Закрытое акционерное общество «Актив-софт»
ИНН: 7729361030 / КПП: 772901001 / ОГРН: 1037700094541
115088, Москва, ул. Шарикоподшипниковская, д.1

Микропрограмма Guardant ID

(функциональные характеристики)

В данном документе представлено описание функциональных характеристик программного обеспечения и информация, необходимая для установки и эксплуатации программного обеспечения.

Содержание:

1. Описание Guardant ID

Guardant ID – это устройство, подключаемое к USB-порту компьютера (напрямую или через удлинитель) и предназначенное для использования в качестве персонального идентификатора администратора и пользователей информационной системы.

Классификация устройств по уровню контроля отсутствия недекларированных возможностей (РД НДВ, Гостехкомиссия России, 1999 г.) – НДВ2.

Варианты исполнения устройств:

	**Полноразмерный корпус** Сертифицированный ФСТЭК России электронный USB-идентификатор в стандартном исполнении.
	**Миниатюрный корпус** Сертифицированный ФСТЭК России электронный USB-идентификатор, исполненный в виде микро-токена. Поддерживает весь функционал стандартного полноразмерного устройства. Миниатюрность исполнения позволяет использовать его в компактных устройствах: ноутбуках, нетбуках, планшетах.

В основе Guardant ID лежит микропрограмма. Именно она обеспечивает надежное хранение и обработку данных.

Комплект поставки:

- электронный USB-идентификатор с установленной микропрограммой;
- драйвер для работы с USB-идентификатором
- паспорт на изделие, маркированный специальным защитным знаком ФСТЭК;
- копия Сертификата ФСТЭК России.

2. Установка

Микропрограмма Guardant ID прошивается в устройства на этапе производства и не требует дополнительных настроек.

3. Эксплуатация

3.1. Общие сведения

Данные, которыми оперирует микропрограмма, хранятся в энергонезависимой памяти. Для их защиты используется кодирование с уникальным для каждого устройства паролем.

Для идентификации пользователя при выполнении команд чтения/записи может быть использован PIN-код длиной от 1 до 32 байт. PIN-код можно изменять командой записи, при этом нужно предъявить “старый” PIN-код.

Память ключа может быть инициализирована (полностью обнулена) специальной командой, требующей предъявления особого Master PIN-кода. Master PIN-код задается не этапе производства Guardant ID и не может быть считан или изменен.

Длина PIN-кода, а также режим “проверять PIN-код” задаются в слове конфигурации Guardant ID.

3.2. Стандартные поля памяти Guardant ID

В энергонезависимой памяти Guardant ID содержатся 2 стандартных поля:

Адрес поля в словах	Длина поля в словах	Название поля	Описание поля
MemSizeW-PINLenW-1	16	PIN-код	Поле содержит PIN-код, который может быть проверен при выполнении команд чтения/записи
MemSizeW-1	1	Слово конфигурации	Поле содержит слово конфигурации, содержащее реальную длину PIN-кода и бит “проверять PIN-код”

Здесь MemSizeW – объем памяти Guardant ID в словах, PINLenW – длина PIN-кода в словах. Обратите внимание, что память, отведенная под стандартные поля, никак не защищена от ее случайного изменения. Значения, записываемые в эту область памяти, не проверяются на корректность. Вы должны делать это сами.

Состав слова конфигурации:

Номер бита	Обозначение	Значение
0	fPinUse	1-при выполнении команд чтения/записи PIN-код будет проверяться 0-PIN-код не будет проверяться
1-5	fPinSize	Размер PIN-кода в БАЙТАХ
6-15	Reserved	12 бит зарезервировано

Флаг **fPinUse** указывает, будет ли будет проверяться PIN-код при выполнении команд чтения/записи. Если флаг содержит “1”, предъявленный PIN-код будет проверяться и, если он неверен, будет возвращена соответствующая ошибка. Если флаг содержит “0”, предъявленный PIN-код не будет проверяться, т.е. в этом случае вы можете предъявить любую

последовательность байт. Обратите внимание, что предъявлять PIN-код или произвольную последовательность байт требуется вне зависимости от значения флага.

Битовое поле **fPinSize** задает реальный размер PIN-кода в байтах. Именно такое количество байт из поступившей к Guardant ID последовательности будет проверено, если fPinUse=1. Однако, несмотря на значение, записанное в битовом поле fPinSize, требуется всегда посылать в Guardant ID 32-байтовую последовательность!

3.3. Команды Guardant ID

3.3.1. GetInfo

Назначение:

получение ID данного устройства

Входные данные

Размер в байтах	Описание
1	Код команды ‘0’ (30h)

Выходные данные

Размер в байтах	Описание
1	Код возврата
1	Аппаратная версия Guardant ID (4 бита maj: 4 бита min)
1	№ протокола обмена с Guardant ID
1	№ микропрограммы в Guardant ID
1	№ заказного Guardant ID (‘0’ – стандартный)
1	Размер памяти в 256-байтовых страницах
8	ID (если нет ошибки)

3.3.2. ReadWords

Назначение:

считать из Guardant ID заданное количество слов начиная с заданного адреса

Входные данные

Размер в байтах	Описание
1	Код команды ‘1’ (31h)
32	PIN-код
8	ID, возвращенное командой GetInfo

2	Начальный адрес в памяти, в СЛОВАХ
2	Количество считываемых СЛОВ

Выходные данные

Размер в байтах	Описание
1	Код возврата
1	Аппаратная версия Guardant ID (4 бита maj: 4 бита min)
1	№ протокола обмена с Guardant ID
1	№ микропрограммы в Guardant ID
1	№ заказного Guardant ID (‘0’ – стандартный)
1	Размер памяти в 256-байтовых страницах
N слов	Считанные данные (если нет ошибки)

3.3.3. WriteWords

Назначение:

Записать в Guardant ID заданное количество слов начиная с заданного адреса.

Входные данные

Размер в байтах	Описание
1	Код команды ‘2’ (32h)
32	PIN-код
8	ID, возвращенное командой GetInfo
2	Начальный адрес в памяти, в СЛОВАХ
2	Количество записываемых СЛОВ

Выходные данные

Размер в байтах	Описание
1	Код возврата
1	Аппаратная версия Guardant ID (4 бита maj: 4 бита min)
1	№ протокола обмена с Guardant ID
1	№ микропрограммы в Guardant ID
1	№ заказного Guardant ID (‘0’ – стандартный)
1	Размер памяти в 256-байтовых страницах

3.3.4. Initialize

Назначение:

Инициализировать (заполнить нулями) всю память Guardant ID.

Входные данные

Размер в байтах	Описание
1	Код команды ‘3’ (33h)
32	Master PIN-код
8	ID, возвращенное командой GetInfo

Выходные данные

Размер в байтах	Описание
1	Код возврата
1	Аппаратная версия Guardant ID (4 бита maj: 4 бита min)
1	№ протокола обмена с Guardant ID
1	№ микропрограммы в Guardant ID
1	№ заказного Guardant ID (‘0’ – стандартный)
1	Размер памяти в 256-байтовых страницах

3.3.5. Комментарии

Таким образом, начинать работу с Guardant ID следует с команды GetInfo, которая вернет ID данного Guardant ID. ID задается не этапе производства Guardant ID и не может быть изменен. ID ключа требуется предъявлять во всех командах, кроме GetInfo. Это дает гарантию, что все команды будут относиться именно к тому Guardant ID, который был идентифицирован вами после вызова GetInfo (это важно в случае, если к компьютеру подсоединено несколько Guardant ID). В отличие от PIN-кода, предъявленный ID проверяется всегда.

Выходная информация, начиная с аппаратной версии Guardant ID по размер памяти включительно, возвращается всегда. Остальная информация возвращается только при успешном выполнении команды (если код возврата = 0).

При работе с Guardant ID входные параметры заносятся в соответствующие поля структуры данных, выходные данные возвращаются в этой же структуре

При работе с Guardant ID напрямую входные/выходные данные передаются/принимаются в соответствующих блоках данных, образующих протокол обмена с Guardant ID (подробнее см. в соответствующем разделе).

3.3.6. Коды возврата Guardant ID

Значение	Описание
0	Команда выполнена успешно
1	Guardant ID с указанным PIN-кодом и/или ID не обнаружен

2	Во время выполнения команды произошла ошибка, не идентифицируемая другими кодами возврата. Команда не завершена успешно. Рекомендуется повторить команду.
3	Неверный номер команды и/или неверные параметры. Например, слишком большой адрес и/или длина при чтении/записи.
4	Ошибка записи или верификации при записи. Не все данные записаны успешно. Рекомендуется повторить запись всего участка.
5	Нарушение протокола обмена с Guardant ID. Обычно выдается при работе с Guardant ID напрямую.
6	Guardant ID занят работой с другим клиентом (т.е. выполняет команду, посланную другим приложением). Рекомендуется повторить команду спустя ~1 сек. Обычно выдается при работе с Guardant ID напрямую.
7	Драйвер Guardant ID не поддерживает протокол обмена, по которому работает данный Guardant ID. Требуется обновить драйвер.
8	Драйвер Guardant ID занят работой с другим клиентом (т.е. выполняет команду, посланную другим приложением). Рекомендуется повторить команду спустя ~1 сек.

3.4. Подготовка Guardant ID к работе

Для того чтобы привести Guardant ID в рабочее состояние, нужно проделать следующие действия:

1. Выполнить команду GetInfo для идентификации Guardant ID и получения его ID.
2. Выполнить команду Initialize для инициализации всей памяти Guardant ID. Для инициализации 16 Кбайт памяти требуется ~45 сек.
3. Командой WriteWords записать PIN-код.
4. Сформировать новое слово конфигурации и записать его командой WriteWords.

3.5. Рекомендации по повышению уровня защищенности

Самое важное для повышения уровня защищенности ваших данных – обеспечить недоступность для взломщиков PIN-кодов, используемых для доступа к Guardant ID. Мы предлагаем следующие рекомендации:

1. Вашим клиентам, скорее всего, нет нужды стирать всю память Guardant ID, включая PIN-код. Поэтому рекомендуем не передавать ПО, которое выполняет команду Initialize, кому бы то ни было.
2. Если ваше приложение не производит инициализацию памяти Guardant ID командой Initialize, *не храните* Master PIN-код в теле приложения.
3. Инициализируйте случайными числами 32-байтовую строку, в которую будете заносить PIN-код перед вызовом команды Guardant ID, – например, при помощи функции **rand()** (Borland C++). Это затруднит взломщику анализ передаваемого вами PIN-кода.

4. Старайтесь не использовать в качестве PIN-кода слова, фразы, наборы цифр, которые могут быть легко идентифицированы взломщиком в составе 32-байтовой строки. PIN-код состоит из набора байт, которые могут иметь любые значения. Демонстрационный PIN-код состоит из набора цифр только для наглядности!
5. Скрывайте PIN-код в теле приложения. Например, можно формировать его непосредственно перед вызовом команды. Пусть вы используете PIN-код “12345678”, тогда для его формирования выполните такую последовательность команд:

   szPIN[0] = ‘1’;
   szPIN[1] = ‘2’;
   szPIN[2] = ‘3’;
   szPIN[3] = ‘4’;
   szPIN[4] = ‘5’;
   szPIN[5] = ‘6’;
   szPIN[6] = ‘7’;
   szPIN[7] = ‘8’;

   Таким образом, нигде в теле приложения не будет содержаться строка с готовым PIN-кодом, и потенциальному взломщику будет сложнее его узнать. После обращения к Guardant ID рекомендуется очистить сформированную строку.

3.6. Работа с Guardant ID

Для работы с Guardant ID в комплект поставки входят USB-драйверы GRDID.SYS, а также INF-файл GRDID.INF.

3.6.1. Установка USB-драйвера Guardant ID

Для установки драйверов нужно использовать USB-драйвер. Файл GRDID.INF нужно записать в папку %WINDIR%\INF\. Файл GRDID.SYS записывается в папку %WINDIR%\SYSTEM32\DRIVERS\.

Пользователь может установить драйвер Guardant ID и самостоятельно – при помощи стандартного мастера Windows. Мастер будет вызван автоматически при первом подсоединении Guardant ID к компьютеру, на который еще не установлены драйверы.

Для завершения установки драйверов может потребоваться перезагрузка.

После установки драйверов можно подключать Guardant ID к любому USB-разъему компьютера – система автоматически распознает и инициализирует его. Признаком того, что Guardant ID готов к работе, является светящийся светодиод внутри его корпуса.

3.6.2. Порядок работы

Работа с Guardant ID возможна из 32-битных Windows-приложений.

Поиск и открытие устройства

Для того чтобы получить доступ к устройству Guardant ID в Windows 98/Me/2000/XP, нужно вызвать функцию **OpenGuardantID(BYTE byIndex)**, экспортируемую GRDID.DLL. Входной параметр byIndex задает порядковый номер Guardant ID среди нескольких одновременно подключенных к компьютеру. Диапазон значений byIndex – от 0 до 126. Таким образом, byIndex – это аналог суффикса имени устройства, используемого в первом методе поиска Guardant ID. Функция OpenGuardantID() возвращает handle открытого устройства – так же, как и CreateFile(), рассмотренная выше.

Для того чтобы среди ключей, одновременно подключенных к компьютеру, найти нужный Guardant ID, требуется организовать поиск, как описано в разделе “Важные замечания”.

Обмен данными с Guardant ID

Для обмена данными с выбранным Guardant ID используется стандартная функция Windows **ReadFile()**. Среди входных параметров ей передается handle устройства, открытого функцией CreateFile() или OpenGuardantID(), и адрес структуры, в поля которой нужно предварительно занести номер команды и соответствующие ей входные параметры. Список параметров для каждой команды приведен выше. Описание структуры содержится в файле RQST_GID.H.

При выполнении функции ReadFile() данная структура будет передана в драйвер Guardant ID – таким образом он получит входные параметры. По окончании обмена с ключом драйвер сформирует блок выходных данных – код возврата, блок считанных из ключа слов и т.п. – и поместит их в эту же структуру. Таким образом, по завершении функции ReadFile() та же структура будет содержать все необходимые выходные данные.

Закрытие устройства

По окончании обмена с Guardant ID необходимо закрыть устройство при помощи стандартной функции Windows **CloseHandle()**. Т.к. ключ Guardant ID, как и любое другое USB-устройство, может быть отсоединен пользователем в любой момент времени, рекомендуется закрывать устройство после выполнения каждой команды (т.е. после каждого вызова функции ReadFile()). Поэтому оптимальный алгоритм работы с ключом Guardant ID содержится в функции CallDriver() (см. файл TEST32.C).

3.7. Важные замечания

При работе с Guardant ID необходимо учитывать:

1. Электронный ключ Guardant ID является динамически подключаемым USB-устройством, которое может быть отключено пользователем в любой момент

времени. Поэтому, найдя подходящий Guardant ID, вы не можете быть уверены в том, что номер устройства (и даже handle открытого устройства) будет оставаться верным критерием для обмена с Guardant ID в течение всего сеанса работы вашего ПО.

Поэтому мы рекомендуем придерживаться следующих правил:

- Если вы хотите работать с любым Guardant ID, вам достаточно найти первый попавшийся из подсоединенных ключей. Но если вы хотите работать с каким-то определенным ключом, вам следует организовать его поиск, выполняя команду GetInfo в цикле и сравнивая возвращенный ею ID с ID нужного вам ключа.
- Найдя подходящий Guardant ID, надо запомнить его ID, который вернет команда GetInfo. Все остальные команды требуют предъявления этого ID.
- Рекомендуется также запомнить и имя (порядковый номер) найденного устройства. Вы ускорите работу с Guardant ID, если все последующие вызовы функции CreateFile() или OpenGuardantID() будете производить с указанием этого имени (номера).
- Если устройство перестало открываться, это может означать, что Guardant ID отсоединен пользователем. В этом случае надо повторить поиск Guardant ID по уже известному вам ID ключа – возможно, ключ был переставлен в другой USB-порт. Те же действия рекомендуется произвести и в случае, если перестал существовать handle уже открытого устройства.
- Для удобства вы можете создать функцию **FindGuardantID(BYTE szID, char *szDevNameOut, BYTE *lpbyIndexOut)**, в которой можно организовать выполнение команды GetInfo в цикле с проверкой ID на каждой итерации. В случае, если ключ с таким ID найден, его порядковый номер возвращается в *lpbyIndexOut, а имя устройства – в szDevNameOut. Этой функцией будет удобно пользоваться для поиска “внезапно пропавшего” Guardant ID.

2. Несмотря на наличие 2 методов открытия устройства, нет необходимости определять тип ОС, в которой работает ваше ПО. Вместо этого вы можете всякий раз пытаться сначала открыть устройство при помощи CreateFile() и, если функция вернет ошибку, – вызывать функцию OpenGuardantID(). Именно такой алгоритм приведен в обновленной функции CallDriver() (см. TEST32.C). Среди параметров функции имеется и строка с именем устройства, и байт с индексом. Единственное условие – нужно обеспечить синхронизацию этих 2-х параметров, т.е. вызывать функцию, например, с параметрами “GRDID000” и 0, “GRDID012” и 12 и т.д. При выполнении этого условия вы можете не заботиться о том, в какой версии Windows работает ваше ПО.