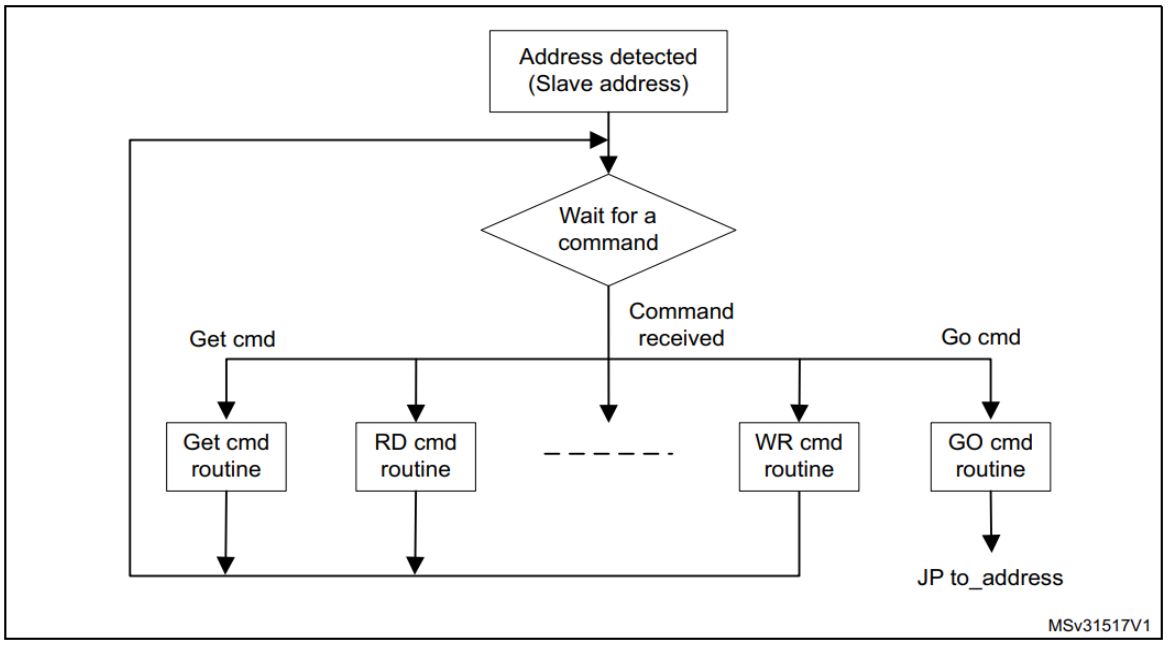
1 Последовательность работы I2C загрузчика

Последовательность работы кода загрузчика I2C для микроконтроллеров STM32, основанных на ядре Arm®(a), показана на рисунке 1.

Рисунок 1. I2C загрузчик для STM32



***Примечание**:* Адрес подчиненного устройства I2C для каждого загрузчика продукта указан в AN2606.

После входа в режим загрузки системной памяти и настройки микроконтроллера STM32 (более подробную информацию см. в примечаниях к применению режима загрузки системной памяти STM32), код загрузчика начинает сканировать вывод линии I2C_SDA, ожидая обнаружения собственного адреса на шине. После обнаружения микропрограмма загрузчика I2C начинает получать команды хоста.

2 Набор команд загрузчика

Команды «без растяжения» (No-Stretch) поддерживаются, начиная с версии протокола V1.1 и позволяют лучше управлять командами, когда хосту приходится ждать значительное время, прежде чем загрузчик завершит операцию.

По возможности рекомендуется использовать команды «без растяжения» (No-Stretch) вместо эквивалентных обычных команд.

Поддерживаемые команды перечислены в таблице 2.

Таблица 2. Команды загрузчика I2C

Команда (1)	Код команды	Описание
Get (2)	0x00	Получает версию и разрешенные команды, поддерживаемые текущей версией загрузчика.
Get Version (2)	0x01	Получает версию загрузчика.
Get ID (2)	0x02	Получает идентификатор чипа
Read Memory	0x11	Читает до 256 байт памяти, начиная с адреса, указанного приложением.
Go (3)	0x21	Переходит к коду пользовательского приложения, расположенному во внутренней флэш-памяти.
Write Memory (3)	0x31	Записывает в память до 256 байт, начиная с адреса, указанного приложением.
No-Stretch Write Memory (3)(4)	0x32	Записывает в память до 256 байт, начиная с адреса, указанного приложением, и возвращает состояние занятости во время выполнения операции.
Erase	0x44	Стирает от одной до всех страниц или секторов флэш-памяти, используя режим двухбайтовой адресации.
No-Stretch Erase (3)(4)	0x45	Стирает от одной до всех страниц или секторов флэш-памяти, используя режим двухбайтовой адресации, и возвращает состояние занятости во время выполнения операции.
Special	0x50	Общая команда, которая позволяет добавлять новые функции в зависимости от ограничений продукта, не добавляя новую команду для каждой функции.
Extended Special	0x51	Общая команда, которая позволяет пользователю отправлять больше данных по сравнению со специальной командой.
Write Protect	0x63	Включает защиту от записи для некоторых секторов.
No-Stretch Write Protect (4)	0x64	Включает защиту от записи для некоторых секторов и возвращает состояние занятости во время выполнения операции.
Write Unprotect	0x73	Отключает защиту от записи для всех секторов флэш-памяти.
No-Stretch Write Unprotect (4)	0x74	Отключает защиту от записи для всех секторов флэш-памяти и возвращает состояние занятости во время выполнения операции.
Readout Protect	0x82	Включает защиту от чтения.
No-Stretch Readout Protect (4)	0x83	Включает защиту от чтения и возвращает состояние занятости во время выполнения операции.
Readout Unprotect (2)	0x92	Отключает защиту от чтения.
No-Stretch Readout Unprotect (2)(4)	0x93	Отключает защиту от чтения и возвращает состояние занятости во время выполнения операции.
No-Stretch Get Memory Checksum (2)	0xA1	Получает значение контрольной суммы CRC для области памяти на основе ее смещения и длины.

1. Если получена запрещенная команда или если во время выполнения команды возникает ошибка, загрузчик отправляет байт NACK и возвращается к проверке команды.
2. Защита от чтения. Когда параметр RDP (защита от чтения) активен, доступно только это ограниченное подмножество команд. Все остальные команды получают NACK и не влияют на устройство. После удаления RDP другие команды становятся активными.
3. Обратитесь к техническому описанию продукта STM32 и к AN2606, чтобы узнать области памяти, допустимые для этих команд.
4. Команды No-Stretch доступны только с протоколом I2C версии 1.1.

Команды «без растяжения» (**No-Stretch commands**)

Команды «без растяжения» (No-Stretch) позволяют выполнять операции Write, Erase, Write Protect, Write Unprotect, Read Protect и Read Unprotect без растягивания линии I2C, пока загрузчик выполняет операцию. Возможна связь с другими устройствами на шине, пока загрузчик выполняет операции, требующие времени ожидания.

Отличие этих команд от стандартных заключается в том, что когда Хост запрашивает подтверждение (ACK или NACK) в конце команды, вместо удержания шины I2C загрузчик отвечает третьим

состоянием — занято BUSY(0x76). Когда хост получает состояние BUSY, он снова опрашивает состояние и считывает один байт, пока не получит ответ ACK или NACK.

Безопасность связи (Communication safety)

Все сообщения от хоста программирования к устройству проверяются по контрольной сумме. Полученные блоки байтов данных подвергаются операции XOR. Байт, содержащий вычисленное XOR всех предыдущих байтов, добавляется в конец каждого сообщения (байт контрольной суммы). При выполнении XOR всех полученных байтов данных и контрольной суммы результат в конце пакета должен быть 0x00.

Для каждой команды хост отправляет байт с кодом команды + байт инверсии (NOT) кода команды. (*В AN4221 указано что делается XOR кода команды, что не верно*).

Каждый пакет либо принимается (ответ ACK), либо отбрасывается (ответ NACK):

- ACK = 0x79
- NACK = 0x1F

С командами «без растяжения» (No-Stretch) состояние занятости BUSY отправляется вместо ACK или NACK:

- BUSY = 0x76

Примечание: Кадр от Хоста может быть одним из следующих:

- Передача кадра команды (Send Command frame): Хост инициирует связь в качестве Ведущего передатчика и отправляет на устройство два байта: код команды + NOT кода команды.

- Ожидание кадра ACK/NACK (Wait for ACK or NACK frame): Хост инициирует связь в качестве Ведущего получателя и получает от устройства один байт: ACK, NACK или BUSY.

- Получение кадра данных (Receive data frame): Хост инициирует связь в качестве Ведущего получателя и получает ответ от устройства. Количество получаемых байтов зависит от команды.

- Передача кадра данных (Send data frame): Хост инициирует связь в качестве Ведущего передатчика и отправляет необходимые байты на устройство. Количество передаваемых байтов зависит от команды.

Внимание!

Для команд Write, Erase и Read Unprotect хост должен соблюдать соответствующие временные интервалы (т. е. запись страницы, стирание сектора), указанные в технических описаниях продуктов. Например, при запуске команды Erase хост должен ждать (до последнего ACK команды) в течение времени, эквивалентного максимальному времени стирания сектора/страницы, указанному в таблице данных (или, по крайней мере, типичному времени стирания сектора/страницы).

Внимание!

Для связи I2C реализован механизм тайм-аута, его необходимо соблюдать для корректного выполнения команд загрузчика. Этот тайм-аут реализуется между двумя кадрами I2C в одной команде. Например, для команды записи в память между кадром отправки команды и кадром отправки адресной памяти вставляется тайм-аут. Кроме того, один и тот же период тайм-аута вставляется между двумя последовательными моментами приема или передачи данных в одном и том же кадре I2C. Если время ожидания истекло, генерируется сброс системы, чтобы избежать сбоя загрузчика. Обратитесь к разделу, посвященному времени подключения I2C AN2606, чтобы получить значение тайм-аута I2C для каждого продукта STM32.

2.1 Команда Get

Команда Get позволяет пользователю получить версию загрузчика и поддерживаемые команды. Когда загрузчик получает команду Get, он передает хосту версию загрузчика и поддерживаемые коды команд, как показано на рисунке 2.

Рисунок 2. Команда Get: сторона хоста

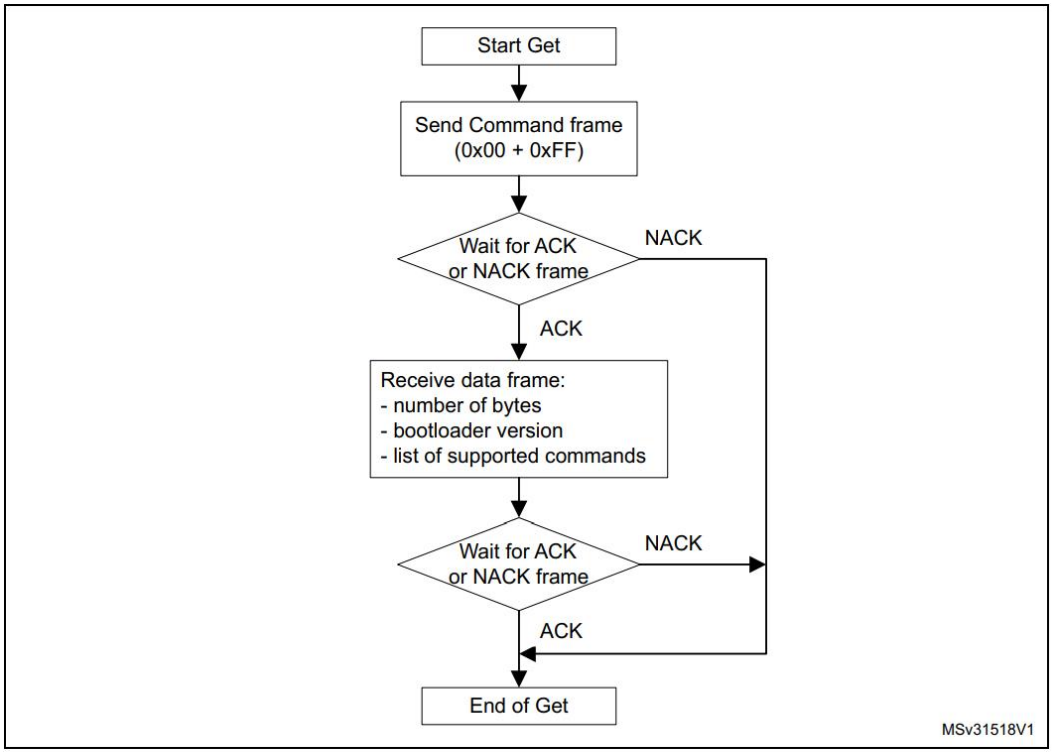
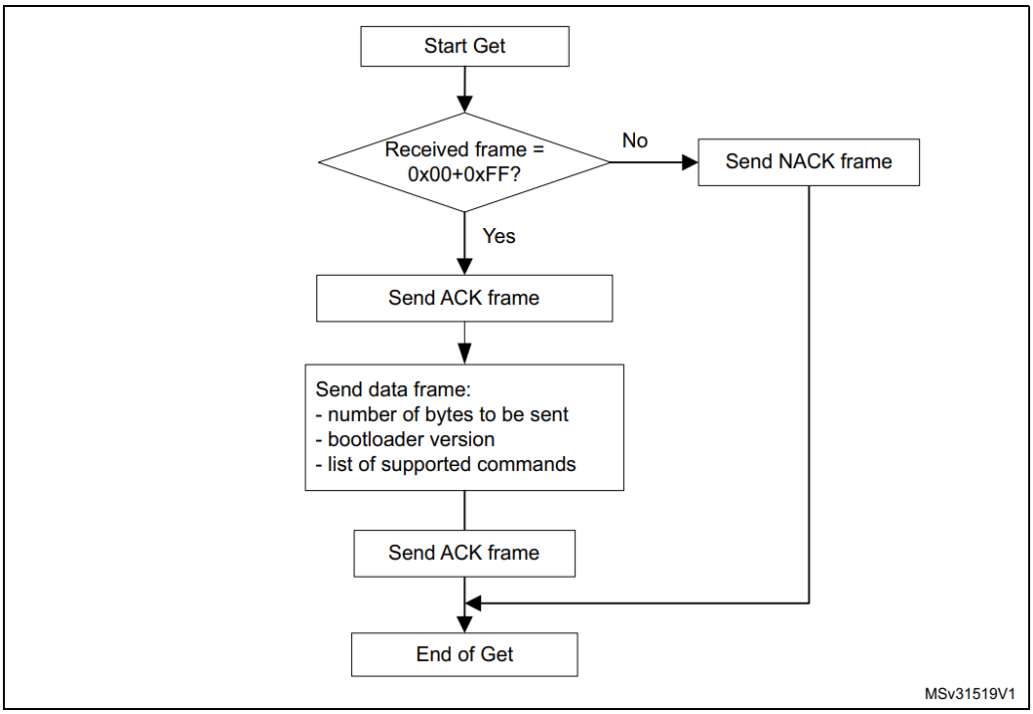


Рисунок 3. Команда Get: сторона устройства



STM32 отправляет байты следующим образом:

<u>для протокола V1.0:</u>

– Byte 1: ACK

– Byte 2: N = 11 = Number of bytes to follow - 1, except current and ACKs

– Byte 3: Bootloader version 0x10 = Version 1.0

– Byte 4: 0x00 - Get command

– Byte 5: 0x01 - Get Version

– Byte 6: 0x02 - Get ID
– Byte 7: 0x11 - Read Memory command
– Byte 8: 0x21 - Go command
– Byte 9: 0x31 - Write Memory command
– Byte 10: 0x44 - Erase command
– Byte 11: 0x63 - Write Protect command
– Byte 12: 0x73 - Write Unprotect command
– Byte 13: 0x82 - Readout Protect command
– Byte 14: 0x92 - Readout Unprotect command
– Byte 15: ACK

для протокола V1.1:

– Byte 1: ACK
– Byte 2: N = 17 = Number of bytes to follow - 1, except current and ACKs
– Byte 3: Bootloader version 0x11 = Version 1.1
– Byte 4: 0x00 - Get command
– Byte 5: 0x01 - Get Version
– Byte 6: 0x02 - Get ID
– Byte 7: 0x11 - Read Memory command
– Byte 8: 0x21 - Go command
– Byte 9: 0x31 - Write Memory command
– Byte 10: 0x44 - Erase command
– Byte 11: 0x63 - Write Protect command
– Byte 12: 0x73 - Write Unprotect command
– Byte 13: 0x82 - Readout Protect command
– Byte 14: 0x92 - Readout Unprotect command
– Byte 15: 0x32 - No-Stretch Write Memory command
– Byte 16: 0x45 - No-Stretch Erase command
– Byte 17: 0x64 - No-Stretch Write Protect command
– Byte 18: 0x74 - No-Stretch Write Unprotect command
– Byte 19: 0x83 - No-Stretch Readout Protect command
– Byte 20: 0x93 - No-Stretch Readout Unprotect command
– Byte 21: ACK

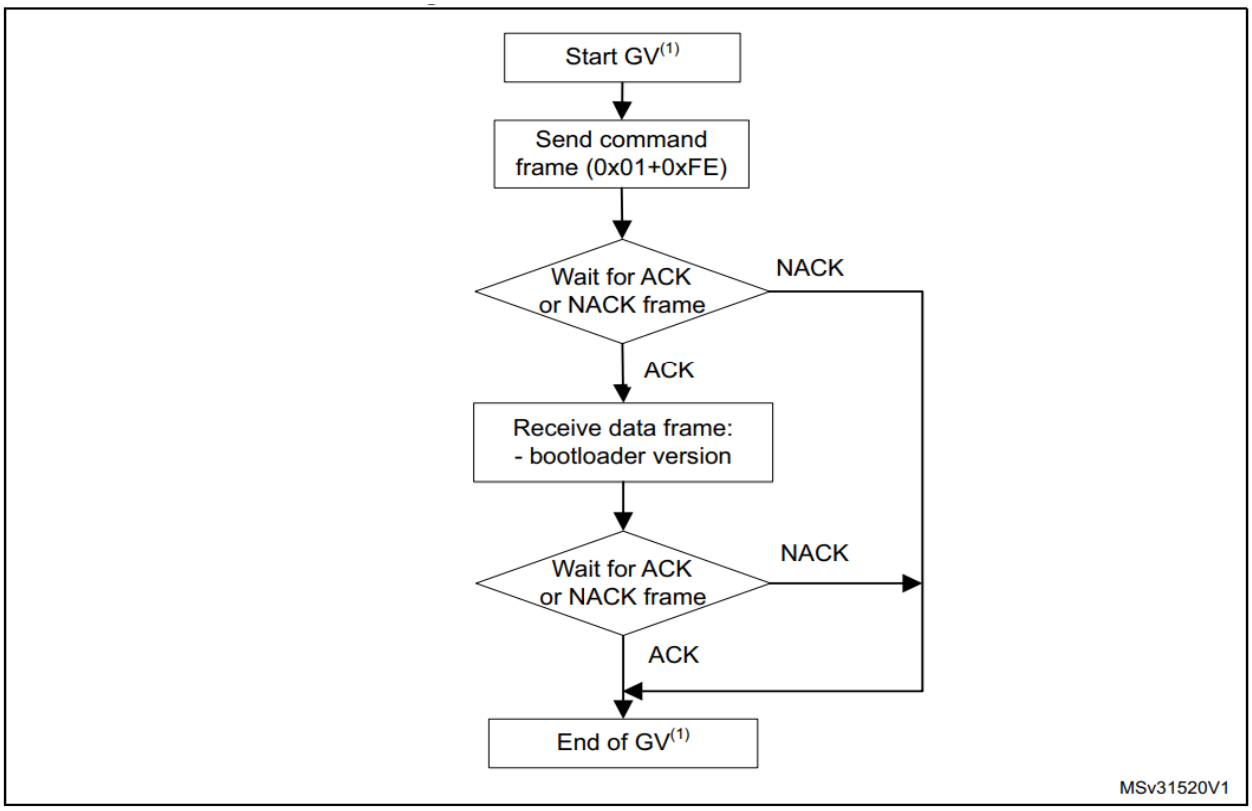
для протокола V1.2:

– Byte 1: ACK
– Byte 2: N = 18 = Number of bytes to follow - 1, except current and ACKs
– Byte 3: Bootloader version 0x12 = Version 1.2
– Byte 4: 0x00 - Get command
– Byte 5: 0x01 - Get Version
– Byte 6: 0x02 - Get ID
– Byte 7: 0x11 - Read Memory command
– Byte 8: 0x21 - Go command
– Byte 9: 0x31 - Write Memory command
– Byte 10: 0x44 - Erase command
– Byte 11: 0x63 - Write Protect command
– Byte 12: 0x73 - Write Unprotect command
– Byte 13: 0x82 - Readout Protect command
– Byte 14: 0x92 - Readout Unprotect command
– Byte 15: 0x32 - No-Stretch Write Memory command
– Byte 16: 0x45 - No-Stretch Erase command
– Byte 17: 0x64 - No-Stretch Write Protect command
– Byte 18: 0x74 - No-Stretch Write Unprotect command
– Byte 19: 0x83 - No-Stretch Readout Protect command
– Byte 20: 0x93 - No-Stretch Readout Unprotect command
– Byte 21: 0xA1 - No-Stretch Get Memory Checksum command
– Byte 22: ACK

2.2 Команда Get Version

Команда Get Version используется для получения версии загрузчика I2C. Когда загрузчик получает команду, он передает хосту описанную ниже информацию (версия загрузчика).

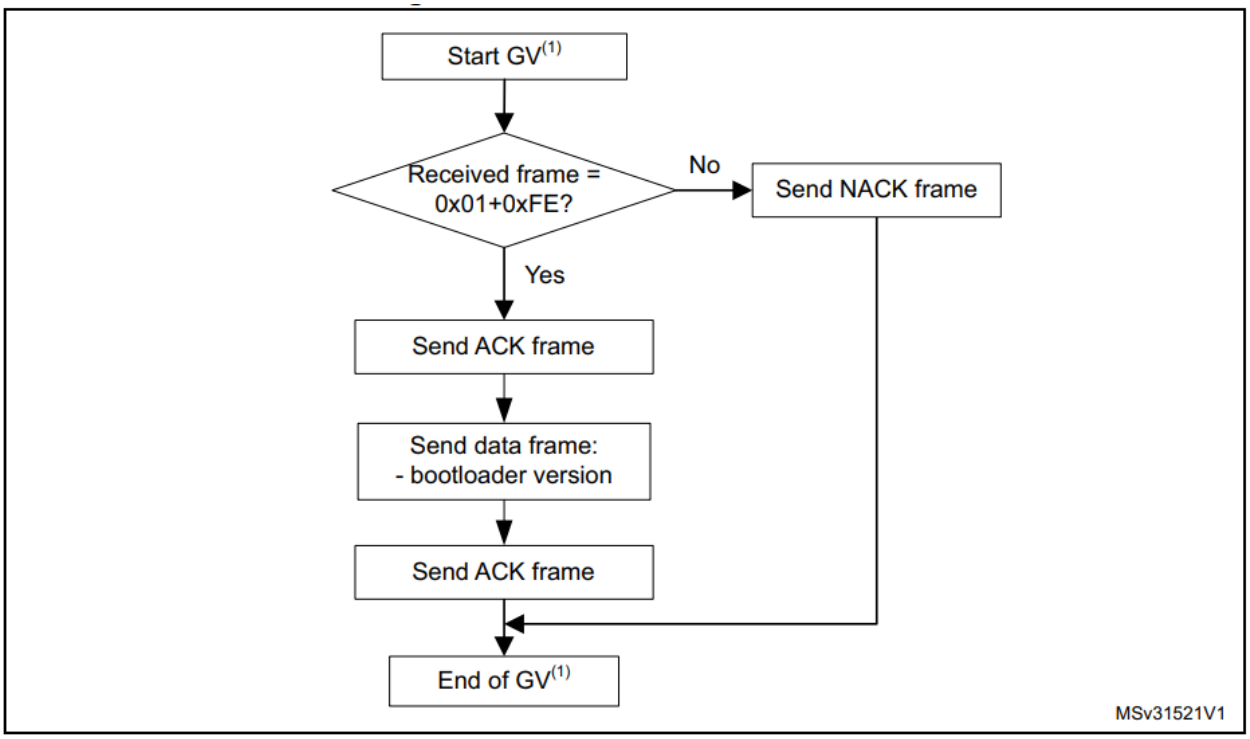
Рисунок 4. Команда Get Version: сторона Хоста



1. GV = Get Version.

STM32 отправляет байты следующим образом:
- Byte 1: ACK
- Byte 2: Bootloader version (0 < Version <= 255) (for example, 0x10 = Version 1.0)
- Byte 3: ACK

Рисунок 5. Команда Get Version: сторона устройства



1. GV = Get Version

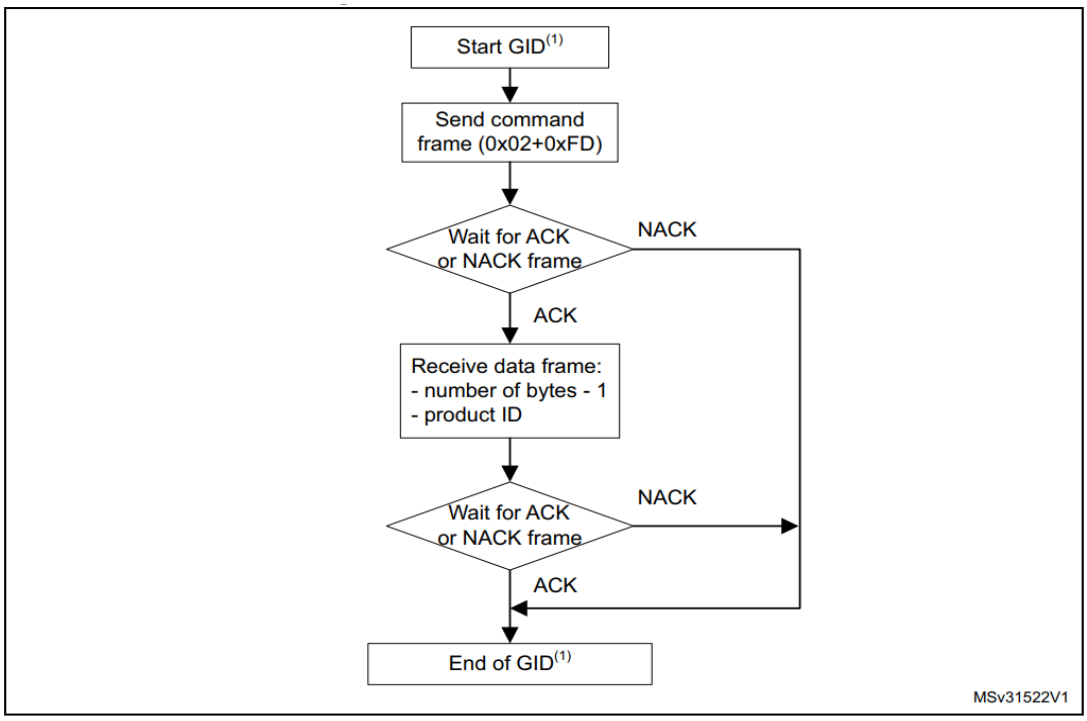
2.3 Команда Get ID

Команда Get ID используется для получения версии идентификатора чипа. Когда загрузчик получает эту команду, он передает идентификатор продукта хосту.

Устройство STM32 отправляет байты следующим образом:

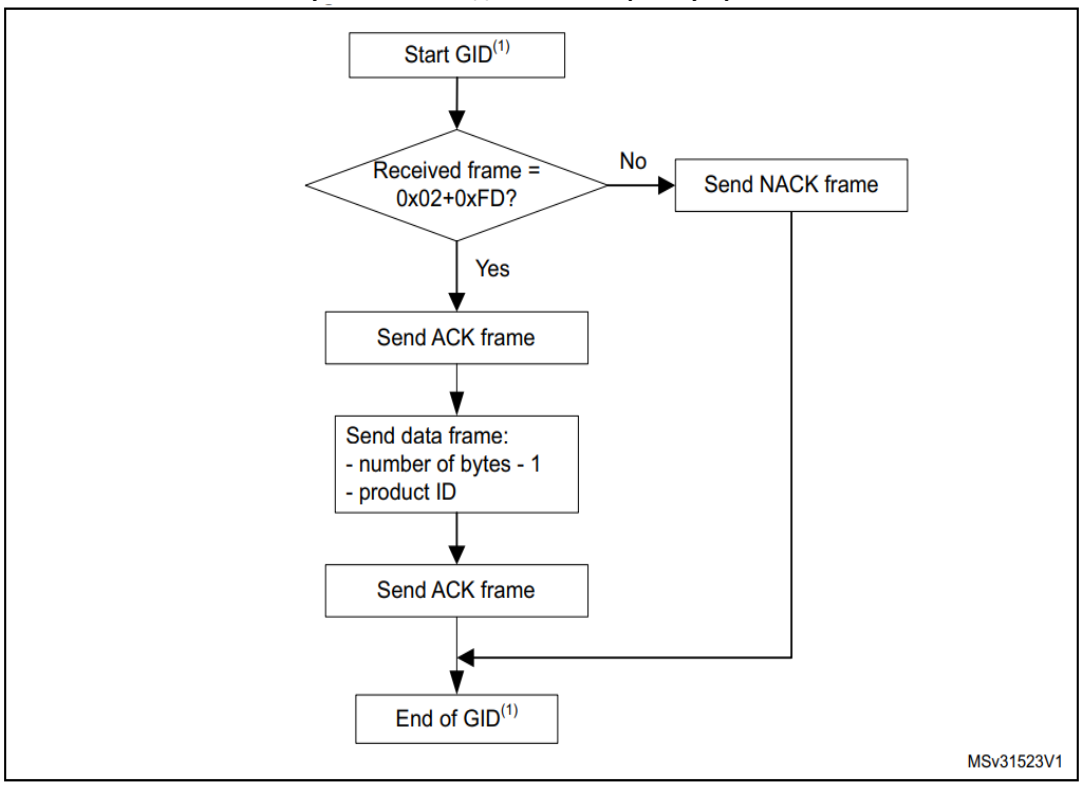
- Byte 1: ACK
- Byte 2: N = the number of bytes-1 (for STM32, N = 1), except for current byte and ACKs
- Bytes 3-4: PID (product ID)
  - Byte 3 = MSB
  - Byte 4 = LSB
- Byte 5: ACK

Рисунок 6. Команда Get ID: сторона хоста



1. GID = Get ID.

Рисунок 7. Команда Get ID: сторона устройства



1. GID = Get ID.

2.4 Команда Read Memory

Команда Read Memory используется для чтения данных с любого допустимого адреса памяти.

Когда загрузчик получает команду Read Memory, он передает приложению байт ACK. Затем загрузчик ожидает 4-байтовый адрес (1-й байт - MSB, 4-й байт - LSB) и байт контрольной суммы, после чего проверяет полученный адрес. Если адрес действителен и контрольная сумма верна, загрузчик передает байт ACK, в противном случае он передает байт NACK и прерывает команду.

Если адрес действителен и контрольная сумма верна, загрузчик ожидает количества передаваемых байтов (N байтов) и его дополненного байта (контрольная сумма). Если контрольная сумма верна, загрузчик передает необходимые данные приложению, начиная с полученного адреса. Если контрольная сумма неверна, он отправляет NACK перед прерыванием команды.

Хост отправляет байты на STM32 следующим образом:

- Byte 1: 0x11
- Byte 2: 0xEE

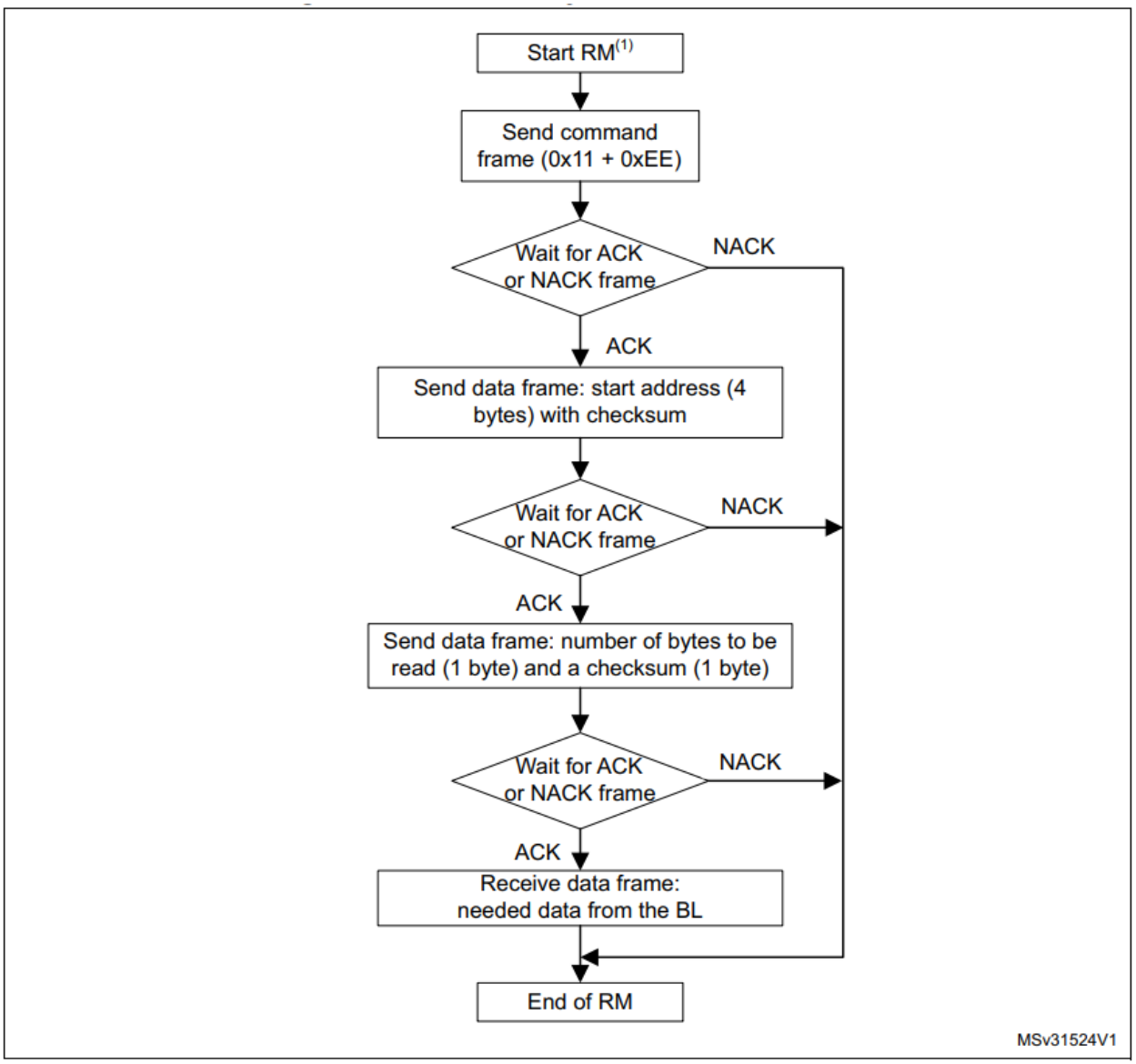
Wait for ACK

- Bytes 3-6: Start address (byte 3: MSB, byte 6: LSB)
- Byte 7: Checksum: XOR (byte 3, byte 4, byte 5, byte 6)

Wait for ACK

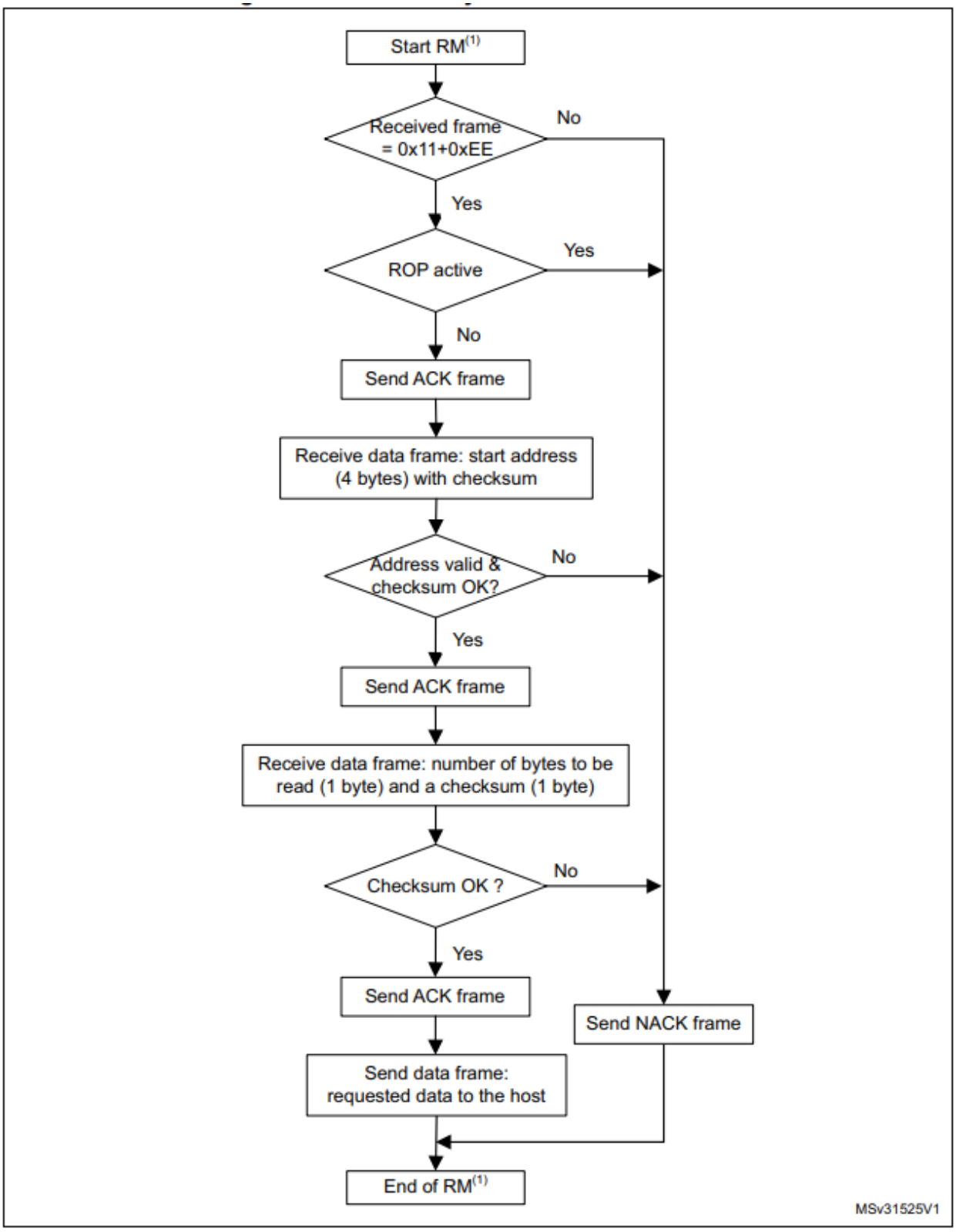
- Byte 8: The number of bytes to be read - 1 (0 < N <= 255)
- Byte 9: Checksum: XOR byte 8 (complement of byte 8)

Рисунок 8. Команда Read Memory: сторона хоста



1. RM = Read Memory.

Рисунок 9. Команда Read Memory: сторона устройства



1. RM = Read Memory.

2.5 Команда Go

Команда Go используется для выполнения загруженного кода или любого другого кода путем ветвления по адресу, указанному приложением.

Когда загрузчик получает команду Go, он передает приложению байт ACK. Затем загрузчик ожидает 4x-байтовый адрес (1-й байт - MSB, 4-й байт - LSB) и байт контрольной суммы, после чего проверяет полученный адрес. Если адрес действителен и контрольная сумма верна, загрузчик передает байт ACK, в противном случае он передает байт NACK и прерывает команду.

Когда адрес действителен и контрольная сумма верна, загрузчик выполняет следующие операции:

1. Инициализирует регистры периферийных устройств, используемых загрузчиком, до значений сброса по умолчанию.
2. Инициализирует указатель основного стека пользовательского приложения.
3. Переходит к ячейке памяти, запрограммированной в полученном 'адресе + 4' (соответствует адресу обработчика сброса приложения). Например, если получен адрес 0x08000000, загрузчик переходит к ячейке памяти, запрограммированной по адресу 0x08000004.

Как правило, хост отправляет адрес для перехода на запрограммированное приложение.

Примечание: Переход к приложению работает только в том случае, если пользовательское приложение правильно устанавливает таблицу векторов так, чтобы она указывала на адрес приложения.

Хост отправляет байты на STM32 следующим образом:
- Byte 1: 0x21
- Byte 2: 0xDE

Wait for ACK
- Byte 3 to byte 6: start address
- Byte 3: MSB
- Byte 6: LSB
- Byte 7: checksum: XOR (byte 3, byte 4, byte 5, byte 6)

Wait for ACK

Рисунок 10. Команда Go: сторона хоста

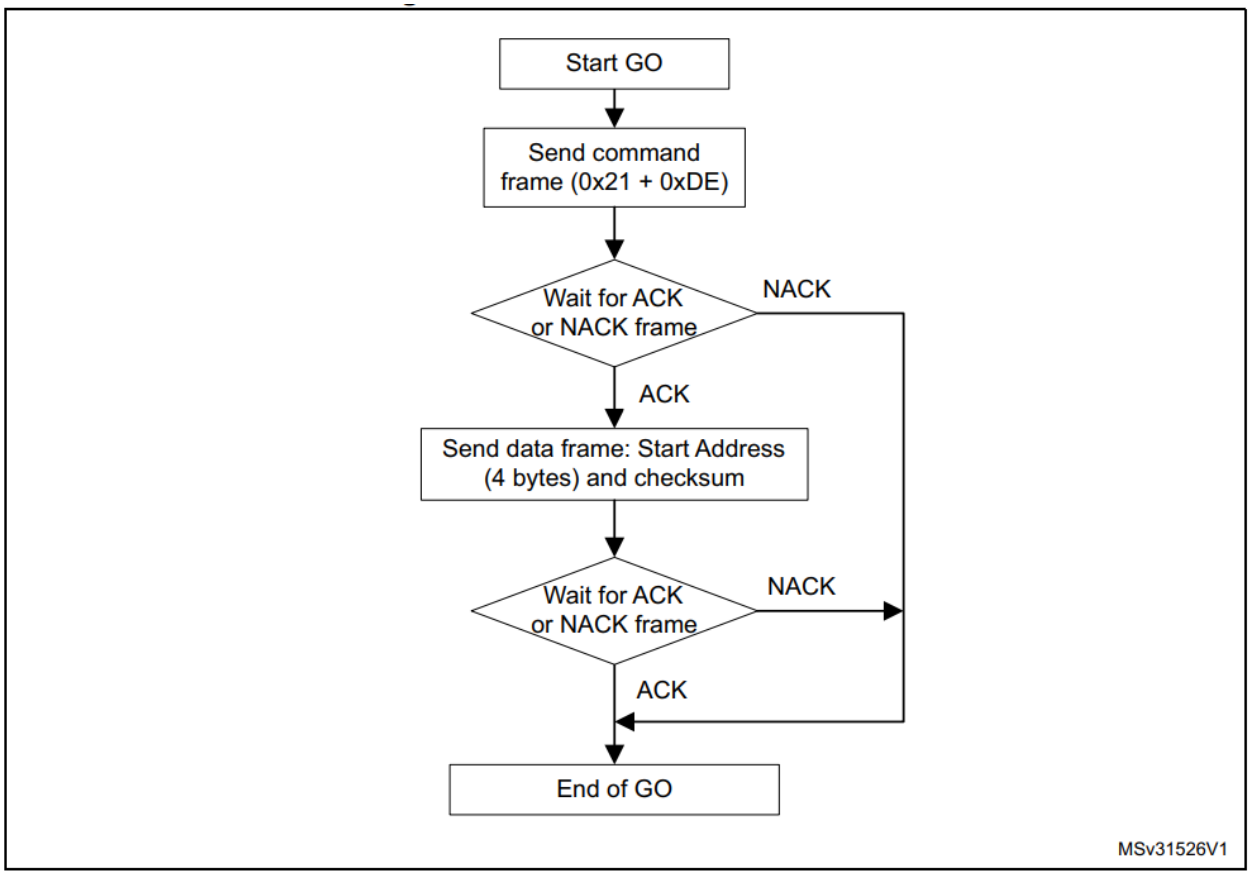
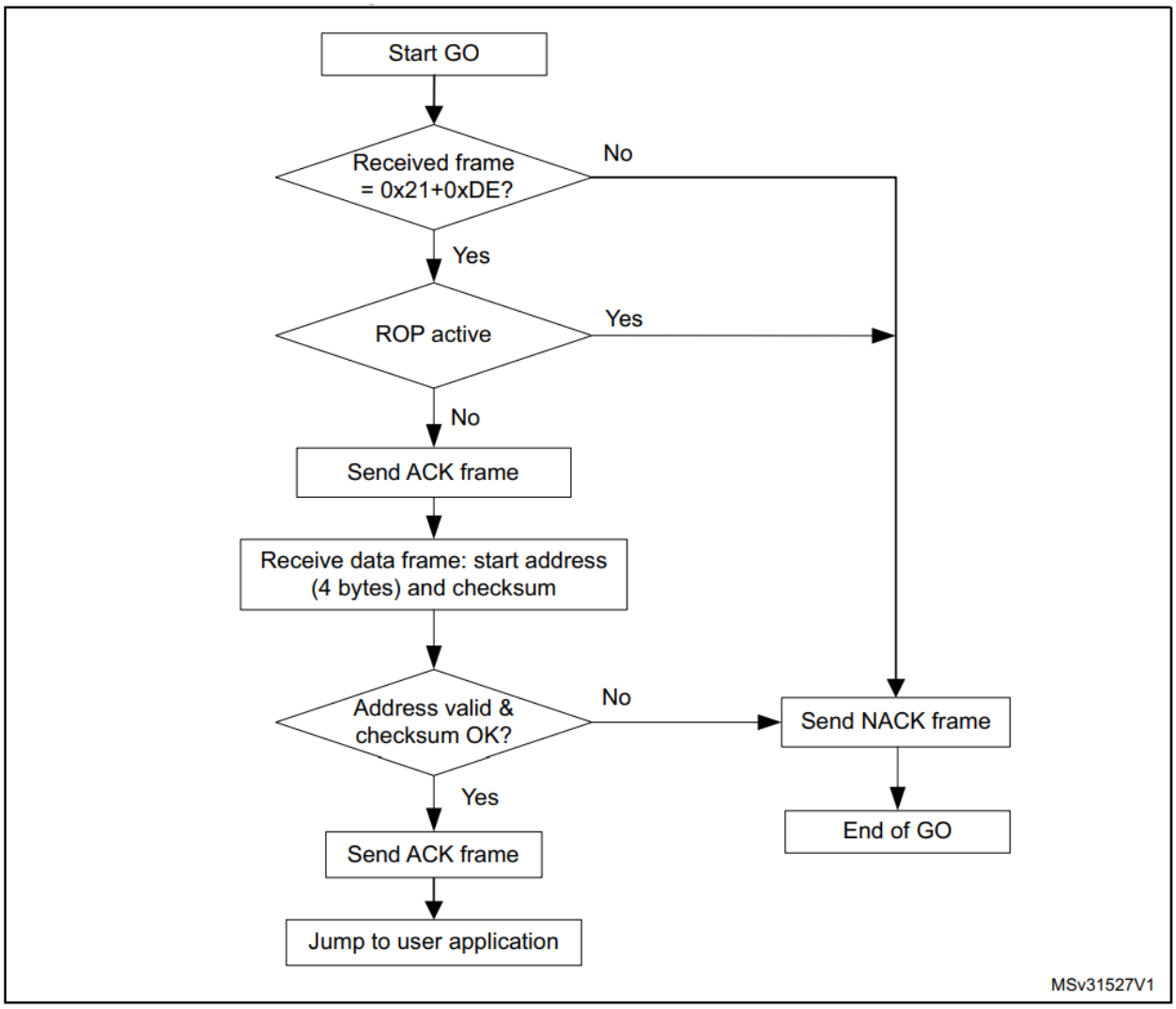


Рисунок 11. Команда Go: сторона устройства



2.6 Команда Write Memory

Команда Write Memory используется для записи данных в любой допустимый адрес памяти (см. Примечание: ниже) ОЗУ, флэш-памяти или в область байтов управления (the option byte area).

Когда загрузчик получает команду Write Memory, он передает приложению байт ACK. Затем загрузчик ожидает 4х-байтовый адрес (1-й байт - MSB, 4-й байт - LSB) и байт контрольной суммы, а затем проверяет полученный адрес.

Если полученный адрес валидный(доступный) и контрольная сумма верна, загрузчик передает байт ACK, в противном случае он передает байт NACK и прерывает команду.

Когда адрес валиден(доступен) и контрольная сумма верна, загрузчик:

1. получает байт N, который содержит количество байтов данных, которые необходимо получить
2. получает пользовательские данные (N + 1) байт и контрольную сумму (исключающее ИЛИ со всеми байтами, начиная с байта N)
3. заносит данные пользователя в память, начиная с полученного 4х-байтового адреса.

В конце команды, если операция записи прошла успешно, загрузчик передает байт ACK, в противном случае он передает приложению байт NACK и прерывает выполнение команды.

Если команда Write Memory производит запись в область байтов управления (the option byte area), все опции стираются перед записью новых значений. В конце команды загрузчик генерирует сброс системы, чтобы применилась записанная конфигурация.

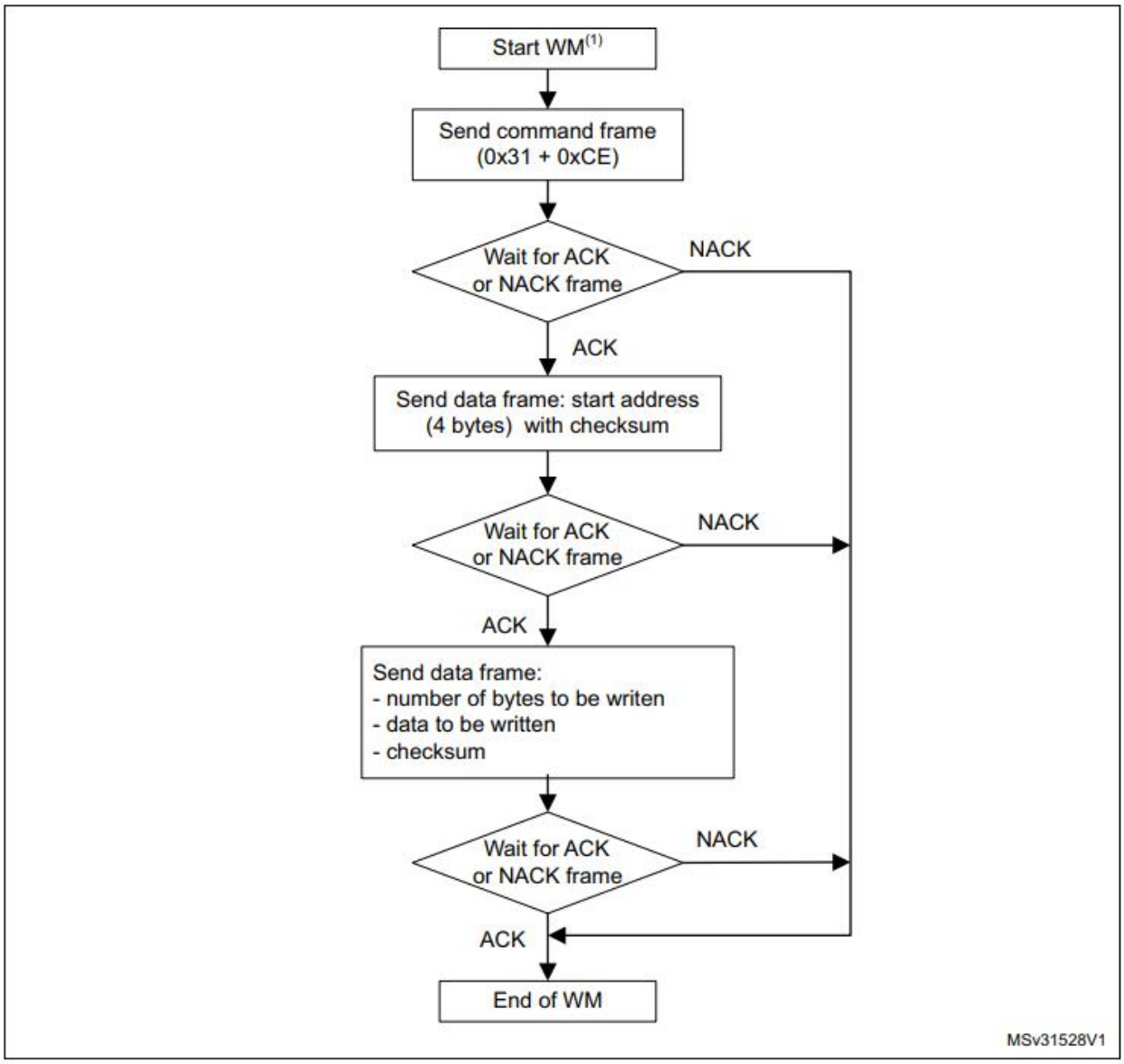
Максимальная длина блока, записываемого в область байтов управления, зависит от продукта STM32, и адрес, полученный от хоста, должен быть начальным адресом области байтов управления. Дополнительные сведения о область байтов управления см. в справочном руководстве по продукту STM32.

Примечание: Максимальная длина блока для записи в ОЗУ или флэш-память составляет 256 байт. При записи в ОЗУ следите за тем, чтобы не перекрывать первое ОЗУ, используемое прошивкой загрузчика.

При выполнении операций записи в защищенные от записи сектора ошибка не возвращается.

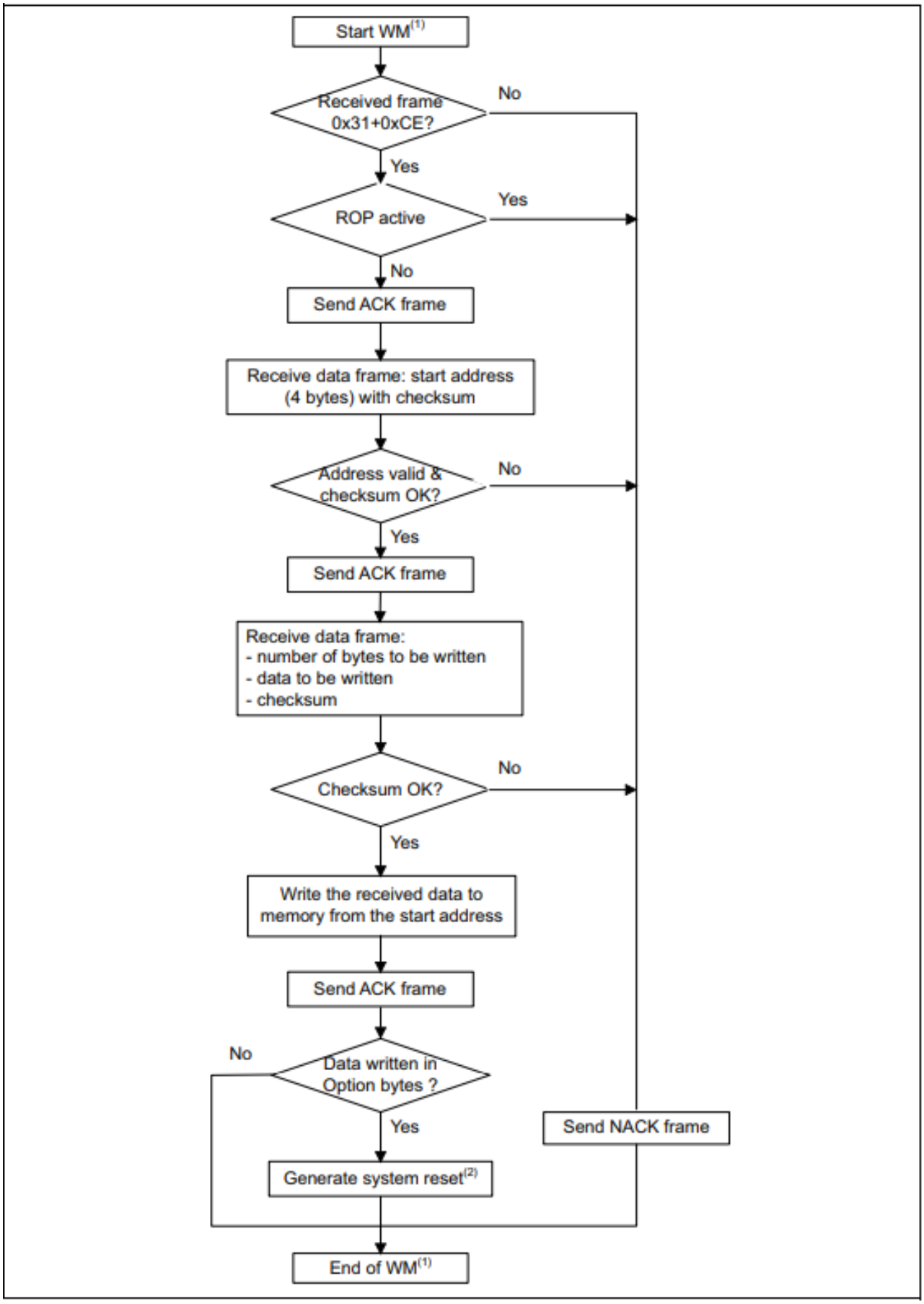
Хост отправляет байты на STM32 следующим образом:
- Byte 1: 0x31
- Byte 2: 0xCE
Wait for ACK
- Byte 3 to byte 6: Start address (Byte 3: MSB, Byte 6: LSB)
- Byte 7: Checksum: XOR (byte 3, byte 4, byte 5, byte 6)
Wait for ACK
- Byte 8: Number of bytes to be received (0 < N <= 255)
- N+1 data bytes: (max 256 bytes)
- Checksum byte: XOR (N, N+1 data bytes)
Wait for ACK

Рисунок 12. Команда Write Memory: сторона хоста



1. WM = Write Memory.

Рисунок 13. Команда Write Memory: сторона устройства



1. WM = Write Memory.
2. System reset is called only for some STM32 BL (STM32F0/F4/F7) and some STM32L4 (STM32L412xx/422xx, STM32L43xxx/44xxx, STM32L45xxx/46xxx) products. In all other STM32 products no system reset is called.

2.7 Команда Erase

Команда Erase позволяет хосту стирать страницы или сектора флэш-памяти, используя режим двухбайтовой адресации. Когда загрузчик получает команду Erase, он передает хосту байт ACK. Затем загрузчик получает два байта (количество страниц или секторов, которые необходимо стереть), коды страниц или секторов флэш-памяти (каждый закодирован двумя байтами, сначала старший байт) и байт контрольной суммы (исключающее ИЛИ отправленных байтов). Если контрольная сумма верна, загрузчик стирает память и отправляет хосту байт ACK, в противном случае он отправляет байт NACK хосту и команда прерывается.

Особенности команды Erase

Загрузчик получает одно полуслово (два байта), содержащее количество страниц или секторов, которые необходимо стереть, уменьшенное на 1. Если получено 0xFFFY (где Y от 0 до F), выполняется специальное стирание (0xFFFF для глобальной очистки, 0xFFFE и 0xFFFD, соответственно, для массового стирания банка 1 и банка 2).

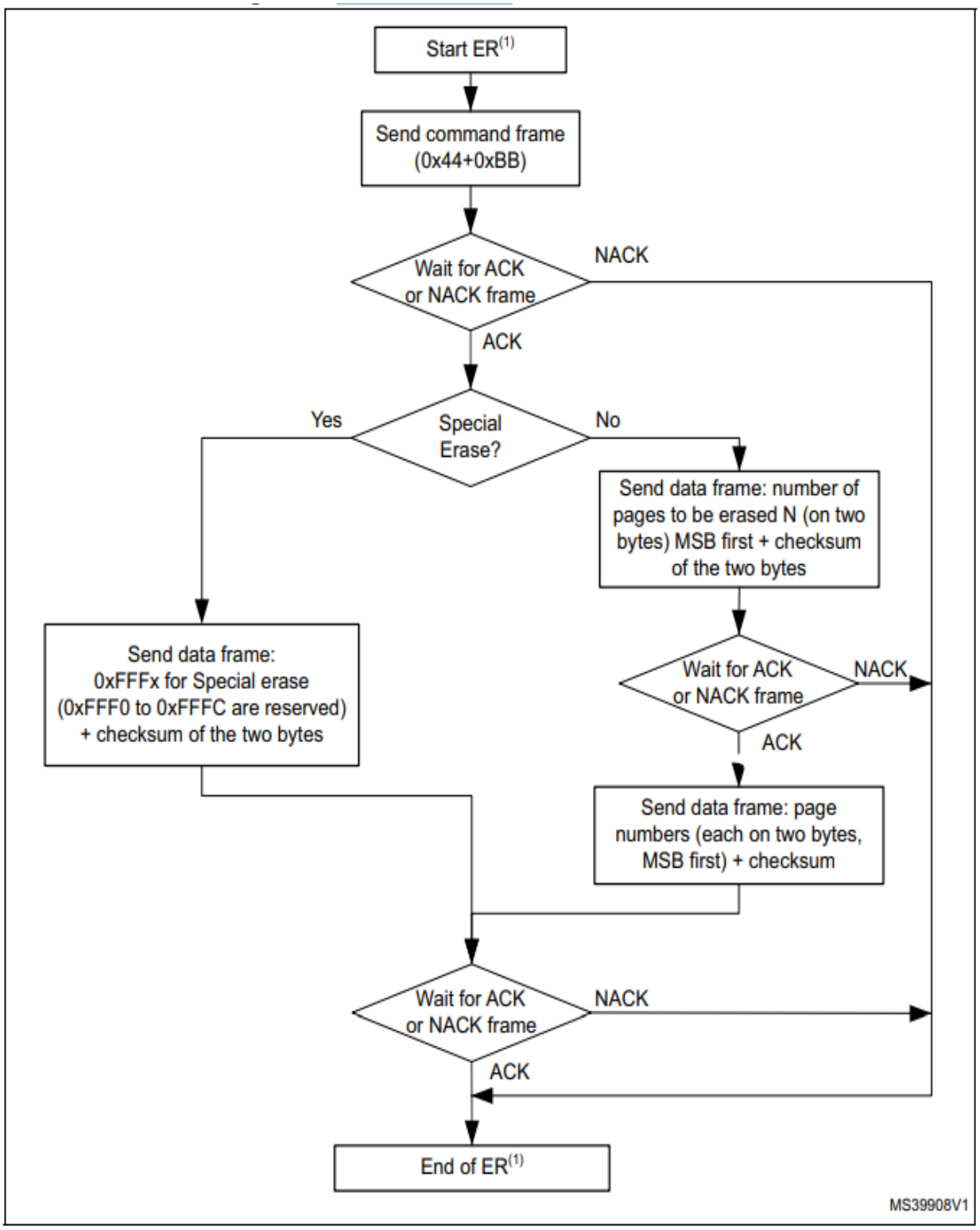
Загрузчик получает:
- в случае специального стирания один байт: контрольная сумма предыдущих байтов (например, 0x00 для 0xFFFF)
- в случае стирания N страниц или секторов загрузчик получает (2 x N) байтов, каждое полуслово которых содержит номер страницы или сектора, закодированный двумя байтами, причем старший бит идет первым. Затем все предыдущие байтовые контрольные суммы принимаются в одном байте.

Примечание: Некоторые продукты не поддерживают функцию массового стирания, в этом случае используйте команду стирания для стирания всех страниц или секторов.
Максимальное количество страниц или секторов связано с продуктом и должно соблюдаться.
Максимальное количество страниц или секторов, которые можно стереть одной командой, равно 512.
Коды от 0xFFFC до 0xFFF0 зарезервированы.
Ошибка не возвращается при выполнении операций стирания в защищенных от записи секторах.

Хост отправляет байты на STM32 следующим образом:
- Byte 1: 0x44
- Byte 2: 0xBB
Wait for ACK
For Special erase:
- Bytes 3-4: Special erase (0xFFFx)
- Bytes 5: Checksum of Bytes 3-4
Wait for ACK
For Page erase:
- Bytes 3-4: Number of pages or sectors to be erased - 1
- Bytes 5: Checksum of Bytes 3-4
Wait for ACK
- (2 x N) bytes (page numbers or sectors coded on two bytes MSB first) and then the checksum for these bytes.
Wait for ACK

Example of I2C frame:
– erase page 1:
0x44 0xBB **Wait ACK** 0x00 0x00 0x00 **Wait ACK** 0x00 0x01 0x01 **Wait ACK**
– erase page 1 and page 2:
0x44 0xBB **Wait ACK** 0x00 0x01 0x01 **Wait ACK** 0x00 0x01 0x00 0x02 0x03 **Wait ACK**

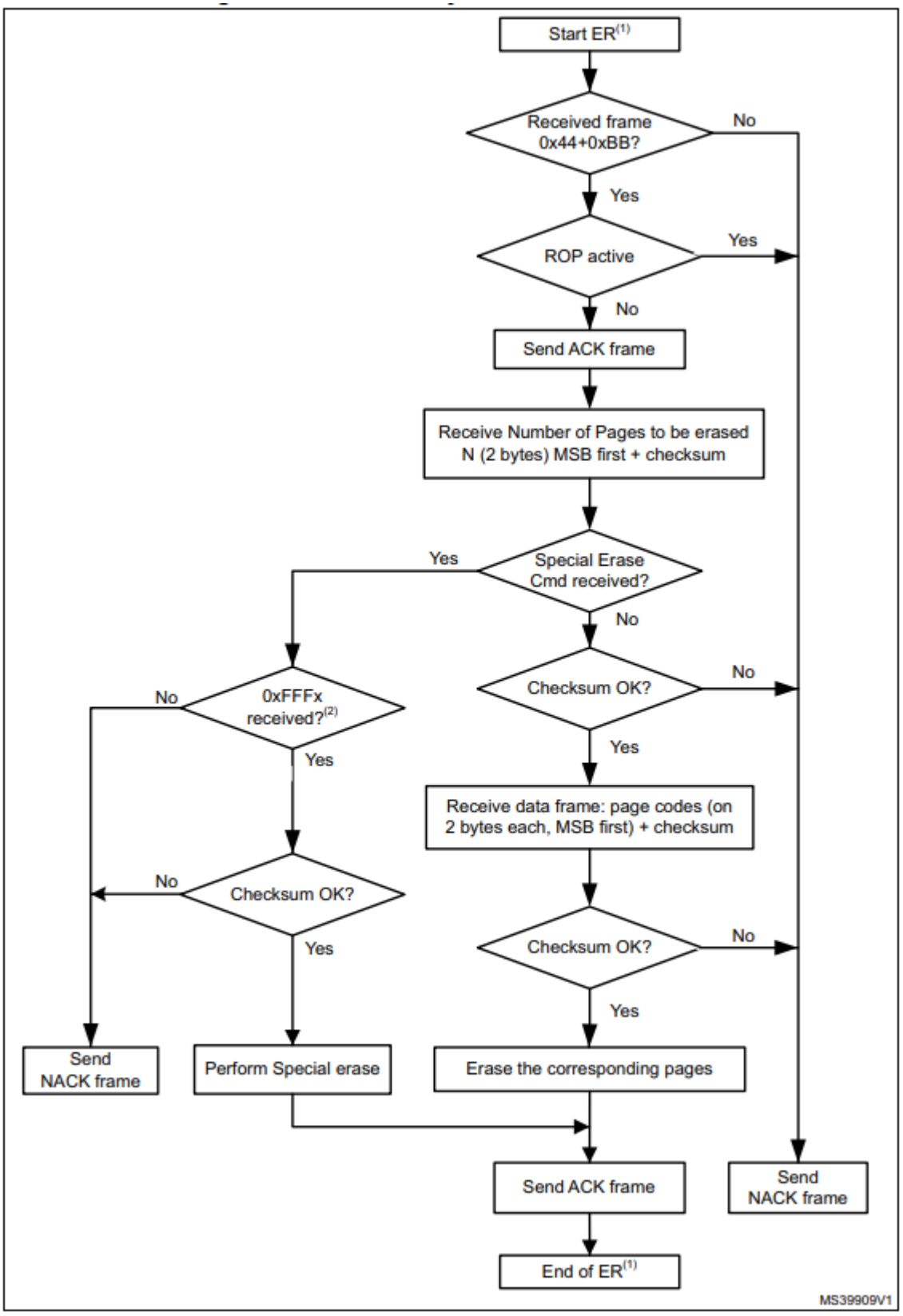
Рисунок 14. Команда Erase: сторона хоста



1. ER = Erase Memory.

Примечание: Некоторые продукты не поддерживают функцию специального стирания. Для этих продуктов эта команда возвращает NACK.

Рисунок 15. Команда Erase: сторона устройства



1. ER = Erase Memory.
2. Requested Special erase command is NACK-ed if not supported by the used STM32 product.

2.8 Команда Write Protect

Команда Write Protect используется для включения защиты от записи для некоторых или всех секторов флэш-памяти. Когда загрузчик получает команду Write Protect, он передает хосту байт ACK. Затем загрузчик ожидает получения определенного количества байтов (секторов, которые необходимо защитить), а затем получает от приложения коды секторов флэш-памяти.

В конце команды Write Protect загрузчик передает байт ACK и генерирует сброс системы, чтобы применились изменения, внесенные в байты конфигурации.

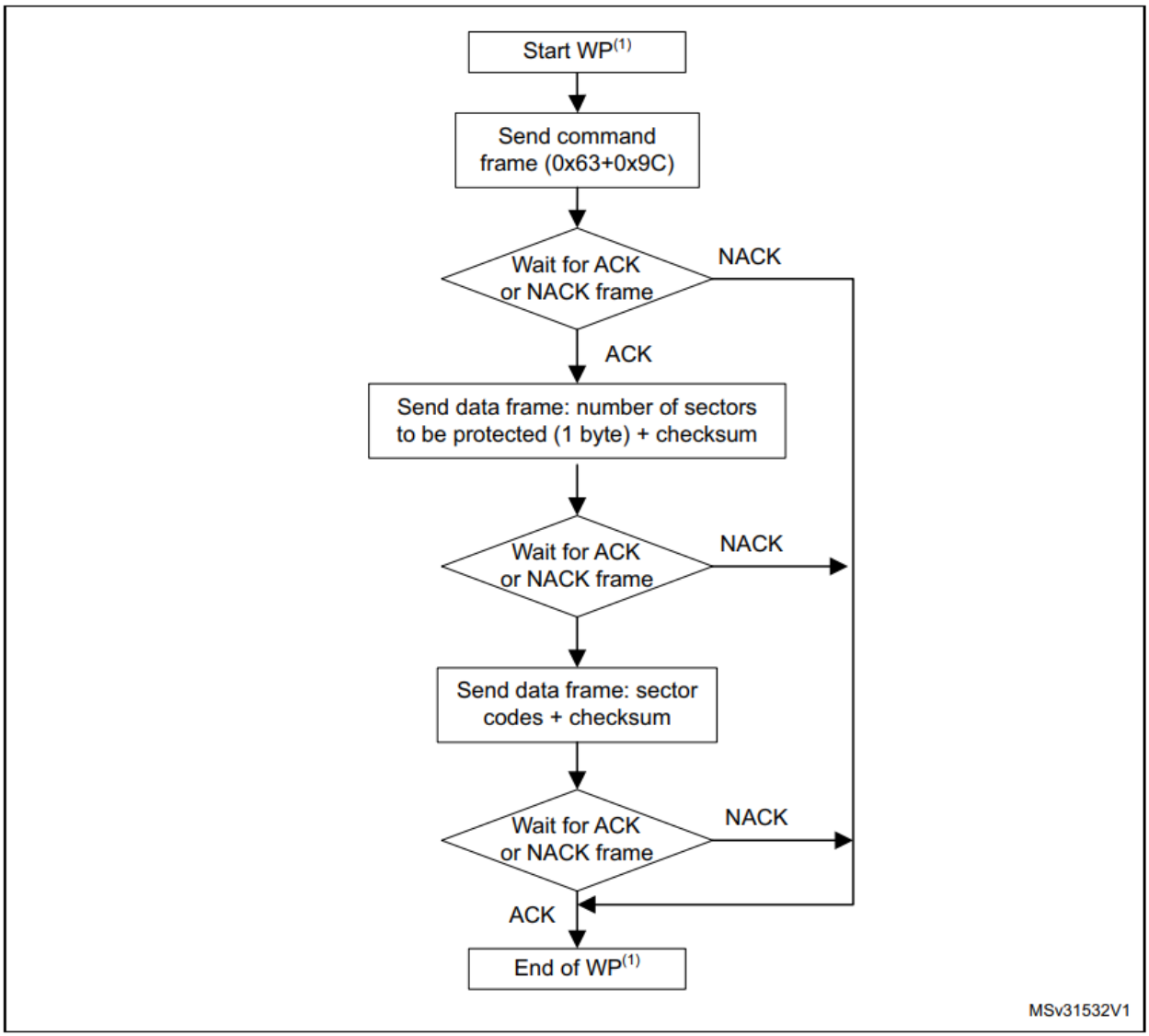
Последовательность команды Write Protect:

- Загрузчик получает один байт, содержащий N защищаемых от записи секторов – 1 (0 < N <= 255)
- Загрузчик получает (N+1) байт, каждый из которых содержит код сектора

Примечание: Ни общее количество секторов, ни номер защищаемого сектора не проверяются. Это означает, что при передаче команды с неправильным количеством защищаемых секторов или с неправильным номером сектора ошибка не возвращается.

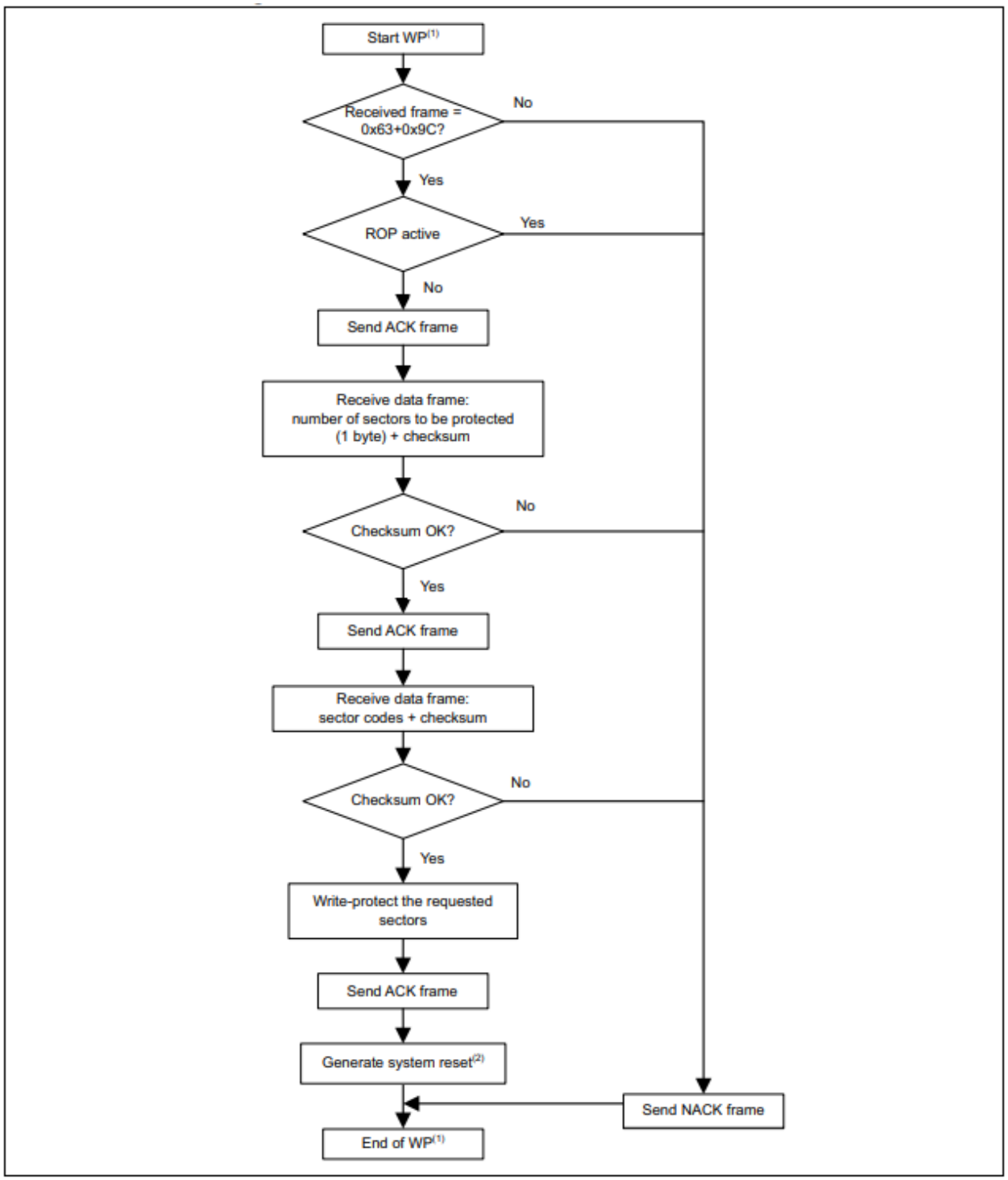
Если выполняется вторая команда Write Protect, секторы флэш-памяти, защищенные первой командой, становятся незащищенными, и защищаются только сектора, переданные второй командой защиты от записи.

Рисунок 16. Команда Write Protect: сторона хоста



1. WP = Write Protect.

Рисунок 17. Команда Write Protect: сторона устройства



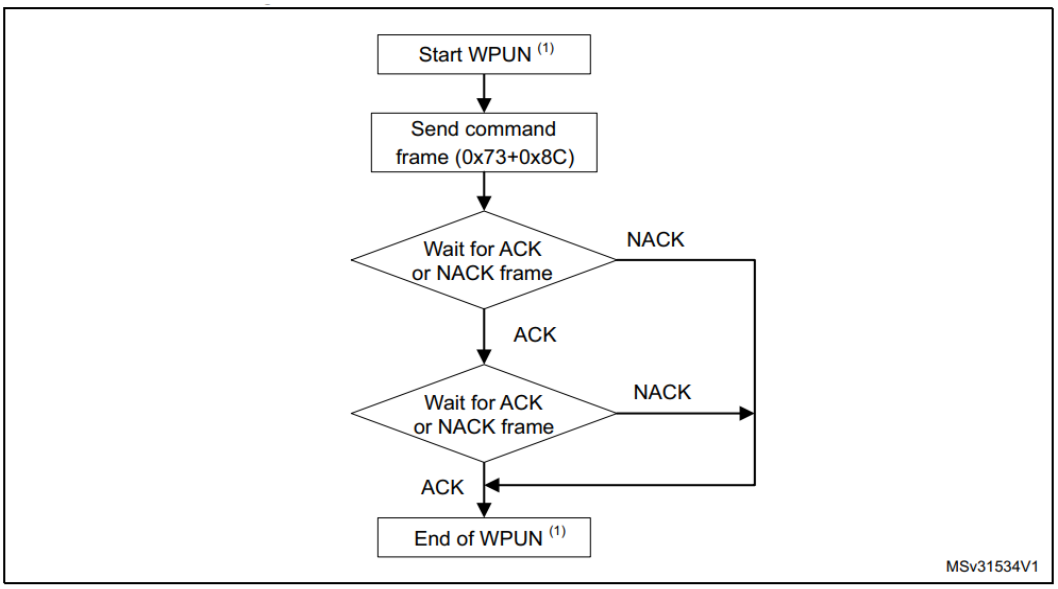
1. WP = Write Protect.
2. System reset is called only for some STM32 BL (STM32F0/F4/F7) and some STM32L4 (STM32L412xx/422xx, STM32L43xxx/44xxx, STM32L45xxx/46xxx) products. In all other STM32 products no system reset is called.

2.9 Команда Write Unprotect

Команда Write Unprotect используется для отключения защиты от записи всех секторов флэш-памяти. Когда загрузчик получает команду Write Unprotect, он передает хосту байт ACK. Затем загрузчик отключает защиту от записи всех секторов флэш-памяти и передает байт ACK.

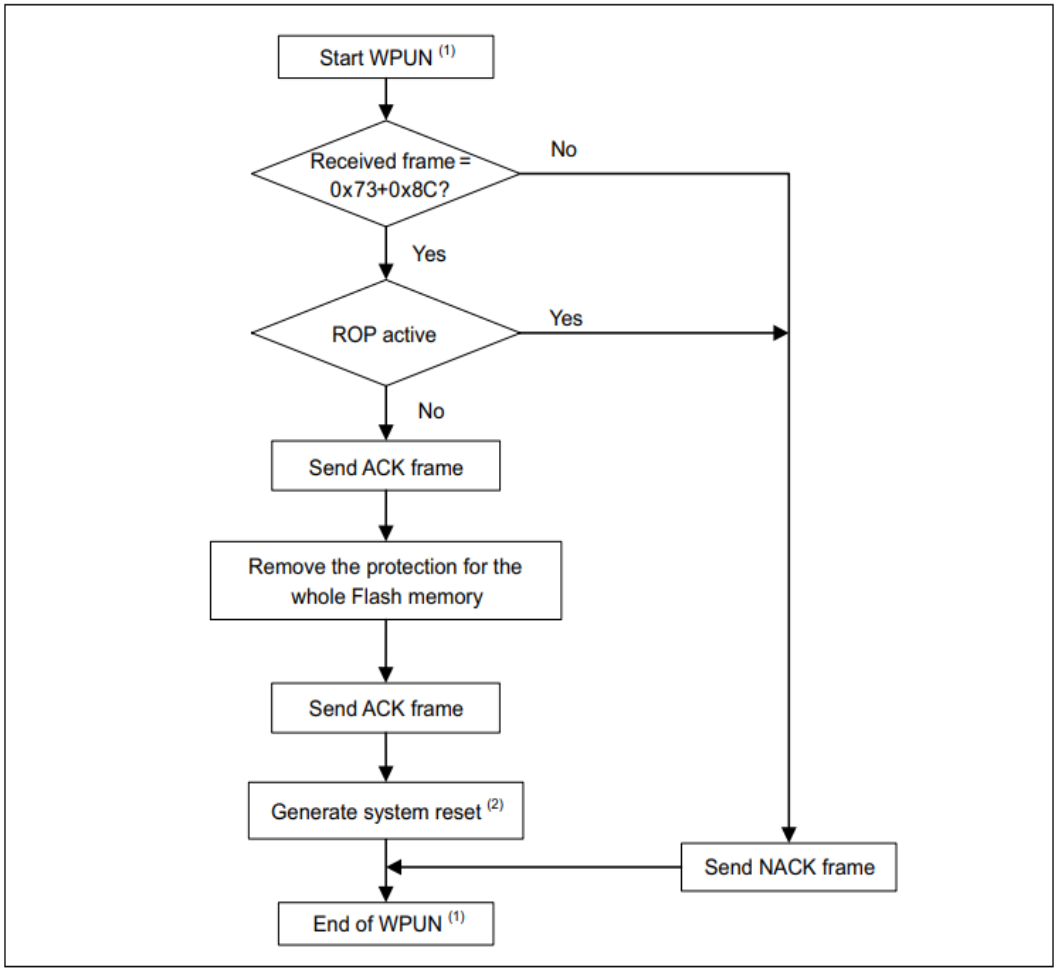
Системный сброс генерируется, чтобы применились изменения, внесенные в байты конфигурации.

Рисунок 18. Команда Write Unprotect: сторона хоста



1. WPUN = Write Unprotect.

Рисунок 19. Команда Write Unprotect: сторона устройства



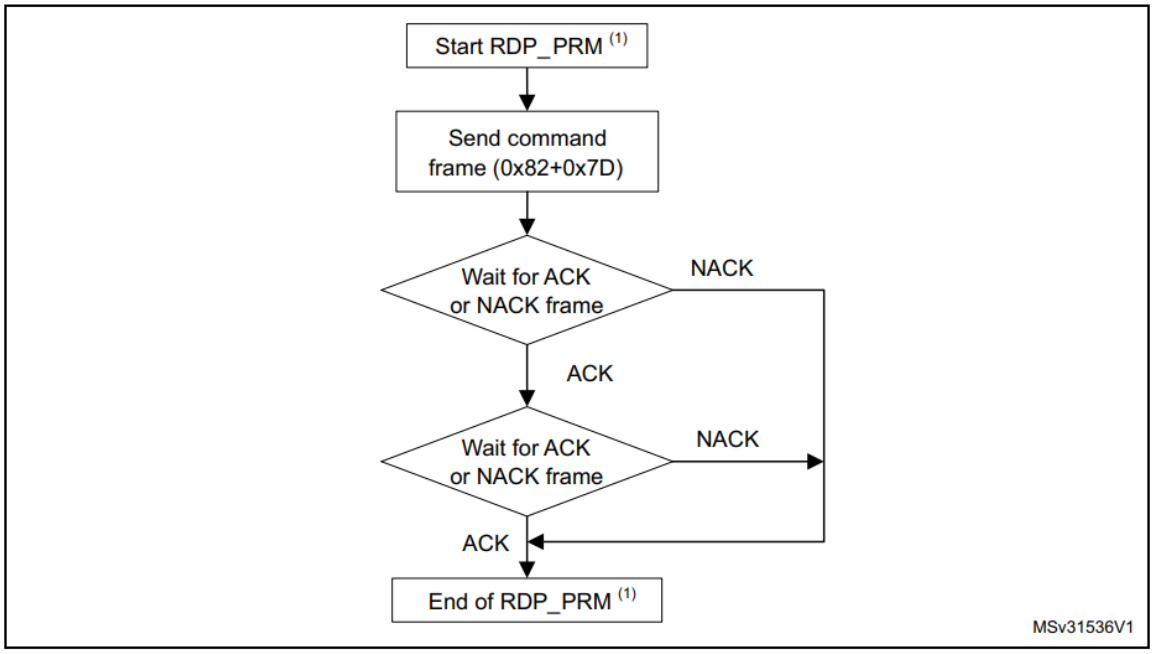
1. WPUN = Write Unprotect.
2. System reset is called only for some STM32 BL (STM32F0/F4/F7) and some STM32L4 (STM32L412xx/422xx, STM32L43xxx/44xxx, STM32L45xxx/46xxx) products. In all other STM32 products no system reset is called.

2.10 Команда Readout Protect

Команда Readout Protect используется для включения защиты флэш-памяти от чтения. Когда загрузчик получает команду Readout Protect, он передает хосту байт ACK и включает защиту от чтения для флэш-памяти.

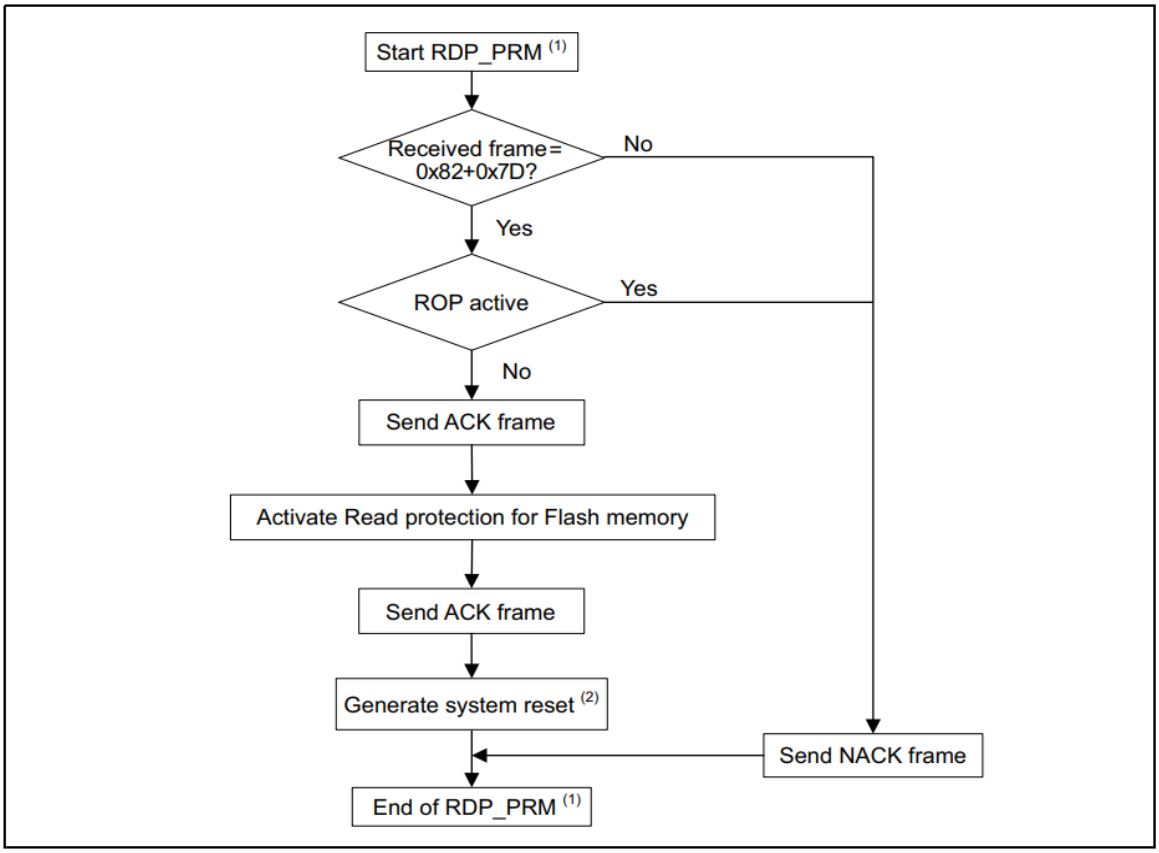
В конце команды Readout Protect загрузчик передает байт ACK и генерирует сброс системы, чтобы применились изменения, внесенные в байты конфигурации.

Рисунок 20. Команда Readout Protect: сторона хоста



1. RDP_PRM = Readout Protect.

Рисунок 21. Команда Readout Protect: сторона устройства



1. RDP_PRM = Readout Protect.
2. System reset is called only for some STM32 BL (STM32F0/F4/F7) and some STM32L4 (STM32L412xx/422xx, STM32L43xxx/44xxx, STM32L45xxx/46xxx) products. In all other STM32 products no system reset is called.

2.11 Команда Readout Unprotect

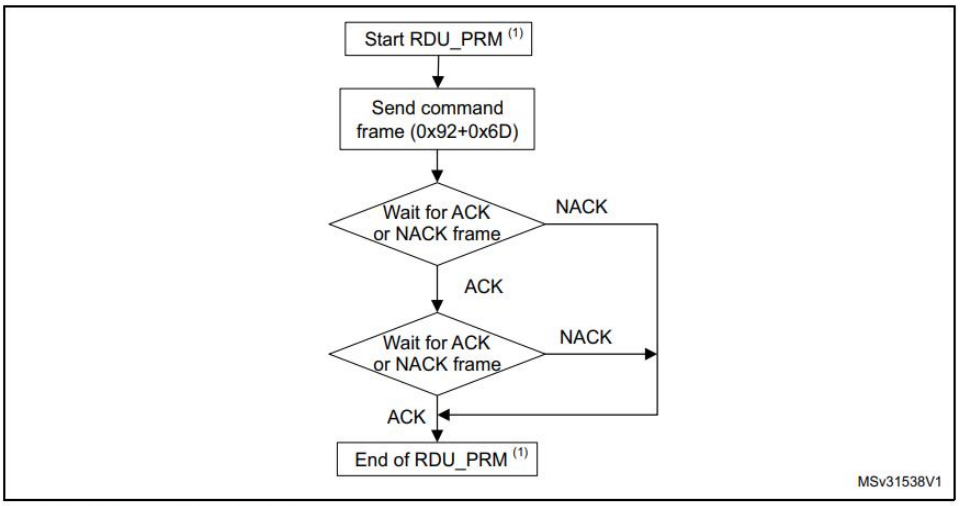
Команда Readout Unprotect используется для отключения защиты от чтения флэш-памяти. Когда загрузчик получает команду Readout Unprotect, он передает хосту байт ACK.

Затем загрузчик отключает защиту от чтения для всей флэш-памяти, что приводит к стиранию всей флэш-памяти. Если операция не удалась, загрузчик передает NACK и защита от чтения остается активной.

Примечание: Эта операция занимает одинаковое время для стирания всех страниц или секторов (или для выполнения массового стирания, если оно поддерживается продуктом), поэтому хост должен дождаться окончания операции.
Время стирания флэш-памяти см. в техническом описании продукта.

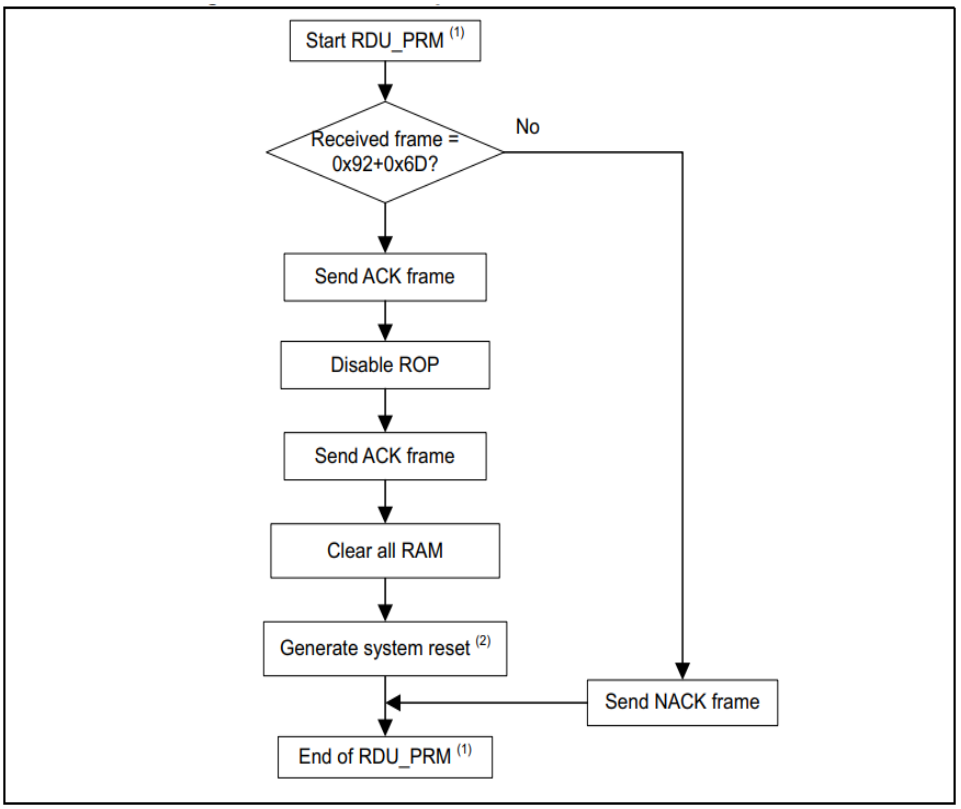
В конце команды Readout Unprotect загрузчик передает ACK и генерирует сброс системы, чтобы применились изменения, внесенные в байты конфигурации.

Рисунок 22. Команда Readout Unprotect: сторона хоста



1. RDU_PRM = Readout Unprotect.

Рисунок 23. Команда Readout Unprotect: сторона устройства



1. RDU_PRM = Readout Unprotect.
2. System reset is called only for some STM32 BL (STM32F0/F4/F7) and some STM32L4 (STM32L412xx/422xx, STM32L43xxx/44xxx, STM32L45xxx/46xxx) products. In all other STM32 products no system reset is called.

2.12 Команда No-Stretch Write Memory

Команда No-Stretch Write Memory используется для записи данных в любую допустимую область памяти.

Когда загрузчик получает команду No-Stretch Write Memory, он передает приложению байт ACK. Затем загрузчик ожидает 4-байтовый адрес (1-й байт — это MSB адреса, 4-й байт — LSB) и байт контрольной суммы, а затем проверяет полученный адрес.

Если полученный адрес действителен и контрольная сумма верна, загрузчик передает байт ACK, в противном случае он передает байт NACK и прерывает команду.

Когда адрес действителен и контрольная сумма верна, загрузчик:

1. Получает байт, содержащий количество байтов данных N, которые необходимо получить.
2. Получает пользовательские данные ((N + 1) байт) и контрольную сумму (исключающее ИЛИ с байтом N и всеми байтами данных)
3. Программирует пользовательские данные в память, начиная с полученного адреса
4. Возвращает состояние BUSY (0x76) во время выполнения операции.

В конце команды, если операция записи прошла успешно, загрузчик передает байт ACK, в противном случае он передает приложению байт NACK и прерывает выполнение команды.

Примечание: Если команда No-Stretch Write Memory используется для области байтов конфигурации (option byte area), то после записи загрузчик осуществляет сброс системы, чтобы применились изменения, внесенные в байты конфигурации.
Максимальная длина блока, который может быть записан в память, составляет 256 байт, за исключением байтов конфигурации (option bytes). Максимальная длина зависит от продукта STM32, а адрес, полученный от хоста, должен быть начальным адресом области байтов конфигурации (option byte area).
Дополнительные сведения см. в справочном руководстве по продукту STM32.
При выполнении операций записи в защищенные от записи сектора ошибка не возвращается.

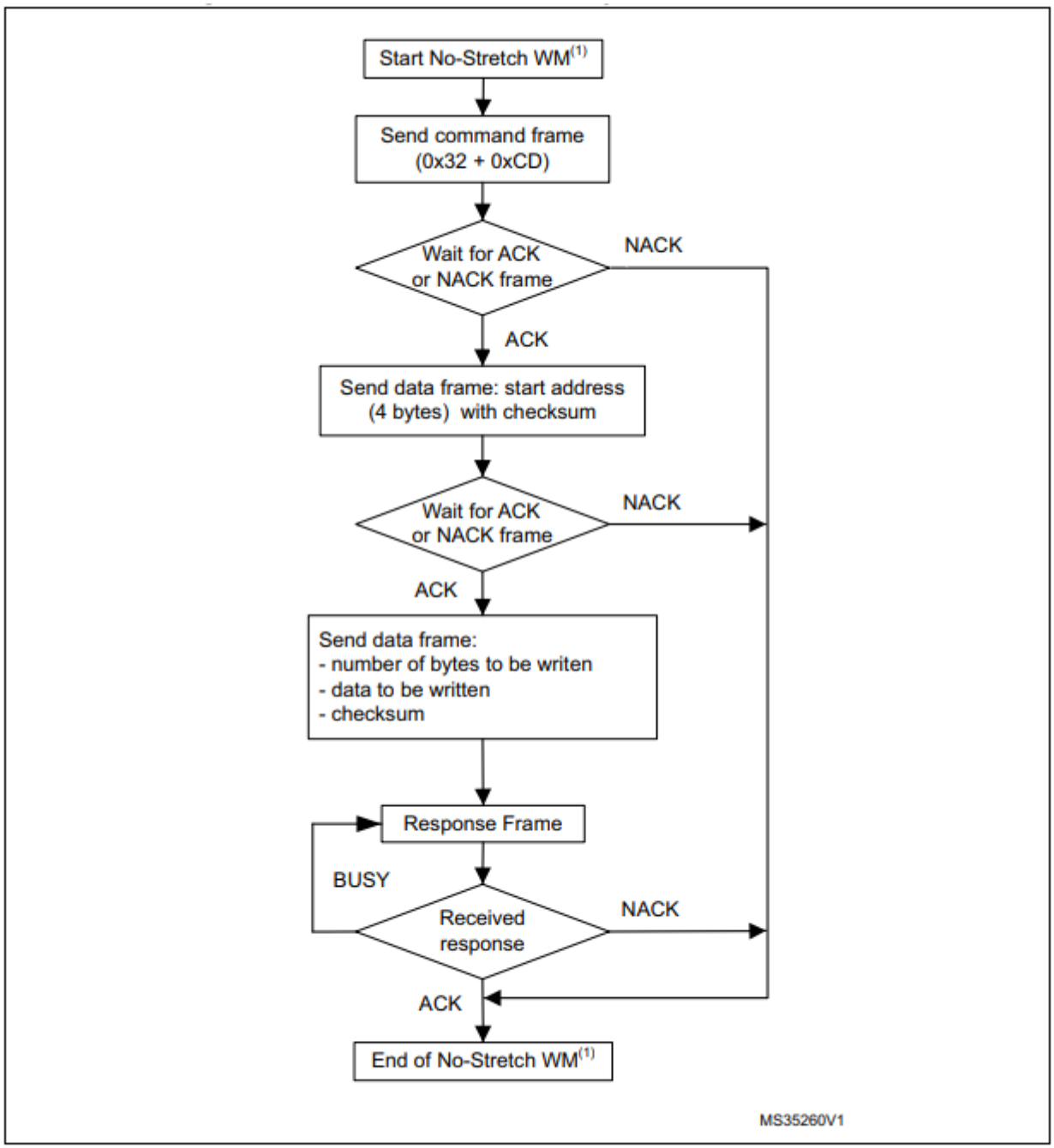
Хост отправляет байты на STM32 следующим образом:
- Byte 1: 0x32
- Byte 2: 0xCD

Wait for ACK
- Byte 3 to byte 6: Start address
  Byte 3: MSB
  Byte 6: LSB
- Byte 7: Checksum: XOR (byte 3, byte 4, byte 5, byte 6)

Wait for ACK
- Byte 8: Number of bytes to be received (0 < N □ 255)
- N +1 data bytes: (Max 256 bytes)
- Checksum byte: XOR (N, N+1 data bytes)

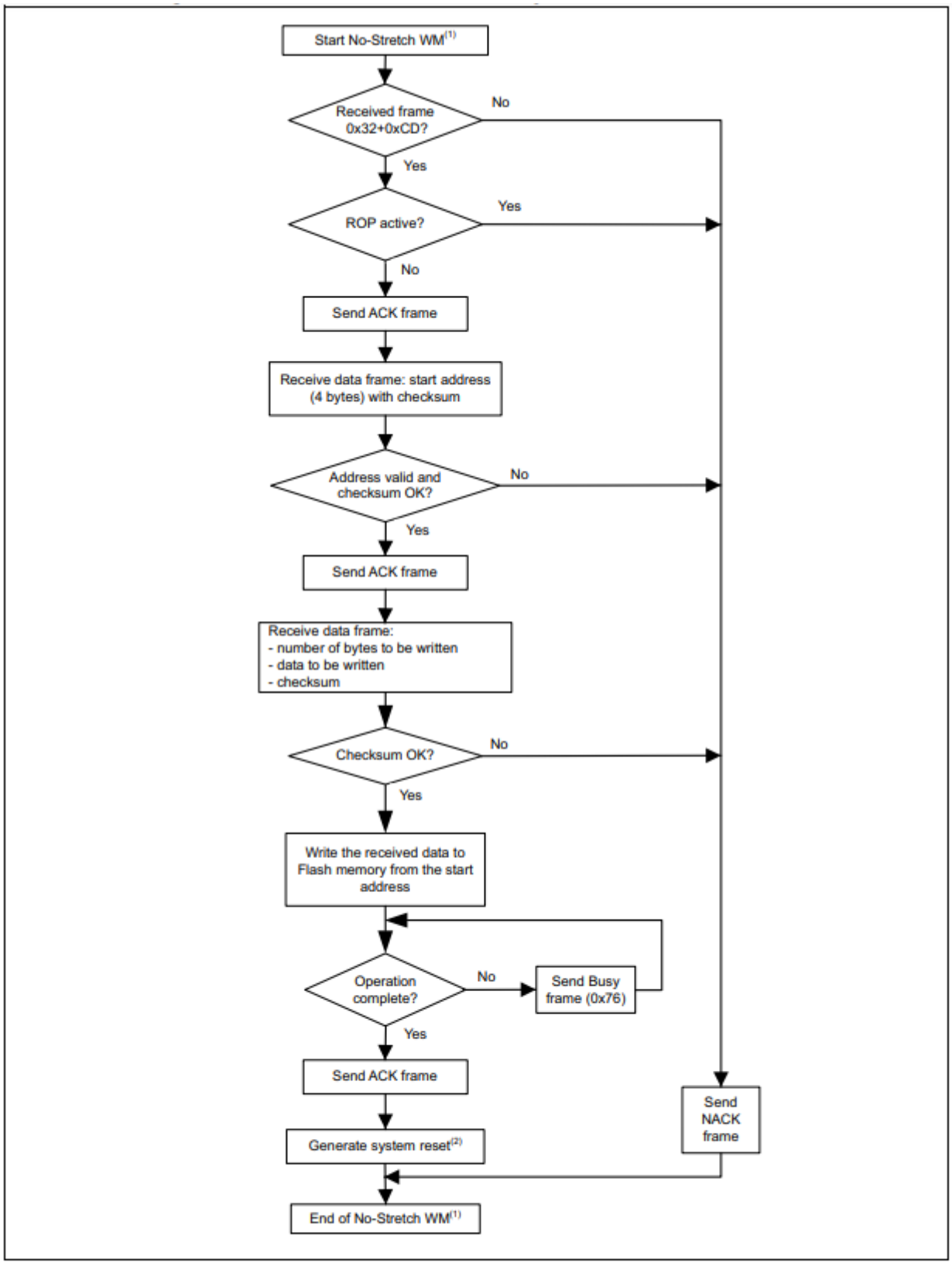
Wait for ACK (if Busy keep polling on ACK/NACK)

Рисунок 24. Команда No-Stretch Write Memory: сторона хоста



1. WM = Write Memory.

Рисунок 25. Команда No-Stretch Write Memory: сторона устройства



1. WM = Write Memory.
2. System reset is called only for some STM32 BL (STM32F4/F7) and some STM32L4 (STM32L412xx/422xx, STM32L43xxx/44xxx, STM32L45xxx/46xxx) products. In all other STM32 products no system reset is called.

2.13 Команда No-Stretch Erase Memory

Команда No-Stretch Erase Memory позволяет хосту стирать страницы или сектора флэш-памяти, используя режим двухбайтовой адресации.

Когда загрузчик получает команду No-Stretch Erase Memory, он передает хосту байт ACK. Затем загрузчик получает два байта (количество страниц или секторов, которые необходимо стереть), коды страниц или секторов флэш-памяти (каждый закодирован двумя байтами, сначала старший байт) и байт контрольной суммы (исключающее ИЛИ отправленных байтов). Если контрольная сумма верна, загрузчик стирает память (возвращает состояние BUSY(0x76) во время выполнения операции), а затем отправляет хосту байт ACK, в противном случае он отправляет байт NACK хосту и команда прерывается.

Особенности команды No-Stretch Erase Memory

Загрузчик получает одно полуслово (два байта), содержащее количество страниц или секторов, которые необходимо стереть, уменьшенное на 1. Если получено 0xFFFY (где Y от 0 до F), выполняется специальное стирание (0xFFFF для глобальной очистки, 0xFFFE и 0xFFFD соответственно для массового стирания банка 1 и банка 2).

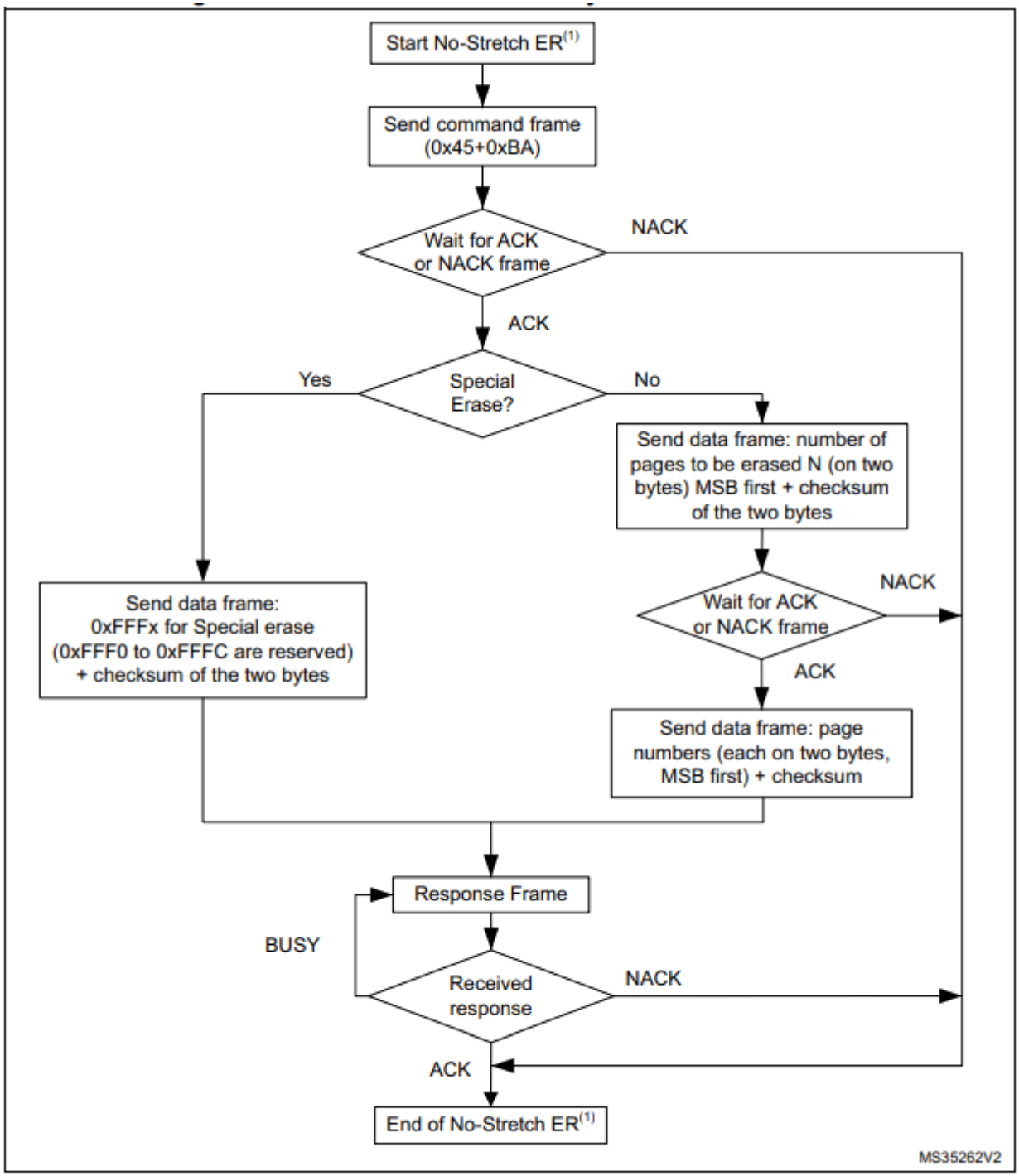
Загрузчик получает:
- В случае специального стирания один байт: контрольная сумма предыдущих байтов (например, 0x00 для 0xFFFF)
- В случае стирания N страниц или секторов загрузчик получает (2 x N) байтов, каждое полуслово которых содержит номер страницы или сектора, закодированный двумя байтами, причем старший бит идет первым. Затем все предыдущие байтовые контрольные суммы принимаются в одном байте.

Примечание: Некоторые продукты не поддерживают функцию массового стирания, в этом случае вместо этого используйте команду стирания, чтобы стереть все страницы или сектора.
Максимальное количество страниц или секторов зависит от продукта и должно соблюдаться.
Максимальное количество страниц или секторов, которые можно стереть одной командой, равно 512.
Коды от 0xFFFC до 0xFFF0 зарезервированы.
При выполнении операций стирания в защищенных от записи секторах ошибка не возвращается.

Хост отправляет байты на STM32 следующим образом:
- Byte 1: 0x45
- Byte 2: 0xBA
Wait for ACK
For Special erase:
- Bytes 3-4: Special erase (0xFFFx)
- Bytes 5: Checksum of bytes 3-4
Wait for ACK (if Busy keep polling on ACK/NACK)
For Page erase:
- Bytes 3-4: Number of pages or sectors to be erased - 1
- Bytes 5: Checksum of bytes 3-4
Wait for ACK
- (2 x N) bytes (page numbers or sectors coded on two bytes MSB first) and then the checksum for these bytes.
- Wait for ACK (if Busy keep polling on ACK/NACK)

Example of I2C frame:
- erase page 1:
0x45 0xBA **Wait ACK** 0x00 0x00 0x00 **Wait ACK** 0x00 0x01 0x01 **Wait ACK**
- erase page 1 and page 2:
0x45 0xBA **Wait ACK** 0x00 0x01 0x01 **Wait ACK** 0x00 0x01 0x00 0x02 0x03 **Wait ACK**

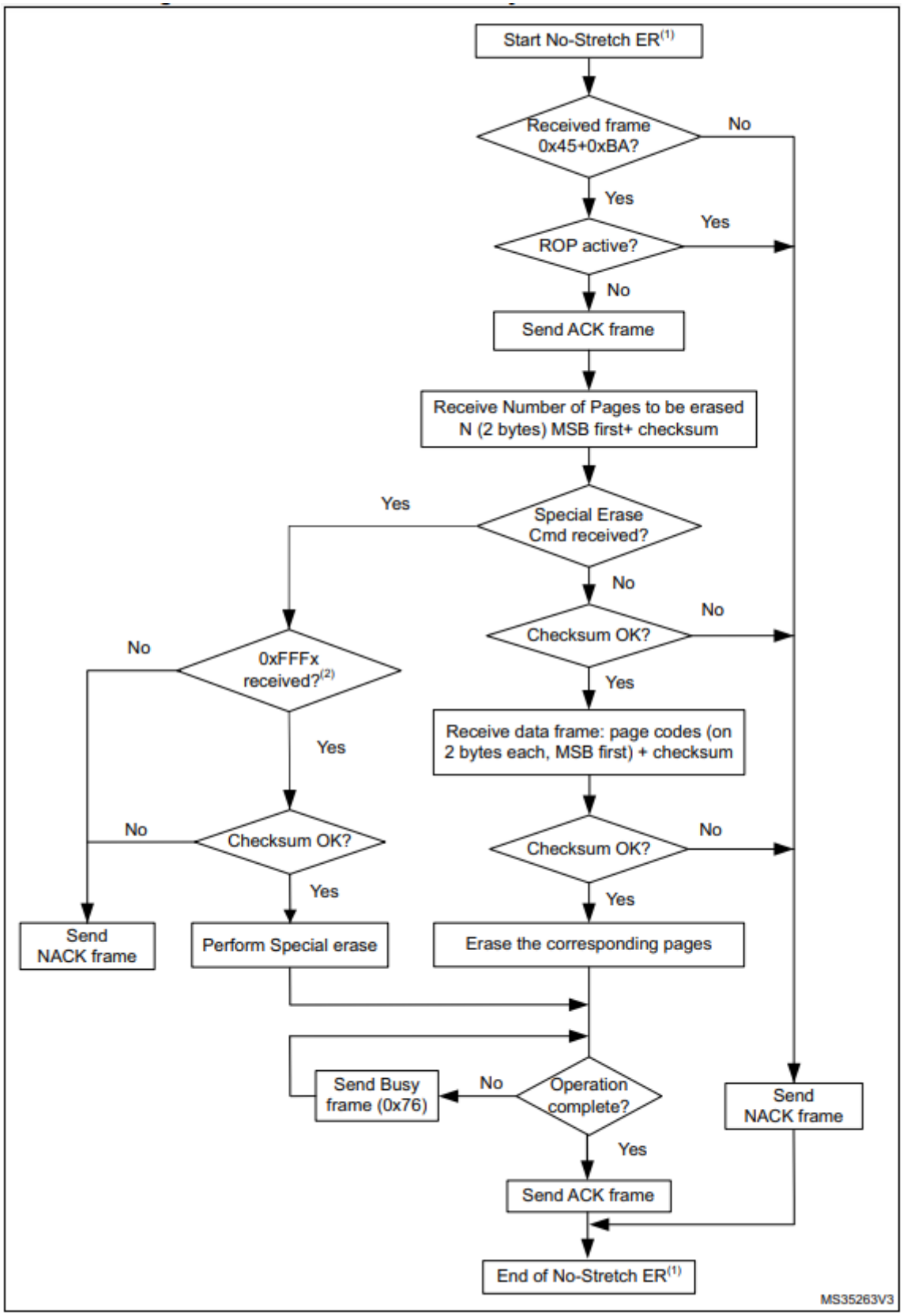
Рисунок 26. Команда No-Stretch Erase Memory: сторона хоста



1. ER = Erase Memory.

Some products do not support the Special erase feature. For these products, this command is NACK-ed.

Рисунок 27. Команда No-Stretch Erase Memory: сторона устройства



1. ER = Erase Memory.
2. Специальная команда стирания возвращает NACK, если она не поддерживается используемым продуктом STM32.

2.14 Команда Special

Новые команды загрузчика необходимы для поддержки новых функций STM32 и удовлетворения потребностей клиентов. Чтобы избежать конкретных команд для одного проекта, была создана универсальная команда Special.

Когда загрузчик получает команду Special, он передает хосту байт ACK. После передачи ACK загрузчик ожидает код операции подкоманды (два байта, первый MSB) и байт контрольной суммы. Если подкоманда поддерживается загрузчиком STM32 и ее контрольная сумма верна, загрузчик передает байт ACK, в противном случае он передает байт NACK и прерывает команду.

Чтобы команда оставалась универсальной, пакет данных, получаемый загрузчиком, может иметь разные размеры в зависимости от потребностей подкоманды.

Таким образом, пакет делится на две части:
- Размер данных (2 байта, старший бит вперед)
- N-байт данных (если N = 0, данные не передаются; N должно быть меньше 128)

Если все условия соблюдены (N ≤ 128 и контрольная сумма верна), загрузчик отправляет ACK, в противном случае он передает байт NACK и прерывает команду.

После выполнения подкоманды с использованием полученных данных загрузчик отправляет ответ, состоящий из двух последовательных пакетов:
- Пакет данных
  - Размер данных (2 байта, старший бит вперед)
  - N-байт данных (если N = 0, данные не передаются)
- Статусный пакет
  - Размер данных состояния (2 байта, старший бит вперед)
  - N байт данных (если N = 0, данные состояния не передаются)

Байт ACK закрывает взаимодействие команды Special между загрузчиком и хостом.

Рисунок 28. Команда Special: сторона хоста

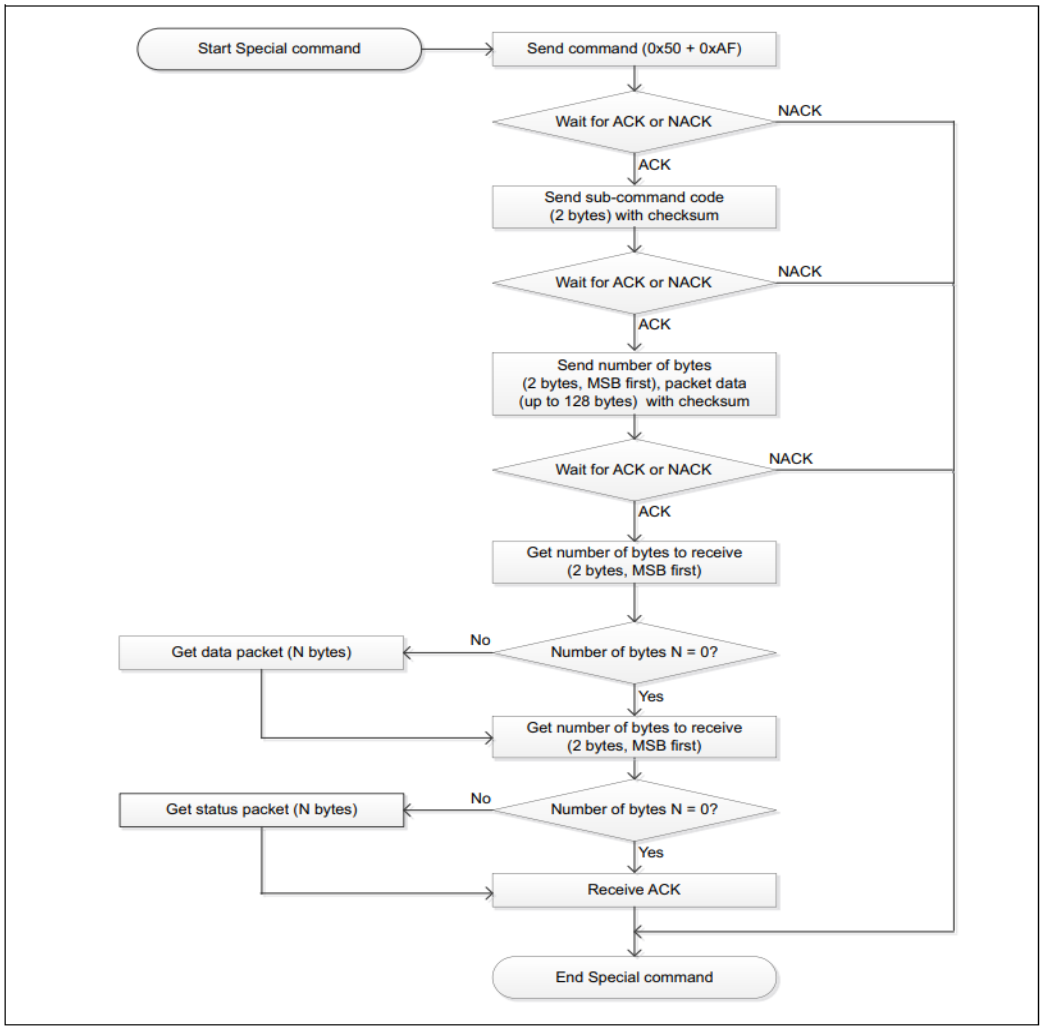
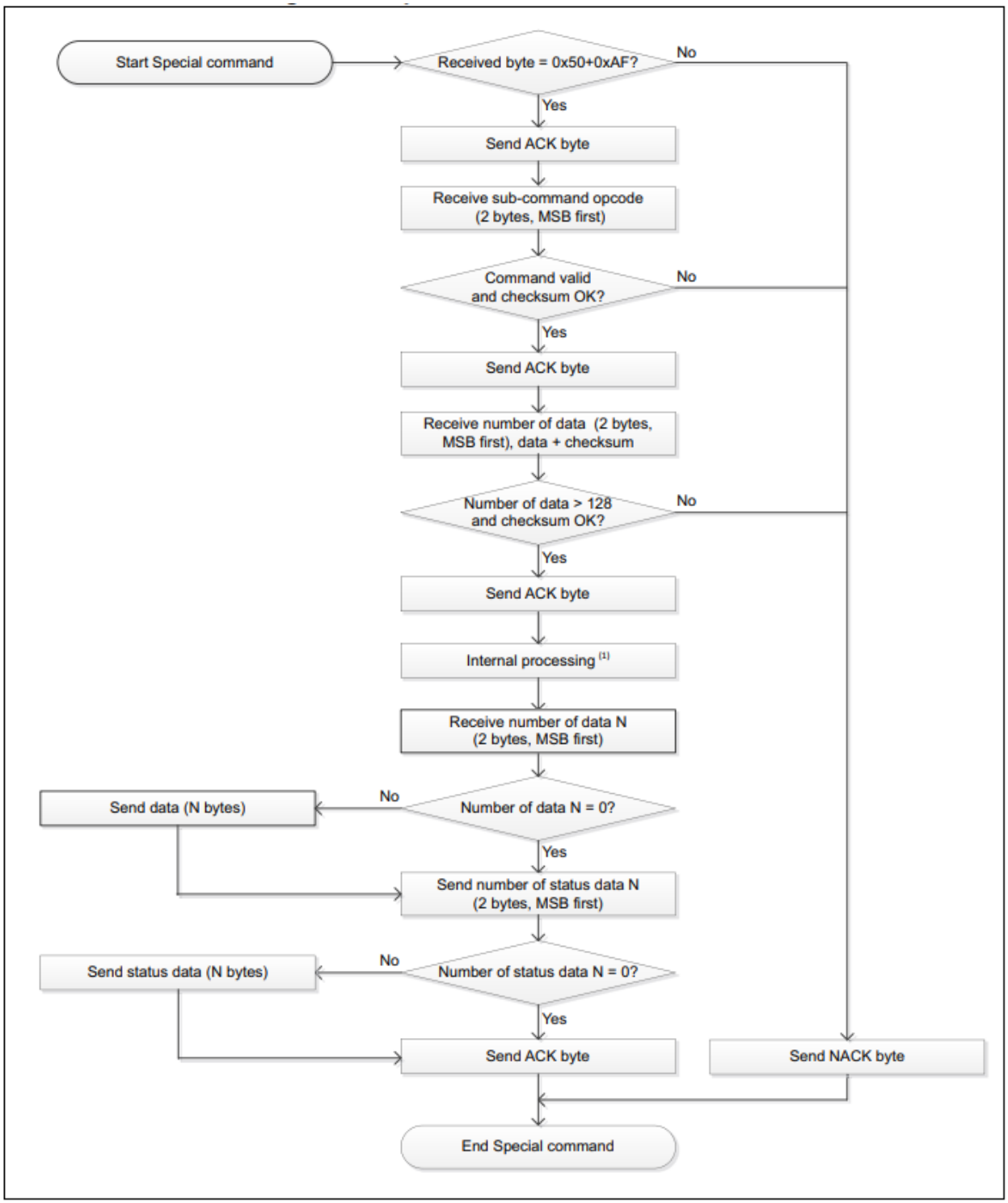


Рисунок 29. Команда Special: сторона устройства



1. Внутренняя обработка зависит от потребностей проекта.

2.15 Команда Extended Special

Эта команда немного отличается от команды Special. Он позволяет пользователю отправлять больше данных с добавлением нового буфера размером 1024 байта, а в качестве ответа возвращает только статус команд.

Когда загрузчик получает команду Extended Special, он передает хосту байт ACK. Как только ACK передан, загрузчик ожидает код подкоманды (два байта, первый MSB) и байт контрольной суммы. Если подкоманда поддерживается загрузчиком STM32 и ее контрольная сумма верна, загрузчик передает байт ACK, в противном случае он передает байт NACK и прерывает команду.

Затем могут быть получены два пакета в зависимости от потребностей подкоманды:
- Packet1: пакет Data1, в котором количество байтов ограничено 128 байтами.
- Packet2: пакет Data2, в котором количество байтов ограничено 1024 байтами.

Если все условия соблюдены (Пакет1: N ≤ 128 и контрольная сумма верна, Пакет2: N ≤ 1024 и контрольная сумма верна), загрузчик передает ACK, в противном случае он передает байт NACK и прерывает выполнение команды.

После выполнения подкоманды с использованием полученных данных загрузчик отправляет ответ, состоящий из одного пакета:
- Размер данных (2 байта, старший бит вперед)
- N-байт данных (если N = 0, данные не передаются)

Байт ACK закрывает взаимодействие команды Extended Special между загрузчиком и хостом.

Рисунок 30. Команда Extended Special: сторона хоста

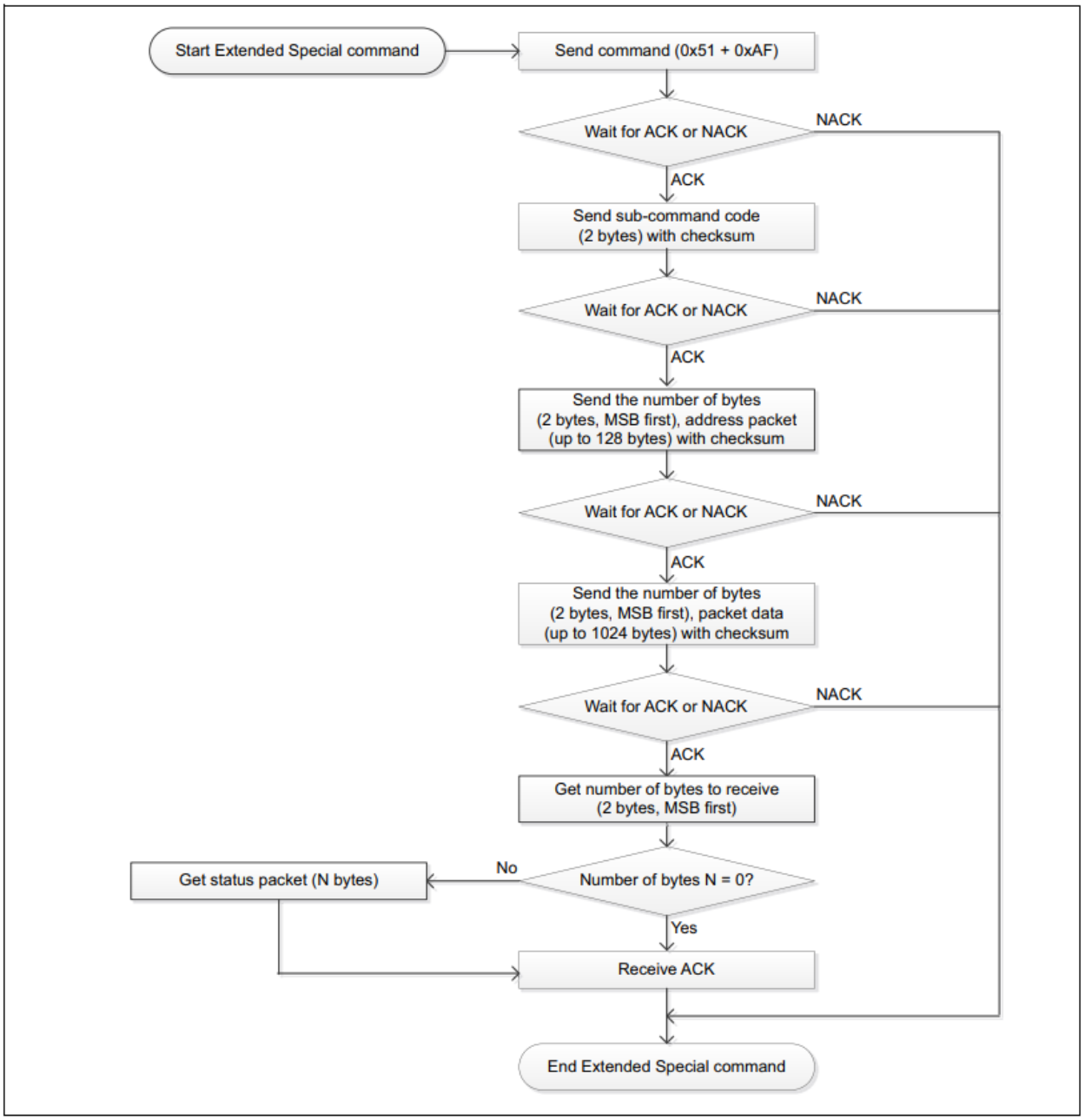
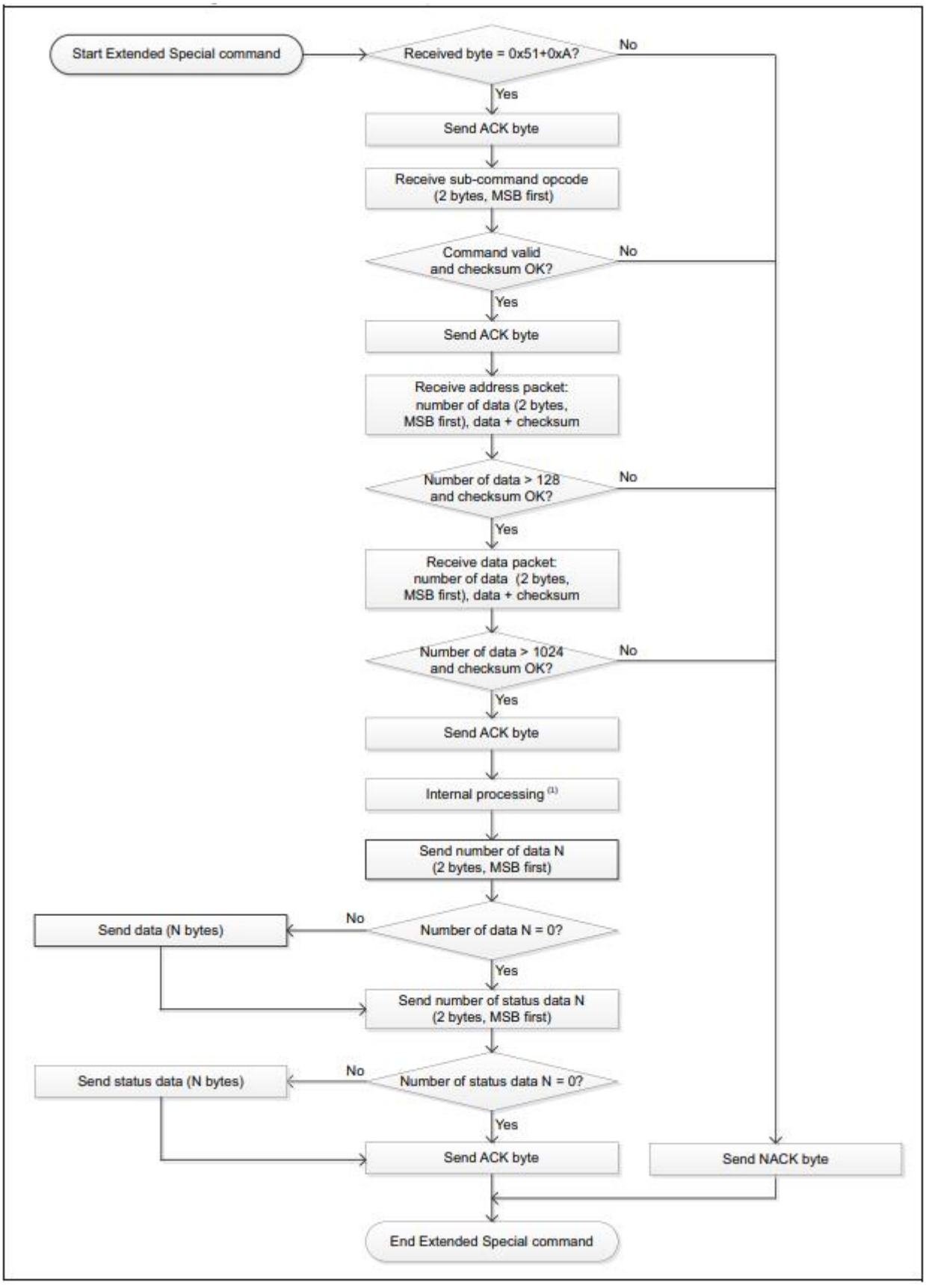


Рисунок 31. Команда Extended Special: сторона устройства



1. Внутренняя обработка зависит от потребностей проекта.

2.16 Команда No-Stretch Write Protect

Команда No-Stretch Write Protect используется для включения защиты от записи для некоторых или всех секторов флэш-памяти.

Когда загрузчик получает команду No-Stretch Write Protect, он передает хосту байт ACK. Затем загрузчик ожидает получения количества байтов (секторов, которые необходимо защитить), затем получает коды секторов флэш-памяти от приложения и возвращает состояние занятости BUSY(0x76), пока операция продолжается.

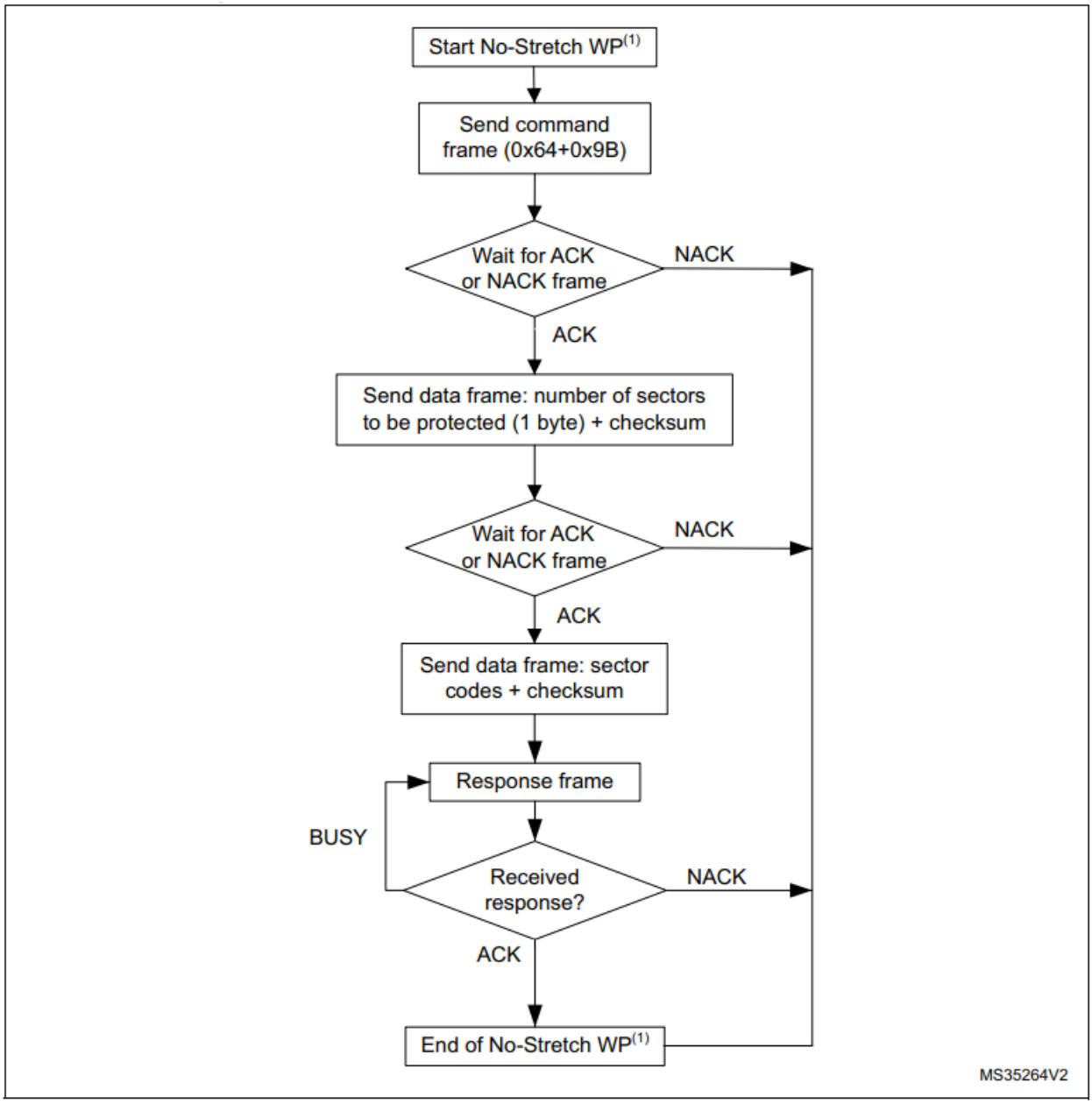
В конце команды No-Stretch Write Protect загрузчик передает байт ACK и генерирует сброс системы, чтобы применились изменения, внесенные в байты конфигурации.

Последовательность команда No-Stretch Write Protect следующая:
- Загрузчик получает байт, содержащий N, количество защищаемых от записи секторов - 1 (0 < N <= 255).
- Загрузчик получает (N+1) байт, каждый байт из которых содержит код сектора.

Примечание: Ни общее количество секторов, ни количество защищаемых секторов не проверяются. Это означает, что при передаче команды с неверным числом защищаемых секторов или с неверным номером сектора ошибка не возвращается.

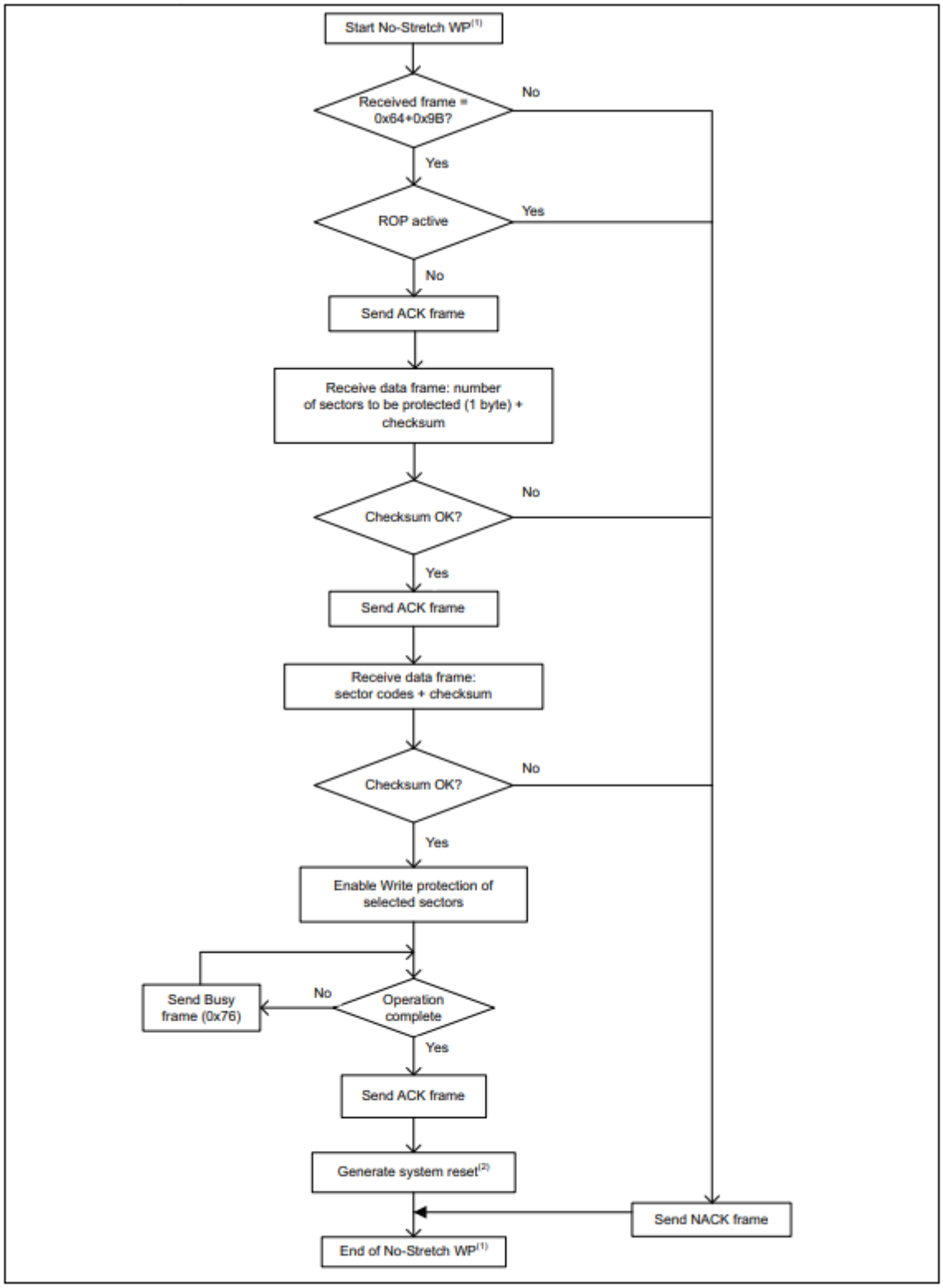
Если выполняется вторая команда защиты от записи, секторы флэш-памяти, которые были защищены первой командой, становятся незащищенными, и только те сектора, которые были переданы второй командой защиты от записи, становятся защищенными.

Рисунок 32. Команда No-Stretch Write Protect: сторона хоста



1. WP = Write Protect.

Рисунок 33. Команда No-Stretch Write Protect: сторона устройства



1. WP = Write Protect.
2. System reset is called only for some STM32 BL (STM32F4/F7) and some STM32L4 (STM32L412xx/422xx, STM32L43xxx/44xxx, STM32L45xxx/46xxx) products. In all other STM32 products no system reset is called.

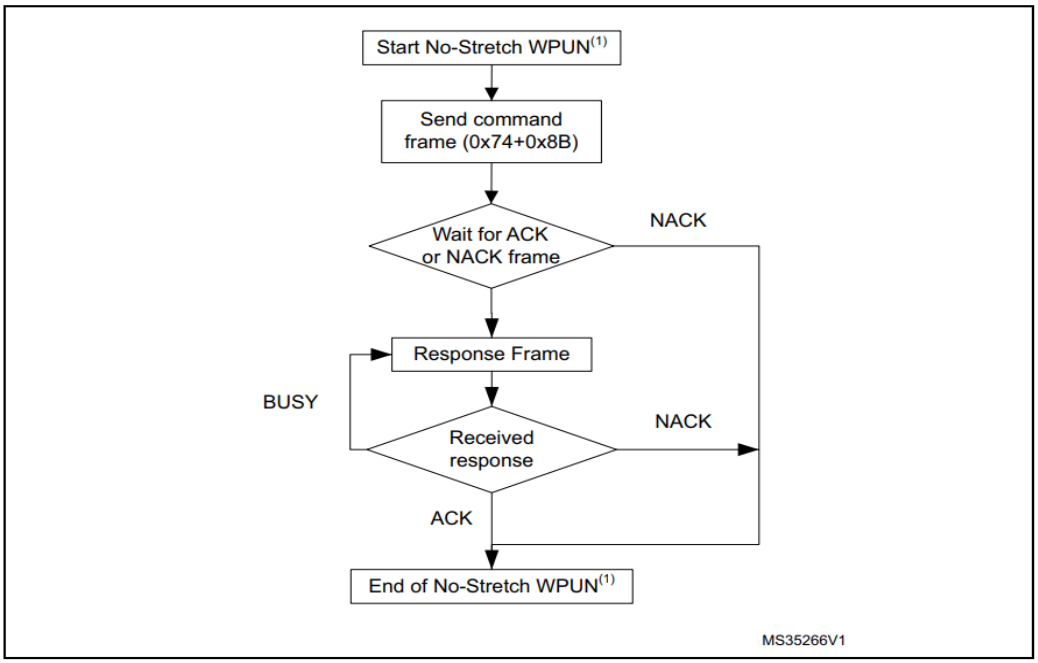
2.17 Команда No-Stretch Write Unprotect

Команда No-Stretch Write Unprotect используется для отключения защиты от записи всех секторов флэш-памяти.

Когда загрузчик получает команду No-Stretch Write Unprotect, он передает хосту байт ACK. Затем загрузчик отключает защиту от записи всех секторов флэш-памяти, возвращает состояние занятости BUSY(0x76) во время выполнения операции.

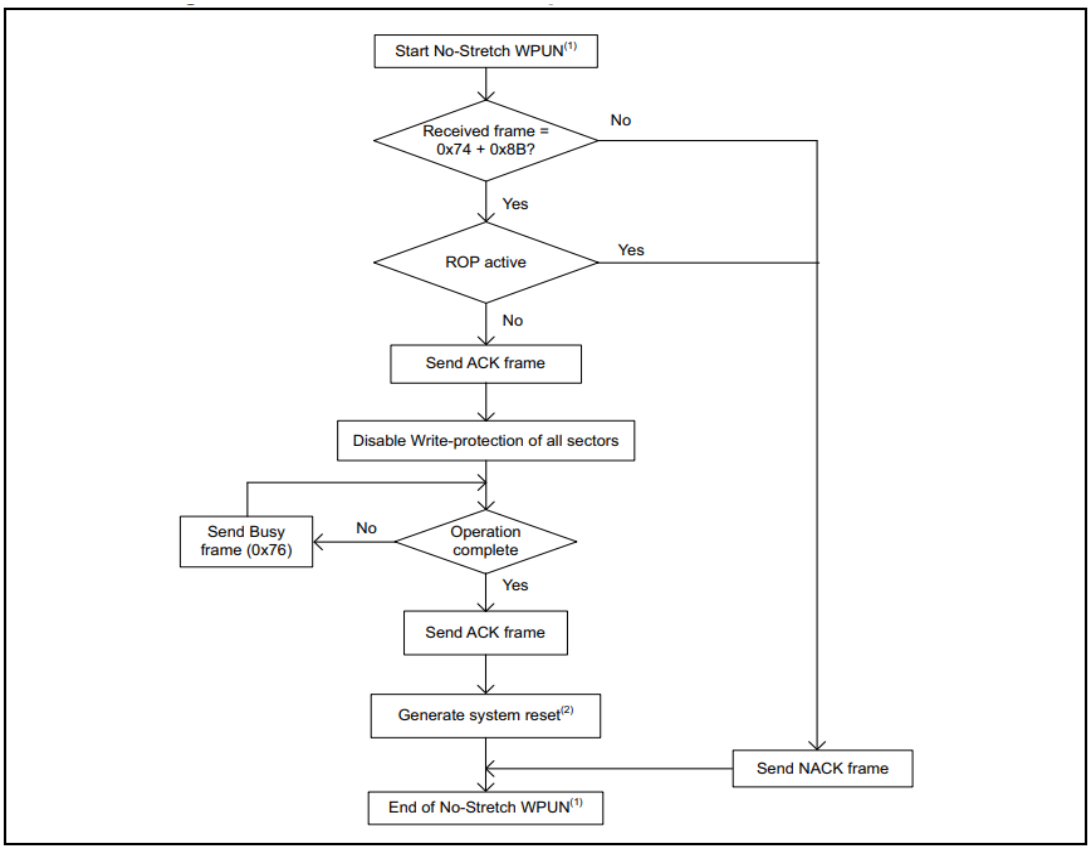
В конце команды No-Stretch Write Unprotect загрузчик передает байт ACK и генерирует сброс системы, чтобы применились изменения, внесенные в байты конфигурации.

Рисунок 34. Команда No-Stretch Write Unprotect: сторона хоста



1. WPUN = Write Unprotect.

Рисунок 35. Команда No-Stretch Write Unprotect: сторона устройства



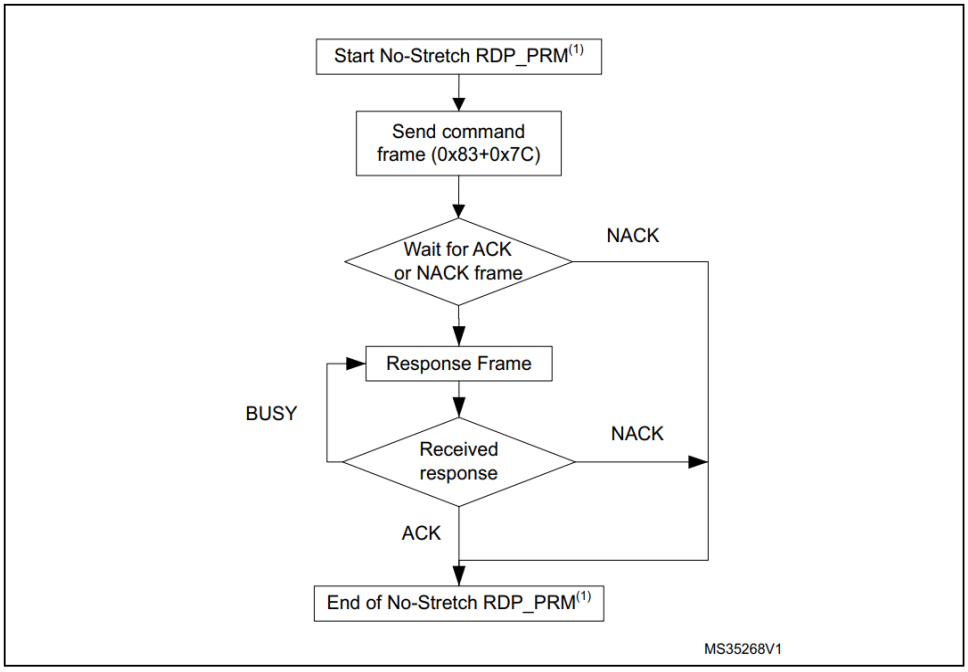
1. WPUN = Write Unprotect.
2. System reset is called only for some STM32 BL (STM32F4/F7) and some STM32L4 (STM32L412xx/422xx, STM32L43xxx/44xxx, STM32L45xxx/46xxx) products. In all other STM32 products no system reset is called.

2.18 Команда No-Stretch Readout Protect

Команда No-Stretch Readout Protect используется для включения защиты от чтения флэш-памяти. Когда загрузчик получает команду No-Stretch Readout Protect, он передает хосту байт ACK, включает защиту от чтения для флэш-памяти и возвращает состояние занятости BUSY(0x76) во время выполнения операции.

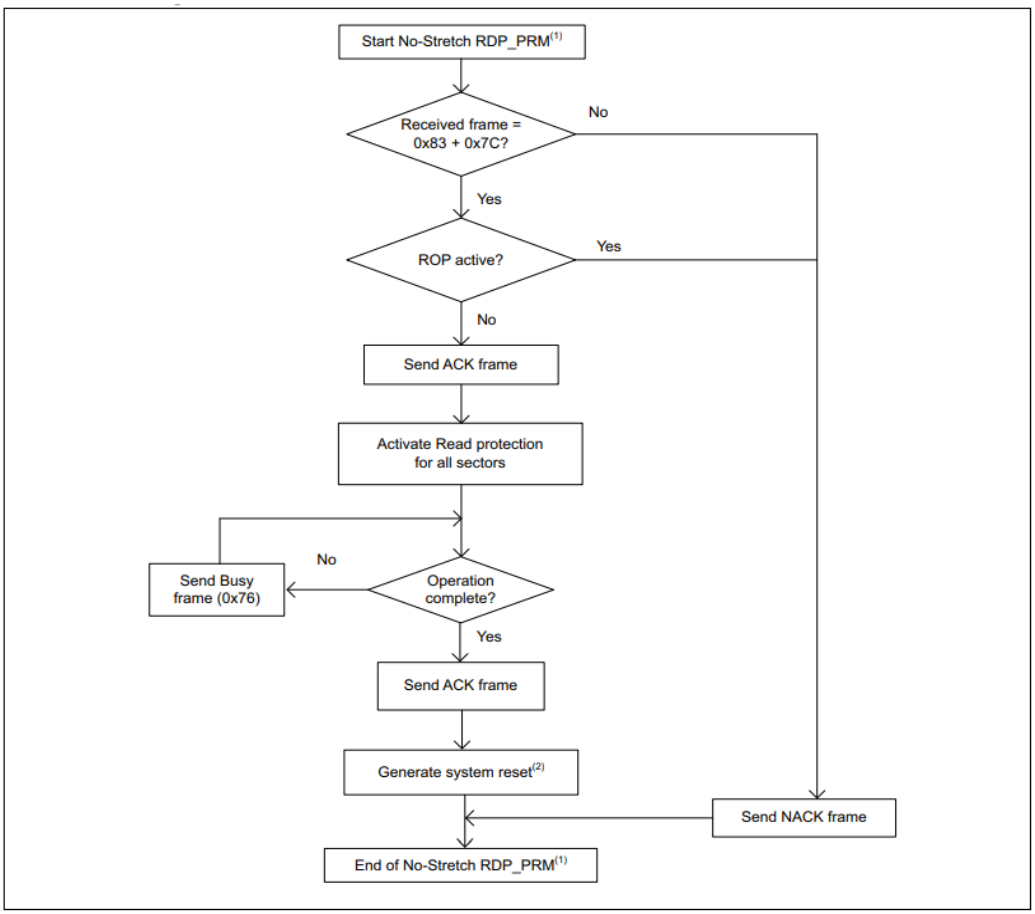
В конце команды No-Stretch Readout Protect загрузчик передает байт ACK и генерирует сброс системы, чтобы применились изменения, внесенные в байты конфигурации.

Рисунок 36. Команда No-Stretch Readout Protect: сторона хоста



1. RDP_PRM = Readout Protect.

Рисунок 37. Команда No-Stretch Readout Protect: сторона устройства



1. RDP_PRM = Readout Protect.
2. System reset is called only for some STM32 BL (STM32F4/F7) and some STM32L4 (STM32L412xx/422xx, STM32L43xxx/44xxx, STM32L45xxx/46xxx) products. In all other STM32 products no system reset is called.

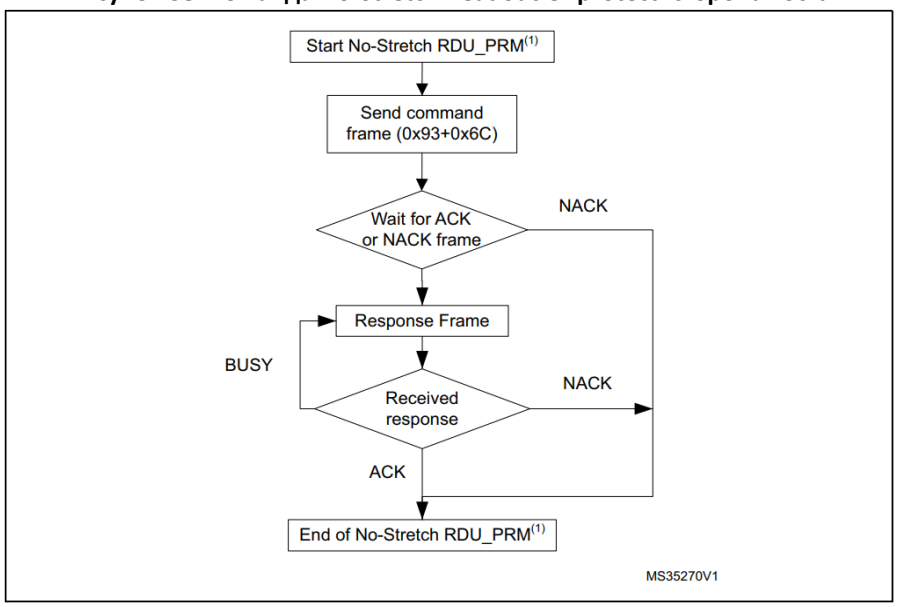
2.19 Команда No-Stretch Readout Unprotect

Команда No-Stretch Readout Unprotect используется для отключения защиты от чтения флэш-памяти.

Когда загрузчик получает команду No-Stretch Readout Unprotect, он передает хосту байт ACK. Затем загрузчик отключает защиту от чтения для всей флэш-памяти, что приводит к стиранию всей флэш-памяти и возвращает состояние занятости BUSY(0x76) во время выполнения операции. Если операция не удалась, загрузчик передает NACK и защита от чтения остается активной.

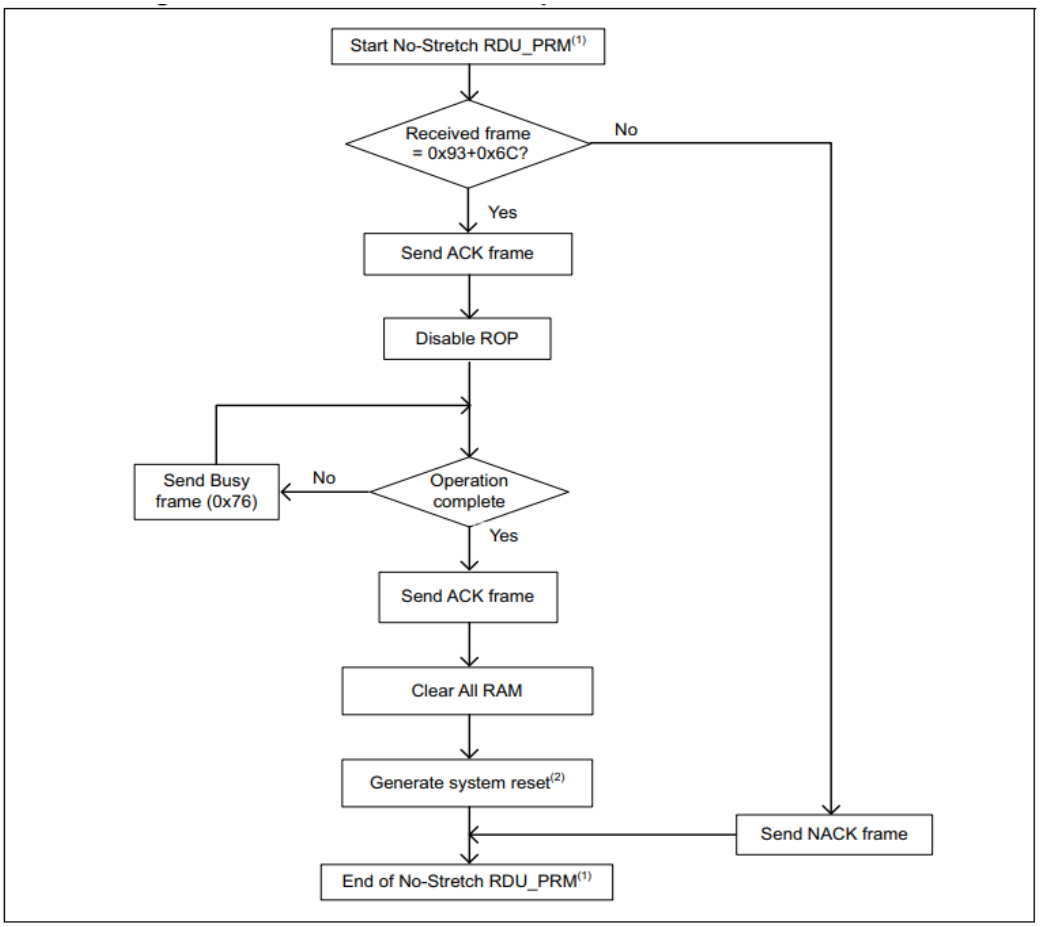
В конце команды No-Stretch Readout Unprotect загрузчик передает байт ACK и генерирует сброс системы, чтобы применились изменения, внесенные в байты конфигурации.

Рисунок 38. Команда No-Stretch Readout Unprotect: сторона хоста



1. RDU_PRM = Readout Unprotect.

Рисунок 39. Команда No-Stretch Readout Unprotect: сторона устройства



1. RDU_PRM = Readout Unprotect.
2. System reset is called only for some STM32 BL (STM32F4/F7) and some STM32L4 (STM32L412xx/422xx, STM32L43xxx/44xxx, STM32L45xxx/46xxx) products. In all other STM32 products no system reset is called.

2.20 Команда No-Stretch GetCheckSum

Команда No-Stretch GetCheckSum используется для вычисления значения CRC (на основе CRC IP) заданного диапазона флэш-памяти, определяемого смещением и размером памяти.

Когда загрузчик получает команду No-Stretch GetCheckSum, он передает приложению байт ACK. Затем загрузчик ожидает 4-байтового адреса (1-й байт — MBS, 4-й байт — LBS) и байта контрольной суммы, затем он проверяет полученный адрес. Если адрес флэш-памяти валиден и контрольная сумма верна, загрузчик передает байт ACK, в противном случае он передает байт NACK и прерывает выполнение команды.

Если адрес флэш-памяти валиден и контрольная сумма верна, загрузчик ожидает размер диапазона памяти (4 байта, 1-й байт — MBS, 4-й байт — LBS) для вычисления контрольной суммы и байт контрольной суммы. Если размер отличен от 0, кратного четырём, в результате чего адрес во Flash прибавляется к адресу, а контрольная сумма верна, загрузчик отправляет ACK приложению, в противном случае он передает байт NACK и прерывает команду.

Если адрес и размер допустимы, приложение ожидает вычисления CRC. Состояние BUSY(0x76) возвращается во время выполнения операции.

В конце команды, если операция GetCheckSum прошла успешно, загрузчик передает результат контрольной суммы (4 байта, старший бит вперед) и ее контрольную сумму.

Рисунок 40. Команда No-Stretch GetCheckSum: сторона хоста

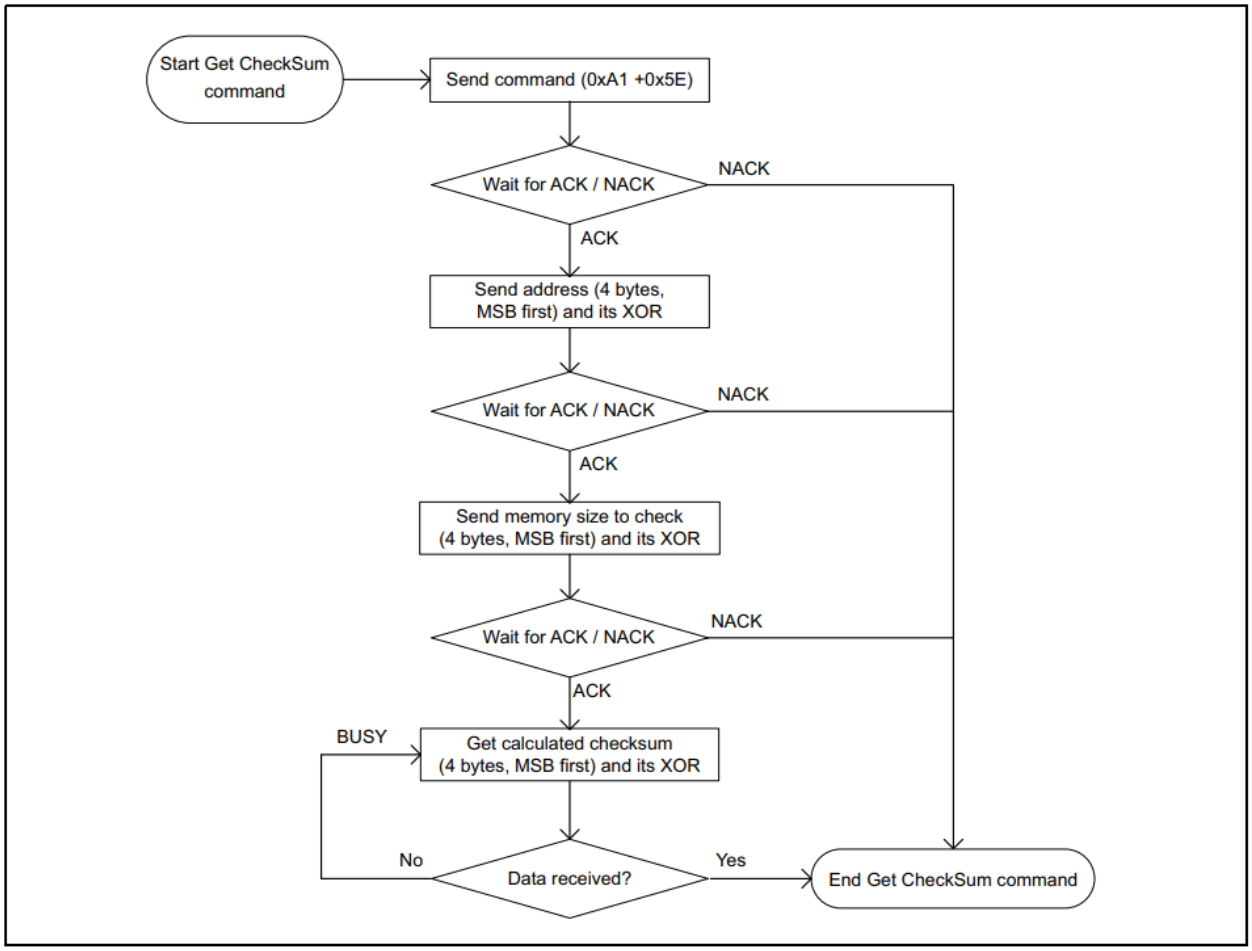


Рисунок 41. Команда No-Stretch GetCheckSum: сторона устройства