МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«НОВОСИБИРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ» (НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ, НГУ)

Факультет **ФИЗИЧЕСКИЙ**

Кафедра **ФИЗИКО-ТЕХНИЧЕСКОЙ ИНФОРМАТИКИ**

Направление подготовки **03.03.02 ФИЗИКА**

Образовательная программа **БАКАЛАВРИАТ**

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

Андреев Андрей Андреевич

(фамилия, имя, отчество автора)

Тема работы Электроника стенда по изучению сцинтилляционных кристаллов

«К защите допущена»

Заведующий кафедрой,
канд. физ.-мат. наук, доцент,
с.н.с. ИЯФ СО РАН
Кроковный, П. П. / _____
(фамилия, И. О.) (подпись)
«____» _____ 2020 г.

Научный руководитель
канд. физ.-мат. наук,
с.н.с. ИЯФ СО РАН
Жуланов В. В. / _____
(фамилия, И. О.) (подпись)
«____» _____ 2020 г.

Дата защиты: «____» _____ 2020 г.

Новосибирск, 2020

Содержание

Введение

Детекторы ионизирующего излучения — это одни из наиболее важных элементов практически любой современной экспериментальной установки в области физики высоких энергий. В Институте ядерной физики СО РАН реализуется проект по выращиванию неорганических сцинтилляционных кристаллов, которые являются неотъемлемой частью таких детекторов. Сцинтилляторы — это вещества, способные излучать фотоны при поглощении ионизирующего излучения.

Для проверки характеристик и качества изготавливаемых сцинтилляционных кристаллов ведётся разработка специального стенда. Данный стенд имеет довольно сложное устройство, о нём будет рассказано подробнее в разделе "Установка стенда по исследованию сцинтилляционных кристаллов". Управляющим компонентом стенда является система на кристалле(СнК) Xilinx Zynq-7000 [1], являющаяся объединением процессора и программируемой логической интегральной схемы. Оператор сможет через порт Ethernet подключиться к веб-серверу, запущенному на СнК, через который будет производиться управление стендом и визуализация данных. Оценка параметров исследуемых сцинтилляционных кристаллов производится путём настройки временных и амплитудных характеристик формирователей входных сигналов.

Ранее для взаимодействия со стендом было начато создание интерфейса — веб-сервера, запускаемого непосредственно на СнК, доступ к которому оператор получал через порт Ethernet. Также была частично реализована программируемая логика, подробнее она будет описана в соответствующей главе.

1 Физика эксперимента

Детектор ионизирующего излучения — это устройство, которое способно преобразовывать энергию излучения в иной вид энегрии, удобный для последующей регистрации. По физическим принципам действия детектора можно выделить основные группы:

- сцинтилляционные;
- ионизационные;
- полупроводниковые.

В рамках данной работы особый интерес представляют сцитилляционные методы детектирования. Но перед их рассмотрением стоит рассказать о сцинтилляторах и некоторых их свойствах.

1.1 Сцинтилляционные кристаллы

Сцинтилляторы — это вещества, способные излучать свет при поглощении ионизирующего излучения. Сцинтилляторы характеризуются множеством параметров, но основными являются:

- конверсионная эффективность;
- технический выход;
- время высвечивания.

Конверсионной эффективностью или физическим выходом называется отношение энергии световой вспышки к энергии, потерянной частицей в кристалле. Таким образом, физический выход характеризует эффективность преобразования энергии ионизирующей частицы в световую в сцинтилляторе. Как правило, данная характеристика лежит в диапазоне от долей процента до десятков процентов.

Однако высокое значение конверсионной эффективности не является показателем пригодности вещества в качестве сцинтиллятора в детекторе. Для его использования необходимо, чтобы излучаемый свет мог свободно покидать пределы кристалла. Отношение энегрии световой вспышки, вышедшей из кристалла, к полной энегрии, потерянной частицей в нём, называется техническим выходом или технической эффективностью. Именно этот параметр является основополагающим в определении удовлетворительности качества сцинтиллятора. Он зависит от множе-

ства аспектов: толщины слоя сцинтиллятора, состояния его поверхности, концентрации поглощающих примесей, прозрачности кристалла к собственному излучению и так далее.

Зачастую интенсивность излучения кристалла I в зависимости от времени t описывается экспоненциальной формулой:

$$I(t) = I_0 e^{-\frac{t}{\tau}}, \tag{1}$$

где I_0 - амплитуда светового импульса, τ - время, в течение которого интенсивность излучения падает в e раз, называется временем высвечивания сцинтиллятора.

В настоящей работе время высвечивания кристалла является очень важным параметром, поскольку оно определяет необходимый промежуток осциллограммы для корректной записи экспериментальных данных. Данная деталь будет описана ниже при рассмотрении технической реализации системы.

1.2 Сцинтилляционные методы детектирования

Первый сцинтилляционный детектор назывался спинтарископом и был изобретён Уильямом Круксом в 1903 году. Главной его частью был небольшой экран, покрытый сульфидом цинка (ZnS). При попадании на него заряженных α-частиц возникала слабая световая вспышка - сцинтилляция, которую можно было наблюдать в микроскоп или даже адаптированным к темноте невооружёным глазом.

В настоящее время сцинтилляционный детектор представляет собой устройство, содержащее кроме сцинтиллятора фотоприёмник и зарядочувствительный усилитель (ЗЧУ). Схема устройства представлена на рисунке 1.

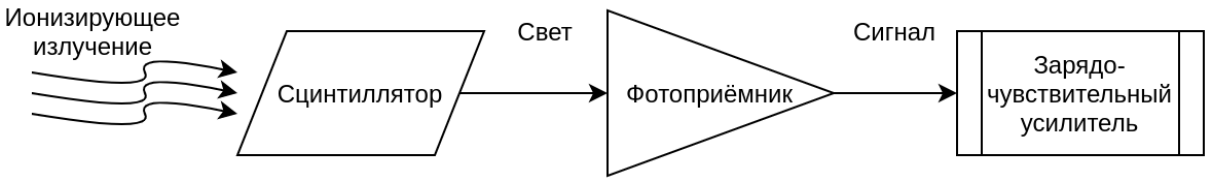

Рис. 1: Схема сцинтилляционного детектора

Фотоприёмник преобразует излучённую кристаллом световую вспышку в импульс электрического тока. Полученный сигнал принимается ЗЧУ, который преобразует электрический ток в заряд. По величине заряда можно восстановить количество энегрии, потраченной сцинтиллятором на высвечивание за определённое время.

2 Установка стенда по исследованию сцинтилляционных кристаллов

Блок-схема установки изображена на рисунке 2.

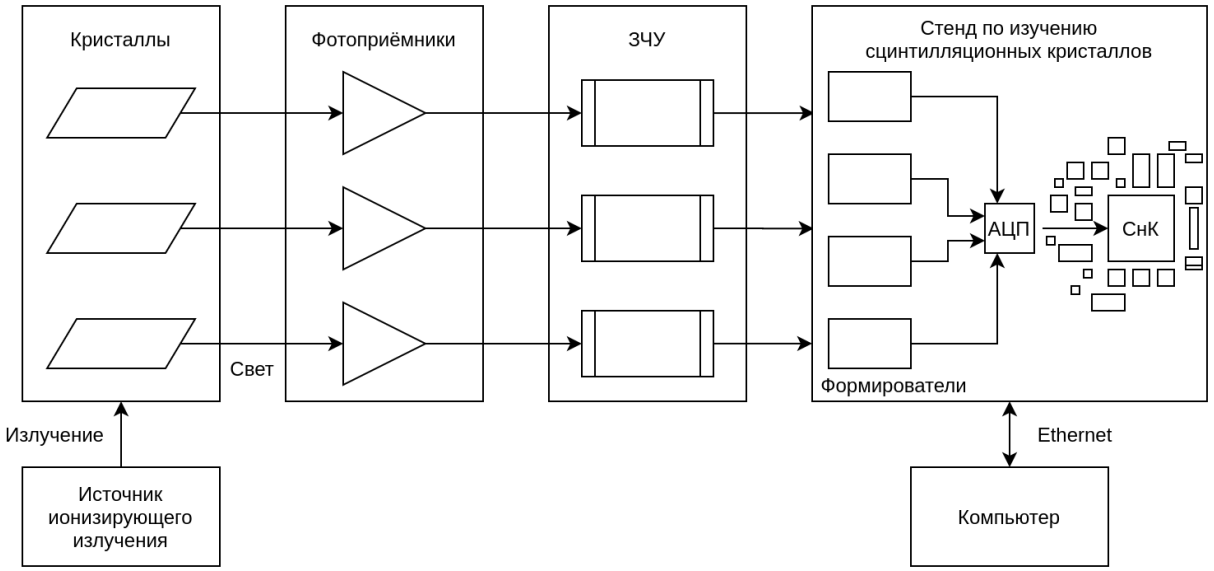

Рис. 2: Блок-схема установки

Ионизирующее излучение с источника попадает на три сцинтилляционных кристалла: исследуемый и два вспомогательных. Излучаемые кристаллами фотоны регистрируются в фотоприёмниках и преобразуются в электрические сигналы. После усиления в ЗЧУ сигналы подаются на входные каналы стенда, где они обрабатываются. Результат обработки отправляется на компьютер оператора через интерфейс Ethernet. Стенд имеет, кроме основного канала, предназначенного для исследуемого кристалла, два дополнительных — для вспомогательных кристаллов.

Стенд является ключевым элементом установки. На рисунке 3 представлена его блок-схема.

Стенд имеет 3 входных канала: основной и 2 вспомогательных. На каждом из них предусмотрен усилитель, сигнал с которого передаётся в набор формирователей, определяющих время формирования. Далее через промежуточный буфер с дифференциальным выходом сигнал поступает в 14-битный АЦП, где производится его конвертация в цифровой вид. Оцифровка происходит на тактовой частоте 100 МГц, выдаваемой

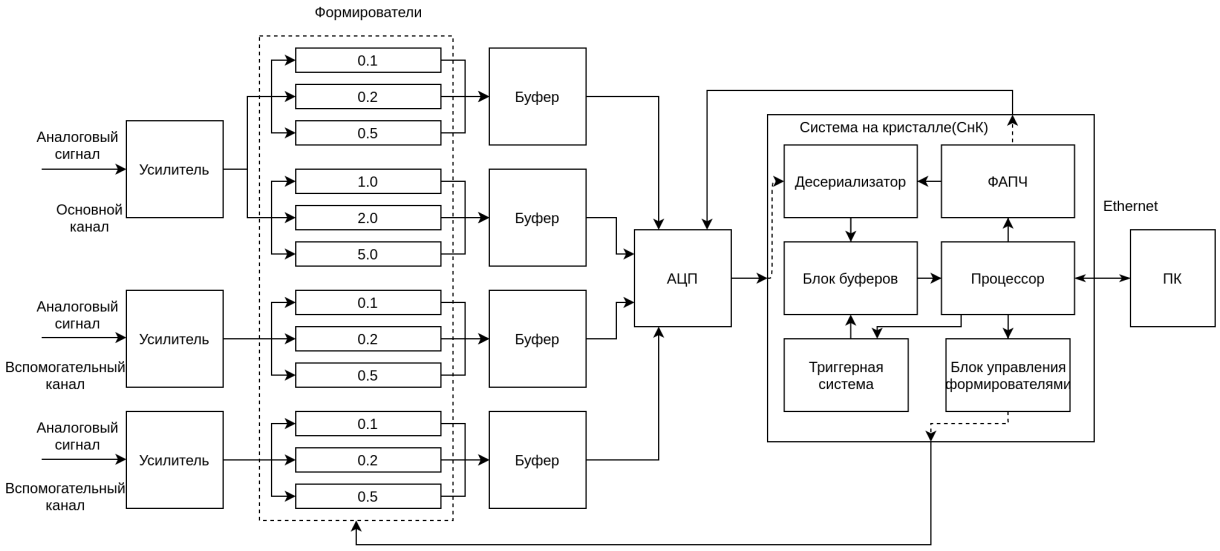

Рис. 3: Блок-схема стенда

модулем фазовой автоподстройки частоты (ФАПЧ), реализованным в системе на кристалле. Цифровые данные в последовательно упакованном формате передаются в СнК, где проводится их обработка.

В первую очередь, происходит обратная конвертация из последовательности бит в число (десериализация), после чего, при срабатывании триггерной системы, данные из блока буферов отправляются в процессор для последующей обработки. Для связи с компьютером используется протокол Ethernet.

На рисунке 4 представлена фотография стенда с выделением основных блоков:

1. Блок питаний;
2. Система на кристалле Zynq-7000 с необходимой периферией;
3. 4-х канальный АЦП;
4. Формирователи основного и вспомогательных каналов;
5. Усилители сигналов;
6. Входные разъёмы.

Далее будут рассмотрены подробнее особенности устройства некоторых частей описанной системы.

Основной канал

Основной канал имеет 2 набора формирователей с различными временами формирования: 0.1, 0.2, 0.5 и 1, 2, 5 мкс соответственно. Данные временные значения формирователей подобраны на основе анализа

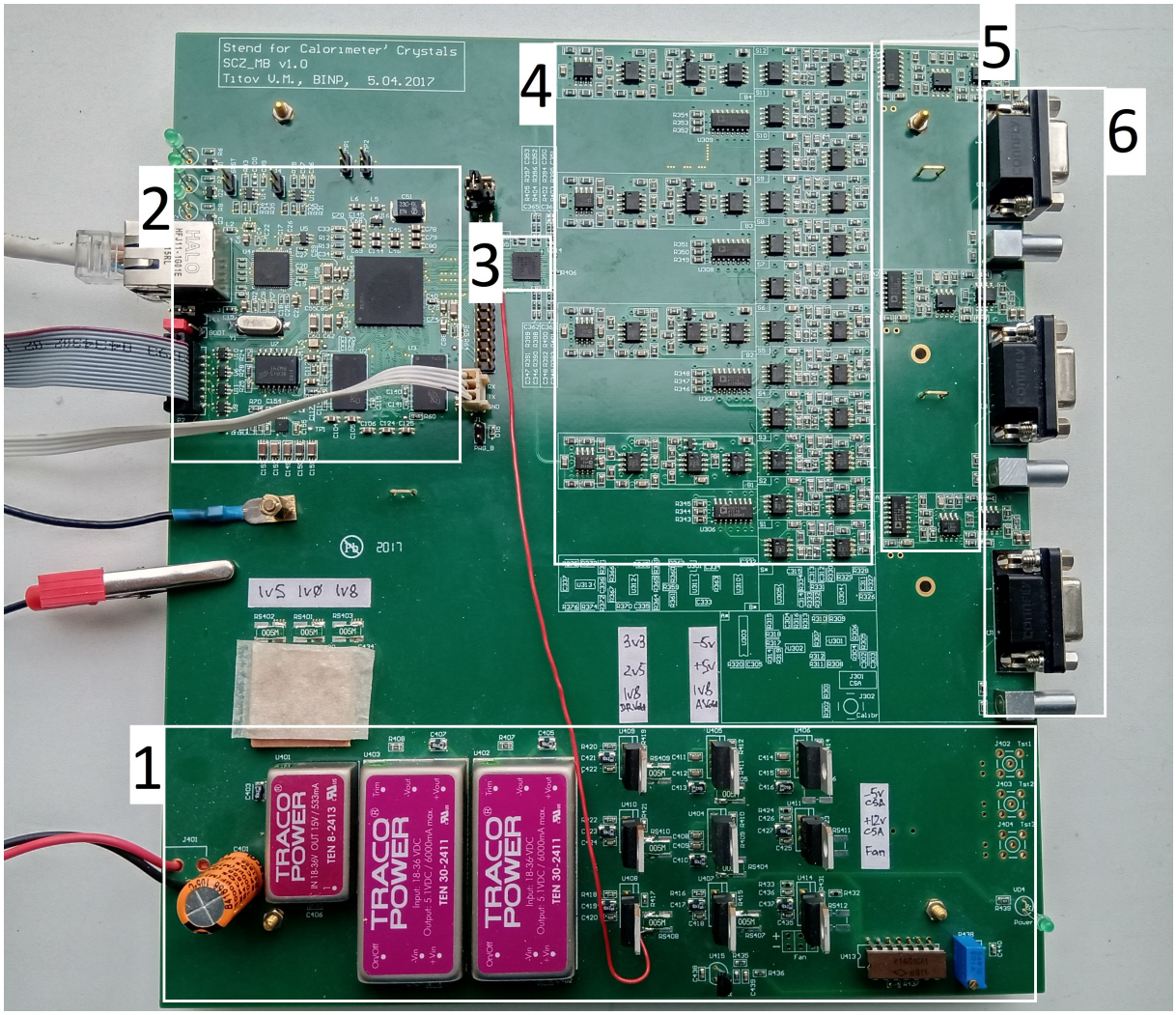

Рис. 4: Фотография стенда: 1 - блок питаний; 2 - СнК ZYNQ-7000 с необходимой периферией; 3 - АЦП; 4 - формирователи основного и вспомогательных сигналов; 5 - усилители сигналов; 6 - входные разъемы

свойств сцинтилляционных кристаллов, а также имеющегося опыта работы с ними. Выбранные значения позволяют обеспечить корректную работу с широким набором кристаллов: от имеющих сравнительно малое время высвечивания до обладающих длительным.

После каждого формирователя установлены электронные ключи, с помощью которых можно подключить выход одной секции из набора через буфер ко входу АЦП. Таким образом, на основном канале имеется возможность подключить одновременно 2 формирователя из диапазона 0.1 - 0.5 и 1 - 5 мкс соответственно. Оцифрованные данные непрерывно записываются в кольцевой буфер, откуда они могут быть выгружены для обработки и сохранения при поступлении сигнала о полезном событии от триггерной системы.

Вспомогательные каналы

Вспомогательные каналы служат источниками дополнительных сигналов, необходимых для правильной работы триггерной системы. Устройство вспомогательных каналов практически аналогично основному. Отличия:

- каждый из них содержит только по одному набору формирователей с тремя секциями с временами 0.1, 0.2, и 0.5 мкс;
- к аналогово-цифровому преобразователю может быть подключен выход только одной секции формирования канала.

Триггерная система

Триггерная система выполняет задачу формирования сигнала, означающего возникновение полезного события, при котором данные из кольцевого буфера необходимо выгрузить для последующей обработки. Система может работать в двух режимах: принудительный старт и срабатывание по порогу. В первом случае триггерная система вырабатывает сигнал при получении команды от оператора. Во втором случае триггер срабатывает при превышении текущими цифровыми значениями основного и/или некоторых вспомогательных каналов заданных оператором порогов.

3 Система на кристалле Xilinx Zynq-7000

В роли управляющего компонента стенда выступает система на кристалле Xilinx ZYNQ-7000 XC7Z020 CLG400, являющаяся объединением процессорной системы и программируемой логики. Архитектура данного кристалла представлена на рисунке 5.

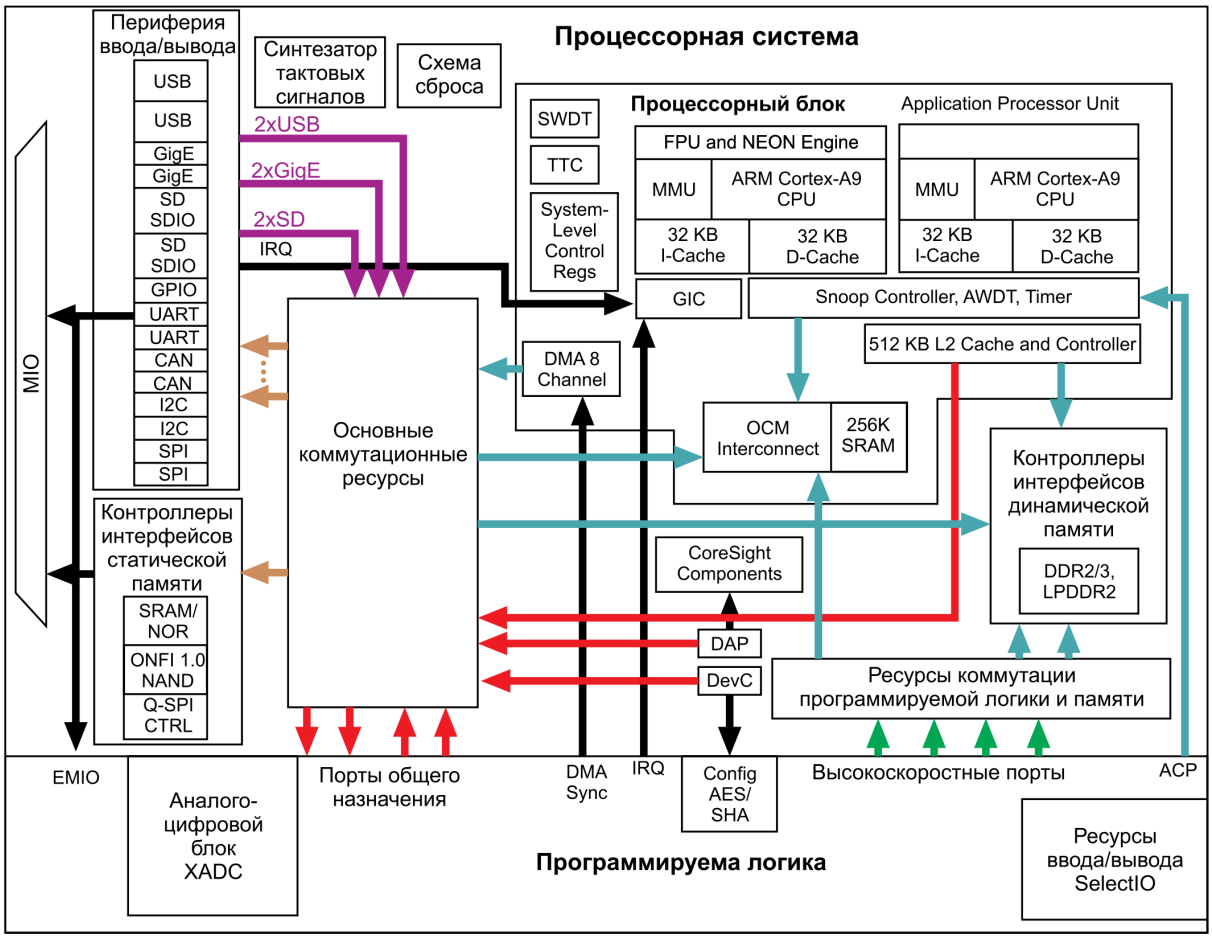

Рис. 5: Архитектура программируемой системы на кристалле XC7Z020

Процессорная система реализована на базе аппаратного блока, включающего в себя два ядра ARM Cortex-A9 MPCore [2], кэш-память первого и второго уровней объёмом 32 и 512 кбайт соответственно. В качестве оперативной памяти присутствует внутрикристалльное ОЗУ ёмкостью 256 кбайт, а также контроллер внешней высокоскоростной оперативной динамической памяти, поддерживающий многие современные спецификации. Периферия ввода/вывода включает в себя интерфейсы USB 2.0, Tri-mode Gigabit Ethernet, SD/SDIO, UART, CAN 2.0, I2C и SPI, каждый

из которых реализован дважды.

Программируемая логика данной системы на кристалле близка к ПЛИС семейства Artix-7. Это решение включает в себя 87040 логических ячеек, 140 модулей памяти Block RAM общей ёмкостью 560 кбайт, 220 секций цифровой обработки сигналов DSP48E1 и один аналогово-цифровой блок XADC.

Взаимодействие двух рассмотренных частей: программируемой логики и процессорной системы осуществляется через порты шины AXI [3]. Данная шина имеет 3 интерфейса:

- AXI4 — для высокопроизводительной работы с устройствами, отображаемых на адресное пространство памяти;
- AXI4-Lite — для простой, низкоскоростной связи с переферийными устройствами, не имеющих высоких требований к пропускной способности(например, для регистров управления и состояния);
- AXI4-Stream — для высокоскоростной потоковой передачи данных.

Система на кристалле даёт возможность одновременно использовать преимущества процессорных вычислений и параллельной обработки данных программируемой логики.

4 Цель и задачи работы

Главной целью данной работы является разработка программного обеспечения для системы на кристалле Zynq-7000 и доведение стенда до готовности к предстоящей эксплуатации по назначению. Таким образом, предстоит довольно широкий ряд работ: от разработки дизайна программируемой логики до конфигурации операционной системы и написания веб-сервера для взаимодействия со стендом.

Ранее для СнК уже проводились следующие работы:

- создание сервера и клиентской части, а также процессорной системы для их тестирования (необходима доработка);
- конфигурация операционной системы (остался только бинарный файл под другую версию кристалла, который установлен на тестовой плате);
- разработка программируемой логики (некоторые модули системы завершены, остальные требуют доработки или написания с нуля).

Стоит отметить, что данные модули были реализованы отдельно друг от друга без возможности совместного функционирования.

В рамках данной работы были поставлены следующие задачи:

- доработка ранее написанной программируемой логики, проектирование процессорной системы и их интеграция для совместной работы;
- разработка дизайна программируемой логики для подсчёта статистики данных с АЦП;
- конфигурация операционной системы;
- доработка сервера и клиентского веб-интерфейса, расширение его функционала.

5 Дизайн системы на кристалле

Для создания программируемой логики внути СнК и реализации её взаимодействия с процессором проектируется дизайн системы на кристалле. Соответственно, он делится на две части: программируемая логика и процессорная система, для каждой из которых разрабатывается свой поддизайн. Важной задачей является осуществление их связи. В данной работе используется следующие типы взаимодействия:

- чтение/запись данных напрямую в блок памяти через выделенный порт;
- чтение/запись регистров.

Разработка производилась в среде программирования Xilinx Vivado Design Suite на языке описания программной аппаратуры интегральных схем VHDL.

5.1 Процессорная система

Для разработки процессорной системы существуют готовые блоки, предоставляемые компанией-производителем Xilinx. В данной работе используются некоторые из них. Также для передачи команд и параметров оператора был разработан пользовательский блок виртуальных регистров. Список всех блоков и их краткое описание представлены в таблице 1.

Как видно из приведённой таблицы, в процессорной системе используется несколько готовых блоков от компании Xilinx и один пользовательский модуль, который имеет смысл рассмотреть подробнее. Он разработан с использованием AXI4-Lite. Данного интерфейса хватает для корректной работы модуля в силу небольшого объёма данных, передаваемых за одну транзакцию. Задача блока `reg_interface` заключается в передаче коротких параметров, команд и сигналов подтверждения из процессора в программируемую логику. Сигналы модуля представлены на рисунке 6.

Каждый пользовательский сигнал (`dataIn`, `regWE`, `regNum`, `dataOut`) ассоциирован с определённым участком в памяти, таким образом, текущее значение по выбранному адресу является состоянием сигнала. Блок

Таблица 1: Блоки дизайна процессорной системы

Наименование блока	Описание
Процессорная система ZYNQ7 Processing System [4]	Программный интерфейс вокруг процессорной системы платформы Zynq-7000
Контроллер блоков памяти AXI BRAM Controller [5]	Является конечным ведомым модулем для интеграции с интерфейсом шины AXI и системными главными устройствами для связи с локальными блоками оперативной памяти
Интерфейс шины AXI Interconnect [6]	Соединяет один или более AXI устройств, отображенных на память в режиме мастера, к одному или более устройствам, отображённых на память в режиме ведомого
Сброс процессорной системы Processor System Reset [7]	Обеспечивает индивидуальные сбросы для всей процессорной системы, включая процессор и периферийные устройства
Интерфейс модуля виртуальных регистров reg_interface	Пользовательский блок, использующий интерфейс AXI4-Lite
Генератор блоков памяти Block Memory Generator [8]	Автоматизирует создание блочных запоминающих устройств для программируемой логики

связан с модулем `reg_file` программируемой логики, подробное описание которого будет приведено в соответствующей главе. Сейчас важно то, что этот модуль содержит виртуальные регистры, операции с ними осуществляются посредством рассматриваемого модуля `reg_interface`, который переводит их в операции с реально существующими регистрами.

Для записи в виртуальный регистр необходимо вставить данные в сигнал `dataOut`, затем установить номер регистра в сигнал `regNum`, после чего подать единицу в сигнал `regWE`.

Для чтения из виртуального регистра необходимо установить номер интересующего регистра в сигнал `regNum`, после считать данные из сигнала `dataIn`.

Общая диаграмма блоков процессорной системы изображена на рисунке 7.

Кроме интерфейса виртуальных регистров, рассмотренного выше,

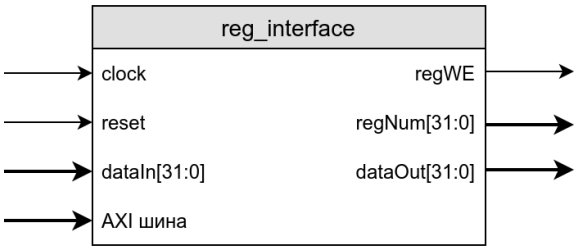

Рис. 6: Сигналы блока reg_interface

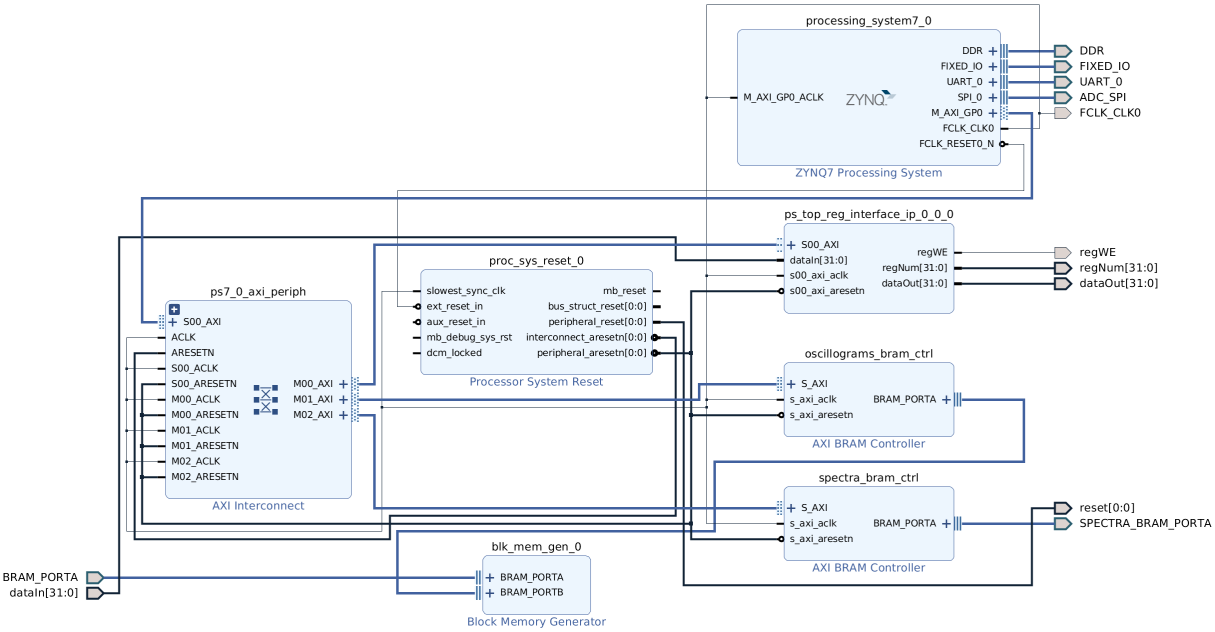

Рис. 7: Диаграмма блоков процессорной системы

для обмена данными между процессорной системой и программируемой логикой используется модуль двухпортовой памяти `blk_mem_gen_0`: через порт A происходит запись данных из программируемой логики, а через порт B информация считывается и отправляется в процессор. Настоящий модуль предназначен для передачи данных осциллограмм для их последующего отображения оператору стенда. Схожий блок двухпортовой памяти, необходимый для передачи статистических данных гистограмм расположен в части программируемой логики. Для их интеграции с интерфейсом шины AXI в процессорной системе предусмотрены контроллеры блоков памяти `oscillograms_bram_ctrl` и `spectra_bram_ctrl` соответственно.

5.2 Программируемая логика

На рисунке 8 представлена схема программируемой логики.

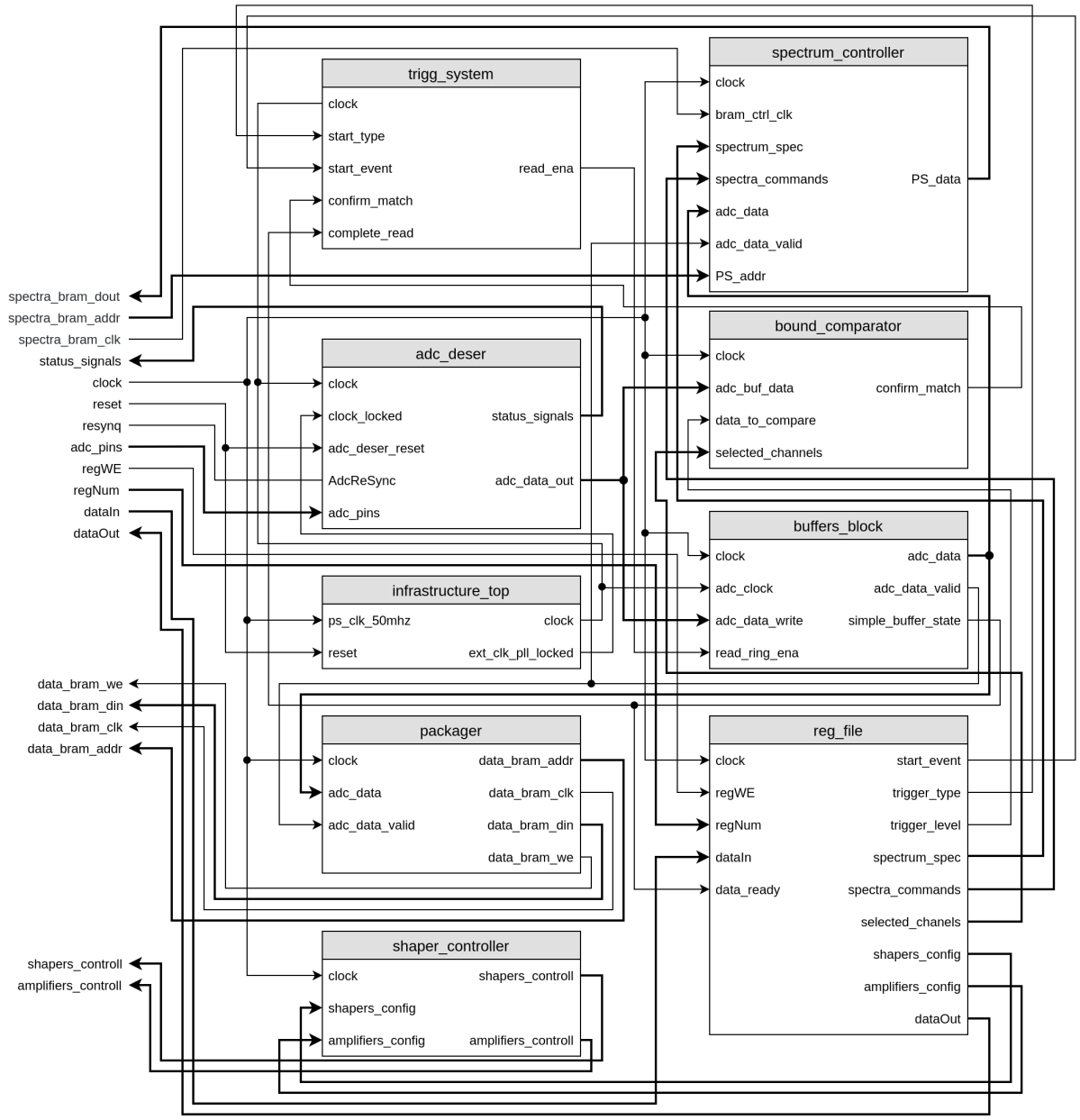

Рис. 8: Блок-схема программируемой логики

Программируемая логика состоит из 9 блоков, краткое описание который представлено в таблице 2.

Десериализатор adc_deser

Одним из основных элементов стенда является АЦП AD-9253 [9]. Преобразователь работает на 100 МГц параллельно в 4-х каналах. Данные с разрешением 14 бит передаются по протоколу LVDS. Этот стандарт

Таблица 2: Блоки программируемой логики

Наименование блока	Описание
adc_deser	Конвертирует упакованные последовательно данные АЦП в численные значения
infrastructure_top	Обеспечивает тактовую частоту для некоторых модулей
buffers_block	Буферизует входные данные
trigg_system	Генерирует сигнал для сохранения данных
bound_comparator	Выполняет сравнение входящих данных с заданными порогами
spectra_controller	Производит обработку данных для набора статистики
shaper_controller	Осуществляет управление формирователями сигналов
reg_file	Реализует блок виртуальных регистров
packager	Упаковывает данные и передаёт их в процессорную систему

предполагает передачу информации в последовательно-упакованном виде по 2 каналам на каждый вход. На рисунке 9 представлена временная диаграмма работы АЦП.

Каждый такт работы АЦП производится оцифровка входного аналогового сигнала. Как видно из временной диаграммы, полученные данные отправляются за 1 такт через 17 тактов после оцифровки по двум каналам. Представляться они могут в одном из двух вариантов: побитовый (bitewise mode) и побайтовый (bytewise mode). Данные режимы отличаются последовательностью упаковки битов — в разных каналах передаются либо четные и нечётные биты, либо младший и старший байты соответственно. В настоящей работе выбран побитовый режим.

Также в работе АЦП участвуют 2 вспомогательных сигнала — кадровый строб, отвечающий за разделение набора бит на кадры оцифровки, и тактовый сигнал передачи данных, который размечает биты внутри каждого кадра.

Таким образом, возникает необходимость реализации модуля, осуществляющего конвертацию последовательности бит, поступающей из

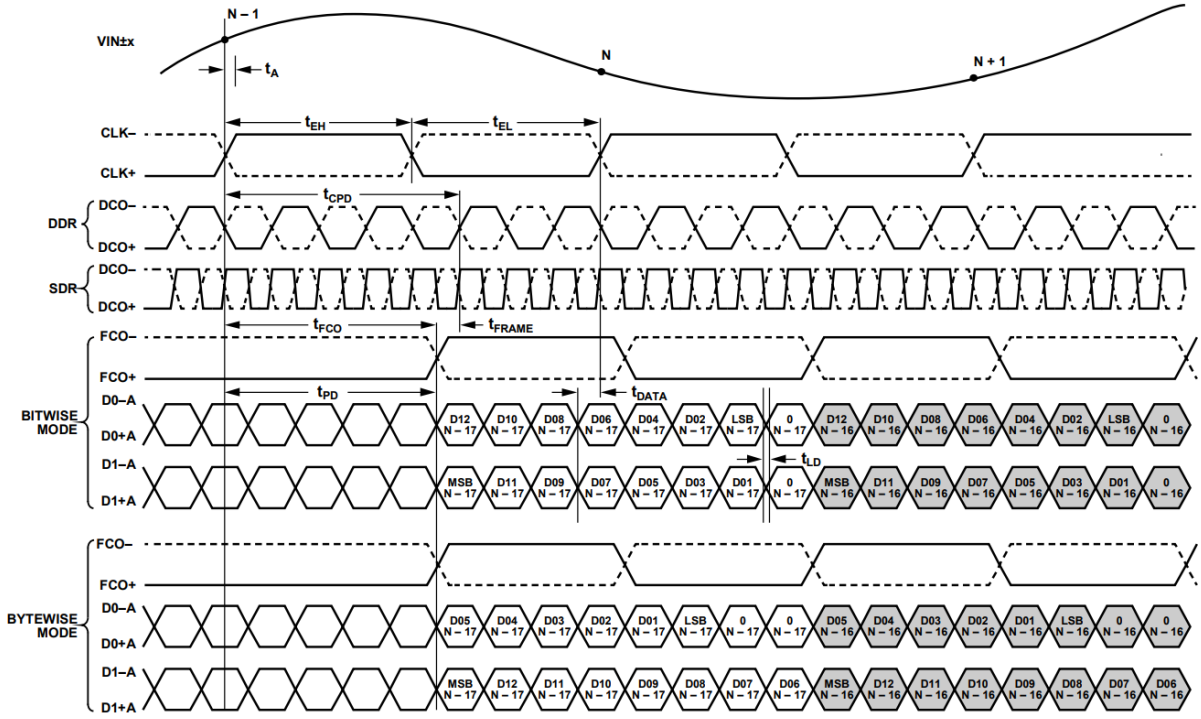

Рис. 9: Временная диаграмма работы АЦП

АЦП, в удобное для обработки численное значение. Так как процесс упаковки бит в определённую последовательность называется сериализацией, то обратную операцию можно назвать десериализаций, а соответствующий модуль — десериализатором. Его сигналы изображены на рисунке 10.

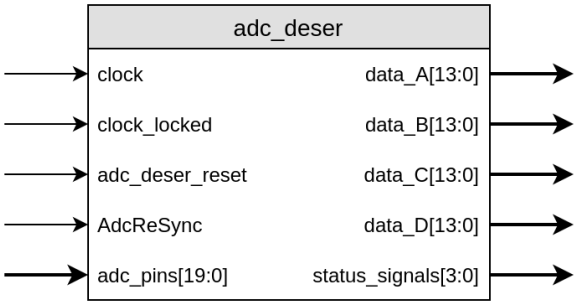

Рис. 10: Сигналы модуля десериализатора

Модуль был разработан ранее на основе готового интерфейса компании Xilinx, поэтому подробного описания его работы приведено не будет. Стоит лишь отметить, что входные данные принимаются через набор сигналов `adc_pins`, а десериализованные значения подаются на выход через сигналы `data_i`.

Модуль infrastructure_top

Блок `infrastructure_top` в настоящее время содержит лишь модуль фазовой автоподстройки частоты (ФАПЧ), необходимый для генерации тактового сигнала для работы АЦП и блоков, которые занимаются обработкой входных данных и их буферизацией (модули десериализатора и блока буферов). Как и десериализатор модуль ФАПЧ был разработан ранее с использованием библиотеки сложных функциональных блоков и рассматриваться подробно не будет. Сигналы модуля `infrastructure_top`, в котором расположен ФАПЧ изображены на рисунке 11.

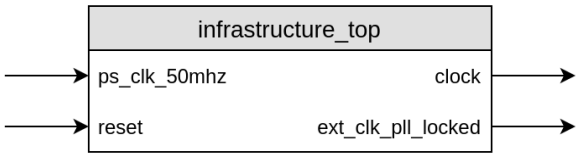

Рис. 11: Сигналы модуля infrastructure_top

Блок буферов buffers_block

В силу случайного характера возникновения полезных событий возникает необходимость временного хранения определённого числа последних измерений с АЦП. Для решения задачи был разработан модуль блока буферов, сигналы которого изображены на рисунке 12.

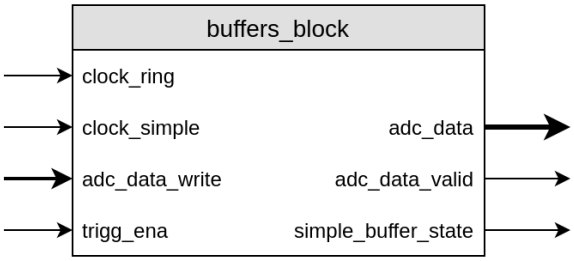

Рис. 12: Сигналы модуля блока буферов

Блок содержит в себе 2 модуля RAM памяти с раздельными портами чтения/записи данных. Первый является кольцевым буфером и непрерывно записывает данные АЦП. Это позволяет вычитывать при необходимости данные АЦП, пришедшие до сигнала триггерной систе-

мы. Такая необходимость обусловлена возможными задержками сигнала триггера, а также потребностью иметь небольшую "предысторию"осциллограммы перед достижением сигнала порогового значения.

Второй модуль памяти выполняет функцию простого буфера, в который временно будут выгружаться полезные данные при возникновении сигнала триггерной системы. Главной задачей такого буфера является хранение данных для гарантированного считывания и передачи их в процессорную систему.

Объём простого буфера определяется необходимым размером осциллограммы, который, в свою очередь, зависит от максимального времени высвечивания кристалла. В настоящей работе объём простого буфера установлен 128 кадрами. Такой размер позволяет поместить в одну осциллограмму данные, оцифрованные за промежуток времени порядка 1 мкс при работе АЦП на 100 МГц. Объём кольцевого буфера, в свою очередь, определяется максимальным количеством данных "предыстории необходимых для анализа. Из свойств сцинтилляционных кристаллов было принято решение об использовании 64 кадров.

Модуль `buffers_block` имеет два входных тактовых сигнала – `clock_ring` и `clock_simple`. Система на кристалле работает на 50 МГц, следовательно, именно на этой частоте должны выгружаться данные в процессорную часть. При этом АЦП работает на 100 МГц. Здесь кроется ещё одна немаловажная задача простого буфера: переход данных из одного домена тактовой частоты в другой. Таким образом, поток данных поступает в блок на тактовом сигнале работы АЦП, а полезная информация выдаётся на частоте работы системы на кристалле.

Из рисунка видно, что модуль также содержит следующие входные сигналы: `adc_data_write` – входные данные с АЦП, `trigg_ena` – сигнал триггерной системы о возникновении полезного события. Можно заметить, что выходной набор `adc_data` содержит большее число сигналов, чем входной `adc_data_write`. Это связано с добавлением к данным по 2 бита, идентифицирующих их отношение к определённому каналу. Таким образом, значение каждой оцифровки хранится ровно в двух байтах.

Триггерная система trigg_system

Как было сказано ранее, для сохранения данных с АЦП в буфер и

последующей их передачи в процессор необходим сигнал, сообщающий о возникновении полезного события. Генерацией такого сигнала занимается модуль триггерной системы, сигналы которого изображены на рисунке 13.

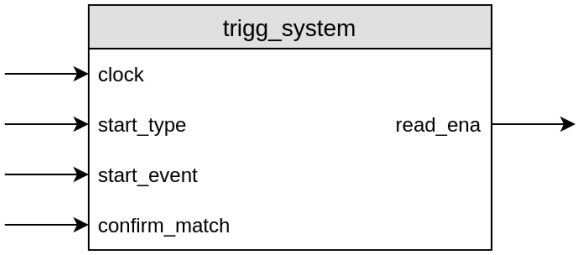

Рис. 13: Сигналы модуля триггерной системы

Триггерная система может работать в двух режимах, задаваемых сигналом `start_type`: принудительно и по порогу. В первом случае модуль отработает сразу при появлении сигнала старта на входе `start_event`. В режиме срабатывания по порогу триггер сгенерирует разрешение на запись только при появлении на входе `confirm_match` высокого уровня от модуля компаратора `bound_comparator`.

Компаратор bound_comparator

Компаратор осуществляет сравнение текущих оцифрованных данных АЦП с заданными оператором значениями порогов. Это необходимо для корректного функционирования триггерной системы в соответствующем режиме работы. Сигналы блока изображены на рисунке 14.

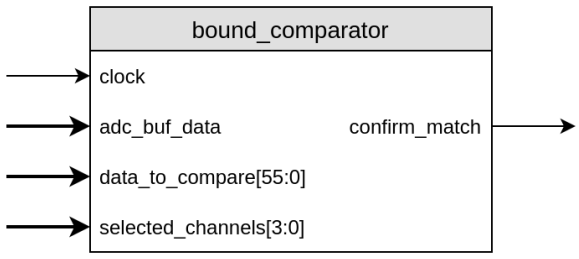

Рис. 14: Сигналы модуля компаратора

Данный модуль функционирует на тактовой частоте работы АЦП, т.к. он должен непрерывно сравнивать каждое оцифрованное значе-

ние, поступающее через вход `adc_buf_data`. При достижении или превышении порогового значения модуль формирует на выходном сигнале `confirm_match` логическую единицу, а в противоположном случае ноль. Порог для каждого канала задаётся отдельно с помощью сигналов `data_to_compare`. Стоит отметить, что реализована возможность выбирать каналы, выполнение условий по которым повлечёт срабатывание модуля. Оператор может назначать их в режиме логического ИЛИ: система выдаст сигнал при срабатывании компаратора хотя бы по одному из выбранных каналов. Эта информация поступает в блок компаратора через сигналы `selected_channels`.

Модуль набора статистики spectra_controller

Для исследования сцинтилляционных кристаллов оказываются полезными не только осциллограммы высвечивания, но также статистика по определённому количеству кадров. Для набора таких данных реализован отдельный модуль, сигналы которого изображены на рисунке(15).

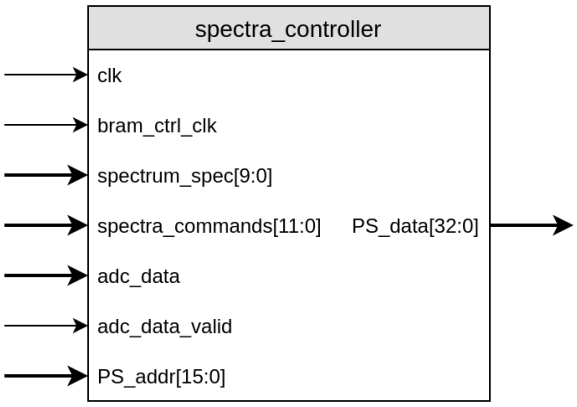

Рис. 15: Сигналы модуля набора статистики

Гистограммы нужны двух типов: по максимальному значению и значению определённого измерения каждого кадра. Для выполнения этих задач был разработан модуль `spectrum_creator`, который способен находить самое большое значение переданной ему осциллограммы в определённом канале и, исходя из него, генерировать сигнал о необходимости инкрементировать соответствующий счетчик, расположенный в двухпортовой памяти. Вместо поиска максимума есть возможность задать конкретное измерение. Исходя из требований эксперимента было при-

нято решение, что на каждый канал будет достаточно одновременного набора гистограмм по максимальному значению и по двум заданным точкам. Таким образом, на каждый канал необходимо по три вышеописанных модуля. Блок `spectra_controller` является объединением двенадцати таких элементов и модуля `spectra_memory`, содержащего в себе двенадцать блоков двухпортовой памяти. Чтение и запись данных со стороны программируемой логики может производиться параллельно во все блоки памяти модуля `spectra_memory`, при этом для чтения в процессорную систему реализовано общее адресное пространство по всем блокам памяти.

Для работы всей системы к модулю подведён тактовый сигнал `clk`, данные с блока буферов `adc_data` и сигнал их готовности `adc_data_valid`. Также для настройки собираемой статистики присутствует канал
`spectra_params` — через него оператор может задавать количество корзин и выбранное измерение осциллограммы для гистограмм второго типа. Выходные данные передаются в процессорную часть с помощью сигналов `bram_ctrl_clk`, `PS_addr` и `PS_data`.

Модуль управления блоком усилителей и формирователей shapers_controller

Сгенерированный сигнал фотоприёмника, попадая на стенд, усиливается и проходит через блок формирователя. На стенде предусмотрена возможность выставлять различные коэффициенты усиления и времена формирования для более точной настройки при работе с различными кристаллами.

Для переключения времени формирования сигнала на плате установлены электронные ключи. Управление ими реализовывается сигналами рассматриваемого модуля `shapers_controller` (рисунок 16).

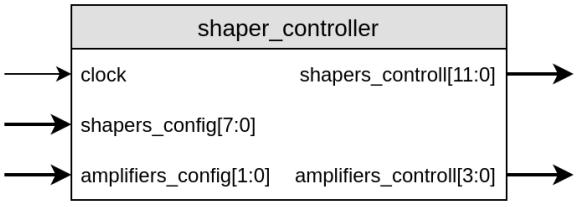

Рис. 16: Сигналы модуля shapers_controller

Блок получает информацию о необходимых настройках через сигнал `shapers_config`. Выходные сигналы `shapers_controll` идут непосредственно к ключам.

Усилители, в отличие от формирователей, имеют согласованные коэффициенты усиления: 1, 2, 4 и 5. Таким образом, их состояние можно задать всего двумя битами. Их управление осуществляется в этом же блоке с соответствующими сигналами `amplifiers_config` и `amplifiers_controll`.

Модуль виртуальных регистров reg_file

Данный модуль содержит в себе виртуальные регистры, с которыми работает блок `reg_interface` процессорной системы. Сигналы блока изображены на рисунке 17.

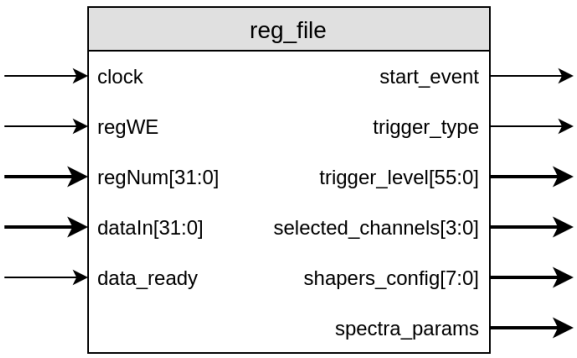

Рис. 17: Сигналы модуля reg_file

При записи данных модуль получает на вход номер виртуального регистра и желаемое значение для записи. После подачи логической единицы в сигнал `regWE` входящая информация обрабатывается и, в зависимости от регистра, выставляется в определённый сигнал для соответствующего модуля.

Для чтения используется сигнал `dataOut`, в который передаются данные из регистра по выставленному номеру. Среди входных сигналов можно заметить сигнал `data_ready`, передающий информацию о состоянии данных. При возникновении на нём единицы процессорная часть может получить сигнал о том, что данные готовы для считывания.

Упаковщик packager

Модуль упаковщика служит для отправки данных в двухпортовую

память процессорной системы. Его сигналы изображены на рисунке 18.

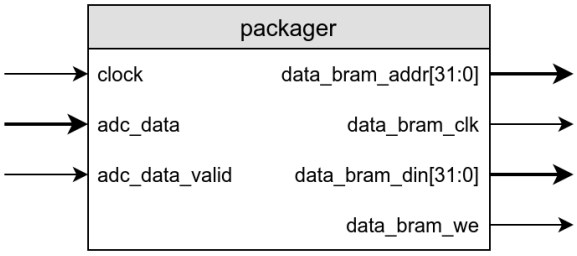

Рис. 18: Сигналы модуля packager

При возникновении высокого уровня на сигнале статуса данных `adc_data_valid` модуль отправляет информацию из сигнала `adc_data`, выставляя адреса в `data_bram_addr`, данные в `data_bram_din` и логическую единицу в `data_bram_we`.

6 Операционная система

В качестве операционной системы был выбран легковесный дистрибутив ОС Linux PetaLinux [10]. Данное решение предоставляется компанией Xilinx и позволяет упаковывать в образ программной платформы дизайн системы на кристалле вместе с операционной системой. Конфигурирование образа PetaLinux реализуется с помощью пакета PetaLinux Tools, который содержит в себе Yocto Extensible SDK необходимый для возможности изменения файловой системы ОС. Среди преимуществ данного решения можно отметить наличие подробной документации [11; 12], что делает его более удобным по сравнению с другими вариантами, такими как OpenBricks или Buildroot.

Среди ограничений, накладываемых на образ операционной системы можно выделить:

- размер образа не должен превышать объёма энегронезависимой памяти, устанновленного на стенде (64 МБ);
- система должна работать автономно.

Для корректной работы системы на стенде к ней предъявляются следующие требования:

- настроенный сетевой интерфейс для подключения с ПК оператора;
- наличие всех необходимых библиотек в файловой системе для работы сервера.

Для этого в файловую систему были добавлены такие компоненты как интерпритатор языка python, пакеты для работы веб-сервера и, непосредственно, сам сервер. Также был написана утилита для настройки сетевого интерфеса, а также скрипт для автоматической установки необходимых пакетов и запуска веб-сервера при старте системы.

7 Веб-сервер

Получение данных со стенда для их последующего анализа и управление устройством в настоящей работе реализованы через веб-интерфейс. Это очень удобное решение в силу простоты использования. При таком подходе, например, отсутствует необходимость установки специального ПО на ПК оператора. Из этого вытекает возможность взаимодействия со стендом с любой операционной системы — требуется лишь браузер.

Для корректной работы всей системы необходимо верно построить взаимосвязь серверной и клиентской частей.

Веб-сервер функционирует с использованием протокола HTTP и архитектурного стиля взаимодействия REST.

7.1 Серверная часть

Серверная часть разработана на языке python с использованием программной платформы Django Framework. Данные инструменты были выбраны ввиду наличия некоторых наработок для настоящей задачи. Среди достоинств такого решения можно выделить:

- простота добавления интерпретатора языка python в файловую систему, ввиду присутствия его в пакете PetaLinux Tools;
- подробная документация;
- скорость разработки (к примеру, язык C++ избыточно сложен для веб-разработки);
- отсутствие необходимости в кросс-компилляции
- расширяемость.

Главной задачей сервера является реализация отправки пользовательских комманд и параметров в модуль виртуальных регистров и считывание данных из двухпортовой памяти. Для этого были разработаны обработчики HTTP запросов, а также функции для работы с регистрами и чтения данных из памяти. Для доступа к переферии использован пакет python-periphery.

7.2 Клиентская часть

На рисунке 19 представлен внешний вид пользовательского веб-интерфейса.

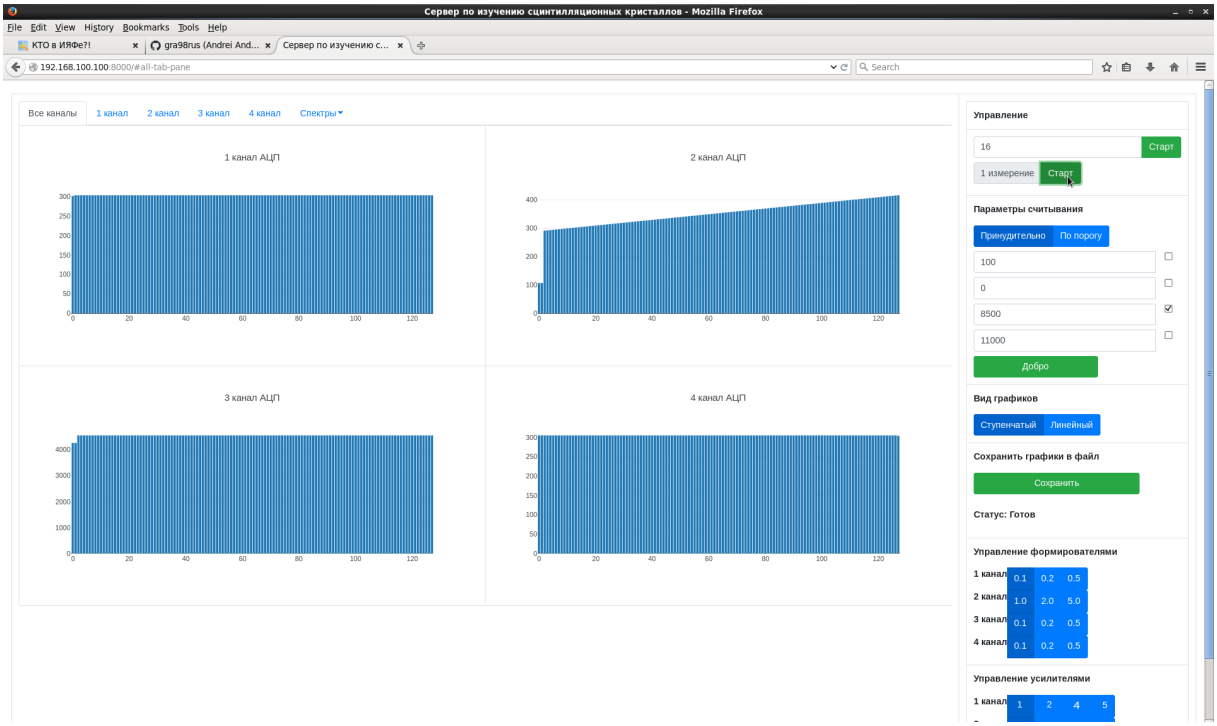

Рис. 19: Пользовательский веб-интерфейс

На главной странице расположены область с графиками и панель управления стендом. В верхней части страницы предусмотрены вкладки для переключения между графиками. Также здесь расположена вкладка “Спектры”, где оператор может управлять гистограммами: добавить новую с выбранными параметрами или удалить существующие. Среди параметров присутствуют следующие параметры: выбранный канал АЦП, количество корзин (от 32 до 4096) и тип гистограммы (по максимальному значению амплитуды или же по амплитуде опорной точки, которую также можно задать).

На боковой панели управления оператор может задавать следующие параметры:

- количество запусков;
- тип считывания (принудительный или по порогу);
- значения порогов для каждого канала АЦП, а также по каким ка-

налам возможно срабатывание;

- вид графиков (ступенчатый или линейный);
- коэффициент усиления сигналов и времена формирования каждого входного сигнала.

Также реализована возможность сохранять данные в файл для последующего анализа.

Разработка велась с использованием языков JavaScript, HTML и CSS.

Заключение

В рамках данной работы было разработано программное обеспечение для системы на кристалле Zynq-7000. Таким образом, были реализованы следующие задачи:

- проектирование архитектуры процессорной системы;
- разработка дизайна программируемой логики и её интеграция с процессорной системой для корректной совместной работы;
- разработка модуля подсчёта статистики данных АЦП;
- конфигурация операционной системы PetaLinux;
- разработка сервера и клиентского веб-интерфейса.

Одной из особенностей настоящей работы можно выделить широкий спектр использованных технологий, от низкоуровнего проектирования архитектуры системы на кристалле до высокоуровневой разработки веб-сервера.

Список литературы

1. Zynq-7000 SoC Data Sheet: Overview. — https://www.xilinx.com/support/documentation/data_sheets/ds190-Zynq-7000-Overview.pdf.
2. Cortex-A9 Technical Reference Manual. — https://static.docs.arm.com/ddi0388/i/DDI0388I_cortex_a9_r4p1_trm.pdf?_ga=2.105311619.1400052031.1590936331-252503034.1590936331.
3. AXI Reference Guide. — https://www.xilinx.com/support/documentation/ip_documentation/ug761_axi_reference_guide.pdf.
4. Processing System 7 v5.5. — https://www.xilinx.com/support/documentation/ip_documentation/processing_system7/v5_5/pg082-processing-system7.pdf.
5. AXI Block RAM (BRAM) Controller v4.0. — https://www.xilinx.com/support/documentation/ip_documentation/axi_bram_ctrl/v4_0/pg078-axi-bram-ctrl.pdf.
6. AXI Interconnect v2.1. — https://www.xilinx.com/support/documentation/ip_documentation/axi_interconnect/v2_1/pg059-axi-interconnect.pdf.
7. Processor System Reset Module v5.0. — https://www.xilinx.com/support/documentation/ip_documentation/proc_sys_reset/v5_0/pg164-proc-sys-reset.pdf.
8. Block Memory Generator v8.4. — https://www.xilinx.com/support/documentation/ip_documentation/blk_mem_gen/v8_4/pg058-blk-mem-gen.pdf.
9. Data Sheet AD9253. — https://www.analog.com/media/en/technical-documentation/data-sheets/ad9253.pdf.
10. PetaLinux Tools Documentation Reference Guide. — https://www.xilinx.com/support/documentation/sw_manuals/xilinx2018_3/ug1144-petalinux-tools-reference-guide.pdf.

11. PetaLinux Tools Documentation Workflow Tutorial. — https://www.xilinx.com/support/documentation/sw_manuals/xilinx2017_2/ug1156-petalinux-tools-workflow-tutorial.pdf.

12. PetaLinux Tools Documentation Command Line Reference Guide. — https://www.xilinx.com/support/documentation/sw_manuals/xilinx2019_2/ug1157-petalinux-tools-command-line-guide.pdf.