GrdHashEx

Функция(метод) **GrdHashEx** вычисляет хеш-сумму блока данных при помощи аппаратно- или программно-реализованного алгоритма (SHA256, HASH64 или CRC32).

C

```
int GRD_API GrdHashEx(
  HANDLE hGrd,
  DWORD dwHash,
  DWORD dwDataLng,
  void *pData,
  DWORD dwMethod,
  DWORD dwDigestLng,
  void *pDigest,
  DWORD dwKeyBufLng,
  void *pKeyBuf,
  DWORD dwContextLng,
  void *pContext,
  void *pReserved);
);
```

hGrd [in]	хэндл, через который будет выполнена данная операция
dwHash [in]	номер дескриптора аппаратного алгоритма (SHA256, HASH64) или номер программно-реализованного алгоритма (SHA256 или CRC32), с помощью которого будет выполняться хеширование. Программно-реализованным алгоритмам SHA256 и CRC32 присвоены номера **GrdSH_SHA256** и **GrdSH_CRC32** соответственно
dwDataLng [in]	длина буфера данных в байтах
pData [in]	указатель на буфер данных для хеширования
dwMethod [in]	параметр задаёт порядок следования блока данных. В таблице перечислены допустимые для данного параметра флаги: **Биты 8-9: номер блока данных** **GrdSC_First** — Первый блок данных **GrdSC_Next** — Следующий блок данных **GrdSC_Last** — Возврат результата **GrdSC_All** — Один блок данных Если хеш-сумму требуется вычислить от непрерывного участка данных, то следует использовать флаг **GrdSC_All**
dwDigestLng [in]	длина выходного буфера для значения хеш-суммы (в байтах). Длина зависит от используемого алгоритма и должна быть не менее следующего значения: • для алгоритма HASH64 - **GrdHASH64_DIGEST_SIZE** • для алгоритма SHA256 - **GrdSHA256_DIGEST_SIZE** • для алгоритма CRC32 - **GrdCRC32_DIGEST_SIZE**
pDigest [in, out]	указатель на выходной буфер, в который будет сохранена хеш-сумма
dwKeyBufLng	параметр не используется, должен быть равен 0

pKeyBuf	параметр не используется, должен быть равен **NULL**
dwContextLng [in]	длина контекста хеш-функции (в байтах). Если предполагается вычисление хеш-суммы от нескольких блоков данных (установлен один из флагов **GrdSC_First** / **GrdSC_Next / GrdSC_Last**), то для сохранения промежуточного состояния , функции необходимо передать указатель на контекст, память для которого должна быть зарезервирована заранее и быть не менее следующего значения: • для алгоритма HASH64 - **GrdHASH_CONTEXT_SIZE** • для алгоритма SHA256 - **GrdSHA256_CONTEXT_SIZE** • для алгоритма CRC32 - **GrdCRC32_CONTEXT_SIZE** Если функция вызывается с флагом **GrdSC_All**, то контекст не используется и параметр должен быть равен 0
pContext [in, out]	указатель на буфер контекста хеш-функции (установлен один из флагов **GrdSC_First** / **GrdSC_Next / GrdSC_Last**). Если функция вызывается с флагом **GrdSC_All**, то контекст не используется и параметр должен быть равен **NULL**
pReserved	параметр не используется, должен быть равен **NULL**

Набор ошибок Guardant API

Функция **GrdHashEx** выполняет преобразование входного буфера данных произвольной длины в выходную битовую строку фиксированной длины. Такое преобразование называется хешированием или хеш-функцией. Вычислить хеш-сумму можно как при помощи программно-реализованного алгоритма (SHA256,CRC32), так и используя аппаратную реализацию внутри электронного ключа (SHA256, HASH64). Если параметр *dwHash* соответствует аппаратному алгоритму, вызов переадресуется в электронный ключ. Иначе, если алгоритм реализован программно (установлен старший бит в параметре *dwHash*), вызывается соответствующая функция программного хеширования.

Данные для хеширования должны быть расположены по адресу, указанному в параметре *pData*. Длина входных данных задаётся параметром *dwDataLng*. Функция считывает входные данные если указан один из флагов **GrdSC_First** или **GrdSC_Next** . Результат операции хранится внутри контекста, обеспечивая возможность вычисления одной хеш-суммы от нескольких блоков данных. Для получения результата следует использовать флаг **GrdSC_Last**. Если вызов функции осуществляется с флагом **GrdSC_All**, то считывание данных и возврат результата выполняются одновременно.

Для аппаратных алгоритмов:

Шифрование выполняется алгоритмом с номером, заданным в параметре *dwAlgoNum*. Этот алгоритм предварительно должен быть создан, в противном случае возвращается ошибка GrdE_AlgoNotFound. Если в дескрипторе алгоритма установлен флаг "уменьшение счетчика", то вычитание счетчика алгоритма происходит при каждом вызове **GrdCryptEx**. При достижении счетчиком нулевого значения, возвращается ошибка GrdE_GPis0. Если при создании алгоритма или в процессе работы он был переведен в неактивное состояние, возвращается ошибка GrdE_InactiveItem.

При работе с аппаратным алгоритмом HASH64 рекомендуется использовать блок данных *pData*, длина которого *dwDataLng* кратна 16 байтам (**GrdARS_HASH64**). Если длина блока данных не будет кратна этой величине, то буфер будет дополнен нулями до ближайшего значения, кратного 16. Для алгоритмов SHA256 и CRC32 таких ограничений нет.

Вызов данной функции с флагом **GrdSC_First** приводит к неявной блокировке электронного ключа операцией GrdLock. Блокируется выполнение всех операций на данном ключе. Вызов GrdUnlock выполняется во время считывания результата, т.е. вызова функции с флагом **GrdSC_Last**.

C#

```
public static GrdE GrdHashEx(Handle grdHandle,GrdAlgNum hashNum, byte[] data, GrdSC method, out byte[] digest)
public static GrdE GrdHashEx(Handle grdHandle,GrdAlgNum hashNum, byte[] data, GrdSC method, out byte[] digest,
byte[]context)
```

grdHandle [in]

Тип: Handle

хэндл, через который будет выполнена данная операция.

hashNum [in]

Тип: GrdAlgNum

Номер аппаратно- или программно-реализованного алгоритма, с помощью которого будет вычисляться хэш-функция.

data [in]

Тип: byte [ ]

Указатель на буфер данных, хэш которых будет вычисляться.

method [in]

Тип: GrdSC

Метод преобразования, который задается комбинацией флагов GrdSC.

digest [out]

Тип: byte [ ]

Указатель на буфер, куда будет помещен результат вычислений.

context [in, out]

Тип: byte [ ]

Буфер для хранения контекста (состояния алгоритма) при последовательном вычислении хэш-функции SHA256 от нескольких блоков данных.
Набор ошибок Guardant API
Метод GrdHashEx вычисляет хэш-функцию блока данных *data.*

В Guardant API реализовано вычисление программно-реализованных функций **CRC32** и **SHA256**, а также аппаратных алгоритмов **HASH64**. Выбор алгоритма осуществляется параметром *hashNum*, который может принимать значения констант GrdSA.CRC32 и GrdSA.SHA256 для программно-реализованных алгоритмов, либо номера используемого аппаратного алгоритма **HASH64**.

Все хэш-функции могут вычисляться от больших блоков данных, поэтому предусмотрена возможность разбиения данных на меньшие блоки и вычисления значения хэша последовательно для нескольких буферов. Для этого параметром *method* задается порядок блока GrdSC.XXX (первый, следующий, последний).

Для передачи состояния алгоритма SHA256 используется специальный контекст *context*, память для которого должна быть зарезервирована и проинициализирована заранее. Для алгоритмов **HASH64** и **CRC32** контекст не используется. Метод **GrdHashEx** самостоятельно разбивает буфер на блоки необходимой длины и выполняет все операции по согласованию.

Если операция выполняется за один прием, то параметром *method* должен быть задан метод GrdSC.All.

При работе с аппаратным алгоритмом **HASH64** рекомендуется использовать блок данных *data*, длина которого кратна 16 байтам (GrdARS.HASH64). Если длина блока данных не будет кратной 16, то буфер дополнится нулями до ближайшего значения, кратного 16. Поэтому для воспроизведения итогового значения хеша при "некратных" значениях размера буфера, важны не только последовательности их обработки, но и их размеры.

На время вычисления **HASH64** блокируется выполнение любых алгоритмов на данном ключе. Поэтому передача буфера большого размера может надолго заблокировать электронный ключ.

Результат вычислений помещается в буфер *digest*, память для которого размером, соответствующим конкретному алгоритму, должна быть зарезервирована заранее.

Примечание

> Результат вычислений возвращается в память по адресу *digest* только в момент вызова с флагом GrdSC.Last (или *GrdSC.All)*. До этого промежуточный результат хранится во внутреннем буфере ключа, вследствие чего (с целью его сохранности) при вычислении hash от длинных блоков данных в несколько подходов рекомендуется блокировать обращение к этому методу путем вызова GrdLock с соответствующими параметрами.

Java

```
public static GrdE GrdHashEx(Handle grdHandle, int hash, byte[] data, GrdSC method, byte[] digest)
public static GrdE GrdHashEx(Handle grdHandle, int hash, byte[] data, GrdSC method, byte[] digest, byte[]
context)
```

grdHandle [in]

Тип: Handle

хэндл, через который будет выполнена данная операция.

hash [in]

Тип: int

Номер аппаратно- или программно-реализованного алгоритма, с помощью которого будет вычисляться хэш-функция.

data [in]

Тип: byte [ ]

Указатель на буфер данных, хэш которых будет вычисляться.

method [in]

Тип: GrdSC

Метод преобразования, который задается комбинацией флагов GrdSC.

digest [out]

Тип: byte [ ]

Указатель на буфер, куда будет помещен результат вычислений.

context [in, out]

Тип: byte [ ]

Буфер для хранения контекста (состояния алгоритма) при последовательном вычислении хэш-функции SHA256 от нескольких блоков данных.
Набор ошибок Guardant API
Метод **GrdHashEx** вычисляет хэш-функцию блока данных *data*.

В Guardant API реализовано вычисление программно-реализованных функций CRC32 и SHA256, а также аппаратных алгоритмов HASH64. Выбор алгоритма осуществляется параметром *hashNum*, который может принимать значения констант GrdSA.CRC32 и GrdSA.SHA256 для программно-реализованных алгоритмов, либо номера используемого аппаратного алгоритма **HASH64**.

Все хэш-функции могут вычисляться от больших блоков данных, поэтому предусмотрена возможность разбиения данных на меньшие блоки и вычисления значения хэша последовательно для нескольких буферов. Для этого параметром *method* задается порядок блока GrdSC.XXX (первый, следующий, последний).

Для передачи состояния алгоритма **SHA256** используется специальный контекст *context*, память для которого должна быть зарезервирована и проинициализирована заранее. Для алгоритмов **HASH64** и **CRC32** контекст не используется. Метод **GrdHashEx** самостоятельно разбивает буфер на блоки необходимой длины и выполняет все операции по согласованию.

Если операция выполняется за один прием, то параметром *method* должен быть задан метод GrdSC.All.

При работе с аппаратным алгоритмом **HASH64** рекомендуется использовать блок данных *data*, длина которого кратна 16 байтам (GrdARS.HASH64). Если длина блока данных не будет кратной 16, то буфер дополнится нулями до ближайшего значения, кратного 16. Поэтому для воспроизведения итогового значения хеша при "некратных" значениях размера буфера, важны не только последовательности их обработки, но и их размеры.

На время вычисления **HASH64** блокируется выполнение любых алгоритмов на данном ключе. Поэтому передача буфера большого размера может надолго заблокировать электронный ключ.

Результат вычислений помещается в буфер *digest*, память для которого размером, соответствующим конкретному алгоритму, должна быть зарезервирована заранее.

Примечание

> *Результат вычислений возвращается в память по адресу digest* только в момент вызова с флагом GrdSC.Last (или GrdSC.All*).* До этого промежуточный результат хранится во внутреннем буфере ключа, вследствие чего (с целью его сохранности) при вычислении hash от длинных блоков данных в несколько подходов рекомендуется блокировать обращение к этому методу путем вызова GrdLock с соответствующими параметрами.