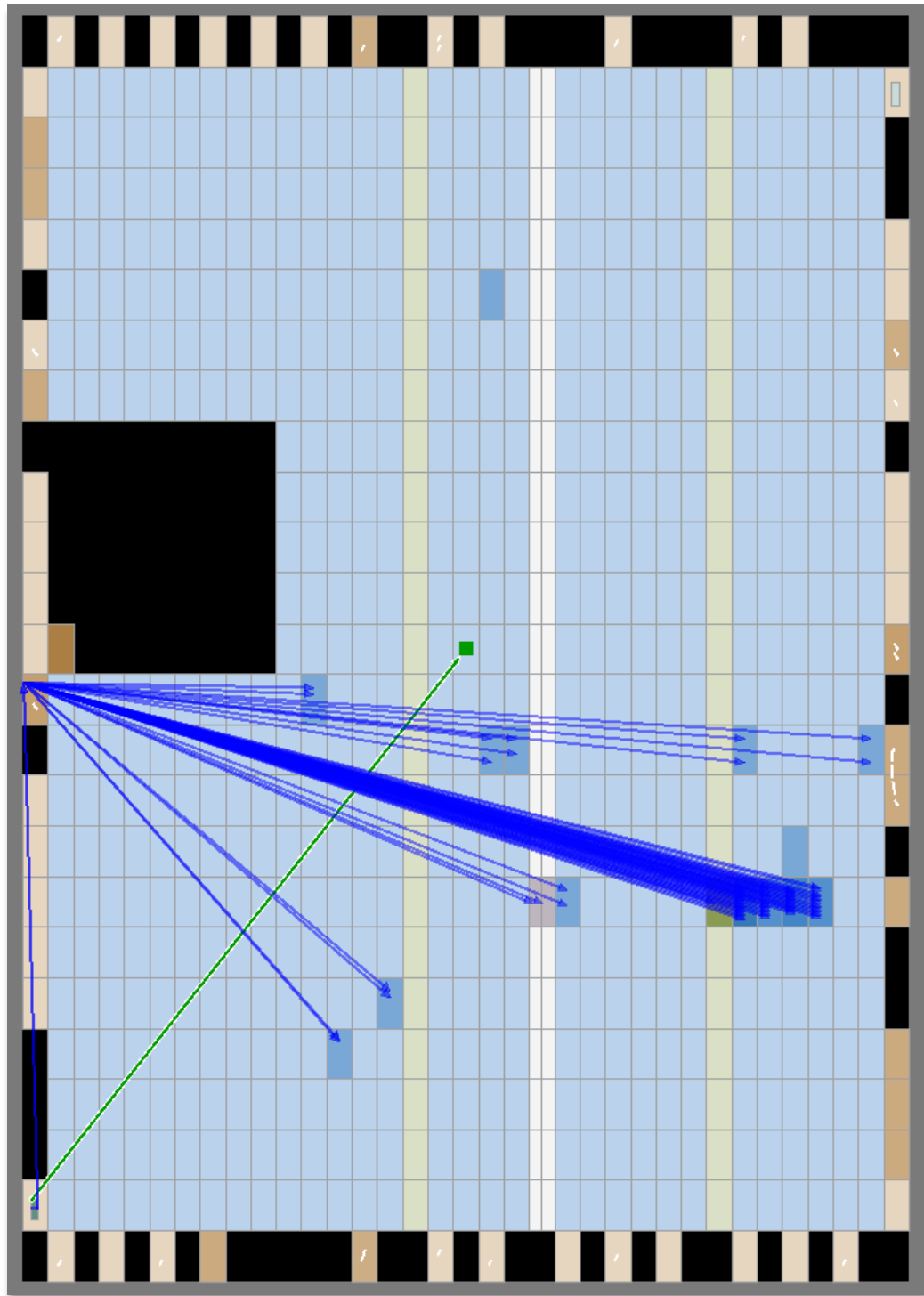
1. Выполните команду: Tools=> Chip Planner



2. Используя Report Resources (раздел Task) найдите и отметьте использованные PLL, DSP Block – умножители, M9K-модули памяти, Clock Control Block – блок управления тактовыми сигналами
3. Выберите использованный модуль умножения (DSP Block), используя Zoom увеличьте его.



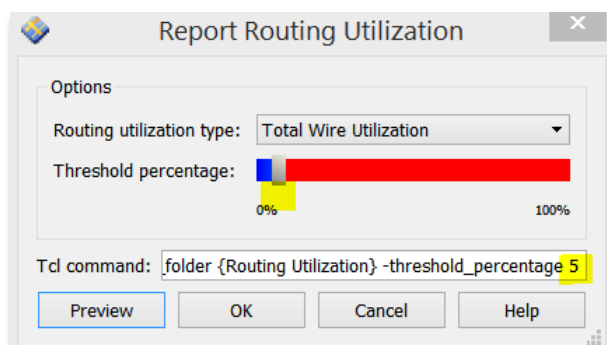
4. В разделе Selected Elements будет отображена структура реализованного модуля умножения (обратите внимание на то, что у модуля имеются регистры на входах данных и выходе данных, что соответствует установленному нами при создании компонента 2-тактному конвейеру)
5. Двойным щелчком по полю Selected Elements будет запущен редактор Resource Property Editor, подробно отображающий особенности реализации модуля. Закройте этот редактор.
6. Выберите используемый модуль PLL и нажмите инструмент  несколько раз:
 - Первый раз: будет показан переход к модулю Clock control
 - Второй раз: будет показан переход внутри модуля Clock Control
 - Третий раз: будут показаны тактовые сигналы, поступающие на все триггеры проекта.



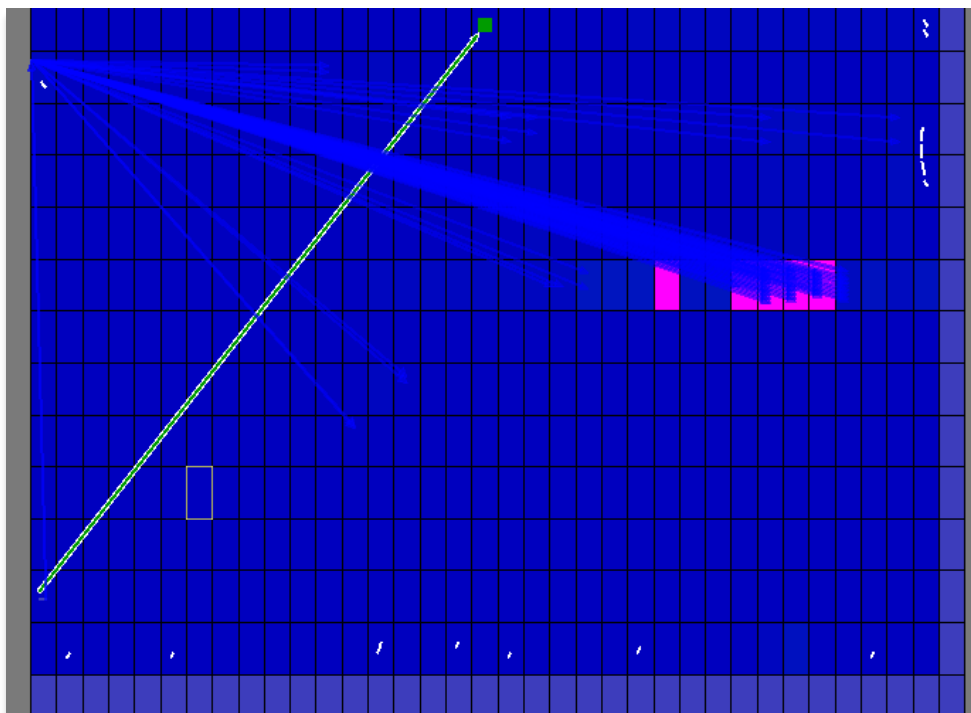
Обратите внимание на то, что тактовые сигналы не поступают в блоки ввода-вывода, следовательно, в этих блоках не используются имеющиеся там триггеры. Это можно исправить с помощью назначений (будет сделано в следующих разделах)

7. Выполните команду Report Routing Utilization

- в появившемся окне нажмите кнопку Preview. Вы увидите, что все поле стало синим (по шкале threshold percentage – означает низкую загрузку каналов связи).
- Движок threshold percentage передвиньте так, чтобы граница была 5%



- Нажмите кнопку Preview – будет отображены зоны СБИС, в которых цепи связи заняты более чем на 5%



8. Закройте редактор Chip Planner

Часть 13 – Назначения

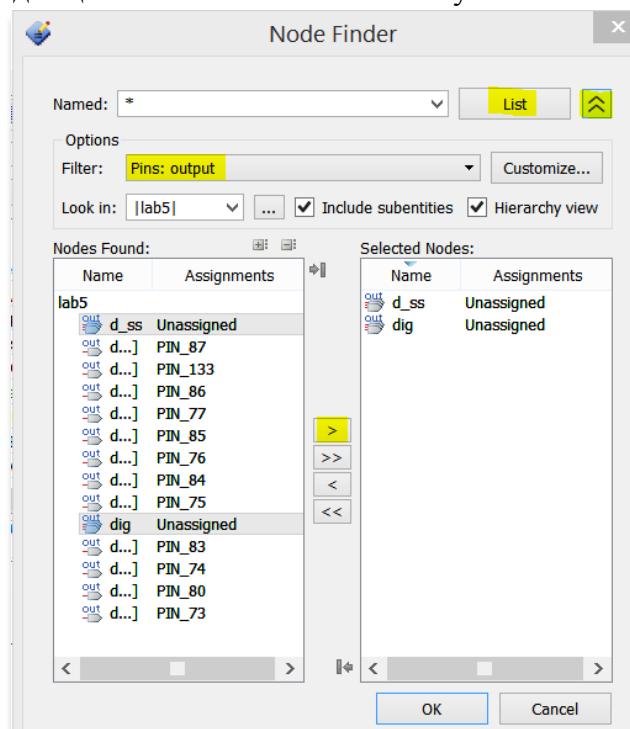
1. Откройте файл lab5.gdf
2. Выберите входы sw[7..0]
3. Выполните команду: Assignment=> Assignment Editor

<<new>> ☒ Filter on node names: sw;sw[7];sw[6];sw[5];sw[4];sw[3];sw[2];sw[1];sw[0]

itatu:	From	To	Assignment Name	Value	Enabled	Entity
1	<input checked="" type="checkbox"/>	in sw[7]	Location	PIN_88	Yes	
2	<input checked="" type="checkbox"/>	in sw[6]	Location	PIN_89	Yes	
3	<input checked="" type="checkbox"/>	in sw[5]	Location	PIN_90	Yes	
4	<input checked="" type="checkbox"/>	in sw[4]	Location	PIN_91	Yes	
5	<input checked="" type="checkbox"/>	in sw[3]	Location	PIN_49	Yes	
6	<input checked="" type="checkbox"/>	in sw[2]	Location	PIN_46	Yes	
7	<input checked="" type="checkbox"/>	in sw[1]	Location	PIN_25	Yes	
8	<input checked="" type="checkbox"/>	in sw[0]	Location	PIN_24	Yes	
9	<input checked="" type="checkbox"/>	in sw[7]	I/O Standard	3.3-V LVCMOS	Yes	lab5
10	<input checked="" type="checkbox"/>	in sw[6]	I/O Standard	3.3-V LVCMOS	Yes	lab5
11	<input checked="" type="checkbox"/>	in sw[5]	I/O Standard	3.3-V LVCMOS	Yes	lab5
12	<input checked="" type="checkbox"/>	in sw[4]	I/O Standard	3.3-V LVCMOS	Yes	lab5
13	<input checked="" type="checkbox"/>	in sw[3]	I/O Standard	3.3-V LVCMOS	Yes	lab5
14	<input checked="" type="checkbox"/>	in sw[2]	I/O Standard	3.3-V LVCMOS	Yes	lab5
15	<input checked="" type="checkbox"/>	in sw[1]	I/O Standard	3.3-V LVCMOS	Yes	lab5
16	<input checked="" type="checkbox"/>	in sw[0]	I/O Standard	3.3-V LVCMOS	Yes	lab5
17	<<new>>	<<new>>				

sw
sw[7]
sw[6]
sw[5]
sw[4]
sw[3]
sw[2]
sw[1]
sw[0]

4. В столбце TO дважды щелкните по полю new и выберите группу sw
5. В столбце Assignment Name задайте опцию Fast Input Register
6. В столбце Value задайте значение ON.
7. В столбце TO дважды щелкните по полю new и запустите Node Finder



8. Выберите выходы (Filters: Pins: output) и нажмите List

9. В полученном списке выделите группы d_ss и dig и скопируйте их в раздел Selected Nodes. Нажмите кнопку ОК.
10. В окне редактора назначений (Assignment Editor) выключите опцию Filter on Node Names

	itatu:	From	To	Assignment Name	Value	Enabled	Entity
1	!		out d_ss			Yes	
2	!		out dig			Yes	
3	✓		in sw	Fast Input Register	On	Yes	lab5
4	✓		in clk	I/O Standard	3.3-V LVCMOS	Yes	lab5
5	✓		out d_ss[0]	I/O Standard	3.3-V LVCMOS	Yes	lab5
6	✓		out d_ss[1]	I/O Standard	3.3-V LVCMOS	Yes	lab5

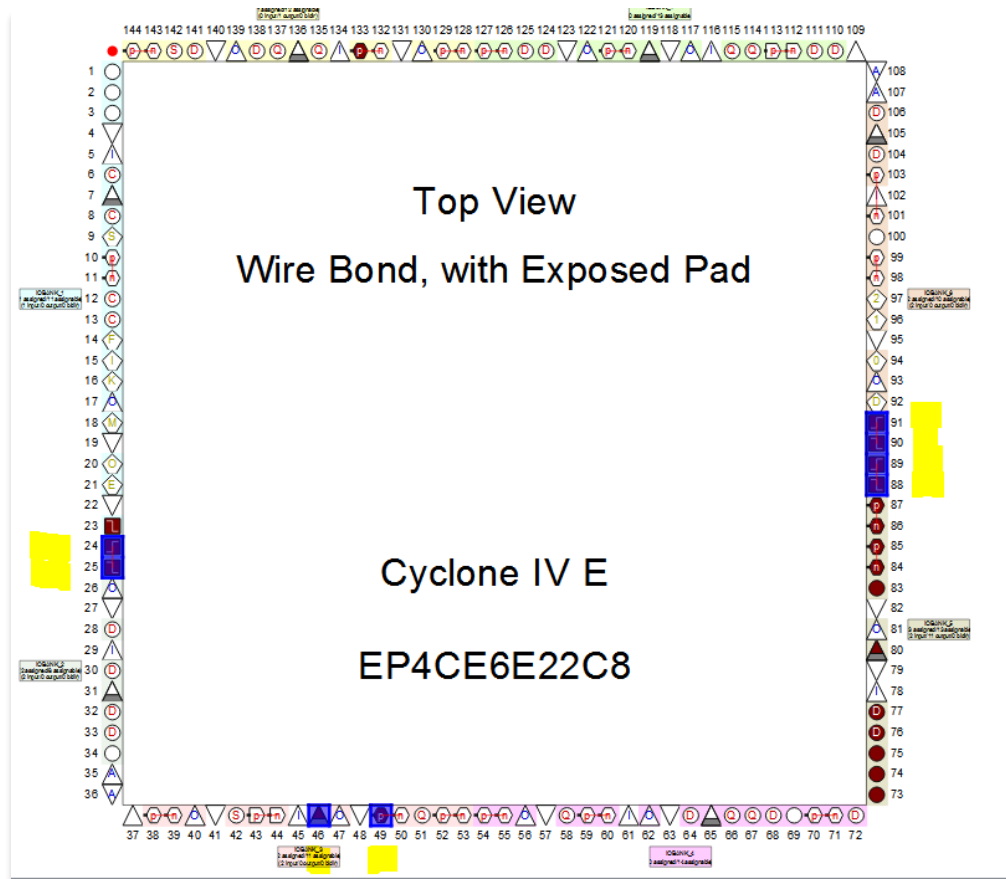
11. Для групп d_ss и dig задайте:
 - В столбце Assignment Name - опцию Fast **Output** Register
 - В столбце Value - значение ON.
12. Сохраните изменения и закройте редактор.

Часть 14 – Компиляция проекта

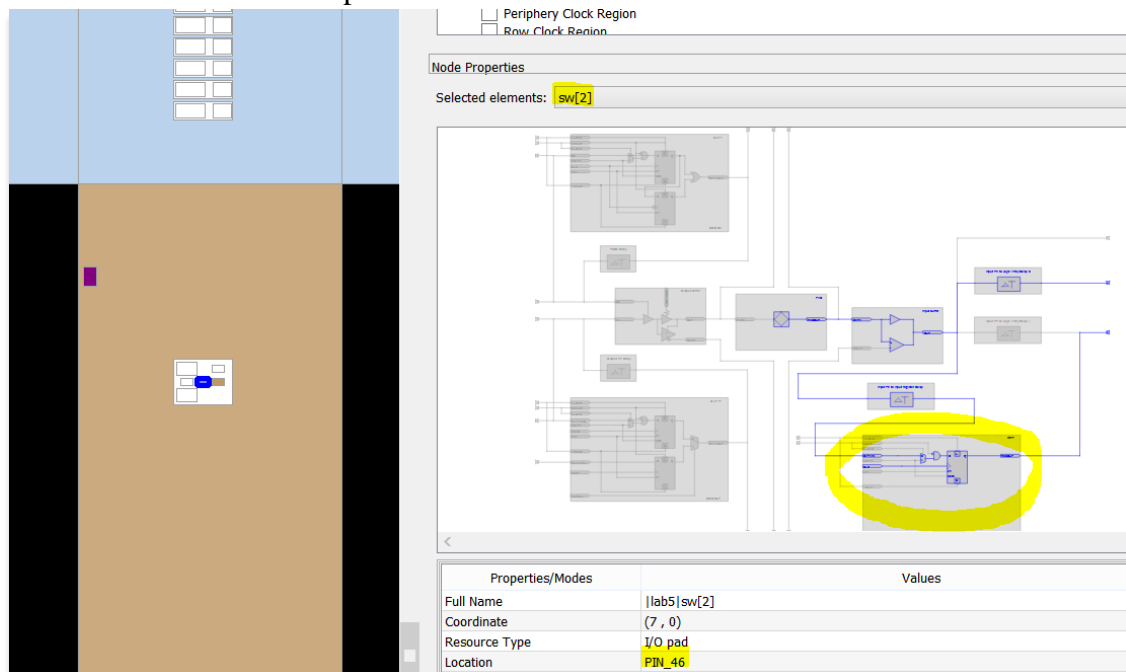
1. В окне задач (Tasks) выберите процедуру Full Design и двойным щелчком левой клавиши мыши по команде Compile Design запустите полную компиляцию проекта.


Часть 15 – Анализ результатов

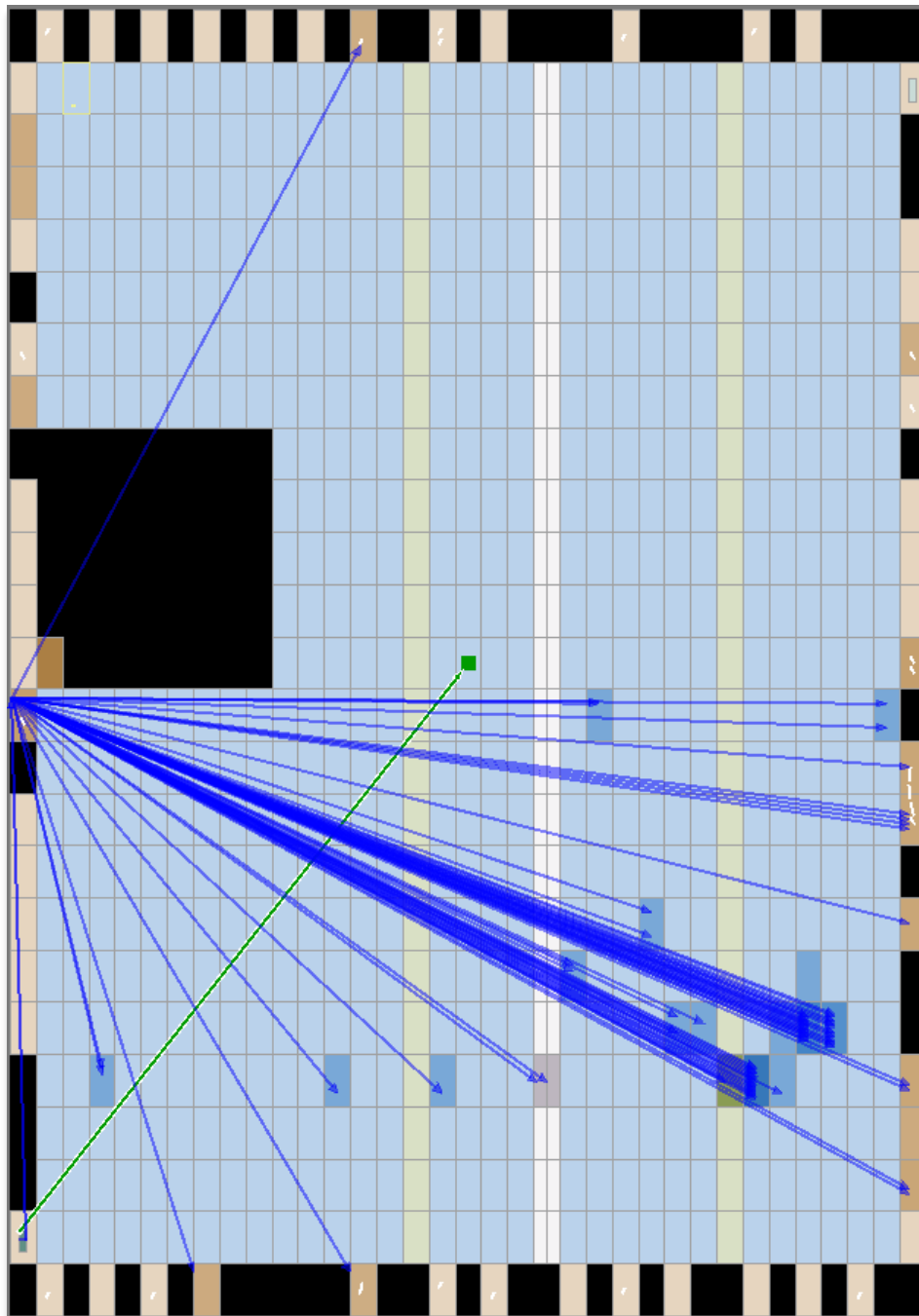
1. Откройте файл lab5.bdf
2. Выберите входы sw[7..0]
3. Нажмите правую клавишу мыши и выполните команду Locate=> Locate in Pin Planer



4. Входы sw[7..0] будут отмечены
5. Выберите вывод 46. Нажмите правую клавишу мыши и выполните команду Locate=>Locate in Chip Planer



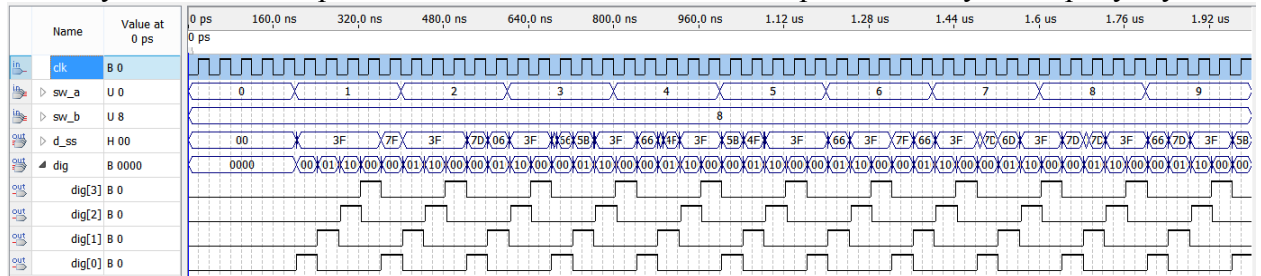
6. Откроется окно редактора Chip Planer, в котором будет выделен выбранный вход, а в окне Selected Element будет отображена структура реализации (обратите внимание, что в элементе ввода-вывода использован триггер)
7. Найдите и выделите использованный PLL, выполните команду  три раза.



8. Теперь тактовый сигнал идет не только к логическим элементам, но и в элементы ввода-вывода (следовательно в элементах ввода-вывода использованы триггеры)
9. *Дополнительное задание:* проверьте был ли использован триггер в элементе ввода-вывода для входов $sw[0]$, $sw[1]$, $sw[7..4]$. Объясните почему.

Часть 16 – Временное моделирование

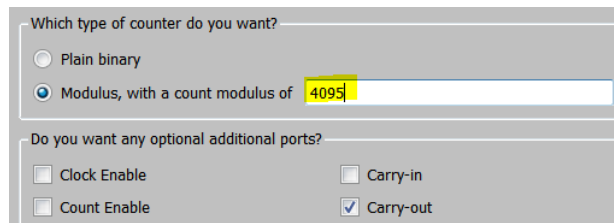
1. Откройте файл lab5.vwf
2. Запустите временное моделирование: Simulation => Run Timing Simulation. Временная диаграмма с результатами моделирования откроется в отдельном окне
3. Результаты моделирования должны соответствовать приведенному ниже рисунку.



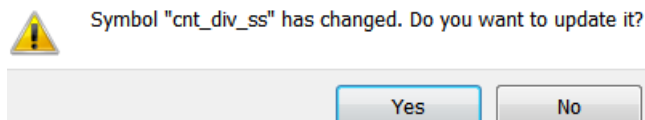
4. Объясните в чем основное отличие полученных результатов от результатов функционального моделирования, полученных ранее.

Часть 17 – Изменение схемы для аппаратной реализации и компиляция проекта

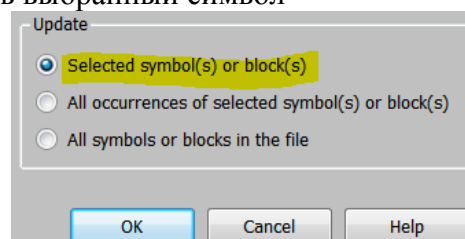
13. Для аппаратной реализации модуля динамической индикации следует изменить модуль счета счетчика cnt_div_ss, обеспечивающего деление входной частоты 25 МГц.
14. В пакете QII откройте схему верхнего уровня иерархии описаний – lab5.gdf, двойным щелчком по компоненту ss_cntr перейдите на нижний уровень иерархии описания – схему компонента.
15. Двойным щелчком в зоне символа счетчика cnt_div_ss запустите помощник MegaWizard Plug-in Manager
16. В окне настроек счетчика перейдите на страницу 2 и задайте модуль счета равным 4095



17. Нажмите кнопку Finish, затем нажмите ее еще раз.
18. В появившемся окне нажмите OK
19. В следующем окне нажмите кнопку YES



20. Далее укажите: обновить выбранный символ

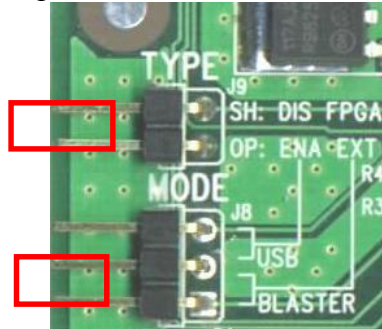


21. Проверьте в схемном редакторе, что символ счетчика обновлен и все выходы подключены правильно.

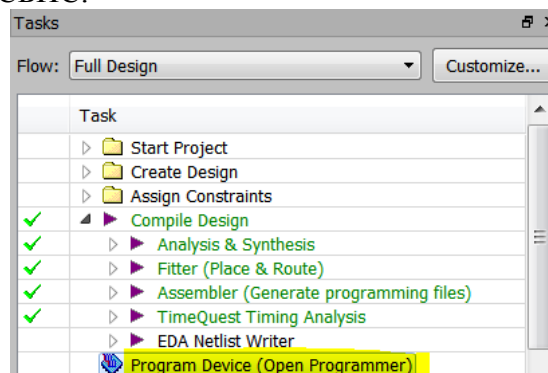
22. Сохраните схему компонента.
23. Осуществите полную компиляцию проекта.

Часть 18 – Конфигурирование СБИС и проверка проекта на плате

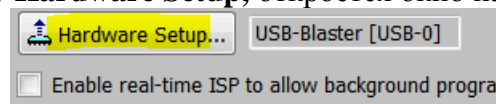
1. На плате miniDiLaB-CIV установите джамперы следующим образом:
 - a. Соедините выводы разъема “TYPE”
 - b. Соедините выводы 1-2 разъема “MODE”



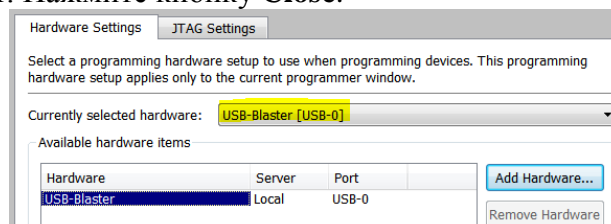
2. Подсоедините входящий в комплект поставки USB кабель A-miniB к USB 2.0 порту компьютера (должен обеспечивать ток до 500мА), а затем к плате miniDiLaB-CIV.
3. Включите плату miniDiLaB-CIV : переключатель Power
4. В окне задач (Tasks) выберите процедуру Full Design и двойным щелчком левой клавиши мыши по команде Program Device запустите приложение, управляющее конфигурированием СБИС.



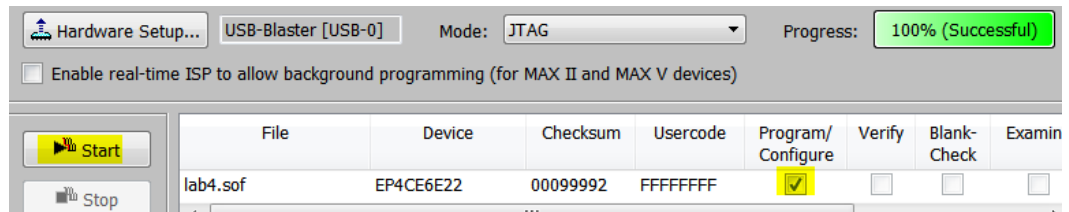
5. Откроется окно управления конфигурированием СБИС.
6. Для установки интегрированного на плату miniDiLaB-CIV средства конфигурирования СБИС нажмите кнопку **Hardware Setup...**, откроется окно настроек.



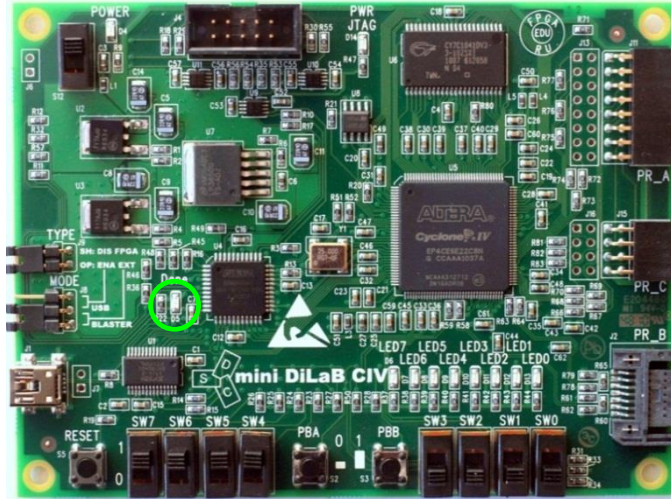
7. В разделе **Available hardware items** выберите (двойным щелчком левой клавиши мыши) USB-Blaster. Нажмите кнопку **Close**.



8. Включите опцию **Program/Configure** и нажмите кнопку **Start**. В окне Progress будет отображаться статус процедуры конфигурирования СБИС.



9. Когда СБИС будет запрограммирована на плате miniDiLaB-CIV загорится зеленый светодиод – “Done”.



10. Проверьте работу проекта:
- Задайте число А и число В
 - На 7-сегментном индикаторе будет отображен результат умножения этих чисел.

Лабораторная работа завершена.