СРАВНЕНИЕ АППАРАТНЫХ И ПРОГРАММНЫХ СРЕДСТВ РАСЧЁТА ОСНОВНЫХ ХАРАКТЕРИСТИК ФИЗИЧЕСКИ НЕКЛОНИРУЕМЫХ ФУНКЦИЙ

Трубач К. И., Иванюк А. А.
Белорусский государственный университет информатики и радиоэлектроники
Минск, Республика Беларусь
E-mail: xenona11x@gmail.com, ivaniuk@bsuir.by

Проводится сравнение средств расчета основных характеристик (стабильности, единообразия, меж- и внутрикристальной уникальности) физически неклонируемых функций. Показано, что аппаратная реализация обеспечивает наивысшую скорость при оптимальных затратах ресурсов, Cortex-A9 предоставляет максимальную гибкость при минимальном быстродействии, а софт-процессорное ядро предлагает сбалансированное решение.

Введение

Физически неклонируемая функция (ФНФ) – специальная цифровая схема, которую легко спроектировать и реализовать, но практически невозможно воспроизвести [1]. Это свойство обуславливает широкое применение ФНФ в сферах защиты информации, физической криптографии, а также источниках случайности и системах идентификации.

Пригодность ФНФ к какому-либо из перечисленных направлений оценивается рядом характеристик, к которым относятся стабильность, надёжность, единообразие и меж- и внутрикристальная уникальность [2]. Регулярное измерение этих характеристик целесообразно и на стадии тестирования готовых ФНФ (в составе заказных ИС или реализованных на ПЛИС), поскольку они позволяют вовремя детектировать деградацию кристалла, а также обнаружить атаки как естественные (сюда относятся такие неблагоприятные условия, как колебания внешнего электромагнитного поля или некорретная работа источника питания кристалла), так и преднамеренные с целью компрометировать ФНФ.

При расчёте характеристик для научных целей важна гибкость, а для производственных – быстродействие и экономия ресурсов, поэтому необходимо осознанно подходить к выбору средств расчёта, которых можно выделить четыре: аппаратное процессорное ядро общего назначения Cortex-A9; софт-процессорное ядро общего назначения; софт-процессорное ядро со специализированным набором инструкций; аппаратная реализация на FPGA.

Все эксперименты в рамках данной работы проводились на плате быстрого прототипирования ZyboZ7 с FPGA Zynq7010 с использованием системы Jupyter Notebook и языка Python.

I. Обзор средств

Средство Cortex-A9 предполагает программный код, управляющий ФНФ (*PUF*) через регистровый файл (*RCU*). ФНФ, состоящая из конфигурируемых кольцевых осцилляторов (CRO) и счётчиков, формирует ответ (*Response*) на основе сравнения их показаний в заданном временном окне (*Window*). Описанная схема представлена на рис. 1.

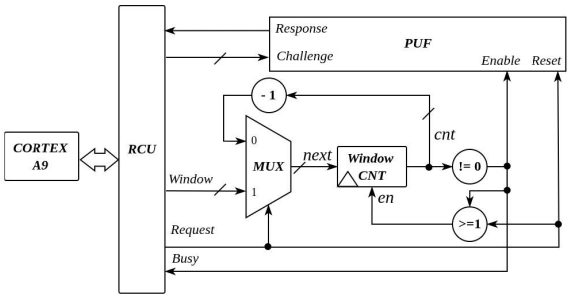

Рис. 1 – Схема ФНФ, контролируемая Cortex-A9

Подход минимизирует аппаратуру, переносся расчёты на процессор общего назначения, и обеспечивает высокую гибкость и скорость разработки. Наиболее подходит для исследовательских работ, так как внедрение процессорного ядра в систему, изначально не предполагавшую его наличие, – это трудозатратный и дорогой процесс. Также ядро управляется языком программирования высокого уровня, что крайне негативно сказывается на общем затраченном времени.

Следующее средство – софт-процессорное микропрограммное ядро общего назначения с базовым набором инструкций. Оно способно работать с ФНФ и легко адаптируемо к любому их количеству. Предлагаемая архитектура, основанная на RISC-V и представленная на рис. 2, получила название *Verity RV* (далее – *VRV*).

Микропрограмма хранится в перепрограммируемом ЗУ (*PROM*). Загрузка инструкций в *PROM* и управление ядром (*MPCU*) осуществляются через интерфейс AXI4 Lite и регистровый файл (*RF*). ОЗУ разделено на две части: одна (*general purpose RAM*) для общего назначения, вторая (*RAM Mapped PUF CSRs*) для работы с

Работа выполнена в совместной учебной лаборатории БГУИР-YADRO
https://www.bsuir.by/ru/kaf-informatiki/yadro

ФНФ ($PUF1$, ..., $PUFn$). Управление ФНФ происходит на языке ассемблера, а Cortex-A9 лишь однократно загружает микрокод и запускает его. Главное преимущество *VRV* – микрокод, обеспечивающий реконфигурацию и гибкость.

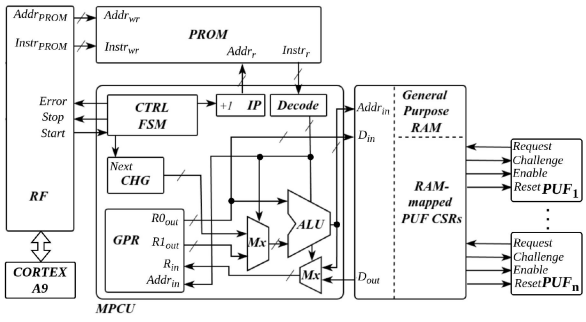

Рис. 2 – Схема софт-процессорного ядра *VRV*

VRV легко адаптируется под нестандартные расширения, что позволяет использовать софт-процессорное ядро со специализированным набором инструкций, созданных для максимального ускорения расчёта характеристик ФНФ и являющихся отдельным средством работы с ними – *VRV-S*. На рис. 3 показаны способ вычленения из алгоритма расчёта стабильности специализированной инструкции под названием *sta_ abs*, а также её аппаратная реализация. Ядро *VRV* требует затрат на разработку, но подходит как для исследований, так и для производства, сочетая гибкость и простоту реализации. Его универсальность выгодно выделяет *VRV* среди средств работы с ФНФ.

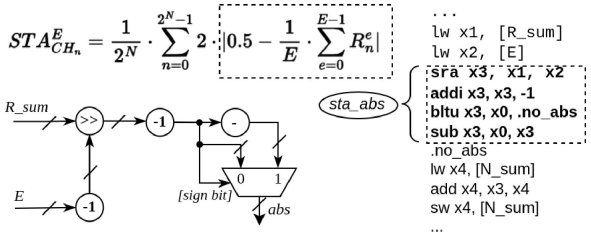

Рис. 3 – Пример получения специализированной инструкции

Последнее рассматриваемое средство — аппаратная реализация на FPGA. Она предполагает HDL-описание некоторой схемы расчёта (см. рис. 4) и её запуск на FPGA.

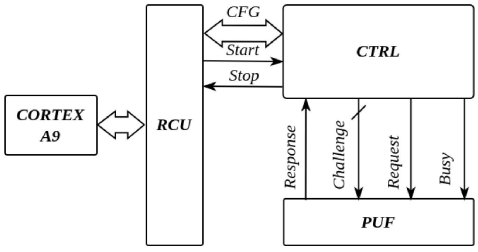

Рис. 4 – Схема аппаратной реализации

Контроль осуществляется через регистровый файл (RCU), включающего сигналы *Start* и *Stop* для сигнализации начала и конца расчёта, а также некоторой шины *CFG*, состав которой может варьироваться в зависимости от алгоритма реализуемой характеристики (например, размер окна измерения, количество итераций и пр.). Сигналы поступают на контрольное устройство *CTRL* с заданной логикой расчёта и прямым доступом к ФНФ. На рис. 4 ФНФ показана в единственном экземпляре для наглядности. Аппаратные схемы расчёта характеристик подходят для производства, не требуя больших затрат ресурсов и обеспечивая наибольшее быстродействие. Однако их переиспользование затруднено, и каждую схему (для каждой характеристики) необходимо разрабатывать и верифицировать отдельно.

II. Сравнение средств и выводы

Для количественного анализа выбрана характеристика стабильности, так как она включает вложенные циклы и требует значительного количества операций, что повышает точность расчёта. Результат показан в табл. 1, где t_{HW} и t_{SW} – время срабатывания аппаратуры и хоста соответственно, t_{SET} – время настройки аппаратуры. Затраты ресурсов для каждого средства не учитывают реализацию ФНФ, но учитывают логику доступа к ним (регистровый файл).

Таблица 1 – Эффективность рассмотренных средств при измерении стабильности ФНФ

Средство	t_{HW}, c	t_{SW}, c	t_{SET}, c	LUT6	FFs
Cortex-A9	429,7	$< 0{,}001$	0,3	324	263
VRV	1	0,2	2,3	1824	1410
VRV-S	1	0,07	2,1	1885	1419
Расчёт на FPGA	1	0,03	0,5	440	327

Таким образом, Cortex-A9 как средство требует минимальных ресурсов FPGA на реализацию ФНФ и доступа к ней, однако демонстрирует наихудшее время расчёта, что делает его пригодным для непроизводительных задач. Софт-процессор *VRV* является значительно более производительным решением, однако требует существенного увеличения ресурсов. Оптимизация *VRV* специализированными инструкциями улучшает скорость с незначительным ростом ресурсов. Наибольшее быстродействие при существенно меньших затратах ресурсов FPGA обеспечивает аппаратная реализация.

III. Список литературы

1. Secure System Design and Trustable Computing / ed.: Ch. H. Chang, M. Potkonjak. – Switzerland : Springer, 2016. – 549 p. DOI: https://doi.org/10.1007/978-3-319-14971-4.
2. Иванюк А. А. Исследование физически неклонируемой функции конфигурируемого кольцевого осциллятора // Информатика. – 2025. – Т. 22. – № 1. – С. 73-89.