Министерство образования Республики Беларусь
Белорусский государственный университет
НАУЧНО-ИССЛЕДОВАТЕЛЬСКИЙ ИНСТИТУТ
ПРИКЛАДНЫХ ПРОБЛЕМ МАТЕМАТИКИ И ИНФОРМАТИКИ

УТВЕРЖДАЮ
Директор НИИ прикладных проблем
математики и информатики

Ю.С.Харин
«___» _____ 2022 г.

МЕТОДИКА ИСПЫТАНИЙ СРЕДСТВ КРИПТОГРАФИЧЕСКОЙ ЗАЩИТЫ ИНФОРМАЦИИ НА СООТВЕТСТВИЕ ТРЕБОВАНИЯМ ГОСТ 28147-89

МИ.28147.10.01

Листов 27

Минск 2022

Предисловие

Настоящая методика испытаний предназначена для использования в испытательных лабораториях при проведении сертификационных испытаний средств криптографической защиты информации на соответствие требованиям ГОСТ 28147-89 «Система обработки информации. Защита криптографическая. Алгоритм криптографического преобразования».

Содержание

1 Нормативные ссылки

В настоящем документе использованы ссылки на следующие стандарты:

ГОСТ 19.202-78 «Единая система программной документации. Спецификация. Требования к содержанию и оформлению».

ГОСТ 19.401-2000 «Единая система программной документации. Текст программы. Требования к содержанию, оформлению и контролю качества».

ГОСТ 19.402-2000 «Единая система программной документации. Описание программы. Требования к содержанию, оформлению и контролю качества».

ГОСТ 19.504-79 «Единая система программной документации. Руководство программиста. Требования к содержанию и оформлению».

ГОСТ 28147-89 «Система обработки информации. Защита криптографическая. Алгоритм криптографического преобразования».

СТБ 34.101.31-2020 «Информационные технологии и безопасность. Алгоритмы шифрования и контроля целостности».

СТБ 34.101.77-2020 «Информационные технологии и безопасность. Криптографические алгоритмы на основе sponge-функции».

2 Термины, обозначения и сокращения

В настоящем документе применяются термины и обозначения ГОСТ 28147, а также следующие обозначения и сокращения:

Σ^n	множество всех слов длины n в алфавите Σ;
Σ^*	множество всех слов конечной длины в алфавите Σ (включая пустое слово длины 0);
Σ^{n*}	множество всех слов из Σ^*, длина которых кратна n;
$\lvert u\rvert$	длина слова $u \in \Sigma^*$;
$\ldots\mathtt{43210}_{\overleftarrow{16}}$	представление $u \in \{0,1\}^{4*}$ шестнадцатеричным словом, октеты которого записаны справа налево (т.е. в порядке big-endian) и у которого последовательным четырем символам u соответствует один шестнадцатеричный символ (например, $10100010 = \mathtt{A2}_{\overleftarrow{16}}$);
$a \leftarrow u$	присвоение переменной a значения u;
ЕСПД	единая система программной документации;
СКЗИ	средство криптографической защиты информации.

3 Объект и цель испытаний

На испытания представляется средство криптографической защиты информации (СКЗИ), реализующее криптографический алгоритм ГОСТ 28147, и документация на СКЗИ.

Целью испытаний является проверка соответствия объекта испытаний требованиям ГОСТ 28147.

4 Требования к объекту испытаний

К программе объекта испытаний предъявляются следующие требования, подлежащие проверке во время проведения испытаний:

– в программе должны быть точно и полно реализовываны режимы криптографического алгоритма ГОСТ 28147, поддерживаемые объектом испытаний;

– программа, реализующая криптографический алгоритм и требования ГОСТ 28147, не должна содержать недокументированные возможности.

Документация на объект испытаний должна включать документы «Спецификация», «Текст программы» и может включать документы «Описание программы», «Руководство программиста» и другие документы. Документация может быть разработана в соответствии с требованиями единой системы программной документации (ЕСПД).

5 Средства и порядок испытаний

5.1 Общие сведения

Испытания программы состоят из трех этапов:

1 Анализ документации.

2 Тестирование программы.

3 Анализ исходных текстов программы.

Выполнение этапа 1 осуществляется экспертами по анализу документации, выполнение этапа 2 — экспертами по тестированию, а выполнение этапа 3 — экспертами по анализу исходных текстов. К проведению испытаний должно быть привлечено не менее двух экспертов по анализу исходных текстов и один или более эксперт по тестированию. К анализу документации должен быть привлечен, по крайней мере, один эксперт по анализу исходных текстов программ.

По результатам выполнения этапа испытаний эксперт оформляет протокол результатов проверок: протокол анализа документации, протокол тестирования, протокол анализа исходных текстов. В протоколе эксперт делает вывод о соответствии (не соответствии) программы требованиям ГОСТ 28147. Если программа не поддерживает некоторые режимы, определенные в ГОСТ 28147, то в протоколе делается соответствующее примечание. Примеры оформления протоколов приводятся в приложениях А, Б, В. Допускается оформления протоколов в иной форме, но с обязательным указанием результатов по каждой проводимой проверке и вывода о соответствии (не соответствии).

Если в испытуемой программе используются реализации алгоритмов ГОСТ 28147, которые в составе других программ имеют действующие сертификаты соответствия требованиям ГОСТ 28147, то проверки по тестированию и анализу исходных текстов для данных реализаций могут не проводиться. При этом для подтверждения соответствия объекта испытаний требованиям ГОСТ 28147 экспертом оформляется протокол проверки совпадения контрольных характеристик (хэш-значений) файлов реализации испытуемой программы с контрольными характеристиками соответствующих файлов, указанными в сертификатах соответствия.

На основании протоколов результатов проверок оформляется протокол испытаний, обобщающий результаты испытаний программы. В протоколе испытаний вывод о соответ-

ствии программы требованиям ГОСТ 28147 делается тогда и только тогда, когда вывод о соответствии содержится во всех протоколах результатов проверок. Оформление протокола испытаний проводится в соответствии с требованиями технических нормативно-правовых актов в области сертификации продукции, а также документации, применяемой в испытательной лаборатории.

5.2 Анализ документации

Эксперт проводит анализ документации путем проверки соответствия документации программе объекта испытаний. Такой анализ состоит в получении экспертных заключений, касающихся проверки следующих документов:

- спецификация (см. п. 6.1.1);
- текст программы (см. п. 6.1.2);
- описание программы (см. п. 6.1.3);
- руководство программиста (см. п. 6.1.4).

Анализ документов «Описание программы» и «Руководство программиста» производится в случае их наличия.

5.3 Тестирование

Эксперт проводит тестирование путем выполнения испытуемой программы для некоторого набора проверочных входных значений и сравнения полученных результатов с истинными. Истинные результаты, используемые при тестировании, формируются с помощью эталонной реализации.

Эталонной считается реализация, которая ранее успешно прошла сертификационные испытания на соответствие ГОСТ 28147 или которая удовлетворяет следующим условиям:

1 Проведен анализ исходных текстов программ эталонной реализации. К анализу привлекались, по меньшей мере, два независимых эксперта. Использовалась методика анализа исходных текстов, определенная в п. 6.3.

2 Проведено тестирование эталонной реализации. При тестировании использовались две другие независимые реализации. Использовались тесты, определенные в п. 6.2.

Тестированию подлежат режимы криптографического алгоритма, реализованные в программе и определенные в ГОСТ 28147, включая:

- режим простой замены (см. п. 6.2.2);
- режим гаммирования (см. п. 6.2.3);
- режим гаммирования с обратной связью (см. п. 6.2.4);
- режим выработки имитовставки (см. п. 6.2.5).

Если программа не реализует некоторые из режимов, определенных в ГОСТ 28147, то тесты для них не выполняются.

Для организации тестирования в исходные тексты программы допускается вносить изменения и дополнения, касающиеся:

- способа чтения входных данных;
- способа записи выходных данных.

При внесении модификаций в исходные тексты должен быть проведен анализ корректности внесенных изменений.

При успешном выполнении тест возвращает признак `УСПЕХ`, иначе — `ОШИБКА`. Если при тестировании программы для некоторых входных значений получены результаты отличные от истинных значений, то эксперт по тестированию должен указать эти входные

значения программы и результат ее работы, а также, по требованию, результаты промежуточных вычислений экспертам по анализу исходных текстов.

5.4 Анализ исходных текстов

Эксперт проводит анализ исходных текстов путем проверки корректности реализации в испытуемой программе криптографического алгоритма ГОСТ 28147. Такой анализ состоит в получении экспертных заключений, касающихся:

- корректности использования локальных переменных (см. п. 6.3.1);
- корректности использования глобальных переменных (см. п. 6.3.2);
- корректности использования констант (см. п. 6.3.3);
- корректности программной логики функций программы (см. п. 6.3.4);
- корректности вызова стандартных функций (см. п. 6.3.5);
- корректности вызова функций программы (см. п. 6.3.6);
- корректности обработки исключительных ситуаций (см. п. 6.3.7);
- корректности реализации криптографических примитивов (см. п. 6.3.8);
- корректности реализации криптографических алгоритмов (см. п. 6.3.9);
- корректности управления секретными данными (см. п. 6.3.10);
- отсутствия недокументированных возможностей (см. п. 6.3.11).

6 Методы испытаний

6.1 Анализ программной документации

6.1.1 Документ «Спецификация»

При анализе документа «Спецификация» эксперт проверяет, что в нем указаны компоненты и документация, представляемые на испытания.

Если документ «Спецификация» разработан в соответствии с требованиями ЕСПД, то эксперт проверяет, что содержание и оформление документа соответствует ГОСТ 19.202.

6.1.2 Документ «Текст программы»

При анализе документа «Текст программы» эксперт проверяет, что исходные тексты программы, реализующие определенные в ГОСТ 28147 режимы криптографического алгоритма, представлены полностью и в виде, который использовался при сборке программы.

Если документ «Текст программы» разработан в соответствии с требованиями ЕСПД, то эксперт проверяет, что содержание и оформление документа соответствует ГОСТ 19.401.

6.1.3 Документ «Описание программы»

При анализе документа «Описание программы» эксперт проверяет выполнение следующих требований:

– в документе должна быть указана информация, однозначно идентифицирующая вызываемые стандартные функции (версия компилятора, используемые стандартные библиотеки и т.п.);

– документ должен определять программные модули, реализующие определенные в ГОСТ 28147 режимы криптографического алгоритма;

– описание программы в терминах программных модулей должно соответствовать исходным текстам программы.

Если документ «Описание программы» разработан в соответствии с требованиями ЕСПД, то эксперт проверяет, что содержание и оформление документа соответствует ГОСТ 19.402.

6.1.4 Документ «Руководство программиста»

При анализе документа «Руководство программиста» эксперт проверяет выполнение следующих требований:

– документ должен содержать описание всех доступных для вызова функций, реализующих определенные в ГОСТ 28147 режимы криптографического алгоритма;

– описание функций, реализующих определенные в ГОСТ 28147 режимы криптографического алгоритма, и условия их использования должны соответствовать исходным текстам программы.

При описании в документации функций должны выполняться следующие условия:

– каждая функция должна иметь описание назначения;

– каждый параметр функции должен иметь описание назначения, типа и, при необходимости, диапазона допустимых значений;

– каждая функция должна иметь описание возвращаемого результата с указанием типа;

– каждая функция должна иметь описание условий, при выполнении которых в ходе работы функции могут возникать ошибочные ситуации, требующие специальной обработки;

– в случае если при реализации криптографического алгоритма используется более одной доступной для вызова функции, должны быть указаны порядок и условия вызова данных функций.

Если документ «Руководство программиста» разработан в соответствии с требованиями ЕСПД, то эксперт проверяет, что содержание и оформление документа соответствует ГОСТ 19.504.

6.2 Тестирование

6.2.1 Блоки подстановки

При тестировании реализации алгоритма криптографического преобразования используются блоки подстановки из таблиц 1 и 2, а также биективные блоки подстановки, сгенерированные псевдослучайным образом. В таблицах представлены значения $y = K_i(x)$, где $x, y \in \{0,1\}^4$. Для x, y используется шестнадцатеричное представление слов.

Таблица 1 — Блок подстановка из приложения А к ГОСТ Р 34.10-94

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
K_1	4	A	9	2	D	8	0	E	6	B	1	C	7	F	5	3
K_2	E	B	4	C	6	D	F	A	2	3	8	1	0	7	5	9
K_3	5	8	1	D	A	3	4	2	E	F	C	7	6	0	9	B
K_4	7	D	A	1	0	8	9	F	E	4	6	C	B	2	5	3
K_5	6	C	7	1	5	F	D	8	4	A	9	E	0	3	B	2
K_6	4	B	A	0	7	2	1	D	3	6	8	5	9	C	F	E
K_7	D	B	4	1	3	F	5	9	0	A	E	7	6	8	2	C
K_8	1	F	D	0	5	7	A	4	9	2	3	E	6	B	8	C

Таблица 2 — Блок подстановки из книги А.В. Домашева и др.

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
K_1	4	2	F	5	9	1	0	8	E	3	B	C	D	7	A	6
K_2	C	9	F	E	8	1	3	A	2	7	4	D	6	0	B	5
K_3	D	8	E	C	7	3	9	A	1	5	2	4	6	F	0	B
K_4	E	9	B	2	5	F	7	1	0	D	C	6	A	4	3	8
K_5	3	E	5	9	6	8	0	D	A	B	7	C	2	1	F	4
K_6	8	F	6	B	1	9	C	5	D	3	7	A	0	E	2	4
K_7	9	B	C	0	3	6	7	5	4	8	E	F	1	A	2	D
K_8	C	6	5	2	B	0	9	D	3	E	7	A	F	4	1	8

Примечание — Блок подстановки из таблицы 2 указан в книге А.В. Домашева и др. «Программирование алгоритмов защиты информации». — М.: Нолидж, 2002. — 416 с.

6.2.2 Режим простой замены

При тестировании реализации режима простой замены выполняются тесты GOST.ECB.1 – GOST.ECB.7.

Входными данными тестов являются блок подстановки K, ключ $X \in \{0,1\}^{256}$ и сообщение $M \in \{0,1\}^{64*}$.

В тестах для хранения результата зашифрования M на X используются слова $E, E' \in \{0,1\}^{|X|}$, а для хранения результата расшифрования E на X — слово $M' \in \{0,1\}^{|E|}$.

Тест GOST.ECB.1

1 Задать блок подстановки K из таблицы 1.

2 Задать ключ:

$$X \leftarrow \texttt{110C733D 0D166568 130E7474 06417967}\ \texttt{1D00626E 161A2065 090D326C 4D393320}_{\overleftarrow{16}}.$$

3 Задать сообщение длины 8 октета:

$$M \leftarrow \texttt{00000000 00000000}_{\overleftarrow{16}}.$$

4 Испытуемой реализацией выполнить зашифрование M и сохранить результат в E.

5 Если

$$E = \texttt{5203EBC8 5D9BCFFD}_{\overleftarrow{16}},$$

то возвратить `УСПЕХ`, иначе — `ОШИБКА`.

Тест GOST.ECB.2

1 Задать блок подстановки K из таблицы 1.

2 Задать ключ:

$$X \leftarrow \texttt{CF68D956 9AA09C1C 8C3B417D 658C24E3}\ \texttt{50428833 59DE3D15 6776A6C1 A4248734}_{\overleftarrow{16}}.$$

3 Задать сообщение длины 8 октета:

$$M \leftarrow \texttt{561C7DE3 3315C034}_{\overleftarrow{16}}.$$

4 Испытуемой реализацией выполнить зашифрование M и сохранить результат в E.
5 Если

$$E = \texttt{3CD1602D DD783E86}_{\overleftarrow{16}},$$

то возвратить `УСПЕХ`, иначе — `ОШИБКА`.

Тест GOST.ECB.3

1 Задать блок подстановки K из таблицы 2.
2 Задать ключ:

$$X \leftarrow \begin{array}{l}\texttt{FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF}\\ \texttt{44444444 33333333 22222222 11111111}_{\overleftarrow{16}}.\end{array}$$

3 Задать сообщение длины 8 октета:

$$M \leftarrow \texttt{AAAAAAAA 55555555}_{\overleftarrow{16}}.$$

4 Испытуемой реализацией выполнить зашифрование M и сохранить результат в E.
5 Если

$$E = \texttt{C4F6F857 3113A05E}_{\overleftarrow{16}},$$

то возвратить `УСПЕХ`, иначе — `ОШИБКА`.

Тест GOST.ECB.4

1 Псевдослучайным методом сгенерировать биективную таблицу узлов замены K.
2 Для $i = 1, 2, \ldots, 10000$ выполнить:
1) псевдослучайным методом сгенерировать ключ X;
2) псевдослучайным методом сгенерировать сообщение M длины 2048 октета;
3) испытуемой реализацией выполнить зашифрование M и сохранить результат в E;
4) испытуемой реализацией выполнить расшифрование E и сохранить результат в M';
5) если $M \neq M'$, то возвратить `ОШИБКА`.
3 Возвратить `УСПЕХ`.

Тест GOST.ECB.5

1 Псевдослучайным методом сгенерировать биективную таблицу узлов замены K.
2 Для $i = 1, 2, \ldots, 10000$ выполнить:
1) псевдослучайным методом сгенерировать ключ X;
2) псевдослучайным методом сгенерировать сообщение M длины 2048 октета;
3) испытуемой реализацией выполнить зашифрование M и сохранить результат в E;
4) эталонной реализацией выполнить зашифрование M и сохранить результат в E';
5) если $E \neq E'$, то возвратить `ОШИБКА`.
3 Возвратить `УСПЕХ`.

Тест GOST.ECB.6

1 Псевдослучайным методом сгенерировать биективную таблицу узлов замены K.
2 Псевдослучайным методом сгенерировать слова $X_1, X_2, X_3, X_4 \in \{0,1\}^{32}$ и задать ключ $X \leftarrow X_1\|X_2\|X_3\|X_4\|X_4\|X_3\|X_2\|X_1$.
3 Псевдослучайным методом сгенерировать сообщение M длины 8 октета.
4 Испытуемой реализацией выполнить зашифрование M и сохранить результат в E.
5 Испытуемой реализацией выполнить зашифрование E и сохранить результат в E'.
6 Если $M = E'$, то возвратить `УСПЕХ`, иначе — `ОШИБКА`.

Тест GOST.ECB.7

1 Псевдослучайным методом сгенерировать биективную таблицу узлов замены K.
2 Псевдослучайным методом сгенерировать ключ $X = X_1\|X_2\|X_3\|X_4\|X_5\|X_6\|X_7\|X_8$, где $X_i \in \{0,1\}^{32}$.
3 Псевдослучайным методом сгенерировать сообщение $M = M_1\|M_2$, где $M_i \in \{0,1\}^{32}$.
4 Испытуемой реализацией выполнить зашифрование M на X и сохранить результат в $E = E_1\|E_2$, где $E_i \in \{0,1\}^{32}$
5 Задать ключ $X' \leftarrow X'_1\|X'_2\|X'_3\|X'_4\|X'_5\|X'_6\|X'_7\|X'_8$, где X'_i получено из X_i инвертированием старшего разряда.
6 Задать сообщение $M' \leftarrow M'_1\|M'_2$, где M'_i получено из M_i инвертированием старшего разряда.
7 Испытуемой реализацией выполнить зашифрование M' на X' и сохранить результат в $E' = E'_1\|E'_2$, где $E'_i \in \{0,1\}^{32}$
8 Если для $i = 1, 2$ значения E_i и E'_i отличаются только в старшем разряде, то возвратить `УСПЕХ`, иначе — `ОШИБКА`.

6.2.3 Режим гаммирования

При тестировании реализации режима гаммирования выполняются тесты GOST.CTR.1 – GOST.CTR.4.

Входными данными тестов являются блок подстановки K, ключ $X \in \{0,1\}^{256}$, синхропосылка $S \in \{0,1\}^{256}$ и сообщение $M \in \{0,1\}^{8*}$.

В тестах для хранения результата зашифрования M на X и S используются слова $E, E' \in \{0,1\}^{|X|}$, а для хранения результата расшифрования E на X и S — слово $M' \in \{0,1\}^{|E|}$.

Тест GOST.CTR.1

1 Задать блок подстановки K из таблицы 1.
2 Задать ключ:

$$X \leftarrow \underbrace{\texttt{00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000}}_{16}.$$

3 Задать синхропосылку:

$$S \leftarrow \texttt{DED0AA48 1BC3D070}_{\overleftarrow{16}}.$$

4 Задать сообщение M, состоящее из 128 октетов $\texttt{00}_{\overleftarrow{16}}$.
5 Испытуемой реализацией выполнить зашифрование M и сохранить результат в E.
6 Если

$$E = \begin{array}{l}\texttt{7C85EAA6 6DCECC0F 9EF54A64 64B64258}\\ \texttt{13BBC727 491886B2 40E72D74 5EA088BC}\\ \texttt{B03A73E8 F88F09D6 1A84470C B622FF2B}\\ \texttt{4C5847B0 D7997922 7E59C6AD E20DDBF8}\\ \texttt{D0B0DE24 AF61C0C3 C4D7DC0B B0A1CBBD}\\ \texttt{0AA2F0EC 9B1943B0 CB09DCFF 9FAB77A2}\\ \texttt{F785BE0C 070CED58 DBEF000E 1ABFBE67}\\ \texttt{6B926A41 C201ACEC 406D077C 605A2C19}_{\overleftarrow{16}},\end{array}$$

то возвратить `УСПЕХ`, иначе — `ОШИБКА`.

Тест GOST.CTR.2

1 Задать блок подстановки K из таблицы 2.
2 Задать ключ:

$$X \leftarrow \begin{array}{l}\texttt{FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF}\\ \texttt{00000000 00000000 00000000 00000000}_{\overleftarrow{16}}.\end{array}$$

3 Задать синхропосылку:

$$S \leftarrow \texttt{22222222 11111111}_{\overleftarrow{16}}.$$

4 Задать сообщение:

$$M \leftarrow \texttt{CCCCCCCC 33333333 33333333 CCCCCCCC}_{\overleftarrow{16}}.$$

5 Испытуемой реализацией выполнить зашифрование M и сохранить результат в E.
6 Если

$$E = \texttt{185E2719 AD763ABC 7C85DF99 936A448D}_{\overleftarrow{16}},$$

то возвратить `УСПЕХ`, иначе — `ОШИБКА`.

Тест GOST.CTR.3

1 Псевдослучайным методом сгенерировать биективную таблицу узлов замены K.
2 Для $i = 1, 2, \ldots, 10000$ выполнить:
1) псевдослучайным методом сгенерировать ключ X;
2) псевдослучайным методом сгенерировать синхропосылку S;
3) псевдослучайным методом сгенерировать сообщение M длины 2048 октета;

4) испытуемой реализацией выполнить зашифрование M и сохранить результат в E;
5) испытуемой реализацией выполнить расшифрование E и сохранить результат в M';
6) если $M \neq M'$, то возвратить `ОШИБКА`.
3 Возвратить `УСПЕХ`.

Тест GOST.CTR.4

1 Псевдослучайным методом сгенерировать биективную таблицу узлов замены K.
2 Для $i = 1, 2, \ldots, 10000$ выполнить:
1) псевдослучайным методом сгенерировать ключ X;
2) псевдослучайным методом сгенерировать синхропосылку S;
3) псевдослучайным методом сгенерировать сообщение M длины 2048 октета;
4) испытуемой реализацией выполнить зашифрование M и сохранить результат в E;
5) эталонной реализацией выполнить зашифрование M и сохранить результат в E';
6) если $E \neq E'$, то возвратить `ОШИБКА`.
3 Возвратить `УСПЕХ`.

6.2.4 Режим гаммирования с обратной связью

При тестировании реализации режима гаммирования с обратной связью выполняются тесты GOST.CFB.1 – GOST.CFB.4.

Входными данными тестов являются блок подстановки K, ключ $X \in \{0,1\}^{256}$, синхропосылка $S \in \{0,1\}^{256}$ и сообщение $M \in \{0,1\}^{8*}$.

В тестах для хранения результата зашифрования M на X и S используются слова $E, E' \in \{0,1\}^{|X|}$, а для хранения результата расшифрования E на X и S — слово $M' \in \{0,1\}^{|E|}$.

Тест GOST.CFB.1

1 Задать блок подстановки K из таблицы 1.
2 Задать ключ:

$$X \leftarrow \texttt{733D2C20 65686573 74746769 79676120 626E7373 20657369 326C6568 33206D54}_{\overleftarrow{16}}.$$

3 Задать синхропосылку:

$$S \leftarrow \texttt{00000000 00000000}_{\overleftarrow{16}}.$$

4 Задать сообщение:

$$M \leftarrow \begin{array}{l}\texttt{42ABBCCE 32BC0B1B 42ABBCCE 32BC0B1B}\\ \texttt{42ABBCCE 32BC0B1B 42ABBCCE 32BC0B1B}\\ \texttt{42ABBCCE 32BC0B1B 42ABBCCE 32BC0B1B}\\ \texttt{42ABBCCE 32BC0B1B 42ABBCCE 32BC0B1B}\\ \texttt{42ABBCCE 32BC0B1B 42ABBCCE 32BC0B1B}\\ \texttt{42ABBCCE 32BC0B1B 42ABBCCE 32BC0B1B}\\ \texttt{42ABBCCE 32BC0B1B 42ABBCCE 32BC0B1B}\\ \texttt{42ABBCCE 32BC0B1B 42ABBCCE 32BC0B1B}_{\overleftarrow{16}}.\end{array}$$

5 Испытуемой реализацией выполнить зашифрование M и сохранить результат в E.

6 Если

$$E = \begin{array}{l}\texttt{00000000 00000000 00000000 00000000}\\ \texttt{00000000 00000000 00000000 00000000}\\ \texttt{00000000 00000000 00000000 00000000}\\ \texttt{00000000 00000000 00000000 00000000}\\ \texttt{00000000 00000000 00000000 00000000}\\ \texttt{00000000 00000000 00000000 00000000}\\ \texttt{00000000 00000000 00000000 00000000}\\ \texttt{00000000 00000000 00000000 00000000}_{\overleftarrow{16}},\end{array}$$

то возвратить `УСПЕХ`, иначе — `ОШИБКА`.

Тест GOST.CFB.2

1 Задать блок подстановки K из таблицы 2.

2 Задать ключ:

$$X \leftarrow \begin{array}{l}\texttt{FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF}\\ \texttt{00000001 00000001 00000001 00000001}_{\overleftarrow{16}}.\end{array}$$

3 Задать синхропосылку:

$$S \leftarrow \texttt{47E3A8FF C3A7802A}_{\overleftarrow{16}}.$$

4 Задать сообщение:

$$M \leftarrow \texttt{CCCCCCCC 00000000 AAAAAAAA 55555555}_{\overleftarrow{16}}.$$

5 Испытуемой реализацией выполнить зашифрование M и сохранить результат в E.

6 Если

$$E = \texttt{7E336363 BB42CB56 099E6FA4 807F0572}_{\overleftarrow{16}},$$

то возвратить `УСПЕХ`, иначе — `ОШИБКА`.

Тест GOST.CFB.3

1 Псевдослучайным методом сгенерировать биективную таблицу узлов замены K.
2 Для $i = 1, 2, \ldots, 10000$ выполнить:
1) псевдослучайным методом сгенерировать ключ X;
2) псевдослучайным методом сгенерировать синхропосылку S;
3) псевдослучайным методом сгенерировать сообщение M длины 2048 октета;
4) испытуемой реализацией выполнить зашифрование M и сохранить результат в E;
5) испытуемой реализацией выполнить расшифрование E и сохранить результат в M';
6) если $M \neq M'$, то возвратить `ОШИБКА`.
3 Возвратить `УСПЕХ`.

Тест GOST.CFB.4

1 Псевдослучайным методом сгенерировать биективную таблицу узлов замены K.
2 Для $i = 1, 2, \ldots, 10000$ выполнить:
1) псевдослучайным методом сгенерировать ключ X;
2) псевдослучайным методом сгенерировать синхропосылку S;
3) псевдослучайным методом сгенерировать сообщение M длины 2048 октета;
4) испытуемой реализацией выполнить зашифрование M и сохранить результат в E;
5) эталонной реализацией выполнить зашифрование M и сохранить результат в E';
6) если $E \neq E'$, то возвратить `ОШИБКА`.
3 Возвратить `УСПЕХ`.

6.2.5 Режим выработки имитовставки

При тестировании реализации режима выработки имитовставки выполняются тесты GOST.MAC.1, GOST.MAC.2.

Входными данными тестов являются блок подстановки K, ключ $X \in \{0,1\}^{256}$ и сообщение $M \in \{0,1\}^{8*}$.

В тестах для хранения результата выработки имитовставки используются слова $I, I' \in \{0,1\}^{32}$.

Тест GOST.MAC.1

1 Задать блок подстановки K из таблицы 1.
2 Задать ключ:

$$X \leftarrow \texttt{110C733D 0D166568 130E7474 06417967}\ \texttt{1D00626E 161A2065 090D326C 4D393320}_{\overleftarrow{16}}.$$

3 Задать сообщение длины 128 октета:

$$M \leftarrow \begin{array}{l}\texttt{7436E9EA CF0C0F6B 7436E9EA CF0C0F6B}\\ \texttt{7436E9EA CF0C0F6B 7436E9EA CF0C0F6B}\\ \texttt{7436E9EA CF0C0F6B 7436E9EA CF0C0F6B}\\ \texttt{7436E9EA CF0C0F6B 7436E9EA CF0C0F6B}\\ \texttt{7436E9EA CF0C0F6B 7436E9EA CF0C0F6B}\\ \texttt{7436E9EA CF0C0F6B 7436E9EA CF0C0F6B}\\ \texttt{7436E9EA CF0C0F6B 7436E9EA CF0C0F6B}\\ \texttt{7436E9EA CF0C0F6B 00000000 00000000}_{\overleftarrow{16}}.\end{array}$$

4 Испытуемой реализацией выполнить выработку имитовставки от M и сохранить результат в I.

5 Если

$$I = \texttt{CF0C0F6B}_{\overleftarrow{16}},$$

то возвратить `УСПЕХ`, иначе — `ОШИБКА`.

Тест GOST.MAC.2

1 Псевдослучайным методом сгенерировать биективную таблицу узлов замены K.

2 Для $i = 1, 2, \ldots, 10000$ выполнить:

1) псевдослучайным методом сгенерировать ключ X;

2) псевдослучайным методом сгенерировать сообщение M длины 2048 октета;

3) испытуемой реализацией выработать имитовставку от M и сохранить результат в I;

4) эталонной реализацией выработать имитовставку от M и сохранить результат в I';

5) если $I \neq I'$, то возвратить `ОШИБКА`.

3 Возвратить `УСПЕХ`.

6.3 Анализ исходных текстов

6.3.1 Корректность использования локальных переменных

Анализ корректности использования локальных переменных проводится для всех функций программы.

Под функцией понимается часть программы, которая выполняет специфические действия и описывается типом возвращаемого значения, именем функции, формальными параметрами. Выполнение функции осуществляется посредством вызова из программы или другой функции. Данному термину в языках программирования соответствуют такие понятия как «функция», «процедура», «метод» и т.п.

Для каждой локальной переменной v функции f эксперт определяет языковые конструкции f, в которых v встречается, и выполняет следующие проверки:

1 При использовании v в левой части оператора присваивания тип присваиваемого значения должен совпадать с типом v, в противном случае эксперт проверяет корректность результата, учитывая стандартные правила преобразования типов, определенные в используемом языке программирования.

2 Перед использованием значения переменной v должна быть выполнена ее инициализация.

3 Обращение на чтение/запись к переменной v должно происходить в пределах установленных для нее границ, в частности, если v является переменной составного типа, то обращение к элементам v должно происходить в пределах заданных размерностей.

4 Если v является переменной вещественного типа, то ее использование в операциях сравнения запрещено.

5 Если память для v выделяется в динамической области, то перед каждым выходом из f динамическая память должна быть освобождена. После освобождения памяти не должно быть языковых конструкций, ссылающихся на нее.

Примечание — В языках программирования, снабженных средствами «сборки мусора», освобождение динамической памяти, выделяемой для локальной переменной, может быть неявным.

6.3.2 Корректность использования глобальных переменных

Для каждой глобальной переменной v эксперт определяет языковые конструкции программы, в которых v встречается. Далее выполняются проверки 1 – 4 из п. 6.3.1 и следующие проверки:

1 Если память для v выделяется в динамической области, то перед каждым выходом из программы динамическая память должна быть освобождена. После освобождения памяти не должно быть языковых конструкций, ссылающихся на нее.

2 Если v может использоваться в многопоточном режиме работы программы, то должны быть реализованы механизмы, обеспечивающие разграничение доступа к v (механизмы синхронизации доступа к глобальной переменной), при этом данные механизмы не должны блокировать доступ к v на неограниченное время.

Примечание – В языках программирования, снабженных средствами «сборки мусора», освобождение динамической памяти, выделяемой для глобальной переменной, может быть неявным.

6.3.3 Корректность использования констант

Эксперт определяет языковые конструкции программы, в которых встречаются константы $C1$ и $C2$, определенные в приложении 2 к ГОСТ 28147. Для каждой языковой конструкции эксперт проверяет, что константы заданы правильно.

6.3.4 Корректность программной логики функций программы

Для каждой функции программы эксперт выполняет следующие проверки:

1 Проверка допустимости переданных параметров и используемых глобальных переменных выполняется до их использования. Проверка может не выполняться, если в документации или в комментариях к функции оговорены ограничения на входные данные, при которых функция работает правильно, и эти ограничения соблюдаются для входных данных во всех вызовах функции.

2 Все заданные варианты условных переходов возможны.

3 Все адреса безусловных переходов доступны.

4 Каждый цикл завершается за конечное число шагов, т.е. завершение цикла гарантировано.

5 После выполнения операторов функции завершение функции гарантировано: достигается одна из точек выхода из функции.

6 Отсутствуют недостижимые участки кода.

7 Цепочки последовательных действий (например, открытие файла, чтение из файла, закрытие файла) корректны. Проверка выполняется, если в функции требуется выполнить некоторое действие, требующее определенной последовательности операций.

6.3.5 Корректность вызова стандартных функций

Эксперт проверяет, что в документации, комментариях исходных текстов программ или конфигурационных файлах указана информация, однозначно идентифицирующая вызываемые стандартные функции (версии компилятора, используемых стандартных библиотек и т.п.).

Для каждого вызова стандартной функции в программе эксперт проверяет:

1 Типы и значения параметров, фактически переданных в функцию, соответствуют типам и допустимым значениям параметров функции, указанным в документации на функцию (с учетом стандартных правил преобразования типов языка программирования).

2 Если в документации на функцию указано, что функция возвращает значение, то проводится анализ корректности использования возвращаемого значения, например, корректность использования в операторе присваивания, допустимость игнорирования возвращаемого значения и т.п.

3 Если в документации на функцию указано, что вызов функции может привести к возникновению исключительной ситуации или ошибки, проверяется наличие и корректность обработки исключительной ситуации.

4 Если в документации на функцию указано, что до и после вызова функции должны выполняться определенные действия, то проверяется наличие и корректность выполнения требуемых действий.

6.3.6 Корректность вызова функций программы

Эксперт проверяет, что в документации или комментариях исходных текстов программ для каждой функции программы указана информация, определяющая:

– допустимые входные параметры и возвращаемые значения функции;

– условия, при выполнении которых в ходе работы функции могут возникать исключительные ситуации (при наличии);

– действия, которые должны выполняться до и(или) после вызова функции (при наличии).

Для каждого вызова функции программы эксперт выполняет следующие проверки:

1 Типы и значения параметров, фактически переданных в функцию, соответствуют типам и допустимым значениям параметров функции (с учетом стандартных правил преобразования типов языка программирования).

2 Если функция возвращает значение, то проводится анализ корректности использования возвращаемого значения, например, корректность использования в операторе присваивания, допустимость игнорирования возвращаемого значения и т.п.

3 Если вызов функции может привести к возникновению исключительной ситуации или ошибки, проверяется наличие и корректность обработки исключительной ситуации.

4 Если до и после вызова функции должны выполняться определенные действия, то проверяется наличие и корректность выполнения требуемых действий.

5 Если функция использует глобальные переменные, то проверяется наличие инициализации данных переменных.

6.3.7 Корректность обработки исключительных ситуаций

Под исключительной ситуацией понимается ошибочная ситуация, возникающая при выполнении программы и требующая специальной обработки. Данному термину в языках программирования соответствует такие понятия как «ошибка», «исключение» и т.п.

Для анализа корректности обработки исключительных ситуаций эксперт формирует список функций, включающий стандартные функции и функции программы, вызов которых может приводить к возникновению исключительной ситуации.

Для каждого вызова функции из составленного списка эксперт проверяет:

1 После каждого вызова функции имеются проверка на случай возникновения исключительной ситуации и соответствующая обработка исключительной ситуации.

2 При проверке и обработке исключительной ситуации учтены все возможные виды исключительных ситуаций, возникновение которых возможно для вызываемой функции.

3 Исключительные ситуации обрабатываются адекватно (возвращаются верные коды ошибок и сообщения об ошибках и т.п.).

6.3.8 Корректность реализации криптографических примитивов

Криптографический примитив — это определенное в ГОСТ 28147 вспомогательное преобразование, являющееся композиционной частью некоторого криптографического алгоритма.

В ГОСТ 28147 определены следующие криптографические примитивы:

- преобразование j-го цикла зашифрования ($j = 1, \ldots, 32$) (п. 2.1.4 ГОСТ 28147);
- преобразование j-го цикла расшифрования ($j = 1, \ldots, 32$) (п. 2.2.3 ГОСТ 28147);
- операция $\oplus$ сложения по модулю 2 (п. 2.1.4 ГОСТ 28147);
- операция $\boxplus$ сложения по модулю 2^{32} (п. 1 приложения 4 к ГОСТ 28147);
- операция $\boxplus'$ сложения по модулю $2^{32} - 1$ (п. 2 приложения 4 к ГОСТ 28147).

Анализируя структуру программы и используя документацию, эксперт формирует список криптографических примитивов, реализованных в программе. Для каждого примитива $g : A \to B$, осуществляющего отображение множества A в множество B, эксперт проверяет:

– наличие реализации примитива g в виде отдельной функции, части функции или композиции нескольких функций;

– тождественность реализации примитива g спецификации;

– отсутствие в g операций, не используемых для реализации примитива (наличие операций, не предусмотренных спецификацией на примитив, отражается в приложении к протоколу результатов анализа исходных текстов).

Допускается, что действие отображения g определено на множестве A^*, которое является подмножеством A. В этом случае эксперт дополнительно проверяет, что при выполнении программы прообразы отображения g всегда являются элементами A^*.

6.3.9 Корректность реализации криптографического алгоритма

В ГОСТ 28147 определены следующие режимы криптографического алгоритма:

- режим простой замены (п. 2 ГОСТ 28147);
- режим гаммирования (п. 3 ГОСТ 28147);
- режим гаммирования с обратной связью (п. 4 ГОСТ 28147);
- режим выработки имитовставки (п. 5 ГОСТ 28147).

Анализируя структуру программы и используя документацию, эксперт формирует список криптографических алгоритмов в различных режимах, реализованных в программе. Для каждого алгоритма $f : X \times \Theta \to Y$, который ставит в соответствие входным данным $x \in X$ и параметру $\theta \in \Theta$ результат криптографического преобразования $y \in Y$, эксперт проверяет наличие соответствующей реализации алгоритма. Затем эксперт определяет множества функций реализации, в которых:

1) задаются параметры $\theta \in \Theta$;
2) задаются входные данные $x \in X$;
3) реализуется отображение f;
4) возвращается результат $y \in Y$.

Данные множества функций обозначаются соответственно F_1, F_2, F_3, F_4. Множества могут пересекаться или совпадать.

Для функций из множества F_1 эксперт проверяет корректность задания параметров $\theta \in \Theta$. При этом допустимым является использование в программном компоненте множества параметров Θ^*, которое является подмножеством Θ. Однако, использованное сужение множества Θ не должно состоять в ограничении области значений секретных параметров.

Для функций из множества F_2 эксперт проверяет корректность задания входных данных $x \in X$. При этом допускается, что множество входных данных X^* алгоритма является подмножеством X. Однако, использованное сужение множества входных данных должно быть оговорено в документации.

Примечание – Программа может обрабатывать не все допустимые входные данные. Например, могут шифроваться сообщения только определенной длины.

Для функций из множества F_3 эксперт проверяет тождественность отображения, реализуемого функциями, спецификации на алгоритм f (при возможных ограничениях на параметры и входные данные, использованные в реализации отображения). Для этого, по результатам анализа элементов множества F_3, составляются использованные в реализации f композиции криптографических примитивов. Затем проверяется тождественность реализованных композиций композициям криптографических примитивов, заданным в спецификации и реализующим анализируемый криптографический алгоритм. Кроме этого, эксперт проводит проверку корректности реализации вспомогательных алгоритмов, использованных в программе и не описанных в спецификации. Если такой анализ провести не удается (алгоритм не описан в документации или описан не полно, без указания использованных источников), то по данному пункту экспертизы выдается отрицательное заключение по причине недостаточности данных. Если использованы простые вспомогательные алгоритмы, призванные оптимизировать выполнение программы и понятные эксперту, то их описание в документации не требуется.

Для функций из множества F_4 эксперт проверяет корректность выдачи результатов $y \in Y$ выполнения криптографического алгоритма. Сужение в реализации алгоритма f множества результатов Y является недопустимым.

6.3.10 Корректность управления секретными данными

Секретные данные — это ключи, параметры и другие данные криптографических алгоритмов, значения которых в соответствии со стандартом или документацией на СКЗИ должны быть защищены от раскрытия, т.е. должны храниться в секрете.

Секретными данными ГОСТ 28147 являются:

– ключ;

– сообщение, подлежащее зашифрованию;

– результат расшифрования;

– сообщение, подлежащее имитозащите, если в соответствии с документацией реализация алгоритма в режиме выработки имитовставки может использоваться для обработки критических данных.

Эксперт проверяет, что секретные данные используются в строгом соответствии с криптографическим алгоритмом. Допускается использование секретных данных во вспомогательных операциях с целью повышения быстродействия программной реализации криптоалгоритма. Другие операции с секретными данными не допускаются.

Эксперт проверяет, что все копии секретных данных в открытом виде уничтожаются при завершении работы с ними, при этом:

– значение секретных данных, размещенное в области памяти глобальной переменной, уничтожается перед каждым выходом из программы;

– значение секретных данных, размещенное в области памяти локальной переменной функции, уничтожается перед каждым выходом из данной функции;

– значение секретных данных, размещенное в динамической памяти, уничтожается перед каждым освобождением динамической памяти.

Примечание – Под уничтожением понимается такое изменение данных, хранящихся в электронных устройствах (оперативная память, память на магнитных носителях и др.), которое предотвращает их последующее восстановление. Например, уничтожение может состоять в записи в области памяти, занимаемой значениями секретных данных, фиксированных или случайно выбранных значений.

6.3.11 Отсутствие недокументированных возможностей

Эксперт определяет отсутствие недокументированных возможностей по результатам проверок, выполненных в п. 6.3.1 – 6.3.10.

Обнаруженные недокументированные возможности отражаются в протоколе анализа исходных текстов или в приложении к нему.

Приложение А
Форма протокола анализа документации

Экз. {Поле 1}

Протокол № {Поле 2} **от** {Поле 3}
результатов анализа документации
объекта испытаний {Поле 4}, реализующего криптографические алгоритмы
согласно ГОСТ 28147–89

1. Документы:

№	Название документа	Номер
1	{Поле 5}	{Поле 6}
2	{Поле 7}	{Поле 8}
3	{Поле 9}	{Поле 10}
4	{Поле 11}	{Поле 12}

2. При анализе документации были выполнены следующие проверки:

№	Название проверки	Отметка о выполнении
1	Проверка документа «Спецификация»	{Поле 13}
2	Проверка документа «Текст программы»	{Поле 13}
3	Проверка документа «Описание программы»	{Поле 13}
4	Проверка документа «Руководство программиста»	{Поле 13}

3. Заключение по результатам анализа документации: документация {Поле 6}, {Поле 8}, {Поле 10}, {Поле 12} соответствует (не соответствует) программе объекта испытаний в части реализации криптографических алгоритмов согласно ГОСТ 28147-89.

Эксперт, {Поле 15} {Поле 16}
{Поле 14}

В поле 1 указывается номер экземпляра протокола.

В поле 2 указывается номер, однозначно идентифицирующий протокол.

В поле 3 указывается дата составления протокола.

В поле 4 указывается название объекта испытаний в соответствии с представленной к испытаниям документацией.

В полях 5 и 6 указываются соответственно полное название документа «Спецификация» и его идентификационный/децимальный номер.

В полях 7 и 8 указываются соответственно полное название документа «Текст программы» и его идентификационный/децимальный номер.

В полях 9 и 10 указываются соответственно полное название документа «Описание программы» и его идентификационный/децимальный номер.

В полях 11 и 12 указываются соответственно полное название документа «Руководство программиста» и его идентификационный/децимальный номер.

В поле 13 указывается результат выполнения проверки: «положительно» — результат проверки положительный, «отрицательно» — результат проверки отрицательный. После завершения анализа документации и заполнения таблицы делается вывод о соответствии (не соответствии) документации программе объекта испытаний в части реализации криптографических алгоритмов согласно ГОСТ 28147. Вывод о соответствии делается только тогда, когда результаты всех проверок являются положительными.

В полях 14 и 16 указываются соответственно должность и Ф. И. О. эксперта.

В поле 15 ставится собственноручная подпись эксперта.

Информация об обнаруженных несоответствиях приводится в протоколе или приложении к протоколу в произвольной форме.

Приложение Б
Форма протокола тестирования

Экз. {Поле 1}

Протокол № {Поле 2} **от** {Поле 3}
результатов тестирования
объекта испытаний {Поле 4}, реализующего криптографические алгоритмы
согласно ГОСТ 28147–89

1. Файлы исходных текстов программ:

№	Имя файла	Хэш-значение
1	{Поле 5}	{Поле 6}
2	{Поле 5}	{Поле 6}
...	...	...

Хэш-значения для файлов вычислены согласно {Поле 7}.

2. В ходе тестирования объекта испытаний были выполнены следующие тесты:

№	Название теста	Отметка о выполнении
1	GOST.ECB.1	{Поле 8}
2	GOST.ECB.2	{Поле 8}
...	...	...

3. Заключение по результатам тестирования: объект испытаний {Поле 4} соответствует (не соответствует) требованиям, установленным в ГОСТ 28147–89.

Эксперт, {Поле 10} {Поле 11}
{Поле 9}

В поле 1 указывается номер экземпляра протокола.

В поле 2 указывается номер, однозначно идентифицирующий протокол.

В поле 3 указывается дата составления протокола.

В поле 4 указывается название объекта испытаний в соответствии с представленной к испытаниям документацией.

В поле 5 указываются имена исходных файлов программ объекта испытаний.

В поле 6 указывается значение функции хэширования для тестируемых файлов, вычисленное в соответствии со стандартом, указанным в поле 7. Разрешается использовать функции хэширования, определенные в СТБ 34.101.31 или СТБ 34.101.77.

В поле 8 указывается результат выполнения теста: «положительно» — тест завершен успешно, «отрицательно» — тест завершен с ошибкой; «не проводился» — тест не проводился, так как программа не поддерживает алгоритм или режим, определенный в тесте.

После завершения тестирования и заполнения таблицы делается вывод о соответствии (не соответствии) программной реализации объекта испытаний ГОСТ 28147. Вывод о соответствии делается только тогда, когда все проводимые тесты выполнены успешно.

В полях 9, 11 указываются соответственно должность и Ф. И. О. эксперта.

В поле 10 ставится собственноручная подпись эксперта.

Приложение В
Форма протокола анализа исходных текстов

Экз. {Поле 1}

Протокол № {Поле 2} **от** {Поле 3}
результатов анализа исходных текстов программ
объекта испытаний {Поле 4}, реализующего криптографические алгоритмы
согласно ГОСТ 28147–89

1. Файлы исходных текстов программ:

№	Имя файла	Хэш-значение
1	{Поле 5}	{Поле 6}
2	{Поле 5}	{Поле 6}
	...	...

Хэш-значения для файлов вычислены согласно {Поле 7}.

2. В ходе анализа исходных текстов программ были выполнены следующие проверки:

№	Название проверки	Результат проверки
1	Корректность использования локальных переменных	{Поле 8}
2	Корректность использования глобальных переменных	{Поле 8}
3	Корректность использования констант	{Поле 8}
4	Корректность программной логики функций программы	{Поле 8}
5	Корректность вызова стандартных функций	{Поле 8}
6	Корректность вызова функций программы	{Поле 8}
7	Корректность обработки исключительных ситуаций	{Поле 8}
8	Корректность реализации криптографических примитивов	{Поле 8}
9	Корректность реализации криптографических алгоритмов	{Поле 8}
10	Корректность управления секретными данными	{Поле 8}
11	Отсутствие недокументированных возможностей	{Поле 8}

3. Заключение по результатам анализа исходных текстов программ: объект испытаний {Поле 4} соответствует требованиям, установленным в ГОСТ 28147–89.

Эксперт, {Поле 9} {Поле 10} {Поле 11}

В поле 1 указывается номер экземпляра протокола.

В поле 2 указывается номер, однозначно идентифицирующий протокол.

В поле 3 указывается дата составления протокола.

В поле 4 указывается название объекта испытаний в соответствии с представленной к испытаниям документацией.

В поле 5 указываются имена исходных файлов программ объекта испытаний.

В поле 6 указывается значение функции хэширования для исходных файлов программ, вычисленное в соответствии со стандартом, указанным в поле 7. Разрешается использовать функции хэширования, определенные в СТБ 34.101.31 или СТБ 34.101.77.

В поле 8 указывается результат выполнения проверки: «положительно» — результат проверки положительный, «отрицательно» — результат проверки отрицательный, «не проводилась» — проверка не требуется по причине специфики реализации программ объекта испытаний (например, в программе не используются глобальные переменные). После завершения анализа исходных текстов программ и заполнения таблицы делается вывод о соответствии (не соответствии) объекта испытаний ГОСТ 28147. Вывод о соответствии делается только тогда, когда результаты всех проводимых проверок являются положительными.

В полях 9, 11 указываются соответственно должность и Ф. И. О. эксперта.

В поле 10 ставится собственноручная подпись эксперта.

Информация об обнаруженных ошибках и недокументированных возможностях приводится в протоколе или приложении к протоколу в произвольной форме и должна включать:

1) описание ошибки или недокументированной возможности;
2) имя файла и номера строк программы, содержащих ошибку.