**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Базовая кафедра «Вычислительные технологии»**

отчет

**по лабораторной работе №3**

**по дисциплине «Проектирование реконфигурируемых систем  
на кристалле»**

Тема: Сравнение характеристик проектов на основе Nios II в кристалле Cyclone V при их аппаратной и программно-аппаратной реализации

Вариант 4

|  |  |  |
| --- | --- | --- |
| Студенты гр. 6309 |  | Васин А. М. |
|  |  | Жвакин К. Э. |
|  |  | Ладыженский Р. С. |
| Преподаватель |  | Шарагина Н.С. |

Санкт-Петербург

2021

**Цель работы.**

Цель работы состоит в анализе особенностей проектирования и отладки и сравнении эффективности проектирования систем на базе soft-процессора Nios II и проектирования беспроцессорных аппаратных модулей при их реализации в условиях одной СнК.

**Основные теоретические положения.**

Один из способов оценки корректности проектов основан на моделировании, предполагающем отладку проекта на модели путем анализа реакций разрабатываемых схем на стимулирующие воздействия. Несмотря на высокую вероятность обнаружения имеющихся дефектов моделирование не всегда позволяет оценить работу схемы. Более эффективны методы, основанные на экспериментах с реальным оборудованием. Получившие в последнее время широкое распространение прототипные платы разработчика, содержащие ПЛИС, позволяют организовать подобные эксперименты.

Одно из основных достоинств СнК – гибкость решения различных задач. На одном и том же кристалле ПЛИС можно реализовать различные способы решения поставленной задачи, оценить результат и выбрать более подходящий в данном случае способ.

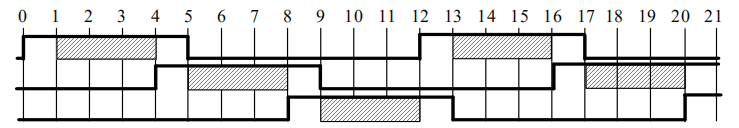


Рис. 1 - Диаграмма сигналов трехфазного управления

В работе предлагается реализовать два независимых решения задачи формирования сигналов трехфазного управления, аппаратно выполненные на одном кристалле ПЛИС фирмы «Altera».

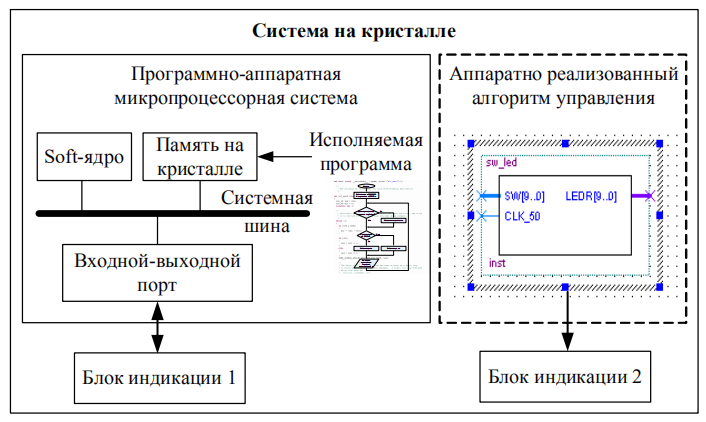


Рис. 2 - Устройство трехфазного управления

Первая система предполагает чисто аппаратную реализацию алгоритма, а вторая должна быть выполнена как программно-аппаратная микропроцессорная система на основе процессорного ядра Nios II. Пример временной диаграммы трехфазного управления с перекрытием на один такт приведен на рис. 1, а структурная схема устройства – на рис. 2.

**Задание на работу.**

Разработать СнК с двумя независимыми решениями задачи трехфазного управления: аппаратным и программно-аппаратным. Выполнить имплементацию проекта СнК в ПЛИС отладочной платы DE0 фирмы «Terasic». Визуальный контроль работоспособности проекта осуществить путем подключения светодиодных индикаторов.

**Выполнение работы.**

***Этап 1. Создание проекта в САПР Quartus II***

1. Выполним конфигурирование процессорной части СнК, а также сгенерируем файлы описания soft-процессора NIOS II со встроенной в кристалл памятью программ и данных и блоком индикации. Выполним компиляцию проекта, назначим контакты, выполним загрузку процессорного ядра в ПЛИС.

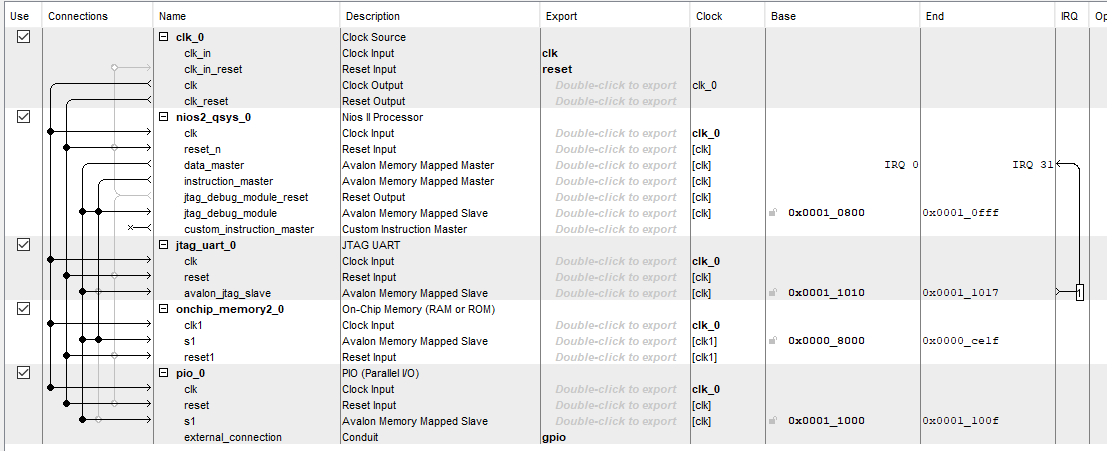


Рис. 3 - Результат конфигурирования системы

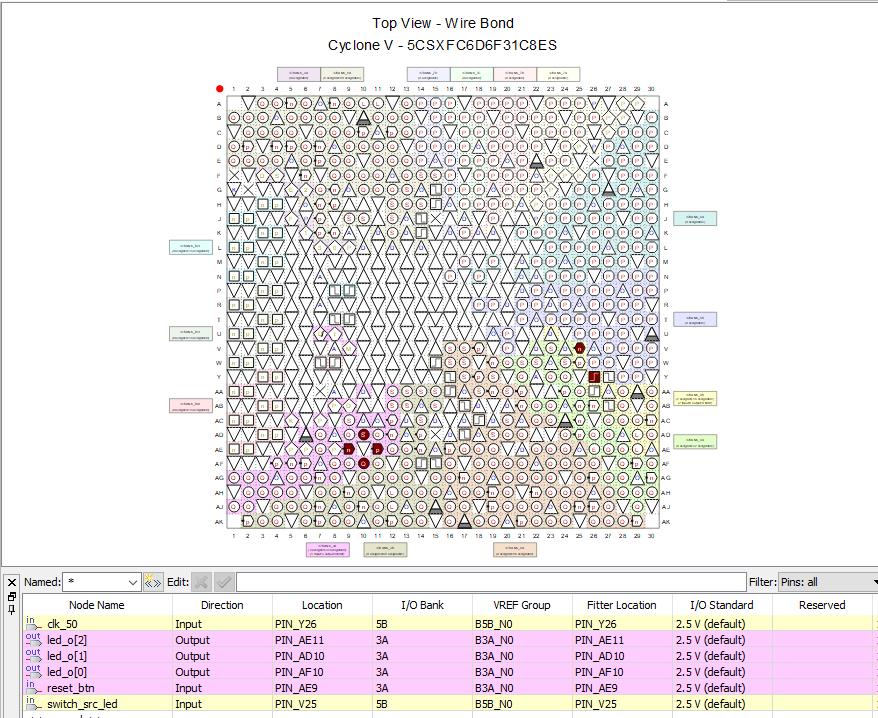


Рис. 4 - Результат назначения контактов

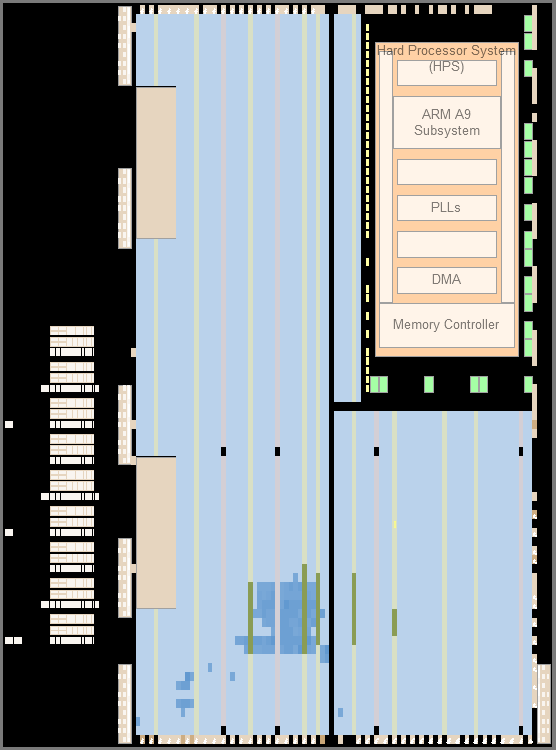


Рис. 5 - Расположение проекта в ПЛИС

1. Загрузим на выполнение программу трехфазного управления для Nios II. Листинг программы, соответствующий временной диаграмме на рис. 1, приведен в листинге 1. Работа программы построена на основе циклического формирования выходных сигналов. led[0]…led[2] – выходные значения сигналов управления для каждой фазы, Counter – счётчик тактов для фаз.

Листинг 1. Исходный код программы.

#include "sys/alt\_stdio.h"

#include "system.h"

#include "altera\_avalon\_pio\_regs.h"

#include <unistd.h>

#define FIRST\_UP 24

#define FIRST\_DOWN 10

#define SECOND\_UP 8

#define SECOND\_DOWN 18

#define THIRD\_UP 16

#define THIRD\_DOWN 2

int main**()**

**{**

alt\_putstr**(**"Hello from Nios II!\n"**);**

int data\_led **=** 0x00**;**

alt\_u8 led **[**3**]** **=** **{**0**,** **};**

int counter **=** 0**;**

/\* Event loop never exits. \*/

**while** **(**1**)**

**{**

counter**++;**

**switch(**counter**)**

**{**

**case** FIRST\_UP**:** led**[**0**]** **=** 1**;** counter **=** 0**;** **break;**

**case** FIRST\_DOWN**:** led**[**0**]** **=** 0**;** **break;**

**case** SECOND\_UP**:** led**[**1**]** **=** 1**;** **break;**

**case** SECOND\_DOWN**:** led**[**1**]** **=** 0**;** **break;**

**case** THIRD\_UP**:** led**[**2**]** **=** 1**;** **break;**

**case** THIRD\_DOWN**:** led**[**2**]** **=** 0**;** **break;**

**default:** **break;**

**}**

data\_led **=** led**[**2**]** **<<** 2 **|** led**[**1**]** **<<** 1 **|** led**[**0**];**

IOWR\_ALTERA\_AVALON\_PIO\_DATA**(**PIO\_0\_BASE**,** data\_led**);**

usleep**(**50000**);**

**}**

**return** 0**;**

**}**

1. Оценка временных и аппаратных затрат на реализацию проекта представлена на рисунках ниже.

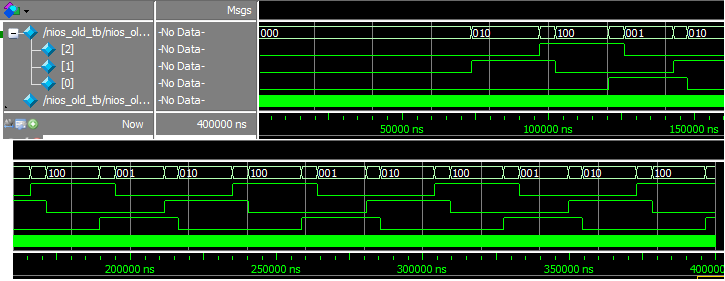


Рис. 6 - Результат моделирование проекта во временной области

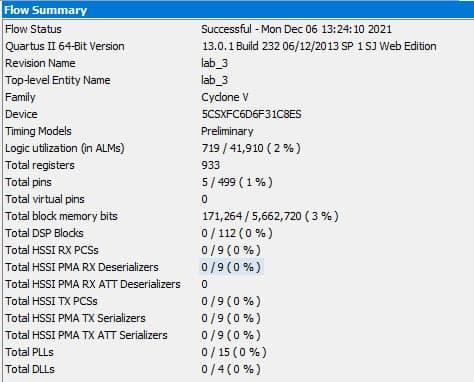


Рис. 7 - Оценка затрат на реализацию проекта



Рис. 8 - RTL - представление проекта

***Этап 2. Аппаратная реализация алгоритма***

В листинге 2 приведёно описание разработанного устройства, реализующего заданный алгоритм управления.

В данном проекте индикация светодиодов осуществляется от выбираемого источника. Если переменная switch\_src\_led равна единице, то светодиоды будут отображать результат работы soft-процессора Nios II, в противном случае светодиоды будут отображать результат реализованного на этом этапе аппаратного решения.

`define FIRST\_UP 5'd24

`define FIRST\_DOWN 5'd10

`define SECOND\_UP 5'd8

`define SECOND\_DOWN 5'd18

`define THIRD\_UP 5'd16

`define THIRD\_DOWN 5'd2

**module** lab\_3 **(**

**input** **logic** clk\_50**,**

**input** **logic** reset\_btn**,**

**input** **logic** switch\_src\_led**,**

**output** **[**2**:**0**]** led\_o

**);**

**logic** **[**2**:**0**]** led\_nios**;**

**logic** **[**2**:**0**]** led\_fpga**;**

**assign** led\_o **=** switch\_src\_led **?** led\_nios **:** led\_fpga**;**

nios\_old nios\_instance **(**

**.**clk\_clk **(**clk\_50**),** // clk.clk

**.**gpio\_export **(**led\_nios**),** // gpio.export

**.**reset\_reset\_n **(**reset\_btn**)** // reset.reset\_n

**);**

**logic** **[**4**:**0**]** counter**;**

**logic** clk\_10Hz**;**

generator **#(**500\_0000**)** //for 10Hz is 500\_0000

generator\_100Hz**(**

**.**nreset\_i **(**reset\_btn**),**

**.**i\_clk **(**clk\_50**),**

**.**o\_clk **(**clk\_10Hz**)**

**);**

**always** **@(posedge** clk\_10Hz**,** **negedge** reset\_btn**)**

**begin**

**if(!**reset\_btn**)**

**begin**

led\_fpga **=** 3'b000**;**

counter **=** 4'd0**;**

**end**

**else**

**begin**

counter**++;**

**case** **(**counter**)**

`FIRST\_UP**:begin** led\_fpga**[**0**]** **=** 1'b1**;** counter **=** 5'd0**;** **end**

`FIRST\_DOWN**:** led\_fpga**[**0**]** **=** 1'b0**;**

`SECOND\_UP**:** led\_fpga**[**1**]** **=** 1'b1**;**

`SECOND\_DOWN**:** led\_fpga**[**1**]** **=** 1'b0**;**

`THIRD\_DOWN**:** led\_fpga**[**2**]** **=** 1'b0**;**

`THIRD\_UP**:** led\_fpga**[**2**]** **=** 1'b1**;**

**default:** **;**//begin led\_fpga[0] = led\_fpga0]; led\_fpga[1] = led\_fpga[1]; led\_fpga[2] = led\_fpga[2]; end

**endcase**

**end**

**end**

**endmodule**

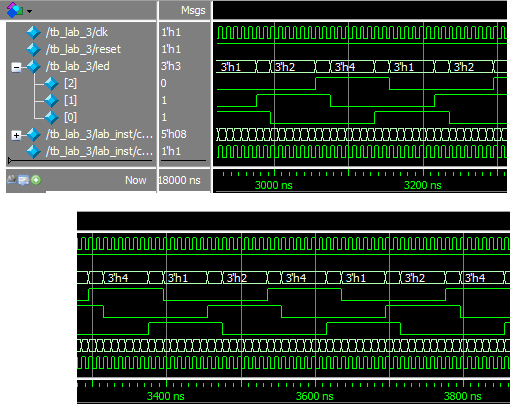


Рис. 9 - Результат моделирования проекта во временной области

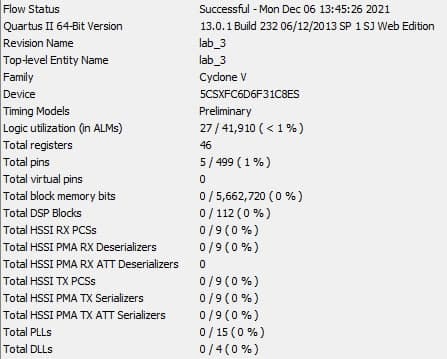
****

Рис. 10 - Оценка затрат на компиляцию проекта

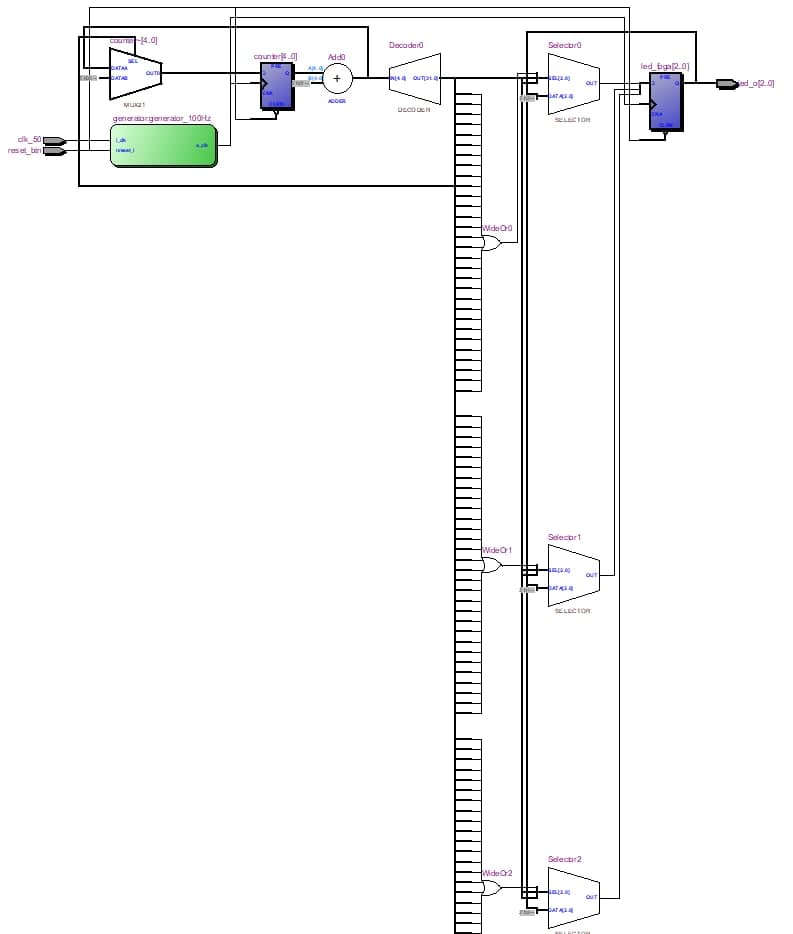


Рис. 11 - RTL-представление проекта

**Выводы.**

В ходе выполнения данной лабораторной работы был произведён анализ особенностей проектирования и отладки систем на базе soft-процессора Nios II и беспроцессорного аппаратного модуля при их реализации в условиях одной СнК.

Сравнения проводились с отключенными задержками на аппаратном входном тактовом сигнале 50МГц, минимальные периоды составили 69000 нс и 241 нс соответственно. Таким образом, разница составила 286 раз по скорости работы в пользу аппаратной реализации.

Далее, сравнивая затраты на реализацию проекта, можно отметить, что применение аппаратных модулей позволяет уменьшить использование логики в 26 раз, сократить количество используемых регистров в 20 раз, а также избавиться от использования блочной памяти ПЛИС (в которой хранится исполняемый код программы). Всё это обусловленно отсутсвием необходимости синтеза процессорного ядра для выполнения одной и той же задачи.

Подводя итоги, можно отметить эффективность применения аппаратных модулей для выполнения задач, требующих большого быстродействия, в то время как применение soft-процессоров позволяет снизить временные затраты на выполнение проекта.