



**Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)
(МГТУ им. Н.Э. Баумана)**

ФАКУЛЬТЕТ «Информатика и системы управления» (ИУ)
КАФЕДРА «Информационная безопасность» (ИУ8)

Домашнее задание
по курсу «Параллельные вычисления»
на тему «Реализация алгоритмов шифрования и расшифрования для
крипtosистемы HFE»

Выполнили ИУ8-112
(группа) Р.С. Малютин
Б.М. Шереметьев
Н.А. Храпов

Преподаватель _____ Карондеев А. М.
(подпись, дата) _____ (И.О. Фамилия)

2025 ε.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ	3
ОСНОВНАЯ ЧАСТЬ.....	4
Теоритическая часть	4
1. Общая схема многомерных квадратичных криптосистем	4
2. Математический аппарат HFE	4
Практическая часть	7
1. Модуль field_operations	7
2. Модуль hfe_base	9
5. Работа с данными (encrypt_block, decrypt_block)	12
3. Модуль hfe_cpu_parallel	13
4. Модуль hfe_gpu_parallel	15
5. Тестирование и сравнение результатов	18
ЗАКЛЮЧЕНИЕ	20
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ	22
ПРИЛОЖЕНИЕ А	23
ПРИЛОЖЕНИЕ Б	24

ВВЕДЕНИЕ

Криптосистемы с открытым ключом, основанные на многомерных квадратичных уравнениях (Multivariate Public Key Cryptography, MPKC), являются одним из перспективных направлений в постквантовой криптографии. Одной из наиболее известных схем этого класса является HFE (Hidden Field Equations), предложенная Жаком Патарином в 1996 году.[1]

Актуальность данной работы обусловлена необходимостью исследования алгоритмов, устойчивых к атакам с использованием квантовых компьютеров (алгоритм Шора), которые угрожают классическим системам RSA и ECC. HFE базируется на сложности решения систем нелинейных уравнений над конечными полями (задача MQ-problem), которая считается НП-трудной.

Целью данной работы является реализация криптосистемы HFE с использованием методов параллельных вычислений на CPU и GPU для повышения производительности процессов генерации ключей, шифрования и, в особенности, трудоемкого процесса расшифрования.

ОСНОВНАЯ ЧАСТЬ

Теоритическая часть

1. Общая схема многомерных квадратичных крипtosистем

Многомерная криптография или многомерная криптография открытого ключа — это общий термин, описывающий асимметричные криптографические схемы, построенные на решениях уравнений, основанных на многомерных полиномах над конечным полем F .

Безопасность многомерной криптографии основывается на предположении, что решения системы квадратичных многочленов над конечным полем F , в общем случае, является NP-полной задачей в сильном смысле или просто NP-полной. Вот почему эти схемы часто считаются хорошими кандидатами для постквантовой криптографии.

Открытый ключ в таких системах задается набором из m квадратичных полиномов от n переменных над полем $GF(q)$:

$$\mathcal{P}(x_1, \dots, x_n) = \begin{pmatrix} f_1(x_1, \dots, x_n) \\ \vdots \\ f_m(x_1, \dots, x_n) \end{pmatrix}; \text{ где } f_i \in \mathbb{F}[x_1, \dots, x_n] \text{ — многочлен второй степени.}$$

2. Математический аппарат HFE

Скрытое уравнения поля (HFE, анг. Hidden Field Equations) — разновидность криптографической системы с открытым ключом, которая является частью многомерной криптографии. Также известна как односторонняя функция с потайным входом HFE.

Система скрытых уравнений поля основана на многочленах над конечными полями K разного размера, чтобы замаскировать связь между закрытым ключом и открытым ключом.

HFE на самом деле является семейством, которое состоит из основных HFE и комбинаций версий HFE. Семейство крипосистем HFE основано на

трудности поиска решений системы многомерных квадратных, поскольку она использует частные аффинные преобразования, чтобы скрыть расширение поля и частные полиномы. Скрытые уравнения поля также использовались для построения схем цифровой подписи, таких как Quartz and Sflash.

Ядро HFE — использование изоморфизма между векторным пространством и расширенным конечным полем для «скрытия» структуры простого полинома.

Поля: Пусть $K = GF(q)$ — малое конечное поле (часто $q=2$), и $L = GF(q^n)$ — его расширение степени n . Существует изоморфизм $\phi: K^n \rightarrow L$, задаваемый выбором базиса $\{\theta_1, \dots, \theta_n\}$ поля L над K : $\phi(x_1, \dots, x_n) = \sum_{i=1}^n x_i \theta_i$.

Центральное отображение (Central Map) $F: L \rightarrow L$: Это единственный нелинейный элемент, известный только создателю ключа. Оно задается одномерным полиномом от одной переменной $X \in L$ специального вида:

$$F(X) = \sum_{\{0 \leq i,j \leq d, q_i + q_j \leq D\}} \alpha_{ij} X^{\{q^i + q^j\}} + \sum_{\{0 \leq i \leq d, q^i \leq D\}} \beta_i X^{\{q^i\}} + \gamma,$$

где $\alpha_{ij}, \beta_i, \gamma \in L$, d — некоторая константа, а D — ключевой параметр системы, ограничивающий максимальную степень в нормальном виде. Ограничение $q^i + q^j \leq D$ обеспечивает, что F как полином от X имеет ограниченную степень, что впоследствии делает возможным эффективное расшифрование через поиск корней. При этом отображение F , будучи записанным через компоненты вектора (x_1, \dots, x_n) с помощью изоморфизма ϕ , становится набором из n квадратичных форм над K (благодаря свойствам отображения Фробениуса $X \rightarrow X^{q^i}$).

Секретные аффинные преобразования S и T : Чтобы скрыть структуру F , вводятся два обратимых аффинных преобразования над векторным пространством K^n :

$$S: K^n \rightarrow K^n, S(x) = M_s * x + v_s,$$

$$T: K^n \rightarrow K^n, T(x) = M_t * x + v_t,$$

где $M_s, M_t \in GL(n, K)$ — невырожденные матрицы, $v_s, v_t \in K^n$ — векторы сдвига.

Построение открытого ключа: Открытое отображение $P: K^n \rightarrow K^n$ является композицией:

$$P = T \circ \varphi^{-1} \circ F \circ \varphi \circ S.$$

Геометрически: входной вектор-сообщение m сначала трансформируется секретным аффинным преобразованием S , затем переводится в элемент большого поля L (φ), к нему применяется «простое» центральное отображение F , результат переводится обратно в векторное пространство (φ^{-1}) и, наконец, «перемешивается» преобразованием T . В результате P представляется как набор из n квадратичных полиномов от n переменных над K , не имеющих очевидной простой внутренней структуры.

Алгоритмы:

- Шифрование (верификация): Для открытого текста $m \in K^n$ вычислить $c = P(m)$. Это прямое вычисление значения n квадратичных полиномов.
- Расшифрование (подписание): Для шифртекста $c \in K^n$ необходимо решить уравнение $P(m) = c$. Владелец секретного ключа выполняет:

$$y = T^{-1}(c).$$

$$Y = \varphi(y) \in L.$$

Решает уравнение $F(X) = Y$ относительно $X \in L$. Так как F — полином ограниченной степени D , его корни можно найти методом Берлекэмпа (или аналогичным) за время, полиномиально зависящее от D .

Для каждого найденного корня X_i вычисляет $x_i = \varphi^{-1}(X_i)$ и затем

$m_i = S^{-1}(x_i)$. Истинное сообщение выбирается среди $\{m_i\}$ по избыточности или хеш-проверке.

Практическая часть

1. Модуль `field_operations`

1.1 Представление данных

В этом коде элементы поля (полиномы) представляются в виде **целых чисел**. Конечным полем $GF(2^n)$ можно управлять как множеством полиномов степени меньше n с коэффициентами из $\{0,1\}$.

Целое число используется как битовая маска – k -й бит числа соответствует коэффициенту при x^k .

Пример: Число 0x13 (в двоичном виде 10011) соответствует полиному

$$1 \cdot x^4 + 0 \cdot x^3 + 0 \cdot x^2 + 1 \cdot x^1 + 1 \cdot x^0 = x^4 + x + 1$$

Константа `KNOWN_IRREDUCIBLE` – это словарь, содержащий заранее вычисленные **неприводимые многочлены** (irreducible polynomials) для различных степеней

Чтобы выполнять умножение в поле $GF(2^n)$, результат обычного умножения полиномов нужно брать по модулю неприводимого многочлена степени n . Неприводимый многочлен играет роль простого числа P в арифметике Z_p .

Словарь позволяет не тратить время на вычисление такого многочлена для популярных размерностей (от 1 до 16).

1.2 Поиск неприводимого многочлена (`_find_irreducible_polynomial, _search`)

Если для заданного n нет готового многочлена в таблице, код пытается найти его алгоритмически:

1. Стратегия: Сначала проверяются простые многочлены-триномы (вида x^n+x^k+1), затем пентаномы (пять членов). Это сделано для оптимизации производительности, так как разреженные полиномы быстрее обрабатываются.

2. Проверка (`_is_irreducible`): Чтобы проверить полином $P(x)$ степени n на неприводимость, нужно убедиться, что он не делится без остатка ни на один полином степени от 1 до $n/2$. Метод перебирает возможные делители (`divisor`) и использует `_polynomial_divide` для проверки остатка.

3. Деление полиномов (`_polynomial_divide`): Реализует классическое деление «столбиком» для двоичных полиномов. Вместо вычитания используется операция XOR (^), так как в GF(2) $1+1=0$ (вычитание равно сложению).

1.3 Арифметические операции

Сложение (`add`) – в поле характеристики 2 сложение эквивалентно побитовому исключающему ИЛИ (**XOR**). Переносов разрядов здесь нет.

```
def add(self, a: int, b: int) -> int:  
    return a ^ b
```

Умножение (`multiply`) – это самая сложная и важная операция. Она состоит из двух этапов:

1. **Обычное полиномиальное умножение:**
 - a. Код проходит по битам второго множителя b .
 - b. Если бит установлен, первый множитель a сдвигается (`<<`) на соответствующую позицию и прибавляется (XOR) к результату.
 - c. Это аналог умножения столбиком.
2. **Приведение по модулю (Modular Reduction):**

```
while result >= (1 << self.n):
    shift = result.bit_length() - self.n - 1
    result ^= self.irreducible << shift
```

a. Результат умножения может иметь степень до $2n-2$, что выходит за пределы поля (максимальная степень должна быть $n-1$).

b. Пока степень результата (`result.bit_length() - 1`) больше или равна n , мы «вычитаем» (XOR) неприводимый многочлен, сдвинутый так, чтобы обнулить старший бит результата.

c. В итоге получается остаток от деления произведения на неприводимый многочлен.

Возвведение в степень (`power`) — реализует алгоритм **быстрого возведения в степень** (Square-and-Multiply).

- Вместо того чтобы умножать число само на себя `exp` раз, алгоритм использует двоичное представление степени.
- Сложность: логарифмическая $O(\log(\exp))$, что критически важно для криптографии, где степени могут быть огромными.

1.4 Утилиты конвертации

vector_to_field: Преобразует список битов (например, `[1, 0, 1]`) в целое число (5). Это нужно, так как в крипtosистемах типа HFE открытый текст часто рассматривается как вектор над $GF(2)$, а операции проводятся как с элементами поля $GF(2^n)$

field_to_vector: Обратная операция — превращает элемент поля в список битов (коэффициентов полинома).

2. Модуль `hfe_base`

Этот модуль реализует **крипtosистему с открытым ключом HFE (Hidden Field Equations)**.

HFE относится к классу **многомерной криптографии** (Multivariate

Cryptography). Её суть заключается в том, чтобы скрыть сложную, но решаемую математическую операцию над большим полем $GF(2^n)$ внутри двух запутывающих линейных (аффинных) преобразований, чтобы для внешнего наблюдателя это выглядело как система случайных квадратичных уравнений.

2.1 Инициализация и генерация ключей

Класс HFEBase – основной класс, который связывает математику поля ($GF(2^n)$ из предыдущего кода) и матричные операции.

Метод `_generate_key` – здесь создается «ловушка» (trapdoor). Система HFE основана на формуле:

$$\text{PublicKey}(x) = T(P(S(x))), \text{ где:}$$

1. **S (Secret):** Аффинное преобразование входного вектора.

Представляется как $y = S_1 \cdot x + S_0$

2. **P (Secret):** Полином над полем $GF(2^n)$

3. **T (Secret):** Аффинное преобразование выходного вектора.

Представляется как $y = T_1 \cdot x + T_0$

В коде генерируются:

- S_1, T_1 : Случайные обратимые матрицы размера $n \times n$ (генерируются методом `_generate_invertible_matrix`).
- S_0, T_0 : Случайные векторы сдвига.
- S_1^{-1}, T_1^{-1} : Обратные матрицы, вычисляются сразу, так как они нужны для расшифрования.

Матричная арифметика (`_matrix_inverse`, `_is_invertible`) – все операции (сложение, умножение) с матрицами проводятся по модулю 2. Используется метод Гаусса-Жордана для нахождения обратной матрицы. Поскольку мы в $GF(2)$, сложение строк — это операция XOR, а умножение — AND.

2.2 Секретный полином

Метод `_hfe_polynomial` – это «сердце» алгоритма. Функция вычисляет $P(x)$.

В классическом HFE полином имеет специальный вид (полином Демидова), который позволяет легко находить корни. В данной реализации используется упрощенная форма полинома:

$$\sum x^{2^i}$$

В коде суммируются степени $x^2, x^4, x^8\dots$ в зависимости от параметра d . Операции сложения (`add`) и возведения в степень (`power`) берутся из класса `GF2n`.

2.3 Шифрование (encrypt)

Процесс превращает открытый текст (вектор бит) в шифротекст.

1. **Вход:** Вектор битов `plaintext`.
2. **Шаг 1 (S):** Применяется секретное преобразование S . Вектор умножается на матрицу S_1 и складывается с вектором S_0 . Результат преобразуется из вектора в элемент поля (число).
3. **Шаг 2 (P):** К полученному числу применяется секретный полином `_hfe_polynomial`. Результат преобразуется обратно в вектор.
4. **Шаг 3 (T):** Применяется секретное преобразование T (умножение на $T_1 + T_0$).
5. **Выход:** Вектор битов `ciphertext`. Для атакующего, который не знает S и T , эта композиция выглядит как набор случайных уравнений, которые очень трудно решить.

2.4 Расшифрование (decrypt)

Обратный процесс. Владелец ключа знает секретные компоненты, поэтому может «раздеть» уравнение слой за слоем.

1. **Вход:** Шифротекст.

2. **Шаг 1 (Inv T):** Снимается внешнее преобразование T.
 - a. Формула: $y' = T^{-1} \cdot 1 \cdot (y - T_0)$
 - b. В коде: $T_{1\text{inv}} @ (-T_0)$ (в поле характеристики 2 вычитание равно сложению).
3. **Шаг 2 (Solve P):** Решается уравнение $P(x) = y'$. Нужно найти такой x , который при подстановке в полином даст y' .
 - **Важный момент (_solve_hfe):** В этой учебной реализации используется **полный перебор (Brute Force)**. Код просто перебирает все числа от 0 до $2^n - 1$ и проверяет, подходит ли корень. *Примечание:* В реальной криптографии (где $n \approx 100$) перебор невозможен. Там используется специальный алгоритм Берлекэмпа (Berlekamp algorithm) для поиска корней полиномов над конечными полями.
4. **Шаг 3 (Inv S):** Снимается внутреннее преобразование S. Формула: $x = S^{-1} \cdot 1 \cdot (x' - S_0)$.
5. **Выход:** Исходный текст.

5. Работа с данными (encrypt_block, decrypt_block)

Эти методы — обертки для удобства. Поскольку HFE работает с блоками фиксированной длины n бит, эти функции:

1. Берут байты на вход.
2. Разбивают их на биты.
3. Шифруют/десифруют.
4. Собирают биты обратно в байты.

Так как по умолчанию $n=8$, система шифрует данные побайтово (как простая замена), что небезопасно для реального использования без режима сцепления блоков (CBC и т.д.), но достаточно для демонстрации алгебраической структуры алгоритма.

3. Модуль hfe_cpu_parallel

Этот модуль представляет собой **оптимизированную версию** крипtosистемы HFE, предназначенную для работы на многоядерных процессорах.

В стандартном Python (CPython) существует **GIL (Global Interpreter Lock)**, который не позволяет потокам (threading) эффективно выполнять вычисления параллельно на разных ядрах CPU. Чтобы обойти это ограничение и ускорить тяжелые математические операции HFE (особенно расшифрование), используется библиотека multiprocessing.

3.1 Архитектура "Master-Worker"

Код разделен на класс-менеджер (HFECPUParallel) и независимые функции-рабочие (_worker и вспомогательные), которые вынесены за пределы класса.

Функции вынесены за пределы класса, потому что библиотека multiprocessing передает данные между процессами с помощью сериализации (pickling).

- Сериализовать целый экземпляр класса со всем его состоянием — дорого и иногда проблематично.
- Передача "чистых" функций и только необходимых данных (матриц ключей) работает быстрее и надежнее.

3.2 Рабочие функции (Workers)

Эти функции выполняются в отдельных процессах операционной системы.

- _encrypt_chunk_worker – функция берет кусок ("чанк") данных и шифрует его.

Принимает кортеж args, содержащий сам чанк данных, матрицы ключей (S,TS, TS,T) и параметры поля (n,dn, dn,d). Внутри функции создается свой

экземпляр `field = GF2n(n)`. Это важно, чтобы процессы не конфликтовали за общие ресурсы. Функция проходит по списку векторов в чанке и для каждого выполняет классические шаги HFE: $T(P(S(x)))$

- `_decrypt_chunk_worker` – аналогичная функция для расшифрования, но с важной оптимизацией. Предварительные вычисления – вместо того чтобы пересчитывать вектор сдвига для каждого блока данных, функция вычисляет константы один раз перед циклом обработки чанка.

Логика: Выполняет обратные преобразования $S^{-1}(P^{-1}(T^{-1}(y)))$

Самая тяжелая часть — P^{-1} (решение уравнения через перебор), именно она получает максимальный прирост производительности от распараллеливания.

Функции `_affine_transform`, `_hfe_polynomial`, `_solve_hfe` продублированы как глобальные функции (вне класса `HFEBASE`). Это сделано для того, чтобы worker-процессы могли их вызывать без необходимости иметь доступ к объекту основного класса.

3.3 Класс HFECPUParallel

Этот класс наследуется от базового `HFEBASE` и добавляет логику управления процессами.

Инициализация:

- `super().__init__`: Генерирует ключи (матрицы S и T) с помощью родительского класса.
- `cpu_count()`: Определяет количество доступных ядер процессора для создания пула процессов.

Метод `encrypt_block` (и `decrypt_block`) – это "менеджер", который распределяет задачи:

1. Подготовка данных: Входные байты превращаются в огромный список векторов (списков из 0 и 1).

2. Чанкинг (Chunking): Данные разбиваются на равные части. Если у вас 8 ядер и 800 байт данных, каждый процесс получит по 100 байт. Это снижает накладные расходы на межпроцессное взаимодействие (IPC).

3. Параллельное выполнение (Pool):

- i. Создается пул процессов.
- ii. pool.map берет список аргументов, "скармливает" каждый набор отдельному процессу и ждет завершения всех.
- iii. Каждый процесс работает независимо на своем ядре CPU.

4. Сборка (Aggregation): Результаты (справки зашифрованных векторов) от разных процессов склеиваются в один длинный список (ciphertexts.extend), который затем преобразуется обратно в байты.

4. Модуль hfe_gpu_parallel

Данный модуль представляет собой GPU-ускоренную реализацию HFE с использованием библиотеки PyTorch.

Основное отличие от предыдущих версий заключается в использовании архитектуры SIMD (Single Instruction, Multiple Data). Вместо того чтобы обрабатывать один элемент за другим (как на CPU) или несколько кусков данных на разных ядрах (multiprocessing), GPU выполняет одну и ту же математическую операцию над тысячами чисел одновременно.

4.1 GF2nGPU: Математика поля на тензорах

Этот класс переопределяет арифметику $GF(2^n)$ для работы с тензорами PyTorch вместо обычных чисел.

- Тензоры вместо чисел: Все данные хранятся в памяти GPU (device='cuda'). Операции выполняются над целыми массивами (батчами) данных сразу.
- Сложение (add): $a \wedge b$. В PyTorch оператор XOR работает поэлементно для тензоров любого размера.

- Умножение (multiply):
 - Реализовано "в лоб" через циклы, но внутри цикла операции (сдвиг $<<$, XOR \wedge , AND $\&$) применяются сразу ко всему тензору result.
 - Это называется векторизацией. Например, если размер батча 1024, то одна команда `result ^ shifted` выполняет 1024 операции XOR одновременно.
- Редукция (`_reduce_mod_polynomial`):
 - Чтобы оставаться в пределах поля, нужно делить полином на неприводимый многочлен. Здесь реализован алгоритм, который проходит от старшего бита к младшему и "срезает" лишние биты для всего тензора сразу, используя маскирование (`bit_mask`).

4.2 HFEGPUParallel: Логика криптосистемы

Класс управляет процессом шифрования/десифрования, используя возможности видеокарты.

Инициализация и перенос данных (`__init__`, `_transfer_keys_to_gpu`):

- Ключи (матрицы S, TS, TS, T и их обратные версии) создаются на CPU, а затем переносятся в память видеокарты (VRAM) как тензоры.
- Матрицы конвертируются в float32. Это связано с тем, что оптимизированные ядра CUDA для матричного умножения (`matmul`) лучше всего работают с числами с плавающей точкой. После умножения результат приводится обратно к целым числам и берется по модулю 2.

Аффинные преобразования (`_affine_transform_gpu`):

- Реализует уравнение $y = Ax + b \pmod{2}$.
- Если на вход приходит пакет из 1000 векторов, код не запускает

цикл. Он выполняет одну матричную операцию:

- $Y_{batch} = X_{batch} \cdot A^T + B$, где X_{batch} — матрица размером $BatchSize \times n$
- Это дает колоссальный прирост производительности по сравнению с циклами на CPU.

Решение уравнения HFE (`_solve_hfe_gpu`) — в версиях для CPU мы перебирали x в цикле. На GPU используется подход "Parallel Brute-Force":

1. Создание: Генерируется тензор `all_x`, содержащий *все возможные элементы поля* (от 0 до $2^n - 1$).
2. Расширение (Broadcasting): Этот массив дублируется для каждого элемента в батче входящих данных. Создается гигантская матрица размером `(batch_size, field_size)`.
3. Массовое вычисление: Функция $P(x)$ применяется к этой гигантской матрице за один проход. Мы вычисляем результат полинома сразу для всех возможных x для всех запросов в батче.
4. Сравнение: Полученная матрица сравнивается с искомыми значениями y .
5. Поиск индекса (`argmax`): Находятся индексы, где совпало значение.

Пакетная обработка (`encrypt_block`, `decrypt_block`) — взаимодействие между CPU (оперативная память) и GPU (видеопамять) — дорогая операция (высокая латентность). Поэтому нельзя отправлять по 1 байту.

1. Данные накапливаются в большой список.
2. Разбиваются на батчи (по умолчанию 1024 элемента).
3. Весь батч одним куском копируется на GPU (`torch.tensor(...)`).
4. Происходит вычисление.
5. Результат копируется обратно на CPU (`.cpu().numpy()`).

5. Тестирование и сравнение результатов

Тестовый запуск осуществлялся на различных размерах данных – от 256 байт до 131072 байт с шагом x2. Ниже приведен анализ полученных результатов.

Накладные ресурсы:

- Base (Single Thread): Не имеет накладных расходов. Для 256 байт это самый быстрый метод (0.27с), так как он просто начинает считать.
- CPU Parallel: Имеет колоссальные накладные расходы. Для 256 байт, 512 байт, 1 КБ, 2 КБ, 4 КБ время работы почти одинаковое — около 5.2 - 6.0 секунд. Это время тратится на multiprocessing.Pool: создание процессов, сериализацию данных (pickle), передачу матриц ключей в каждый процесс. Вывод: Для малых данных (меньше 8-16 КБ) параллелизация на CPU вредна.
- GPU Parallel: Имеет умеренные накладные расходы (~0.15 - 0.20с). Это время уходит на инициализацию CUDA контекста и копирование данных из RAM в VRAM.

В сравнении обычной и CPU версией параллелизация начинает окупаться только на 8 КБ – CPU версия (6.2с) впервые становится быстрее Base версии (8.14с). Ускорение 1.31x.

При этом GPU версия быстрее непараллельной версии всегда, даже на 256 байтах (0.16с против 0.27с), несмотря на оверхед. Это связано с тем, что даже "медленный" старт GPU быстрее, чем последовательный перебор корней HFE на одном ядре.

Сводная таблица производительности представлена ниже.

Таблица 1 – Сводная таблица производительности

Размер данных	Base (сек)	CPU (сек)	GPU (сек)	Лучший метод
256 B	0.27	5.22	0.16	GPU
1 KB	1.05	5.75	0.17	GPU
4 KB	4.19	6.09	0.32	GPU
8 KB	8.15	6.21	0.50	GPU
16 KB	16.00	7.10	0.91	GPU
65 KB	63.48	11.63	3.21	GPU
131 KB	129.60	17.55	6.30	GPU

Итог:

- Для реального применения HFE: Использование GPU является обязательным для достижения приемлемой производительности, если n мало (как в примере, $n=8$). Если n будет большим (например, $n=100$, как в реальной криптографии), метод "грубой силы" на GPU перестанет работать из-за нехватки памяти, и потребуется реализация алгебраических алгоритмов (Берлекэмп-Месси) на CUDA.
- CPU multiprocessing: Имеет смысл только при обработке больших файлов (>100 КБ) одним куском.
- Base: Подходит только для учебных целей или передачи коротких сообщений (до 1 КБ).

ЗАКЛЮЧЕНИЕ

В ходе работы было установлено, что криптосистема HFE обладает значительным внутренним потенциалом для параллельной реализации, который, однако, асимметричен для операций шифрования и расшифрования.

Шифрование, будучи задачей вычисления набора квадратичных форм, является регулярной, легко распараллелимой задачей с высокой арифметической интенсивностью. Его реализация на многоядерных CPU и, особенно, на GPU позволяет достичь почти линейного ускорения, делая HFE конкурентоспособной для высоконагруженных приложений.

Расшифрование упирается в сложность параллелизации алгоритма факторизации полиномов (алгоритм Берлекэмпа). Хотя отдельные его этапы (линейная алгебра, перебор корней) поддаются распараллеливанию, общий выигрыш ограничен наличием последовательных фаз и нерегулярной структурой алгоритма.

Также в ходе работы были реализованы три различных подхода к одной и той же математической задаче:

1. Последовательный (Base).

Принцип: Классическое исполнение инструкций одна за другой.

Вердикт: Идеален для понимания алгоритма и отладки математики.

Имеет минимальную латентность (мгновенный старт), что делает его лучшим выбором для шифрования очень малых пакетов данных (< 1 КБ), но абсолютно непригоден для обработки массивов данных из-за линейной сложности расшифрования.

2. Многопроцессорный (CPU Parallel):

Принцип: Обход Python GIL (Global Interpreter Lock) путем создания независимых процессов ОС.

Вердикт: Демонстрирует высокие накладные расходы (overhead) на

запуск процессов и сериализацию данных (pickle). Эффективен только при длительных, вычислительно сложных задачах (расшифрование больших файлов). Масштабируется линейно от количества ядер CPU (в тесте ускорение ~7.4x на 8 ядрах).

3. Массово-параллельный (GPU Parallel):

Принцип: SIMD (Single Instruction, Multiple Data). Преобразование алгоритмической задачи в тензорные операции линейной алгебры.

Вердикт: Абсолютный лидер по пропускной способности (Throughput). Ускорение более 20x на больших объемах. GPU нивелирует сложность перебора корней за счет одновременной проверки тысяч гипотез.

Однако требует времени на инициализацию контекста CUDA и пересылку данных через шину PCI-E.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Алферов А. П., Зубов А. Ю., Кузьмин А. С., Черемушкин А. В. Основы криптографии: учебное пособие. – М.: Гелиос АРВ, 2005. – 480 с.
2. Лидл Р., Нидеррайтер Г. Конечные поля. – М.: Мир, 1988. – 822 с.
3. Смарт Н. Криптография. – М.: Техносфера, 2005. – 528 с.
4. Молдовян Н. А., Молдовян А. А. Криптография: от примитивов к синтезу алгоритмов. – СПб.: БХВ-Петербург, 2004. – 448 с.
5. Василенко О. Н. Теоретико-числовые алгоритмы в криптографии. – М.: МЦНМО, 2003. – 328 с.
6. Воеводин В. В., Воеводин Вл. В. Параллельные вычисления. – СПб.: БХВ-Петербург, 2002. – 608 с.
7. Шолле Ф. Глубокое обучение на Python. – СПб.: Питер, 2018. – 400 с.
8. Бизли Д. Python. Подробный справочник. – СПб.: Символ-Плюс, 2010. – 864 с.

ПРИЛОЖЕНИЕ А

<https://github.com/TheRealMal/HFE-crypto-system/tree/main>

ПРИЛОЖЕНИЕ Б

Результаты тестирования производительности

Дата: 2025-12-01 12:48:03

Размер данных: 256 байт

```
=====
2025-12-01 12:48:04 - HFE_Main - INFO -
=====
2025-12-01 12:48:04 - HFE_Main - INFO - HFE Криптосистема
2025-12-01 12:48:04 - HFE_Main - INFO -
=====
2025-12-01 12:48:04 - HFE_Main - INFO - Параметры: n=8, d=3, seed=42
2025-12-01 12:48:04 - HFE_Main - INFO - Размер тестовых данных: 256 байт
2025-12-01 12:48:04 - HFE_Main - INFO - Режим работы: all
2025-12-01 12:48:04 - HFE_Main - INFO - Тестовые данные сгенерированы:
390c8c7d7247342cd8100f2f6f770d65d670e58e...
2025-12-01 12:48:04 - HFE_Main - INFO -
=====
2025-12-01 12:48:04 - HFE_Main - INFO - ЗАПУСК ОБЫЧНОЙ ВЕРСИИ (без
распараллеливания)
2025-12-01 12:48:04 - HFE_Main - INFO -
=====
2025-12-01 12:48:04 - HFE_Main - INFO - === Бенчмарк для режима: Обычная
версия ===
2025-12-01 12:48:04 - HFE_Main - INFO - Размер данных: 256 байт
2025-12-01 12:48:04 - HFE_Main - INFO - Время шифрования: 0.0040 секунд
2025-12-01 12:48:04 - HFE_Main - INFO - Скорость шифрования: 64004.64
байт/сек
2025-12-01 12:48:04 - HFE_Main - INFO - Время расшифрования: 0.2695 секунд
2025-12-01 12:48:04 - HFE_Main - INFO - Скорость расшифрования: 949.81
байт/сек
2025-12-01 12:48:04 - HFE_Main - INFO - [OK] Расшифрование успешно: данные
совпадают с оригиналом
2025-12-01 12:48:04 - HFE_Main - INFO - Общее время: 0.2735 секунд
2025-12-01 12:48:04 - HFE_Main - INFO -
=====
2025-12-01 12:48:04 - HFE_Main - INFO -
=====
2025-12-01 12:48:04 - HFE_Main - INFO - ЗАПУСК CPU-ПАРАЛЛЕЛЬНОЙ ВЕРСИИ
2025-12-01 12:48:04 - HFE_Main - INFO -
=====
2025-12-01 12:48:04 - HFE_Main - INFO - === Бенчмарк для режима: CPU-
параллельная версия ===
2025-12-01 12:48:04 - HFE_Main - INFO - Размер данных: 256 байт
2025-12-01 12:48:07 - HFE_Main - INFO - Время шифрования: 2.5185 секунд
2025-12-01 12:48:07 - HFE_Main - INFO - Скорость шифрования: 101.65 байт/сек
2025-12-01 12:48:10 - HFE_Main - INFO - Время расшифрования: 2.6969 секунд
2025-12-01 12:48:10 - HFE_Main - INFO - Скорость расшифрования: 94.92
```

байт/сек

```
2025-12-01 12:48:10 - HFE_Main - INFO - [OK] Расшифрование успешно: данные совпадают с оригиналом
2025-12-01 12:48:10 - HFE_Main - INFO - Общее время: 5.2154 секунд
2025-12-01 12:48:10 - HFE_Main - INFO -
=====
2025-12-01 12:48:10 - HFE_Main - INFO -
=====
2025-12-01 12:48:10 - HFE_Main - INFO - ЗАПУСК GPU-ПАРАЛЛЕЛЬНОЙ ВЕРСИИ
2025-12-01 12:48:10 - HFE_Main - INFO -
=====
2025-12-01 12:48:10 - HFE_Main - INFO - === Бенчмарк для режима: GPU-параллельная версия ===
2025-12-01 12:48:10 - HFE_Main - INFO - Размер данных: 256 байт
2025-12-01 12:48:10 - HFE_Main - INFO - Время шифрования: 0.1161 секунд
2025-12-01 12:48:10 - HFE_Main - INFO - Скорость шифрования: 2204.36 байт/сек
2025-12-01 12:48:10 - HFE_Main - INFO - Время расшифрования: 0.0480 секунд
2025-12-01 12:48:10 - HFE_Main - INFO - Скорость расшифрования: 5333.32 байт/сек
2025-12-01 12:48:10 - HFE_Main - INFO - [OK] Расшифрование успешно: данные совпадают с оригиналом
2025-12-01 12:48:10 - HFE_Main - INFO - Общее время: 0.1641 секунд
2025-12-01 12:48:10 - HFE_Main - INFO -
=====
2025-12-01 12:48:10 - HFE_Main - INFO -
=====
2025-12-01 12:48:10 - HFE_Main - INFO - СРАВНЕНИЕ РЕЗУЛЬТАТОВ
2025-12-01 12:48:10 - HFE_Main - INFO -
=====
2025-12-01 12:48:10 - HFE_Main - INFO - [OK] BASE : 0.2735 сек
2025-12-01 12:48:10 - HFE_Main - INFO - [OK] CPU : 5.2154 сек
(ускорение: 0.05x)
2025-12-01 12:48:10 - HFE_Main - INFO - [OK] GPU : 0.1641 сек
(ускорение: 1.67x)
2025-12-01 12:48:10 - HFE_Main - INFO -
=====
2025-12-01 12:48:10 - HFE_Main - INFO - РАБОТА ЗАВЕРШЕНА
2025-12-01 12:48:10 - HFE_Main - INFO -
=====
```

Размер данных: 512 байт

```
-----
2025-12-01 12:48:11 - HFE_Main - INFO -
=====
2025-12-01 12:48:11 - HFE_Main - INFO - HFE Криптосистема
2025-12-01 12:48:11 - HFE_Main - INFO -
=====
2025-12-01 12:48:11 - HFE_Main - INFO - Параметры: n=8, d=3, seed=42
2025-12-01 12:48:11 - HFE_Main - INFO - Размер тестовых данных: 512 байт
```

```
2025-12-01 12:48:11 - HFE_Main - INFO - Режим работы: all
2025-12-01 12:48:11 - HFE_Main - INFO - Тестовые данные сгенерированы:
390c8c7d7247342cd8100f2f6f770d65d670e58e...
2025-12-01 12:48:11 - HFE_Main - INFO -
=====
2025-12-01 12:48:11 - HFE_Main - INFO - ЗАПУСК ОБЫЧНОЙ ВЕРСИИ (без
распараллеливания)
2025-12-01 12:48:11 - HFE_Main - INFO -
=====
2025-12-01 12:48:11 - HFE_Main - INFO - === Бенчмарк для режима: Обычная
версия ===
2025-12-01 12:48:11 - HFE_Main - INFO - Размер данных: 512 байт
2025-12-01 12:48:11 - HFE_Main - INFO - Время шифрования: 0.0080 секунд
2025-12-01 12:48:11 - HFE_Main - INFO - Скорость шифрования: 64014.18
байт/сек
2025-12-01 12:48:12 - HFE_Main - INFO - Время расшифрования: 0.5455 секунд
2025-12-01 12:48:12 - HFE_Main - INFO - Скорость расшифрования: 938.53
байт/сек
2025-12-01 12:48:12 - HFE_Main - INFO - [OK] Расшифрование успешно: данные
совпадают с оригиналом
2025-12-01 12:48:12 - HFE_Main - INFO - Общее время: 0.5535 секунд
2025-12-01 12:48:12 - HFE_Main - INFO -
=====
2025-12-01 12:48:12 - HFE_Main - INFO -
=====
2025-12-01 12:48:12 - HFE_Main - INFO - ЗАПУСК CPU-ПАРАЛЛЕЛЬНОЙ ВЕРСИИ
2025-12-01 12:48:12 - HFE_Main - INFO -
=====
2025-12-01 12:48:12 - HFE_Main - INFO - === Бенчмарк для режима: CPU-
параллельная версия ===
2025-12-01 12:48:12 - HFE_Main - INFO - Размер данных: 512 байт
2025-12-01 12:48:15 - HFE_Main - INFO - Время шифрования: 2.5110 секунд
2025-12-01 12:48:15 - HFE_Main - INFO - Скорость шифрования: 203.90 байт/сек
2025-12-01 12:48:17 - HFE_Main - INFO - Время расшифрования: 2.9417 секунд
2025-12-01 12:48:17 - HFE_Main - INFO - Скорость расшифрования: 174.05
байт/сек
2025-12-01 12:48:17 - HFE_Main - INFO - [OK] Расшифрование успешно: данные
совпадают с оригиналом
2025-12-01 12:48:17 - HFE_Main - INFO - Общее время: 5.4528 секунд
2025-12-01 12:48:17 - HFE_Main - INFO -
=====
2025-12-01 12:48:17 - HFE_Main - INFO -
=====
2025-12-01 12:48:17 - HFE_Main - INFO - ЗАПУСК GPU-ПАРАЛЛЕЛЬНОЙ ВЕРСИИ
2025-12-01 12:48:17 - HFE_Main - INFO -
=====
2025-12-01 12:48:18 - HFE_Main - INFO - === Бенчмарк для режима: GPU-
параллельная версия ===
2025-12-01 12:48:18 - HFE_Main - INFO - Размер данных: 512 байт
2025-12-01 12:48:18 - HFE_Main - INFO - Время шифрования: 0.1231 секунд
```

```
2025-12-01 12:48:18 - HFE_Main - INFO - Скорость шифрования: 4160.21 байт/сек
2025-12-01 12:48:18 - HFE_Main - INFO - Время расшифрования: 0.0530 секунд
2025-12-01 12:48:18 - HFE_Main - INFO - Скорость расшифрования: 9660.34
байт/сек
2025-12-01 12:48:18 - HFE_Main - INFO - [OK] Расшифрование успешно: данные
совпадают с оригиналом
2025-12-01 12:48:18 - HFE_Main - INFO - Общее время: 0.1761 секунд
2025-12-01 12:48:18 - HFE_Main - INFO -
=====
2025-12-01 12:48:18 - HFE_Main - INFO -
=====
2025-12-01 12:48:18 - HFE_Main - INFO - СРАВНЕНИЕ РЕЗУЛЬТАТОВ
2025-12-01 12:48:18 - HFE_Main - INFO -
=====
2025-12-01 12:48:18 - HFE_Main - INFO - [OK] BASE : 0.5535 сек
2025-12-01 12:48:18 - HFE_Main - INFO - [OK] CPU : 5.4528 сек
(ускорение: 0.10x)
2025-12-01 12:48:18 - HFE_Main - INFO - [OK] GPU : 0.1761 сек
(ускорение: 3.14x)
2025-12-01 12:48:18 - HFE_Main - INFO -
=====
2025-12-01 12:48:18 - HFE_Main - INFO - РАБОТА ЗАВЕРШЕНА
2025-12-01 12:48:18 - HFE_Main - INFO -
=====
```

Размер данных: 1024 байт

```
-----
2025-12-01 12:48:19 - HFE_Main - INFO -
=====
2025-12-01 12:48:19 - HFE_Main - INFO - HFE Криптосистема
2025-12-01 12:48:19 - HFE_Main - INFO -
=====
2025-12-01 12:48:19 - HFE_Main - INFO - Параметры: n=8, d=3, seed=42
2025-12-01 12:48:19 - HFE_Main - INFO - Размер тестовых данных: 1024 байт
2025-12-01 12:48:19 - HFE_Main - INFO - Режим работы: all
2025-12-01 12:48:19 - HFE_Main - INFO - Тестовые данные сгенерированы:
390c8c7d7247342cd8100f2f6f770d65d670e58e...
2025-12-01 12:48:19 - HFE_Main - INFO -
=====
2025-12-01 12:48:19 - HFE_Main - INFO - ЗАПУСК ОБЫЧНОЙ ВЕРСИИ (без
распараллеливания)
2025-12-01 12:48:19 - HFE_Main - INFO -
=====
2025-12-01 12:48:19 - HFE_Main - INFO - === Бенчмарк для режима: Обычная
версия ===
2025-12-01 12:48:19 - HFE_Main - INFO - Размер данных: 1024 байт
2025-12-01 12:48:19 - HFE_Main - INFO - Время шифрования: 0.0180 секунд
2025-12-01 12:48:19 - HFE_Main - INFO - Скорость шифрования: 56877.19
байт/сек
```

```
2025-12-01 12:48:20 - HFE_Main - INFO - Время расшифрования: 1.0326 секунд
2025-12-01 12:48:20 - HFE_Main - INFO - Скорость расшифрования: 991.68
байт/сек
2025-12-01 12:48:20 - HFE_Main - INFO - [OK] Расшифрование успешно: данные
совпадают с оригиналом
2025-12-01 12:48:20 - HFE_Main - INFO - Общее время: 1.0506 секунд
2025-12-01 12:48:20 - HFE_Main - INFO -
=====
2025-12-01 12:48:20 - HFE_Main - INFO -
=====
2025-12-01 12:48:20 - HFE_Main - INFO - ЗАПУСК CPU-ПАРАЛЛЕЛЬНОЙ ВЕРСИИ
2025-12-01 12:48:20 - HFE_Main - INFO -
=====
2025-12-01 12:48:20 - HFE_Main - INFO - === Бенчмарк для режима: CPU-
параллельная версия ===
2025-12-01 12:48:20 - HFE_Main - INFO - Размер данных: 1024 байт
2025-12-01 12:48:23 - HFE_Main - INFO - Время шифрования: 2.6558 секунд
2025-12-01 12:48:23 - HFE_Main - INFO - Скорость шифрования: 385.57 байт/сек
2025-12-01 12:48:26 - HFE_Main - INFO - Время расшифрования: 3.0947 секунд
2025-12-01 12:48:26 - HFE_Main - INFO - Скорость расшифрования: 330.89
байт/сек
2025-12-01 12:48:26 - HFE_Main - INFO - [OK] Расшифрование успешно: данные
совпадают с оригиналом
2025-12-01 12:48:26 - HFE_Main - INFO - Общее время: 5.7505 секунд
2025-12-01 12:48:26 - HFE_Main - INFO -
=====
2025-12-01 12:48:26 - HFE_Main - INFO -
=====
2025-12-01 12:48:26 - HFE_Main - INFO - ЗАПУСК GPU-ПАРАЛЛЕЛЬНОЙ ВЕРСИИ
2025-12-01 12:48:26 - HFE_Main - INFO -
=====
2025-12-01 12:48:26 - HFE_Main - INFO - === Бенчмарк для режима: GPU-
параллельная версия ===
2025-12-01 12:48:26 - HFE_Main - INFO - Размер данных: 1024 байт
2025-12-01 12:48:26 - HFE_Main - INFO - Время шифрования: 0.1205 секунд
2025-12-01 12:48:26 - HFE_Main - INFO - Скорость шифрования: 8497.23 байт/сек
2025-12-01 12:48:26 - HFE_Main - INFO - Время расшифрования: 0.0510 секунд
2025-12-01 12:48:26 - HFE_Main - INFO - Скорость расшифрования: 20077.45
байт/сек
2025-12-01 12:48:26 - HFE_Main - INFO - [OK] Расшифрование успешно: данные
совпадают с оригиналом
2025-12-01 12:48:26 - HFE_Main - INFO - Общее время: 0.1715 секунд
2025-12-01 12:48:26 - HFE_Main - INFO -
=====
2025-12-01 12:48:26 - HFE_Main - INFO -
=====
2025-12-01 12:48:26 - HFE_Main - INFO - СРАВНЕНИЕ РЕЗУЛЬТАТОВ
2025-12-01 12:48:26 - HFE_Main - INFO -
=====
2025-12-01 12:48:26 - HFE_Main - INFO - [OK] BASE : 1.0506 сек
```

```
2025-12-01 12:48:26 - HFE_Main - INFO - [OK] CPU : 5.7505 сек
(ускорение: 0.18x)
2025-12-01 12:48:26 - HFE_Main - INFO - [OK] GPU : 0.1715 сек
(ускорение: 6.13x)
2025-12-01 12:48:26 - HFE_Main - INFO -
=====
2025-12-01 12:48:26 - HFE_Main - INFO - РАБОТА ЗАВЕРШЕНА
2025-12-01 12:48:26 - HFE_Main - INFO -
=====

Размер данных: 2048 байт
-----
2025-12-01 12:48:28 - HFE_Main - INFO -
=====
2025-12-01 12:48:28 - HFE_Main - INFO - HFE Криптосистема
2025-12-01 12:48:28 - HFE_Main - INFO -
=====
2025-12-01 12:48:28 - HFE_Main - INFO - Параметры: n=8, d=3, seed=42
2025-12-01 12:48:28 - HFE_Main - INFO - Размер тестовых данных: 2048 байт
2025-12-01 12:48:28 - HFE_Main - INFO - Режим работы: all
2025-12-01 12:48:28 - HFE_Main - INFO - Тестовые данные сгенерированы:
390c8c7d7247342cd8100f2f6f770d65d670e58e...
2025-12-01 12:48:28 - HFE_Main - INFO -
=====
2025-12-01 12:48:28 - HFE_Main - INFO - ЗАПУСК ОБЫЧНОЙ ВЕРСИИ (без
распараллеливания)
2025-12-01 12:48:28 - HFE_Main - INFO -
=====
2025-12-01 12:48:28 - HFE_Main - INFO - === Бенчмарк для режима: Обычная
версия ===
2025-12-01 12:48:28 - HFE_Main - INFO - Размер данных: 2048 байт
2025-12-01 12:48:28 - HFE_Main - INFO - Время шифрования: 0.0315 секунд
2025-12-01 12:48:28 - HFE_Main - INFO - Скорость шифрования: 65008.28
байт/сек
2025-12-01 12:48:30 - HFE_Main - INFO - Время расшифрования: 2.0497 секунд
2025-12-01 12:48:30 - HFE_Main - INFO - Скорость расшифрования: 999.17
байт/сек
2025-12-01 12:48:30 - HFE_Main - INFO - [OK] Расшифрование успешно: данные
совпадают с оригиналом
2025-12-01 12:48:30 - HFE_Main - INFO - Общее время: 2.0812 секунд
2025-12-01 12:48:30 - HFE_Main - INFO -
=====
2025-12-01 12:48:30 - HFE_Main - INFO -
=====
2025-12-01 12:48:30 - HFE_Main - INFO - ЗАПУСК CPU-ПАРАЛЛЕЛЬНОЙ ВЕРСИИ
2025-12-01 12:48:30 - HFE_Main - INFO -
=====
2025-12-01 12:48:30 - HFE_Main - INFO - === Бенчмарк для режима: CPU-
параллельная версия ===
```

```
2025-12-01 12:48:30 - HFE_Main - INFO - Размер данных: 2048 байт
2025-12-01 12:48:33 - HFE_Main - INFO - Время шифрования: 2.6536 секунд
2025-12-01 12:48:33 - HFE_Main - INFO - Скорость шифрования: 771.80 байт/сек
2025-12-01 12:48:36 - HFE_Main - INFO - Время расшифрования: 3.1931 секунд
2025-12-01 12:48:36 - HFE_Main - INFO - Скорость расшифрования: 641.39
байт/сек
2025-12-01 12:48:36 - HFE_Main - INFO - [OK] Расшифрование успешно: данные
совпадают с оригиналом
2025-12-01 12:48:36 - HFE_Main - INFO - Общее время: 5.8466 секунд
2025-12-01 12:48:36 - HFE_Main - INFO -
=====
2025-12-01 12:48:36 - HFE_Main - INFO -
=====
2025-12-01 12:48:36 - HFE_Main - INFO - ЗАПУСК GPU-ПАРАЛЛЕЛЬНОЙ ВЕРСИИ
2025-12-01 12:48:36 - HFE_Main - INFO -
=====
2025-12-01 12:48:36 - HFE_Main - INFO - === Бенчмарк для режима: GPU-
параллельная версия ===
2025-12-01 12:48:36 - HFE_Main - INFO - Размер данных: 2048 байт
2025-12-01 12:48:36 - HFE_Main - INFO - Время шифрования: 0.1398 секунд
2025-12-01 12:48:36 - HFE_Main - INFO - Скорость шифрования: 14652.84
байт/сек
2025-12-01 12:48:36 - HFE_Main - INFO - Время расшифрования: 0.0845 секунд
2025-12-01 12:48:36 - HFE_Main - INFO - Скорость расшифрования: 24230.38
байт/сек
2025-12-01 12:48:36 - HFE_Main - INFO - [OK] Расшифрование успешно: данные
совпадают с оригиналом
2025-12-01 12:48:36 - HFE_Main - INFO - Общее время: 0.2243 секунд
2025-12-01 12:48:36 - HFE_Main - INFO -
=====
2025-12-01 12:48:36 - HFE_Main - INFO -
=====
2025-12-01 12:48:36 - HFE_Main - INFO - СРАВНЕНИЕ РЕЗУЛЬТАТОВ
2025-12-01 12:48:36 - HFE_Main - INFO -
=====
2025-12-01 12:48:36 - HFE_Main - INFO - [OK] BASE : 2.0812 сек
2025-12-01 12:48:36 - HFE_Main - INFO - [OK] CPU : 5.8466 сек
(ускорение: 0.36x)
2025-12-01 12:48:36 - HFE_Main - INFO - [OK] GPU : 0.2243 сек
(ускорение: 9.28x)
2025-12-01 12:48:36 - HFE_Main - INFO -
=====
2025-12-01 12:48:36 - HFE_Main - INFO - РАБОТА ЗАВЕРШЕНА
2025-12-01 12:48:36 - HFE_Main - INFO -
=====
```

Размер данных: 4096 байт

2025-12-01 12:48:38 - HFE_Main - INFO -

```
=====
2025-12-01 12:48:38 - HFE_Main - INFO - HFE Крипtosистема
2025-12-01 12:48:38 - HFE_Main - INFO -
=====
2025-12-01 12:48:38 - HFE_Main - INFO - Параметры: n=8, d=3, seed=42
2025-12-01 12:48:38 - HFE_Main - INFO - Размер тестовых данных: 4096 байт
2025-12-01 12:48:38 - HFE_Main - INFO - Режим работы: all
2025-12-01 12:48:38 - HFE_Main - INFO - Тестовые данные сгенерированы:
390c8c7d7247342cd8100f2f6f770d65d670e58e...
2025-12-01 12:48:38 - HFE_Main - INFO -
=====
2025-12-01 12:48:38 - HFE_Main - INFO - ЗАПУСК ОБЫЧНОЙ ВЕРСИИ (без
распараллеливания)
2025-12-01 12:48:38 - HFE_Main - INFO -
=====
2025-12-01 12:48:38 - HFE_Main - INFO - === Бенчмарк для режима: Обычная
версия ===
2025-12-01 12:48:38 - HFE_Main - INFO - Размер данных: 4096 байт
2025-12-01 12:48:38 - HFE_Main - INFO - Время шифрования: 0.0645 секунд
2025-12-01 12:48:38 - HFE_Main - INFO - Скорость шифрования: 63489.94
байт/сек
2025-12-01 12:48:42 - HFE_Main - INFO - Время расшифрования: 4.1258 секунд
2025-12-01 12:48:42 - HFE_Main - INFO - Скорость расшифрования: 992.79
байт/сек
2025-12-01 12:48:42 - HFE_Main - INFO - [OK] Расшифрование успешно: данные
совпадают с оригиналом
2025-12-01 12:48:42 - HFE_Main - INFO - Общее время: 4.1903 секунд
2025-12-01 12:48:42 - HFE_Main - INFO -
=====
2025-12-01 12:48:42 - HFE_Main - INFO -
=====
2025-12-01 12:48:42 - HFE_Main - INFO - ЗАПУСК CPU-ПАРАЛЛЕЛЬНОЙ ВЕРСИИ
2025-12-01 12:48:42 - HFE_Main - INFO -
=====
2025-12-01 12:48:42 - HFE_Main - INFO - === Бенчмарк для режима: CPU-
параллельная версия ===
2025-12-01 12:48:42 - HFE_Main - INFO - Размер данных: 4096 байт
2025-12-01 12:48:45 - HFE_Main - INFO - Время шифрования: 2.6880 секунд
2025-12-01 12:48:45 - HFE_Main - INFO - Скорость шифрования: 1523.81 байт/сек
2025-12-01 12:48:48 - HFE_Main - INFO - Время расшифрования: 3.3997 секунд
2025-12-01 12:48:48 - HFE_Main - INFO - Скорость расшифрования: 1204.81
байт/сек
2025-12-01 12:48:48 - HFE_Main - INFO - [OK] Расшифрование успешно: данные
совпадают с оригиналом
2025-12-01 12:48:48 - HFE_Main - INFO - Общее время: 6.0877 секунд
2025-12-01 12:48:48 - HFE_Main - INFO -
=====
2025-12-01 12:48:48 - HFE_Main - INFO -
=====
2025-12-01 12:48:48 - HFE_Main - INFO - ЗАПУСК GPU-ПАРАЛЛЕЛЬНОЙ ВЕРСИИ
```

```
2025-12-01 12:48:48 - HFE_Main - INFO -
=====
2025-12-01 12:48:48 - HFE_Main - INFO - === Бенчмарк для режима: GPU-
параллельная версия ===
2025-12-01 12:48:48 - HFE_Main - INFO - Размер данных: 4096 байт
2025-12-01 12:48:48 - HFE_Main - INFO - Время шифрования: 0.1733 секунд
2025-12-01 12:48:48 - HFE_Main - INFO - Скорость шифрования: 23635.83
байт/сек
2025-12-01 12:48:49 - HFE_Main - INFO - Время расшифрования: 0.1460 секунд
2025-12-01 12:48:49 - HFE_Main - INFO - Скорость расшифрования: 28047.03
байт/сек
2025-12-01 12:48:49 - HFE_Main - INFO - [OK] Расшифрование успешно: данные
совпадают с оригиналом
2025-12-01 12:48:49 - HFE_Main - INFO - Общее время: 0.3193 секунд
2025-12-01 12:48:49 - HFE_Main - INFO -
=====
2025-12-01 12:48:49 - HFE_Main - INFO -
=====
2025-12-01 12:48:49 - HFE_Main - INFO - СРАВНЕНИЕ РЕЗУЛЬТАТОВ
2025-12-01 12:48:49 - HFE_Main - INFO -
=====
2025-12-01 12:48:49 - HFE_Main - INFO - [OK] BASE : 4.1903 сек
2025-12-01 12:48:49 - HFE_Main - INFO - [OK] CPU : 6.0877 сек
(ускорение: 0.69x)
2025-12-01 12:48:49 - HFE_Main - INFO - [OK] GPU : 0.3193 сек
(ускорение: 13.12x)
2025-12-01 12:48:49 - HFE_Main - INFO -
=====
2025-12-01 12:48:49 - HFE_Main - INFO - РАБОТА ЗАВЕРШЕНА
2025-12-01 12:48:49 - HFE_Main - INFO -
=====
```

Размер данных: 8192 байт

```
-----
2025-12-01 12:48:50 - HFE_Main - INFO -
=====
2025-12-01 12:48:50 - HFE_Main - INFO - HFE Криптосистема
2025-12-01 12:48:50 - HFE_Main - INFO -
=====
2025-12-01 12:48:50 - HFE_Main - INFO - Параметры: n=8, d=3, seed=42
2025-12-01 12:48:50 - HFE_Main - INFO - Размер тестовых данных: 8192 байт
2025-12-01 12:48:50 - HFE_Main - INFO - Режим работы: all
2025-12-01 12:48:50 - HFE_Main - INFO - Тестовые данные сгенерированы:
390c8c7d7247342cd8100f2f6f770d65d670e58e...
2025-12-01 12:48:50 - HFE_Main - INFO -
=====
2025-12-01 12:48:50 - HFE_Main - INFO - ЗАПУСК ОБЫЧНОЙ ВЕРСИИ (без
распараллеливания)
2025-12-01 12:48:50 - HFE_Main - INFO -
```

```
=====
2025-12-01 12:48:50 - HFE_Main - INFO - === Бенчмарк для режима: Обычная
версия ===
2025-12-01 12:48:50 - HFE_Main - INFO - Размер данных: 8192 байт
2025-12-01 12:48:50 - HFE_Main - INFO - Время шифрования: 0.1262 секунд
2025-12-01 12:48:50 - HFE_Main - INFO - Скорость шифрования: 64922.55
байт/сек
2025-12-01 12:48:58 - HFE_Main - INFO - Время расшифрования: 8.0213 секунд
2025-12-01 12:48:58 - HFE_Main - INFO - Скорость расшифрования: 1021.28
байт/сек
2025-12-01 12:48:58 - HFE_Main - INFO - [OK] Расшифрование успешно: данные
совпадают с оригиналом
2025-12-01 12:48:58 - HFE_Main - INFO - Общее время: 8.1475 секунд
2025-12-01 12:48:58 - HFE_Main - INFO -
=====
2025-12-01 12:48:58 - HFE_Main - INFO -
=====
2025-12-01 12:48:58 - HFE_Main - INFO - ЗАПУСК CPU-ПАРАЛЛЕЛЬНОЙ ВЕРСИИ
2025-12-01 12:48:58 - HFE_Main - INFO -
=====
2025-12-01 12:48:58 - HFE_Main - INFO - === Бенчмарк для режима: CPU-
параллельная версия ===
2025-12-01 12:48:58 - HFE_Main - INFO - Размер данных: 8192 байт
2025-12-01 12:49:01 - HFE_Main - INFO - Время шифрования: 2.5545 секунд
2025-12-01 12:49:01 - HFE_Main - INFO - Скорость шифрования: 3206.85 байт/сек
2025-12-01 12:49:05 - HFE_Main - INFO - Время расшифрования: 3.6552 секунд
2025-12-01 12:49:05 - HFE_Main - INFO - Скорость расшифрования: 2241.18
байт/сек
2025-12-01 12:49:05 - HFE_Main - INFO - [OK] Расшифрование успешно: данные
совпадают с оригиналом
2025-12-01 12:49:05 - HFE_Main - INFO - Общее время: 6.2097 секунд
2025-12-01 12:49:05 - HFE_Main - INFO -
=====
2025-12-01 12:49:05 - HFE_Main - INFO -
=====
2025-12-01 12:49:05 - HFE_Main - INFO - ЗАПУСК GPU-ПАРАЛЛЕЛЬНОЙ ВЕРСИИ
2025-12-01 12:49:05 - HFE_Main - INFO -
=====
2025-12-01 12:49:05 - HFE_Main - INFO - === Бенчмарк для режима: GPU-
параллельная версия ===
2025-12-01 12:49:05 - HFE_Main - INFO - Размер данных: 8192 байт
2025-12-01 12:49:05 - HFE_Main - INFO - Время шифрования: 0.2330 секунд
2025-12-01 12:49:05 - HFE_Main - INFO - Скорость шифрования: 35165.67
байт/сек
2025-12-01 12:49:05 - HFE_Main - INFO - Время расшифрования: 0.2658 секунд
2025-12-01 12:49:05 - HFE_Main - INFO - Скорость расшифрования: 30820.66
байт/сек
2025-12-01 12:49:05 - HFE_Main - INFO - [OK] Расшифрование успешно: данные
совпадают с оригиналом
2025-12-01 12:49:05 - HFE_Main - INFO - Общее время: 0.4988 секунд
```

```
2025-12-01 12:49:05 - HFE_Main - INFO -
=====
2025-12-01 12:49:05 - HFE_Main - INFO -
=====
2025-12-01 12:49:05 - HFE_Main - INFO - СРАВНЕНИЕ РЕЗУЛЬТАТОВ
2025-12-01 12:49:05 - HFE_Main - INFO -
=====
2025-12-01 12:49:05 - HFE_Main - INFO - [OK] BASE : 8.1475 сек
2025-12-01 12:49:05 - HFE_Main - INFO - [OK] CPU : 6.2097 сек
(ускорение: 1.31x)
2025-12-01 12:49:05 - HFE_Main - INFO - [OK] GPU : 0.4988 сек
(ускорение: 16.34x)
2025-12-01 12:49:05 - HFE_Main - INFO -
=====
2025-12-01 12:49:05 - HFE_Main - INFO - РАБОТА ЗАВЕРШЕНА
2025-12-01 12:49:05 - HFE_Main - INFO -
=====
```

Размер данных: 16384 байт

```
-----
2025-12-01 12:49:07 - HFE_Main - INFO -
=====
2025-12-01 12:49:07 - HFE_Main - INFO - HFE Криптосистема
2025-12-01 12:49:07 - HFE_Main - INFO -
=====
2025-12-01 12:49:07 - HFE_Main - INFO - Параметры: n=8, d=3, seed=42
2025-12-01 12:49:07 - HFE_Main - INFO - Размер тестовых данных: 16384 байт
2025-12-01 12:49:07 - HFE_Main - INFO - Режим работы: all
2025-12-01 12:49:07 - HFE_Main - INFO - Тестовые данные сгенерированы:
390c8c7d7247342cd8100f2f6f770d65d670e58e...
2025-12-01 12:49:07 - HFE_Main - INFO -
=====
2025-12-01 12:49:07 - HFE_Main - INFO - ЗАПУСК ОБЫЧНОЙ ВЕРСИИ (без
распараллеливания)
2025-12-01 12:49:07 - HFE_Main - INFO -
=====
2025-12-01 12:49:07 - HFE_Main - INFO - === Бенчмарк для режима: Обычная
версия ===
2025-12-01 12:49:07 - HFE_Main - INFO - Размер данных: 16384 байт
2025-12-01 12:49:07 - HFE_Main - INFO - Время шифрования: 0.2685 секунд
2025-12-01 12:49:07 - HFE_Main - INFO - Скорость шифрования: 61015.26
байт/сек
2025-12-01 12:49:23 - HFE_Main - INFO - Время расшифрования: 15.7298 секунд
2025-12-01 12:49:23 - HFE_Main - INFO - Скорость расшифрования: 1041.59
байт/сек
2025-12-01 12:49:23 - HFE_Main - INFO - [OK] Расшифрование успешно: данные
совпадают с оригиналом
2025-12-01 12:49:23 - HFE_Main - INFO - Общее время: 15.9983 секунд
2025-12-01 12:49:23 - HFE_Main - INFO -
```

```
=====
2025-12-01 12:49:23 - HFE_Main - INFO -
=====
2025-12-01 12:49:23 - HFE_Main - INFO - ЗАПУСК СРУ-ПАРАЛЛЕЛЬНОЙ ВЕРСИИ
2025-12-01 12:49:23 - HFE_Main - INFO -
=====
2025-12-01 12:49:23 - HFE_Main - INFO - === Бенчмарк для режима: CPU-
параллельная версия ===
2025-12-01 12:49:23 - HFE_Main - INFO - Размер данных: 16384 байт
2025-12-01 12:49:25 - HFE_Main - INFO - Время шифрования: 2.7646 секунд
2025-12-01 12:49:25 - HFE_Main - INFO - Скорость шифрования: 5926.45 байт/сек
2025-12-01 12:49:30 - HFE_Main - INFO - Время расшифрования: 4.3341 секунд
2025-12-01 12:49:30 - HFE_Main - INFO - Скорость расшифрования: 3780.28
байт/сек
2025-12-01 12:49:30 - HFE_Main - INFO - [OK] Расшифрование успешно: данные
совпадают с оригиналом
2025-12-01 12:49:30 - HFE_Main - INFO - Общее время: 7.0986 секунд
2025-12-01 12:49:30 - HFE_Main - INFO -
=====
2025-12-01 12:49:30 - HFE_Main - INFO -
=====
2025-12-01 12:49:30 - HFE_Main - INFO - ЗАПУСК GPU-ПАРАЛЛЕЛЬНОЙ ВЕРСИИ
2025-12-01 12:49:30 - HFE_Main - INFO -
=====
2025-12-01 12:49:30 - HFE_Main - INFO - === Бенчмарк для режима: GPU-
параллельная версия ===
2025-12-01 12:49:30 - HFE_Main - INFO - Размер данных: 16384 байт
2025-12-01 12:49:30 - HFE_Main - INFO - Время шифрования: 0.3713 секунд
2025-12-01 12:49:30 - HFE_Main - INFO - Скорость шифрования: 44127.92
байт/сек
2025-12-01 12:49:31 - HFE_Main - INFO - Время расшифрования: 0.5346 секунд
2025-12-01 12:49:31 - HFE_Main - INFO - Скорость расшифрования: 30646.41
байт/сек
2025-12-01 12:49:31 - HFE_Main - INFO - [OK] Расшифрование успешно: данные
совпадают с оригиналом
2025-12-01 12:49:31 - HFE_Main - INFO - Общее время: 0.9059 секунд
2025-12-01 12:49:31 - HFE_Main - INFO -
=====
2025-12-01 12:49:31 - HFE_Main - INFO -
=====
2025-12-01 12:49:31 - HFE_Main - INFO - СРАВНЕНИЕ РЕЗУЛЬТАТОВ
2025-12-01 12:49:31 - HFE_Main - INFO -
=====
2025-12-01 12:49:31 - HFE_Main - INFO - [OK] BASE : 15.9983 сек
2025-12-01 12:49:31 - HFE_Main - INFO - [OK] CPU : 7.0986 сек
(ускорение: 2.25x)
2025-12-01 12:49:31 - HFE_Main - INFO - [OK] GPU : 0.9059 сек
(ускорение: 17.66x)
2025-12-01 12:49:31 - HFE_Main - INFO -
=====
```

```
2025-12-01 12:49:31 - HFE_Main - INFO - РАБОТА ЗАВЕРШЕНА
2025-12-01 12:49:31 - HFE_Main - INFO -
=====
Размер данных: 32768 байт
-----
2025-12-01 12:49:32 - HFE_Main - INFO -
=====
2025-12-01 12:49:32 - HFE_Main - INFO - HFE Крипtosистема
2025-12-01 12:49:32 - HFE_Main - INFO -
=====
2025-12-01 12:49:32 - HFE_Main - INFO - Параметры: n=8, d=3, seed=42
2025-12-01 12:49:32 - HFE_Main - INFO - Размер тестовых данных: 32768 байт
2025-12-01 12:49:32 - HFE_Main - INFO - Режим работы: all
2025-12-01 12:49:32 - HFE_Main - INFO - Тестовые данные сгенерированы:
390c8c7d7247342cd8100f2f6f770d65d670e58e...
2025-12-01 12:49:32 - HFE_Main - INFO -
=====
2025-12-01 12:49:32 - HFE_Main - INFO - ЗАПУСК ОБЫЧНОЙ ВЕРСИИ (без
распараллеливания)
2025-12-01 12:49:32 - HFE_Main - INFO -
=====
2025-12-01 12:49:32 - HFE_Main - INFO - === Бенчмарк для режима: Обычная
версия ===
2025-12-01 12:49:32 - HFE_Main - INFO - Размер данных: 32768 байт
2025-12-01 12:49:33 - HFE_Main - INFO - Время шифрования: 0.4955 секунд
2025-12-01 12:49:33 - HFE_Main - INFO - Скорость шифрования: 66127.76
байт/сек
2025-12-01 12:50:04 - HFE_Main - INFO - Время расшифрования: 31.5803 секунд
2025-12-01 12:50:04 - HFE_Main - INFO - Скорость расшифрования: 1037.61
байт/сек
2025-12-01 12:50:04 - HFE_Main - INFO - [OK] Расшифрование успешно: данные
совпадают с оригиналом
2025-12-01 12:50:04 - HFE_Main - INFO - Общее время: 32.0758 секунд
2025-12-01 12:50:04 - HFE_Main - INFO -
=====
2025-12-01 12:50:04 - HFE_Main - INFO -
=====
2025-12-01 12:50:04 - HFE_Main - INFO - ЗАПУСК CPU-ПАРАЛЛЕЛЬНОЙ ВЕРСИИ
2025-12-01 12:50:04 - HFE_Main - INFO -
=====
2025-12-01 12:50:04 - HFE_Main - INFO - === Бенчмарк для режима: CPU-
параллельная версия ===
2025-12-01 12:50:04 - HFE_Main - INFO - Размер данных: 32768 байт
2025-12-01 12:50:07 - HFE_Main - INFO - Время шифрования: 2.8477 секунд
2025-12-01 12:50:07 - HFE_Main - INFO - Скорость шифрования: 11506.64
байт/сек
2025-12-01 12:50:13 - HFE_Main - INFO - Время расшифрования: 5.8902 секунд
2025-12-01 12:50:13 - HFE_Main - INFO - Скорость расшифрования: 5563.18
```

байт/сек

2025-12-01 12:50:13 - HFE_Main - INFO - [OK] Расшифрование успешно: данные совпадают с оригиналом

2025-12-01 12:50:13 - HFE_Main - INFO - Общее время: 8.7379 секунд

2025-12-01 12:50:13 - HFE_Main - INFO -

=====

2025-12-01 12:50:13 - HFE_Main - INFO -

=====

2025-12-01 12:50:13 - HFE_Main - INFO - ЗАПУСК GPU-ПАРАЛЛЕЛЬНОЙ ВЕРСИИ

2025-12-01 12:50:13 - HFE_Main - INFO -

=====

2025-12-01 12:50:13 - HFE_Main - INFO - === Бенчмарк для режима: GPU-параллельная версия ===

2025-12-01 12:50:13 - HFE_Main - INFO - Размер данных: 32768 байт

2025-12-01 12:50:14 - HFE_Main - INFO - Время шифрования: 0.6286 секунд

2025-12-01 12:50:14 - HFE_Main - INFO - Скорость шифрования: 52130.76 байт/сек

2025-12-01 12:50:15 - HFE_Main - INFO - Время расшифрования: 1.0346 секунд

2025-12-01 12:50:15 - HFE_Main - INFO - Скорость расшифрования: 31671.75 байт/сек

2025-12-01 12:50:15 - HFE_Main - INFO - [OK] Расшифрование успешно: данные совпадают с оригиналом

2025-12-01 12:50:15 - HFE_Main - INFO - Общее время: 1.6632 секунд

2025-12-01 12:50:15 - HFE_Main - INFO -

=====

2025-12-01 12:50:15 - HFE_Main - INFO -

=====

2025-12-01 12:50:15 - HFE_Main - INFO - СРАВНЕНИЕ РЕЗУЛЬТАТОВ

2025-12-01 12:50:15 - HFE_Main - INFO -

=====

2025-12-01 12:50:15 - HFE_Main - INFO - [OK] BASE : 32.0758 сек

2025-12-01 12:50:15 - HFE_Main - INFO - [OK] CPU : 8.7379 сек (ускорение: 3.67x)

2025-12-01 12:50:15 - HFE_Main - INFO - [OK] GPU : 1.6632 сек (ускорение: 19.29x)

2025-12-01 12:50:15 - HFE_Main - INFO -

=====

2025-12-01 12:50:15 - HFE_Main - INFO - РАБОТА ЗАВЕРШЕНА

2025-12-01 12:50:15 - HFE_Main - INFO -

=====

Размер данных: 65536 байт

2025-12-01 12:50:16 - HFE_Main - INFO -

=====

2025-12-01 12:50:16 - HFE_Main - INFO - HFE Криптосистема

2025-12-01 12:50:16 - HFE_Main - INFO -

=====

2025-12-01 12:50:16 - HFE_Main - INFO - Параметры: n=8, d=3, seed=42

```
2025-12-01 12:50:16 - HFE_Main - INFO - Размер тестовых данных: 65536 байт
2025-12-01 12:50:16 - HFE_Main - INFO - Режим работы: all
2025-12-01 12:50:16 - HFE_Main - INFO - Тестовые данные сгенерированы:
390c8c7d7247342cd8100f2f6f770d65d670e58e...
2025-12-01 12:50:16 - HFE_Main - INFO -
=====
2025-12-01 12:50:16 - HFE_Main - INFO - ЗАПУСК ОБЫЧНОЙ ВЕРСИИ (без
распараллеливания)
2025-12-01 12:50:16 - HFE_Main - INFO -
=====
2025-12-01 12:50:16 - HFE_Main - INFO - === Бенчмарк для режима: Обычная
версия ===
2025-12-01 12:50:16 - HFE_Main - INFO - Размер данных: 65536 байт
2025-12-01 12:50:17 - HFE_Main - INFO - Время шифрования: 0.9771 секунд
2025-12-01 12:50:17 - HFE_Main - INFO - Скорость шифрования: 67068.80
байт/сек
2025-12-01 12:51:20 - HFE_Main - INFO - Время расшифрования: 62.5027 секунд
2025-12-01 12:51:20 - HFE_Main - INFO - Скорость расшифрования: 1048.53
байт/сек
2025-12-01 12:51:20 - HFE_Main - INFO - [OK] Расшифрование успешно: данные
совпадают с оригиналом
2025-12-01 12:51:20 - HFE_Main - INFO - Общее время: 63.4799 секунд
2025-12-01 12:51:20 - HFE_Main - INFO -
=====
2025-12-01 12:51:20 - HFE_Main - INFO -
=====
2025-12-01 12:51:20 - HFE_Main - INFO - ЗАПУСК CPU-ПАРАЛЛЕЛЬНОЙ ВЕРСИИ
2025-12-01 12:51:20 - HFE_Main - INFO -
=====
2025-12-01 12:51:20 - HFE_Main - INFO - === Бенчмарк для режима: CPU-
параллельная версия ===
2025-12-01 12:51:20 - HFE_Main - INFO - Размер данных: 65536 байт
2025-12-01 12:51:23 - HFE_Main - INFO - Время шифрования: 2.9841 секунд
2025-12-01 12:51:23 - HFE_Main - INFO - Скорость шифрования: 21961.53
байт/сек
2025-12-01 12:51:31 - HFE_Main - INFO - Время расшифрования: 8.6469 секунд
2025-12-01 12:51:31 - HFE_Main - INFO - Скорость расшифрования: 7579.14
байт/сек
2025-12-01 12:51:31 - HFE_Main - INFO - [OK] Расшифрование успешно: данные
совпадают с оригиналом
2025-12-01 12:51:31 - HFE_Main - INFO - Общее время: 11.6310 секунд
2025-12-01 12:51:31 - HFE_Main - INFO -
=====
2025-12-01 12:51:31 - HFE_Main - INFO -
=====
2025-12-01 12:51:31 - HFE_Main - INFO - ЗАПУСК GPU-ПАРАЛЛЕЛЬНОЙ ВЕРСИИ
2025-12-01 12:51:31 - HFE_Main - INFO -
=====
2025-12-01 12:51:31 - HFE_Main - INFO - === Бенчмарк для режима: GPU-
параллельная версия ===
```

```
2025-12-01 12:51:31 - HFE_Main - INFO - Размер данных: 65536 байт
2025-12-01 12:51:33 - HFE_Main - INFO - Время шифрования: 1.1887 секунд
2025-12-01 12:51:33 - HFE_Main - INFO - Скорость шифрования: 55133.46
байт/сек
2025-12-01 12:51:35 - HFE_Main - INFO - Время расшифрования: 2.0235 секунд
2025-12-01 12:51:35 - HFE_Main - INFO - Скорость расшифрования: 32387.36
байт/сек
2025-12-01 12:51:35 - HFE_Main - INFO - [OK] Расшифрование успешно: данные
совпадают с оригиналом
2025-12-01 12:51:35 - HFE_Main - INFO - Общее время: 3.2122 секунд
2025-12-01 12:51:35 - HFE_Main - INFO -
=====
2025-12-01 12:51:35 - HFE_Main - INFO -
=====
2025-12-01 12:51:35 - HFE_Main - INFO - СРАВНЕНИЕ РЕЗУЛЬТАТОВ
2025-12-01 12:51:35 - HFE_Main - INFO -
=====
2025-12-01 12:51:35 - HFE_Main - INFO - [OK] BASE : 63.4799 сек
2025-12-01 12:51:35 - HFE_Main - INFO - [OK] CPU : 11.6310 сек
(ускорение: 5.46x)
2025-12-01 12:51:35 - HFE_Main - INFO - [OK] GPU : 3.2122 сек
(ускорение: 19.76x)
2025-12-01 12:51:35 - HFE_Main - INFO -
=====
2025-12-01 12:51:35 - HFE_Main - INFO - РАБОТА ЗАВЕРШЕНА
2025-12-01 12:51:35 - HFE_Main - INFO -
=====
```

Размер данных: 131072 байт

```
-----
2025-12-01 12:51:36 - HFE_Main - INFO -
=====
2025-12-01 12:51:36 - HFE_Main - INFO - HFE Криптосистема
2025-12-01 12:51:36 - HFE_Main - INFO -
=====
2025-12-01 12:51:36 - HFE_Main - INFO - Параметры: n=8, d=3, seed=42
2025-12-01 12:51:36 - HFE_Main - INFO - Размер тестовых данных: 131072 байт
2025-12-01 12:51:36 - HFE_Main - INFO - Режим работы: all
2025-12-01 12:51:36 - HFE_Main - INFO - Тестовые данные сгенерированы:
390c8c7d7247342cd8100f2f6f770d65d670e58e...
2025-12-01 12:51:36 - HFE_Main - INFO -
=====
2025-12-01 12:51:36 - HFE_Main - INFO - ЗАПУСК ОБЫЧНОЙ ВЕРСИИ (без
распараллеливания)
2025-12-01 12:51:36 - HFE_Main - INFO -
=====
2025-12-01 12:51:36 - HFE_Main - INFO - === Бенчмарк для режима: Обычная
версия ===
2025-12-01 12:51:36 - HFE_Main - INFO - Размер данных: 131072 байт
```

```
2025-12-01 12:51:38 - HFE_Main - INFO - Время шифрования: 1.9932 секунд
2025-12-01 12:51:38 - HFE_Main - INFO - Скорость шифрования: 65758.19
байт/сек
2025-12-01 12:53:46 - HFE_Main - INFO - Время расшифрования: 127.6098 секунд
2025-12-01 12:53:46 - HFE_Main - INFO - Скорость расшифрования: 1027.13
байт/сек
2025-12-01 12:53:46 - HFE_Main - INFO - [OK] Расшифрование успешно: данные
совпадают с оригиналом
2025-12-01 12:53:46 - HFE_Main - INFO - Общее время: 129.6030 секунд
2025-12-01 12:53:46 - HFE_Main - INFO -
=====
2025-12-01 12:53:46 - HFE_Main - INFO -
=====
2025-12-01 12:53:46 - HFE_Main - INFO - ЗАПУСК CPU-ПАРАЛЛЕЛЬНОЙ ВЕРСИИ
2025-12-01 12:53:46 - HFE_Main - INFO -
=====
2025-12-01 12:53:46 - HFE_Main - INFO - === Бенчмарк для режима: CPU-
параллельная версия ===
2025-12-01 12:53:46 - HFE_Main - INFO - Размер данных: 131072 байт
2025-12-01 12:53:49 - HFE_Main - INFO - Время шифрования: 3.1575 секунд
2025-12-01 12:53:49 - HFE_Main - INFO - Скорость шифрования: 41511.17
байт/сек
2025-12-01 12:54:03 - HFE_Main - INFO - Время расшифрования: 14.3965 секунд
2025-12-01 12:54:03 - HFE_Main - INFO - Скорость расшифрования: 9104.41
байт/сек
2025-12-01 12:54:03 - HFE_Main - INFO - [OK] Расшифрование успешно: данные
совпадают с оригиналом
2025-12-01 12:54:03 - HFE_Main - INFO - Общее время: 17.5541 секунд
2025-12-01 12:54:03 - HFE_Main - INFO -
=====
2025-12-01 12:54:03 - HFE_Main - INFO -
=====
2025-12-01 12:54:03 - HFE_Main - INFO - ЗАПУСК GPU-ПАРАЛЛЕЛЬНОЙ ВЕРСИИ
2025-12-01 12:54:03 - HFE_Main - INFO -
=====
2025-12-01 12:54:03 - HFE_Main - INFO - === Бенчмарк для режима: GPU-
параллельная версия ===
2025-12-01 12:54:03 - HFE_Main - INFO - Размер данных: 131072 байт
2025-12-01 12:54:06 - HFE_Main - INFO - Время шифрования: 2.2703 секунд
2025-12-01 12:54:06 - HFE_Main - INFO - Скорость шифрования: 57733.26
байт/сек
2025-12-01 12:54:10 - HFE_Main - INFO - Время расшифрования: 4.0286 секунд
2025-12-01 12:54:10 - HFE_Main - INFO - Скорость расшифрования: 32535.35
байт/сек
2025-12-01 12:54:10 - HFE_Main - INFO - [OK] Расшифрование успешно: данные
совпадают с оригиналом
2025-12-01 12:54:10 - HFE_Main - INFO - Общее время: 6.2989 секунд
2025-12-01 12:54:10 - HFE_Main - INFO -
=====
2025-12-01 12:54:10 - HFE_Main - INFO -
```

```
=====
2025-12-01 12:54:10 - HFE_Main - INFO - СРАВНЕНИЕ РЕЗУЛЬТАТОВ
2025-12-01 12:54:10 - HFE_Main - INFO -
=====
2025-12-01 12:54:10 - HFE_Main - INFO - [OK] BASE : 129.6030 сек
2025-12-01 12:54:10 - HFE_Main - INFO - [OK] CPU : 17.5541 сек
(ускорение: 7.38x)
2025-12-01 12:54:10 - HFE_Main - INFO - [OK] GPU : 6.2989 сек
(ускорение: 20.58x)
2025-12-01 12:54:10 - HFE_Main - INFO -
=====
2025-12-01 12:54:10 - HFE_Main - INFO - РАБОТА ЗАВЕРШЕНА
2025-12-01 12:54:10 - HFE_Main - INFO -
=====
```