**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Базовая кафедра «Вычислительные технологии»**

отчет

**по лабораторной работе №2**

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

Тема: Разработка проектов с использованием soft-ядра процессора NIOS II

Вариант 5

|  |  |  |
| --- | --- | --- |
| 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 освоении конфигурирования системы на кристалле (СнК) на базе процессора Nios II с использованием среды SOPC Builder и получения навыков разработки программного обеспечения в среде Nios II IDE.

**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.

Для получения навыков работы с реальной ПЛИС предлагается провести модельную отладку проекта, представленного исходным описанием на языке VHDL, имплементацию проекта в ПЛИС учебного стенда и анализ работоспособности полученной схемы.

Отладка созданного проекта будет состоять из двух этапов: моделирования проекта и отладки на реальной схеме.

Отладка проекта на реальной схеме будет производиться с помощью встроенного в пакет Quartus II логического анализатора Signal Tap II Logic Analyzer. Использование встроенного логического анализатора – это один из способов внутрикристальной отладки проекта, возможный при наличии неиспользованных ресурсов ПЛИС. Созданные в процессе проектирования логические анализаторы загружаются в ПЛИС и подсоединяются к интересующим разработчика цепям; таким образом, разработчик имеет возможность наблюдать за реальными состояниями различных сигналов, фиксируемых логическим анализатором. Signal Tap II позволяет создавать и встраивать в проект определенное число логических анализаторов, оперативно изменять условия фиксации данных в их памяти и отображать эти данные на экране компьютера.

**Задание на работу.**

Разработать программно-аппаратную систему, состоящую из ядра процессора Nios II, памяти и блока ввода-вывода и реализующую функцию в соответствии с индивидуальным заданием.

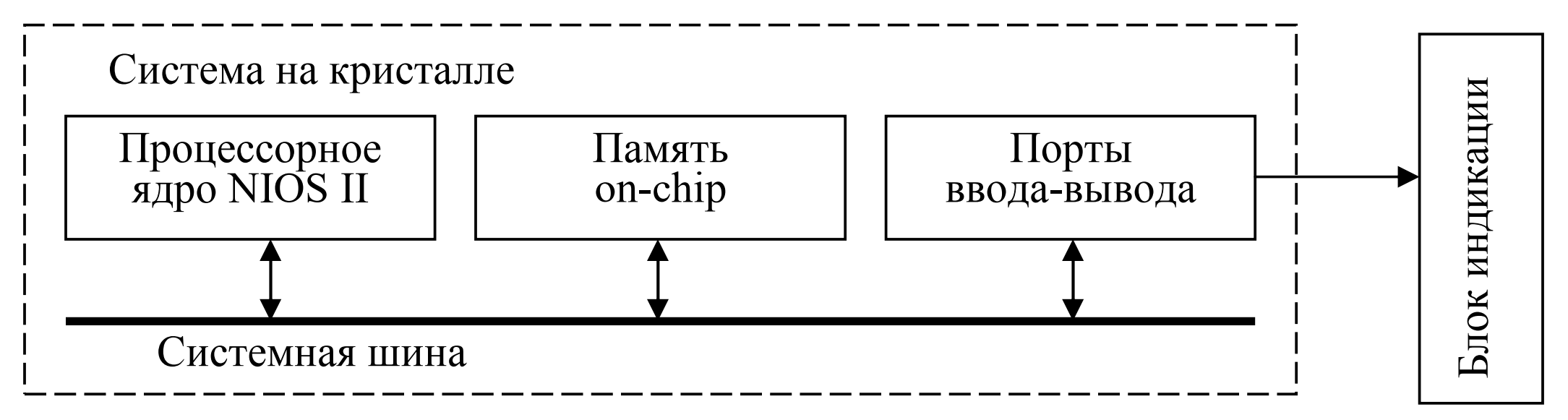


Рис. 1 Структура разрабатываемой системы на кристалле

Выполнить имплементацию проекта СнК в ПЛИС отладочной платы DE0 фирмы «Terasic». Для отображения результатов работы системы использовать светодиодные индикаторы, расположенные на плате.

Релизовать систему, управляющую перемещением светящегося сегмента по четырём семисегментным индикаторам.

**Выполнение работы.**

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

При создании проекта использовалась последовательность действий, приведенная в пп. 1-3 этапа 1 лаб. раб. 1.

***Этап 2. Конфигурирование аппаратных модулей СнК***

1.1. В пакете конфигурации NIOS II - SOPC Builder, были указаны параметры тактового сигнала (50 МГц), а также сконфигурировано оптимизированное по площади, но ограниченное по функциональным возможностям soft-ядро процессора - Nios II/e (Economy).

1.2. Далее была сконфигурирована RAM-память.

1.3. На этом шаге были определены такие параметры порта ввода-вывода, как разрядность (4 бита), направление (output), значения после сброса и возможность побитового обращения к выходному регистру, через который будут выводиться данные на светоизлучающие диоды.

1.4. Определить форму взаимодействия этих компонентов в системе: назначить базовый адрес для каждого ведомого компонента и приоритеты прерываний.

1.5. Были определены адреса векторов прерываний и исключений (см. Рисунок).

1.6. Сгенерировать код Nios II нажатием кнопки Generate. После генерации в папке проекта появилась директория «sopc\_builder» с автоматически сгенерированным HDL-кодом (RTL-описание проекта Nios II).

***Этап 3. Конфигурирование системы на кристалле ПЛИС***

На рисунке виден результат конфигурирования системы, а также адресную карту, состоящую из диапазона адресов ядра процессора, jtag\_uart интерфейса, ROM-памяти и портов ввода/вывода.



Учебная плата была подключена к компьютеру через программатор и проект был загружен в ПЛИС.

***Этап 4. Разработка программного обеспечения***

Программа управляет выводом сигнала на блок светодиодной индикации учебного стенда, создавая бегущий сигнал горящего индикатора, справа налево, затем наоборот. Это реализуется выведением на внешние выводы СнК двоичного слова, содержащего одну единицу, и реализацией логического сдвига этого слова с задержкой.

#include "sys/alt\_stdio.h"

#include "system.h"

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

#include <unistd.h>

int main**()**

**{**

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

int data\_led **=** 0x01**;**

/\* Event loop never exits. \*/

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

**{**

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

usleep**(**500000**);**

data\_led **=** data\_led **<<** 1**;**

**if(**data\_led **==** 0x10**)**

data\_led **=** 0x01**;**

**}**

**return** 0**;**

**}**

***Этап 5. Определения интервала дребезга контактов***

Для определения интервала дребезга был запущен логический анализатор SignalTap II. Затем был выбран сигнал тактирования анализатора clk (10 МГц), создан список сигналов, необходимых для отладки схемы, а также выбрано количество записываемых сэмплов логическим анализатором (64к):

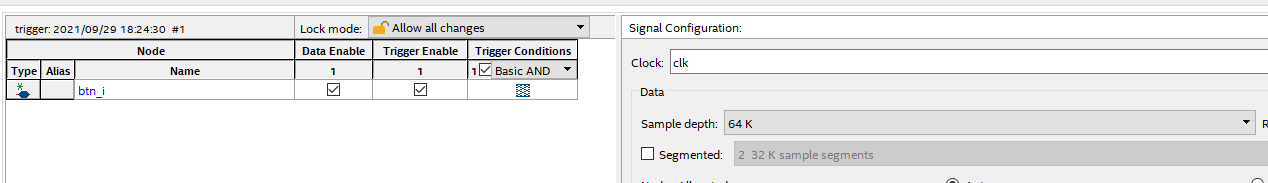


Рисунок 6 – Задание списка сигналов и определение сигнала тактирования в Signal Tap II

Проект снова был скомпилирован и загружен в ПЛИС с помощью SignalTap II Logic Analyser. Было задано условие срабатывания логического анализатора в правой области окна настройки:

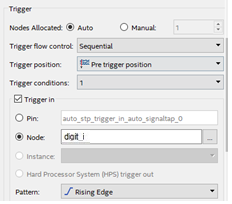


Рисунок 7 – Задание условия срабатывания логического анализатора

Логический анализатор был запущен в работу, и была получена следующая временная диаграмма:

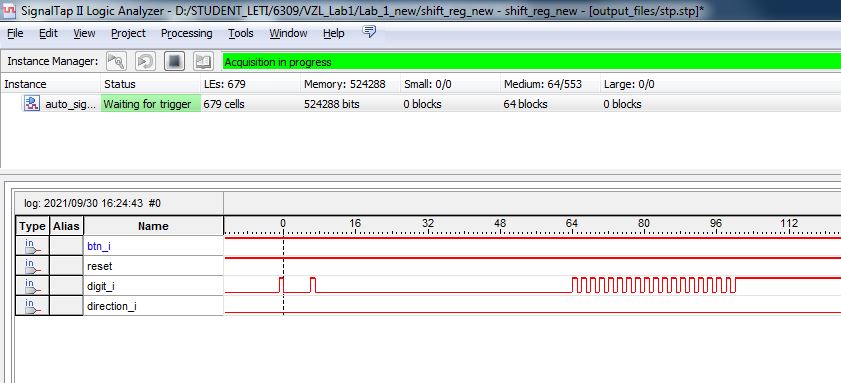


Рисунок 8 – Временная диаграмма зафиксированных сигналов

В результате работы логического анализатора мы можем наблюдать дребезг контактов на входном сигнале, который подключен к выводу с тактовой кнопки.

Было выполнено 4 измерения времени дребезга.

Таблица 1. Измерение времени дребезга контактов.

|  |  |  |
| --- | --- | --- |
| N | Временная диаграмма | Время дребезга, тактов clk |
| 1 |  | 1281 |
| 2 |  | 446 |
| 3 |  | 840 |

Максимальное время дребезга составляет 1300 тактов (округляя) сигнала clk. Исходя из того, что это 1300 тактов генератора 10МГц, можно сказать, что максимальное время дребезга составляет 1300\*5 = 6500 тактов основного генератора (50МГц).

***Этап 6. Устранение дребезга контактов***

Для устранения влияния дребезга в основной проект был создан дополнительный модуль, в котором происходит устранение дребезга.

Листинг 2. Исходный код схемы устранения дребезга.

`define LEVEL\_COUNT 32'd10000

module bouncing (

input clk,

input signal\_i,

output logic signal\_o

);

logic prev\_signal;

logic signal;

assign signal = signal\_i;

int counter;

always @(posedge clk)

begin

if (prev\_signal == signal)

begin

counter++;

end

else

begin

prev\_signal = signal;

counter = 0;

end

if(counter == `LEVEL\_COUNT)

begin

counter = 0;

signal\_o = signal;

end

end

endmodule

В исходном коде листинга 2 видно, что при изменении входного сигнала происходит запуск счетчика и если счетчик досчитал до числа, определенного дефайном, то это значит, что входной сигнал устоялся и выходной становится таким же как входной.

Тестбенч для моделирования был доработан с учетом дребезга входного сигнала. Также для моделирования было изменено число счетчика дребезга для удобства моделирования. Результаты повторного моделирование представлены ниже, на рисунке 9.

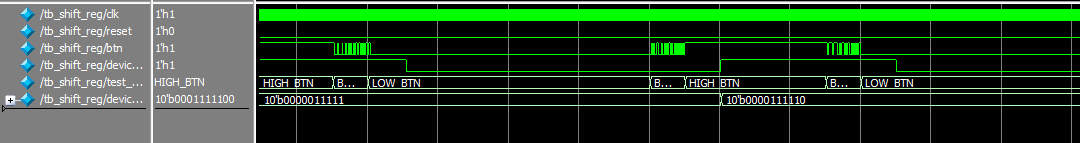


Рисунок 9 – Повторное моделирование с учетом дребезга

Как видно из повторного моделирования: схема устранения дребезга работает успешно и ложных срабатываний не происходит.

Проект был скомпилирован и загружен в ПЛИС. В ходе макетирования явления дребезга не наблюдалось: каждому нажатию кнопки соответствовал сдвиг числа на 1 бит.

**Выводы.**

В ходе выполнения данной лабораторной работы были получены навыки создания проекта, его моделирования, имплементации в программируемую логическую интегральную схему (ПЛИС) с последующей внутрикристальной отладкой проекта с использованием встроенного в САПР логического анализатора Signal Tap Logic Analyzer.