УДК 004.415.2

ПРОГРАММНО-АППАРАТНАЯ СИСТЕМА ДЛЯ ИССЛЕДОВАНИЯ ФИЗИЧЕСКИ НЕКЛОНИРУЕМЫХ ФУНКЦИЙ В БАЗИСЕ ПЛИС

Боронников А.С., Кряхтунов Г.М., Деменкова Т.А.

Федеральное государственное бюджетное образовательное учреждение высшего образования «МИРЭА – Российский технологический университет», 119454, Российская Федерация, г. Москва, пр-т Вернадского, 78, e-mail: boronnikov-anton@mail.ru, gleb.kryakhtunov@yandex.ru, demenkova@mirea.ru

В работе представлены результаты исследований в области доверенного проектирования вычислительных систем и их элементов. Рассматривается задача применения физически неклонируемых функций для защиты от нелегального копирования интегральных микросхем, которые используются в информационно-вычислительных системах различного назначения. Решается проблема создания универсальных программно-аппаратных средств для исследования физически неклонируемых функций в базисе программируемых логических интегральных схем, за счет которых принципиально меняется подход к практической реализации защитных средств путем введения программной надстройки для корректного определения идентификаторов элементов цифровых устройств вычислительной техники. В качестве объекта исследования выбрана физически неклонируемая функция типа арбитр. Предложена оригинальная архитектура системы и рассмотрена реализация сложных функциональных блоков, которые составляют основу этой системы. Разработано унифицированное программно-аппаратное решение для сбора и анализа данных с целью нахождения уникальных идентификаторов интегральных микросхем. Рассмотрен формат команд для взаимодействия аппаратной и программной частей разработанных средств. Подробно описана аппаратная часть системы, включая интерфейсы и графы переходов конечных автоматов сложных функциональных блоков. Отражены возможные нюансы, которые необходимо учитывать при проектировании. Разработано программное решение, с помощью которого осуществляется сбор, анализ и сохранение результатов в текстовом виде согласно заданному формату. Описаны алгоритмы основных режимов работы данного программного обеспечения. Приведены примеры использования системы для поиска эталонных значений идентификаторов.

Ключевые слова: доверенное проектирование, программно-аппаратные системы, физически неклонируемые функции, программируемые логические интегральные схемы.

SOFTWARE AND HARDWARE SYSTEM FOR STUDYING PHYSICAL UNCLONABLE FUNCTIONS ON FPGA

Boronnikov A.S., Kryakhtunov G.M., Demenkova T.A.

Federal State Budget Educational Institution of Higher Education «MIREA – Russian Technological University», 119454, Russian Federation, Moscow, Vernadsky pr., 78, e-mail: boronnikov-anton@mail.ru, gleb.kryakhtunov@yandex.ru, demenkova@mirea.ru

The paper presents the results of research in the field of trusted design of computing systems and their elements. The problem of using physically unclonable functions to protect against illegal copying of integrated circuits that are used in information computing systems for various purposes is considered. The problem of creating universal software and hardware for studying physically unclonable functions based on programmable logic integrated circuits is being solved, due to which the approach to the practical implementation of protective equipment is fundamentally changed by introducing a software add-on for correctly determining the identifiers of elements of digital computer devices. A physically unclonable function of the arbiter type was chosen as the object of study. An original system architecture is proposed and the implementation of complex functional blocks that form the basis of this system is considered. A unified hardware and software solution has been developed for collecting and analyzing data in order to find unique identifiers of integrated circuits. The format of commands for the interaction of hardware and software parts of the developed tools is considered. The hardware of the system is described in detail, including interfaces and transition graphs of finite state machines of complex functional blocks. Possible nuances that must be taken into account during design are reflected. A software solution has been developed with the help of which the results are collected, analyzed and saved in text form according to a given

format. Algorithms of the main operating modes of this software are described. Examples of using the system to search for reference identifier values are given.

Keywords: trusted design, hardware and software systems, physically unclonable functions, FPGA.

Введение

В настоящее время борьба с контрафактной продукцией стала одной из наиболее актуальных проблем в области электроники. Она не только наносит ущерб экономике, но и представляет серьезную угрозу безопасности, так как может привести к нарушению функционирования систем в работе оборудования. Решением данного вопроса является доверенное проектирование, в котором в последнее время выделяют использование так называемых физически неклонируемых функций.

Физически неклонируемые функции (ФНФ) представляют собой перспективное направление в области безопасности информационных технологий. ФНФ – это односторонняя функция, реализованная в физической структуре. Свойство неклонируемости ФНФ связано с несовершенством производственного процесса, поэтому невозможно создать два идентичных физических устройства с одинаковыми характеристиками. Можно считать, что эти характеристики принимают случайные значения [1].

В общем случае ФНФ описывается значениями пар входных и соответствующих им выходных сигналов. Такая пара, состоящая из входного параметра «запрос» (Challenge, CH) и выходного параметра «отклик» (Response, R), называется парой запрос-отклик (Challenge-Response Pairs, CRP). Тем самым, ФНФ можно рассматривать как функцию *f*, которая преобразует запросы CH_i в ответы R_i (1). Математический аналог ФНФ – это хэш-функция [2].

$$R_i = f(CH_i) \quad (1)$$

Множество CRP уникально идентифицирует физическое устройство и не может быть скопировано даже при условии использования абсолютно одинакового проектного описания.

Один из способов классификации заключается в том, в какой физической среде реализована ФНФ. Различают следующие виды [3]:

- Оптические ФНФ, где в качестве физической структуры выступает некий прозрачный материал (например, стекло);
- Магнитные ФНФ, где в качестве физической структуры выступает магнитный носитель;
- Кремниевые ФНФ, где в качестве физической структуры выступает кремниевая подложка.

В зависимости от вида физической структуры в качестве параметров выступают различные характеристики, которые задают значение случайной величины отклика *Response*. Вид ФНФ зависит от предполагаемого применения в конкретной задаче предметной области.

ФНФ, реализованная на кремниевой интегральной схеме (ИС), формируется неконтролируемым образом на всех этапах производства микросхемы, начиная с выращивания кристалла и заканчивая литографией на нем. Следовательно, ФНФ зависит от свойств материала подложки, электрических и физических свойств электронных компонентов [4]. Выделяют следующие основные типы ФНФ, реализованные в базисе ИС:

- ФНФ на базе СОЗУ (Статическое оперативное запоминающее устройство): основаны на случайных вариациях содержимого ячеек памяти при подаче питания на СОЗУ. При каждом включении устройства ячейки СОЗУ устанавливаются в случайные значения (0 или 1), которые можно использовать для создания уникального отклика.
- ФНФ типа арбитр (АФНФ): основаны на разных задержках прохождения сигналов в логических вентилях микросхемы. В силу не идеальности технологического процесса при изготовлении микросхемы значения задержек будет зависеть от физических характеристик полупроводников и проводников.
- ФНФ на базе кольцевых генераторов: основаны на различиях в частотных характеристиках колебательных контуров.

Вариативность реализации ФНФ на кристалле интегральных схем позволяет выделить основные направления применения ФНФ, такие как цифровые подписи, генерация псевдослучайных числовых последовательностей, идентификация и аутентификация устройств, реализация аппаратных хэш–функций, обнаружение аппаратных троянов, генерация ключей шифрования.

Для понимания того, как исследовать поведение ФНФ в базисе ИС, в работе рассмотрен пример системы с АФНФ. Классическая схема АФНФ основана на цепочке сдвоенных мультиплексоров, представляющих собой линию задержки. Она позволяет образовывать пары симметричных путей, форма которых определяется параметром сигнала запроса *Challenge* на адресных входах мультиплексоров. Несмотря на то, что пути топологически симметричны, задержка на этих путях будет отличаться от кристалла к кристаллу в силу не идеальности производства. На конце цепочки размещается арбитр, который позволяет представить соотношение задержек на путях, т.е. отклик *Response*. В качестве арбитра могут выступать защёлки или синхронные триггеры. В качестве примера рассмотрена АФНФ с арбитром в виде синхронного D-триггера (рис. 1). Чтобы получить ответ *Response*, на вход АФНФ подается импульс (*Pulse*), который вначале переходит на все информационные входы первых мультиплексоров. Далее сигналы распространяются по двум симметричным путям и в конце результата гонки сигналов защелкиваются арбитром. На выходе *Response* могут быть устоявшиеся значения – 0 или 1, а также метастабильные состояния [5]. На основе стабильных ответов *Response* можно сформировать последовательность произвольной длины (вектор откликов) и использовать ее в качестве идентификатора ИС [6].

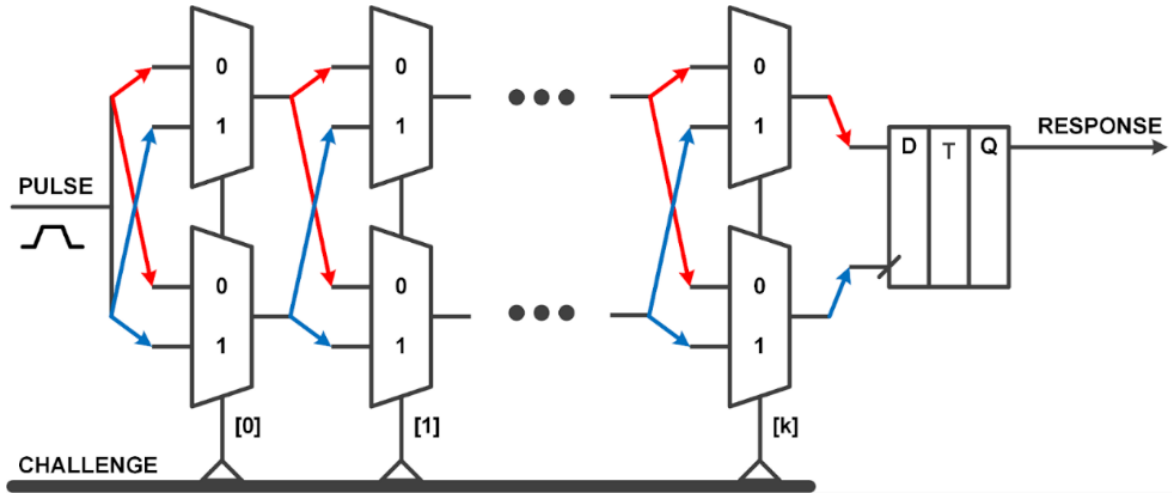

Рисунок 1. ФНФ типа арбитр

Постановка задачи

Для исследования поведения ФНФ и поиска уникальных идентификаторов ИС необходимо дополнительное окружение (тестовый стенд) в виде управляющей системы, реализованной в базисе программируемой логической интегральной схемы (ПЛИС), которая осуществляет обмен данными с персональной электронной вычислительной машиной (ПЭВМ).

Рассмотрены работы, в которых реализованы системы, осуществляющие анализ поведения ФНФ. В [5-11] в качестве экспериментальной установки на базе ПЛИС предложено использование софт-процессора MicroBlaze в САПР Vivado, входящего в состав САПР в виде IP-блока. В [12-13] исследование характеристик ФНФ производилось на базисе микросхемы Zynq, в которой в качестве управляющей стороны выступает аппаратная процессорная система. В работе [14] представлена схема тестового стенда, который лучше всего подходит для исследования ФНФ в разных кристаллах ПЛИС, но главными недостатками которой является малая степень описания системы и ее сложность.

Анализ публикаций показал, что в основном предлагаемые стенды для исследования работы ФНФ специализируются под конкретного производителя микросхемы ПЛИС и соответствующего к ней САПР. Поэтому необходимо реализовать универсальную архитектуру системы для анализа ФНФ. Кроме того, в рассмотренных работах отсутствует программное обеспечение, которое ускоряет процесс исследования.

Таким образом, поставлена задача разработать универсальную программно-аппаратную систему для исследования ФНФ, реализованной в объеме кристалла ПЛИС, с целью удобного извлечения данных для последующего анализа. В статье будет рассмотрен пример построения системы для исследования ФНФ типа арбитр.

Аппаратная система для исследования ФНФ

Предложена структурная схема стенда для анализа и поиска уникальных идентификаторов ФНФ, которая представлена на рис. 2. В качестве основных элементов выступают ПЭВМ, на которой установлена

разработанная авторами программа для анализа ФНФ «PUF Analyzer v1.0», и ПЛИС, на кристалле которой реализована система для работы с ФНФ. Обмен данными между ПЭВМ и ПЛИС осуществляется по протоколу UART через COM-порт, согласно заданному формату команд. В рамках данной статьи система для работы ФНФ взаимодействует с АФНФ.

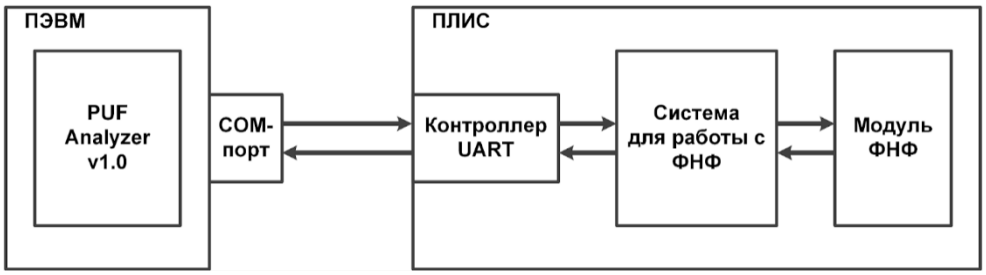

Рисунок 2. Структурная схема стенда для анализа уникальных идентификаторов ФНФ

Протокол UART предназначен для полнодуплексной последовательной связи. UART используется для передачи или приема данных из одной системы в другую. Он применяется там, где нет необходимости в высокоскоростной передаче данных. Упрощенная схема взаимодействия по протоколу UART представлена на рис. 3.

Рисунок 3. Взаимодействие по протоколу UART

Протокол UART организует обмен данными по двум линиями данных – передаваемые данные (TX) и принимаемые данные (RX). Обмен данными происходит с помощью кадров данных, структура которых представлена на рис. 4. Биты данных в составе кадра передаются и принимаются, начиная с младшего.

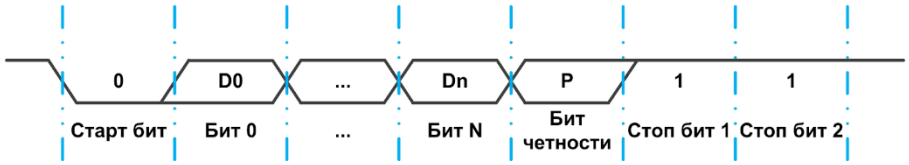

Рисунок 4. Формат кадра UART

Обмен данными начинается со стартового бита, сообщающего начало кадра данных. Значение стартового бита равно «0». После стартового бита идут биты данных, количество которых обычно равно 8. Когда все биты данных переданы, идет бит четности (при наличии). Кадр данных закрывается стоповыми битами, которые сообщают о конце кадра данных. Возможное количество стоповых битов – 1, 1.5 и 2. Возможны следующие варианты проверки четности:

- None – контроль четности не осуществляется, бит четности отсутствует в составе кадра;
- Odd – устанавливает бит четности так, чтобы число установленных битов всегда было нечетным;
- Even – устанавливает бит четности так, чтобы число установленных битов всегда было четным;
- Mark – оставляет бит четности равным 1;
- Space – оставляет бит четности равным 0.

Контроллер UART разрабатывается согласно выбранному формату кадров, с помощью которых осуществляется взаимодействие по UART.

Система для работы с АФНФ, реализованная на кристалле ПЛИС, поддерживает команды согласно заданному формату. В конкретной реализации система поддерживает три типа команд:

1. Проход диапазона от начального (включительно) до последнего значения (не включительно), где значение запроса *Challenge* равно текущему значению из диапазона [STR_RANGE: END_RANGE).
2. Проход диапазона от начального (включительно) до последнего значения (не включительно), где значение запроса *Challenge* имеет фиксированное значение.
3. Останов поданной команды.

Первый и второй тип команды имеют форматы, представленные на рис. 5–6.

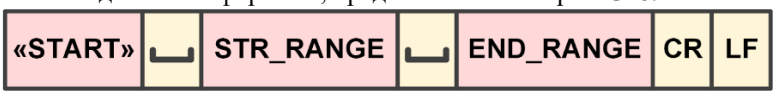

Рисунок 5 – Формат команды (Тип №1)

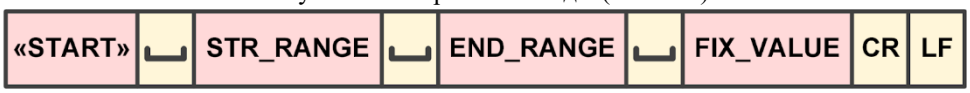

Рисунок 6. Формат команды (Тип №2)

Форматы команды состоят из следующих аргументов:

- Ключевое слово «START», которое допускает нижний или верхний регистр символов;
- Аргумент STR_RANGE – начальное значение диапазона (0-9, A-F, a-f);
- Аргумент END_RANGE – последнее значение диапазона (0-9, A-F, a-f);
- Аргумент FIX_VALUE (при наличии) – фиксированное значение запроса *Challenge* (0-9, A-F, a-f).

Разрядность STR_RANGE, END_RANGE и FIX_VALUE соответствует разрядности запроса *Challenge* и в рамках тестового стенда он имеет разрядность 32 бита.

Ключевое слово команды и аргументы разделяются символами пробела.

Третий тип команды имеет формат, представленный на рис. 7.

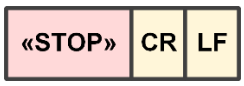

Рисунок 7. Формат команды (Тип №3)

В конце каждой записи идут специальные символы разделители – возврата каретки (CR) и переноса на новую строку (LF). Специальные символы предназначены для разделения поданных команд.

Формат результата, передаваемого системой для работы с АФНФ, представлен на рис. 8.

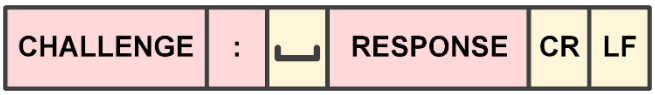

Рисунок 8. Формат результата

Разрядность *Response* в рамках тестового стенда составляет 32 бита.

Первый тип команды предназначен для прохода по заданному диапазону с целью выявления потенциальных идентификаторов. Второй тип команды предназначен для проверки потенциального идентификатора на стабильность, с целью исключения возможного попадания в метастабильное состояние арбитра. Третий тип команды необходим для остановки выполнения команды с неверным выбранным диапазоном или если АФНФ не выдает потенциальные идентификаторы.

Архитектура системы для работы с ФНФ, которая реализована в базисе ПЛИС, изображена на рис. 9.

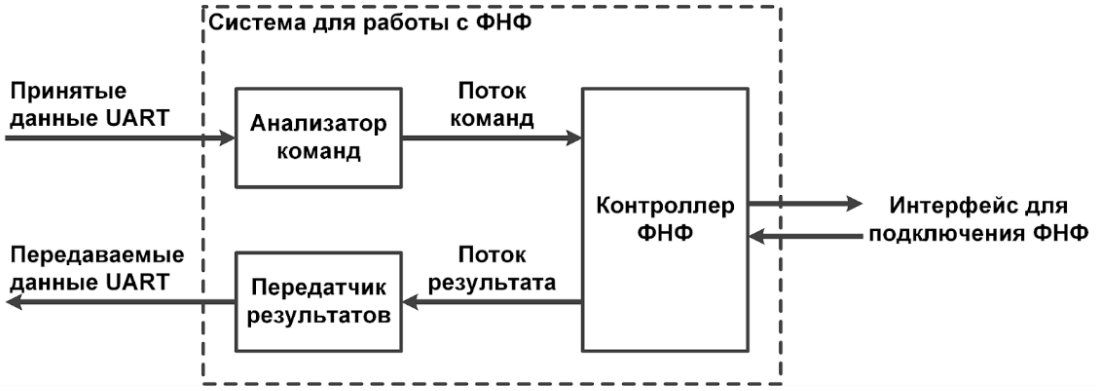

Рисунок 9. Система для работы с ФНФ

Архитектура системы состоит из следующих основных сложных функциональных блоков:

– Анализатор команд – анализирует поданные команды с контроллера UART;

– Контроллер ФНФ – осуществляет опрос подключаемого модуля ФНФ (с целью получения идентификаторов) и отправляет полученные данные на передатчик результатов.

– Передатчик результатов – передает полученные результаты на контроллер UART.

Для взаимодействия с контроллерами UART и ФНФ анализатор команд имеет интерфейс, представленный на рис. 10.

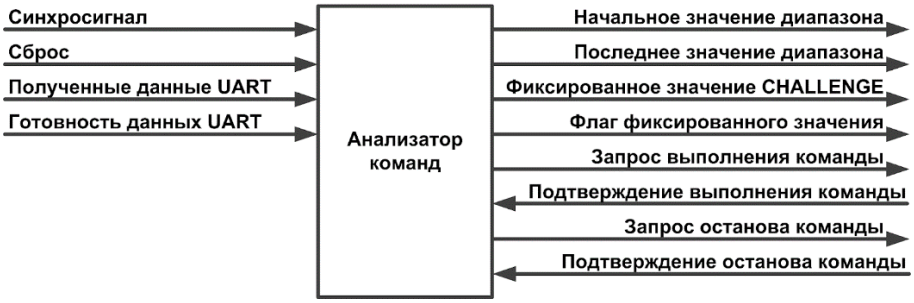

Рисунок 10. Интерфейс модуля анализатора команд

Для взаимодействия с контроллером UART используются следующие сигналы: полученные данные UART и их готовность, которые передаются по любому протоколу.

Для обмена данными с контроллером ФНФ используются следующие шины и сигналы: начальное и последнее значение диапазона – аргументы команды STR_RANGE и END_RANGE (рис. 5-6); фиксированное значение *Challenge* – аргумент команды FIX_VALUE (рис. 6); флаг фиксированного значения – подана команда соответствующая формату, представленному на рис. 6; запрос выполнения команды – готовность данных, поступающих на контроллер ФНФ; подтверждение выполнения команды – готовность контроллера ФНФ принять данные анализатора команд; запрос останова – сообщает контроллеру ФНФ, что необходимо остановить выполняемую команду; подтверждение останова команды – сообщает анализатору, что контроллер ФНФ завершил выполнение команды

Графы переходов конечного автомата анализатора команд представлены на рис. 11-13. Первая часть (рис. 11) отвечает за распознавание ключевых слов команд и передачу команды «STOP» на контроллер ФНФ. После сигнала сброса происходит инициализация внутренних регистров и автомат переходит в начальное состояние, после чего распознает ключевые слова «STOP» и «START». Вторая часть графа (рис. 12) отвечает за распознавание аргументов STR_RANGE и END_RANGE команды «START». Третья часть графа (рис. 13) переходов отвечает за распознавание аргумента FIX_VALUE команды «START» и передачу команды «START». Описание основных состояний конечного автомата анализатора команд представлено в табл. 1.

Таблица 1 – Описание основных состояний конечного автомата анализатора команд

Название состояние	**Описание**

IDLE	Начальное состояние
STRR	Распознавание аргумента STR_RANGE
ENDR	Распознавание аргумента END_RANGE
FIXC	Распознавание аргумента FIX_VALUE
ISTR	Начало передачи команды START
WSTR	Ожидание передачи команды START
ISTP	Начало передачи команды STOP
WSTP	Ожидание передачи команды STOP
SP0 – SP4	Ожидание пробелов
Остальные (S, T1, O, P …)	Ожидание символов, соответствующих ключевым словам команды

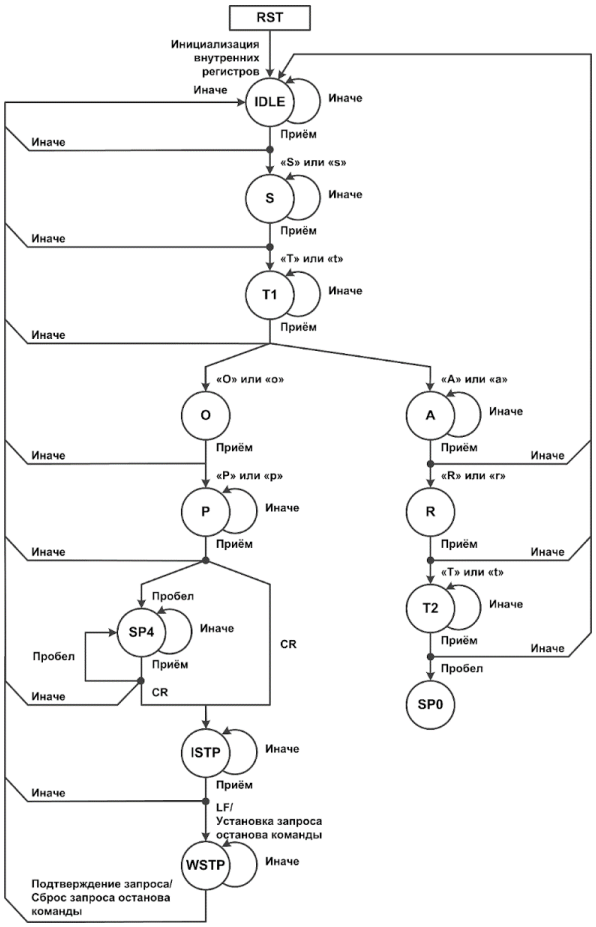

Рисунок 11. Граф переходов конечного автомата анализатора команд (Часть 1)

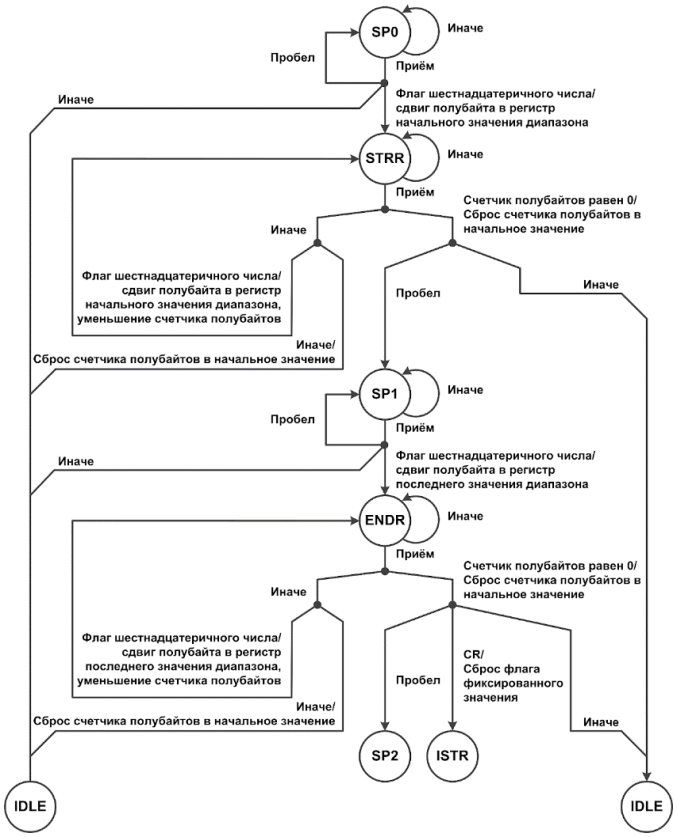

Рисунок 12. Граф переходов конечного автомата анализатора команд (Часть 2)

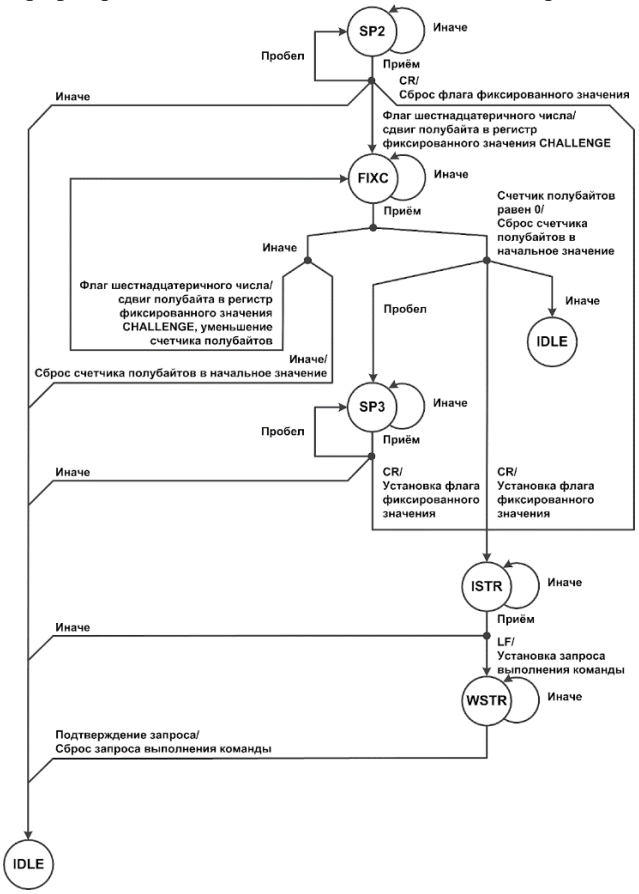

Рисунок 13. Граф переходов конечного автомата анализатора команд (Часть 3)

Для взаимодействия с анализатором команд, модулем ФНФ и передатчиком результатов контроллер ФНФ имеет интерфейс, представленный на рис. 14.

Для взаимодействия с анализатором команд используются сигналы, которые были описаны выше.

Для взаимодействия с модулем ФНФ используются следующие шины и сигналы: импульс – необходим для реализации гонки сигналов по симметричным путям линии задержки АФНФ (для других типов ФНФ назначение данного сигнала может быть иным); значение *Challenge* – запрос; разрешение синхронизации *Challenge* – разрешение загрузки или сдвига регистра запроса (в зависимости от конкретной реализации); разрешение синхронизации *Response* – разрешение загрузки или сдвига регистра вектора откликов (в зависимости от конкретной реализации); значение *Response* – вектор откликов.

Для взаимодействия с передатчиком результатов используются следующие шины и сигналы: запрос передачи данных – готовность контроллера ФНФ передать данные; полученные данные – данные, поступающие от контроллера ФНФ на передатчик результатов; подтверждение передачи данных – готовность передатчика результатов принять данные.

Рисунок 14. Интерфейс модуля контроллера ФНФ

Граф переходов конечного автомата контроллера ФНФ представлен на рис. 15. Описание основных состояний конечного автомата контроллера ФНФ представлено в табл. 2.

Таблица 2 – Описание основных состояний конечного автомата контроллера ФНФ

Название состояние	**Описание**
IDLE	Начальное состояние
RCV	Прием данных от анализатора команд
VER	Проверка начального и последнего значения диапазона
PUF	Сбор данных ФНФ
EPUF	Проверка конца сбора данных ФНФ
TRF	Ожидание передачи полученных результатов

Длительность импульса составляет 1 такт синхросигнала. Для обеспечения стабильности получаемого вектора отклика *Response* между импульсами реализуется задержка, необходимая для того, чтобы дождаться завершения переходных физических процессов в арбитре (в случае метастабильности). Задержка осуществляется в состоянии «PUF» и в рамках тестового стенда составляет 64 такта синхросигнала.

Реализованный модуль ФНФ имеет сдвиговый регистр LFSR [15], в который загружается значение запроса *Challenge*. Следующие 32 такта (в зависимости от разрядности регистра вектора откликов *Response*) с LFSR поступают различные значения запроса на АФНФ. Такой подход позволяет получать уникальные отклики *Response*, от одного загружаемого значения *Challenge*, поступающего с контроллера ФНФ. Аналогично для обеспечения стабильности реализуется задержка после сдвига значения LFSR. Задержка осуществляется в состоянии «EPUF» и в рамках тестового стенда составляет 64 такта синхросигнала.

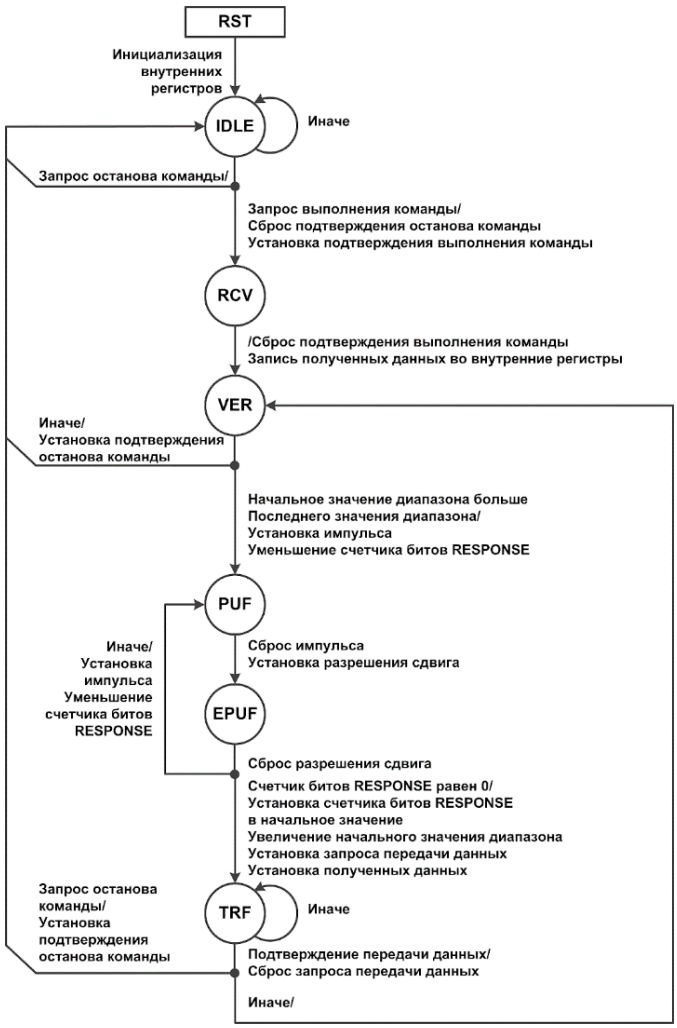

Рисунок 15. Граф переходов конечного автомата контроллера ФНФ

Передатчик результатов имеет интерфейс, изображенный на рис. 16. Для взаимодействия с контроллером ФНФ используется сигналы, которые были описаны выше.

Для взаимодействия с контроллером UART используются следующие сигналы: запрос передачи UART – готовность передатчика результата передать данные; передаваемые данные UART – данные, которые идут на контроллер UART; подтверждение передачи UART – готовность контроллера UART принять данные.

Рисунок 16. Интерфейс модуля передатчика результатов

Граф переходов конечного автомата передатчика результатов представлен на рис. 17. Описание основных состояний конечного автомата передатчика результатов представлено в табл. 3.

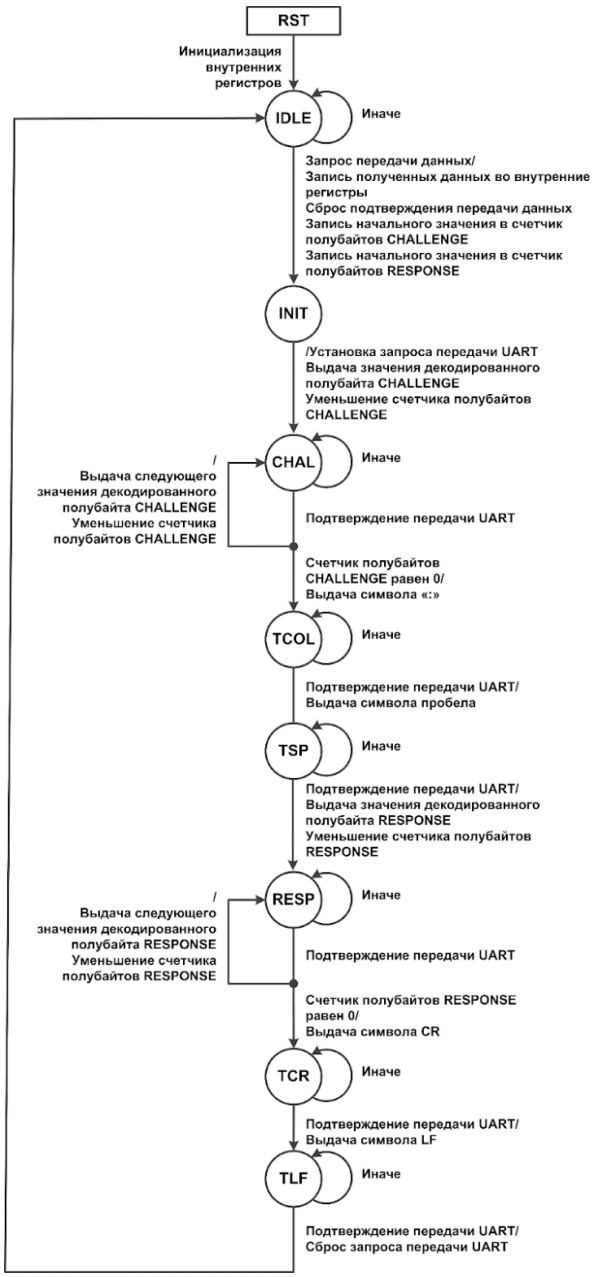

Рисунок 17. Граф переходов конечного автомата передатчика результатов

Таблица 3 – Описание основных состояний конечного автомата передатчика результатов

Название состояние	**Описание**
IDLE	Начальное состояние
INIT	Инициализация передачи данных на UART
CHAL	Передача данных запроса Challenge
TCOL	Передача символа «:»
TSP	Передача символа пробела
RESP	Ожидание передачи полученных результатов
TCR	Передача символа CR
TLF	Передача символа LF

Основные этапы и нюансы при проектировании модуля АФНФ в базисе ПЛИС, такие как возникновение возможных ошибок при синтезе, размещение АФНФ на кристалле ПЛИС, формирование файлов проектных ограничений и т.д., рассмотрены авторами данной статьи в работе [6].

Программная система для исследования ФНФ

Для исследования и анализа ФНФ в рамках тестового стенда необходимы следующие виды операций:

1. Сбор данных ФНФ.
2. Анализ собранных данных ФНФ на потенциальные идентификаторы.
3. Анализ полученных потенциальных идентификаторов на эталонные с целью исключения возможных метастабильных и зашумленных откликов *Response*.

Разработано ПО «PUF Analyzer v1.0» для ПЭВМ на языке программирования высокого уровня C#, в котором были реализованы заданные операции. Для начала сбора данных необходимо настроить профиль COM-порта (рис. 18), согласно выбранным параметрам передачи данных по протоколу UART. Так как в предложенных форматах команд символами-разделителями являются CR+LF, то это необходимо учесть при редактировании профиля.

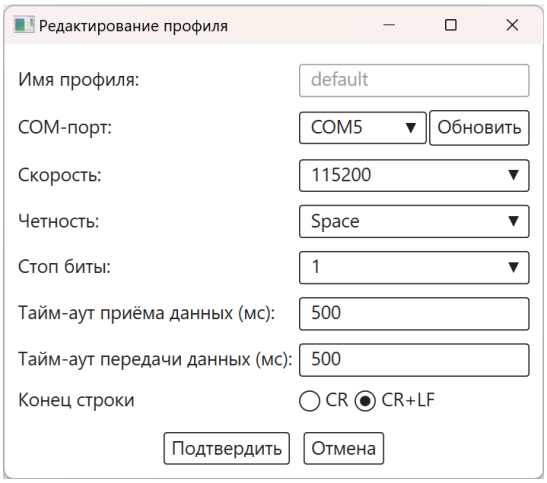

Рисунок 18. Диалоговое окно «Редактирования профиля»

Затем необходимо указать количество выборок, начальное и конечное значение диапазона запроса *Challenge* в шестнадцатеричной системе счисления. От количества выборок будет зависеть сколько раз будет повторяться подача значений из заданного диапазона *Challenge*. Для каждой отдельной выборки создается свой текстовый файл.

Так как изначальной задачей является поиск потенциальных идентификаторов, поэтому пункт «Фиксированное значение Challenge» не отмечен.

Затем указывается имя файла и выходная директория. Если выборок было несколько, то имя файлов будет различаться между собой, согласно заданному формату «имя файла_N.txt», где N – номер выборки. Например, если количество выбор равно 2 и имя файла «crp», то имена файлов с собранными данными будут crp_0.txt и crp_1.txt.

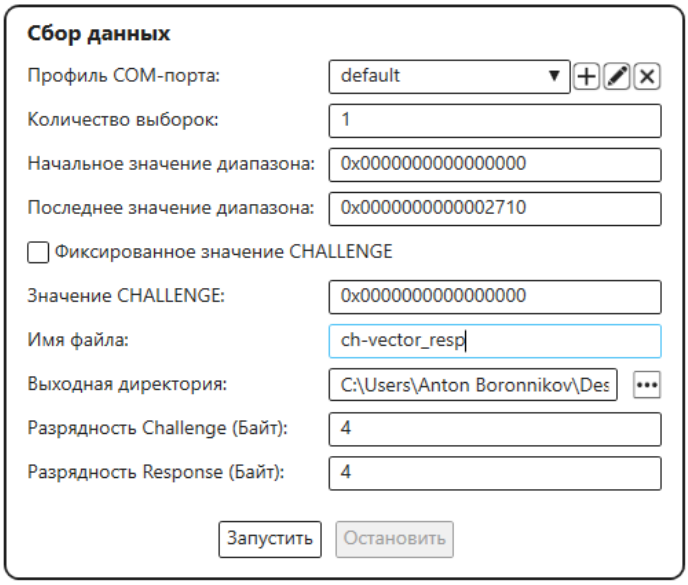

Рисунок 19. Настройки вкладки «Сбор данных»

Система для работы с ФНФ, и сама ФНФ, может масштабироваться, поэтому необходимо следующим шагом указать разрядность *Challenge* и *Response* в байтах. После необходимо нажать кнопку «Запустить», после чего в нижнем поле будет отображаться информация о прогрессе выполнения операции. После завершения операции в указанной выходной директории появяться файлы с результатами CRP, пример содержимого файла изображен на рис. 20 (формат «Challenge: Response»).

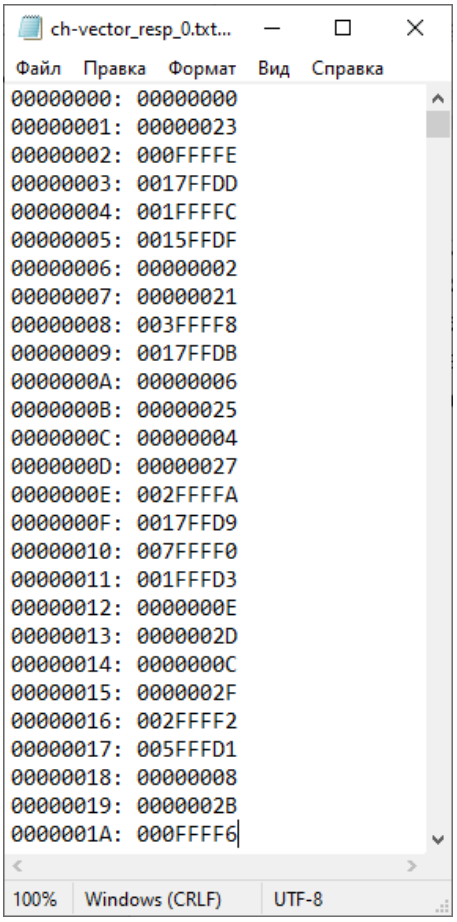

Рисунок 20. Содержимое файла с собранными данными

Общая блок-схема алгоритма сбора данных ФНФ представлена на рис. 21.

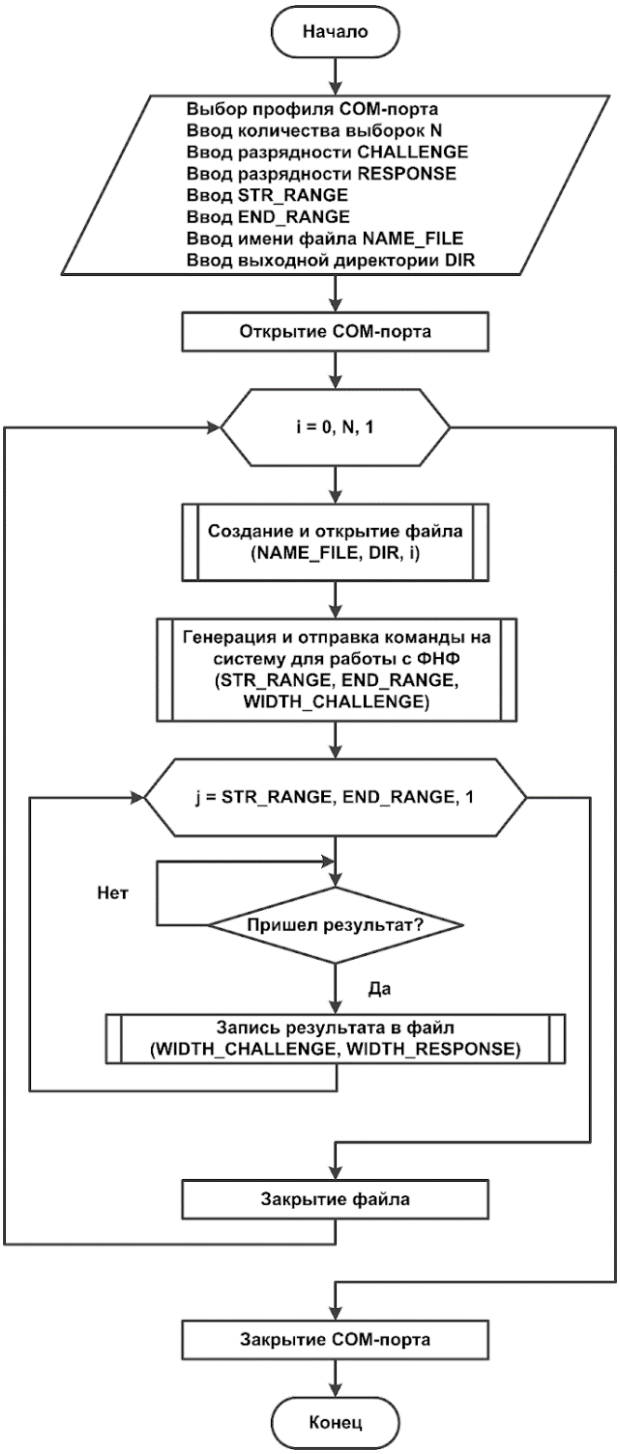

Рисунок 21. Блок-схема алгоритма сбора данных

Из множества полученных данных от ФНФ необходимо определить потенциальные идентификаторы. Для этого во вкладке «Анализ данных» выбирается тип операции «Анализ первичных идентификаторов», где затем устанавливаются количество выборок и входная директория с файлами собранных данных. Аналогично указывается разрядность *Challenge* и *Response*. Затем необходимо указать в профиле исключающих значений те векторы откликов, которые не подходят в качестве идентификаторов (например, вектор из всех нулей или единиц). В общем и целом, исследователь должен сам принимать решение о том, какие значения не соответствуют потенциальным идентификаторам. Результаты анализа сохраняются в файл, которому необходимо также указать имя и выходную директорию. Пример настройки вкладки «Анализ данных» представлен на рис. 22.

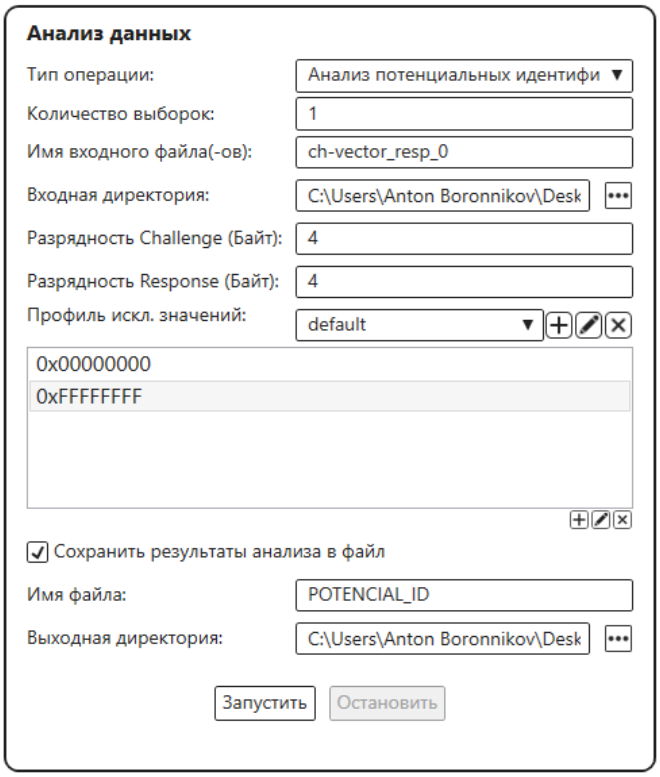

Рисунок 22. Настройки вкладки «Анализ данных» для анализа потенциальных идентификаторов

После необходимо нажать кнопку «Запустить», после чего в нижнем поле будет отображаться информация о прогрессе выполнения операции. После завершения операции в указанной выходной директории появятся файлы с результатами анализа потенциальных идентификаторов, пример которого изображен на рис. 23 (формат «Response: кол-во повторяющихся значений {перечисление Challenge}»).

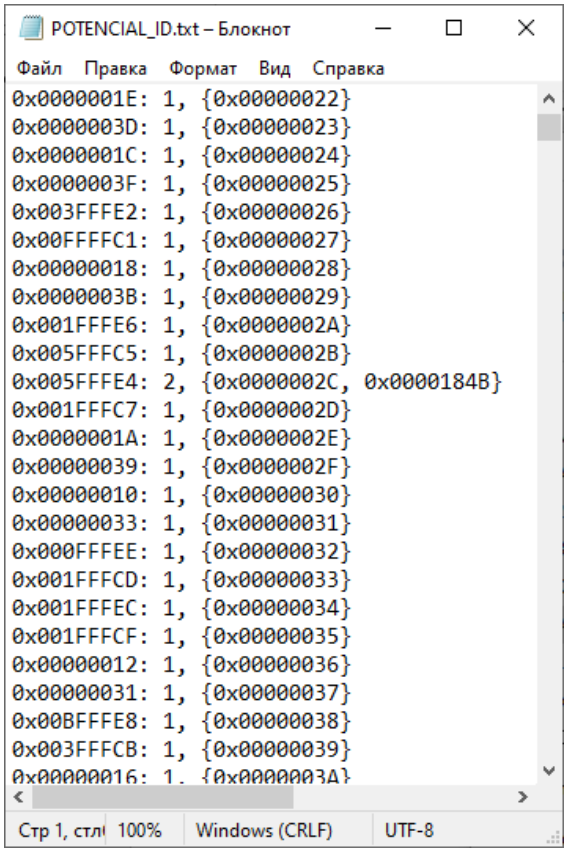

Рисунок 23. Содержимое файла с результатами анализа потенциальных идентификаторов

Общая блок-схема алгоритма анализа потенциальных идентификаторов ФНФ представлена на рис. 24.

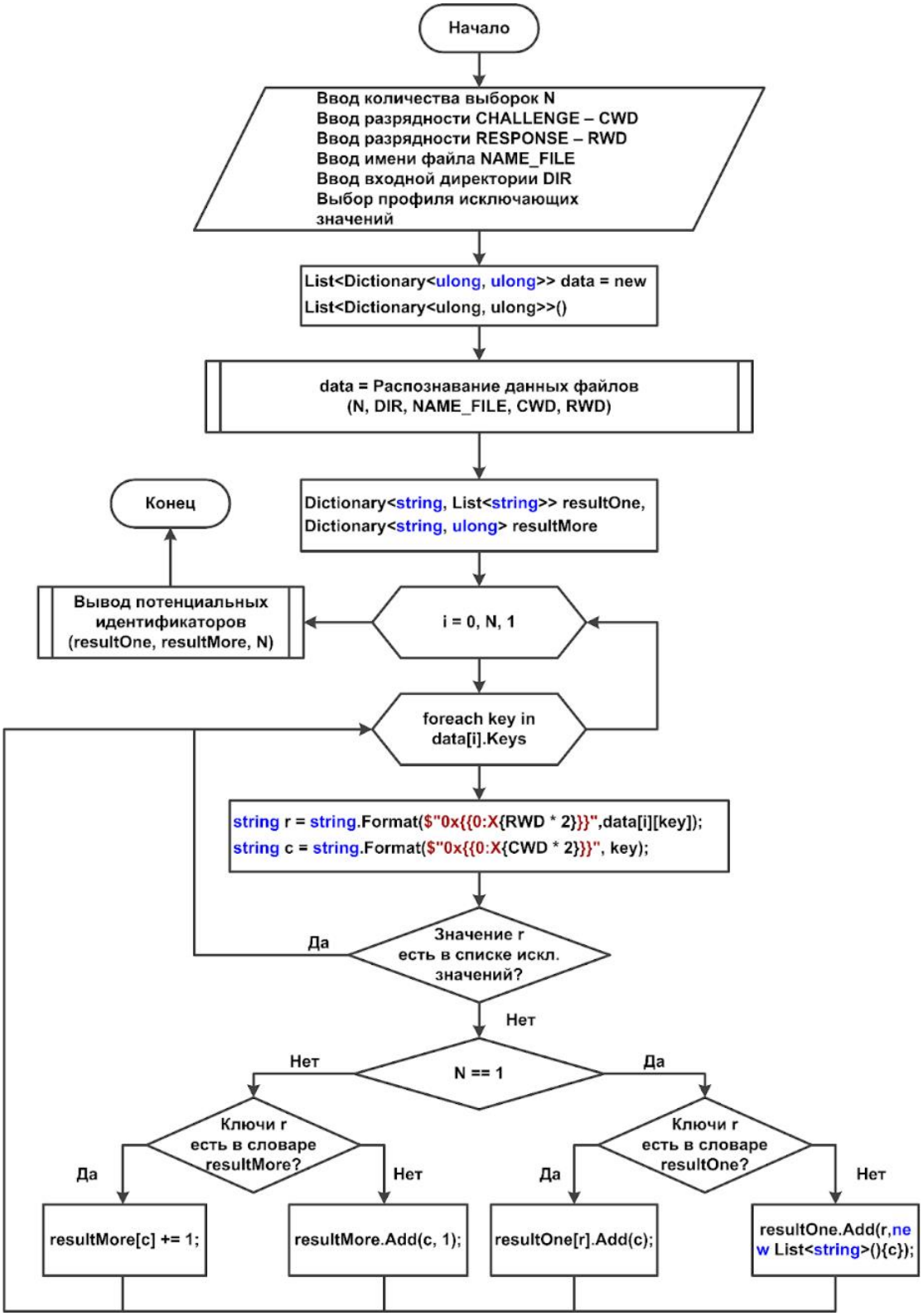

Рисунок 24. Блок-схема алгоритма анализа потенциальных идентификаторов

Из файла потенциальных идентификаторов исследователь должен выбрать подходящие для его задач вектора откликов, но эти вектора могут не являться идентификаторами. Поэтому следующим этапом выполняется сбор данных уже для фиксированного значения *Challenge*. Указываемый диапазон в данном режиме является уже

количеством передачи в систему с ФНФ фиксированного значения *Challenge*. Пример содержимого файла изображен на рис. 25 (формат «фиксированный Challenge: Response»).

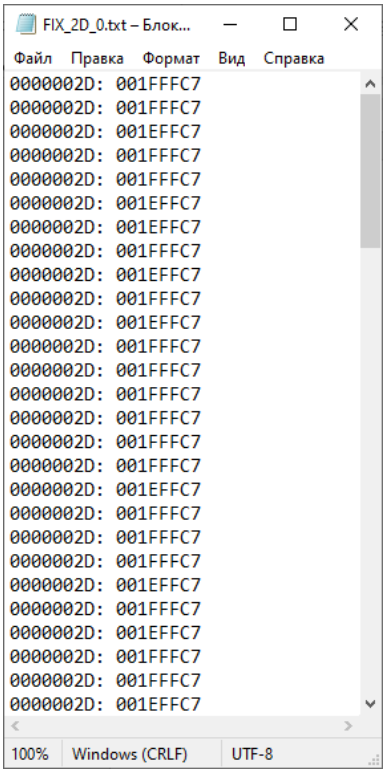

Рисунок 25. Содержимое файла с фиксированным Challenge

В связи с тем, что арбитр ФНФ может попасть в метастабильное состояние, потенциальный идентификатор может быть неверным, поэтому следующим этапом необходимо выяснить является ли выбранный исследователем вектор откликов эталонным идентификатором или нет. В вкладке анализа данных (рис. 26) нужно задать тип операции «Вычисление эталонного идентификатора».

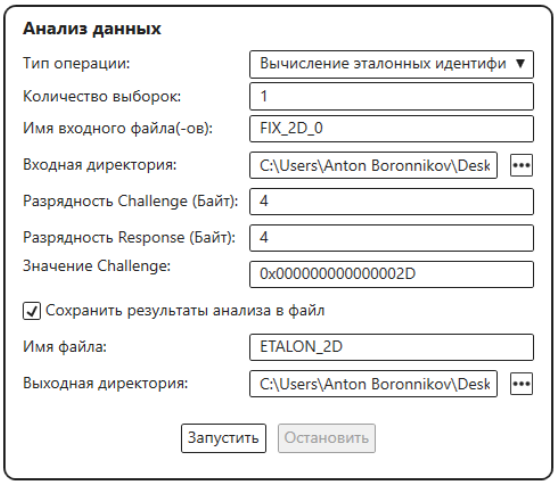

Рисунок 26. Настройки вкладки «Анализ данных» для вычисления эталонного идентификатора

Заполнив все данные и запустив анализ, программа формирует результат, пример которого изображен на рис. 27. Вычисление эталонного ключа происходит анализом каждого бита вектора отклика путем подсчета количества нулей и единиц. Затем усредненным способом вычисляется результат. В этом примере в бите №16 наблюдается метастабильность в арбитре ФНФ.

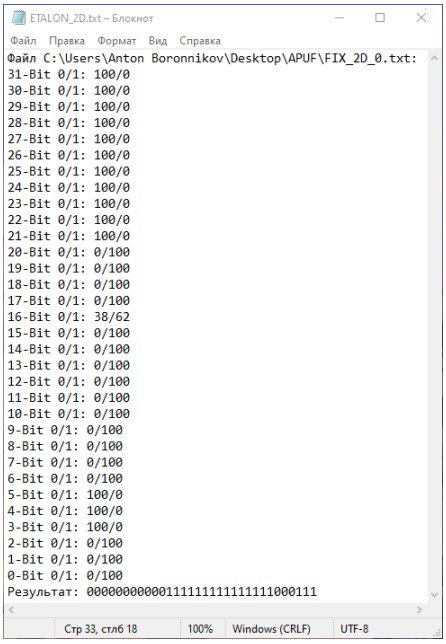

Рисунок 27. Содержимого файла с результатом вычисления эталонного идентификатора

Общая блок-схема алгоритма вычисления эталонного идентификатора ФНФ представлена на рис. 28.

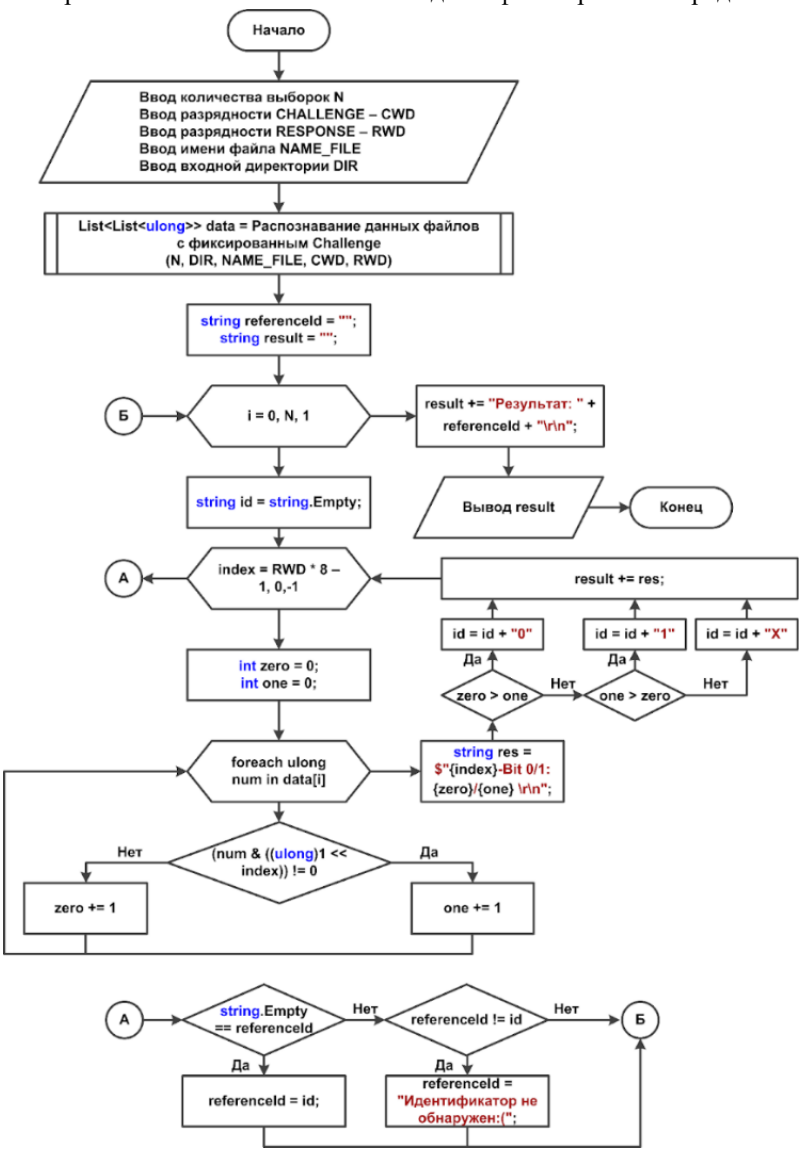

Рисунок 28. Блок-схема алгоритма вычисления эталонного идентификатора

Заключение

В работе рассматривается создание программно-аппаратной системы для исследования ФНФ, реализованной в кристалле ПЛИС. Предложена универсальная схема стенда для сбора и анализа идентификаторов ФНФ, где в качестве основных блоков выступает ПЭВМ и ПЛИС. Обмен данными между ними осуществляется по протоколу UART. В базисе ПЛИС реализованы три функциональных блока: контроллер UART, система для работы с ФНФ и модуль ФНФ. В качестве исследуемого модуля рассмотрен ФНФ типа арбитр. Особенностью архитектуры является универсальность, то есть блок можно реализовать на любом кристалле ПЛИС без привязки к конкретному производителю.

Система для работы с АФНФ поддерживает три типа команд, формат которых задается разработчиками. В рамках данной работы определены две команды прохода диапазона, где в одной значение запроса *Challenge* равно текущему значению из диапазона, а в другой имеет фиксированное значение. Третьей командой является команда останов. Первый тип команды предназначен для прохода по заданному диапазону с целью выявления потенциальных идентификаторов. Второй тип команды предназначен для проверки потенциального идентификатора на стабильность с целью исключения возможного попадания в метастабильное состояние арбитра. Третий тип команды необходим для остановки выполнения команды с неверным выбранным диапазоном или если АФНФ не выдает потенциальные идентификаторы. Предложенная архитектура системы для работы с ФНФ состоит из следующих основных функциональных блоков: анализатор команд, контроллер ФНФ и передатчик результатов. Их функционирование подробно описано с помощью графов конечных автоматов. В работе выявлено, что для корректного заполнения вектора откликов *Response* откликами от работы ФНФ необходимы задержки между подачами импульсов и запросов *Challenge* в АФНФ минимум в 64 такта синхросигнала.

Разработано программное обеспечение «PUF Analyzer v1.0» для ПЭВМ, в котором реализованы операции, необходимые для удобного исследования ФНФ с целью поиска эталонных идентификаторов ИС: сбор данных, анализ полученных данных ФНФ на наличие потенциальных идентификаторов и анализ полученных потенциальных идентификаторов на эталонные с целью исключения возможных метастабильных и зашумленных откликов *Response*.

Полученные результаты демонстрируют успешную реализацию системы, которая способна эффективно собирать, анализировать и сохранять данные работы ФНФ. Предложенные подходы к реализации тестового стенда могут быть полезными при исследовании работы ФНФ в области идентификации и аутентификации устройств, генерации псевдослучайных числовых последовательностей, протоколов аппаратной криптографии и др. Дальнейшее развитие системы направлено на расширение её функциональности, улучшение производительности и поддержку новых видов ФНФ.

Работа выполнена при поддержке Министерства науки и высшего образования РФ (Государственное задание для университетов № ФГФЗ-2023-0005).

Список литературы

1. Herder Ch., Yu M-D., Koushanfar F., Devadas S. Physical Unclonable Functions and Applications: A Tutorial // Proceedings of the IEEE, 2014. — Vol. 102, № 8. — P. 1126-1141.

2. Бельский В.С., Чижов И.В., Чичаева А.А., Шишкин В.А. Физически неклонируемые функции в криптографии // International Journal of Open Information Technologies, 2020. — Т. 8, № 10. — С. 10-26.

3. Комлева Е.Р., Никифоров М.Б. Физически неклонируемые функции. Проблемы и перспективы. // Известия Тульского государственного университета. Технические науки, 2021. — № 6. — С. 61-69.

4. Алиев Г.С. Разработка метода идентификации устройств интернета вещей с использованием физически неклонируемых функций: дис. … канд. техн. наук. — Ставрополь, 2021 — С. 73-81.

5. Лебедев В.Р., Певцов Е.Ф., Деменкова Т.А., Малето М.И., Филимонов В.В. Метод исследования реализации физически неклонируемых функций в информационных системах // International Journal of Open Information Technologies. — 2024. — Т. 12, № 1. — С. 28-36.

6. Боронников А.С., Деменкова Т.А. Методика проектирования уникальных идентификаторов интегральных микросхем на FPGA // Фундаментальные, поисковые, прикладные исследования и инновационные проекты. Сборник трудов Национальной научно-практической конференции, 2023. — С. 6-11.

7. Пучков А.В. Исследование физически неклонируемых функций с использованием софт-процессора MicroBlaze // Компьютерные системы и сети: материалы 52-й научной конференции аспирантов, магистрантов и

студентов, Минск, 25–30 апреля 2016 г. / Белорусский государственный университет информатики и радиоэлектроники; редкол.: В. А. Прытков [и др.]. — Минск, 2016. — С. 40-41.

8. Ахмед А.Н. Аппаратно-программное средство тестирования физически неклонируемых функций // Компьютерные системы и сети: сборник статей 59-й научной конференции аспирантов, магистрантов и студентов, Минск, 17–21 апреля 2023 г. / Белорусский государственный университет информатики и радиоэлектроники. — Минск, 2023. — С. 130-132.

9. Лебедев В.Р., Певцов Е.Ф. Реализация физически неклонируемой функции на ПЛИС // Перспективные материалы и технологии (ПМТ-2023). Сборник докладов Национальной научно-технической конференции с международным участием Института перспективных технологий и индустриального программирования РТУ МИРЭА; редкол.: А.Н. Юрасов [и др.]. — Москва, 2023. — Т. № 1. — С. 365-372.

10. Шамына А.Ю. Анализ характеристик физически неклонируемой функции типа арбитр на FPGA Artix-7 // Компьютерные системы и сети: сборник статей 57-ой научной конференции аспирантов, магистрантов и студентов, Минск, 19-23 апреля 2021 г. / Белорусский государственный университет информатики и радиоэлектроники. — Минск, 2021.— С. 78-81.

11. Шамына А.Ю., Иванюк А.А. Исследование временных параметров физически неклонируемой функции типа арбитр с использованием кольцевого осциллятора // Цифровая трансформация, 2022. — Т. 28, № 1. — С. 27-38.

12. Ярмолик В.Н., Иванюк А.А. Двухмерные физически неклонируемые функции типа арбитр // Информатика, 2023. — Т. 20, № 1. — С. 7–26.

13. Ярмолик В.Н., Иванюк А.А. Сбалансированные физически неклонируемые функции типа арбитр // Безопасность информационных технологий, 2023. — Т. 30, № 1. — С. 92–107.

14. Заливако С.С., Иванюк А.А. Физически неклонируемые функции // Информационные технологии и системы 2019 (ИТС 2019): материалы международной научной конференции, Минск, 30 октября 2019 г. / Белорусский государственный университет информатики и радиоэлектроники; редкол.: Л. Ю. Шилин [и др.]. — Минск, 2019. — С. 8-21.

15. Принцип работы РСЛОС. https://habr.com/ru/articles/534732/

References

1. Herder Ch., Yu M-D., Koushanfar F., Devadas S. Physical Unclonable Functions and Applications: A Tutorial // Proceedings of the IEEE, 2014. — Vol. 102, № 8. — P. 1126-1141.

2. Belsky V., Chizhov I., Chichaeva A., Shishkin V. Physically Unclonable Functions in cryptography // International Journal of Open Information Technologies, 2020. — Vol. 8, № 10. — P. 10-26.

3. Komleva E.R., Nikiforov M.B. Physical unclonable functions. Problems and prospects. // Izvestia Tula State University. Technical Sciences, 2021. — № 6. — P. 61-69.

4. Aliev G.S. Development of the method of identification of devices of the Internet of Things with the use of physically unclonable functions: thesis ... PhD. — Stavropol, 2021 — P. 73-81.

5. Lebedev V.R., Pevtsov E.Ph., Demenkova T.A., Maleto M.I., Filimonov V.V. Method for studying the implementation of Physical Unclonable Function in information systems // International Journal of Open Information Technologies. — 2024. — Vol. 12, №1. — P. 28-36.

6. Boronnikov A.S., Demenkova T.A. Methodology for designing unique identifiers of integrated circuits on FPGA // Fundamental, search, applied research and innovative projects. Collection of proceedings of the National Scientific and Practical Conference, 2023. — P. 6-11.

7. Puchkov A.V. Investigation of physically unclonable functions using MicroBlaze soft-processor // Computer systems and networks: proceedings of the 52nd scientific conference of graduate students, undergraduates and students, Minsk, April 25-30, 2016 / Belarusian State University of Informatics and Radioelectronics; editor: V. A. Prytkov [et al.]. — Minsk, 2016. — P. 40-41.

8. Akhmed A.N. Hardware and software means of testing physically unclonable functions // Computer systems and networks: collection of articles of the 59th scientific conference of graduate students, undergraduates and students, Minsk, April 17-21, 2023 / Belarusian State University of Informatics and Radioelectronics. — Minsk, 2023. — P. 130-132.

9. Lebedev V.R., Pevtsov E.F. Realization of physically unclonable function on FPGA // Advanced Materials and Technologies (AMT-2023). Proceedings of the National Scientific and Technical Conference with International

Participation of the Institute of Advanced Technologies and Industrial Programming of RTU MIREA; Editor: A.N. Yurasov [and others]. — Moscow, 2023. — Vol. № 1. — P. 365-372.

10. Shamyna A.Y. Analysis of the characteristics of arbiter PUF on FPGA Artix-7 // Computer systems and networks: collection of articles of the 57th scientific conference of postgraduate, graduate and undergraduate students, Minsk, 19-23 April 2021 / Belarusian State University of Informatics and Radioelectronics. — Minsk, 2021.— P. 78-81.

11. Shamyna A.Y., Ivaniuk A.A. Investigation of the Timing Parameters of The Arbiter-Based Physically Unclonable Function Using a Ring Oscillator // Digital Transformation, 2022. — Vol. 28, № 1. — P. 27-38.

12. Yarmolik V.N., Ivaniuk A.A. 2D physically unclonable functions of the arbiter type // Informatics, 2023. — Vol. 20, № 1. — P. 7–26.

13. Yarmolik V.N., Ivaniuk A.A. Balanced arbiter physical uncloneable functions // Information technology security, 2023. — Vol. 30, № 1. — P. 92–107.

14. Zalivako S.S., Ivanyuk A.A. Physically unclonable functions // Information Technologies and Systems 2019 (ITS 2019): proceedings of the international scientific conference, Minsk, October 30, 2019 / Belarusian State University of Informatics and Radioelectronics; editor: L. Y. Shilin [et al.]. — Minsk, 2019. — P. 8-21.

15. Principle of operation of RSLOS. https://habr.com/ru/articles/534732/