Протокол связи по RS-232 (USART) используемый в загрузчике STM32™

Введение

Данные указания по применению описывают протокол связи по RS-232 (USART), используемый в загрузчике микроконтроллера STM32. Здесь подробно описывается каждая поддерживаемая команда. Более подробная информация по аппаратным ресурсам USART и требованиям загрузчика вашего устройства находится в указаниях по применению «STM32 system memory boot mode» («Режим загрузки из системной памяти STM32») № AN2606.

Документы, связанные с данным

Доступны на www.st.com:

AN2606 «STM32 system memory boot mode» («Режим загрузки из системной памяти STM32»)

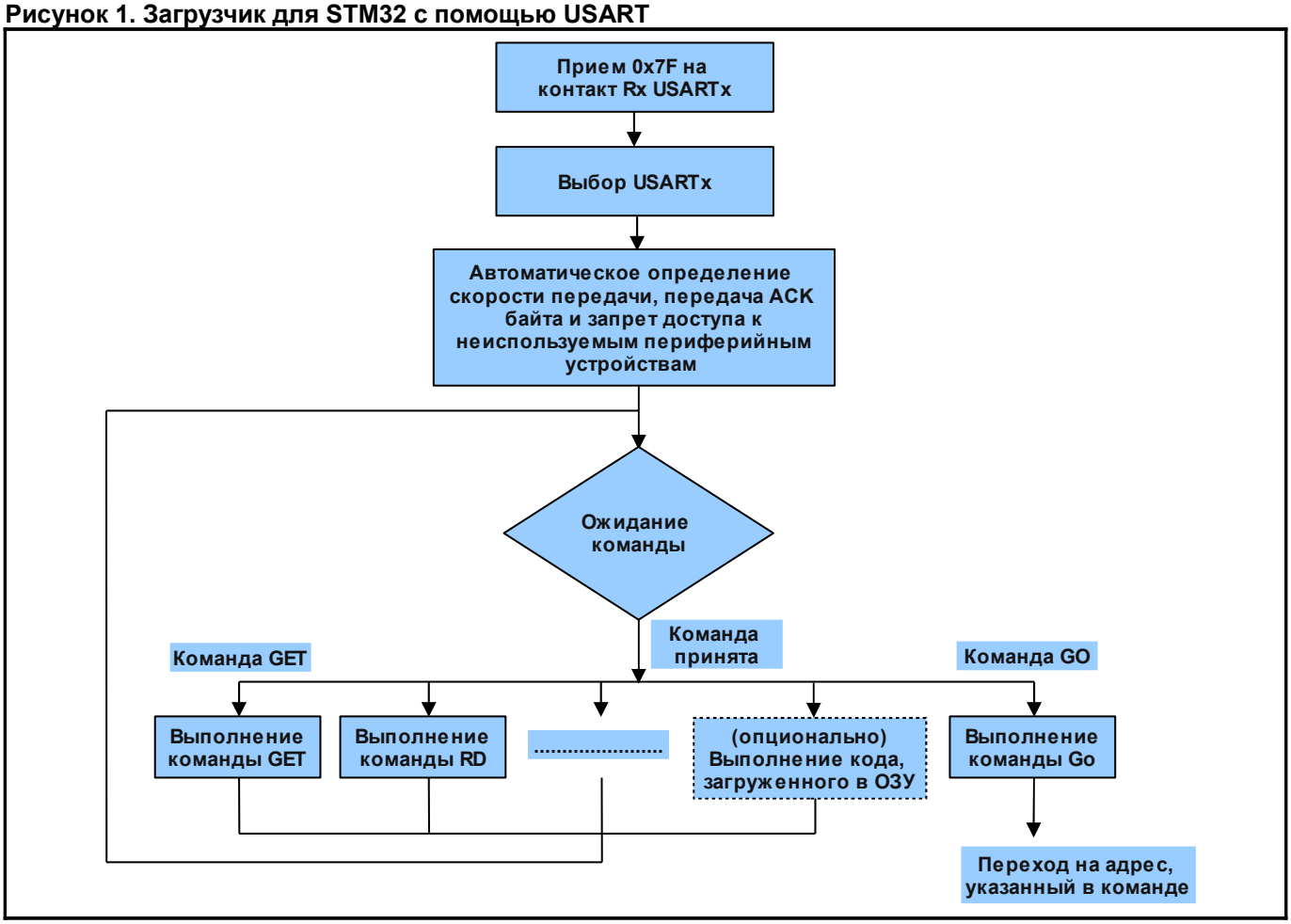
Содержание

Список таблиц

Список рисунков

1 Алгоритм работы загрузчика USART

Рисунок 1. Загрузчик для STM32 с помощью USART



Когда установлен режим загрузки из системной памяти и микроконтроллер STM32 скомпонован (более детально смотри указания по применению № AN2606 «STM32 system memory boot mode» («Режим загрузки из системной памяти STM32»)), код загрузчика начинает опрашивать контакт USARTx_RX ожидая приема пакета данных «0x7F»: один старт-бит, 0x7F бит данных, бит проверки на четность (even) и один стоп-бит.

Длительность этого пакета данных определяется с помощью системного таймера (Systick timer). Значение, полученное от таймера, используется для вычисления соответствующего множителя скорости передачи относительно текущей частоты системы.

Затем код загрузчика соответственно настраивает последовательный интерфейс. Используя эту вычисленную скорость передачи на хост возвращается байт подтверждения или ACK-байт, равный 0x79. Он является сигналом того, что STM32 готов к приему команд.

2 Выбор скорости передачи данных

Значение скорости передачи данных USARTx, вычисленное по времени приема первого байта, используется для работы загрузчика в широком диапазоне скоростей передачи. Однако верхний и нижний пределы должны быть постоянны для полной гарантии правильной передачи данных.

Для правильной передачи данных от хоста на микроконтроллер максимальное отклонение между внутренней установленной скоростью передачи USARTx и реальной скоростью передачи хоста должно быть меньше 2.5 %. Отклонение (f_B, в процентах) между скоростью передачи хоста и скоростью передачи микроконтроллера вычисляется по формуле, представленной ниже:

$$f_B = \left| \frac{скорость\ передачи\ STM32 - скорость\ передачи\ хоста}{скорость\ передачи\ STM32} \right| \times 100$$, где $f_B \leq 2.5$ %

Данное отклонение скорости передачи является нелинейной функцией, зависящей от частоты процессора и скорости передачи хоста. Максимальное значение этой функции (f_B) возрастает с увеличением скорости передачи данных хоста. Это является следствием использования масштабируемых множителей наименьшей скорости передачи и приводит к высокой ошибке квантования.

2.1 Минимальная скорость передачи данных

Наименьшая возможная скорость передачи (B_{Low}) равна 1200. Скорости передачи менее B_{Low} могут привести к переполнению системного таймера. В этом случае USARTx не может быть корректно настроен.

2.2 Максимальная скорость передачи данных

B_{High} - это наибольшая скорость передачи, на которой отклонение не будет превышать предельное значение. Все скорости передачи между B_{Low} и B_{High} имеют отклонения ниже предельного значения. Наибольшая возможная скорость передачи (B_{High}) равна 115200.

3 Набор команд загрузчика

Поддерживаемые команды представлены в Таблице 1. Кроме того, в ней приводится описание каждой команды.

Таблица 1. Команды загрузчика USART

Команда[1]	Код команды	Описание команды
Get[2]	0x00	Запрашивает версию и список команд, поддерживаемых текущей версией загрузчика
Get Version & Read Protection Status[2]	0x01	Запрашивает версию загрузчика и уровень защиты от чтения (Read Protection status) flash-памяти
Get ID[2]	0x02	Запрашивает ID кристалла микроконтроллера
Read Memory	0x11	Считывает до 256 байтов памяти начиная с адреса, определяемого программно
Go	0x21	Переходит на выполнение программного кода пользователя, расположенного во внутренней flash-памяти или в ОЗУ
Write Memory	0x31	Записывает до 256 байтов в ОЗУ или flash-память начиная с адреса, определяемого программно
Erase[3]	0x43	Очищает от одной до всех страниц flash-памяти
Extended Erase[3]	0x44	Очищает от одной до всех страниц flash-памяти с использованием двухбайтного режима адресации (доступна только в версии 3.0 и более поздних версиях USART загрузчика)
Write Protect[4]	0x63	Разрешает защиту от записи для выбранных секторов памяти
Write Unprotect[4]	0x73	Запрещает защиту от записи для всех секторов flash-памяти
Readout Protect	0x82	Разрешает защиту от чтения
Readout Unprotect[2]	0x92	Запрещает защиту от чтения

1. Если принимается неизвестная команда или происходит ошибка при выполнении команды, загрузчик передает NACK-байт и переходит назад на определение команды.
2. Защита от чтения - Когда опция RDP (read protection) установлена, доступно только это ограниченное подмножество команд. Все остальные команды заблокированы и не оказывают никакого эффекта на устройство. В случае отключения опции RDP другие команды становятся активными.
3. Команды Erase (0x43) и Extended Erase (0x44) являются взаимоисключающими. Устройство может поддерживать или команду Erase, или команду Extended Erase, но не обе сразу.
4. Смотри ***Секцию 3.1: Параметры загрузчика, зависящие от устройства***.

Безопасность связи

Все пакеты данных между программой на компьютере и устройством на микроконтроллере должны удовлетворять следующим требованиям:

1. контрольная сумма: принимаемые блоки байтов данных должны подчиняться операции XOR (ИСКЛЮЧАЮЩЕЕ ИЛИ). Байт, содержащий результат операции XOR со всеми предыдущими байтами добавляется в конец каждого пакета данных (байт контрольной суммы). В результате выполнения операции XOR над всеми принятыми байтами, данными + байт контрольной суммы, должно получиться значение 0x00
2. для каждой команды хост передает байт и его контрольную сумму (XOR = 0x00)
3. UART: включена проверка на четность (even)

4. Каждый пакет может быть принят (в ответ посылается ACK-байт) или отвергнут (в ответ посылается NACK-байт):
   - ACK = 0x79
   - NACK = 0x1F

3.1 Параметры загрузчика, зависящие от устройства

Хотя список команд протокола загрузчика USART одинаков для всех устройств на основе STM32, некоторые параметры загрузчика зависят от конкретного устройства. Для небольшого количества команд значения некоторых параметров могут зависеть от того, какое устройство используется. Данные параметры представлены ниже:

- PID (product ID), который изменяется в зависимости от устройства
- Действительные адреса памяти (ОЗУ, flash-памяти, системной памяти, области байт настроек), определяемые загрузчиком при выполнении команд Read Memory, Go и Write Memory.
- Размер сектора flash-памяти, используемого при выполнении команды Write Protect.

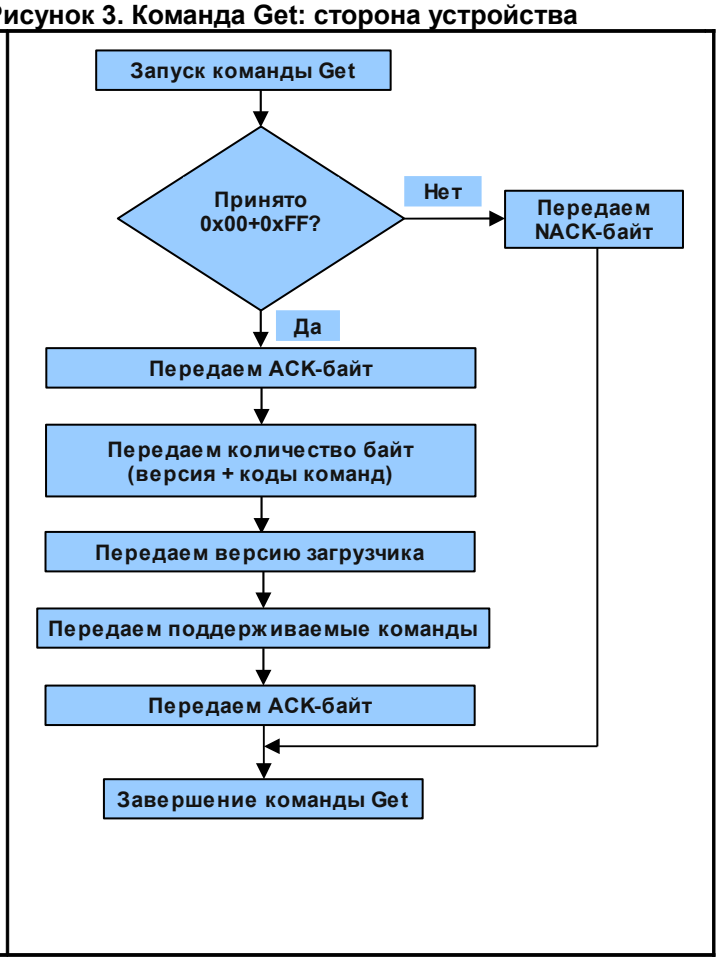
Более подробно прочитать о значениях этих параметров вы можете в секции «Device-dependent bootloader parameters» («Параметры загрузчика, зависящие от устройства») в указаниях по применению «STM32 system memory boot mode» («Режим загрузки из системной памяти STM32») № AN2606.

3.2 Команда «Get» (Определить)

Команда Get позволяет вам определить версию загрузчика и список поддерживаемых команд. Когда загрузчик принимает команду Get, он передает версию загрузчика и коды поддерживаемых команд, как показано на *Рисунке 2*.

Рисунок 2. Команда Get: сторона хоста

Рисунок 3. Команда Get: сторона устройства



Микроконтроллер STM32 передает следующую последовательность байтов:

Байт 1: ACK

Байт 2: N = 11 = количество передаваемых байт, следующих за этим - 1 исключая текущий и ACK

Байт 3: Версия загрузчика (0 < Версия < 255), например: 0x10 = Версия 1.0

Байт 4: 0x00 - команда «Get»

Байт 5: 0x01 - команда «Get Version and Read Protection Status»

Байт 6: 0x02 - команда «Get ID»

Байт 7: 0x11 - команда «Read Memory»

Байт 8: 0x21 - команда «Go»

Байт 9: 0x31 - команда «Write Memory»

Байт 10: 0x43 или 0x44 - команда «Erase» или «Extended Erase» (эти команды взаимоисключающие)

Байт 11: 0x63 - команда «Write Protect»

Байт 12: 0x73 - команда «Write Unprotect»

Байт 13: 0x82 - команда «Readout Protect»

Байт 14: 0x92 - команда «Readout Unprotect»

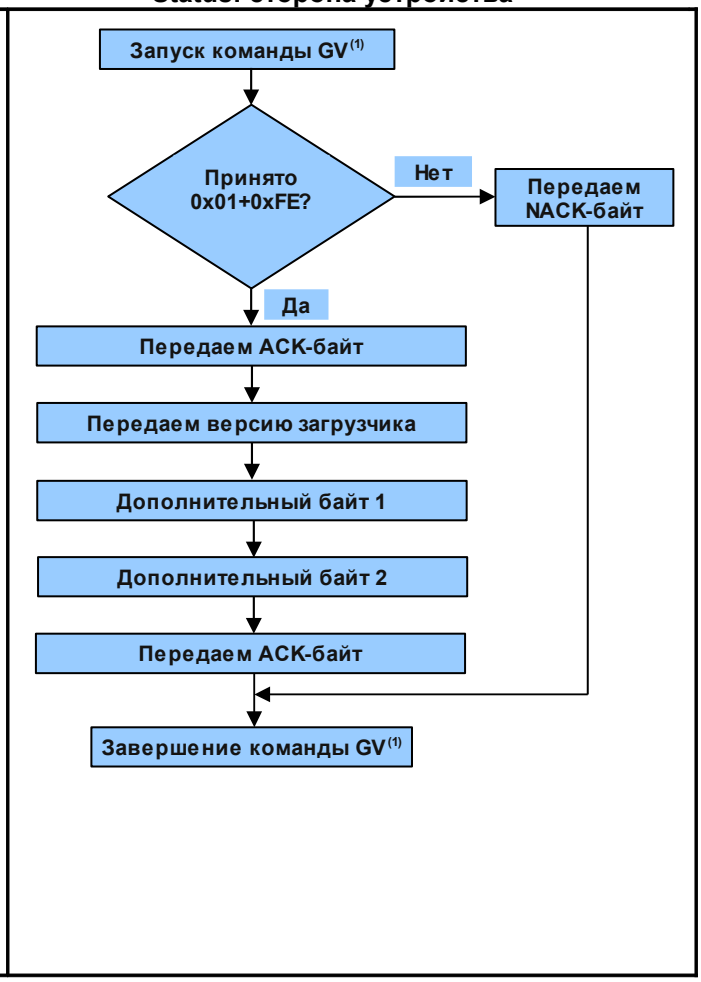
Последний байт (15): ACK

3.3 Команда «Get Version & Read Protection Status» (Определить версию и уровень защиты от чтения)

Команда «Get Version & Read Protection Status» используется для определения версии загрузчика и уровня защиты от чтения. Когда загрузчик принимает эту команду, он передает информацию, описанную ниже (версию, защиту от чтения: время доступа и время запрета) на хост.

Рисунок 4. Команда Get Version & Read Protection Status: сторона хоста

Рисунок 5. Команда Get Version & Read Protection Status: сторона устройства



1. GV = Get Version & Read Protection Status.

Микроконтроллер STM32 передает следующую последовательность байтов:

Байт 1: ACK

Байт 2: Версия загрузчика (0 < Версия < 255), например: 0x10 = Версия 1.0

Байт 3: Дополнительный байт 1: 0x00 для сохранения совместимости с общим протоколом загрузчика

Байт 4: Дополнительный байт 2: 0x00 для сохранения совместимости с общим протоколом загрузчика

Байт 5: ACK

3.4 Команда «Get ID» (Определить ID)

Команда Get ID используется для определения версии кристалла микроконтроллера (chip identification). Когда загрузчик принимает эту команду, он передает ID микроконтроллера на хост.

Рисунок 6. Команда Get ID: сторона хоста

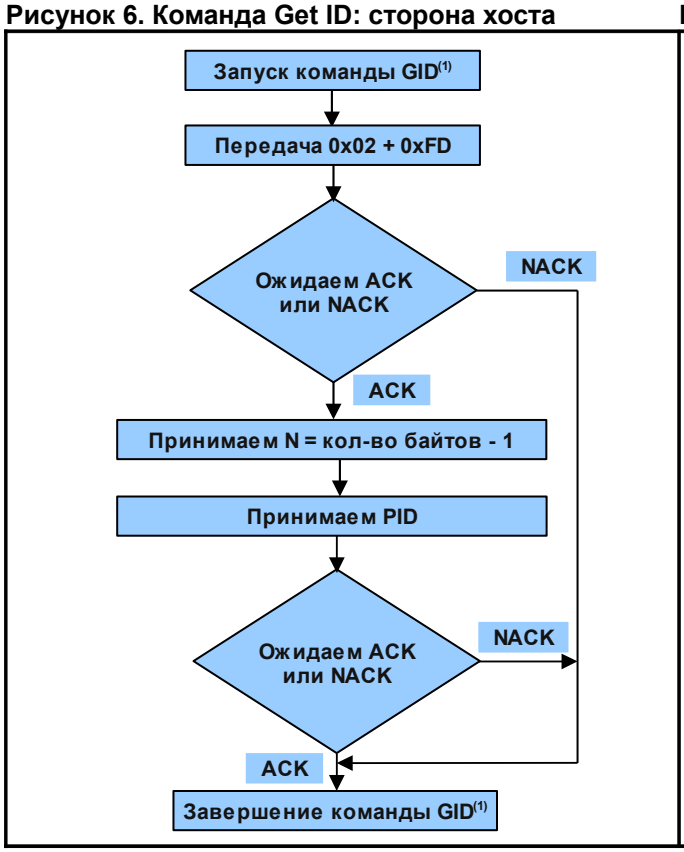
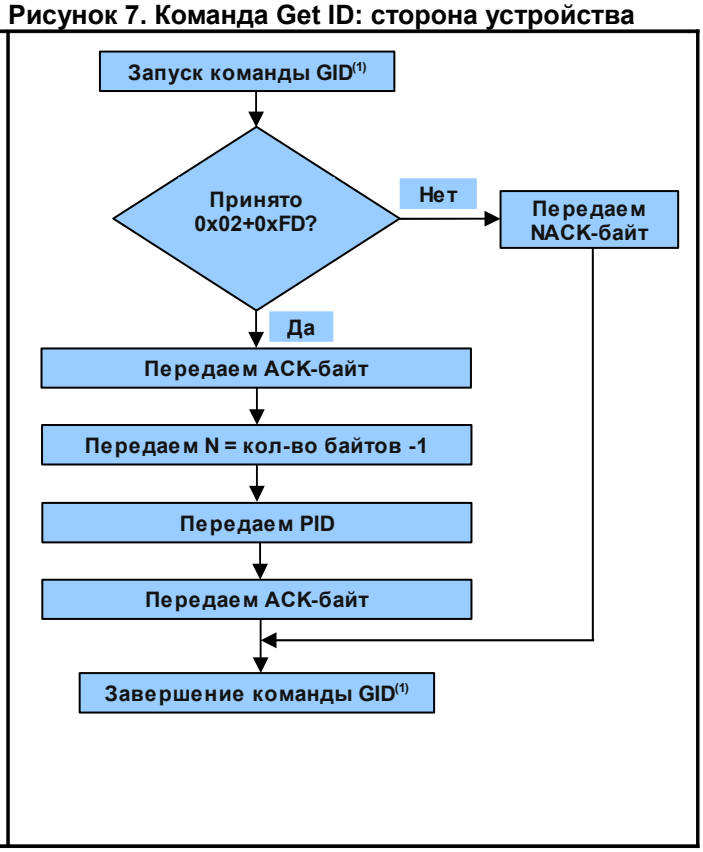


Рисунок 7. Команда Get ID: сторона устройства



1. GID = Get ID.

Микроконтроллер STM32 передает следующую последовательность байтов:

Байт 1: ACK

Байт 2: N = количество байтов - 1 (N = 1 для STM32), не считается этот байт и байты ACK.

Байты 3-4: PID(1) байт 3 = 0x04, байт 4 = 0x1X

Байт 5: ACK

1. PID расшифровывается как product ID. Байт 1 - старший, а байт 2 - младший байт адреса. Смотри *Секцию 3.1: Параметры загрузчика, независимые от устройства* про PID используемого устройства.

3.5 Команда «Read Memory» (Чтение памяти)

Команда Read Memory используется для чтения данных с любого реально существующего адреса ОЗУ, flash-памяти и информационного блока (системная память или области байт настроек).

Внимание: *В Секции 3.1: Параметры загрузчика, независимые от устройства более подробно рассказано о реальных адресах памяти используемого вами устройства.*

Когда загрузчик принимает команду «Read Memory», он передает ACK-байт вашей программе. После передачи ACK-байта загрузчик ожидает приема адреса (4 байта, первый байт старший, четвертый младший) и байт контрольной суммы, необходимый для проверки принятого адреса. Если адрес реальный и контрольная сумма правильная, загрузчик передает ACK-байт, иначе он передает NACK-байт и завершает работу команды.

Когда адрес реальный и контрольная сумма правильная загрузчик ожидает приема количества байт для передачи -1 (N байт) и один дополнительный байт (контрольная сумма). Если контрольная сумма правильная, он передает необходимые данные (N+1 байт) вашей программе, начиная с принятого адреса. Если контрольная сумма неправильная, он передает NACK-байт и завершает работу команды.

Хост передает на STM32 следующую последовательность байтов:

Байты 1-2: 0x11 + 0xEE

Ожидает приема ACK-байта

Байты 3-6: начальный адрес

- байт 3: старший байт адреса
- байт 6: младший байт адреса

Байт 7: контрольная сумма: операция XOR между байтами 3, 4, 5, 6

Ожидает приема ACK-байта

Байт 8: количество байтов, которое необходимо прочитать - 1 ($0 < N \leq 255$);

Байт 9: контрольная сумма: XOR с байтом 8 (дополнение к байту 8)

Рисунок 8. Команда Read Memory: сторона хоста

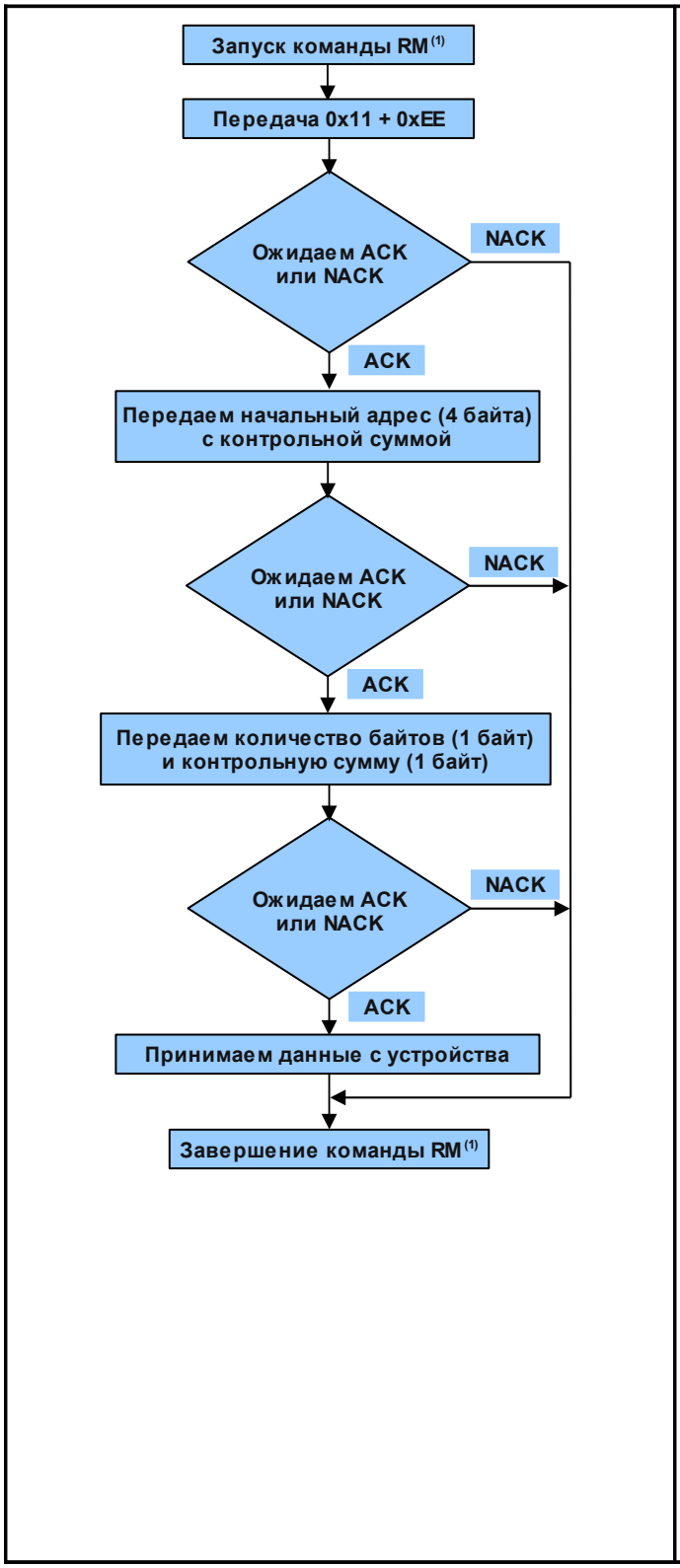
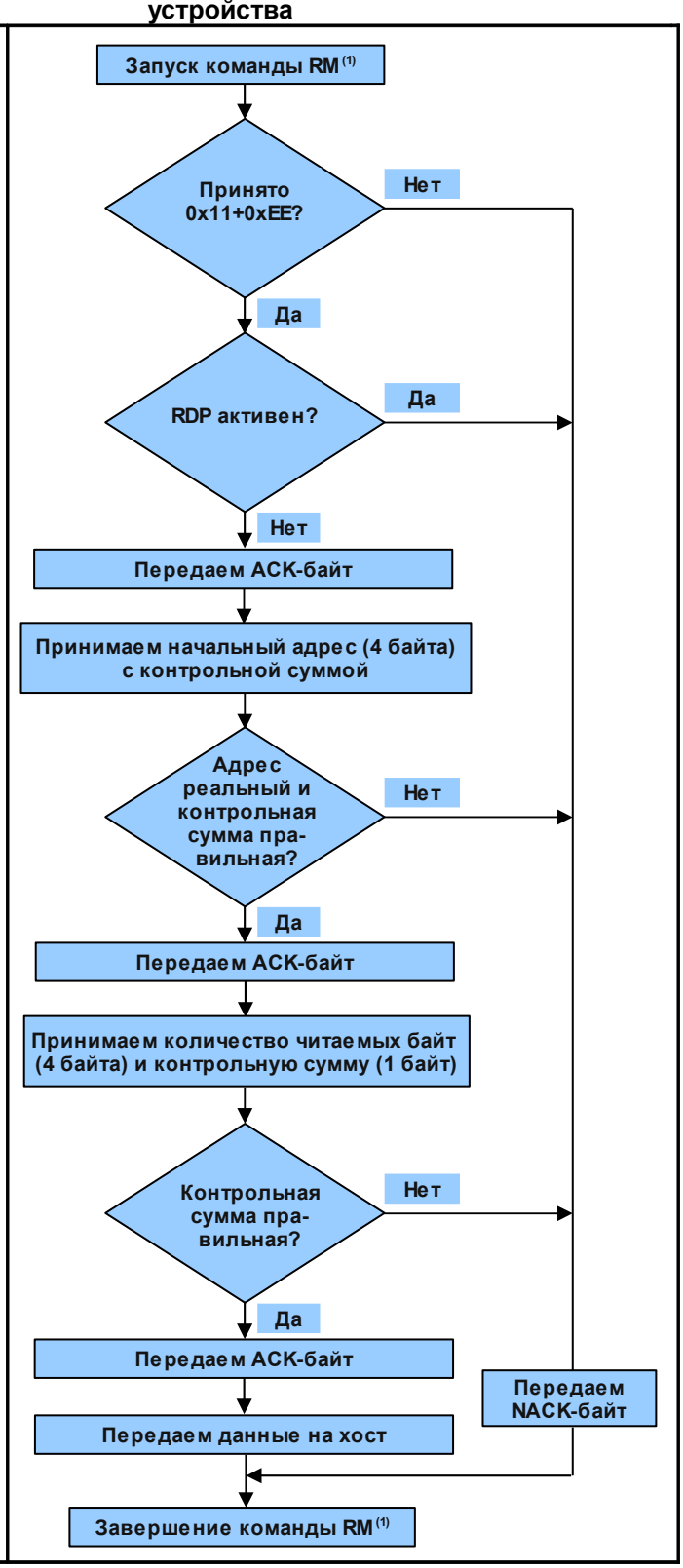


Рисунок 9. Команда Read Memory: сторона устройства



1. RM = Read Memory.

3.6 Команда «Go» (Пуск)

Команда Go используется для запуска на выполнение загруженного кода с помощью перехода на адрес, с которого начинается выполнение данного кода. Когда загрузчик принимает команду Go, он передает ACK-байт программе на компьютере. После передачи ACK-байта, загрузчик ожидает адрес перехода (4 байта, 1 байт старший, 4 байт младший) и байт контрольной суммы для проверки правильности принятого адреса. Если адрес реальный и контрольная сумма правильная загрузчик передает ACK-байт, в противном случае он передает NACK-байт и завершает работу команды.

Когда адрес реальный и контрольная сумма правильная, программа загрузчика выполняет следующие действия:

- устанавливает регистры периферийных устройств, используемых загрузчиком в их значения по умолчанию
- устанавливает главный указатель стека пользовательской программы
- переходит к области памяти, указанной в принятом значении адреса + 4 (что соответствует адресу запуска пользовательской программы). Например, если принятый адрес равен 0x08000000, загрузчик перейдет на область памяти по адресу 0x08000004. В общем случае хост передает базовый адрес, где программа сама переходит в нужную область памяти.

Рисунок 10. Команда Go: сторона хоста

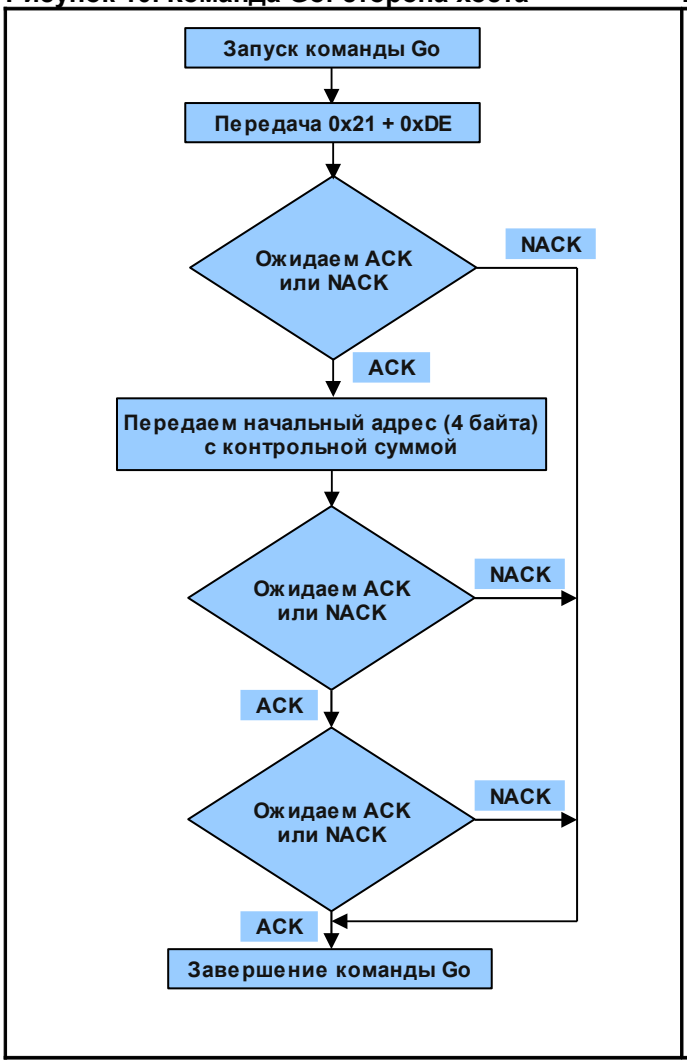
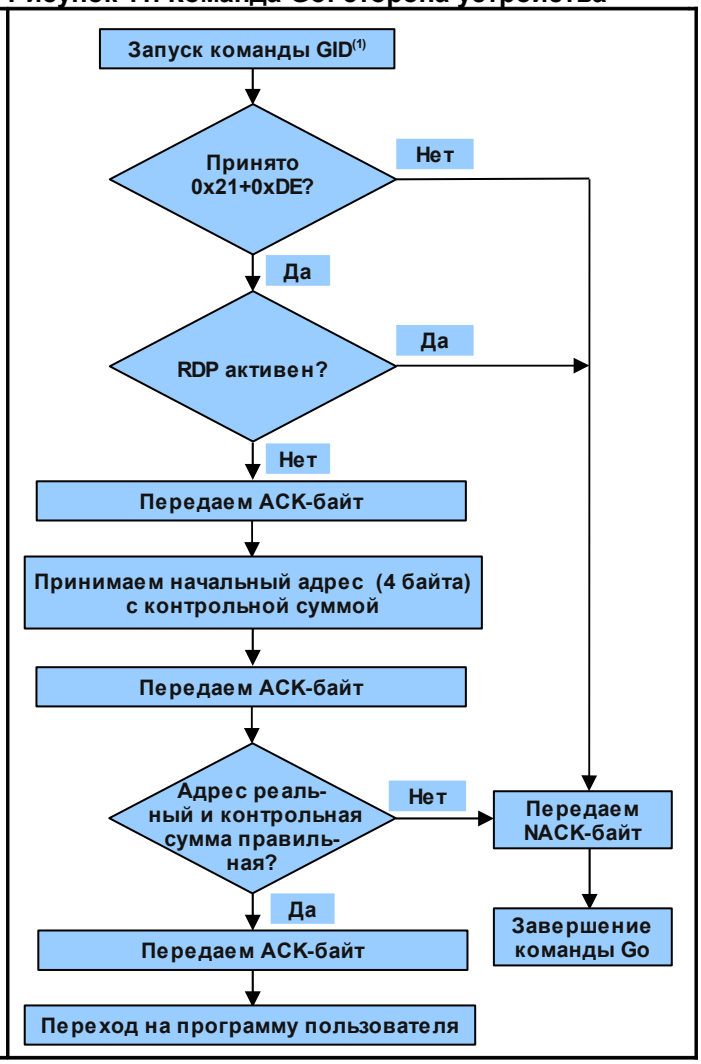


Рисунок 11. Команда Go: сторона устройства



Внимание: *1 Реальные адреса для команды Go находятся в ОЗУ или flash-памяти (смотри Секцию 3.1: Параметры загрузчика, независимые от устройства для более подробной информации о реальных адресах используемого вами устройства). Все другие адреса рассматриваются как нереальные и при их использовании устройство посылает NACK-байт.*

2 Когда программа загружена в ОЗУ, и затем произведен переход на нее, программа должна быть сконфигурирована для запуска с отсутствием перекрытия с первичной областью ОЗУ, используемой встроенной программой загрузчика (смотри Секцию 3.1: Параметры загрузчика, независимые от устройства для более подробной информации о распределении ОЗУ в используемом вами устройстве).

3 Переход на программу пользователя происходит только если она установила таблицу векторов корректно определяющую адреса программы.

Хост передает на STM32 следующую последовательность байтов:

Байт 1: 0x21

Байт 2: 0xDE

Ожидает приема ACK-байта

Байты 3-6: начальный адрес

байт 3: старший байт адреса

байт 6: младший байт адреса

Байт 7: контрольная сумма: операция XOR между байтами 3, 4, 5, 6

3.7 Команда «Write Memory» (Запись в память)

Команда Write Memory используется для записи данных по любому реальному адресу памяти (смотри пункт «Внимание» ниже) в ОЗУ, flash-памяти или в области байт настроек.

Когда загрузчик принимает команду Write Memory, он передает ACK-байт программе на компьютере. После передачи ACK-байта загрузчик ожидает приема начального адреса (4 байта, байт 1 - старший, байт 4 - младший) и байта контрольной суммы, необходимого для проверки принятого адреса. Для Области байтов настроек начальный адрес должен быть базовым адресом Области байтов настроек (смотри пункт «Внимание») для корректной записи данных в эту область.

Внимание: *1 Операции записи во flash-память или статическое ОЗУ производятся над последовательностью 32-битных слов и данные соответственно представляются в виде ряда из 4 байт.*

2 Смотри Секцию 3.1: Параметры загрузчика, независимые от устройства для более подробной информации о реальных адресах используемого вами устройства.

Если принятый адрес реальный и контрольная сумма правильная, загрузчик передает ACK-байт, иначе он передает NACK-байт программе на компьютере и завершает работу команды. Когда адрес реальный и контрольная сумма правильная, загрузчик выполняет следующие операции:

- получает байт, содержащий количество байтов для записи, равное N.

- принимает данные пользователя ((N + 1) байтов) и контрольную сумму (XOR между N и всеми байтами данных)
- записывает (прошивает) данные пользователя в память, начиная с принятого адреса
- при завершении команды, если операция записи произведена успешно, загрузчик передает ACK-байт, иначе он передает NACK-байт программе на компьютере и завершает работу команды

Максимальная длина блока данных для записи в STM32 равна 256 байтам.

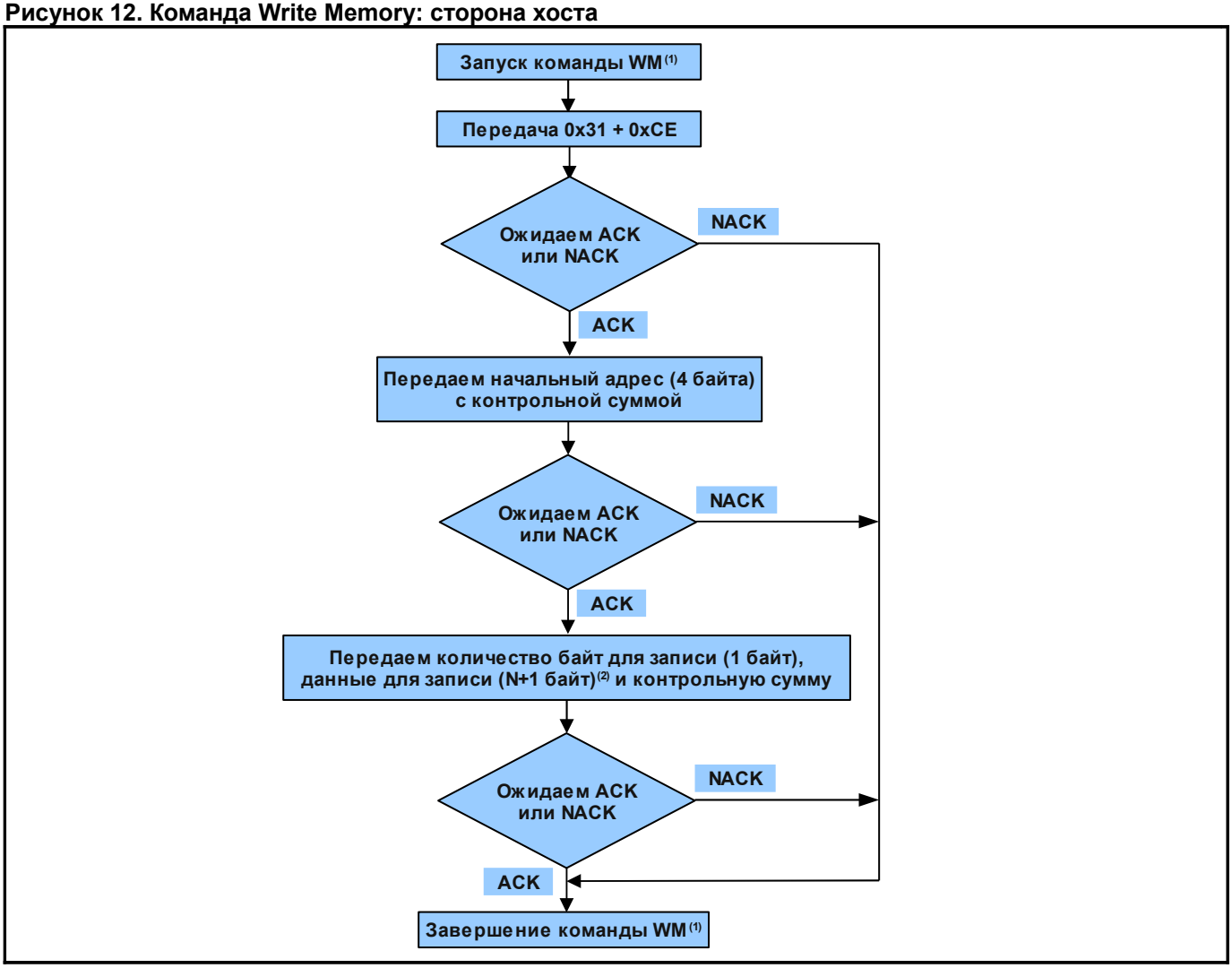
Если команда Write Memory выполняется для Области байт настроек, все настройки сбрасываются перед записью новых значений, и после завершения команды загрузчик генерирует системный сброс для применения новой конфигурации байта настроек.

Внимание: *1 Во время записи в ОЗУ вы должны заботиться о том, чтобы не было перекрытия с первичной областью ОЗУ, используемой встроенной программой загрузчика.*

2 В случае записи в защищенные от записи сектора, ошибки записи не формируются.

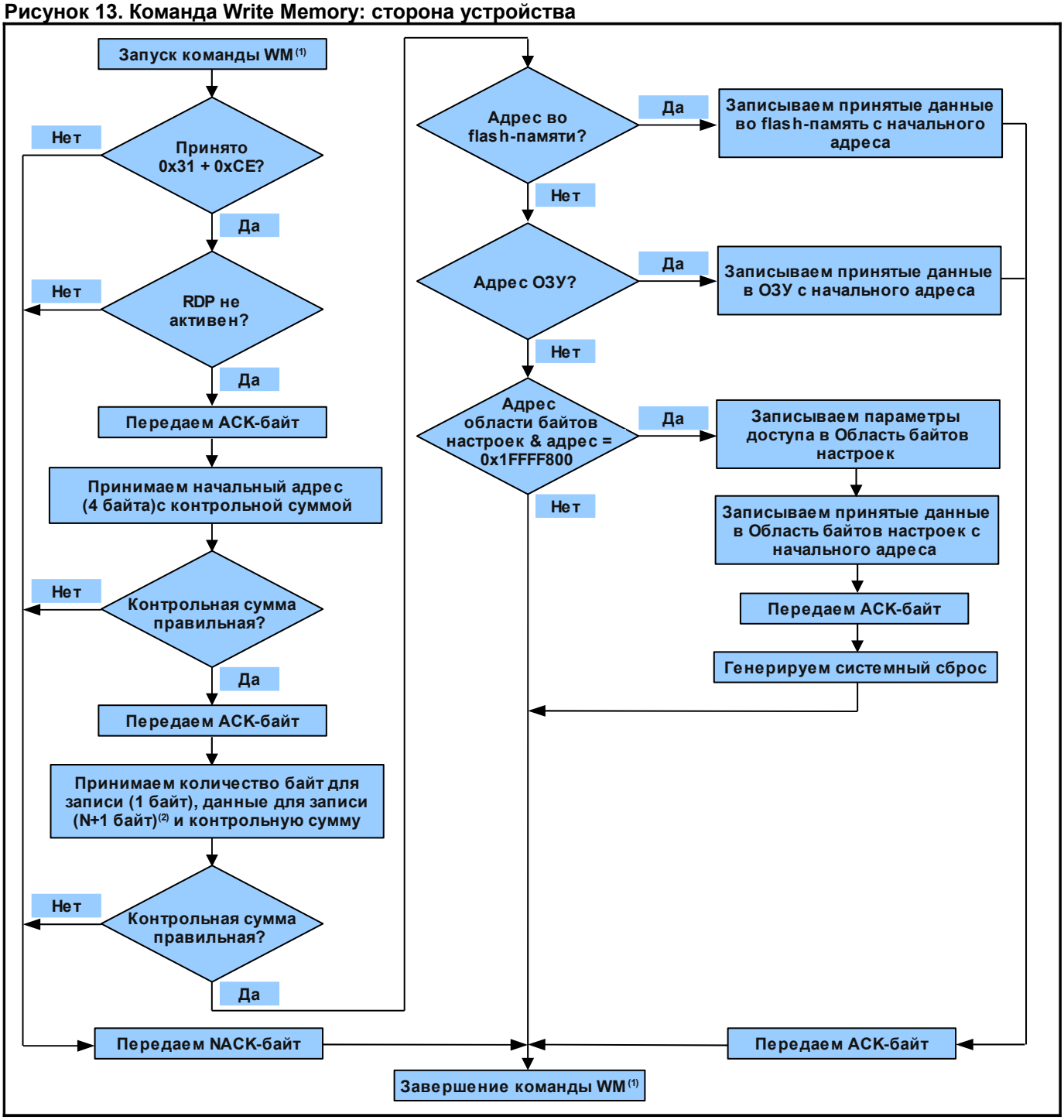
3 В случае реально не существующего (неправильного) начального адреса, ошибки записи не формируются.

Рисунок 12. Команда Write Memory: сторона хоста



1. WM = Write Memory. 2. (N + 1) всегда является сдвигом на 4 байта.

Рисунок 13. Команда Write Memory: сторона устройства



1. WM = Write Memory. 2. (N + 1) всегда является сдвигом на 4 байта.

Хост передает следующую последовательность байтов на STM32

Байт 1: 0x31

Байт 2: 0xCE

Ожидает приема ACK-байта

Байты 3-6: начальный адрес

байт 3: старший байт адреса

байт 6: младший байт адреса

Байт 7: контрольная сумма: операция XOR между байтами 3, 4, 5, 6

Ожидает приема ACK-байта

Байт 8: количество принимаемых байтов (0 < N ≤ 255)

N + 1 байтов данных: (максимально 256 байтов)

Байт контрольной суммы: операция XOR между N и N+1 байтами данных.

3.8 Команда «Erase Memory» (Очистка памяти)

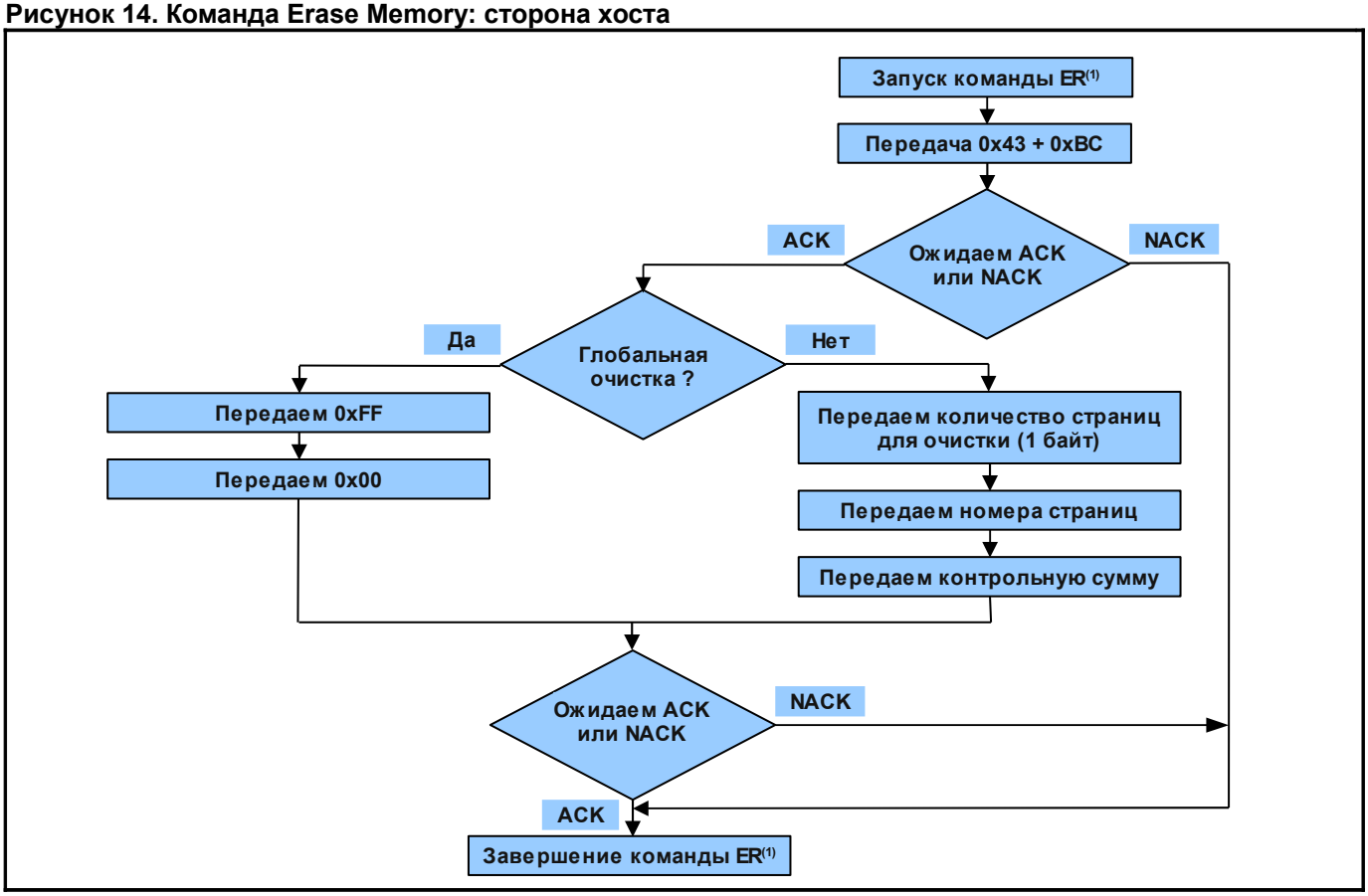
Команда Erase Memory позволяет хосту очищать страницы flash-памяти. Когда загрузчик принимает команду Erase Memory, он передает ACK-байт на хост. После передачи ACK-байта загрузчик принимает один байт (количество очищаемых страниц), коды страниц flash-памяти и байт контрольной суммы; если контрольная сумма правильная, то загрузчик очищает память и передает ACK-байт на хост, иначе он передает NACK-байт на хост и завершает работу команды.

Процесс выполнения команды Erase Memory:

1. загрузчик принимает один байт, содержащий число N (количество очищаемых страниц) - 1. N = 255 зарезервировано для требования глобальной очистки. Для (0 ≤ N ≤ 254) очищается (N + 1) страниц.
2. загрузчик принимает (N + 1) байт, каждый из которых содержит номер страницы.

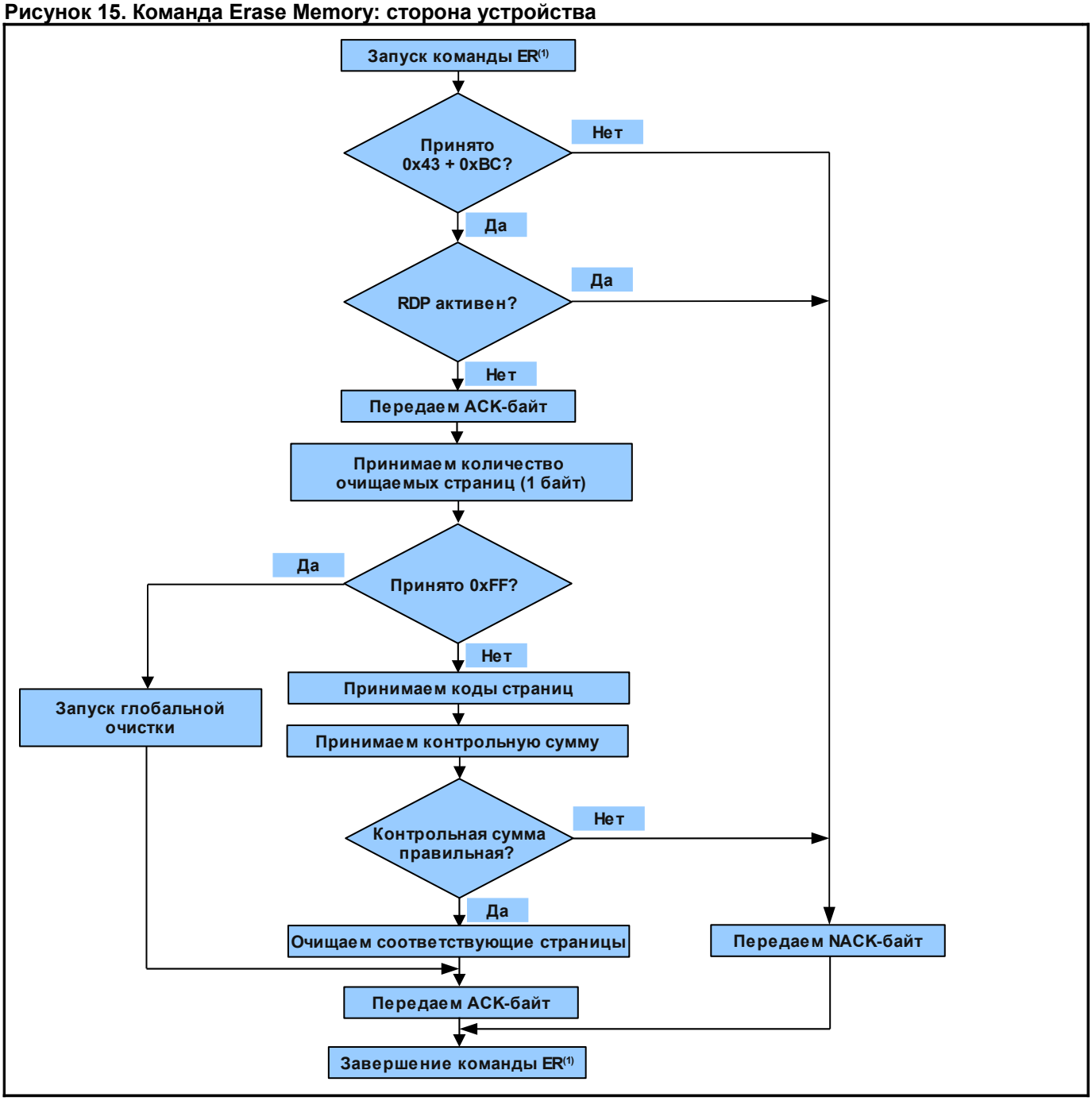
Внимание: *При выполнении операции очистки в защищенные от записи сектора ошибки очистки памяти не формируются.*

Рисунок 14. Команда Erase Memory: сторона хоста



1. ER = Erase Memory.

Рисунок 15. Команда Erase Memory: сторона устройства



1. ER = Erase Memory.

Внимание: *После передачи команды Erase Memory и ее контрольной суммы, если хост передает 0xFF и следующий за этим байт, отличный от 0x00, глобальная очистка не будет выполнена, хотя ACK-байт будет передан устройством.*

Хост передает следующую последовательность байтов на STM32:

Байт 1: 0x43

Байт 2: 0xBC

Ожидает приема ACK-байта

Байт 3: 0xFF или количество очищаемых страниц - 1 (0 ≤ N ≤ максимальное количество страниц)

Байт 4: 0x00 (в случае глобальной очистки) или (N+1 байтов (номера страниц) и контрольная сумма (операция XOR между N и N+1 байтами))

3.9 Команда «Extended Erase Memory» (Улучшенная очистка памяти)

Команда Extended Erase Memory позволяет хосту очищать страницы flash-памяти с использованием двухбайтного режима адресации. Когда загрузчик принимает команду Extended Erase Memory, он передает ACK-байт на хост. После передачи ACK-байта, загрузчик принимает два байта (количество очищаемых страниц), коды страниц flash-памяти (код одной страницы занимает 2 байта, первым идет старший байт) и байт контрольной суммы (операция XOR с переданными байтами); если контрольная сумма правильная, загрузчик очищает память и передает ACK-байт на хост. Иначе он передает NACK-байт на хост и завершает работу команды.

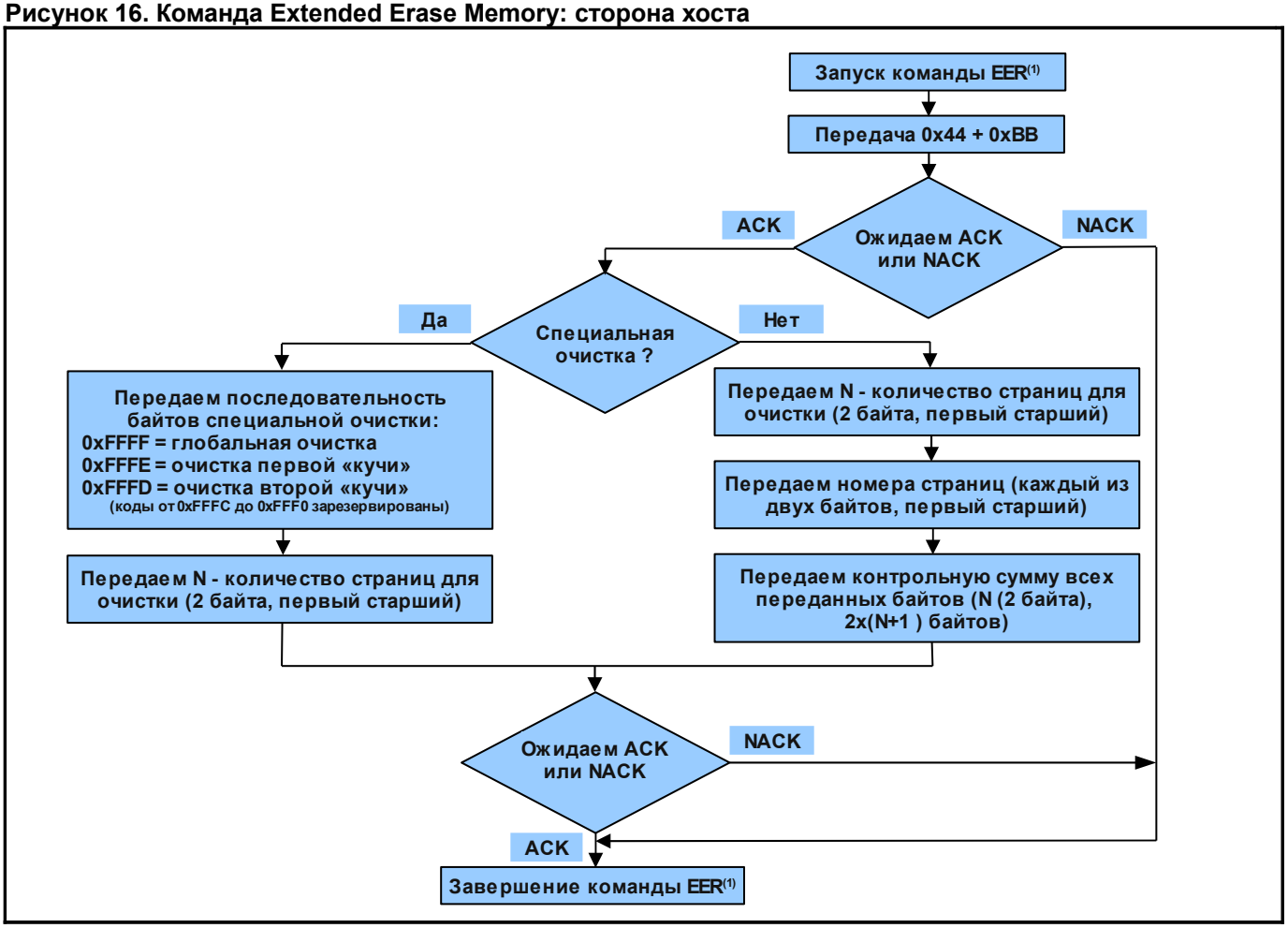
Процесс выполнения команды Extended Erase Memory:

1. Загрузчик принимает одно половинное слово (2 байта), что содержит N, количество удаляемых страниц:
   a) Для N = 0xFFFY (где Y от 0 до F) выполняются специальные режимы очистки:
      - 0xFFFF - глобальная очистка;
      - 0xFFFE - очистка первой «кучи» памяти;
      - 0xFFFD - очистка второй «кучи» памяти;
      - Коды ст 0xFFFC до 0xFFF0 зарезервированы.
   b) Для других значений 0 ≤ N ≤ максимальное количество страниц: удаляется N + 1 страниц.
2. Загрузчик принимает:
   a) В случае специальных режимов очистки один байт - контрольную сумму предыдущих байтов:
      - 0x00 для 0xFFFF
      - 0x01 для 0xFFFE
      - 0x02 для 0xFFFD
   a) В случае очистки (N + 1) страниц загрузчик принимает (2 x (N + 1)) байтов, каждое половинное слово содержит номер страницы (состоящий из 2 байтов, первый старший), затем контрольную сумму всех предыдущих байтов (из одного байта).

Внимание: *При выполнении операции очистки в защищенные от записи сектора ошибки очистки памяти не формируются.*

Максимальное число страниц зависит от типа микроконтроллера и не может быть изменено.

Рисунок 16. Команда Extended Erase Memory: сторона хоста



1. EER = Extended Erase Memory.

Хост передает следующую последовательность байтов на STM32F1xx:

Байт 1: 0x44

Байт 2: 0xBB

Ожидает приема ACK-байта

Байты 3-4: - Специальная очистка (0xFFFF, 0xFFFE или 0xFFFD)

ИЛИ

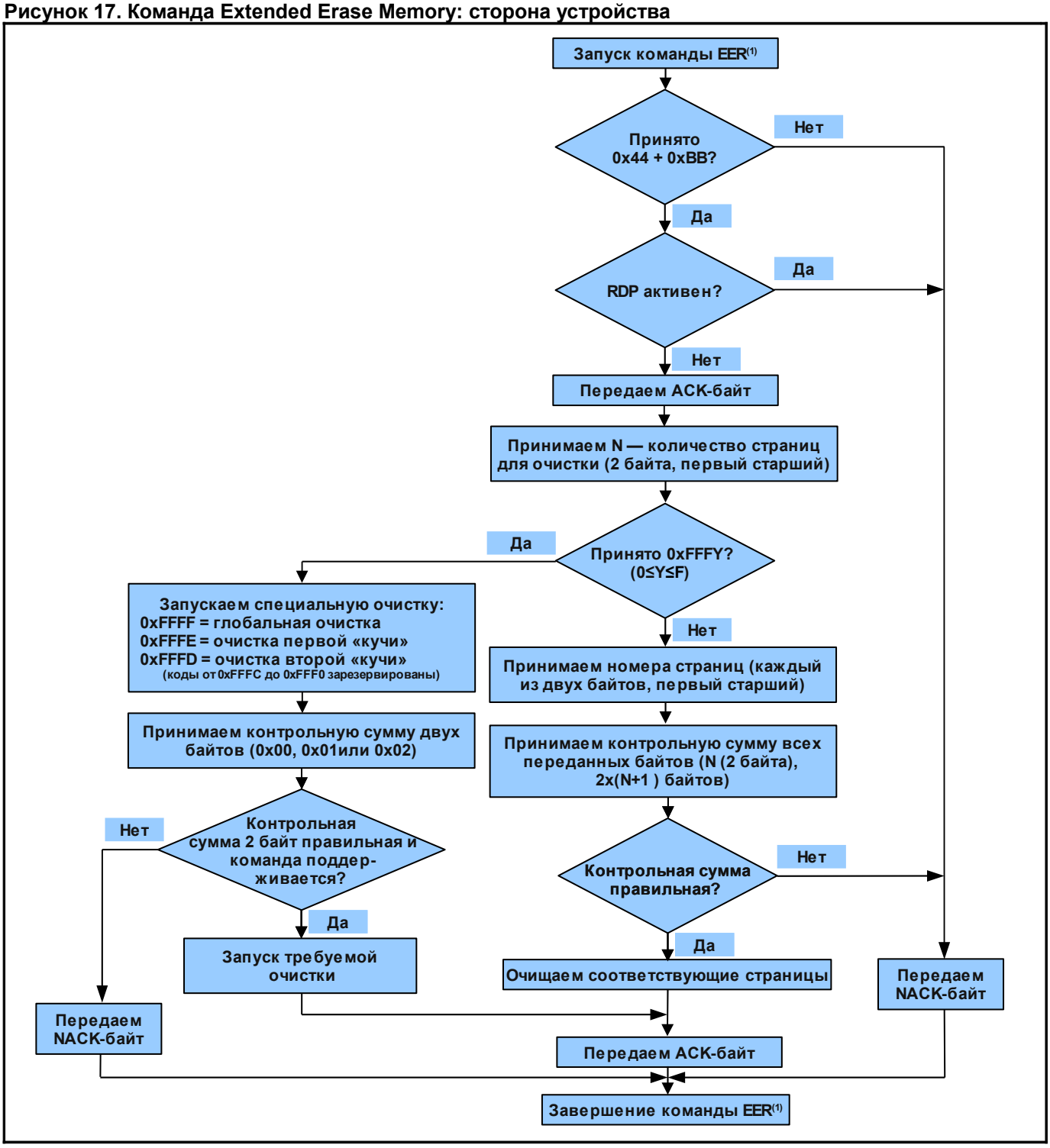
- Количество очищаемых страниц (N + 1 где 0 ≤ N < максимального количества страниц).

Остальные байты - Контрольная сумма байтов 3-4 в случае специальной очистки (0x00 если 0xFFFF, 0x01 если 0xFFFE или 0x02 если 0xFFFD).

ИЛИ

- (2 x (N + 1)) байтов (двухбайтные коды номеров страниц, первый старший), а затем контрольная сумма байтов 3-4 и следующих за ними байтов)

Рисунок 17. Команда Extended Erase Memory: сторона устройства



1. EER = Extended Erase Memory.

3.10 Команда «Write Protect» (Защита от записи)

Команда Write Protect используется для установления защиты от записи некоторых или всех секторов flash-памяти. Когда загрузчик принимает команду Write Protect, он передает ACK-байт на хост. После передачи ACK-байта загрузчик ожидает приема количества принимаемых байтов (защищенных секторов), после чего принимает коды секторов flash-памяти, посылаемые программой на компьютере.

В завершение команды Write Protect загрузчик передает ACK-байт и генерирует системный сброс для вступления в силу новой конфигурации перезаписанных байтов настроек.

Внимание: *В Секции 3.1: Параметры загрузчика, независимые от устройства более подробно рассказано о размерах секторов памяти используемого вами устройства.*

Последовательность выполнения команды Write Protect показана ниже:

- загрузчик принимает один байт, содержащий N, количество секторов для записи в защищенную область минус 1 (0 ≤ N ≤ 255)
- загрузчик принимает (N + 1) байтов, каждый байт содержит код сектора

Внимание: *1 Наличие общего количества секторов и номера защищенного сектора не проверяется, зто значит, что ошибки не формируются когда команда передается с неправильным количеством секторов или ошибочным номером сектора.*

2 Если выполняется вторая подряд команда Write Protect, сектора flash-памяти, защищенные первой командой, становятся незащищенными, и только сектора, указанные во второй команде Write Protect, становятся защищенными.

Рисунок 18. Команда Write Protect: сторона хоста

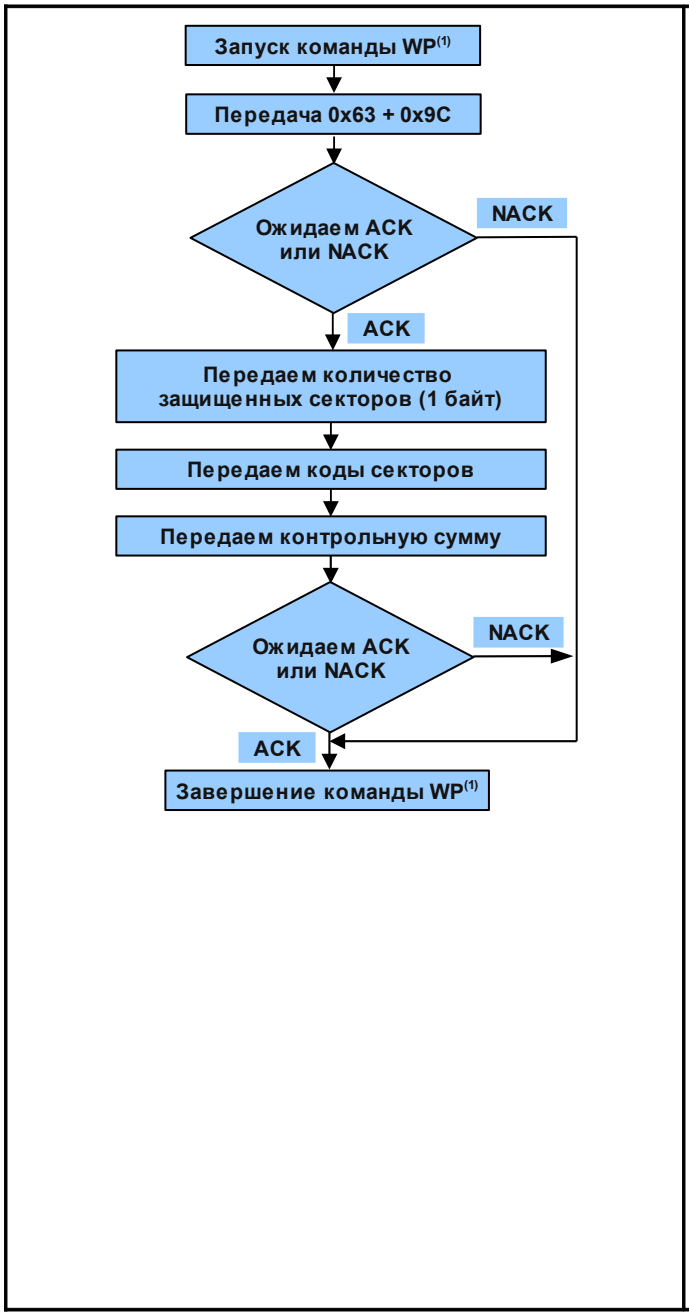
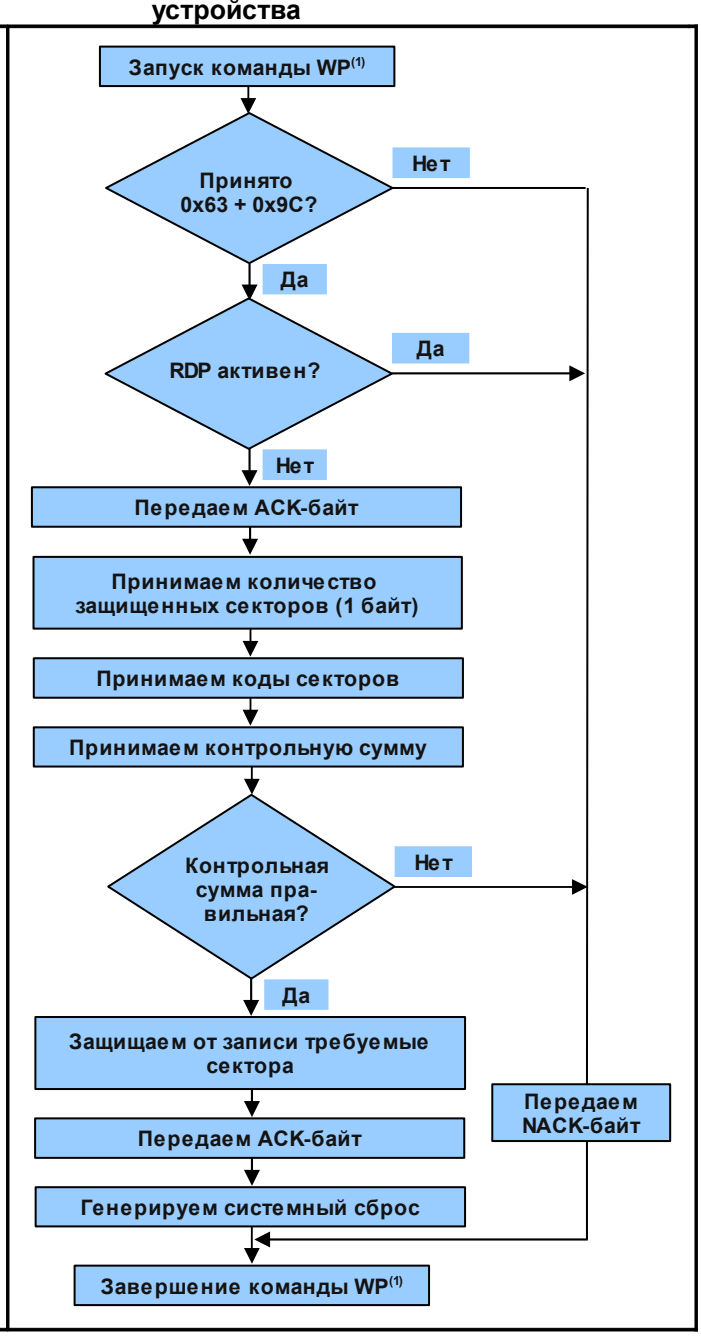


Рисунок 19. Команда Write Protect: сторона устройства



1. WP = Write Protect.

3.11 Команда «Write Unprotect» (Разрешение записи)

Команда Write Unprotect используется для отключения защиты от записи для всех секторов flash-памяти. Когда загрузчик принимает команду Write Unprotect, он передает ACK-байт на хост. После этой передачи ACK-байта, загрузчик отключает защиту от записи всех секторов flash-памяти. После завершения этой операции загрузчик передает ACK-байт.

В завершение команды Write Unprotect загрузчик передает ACK-байт и генерирует системный сброс для вступления в силу новой конфигурации перезаписанных байтов настроек.

Рисунок 20. Команда Write Unprotect: сторона хоста

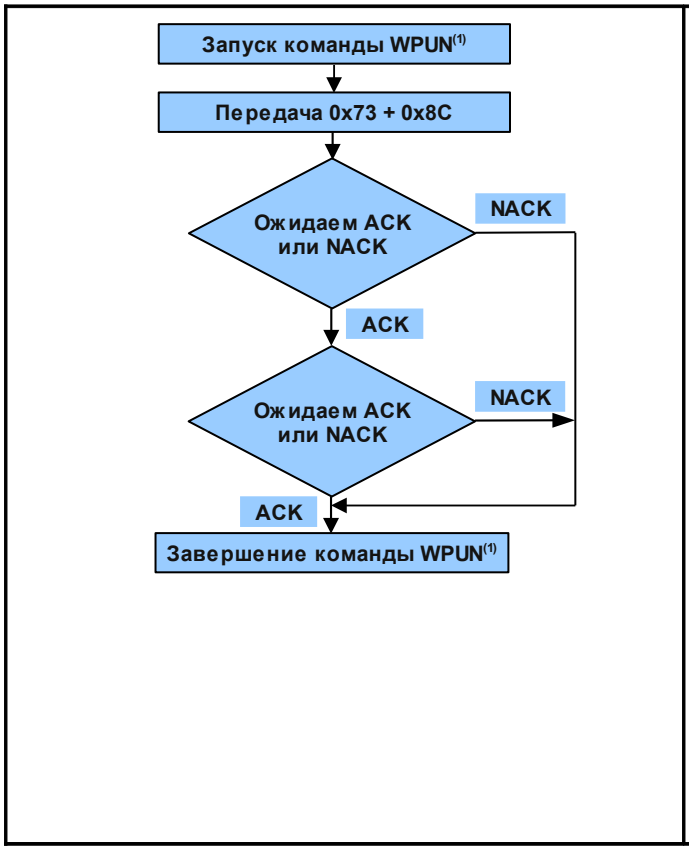
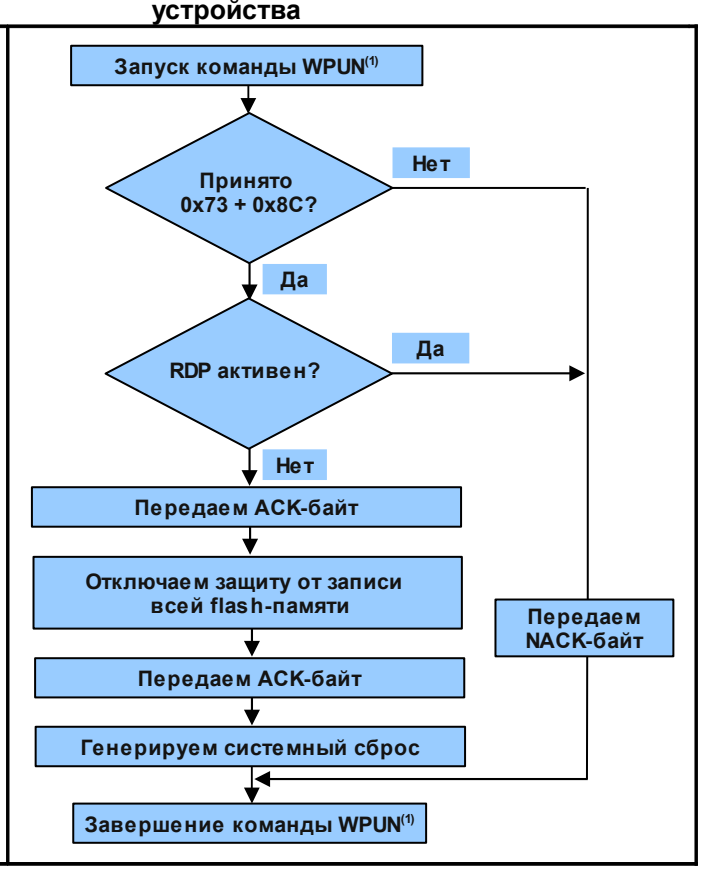


Рисунок 21. Команда Write Unprotect: сторона устройства



1. WPUN = Write Unprotect.

3.12 Команда «Readout Protect» (Защита от чтения)

Команда Readout Protect используется для разрешения защиты от чтения flash-памяти. Когда загрузчик принимает команду Readout Protect, он передает ACK-байт на хост. После передачи ACK-байта загрузчик разрешает защиту от чтения flash-памяти.

В завершение команды Readout загрузчик передает ACK-байт и генерирует системный сброс для вступления в силу новой конфигурации перезаписанных байтов настроек.

Рисунок 22. Команда Readout Protect: сторона хоста

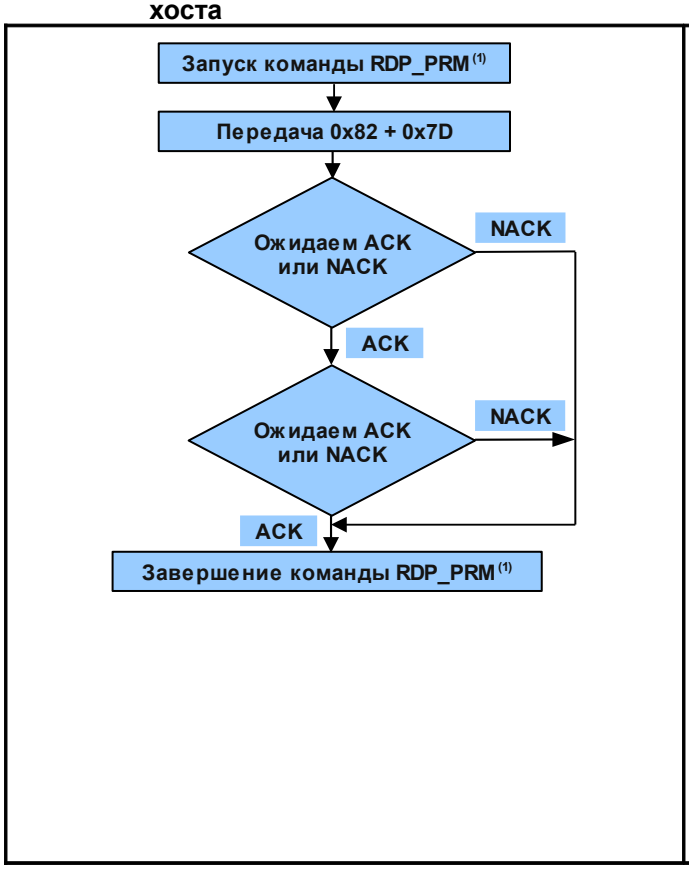
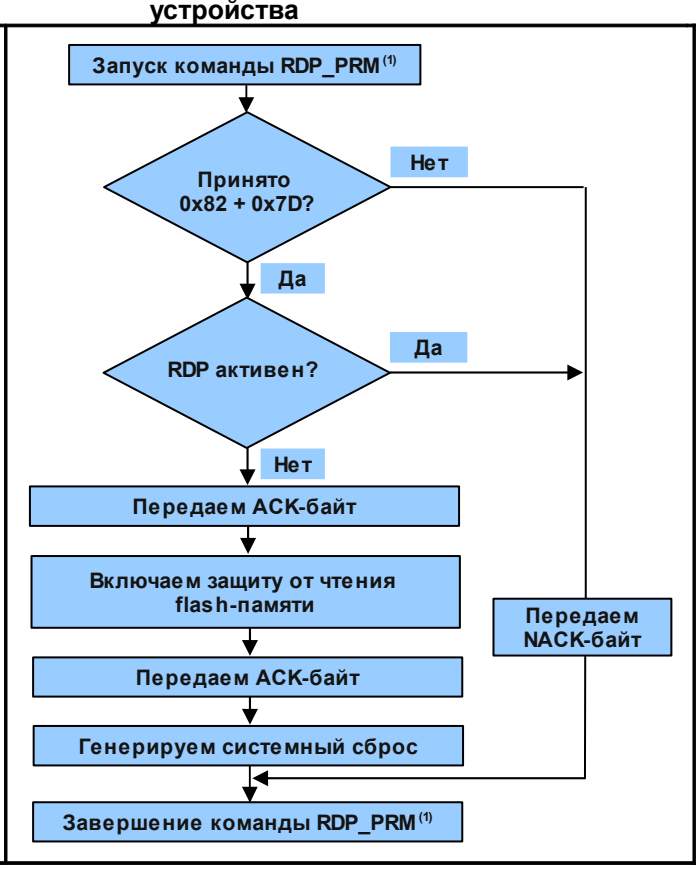


Рисунок 23. Команда Readout Protect: сторона устройства



1. RDP_PRM = Readout Protect.

3.13 Команда «Readout Unprotect» (Отключение защиты от чтения)

Команда Readout Unprotect используется для отключения защиты от чтения flash-памяти. Когда загрузчик принимает команду Readout Unprotect, он передает ACK-байт на хост. После передачи ACK-байта загрузчик очищает все сектора ОЗУ и отключает защиту от чтения всей flash-памяти. Если операция очистки завершена успешно, загрузчик отключает RDP.

Если операция очистки завершилась неправильно, загрузчик передает NACK-байт, а защита от чтения остается активной.

В завершение команды Readout Unprotect загрузчик передает ACK-байт и генерирует системный сброс для вступления в силу новой конфигурации перезаписанных байтов настроек.

Рисунок 24. Команда Readout Unprotect: сторона хоста

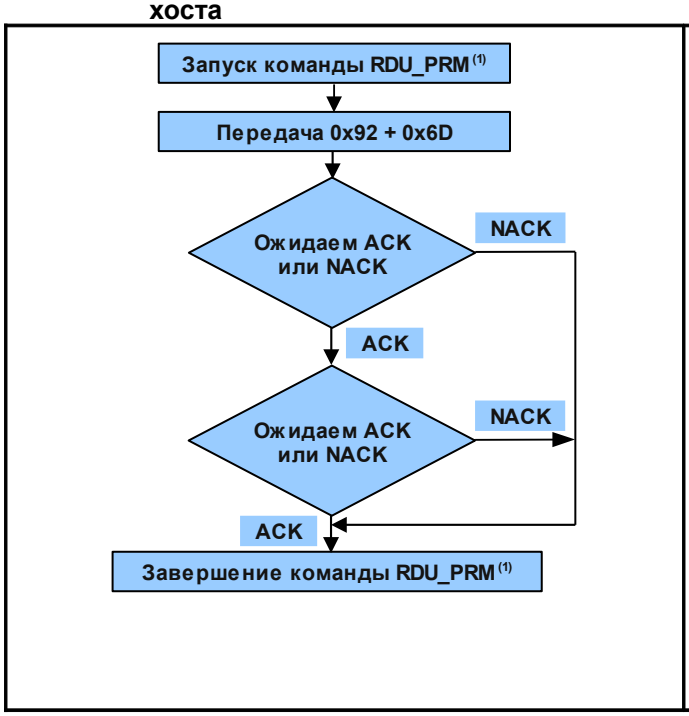
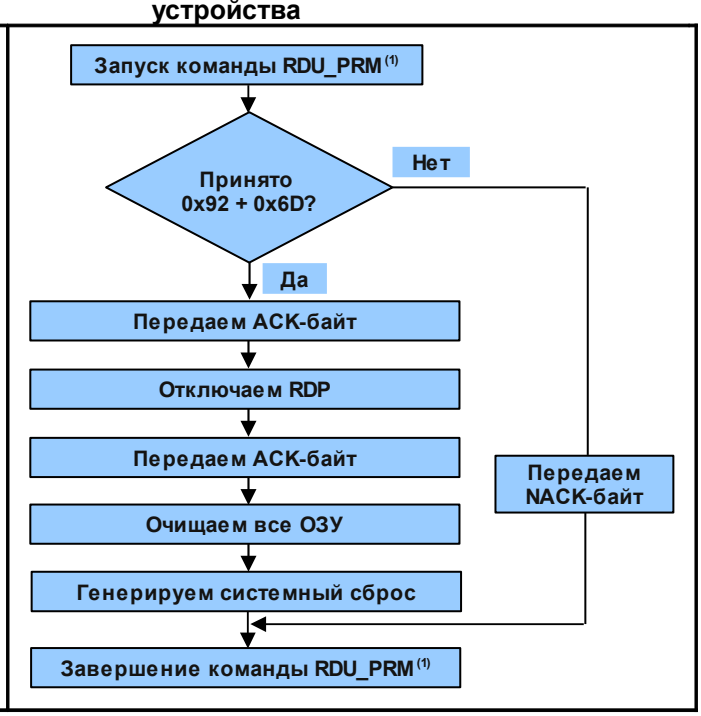


Рисунок 25. Команда Readout Unprotect: сторона устройства



1. RDU_PRM = Readout Unprotect.

4 Развитие протокола загрузчика по версиям

В *Таблице 2* показаны версии загрузчика.

Таблица 2. Версии протокола загрузчика

Версия	Описание
V2.0	Начальная версия загрузчика
V2.1	- Обновление команды Go: определение указателя основного стека - Обновление команды Go: возвращение NACK-байта когда адрес перехода в области байтов настроек или в системной памяти - Обновление команды Get ID: возвращение ID устройства в виде 2 байтов - Обновление версии загрузчика до V2.1
V2.2	- Обновление команд Read Memory, Write Memory и Go: запрещение доступа к начальным байтам ОЗУ, используемым загрузчиком и ответ в виде NACK-байта при попытке такого доступа - Обновление команды Readout Unprotect: заполнение всего ОЗУ нулями (0x0) перед отключением защиты от чтения flash-памяти (RDP)
V3.0	- Добавлена команда Extended Erase для поддержки количества страниц памяти более чем 256 и разделения памяти на «кучи» при общей очистке. - Команда Erase не модифицировалась в этой версии, но после добавления команды Extended Erase она больше не поддерживается (команды Erase и Extended Erase взаимоисключающие)

5 История исправлений

Таблица 3. Истроия исправлений в документе

Дата	Исправление	Изменения
9 марта 2010	1	Исходный документ
20 апреля 2010	2	*Таблица 1: Команды загрузчика USART:* добавлена команда Extended Erase; удалена сноска 2, описывающая защиту от чтения из команды Readout Protect. *Безопасность связи (стр. 7):* исправлен *Пункт 1* *Секция 3.2: Команда Get:* обновлен байт 10 Обновлен *Рисунок 10: Команда Go: Сторона хоста* для отсутствующего ACK-байта *Секция 3.7: Команда Write Memory:* Добавлены пункты «Внимание» на стр. 15 *Рисунок 12* и *Рисунок 13:* добавлены пункты относительно N + 1 *Таблица 2: Версии протокола загрузчика:* добавлена версия v3.0

Просим прочитать внимательно

Информация в этом документе предоставлена исключительно в связи с ST-изделиями. STMicroelectronics NV и ее дочерние подразделения («ST») сохраняют за собой право производить изменения, коррекции, модификации или улучшения в этом документе, в изделиях и их возможностях, описанных в нем, в любое время без каких-либо предупреждений.

Все ST-изделия имеют соответствующие сроки и условия продажи.

Покупатели ответственны только за выбор, подбор и использование ST-изделий, а также их возможностей, описанных в этом документе. ST не берет на себя каких бы то ни было обязательств относительно выбора, подбора или использования ST-изделий и их возможностей, описанных в данном документе.

Не разрешается, прямо или косвенно, использование любых интеллектуальных прав из этого документа без ссылки на данный документ. Применение какой-либо части этого документа в сторонних изделиях или любых интеллектуальных прав, содержащихся в этом документе или рассматриваемых как основа для использования любыми способами какого бы то ни было стороннего изделия приводит к запрету использование таких сторонних изделий.

ПОКА НЕ УКАЗАНЫ ДАЛЬНЕЙШИЕ СРОКИ И УСЛОВИЯ ПРОДАЖ ST ОТКАЗЫВАЕТСЯ ДАТЬ КАКИЕ-ЛИБО ТОЧНЫЕ ИЛИ ПРЕДПОЛАГАЕМЫЕ ОСНОВАНИЯ ДЛЯ ПРИЗНАНИЯ ИСПОЛЬЗОВАНИЯ И/ИЛИ ПРОДАЖИ ST-ИЗДЕЛИЙ В ТОМ ЧИСЛЕ И В СЛУЧАЕ ОТСУТСТВИЯ ГОДНОСТИ ДЛЯ ПРОДАЖИ, СООТВЕТСТВИЯ РЕДКОМУ НАЗНАЧЕНИЮ (И ИХ ЭКВИВАЛЕНТОВ СОГЛАСНО ЗАКОНАМ), А ТАКЖЕ В СЛУЧАЕ НАРУШЕНИЯ КАКОГО-ЛИБО ПАТЕНТА, АВТОРСКОГО ПРАВА ИЛИ ДРУГИХ ИНТЕЛЛЕКТУАЛЬНЫХ ПРАВ.

ПОКА ТОЧНО НЕ ЗАДОКУМЕНТИРОВАНО АВТОРСТВО ST-ПРЕДСТАВИТЕЛЯ, ST-ИЗДЕЛИЯ НЕ РАЗРЕШЕНО ИСПОЛЬЗОВАТЬ В АРМИИ, АВИАЦИИ, КОСМОСЕ, ДЛЯ СПАСЕНИЯ ИЛИ ПОДДЕРЖАНИЯ ЖИЗНИ (В МЕДИЦИНЕ), А ТАКЖЕ В ИЗДЕЛИЯХ ИЛИ СИСТЕМАХ, ГДЕ ПОЛОМКА ИЛИ НЕИСПРАВНОСТЬ ЯВЛЯЕТСЯ РЕЗУЛЬТАТОМ СОЗНАТЕЛЬНОГО ВРЕДА, УНИЧТОЖЕНИЯ ИЛИ ЗАГРЯЗНЕНИЯ ОКРУЖАЮЩЕЙ СРЕДЫ. ST-ИЗДЕЛИЯ, НЕ ОПРЕДЕЛЕННЫЕ КАК «АВТОМАТИЧЕСКИЙ УРОВЕНЬ», МОГУТ ИСПОЛЬЗОВАТЬСЯ ТОЛЬКО В АВТОМАТИЧЕСКИХ УСТРОЙСТВАХ НА ПОЛНУЮ ОТВЕТСТВЕННОСТЬ (СТРАХ И РИСК) ПОЛЬЗОВАТЕЛЯ.

Перепродажа ST-изделий с различными отклонениями от заявленных характеристик и технических особенностей устанавливается вне этого документа и полностью освобождается от установленных разрешений для ST-изделий или их возможностей, описанных в этом документе. Ее описание не будет создаваться или продлеваться ST в любом виде.

ST и ее эмблема (логотип) являются торговыми марками или зарегистрированными торговыми знаками ST в различных странах.

Информация в этом документе отменяет и заменяет всю информацию, предоставленную ранее.

Эмблема (логотип) ST является зарегистрированным товарным знаком фирмы STMicroelectronics.
Все другие названия являются собственностью соответствующих владельцев.

Группа компаний STMicroelectronics

Австрия - Бельгия - Бразилия - Великобритания - Германия - Гонконг - Греция - Израиль - Индия - Испания - Италия - Канада - Китай - Малазия - Мальта - Морокко - Сингапур - США - Филлипины - Финляндия - Франция - Швейцария - Швеция - Япония

www.st.com