УДК 004.4'424

Хлесткин Андрей Юрьевич, канд. техн. наук, доц., Поволжский государственный университет телекоммуникаций и информатики, г. Самара.

Кунбуттаев Магомед-Гаджи Мурадович, бакалавр, Поволжский государственный университет телекоммуникаций и информатики, г. Самара.

Гришин Илья Олегович, бакалавр, Поволжский государственный университет телекоммуникаций и информатики, г. Самара.

АССЕМБЛЕР И КРИПТОГРАФИЯ: РЕАЛИЗАЦИЯ ШИФРОВ И ХЕШ-ФУНКЦИЙ НА НИЗКОМ УРОВНЕ

Работа посвящена применению языка ассемблера в криптографии и преимуществам низкоуровневой оптимизации при реализации шифров и хеш-функций. Показано, что ассемблер повышает производительность алгоритмов за счёт контроля регистров, порядка выполнения инструкций и SIMD-параллелизма. Особое внимание уделено аппаратным расширениям (AES-NI, ARM Crypto Extensions) для ускорения AES и ChaCha20, оптимизации хеш-функций (SHA-256, BLAKE2) и ускорению математических операций для RSA и ECC. Отмечается важность защиты от атак по побочным каналам через constant-time выполнение и минимизацию зависимостей от секретных данных. Ассемблер остаётся ключевым инструментом для высокопроизводительных и безопасных криптографических реализаций.

Ключевые слова: Ассемблер, Криптография, AES, SHA-256, SIMD-оптимизация.

The work focuses on the use of assembly language in cryptography and the advantages of low-level optimization when implementing ciphers and hash functions. It shows that assembly improves algorithm performance through precise control over registers, instruction ordering, and SIMD parallelism. Special attention is given to hardware extensions (AES-NI, ARM Crypto Extensions) for accelerating AES and

ChaCha20, to optimization of hash functions (SHA-256, BLAKE2), and to speeding up mathematical operations used in RSA and ECC. The importance of protecting against side-channel attacks through constant-time execution and minimizing dependencies on secret data is emphasized. Assembly remains a key tool for building high-performance and secure cryptographic implementations.

Keywords: Assembly, Cryptography, AES, SHA-256, SIMD optimization.

Почему ассемблер важен для криптографии

1. Высокая производительность

Ассемблер позволяет управлять распределением данных по регистрам, порядком выполнения инструкций и распараллеливанием операций на SIMD-блоках. Например, в реализации AES можно использовать специализированные инструкции AES-NI:

```
aesenc xmm0, xmm1          ; раундовое преобразование
aesdec xmm0, xmm1          ; раунд дешифрования
```

Использование AES-NI ускоряет алгоритм в 3–10 раз по сравнению с реализацией без аппаратного ускорения.

Для потокового шифра ChaCha20 на ARM оптимизация сводится к эффективной обработке 32-битных слов:

```
eor   w0, w1, w2           ; XOR
ror   w3, w3, #16          ; циклический сдвиг
adds  w4, w4, w5           ; сложение modulo 2^32
```

Такие операции идеально подходят для SIMD-инструкций NEON.

2. Контроль над железом

Ассемблер предоставляет возможность: точно контролировать порядок байтов (endianess), управлять выравниванием данных, вручную настраивать работу с кэшем (prefetch, избегание конфликтов), минимизировать пропуски инструкций из-за прогнозирования ветвлений.

Для криптографии особенно важно избегать переходов, зависящих от секретных данных и исключать обращение к таблицам (табличный AES), которое может привести к утечкам через кеш.

3. Минимизация побочных эффектов

При реализации криптографии важно обеспечивать *constant-time* исполнение — одинаковое время выполнения независимо от входных данных.

Ассемблер позволяет заменить ветвления на арифметические операции, выполнять маскирование вместо условий и контролировать выход в память и предотвращать вариативный доступ.

Пример замены ветвления:

```
// if (x < y) x = y;
cmp    x0, x1
csel   x0, x1, x0, lt     ; выбрать минимальное без ветвления
```

Основные алгоритмы, реализуемые на ассемблере

Симметричные шифры:

AES (Advanced Encryption Standard)

На x86 ускоряется с помощью: AES-NI (aesenc, aesenclast, aeskeygenassist), AVX/AVX2 для параллельной обработки сразу нескольких блоков, AVX-512 — до 8–16 блоков одновременно

На ARMv8 применяются Crypto Extensions:

```
aese v0.16b, v1.16b
aesmc v0.16b, v0.16b
```

ChaCha20

Оптимален для ARM-архитектур, не использующих таблицы. Реализация выигрывает от: операций над 32-битными словами, NEON SIMD и возможности разворачивать 20 раундов в pipeline.

Хеш-функции:

SHA-256

Оптимизация включает использование инструкций rotate (x86: rorx, ARM: ror), хранение всех восьми рабочих регистров (a–h) в физических регистрах и разворачивание компрессионного цикла вручную.

Пример операции:

```
rorx rax, rbx, 7      ; ROTR(x, 7)
xor  rcx, rdx
add  rax, rcx
```

На ARMv8 есть специальные SHA-инструкции (sha256h, sha256su1).

Blake2

Эта хеш-функция активно использует параллельные операции: на x86 — AVX2/AVX-512, на ARM — NEON. Можно обрабатывать до 4–8 блоков BLAKE2 параллельно.

Асимметричные алгоритмы: хотя полная реализация RSA и ECC обычно пишется на C, ключевые операции часто оптимизируются на ассемблере:

RSA

Используются: Montgomery multiplication, Karatsuba и Toom-Cook умножение, SIMD-ускорение для больших чисел.

ECC (эллиптические кривые)

Оптимизируются операции над полями GF(p), сложение точек и удвоение, специальные форматы координат (например, Jacobian).

Для Curve25519 часто ручная сборка даёт 10–20% ускорения.

Особенности реализации на ассемблере

1. Регистры и SIMD

Успешная реализация основывается на том, чтобы: держать как можно больше промежуточных данных в регистрах, устранять лишние загрузки/выгрузки, параллелить обработку блоков.

Например, AVX2 позволяет обрабатывать 4 блока SHA-256 одновременно.

2. Endianess и выравнивание

Ошибка выравнивания может привести к: падению производительности и аппаратным исключениям (на ARM).

Для SHA-256 важно преобразование little-endian ↔ big-endian:

```
rev32 v0.16b, v0.16b
```

3. Минимизация ветвлений

Вместо условий используются: инструкции условного выбора (csel на ARM), битовые маски и заранее развёрнутые циклы. Это уменьшает поверхности атак типа Spectre и атак через кеш.

4. Защита от атак по побочным каналам

Ключевые принципы:

- избегать lookup-таблиц, зависимых от секретных данных (например, табличный AES),
- обеспечить фиксированное время выполнения операций,
- работать только с регистровыми данными при критических операциях,
- применять маскирование (хотя это выше уровнем, чем просто ассемблер),
- снижать утечки через энергопотребление и шину памяти.

Список литературы

1. Armando Faz Hernandez, Julio López. High Performance Elliptic Curve Cryptography: A SIMD Approach to Modern Curves [Электронный ресурс] // CLEI Electronic Journal. – 2024. – URL: https://www.clei.org/cleiej/index.php/cleiej/article/view/633 (дата обращения: 29.11.2025).
2. Pengchang Ren, Reiji Suda, Vorapong Suppakitpaisarn. Efficient Additions and Montgomery Reductions of Large Integers for SIMD [Электронный ресурс] // arXiv. – 2023. – URL: https://arxiv.org/abs/2308.16432 (дата обращения: 29.11.2025).
3. Joel Kuepper и др. CryptOpt: Verified Compilation with Randomized Program Search for Cryptographic Primitives [Электронный ресурс] // arXiv. – 2022. – URL: https://arxiv.org/abs/2211.10665 (дата обращения: 29.11.2025).
4. Ghada F. Elkabbany, Heba K. Aslan, Mohamed N. Rasslan. A Design of a Fast Parallel Pipelined Implementation of AES: Advanced Encryption Standard [Электронный ресурс] // arXiv. – 2015. – URL: https://arxiv.org/abs/1501.01427 (дата обращения: 29.11.2025).
5. Криптографические методы защиты информации. Учебник [Электронный ресурс] // Самарский университет. – URL: https://repo.ssau.ru/bitstream/Uchebnye-izdaniya/Kriptograficheskie-metody-

zashity-informacii-111844/1/987-5-7883-2074-8_2024.pdf (дата обращения: 29.11.2025).

6. Алгоритмы шифрования TEA и XTEA на Ассемблере [Электронный ресурс] // ManHunter.ru. – URL: https://www.manhunter.ru/assembler/1330_algoritmi_shifrovaniya_tea_i_xtea_na_assemblere.html (дата обращения: 29.11.2025).
7. Криптография и хеширование — Учебник Assembler [Электронный ресурс] // nweb42.com. – URL: https://nweb42.com/books/assembler/kriptografiya-i-kheshirovanie/ (дата обращения: 29.11.2025).
8. AES — американский стандарт шифрования. Часть I [Электронный ресурс] // Habr.com. – URL: https://habr.com/ru/articles/508442/ (дата обращения: 29.11.2025).
9. Работа с SHA-256 [Электронный ресурс] // Electromicro.ru. – URL: https://electromicro.ru/resources/wiki/sha256/sha256/ (дата обращения: 29.11.2025).