Встроенная системная программа «ВПО Aladdin eFlash»

Описание программы

RU.АЛДЕ.01.02.006 13 01

Содержание

Аннотация

Данный документ содержит описание работы встроенной системной программы «ВПО Aladdin eFlash», а также описание:

- SCSI-команд, реализованных и используемых изделием «Доверенный корпоративный флеш-накопитель Aladdin eFlash»;
- перечня изменяемых и неизменных параметров, задаваемых при кастомизации Aladdin eFlash.

Изделие поддерживает работу со следующими видами SCSI-команд:

- Стандартные (обязательные).
- Дополнительные стандартные.
- Проприетарные (нестандартные).

1. Общие сведения

1.1. Назначение

Встроенная системная программа «ВПО Aladdin eFlash» RU.АЛДЕ.01.02.006 предназначена для функционирования в изделии «Доверенный корпоративный флеш-накопитель Aladdin eFlash», обеспечивающая изделие базовым набором SCSI-команд, а также дополнительными функциональными возможностями.

1.2. Совместное использование с другими изделиями

Встроенная системная программа предназначена для применения в составе Aladdin eFlash и является модулем для среды функционирования «Встроенная системная программа JaCarta OS» RU.АЛДЕ.01.02.002-01. ВПО Aladdin eFlash хранится в отдельном SPI модуле (памяти), а её загрузка осуществляется с помощью встроенной системной программы «Загрузчик JaCarta OS» RU.АЛДЕ.01.01.021, установленной в микроконтроллере изделий Aladdin eFlash.

Большая часть функций ВПО Aladdin eFlash может использоваться стандартными средствами операционных систем семейств Linux, Windows, macOS путём отправки SCSI-команд и получения ответов на них.

Также ВПО Aladdin eFlash в составе исполнений 1 и 2 изделия «Доверенный корпоративный флеш-накопитель Aladdin eFlash» реализует часть функций безопасности при совместном использовании с программным средством «Средство защиты информации на съёмных машинных носителях Aladdin SecurFlash» RU.АЛДЕ.03.01.046.

2. Стандартные (обязательные) команды

Во встроенной системной программе «ВПО Aladdin eFlash» реализованы следующие стандартные обязательные команды:

Таблица 1 — Перечень поддерживаемых стандартных SCSI-команд

Команда	Код
TEST UNIT READY	(0x00)
REQUEST SENSE (6)	(0x03)
INQUIRY	(0x12)
MODE SENCE(6)	(0x1A)
START STOP UNIT	(0x1B)
PREVENT ALLOW MEDIUM REMOVAL	(0x1E)
READ CAPACITY(10)	(0x25)
READ FORMAT CAPACITIES	(0x23)
READ(10)	(0x28)
WRITE(10)	(0x2A)
VERIFY(10)	(0x2F)

2.1. Описание команды TEST UNIT READY (0x00)

Команда TEST UNIT READY позволяет проверить готовность к работе логической единицы. Команда возвращает статус GOOD если логический блок может принять соответствующую команду доступа без возврата статуса CHECK CONDITION. Команда заканчивается статусом CHECK CONDITION (при этом устройство считается не готовым к работе – NOT READY) если логический блок не может считается работоспособным или находится в таком состоянии что требуется дополнительное действие для подготовки логического блока.

Формат команды представлен ниже.

Таблица 2 — Формат команды TEST UNIT READY (0x00)

Bit Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (00h)							
1	Reserved							
...								
4								
5	CONTROL							

Описание ответов на команду TEST UNIT READY представлено в таблице ниже.

Таблица 3 — Описание ответов на команду TEST UNIT READY (0x00)

Статус	Ключевое значение	Дополнительное описание значения (англ.)	Дополнительное описание значения (рус.)
GOOD	not applicable	not applicable	Блок работает корректно
CHECK CONDITION	ILLEGAL REQUEST	LOGICAL UNIT NOT SUPPORTED	Логический блок не поддерживается
CHECK CONDITION	NOT READY	LOGICAL UNIT DOES NOT RESPOND TO SELECTION	Логический блок не реагирует на выбор
CHECK CONDITION	NOT READY	MEDIUM NOT PRESENT	Среда отсутствует
CHECK CONDITION	NOT READY	LOGICAL UNIT NOT READY, CAUSE NOT REPORTABLE	Логический блок не готов, причина не сообщается
CHECK CONDITION	NOT READY	LOGICAL UNIT IS IN PROCESS OF BECOMING READY	Логический блок в процессе подготовки к работе

Статус	Ключевое значение	Дополнительное описание значения (англ.)	Дополнительное описание значения (рус.)
CHECK CONDITION	NOT READY	LOGICAL UNIT NOT READY, INITIALIZING COMMAND REQUIRED	Логический блок не готов, необходима команда по инициализации
CHECK CONDITION	NOT READY	LOGICAL UNIT NOT READY, MANUAL INTERVENTION REQUIRED	Логический блок не готов, требуется ручное вмешательство
CHECK CONDITION	NOT READY	LOGICAL UNIT NOT READY, FORMAT IN PROGRESS	Логический блок не готов, выполняется форматирование

2.2. Описание команды REQUEST SENSE (0x03)

Команда REQUEST SENSE запрашивает перевод данных считывания устройства клиенту приложения.

Формат команды представлен ниже.

Таблица 4 — Формат команды REQUEST SENSE (0x03)

Bit Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (03h)							
1	Reserved							DESC
2	Reserved							
3								
4	ALLOCATION LENGTH							
5	CONTROL							

DESC (Description Format) bit

Бит формата описания указывает в каком виде должна быть представлена информация:

0 – формат описания соответствует п.2.4.1.2 [1].

1 – формат описания соответствует п.2.4.1.1 [1].

ALLOCATION LENGTH field

Поле ALLOCATION LENGTH определено в п.2.2.6 [1]. Клиенты приложений запрашивают 252 байта данных считывания, чтобы гарантировать, что они получают все данные. Если запрашивается менее 252 байт, данные считывания могут быть потеряны, так как команда REQUEST SENSE с любой длиной выделения очищает критически важные данные.

Критически важные данные должны быть доступны и очищены в соответствии с условиями, определенными в SAM-5. Если токен не имеет других данных считывания доступных для возврата, он:

1. Возвращает sense key значение NO SENSE, дополнительный sense code в значение NO ADDITIONAL SENSE INFORMATION.
2. Завершает команду REQUEST SENSE со статусом GOOD.
3. После завершения команды логический блок возвращается к тому же состоянию питания, которое было активно до REQUEST SENSE. Команда REQUEST SENSE не должна сбрасывать таймеры режима питания.

Токен возвращает статус CHECK CONDITION для команды REQUEST SENSE только для сообщения об исключительных условиях непосредственно для команды REQUEST SENSE.

Примеры условий, при которых команда REQUEST SENSE возвращает статус CHECK CONDITION:

1. В CDB обнаружено недопустимое значение поля.

2. Токен не поддерживает команду REQUEST SENSE.
3. Неустраненная ошибка обнаружена подсистемой предоставления услуг.
4. Сбой препятствует возврату данных считывания.

Токены возвращают не менее 18 байт данных параметров в ответ на команду REQUEST SENSE, если длина выделения равна 18 или более, а бит DESC установлен в 0. Клиенты приложения могут определить, сколько смысловых данных было возвращено путём проверки поля ALLOCATION LENGTH в CDB и поля ADDITIONAL SENSE LENGTH в данных считывания.

2.3. Описание команды INQUIRY (0x12)

Команда INQUIRY запрашивает передачу информации о логическом блоке и целевом устройстве SCSI в клиент приложения.

Формат команды представлен ниже.

Таблица 5 — Формат команды INQUIRY (0x12)

Bit Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (12h)							
1	Reserved						Obsolete Formerly CMDDT	EVPD
2	PAGE CODE							
3	(MSB)	ALLOCATION LENGTH						
4								(LSB)
5	CONTROL							

EVPD (Enable Vital Product Data) bit

Бит включения существенных данных о продукте (EVPD), установленный в единицу, указывает, что токен должен возвращать существенные данные о продукте, указанные в поле PAGE CODE.

0 — Если бит EVPD установлен на ноль, токен возвращает стандартные данные INQUIRY. Если значение поля PAGE CODE не равно 0 когда бит EVPD установлен в 0, команда завершается со статусом CHECK CONDITION, с ключом считывания, установленным в значение ILLEGAL REQUEST, а дополнительный код считывания устанавливается в INVALID FIELD в CDB.

1 — Когда бит EVPD установлен в единицу, поле PAGE CODE указывает, какую страницу информации о продукте токен должен вернуть.

CMDDT (Command Support Data) bit

Если оба бита EVPD и CMDDT равны 1, команда должна вернуть статус CHECK CONDITION с ключом считывания, установленным в значение ILLEGAL REQUEST, и дополнительный смысловой код недопустимого поля в CDB. Когда бит EVPD равен 1, поле PAGE или OPERATION CODE указывает, какую страницу информации о продукте должен вернуть адресат.

ALLOCATION LENGTH field

Поле ALLOCATION LENGTH определено в п.2.2.6 [1]. Если значение EVPD равно 0, то длина распределения должна быть не менее пяти, чтобы возвращалось поле ADDITIONAL LENGTH в данных параметра. Если значение EVPD равно единице, длина распределения должна быть не менее четырех, чтобы возвращалось поле PAGE LENGTH в данных параметра.

2.4. Описание команды MODE SENSE(6) (0x1A)

Команда MODE SENSE (6) предоставляет токену средства для сообщения параметров клиенту приложения. Эта команда дополняет команду MODE SELECT.

Формат команды представлен ниже.

Таблица 6 — Формат команды MODE SENSE(6) (0x1A)

Bit Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (1Ah)							
1	Reserved				DBD	Reserved		
2	PC		PAGE CODE					
3	SUBPAGE CODE							
4	ALLOCATION LENGTH							
5	CONTROL							

DBD (disable block descriptors) bit

0 — Бит отключения дескрипторов блоков (DBD), установленный в 0, указывает, что токен может вернуть 0 или более дескрипторов блоков в возвращенные данные MODE SENSE.

1 — Бит DBD, установленный в 1, указывает, что токен не должен возвращать какие-либо блочные дескрипторы в возвращенных данных MODE SENSE.

PC (Page Control) field

Поле «Управление страницами» (PC) указывает тип значений параметров режима, возвращаемых на страницах режима. Описание поля представлено в таблице ниже.

Таблица 7 — Описание поля Page Control (PC)

Код	Тип параметра	Ссылка на [1]
00b	Current values	3.11.1.1
01b	Changeable values	3.11.1.2
10b	Default values	3.11.1.3
11b	Saved values	3.11.1.4

Поле PC влияет только на параметры режима в страницах режима, однако бит PS, бит SPF, поле PAGE CODE, поле SUBPAGE CODE и поле PAGE LENGTH возвращают текущие значения (то есть, как если бы PC был установлен в 00b). Заголовок и дескриптор блока параметров режима возвращают текущие значения.

PAGE CODE and SUBPAGE CODE fields

Поля PAGE CODE и SUBPAGE CODE указывают, какие страницы и подстраницы режима должны возвращаться.

ALLOCATION LENGTH field

Поле ALLOCATION LENGTH определено в п.2.2.6 [1].

Клиент приложения может запросить любую одну или все страницы поддерживаемого режима с токена. Если клиент приложения выдает команду MODE SENSE с кодом страницы или значением кода подстраницы, не реализованным логическим блоком, команда завершается со статусом CHECK CONDITION, с ключом считывания, установленным в ILLEGAL REQUEST, и дополнительным кодом считывания, установленным в INVALID FIELD IN CDB.

2.5. Описание команды START STOP UNIT (0x1B)

Команда START STOP UNIT запрашивает у токена изменение состояния питания логического блока или загрузку, или извлечение носителя. Это включает в себя указание, того что токен включает или отключает устройство блокировки прямого доступа для операций доступа к среде путем управления условиями питания и таймерами.

Если команда START STOP UNIT обрабатывается токеном, и последующая команда START STOP UNIT для которой CDB проверяется, запрашивает, чтобы логический блок переключился на состояние питания, отличное от указанного в START STOP UNIT, то токен должен завершить последующую команду START STOP UNIT с состоянием CHECK CONDITION с ключом считывания, установленным в состояние NOT READY, и дополнительным кодом считывания, установленным в состояние LOGICAL UNIT NOT READY, START STOP UNIT COMMAND IN PROGRESS.

Формат команды представлен ниже.

Таблица 8 — Формат команды START STOP UNIT (0x1B)

Bit Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (1Bh)							
1	Reserved							IMMED
2	Reserved							
3	Reserved				POWER CONDITION MODIFER			
4	POWER CONDITION				Reserved	NO_ FLUSH	LOEJ	START
5	CONTROL							

IMMED (Immediate) bit

0 — Если бит IMMED установлен в 0, то токен должен вернуть статус после завершения операции.

1 — Если бит IMMED равен 1, то токен должен вернуть статус, как только CDB будет проверен.

POWER CONDITION field and POWER CONDITION MODIFIER field

Комбинации значений в полях POWER CONDITION и POWER CONDITION MODIFIER определены в таблице ниже. Если поле POWER CONDITION поддерживается и установлено на значение, отличное от 0h, то биты START и LOEJ игнорируются.

Таблица 9 — Описание взаимодействия полей POWER CONDITION и POWER CONDITION MODIFIER

POWER CONDITION	POWER CONDITION Name	POWER CONDITION MODIFIER value	Описание (англ.)	Описание (рус.)
0h	START_VALID	0h	Process the START and LOEJ bits	Обработка битов START и LOEJ
1h	ACTIVE	0h	Cause the logical unit to transition to the active power condition (see SPC-5)	Вызвать переход логического блока в активное состояние питания
2h	IDLE	0h	Cause the logical unit to transition to the idle_a power condition (see SPC-5)	Вызвать переход логического блока в состояние idle_a питания
		1h	Cause the logical unit to transition to the idle_b power condition (see SPC-5)	Вызвать переход логического блока в состояние idle_b питания
			Cause the logical unit to transition to the idle_c power condition (see SPC-5)	Вызвать переход логического блока в состояние idle_c питания

POWER CONDITION	POWER CONDITION Name	POWER CONDITION MODIFIER value	Описание (англ.)	Описание (рус.)
3h	STANDBY	0h	Cause the logical unit to transition to the standby_z power condition (see SPC-5)	Вызвать переход логического блока в состояние standby_z питания
		1h	Cause the logical unit to transition to the standby_y power condition (see SPC-5)	Вызвать переход логического блока в состояние standby_y питания
5h	Obsolete	0h to Fh	Obsolete	Устарел
7h	LU_CONTROL	0h	Initialize and start all of the idle condition timers that are enabled (see SPC-5), and initialize and start all of the standby condition timers that are enabled (see SPC-5)	Инициализация и запуск всех активизированных таймеров состояния ожидания, а также инициализация и запуск всех активизированных таймеров состояния ожидания
Ah	FORCE_IDLE_0	0h	Force the idle_a condition timer to be set to zero (see SPC-5)	Принудительно установить таймер состояния idle_a на ноль
		1h	Force the idle_b condition timer to be set to zero (see SPC-5)	Принудительно установить таймер состояния idle_b на ноль
		2h	Force the idle_c condition timer to be set to zero (see SPC-5)	Принудительно установить таймер состояния idle_c на ноль
Bh	FORCE_STANDBY_0	0h	Force the standby_z condition timer to be set to zero (see SPC-5)	Принудительно установить таймер состояния standby_z на ноль (см. SPC-5)
		1h	Force the standby_y condition timer to be set to zero (see SPC-5)	Принудительно установить таймер состояния standby_y на ноль
Любые другие комбинации			Зарезервировано	

NO_FLUSH bit

0 — Если NO_FLUSH установлен в 0, то логические блоки, содержащие кэш, записывают все кэшированные логические блоки на носитель (например, как они делали бы в ответ на команду SYNCHRONIZE CACHE с нулевым битом SYNC_NV, нулевым полем LOGICAL BLOCK ADDRESS и нулевым полем NUMBER OF LOGICAL BLOCKS) до входа в любой режим питания, который препятствует доступу к среде.

1 — Если NO_FLUSH установлен в 1, то кэшированные логические блоки не записываются на носитель логическим блоком до ввода в любое состояние питания, которое препятствует доступу к среде.

LOEJ (load eject) bit

0 — Если бит LOEJ установлен в 0, то логический блок не должен предпринимать никаких действий относительно загрузки или извлечения среды.

1 — Если бит LOEJ установлен на 1, то логический блок должен выгрузить носитель, если бит START установлен в 0. Если бит LOEJ установлен в 1, то логический блок должен загрузить носитель, если бит START установлен в 1. Если поле POWER CONDITION поддерживается и имеет значение, отличное от 0h, то токен должен игнорировать бит LOEJ.

START bit

0 — Если бит START установлен в 0, то токен должен:

a) вызвать переход логического блока в состояние остановленного питания;

b) остановить любой таймер состояния простоя, который включен;

c) остановить любой таймер состояния ожидания, который включен.

1 — Если бит START установлен в 1, то токен должен:

a) соответствовать требованиям, определенным в стандартах транспортных протоколов SCSI;

b) вызвать переход логического блока в активное состояние питания;

c) инициализировать и запустить любой включенный таймер состояния простоя;

d) инициализировать и запустить любой включенный таймера состояния ожидания.

Если поле POWER CONDITION поддерживается и имеет значение, отличное от 0h, то токен должен игнорировать бит START.

2.6. Описание команды PREVENT/ALLOW MEDIUM REMOVAL (0x1E)

Команда PREVENT/ALLOW MEDIUM REMOVAL требует, чтобы библиотека разрешала или запрещала доступ оператора к почтовому слоту и журналам.

- Если разрешено, почтовый слот и журналы разблокированы. Почтовый слот может быть открыт, и журналы могут быть разблокированы из пользовательского интерфейса.
- Если это невозможно, почтовый слот и журналы заблокированы. Невозможно открыть почтовый слот и разблокировать журналы из пользовательского интерфейса.

Формат команды представлен ниже.

Таблица 10 — Формат команды PREVENT/ALLOW MEDIUM REMOVAL (0x1E)

	Bit							
Byte	7	6	5	4	3	2	1	0
0	Operation Code (1Eh)							
1	Ignored			Reserved				
2 to 3	Reserved							
4	Reserved							Prevent
5	Control Byte (00h)							

Prevent

0 — Разрешить – открывается доступ к почтовому слоту и журналам.

1 — Предотвратить – блокируется доступ к почтовым слотам и журналам.

2.7. Описание команды READ CAPACITY(10) (0x25)

2.7.1. Описание команды READ CAPACITY (10) overview

Команда READ CAPACITY (10) требует, чтобы токен передал 8 байтов данных параметров, описывающих емкость и формат среды устройства блока прямого доступа, в буфер ввода данных. Эта команда может быть обработана так, как если бы она имела атрибут задачи HEAD OF QUEUE. Если логический блок поддерживает защиту информации, клиент приложения должен использовать команду READ CAPACITY (16) вместо команды READ CAPACITY (10).

Формат команды представлен ниже.

Таблица 11 — Формат команды READ CAPACITY (10) overview

Bit Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (25h)							
1	Reserved							Obsolete
2	(MSB)							
...	LOGICAL BLOCK ADDRESS (Obsolete)							
5								(LSB)
6	Reserved							
7								
8	Reserved							PMI (Obsolete)
9	CONTROL							

Определение поля LOGICAL BLOCK ADDRESS представлено в разделе 2.2.3 [1].

LOGICAL BLOCK ADDRESS field

Поле LOGICAL BLOCK ADDRESS устанавливается в 0, если бит PMI установлен в 0. Если бит PMI установлен в 0, а поле LOGICAL BLOCK ADDRESS не установлено в 0, токен должен завершить команду со статусом CHECK CONDITION с ключом считывания, установленным на ILLEGAL REQUEST, и дополнительным кодом считывания, установленным на INVALID FIELD IN CDB.

PMI (Partial Medium Indicator) bit

0 — Бит индикатора частичной среды (PMI), установленный в 0, указывает, что токен возвращает информацию о последнем логическом блоке на устройстве блока прямого доступа.

1 — Бит PMI, установленный в 1, указывает, что токен возвращает информацию о последнем логическом блоке после указанного в поле LOGICAL BLOCK ADDRESS, прежде чем может возникнуть существенная задержка передачи данных, зависящая от производителя.

Эта функция предназначена для помощи программному обеспечению управления хранением в определении того, достаточно ли места, начиная с адреса логического блока, указанного в CDB, чтобы содержать часто используемую структуру данных (например, каталог файлов или индекс файлов) без дополнительной задержки.

2.7.2. Описание команды READ CAPACITY (10) parameter data

Данные параметра READ CAPACITY (10) определены в таблице ниже. Каждый раз, когда изменяются данные параметра READ CAPACITY (10), токен должен установить состояние внимания устройства.

Таблица 12 — Формат команды READ CAPACITY (10) parameter data

Bit Byte	7	6	5	4	3	2	1	0
0	(MSB)							
...	RETURNED LOGICAL BLOCK ADDRESS							
3								(LSB)
4	(MSB)							
...	BLOCK LENGTH IN BYTES							
7								(LSB)

RETURNED LOGICAL BLOCK ADDRESS field

Если количество логических блоков превышает максимальное значение, которое может быть указано в поле RETURN LOGICAL BLOCK ADDRESS, токен устанавливает в поле RETURN LOGICAL BLOCK ADDRESS значение FFFFFFFFh. Затем клиент приложения выдаёт команду READ CAPACITY (16) для извлечения данных параметра READ CAPACITY (16).

0 — Если бит PMI установлен в 0, токен устанавливает в поле RETURN LOGICAL BLOCK ADDRESS значение, меньшее из:

a) LBA последнего логического блока на устройстве блока прямого доступа;

b) FFFFFFFFh.

1 — Если бит PMI установлен в 1, токен устанавливает в поле RETURN LOGICAL BLOCK ADDRESS значение, меньшее из:

a) последнее LBA после указанной в поле LOGICAL BLOCK ADDRESS в CDB до того, как может возникнуть существенная зависящая от поставщика задержка при передаче данных;

b) LBA последнего логического блока на устройстве блока прямого доступа.

RETURNED LOGICAL BLOCK ADDRESS должен быть больше или равен адресу, указанному в поле LOGICAL BLOCK ADDRESS в CDB.

BLOCK LENGTH IN BYTES field

Поле BLOCK LENGTH IN BYTES содержит количество байтов пользовательских данных в логическом блоке, указанное в поле RETURN LOGICAL BLOCK ADDRESS. Это значение не включает информацию о защите или дополнительную информацию (например, байты ECC), записанную на носителе.

2.8. Описание команды READ FORMAT CAPACITIES (0x23)

Формат команды представлен ниже.

Таблица 13 — Формат команды READ FORMAT CAPACITIES (0x23)

Bit Byte	7	6	5	4	3	2	1	0
0	Operation Code (23h)							
1	Reserved							
2-6	Reserved							
7-8	Allocation Length							
9-11	Reserved							

Команда READ FORMAT CAPACITIES передает на хост данные о ёмкости носителя, загруженного в данный момент. Если носитель не загружен, эта команда возвращает максимальное значение ёмкости поддерживаемого носителя на хост.

2.9. Описание команды READ(10) (0x28)

Команда READ (10) запрашивает, чтобы токен прочитал указанные логические блоки и передал их в буфер ввода данных. Каждый считанный логический блок включает пользовательские данные и, если носитель отформатирован с включенной информацией о защите, информацию о защите. Каждый передаваемый логический блок включает в себя пользовательские данные и может включать в себя информацию о защите,

основанную на поле RDPROTECT и формате среды. Возвращается самое последнее значение данных, записанное в адресуемом логическом блоке.

Формат команды представлен ниже.

Таблица 14 — Формат команды READ(10) (0x28)

Bit Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (28h)							
1	RDPROTECT			DPO	FUA	RARC	Obsolete	Obsolete
2	(MSB)							
...	LOGICAL BLOCK ADDRESS							
5								(LSB)
6	Reserved			GROUP NUMBER				
7	(MSB)							
8	TRANSFER LENGTH							(LSB)
9	CONTROL							

RDPROTECT field

Токен проверяет информацию о защите, считанную с носителя, перед возвратом состояния для команды на основе поля RDPROTECT, как описано в таблице ниже.

Таблица 15 — Описание поля RDPROTECT

Код	Логический блок отформатирован с защитой информации	Должен токен защищать информацию при передаче	Поле в защищаемой информации	Расширенное значение бита страницы данных запроса VPD	Сообщения об ошибках
000b	Да	Нет	LOGICAL BLOCK GUARD	GRD_CHK = 1	LOGICAL BLOCK GUARD CHECK FAILED
				GRD_CHK = 0	NO CHECK PERFORMED
			LOGICAL BLOCK APPLICATION TAG	APP_CHK = 1	LOGICAL BLOCK APPLICATION TAG CHECK FAILED
				APP_CHK = 0	NO CHECK PERFORMED
			LOGICAL BLOCK REFERENCE TAG	REF_CHK = 1	LOGICAL BLOCK REFERENCE TAG CHECK FAILED
				REF_CHK = 0	NO CHECK PERFORMED
	Нет	Отсутствует информация о защите для проверки			
001b	Да	Да	LOGICAL BLOCK GUARD	GRD_CHK = 1	LOGICAL BLOCK GUARD CHECK FAILED
				GRD_CHK = 0	NO CHECK PERFORMED
			LOGICAL BLOCK APPLICATION TAG	APP_CHK = 1	LOGICAL BLOCK APPLICATION TAG CHECK FAILED
				APP_CHK = 0	NO CHECK PERFORMED
			LOGICAL BLOCK REFERENCE TAG	REF_CHK = 1	LOGICAL BLOCK REFERENCE TAG CHECK FAILED
				REF_CHK = 0	NO CHECK PERFORMED
	Нет	Информация о защите недоступна для передачи в буфер ввода данных или для проверки			
010b	Да	Да	LOGICAL BLOCK GUARD	NO CHECK PERFORMED	
			LOGICAL BLOCK APPLICATION TAG	APP_CHK = 1	LOGICAL BLOCK APPLICATION TAG CHECK FAILED
				APP_CHK = 0	NO CHECK PERFORMED
			LOGICAL BLOCK REFERENCE TAG	REF_CHK = 1	LOGICAL BLOCK REFERENCE TAG CHECK FAILED
				REF_CHK = 0	NO CHECK PERFORMED

Код	Логический блок отформатирован с защитой информации	Должен токен защищать информацию при передаче	Поле в защищаемой информации	Расширенное значение бита страницы данных запроса VPD	Сообщения об ошибках
	Нет	Информация о защите недоступна для передачи в буфер ввода данных или для проверки			
011b	Да	Да	LOGICAL BLOCK GUARD	NO CHECK PERFORMED	
			LOGICAL BLOCK APPLICATION TAG	NO CHECK PERFORMED	
			LOGICAL BLOCK REFERENCE TAG	NO CHECK PERFORMED	
	Нет	Информация о защите недоступна для передачи в буфер ввода данных или для проверки			
100b	Да	Да	LOGICAL BLOCK GUARD	GRD_CHK = 1	LOGICAL BLOCK GUARD CHECK FAILED
				GRD_CHK = 0	NO CHECK PERFORMED
			LOGICAL BLOCK APPLICATION TAG	NO CHECK PERFORMED	
			LOGICAL BLOCK REFERENCE TAG	NO CHECK PERFORMED	
	Нет	Информация о защите недоступна для передачи в буфер ввода данных или для проверки			
101b-111b	Зарезервировано				

DPO (Disable Page Out) bit

0 — Бит Disable Page Out (DPO), установленный в 0, указывает, что приоритет хранения должен определяться полями RETENTION PRIORITY на странице режима кэширования.

1 — Бит DPO, установленный в 1, указывает, что токен назначает логическим блокам, к которым обращается эта команда, самый низкий приоритет хранения для выборки или сохранения в кэше. Бит DPO, установленный в 1, переопределяет любой приоритет хранения, указанный на странице режима кэширования.

FUA bit

0 — Бит Force unit Access (FUA), установленный в 0, указывает, что токен считывает логические блоки из энергозависимого кэша (если таковой имеется), заданный шаблон данных для этого LBA (например, шаблон данных для не сопоставленных данных), энергонезависимого кэша или носителя.

1 — Бит FUA, установленный в 1, указывает, что токен считывает логические блоки из указанного шаблона данных для этой LBA, энергонезависимой кэш-памяти (если таковая имеется) или среды. Если энергозависимая кэш-память содержит более позднюю версию логического блока, то токен записывает этот логический блок в энергонезависимую кэш-память или носитель перед считыванием логического блока.

RARC bit

Если режим помощи при перестроении не поддерживается и бит RARC установлен в 1, то токен завершает команду со статусом CHECK CONDITION с ключом считывания, установленным в значение ILLEGAL REQUEST, и дополнительным кодом считывания, установленным в значение INVALID FIELD IN CDB.

LOGICAL BLOCK ADDRESS field

Поле LOGICAL BLOCK ADDRESS указывает первый логический блок, к которому обращается эта команда. Если адрес логического блока превышает емкость носителя, токен завершает команду со статусом CHECK CONDITION с ключом считывания, установленным в значение ILLEGAL REQUEST, и дополнительным кодом считывания, установленным в значение LOGICAL BLOCK ADDRESS OUT OF RANGE.

GROUP NUMBER field

Поле GROUP NUMBER указывает группу, в которую должны быть собраны атрибуты, связанные с командой. Нулевое значение в поле GROUP NUMBER указывает, что любые атрибуты, связанные с командой, не должны собираться в какую-либо группу.

TRANSFER LENGTH field

Поле TRANSFER LENGTH указывает количество смежных логических блоков данных, которые должны быть считаны и переданы в буфер ввода данных, начиная с логического блока, заданного полем LOGICAL BLOCK ADDRESS. Поле TRANSFER LENGTH, установленное в 0, указывает, что никакие логические блоки не должны считываться. Данное условие не считается ошибкой. Любое другое значение указывает количество логических блоков, которые должны быть считаны. Если адрес логического блока плюс длина передачи превышает емкость носителя, токен должен завершить команду со статусом CHECK CONDITION с ключом считывания, установленным в значение ILLEGAL REQUEST, и дополнительным кодом считывания, установленным в значение LOGICAL BLOCK ADDRESS OUT OF RANGE. Поле TRANSFER LENGTH ограничено полем MAXIMUM TRANSFER LENGTH на странице VPD Block Limits.

2.10.Описание команды WRITE(10) (0x2A)

Команда WRITE (10) требует, чтобы токен передал указанный логический блок(-и) из буфера вывода данных и записал их. Каждый передаваемый логический блок включает в себя пользовательские данные и может включать в себя информацию о защите на основе поля WRPROTECT и формата носителя. Каждый записанный логический блок включает пользовательские данные и, если носитель отформатирован с включенной информацией о защите, информацию о защите.

Формат команды представлен ниже.

Таблица 16 — Формат команды WRITE(10) (0x2A)

Bit Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (2Ah)							
1	WRPROTECT			DPO	FUA	Reserved	Obsolete	Obsolete
2	(MSB)							
...	LOGICAL BLOCK ADDRESS							
5								(LSB)
6	Reserved			GROUP NUMBER				
7	(MSB)							
8	TRANSFER LENGTH							(LSB)
9	CONTROL							

WRPROTECT

Токен проверяет информацию о защите, передаваемую из буфера вывода данных, на основе поля WRPROTECT, как описано в таблице ниже.

Таблица 17 — Описание поля WRPROTECT

Код	Логический блок отформатирован с защитой информации	Поле в информации о защите	Проверка токена	Сообщения об ошибках
000b	Да	Нет информации о защите, полученной от клиента приложения для проверки		
	Нет	Нет информации о защите, полученной от клиента приложения для проверки		
001b	Да	LOGICAL BLOCK GUARD	Должен	LOGICAL BLOCK GUARD CHECK FAILED
		LOGICAL BLOCK APPLICATION TAG	Зависит от RWWP	LOGICAL BLOCK APPLICATION TAG CHECK FAILED
		LOGICAL BLOCK REFERENCE TAG	Должен	LOGICAL BLOCK REFERENCE TAG CHECK FAILED
	Нет	Информация о защите недоступна для проверки		
010b	Да	LOGICAL BLOCK GUARD	Не должен	Проверка не выполнена

		LOGICAL BLOCK APPLICATION TAG	Зависит от RWWP	LOGICAL BLOCK APPLICATION TAG CHECK FAILED
		LOGICAL BLOCK REFERENCE TAG	Может	LOGICAL BLOCK REFERENCE TAG CHECK FAILED
	Нет	Информация о защите недоступна для проверки		
011b	Да	LOGICAL BLOCK GUARD	Не должен	Проверка не выполнена
		LOGICAL BLOCK APPLICATION TAG	Не должен	Проверка не выполнена
		LOGICAL BLOCK REFERENCE TAG	Не должен	Проверка не выполнена
	Нет	Информация о защите недоступна для проверки		
100b	Да	LOGICAL BLOCK GUARD	Должен	LOGICAL BLOCK GUARD CHECK FAILED
		LOGICAL BLOCK APPLICATION TAG	Не должен	Проверка не выполнена
		LOGICAL BLOCK REFERENCE TAG	Не должен	Проверка не выполнена
	Нет	Информация о защите недоступна для проверки		
101b	Да	LOGICAL BLOCK GUARD	Должен	LOGICAL BLOCK GUARD CHECK FAILED
		LOGICAL BLOCK APPLICATION TAG	Зависит от RWWP	LOGICAL BLOCK APPLICATION TAG CHECK FAILED
		LOGICAL BLOCK REFERENCE TAG	Может	LOGICAL BLOCK REFERENCE TAG CHECK FAILED
	Нет	Информация о защите недоступна для проверки		
110b-111b	Зарезервировано			

FUA bit

0 — Бит Force unit Access (FUA), установленный в 0, указывает, что токен должен записывать логические блоки в энергозависимый кэш (если таковой имеется), энергонезависимый кэш (если таковой имеется) или носитель.

1 — Бит FUA, установленный в 1, указывает, что токен должен записывать логические блоки в энергонезависимый кэш (если таковой имеется) или носитель.

Если логические блоки передаются непосредственно в кэш, токен может вернуть состояние GOOD до записи логических блоков на носитель. Любая ошибка, возникающая после возврата статуса GOOD, является отложенной ошибкой, и информация об ошибке не сообщается до тех пор, пока не будет получена следующая команда.

TRANSFER LENGTH field

Поле TRANSFER LENGTH указывает количество смежных логических блоков данных, которые должны быть переданы из буфера вывода данных и записаны, начиная с логического блока, заданного полем LOGICAL BLOCK ADDRESS. Нулевое значение в поле TRANSFER LENGTH указывает, что логические блоки не должны записываться. Данное условие не считается ошибкой. Любое другое значение указывает количество логических блоков, которые должны быть записаны. Если адрес логического блока плюс длина передачи превышает емкость носителя, токен должен завершить команду со статусом CHECK CONDITION с ключом считывания, установленным в значение ILLEGAL REQUEST, и дополнительным кодом считывания, установленным в значение LOGICAL BLOCK ADDRESS OUT OF RANGE. Поле TRANSFER LENGTH ограничено полем MAXIMUM TRANSFER LENGTH на странице VPD Block Limits.

2.11.Описание команды VERIFY(10) (0x2F)

Команда VERIFY (10) требует, чтобы токен проверил указанный логический блок (блоки) на носителе. Каждый логический блок включает в себя пользовательские данные и может включать в себя информацию о защите на основе поля VRPROTECT и формата среды.

Формат команды представлен ниже.

Таблица 18 — Формат команды VERIFY(10) (0x2F)

Bit Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (2Fh)							
1	VRPROTECT			DPO	Reserved	BYTCHK		Obsolete
2	(MSB)							
...	LOGICAL BLOCK ADDRESS							
5								(LSB)
6	RESTRICTED FOR MMC-5	Reserved		GROUP NUMBER				
7	(MSB)							
8	VERIFICATION LENGTH							(LSB)
9	CONTROL							

Логические блоки, содержащие кэш, должны записывать указанные кэшированные логические блоки на носитель для логического блока (например, как они делали бы в ответ на команду SYNCHRONIZE CACHE с битом SYNC_NV, установленным в 0, полем LOGICAL BLOCK ADDRESS, установленным в значение поля LOGICAL BLOCK ADDRESS команды VERIFY, и полем NUMBER OF BLOCKS, установленным в значение поля VERIFICATION LENGTH команды VERIFY).

GROUP NUMBER field

Поле GROUP NUMBER указывает группу, в которую должны быть собраны атрибуты, связанные с командой. Нулевое значение в поле GROUP NUMBER указывает, что любые атрибуты, связанные с командой, не должны собираться в какую-либо группу.

Если реализована страница Verify Error Recovery mode, то текущие настройки на этой странице определяют критерии проверки. Если страница Verify Error Recovery mode не реализована, то критерии проверки зависят от поставщика.

BYTCHK field

Если в поле BYTCHK установлено значение 00b, то:

a) передача данных из буфера не происходит;

b) для любого сопоставленного LBA, указанного командой, токен должен проверить информацию о защите из операции верификации на основе поля VRPROTECT, как определено в таблице 208 [1];

c) для любого не сопоставленного LBA, указанного командой, операция проверки должна быть завершена без ошибок.

Если:

a) в поле BYTCHK установлено значение 01b или 11b;

b) бит VBULS устанавливается в 0 на странице Block Device Characteristics VPD;

c) любой LBA, указанный командой, является не сопоставленным (т.е. освобожденным или закрепленным).

Токен завершает команду со статусом CHECK CONDITION с ключом считывания, установленным в MISCOMPARE, и дополнительным кодом считывания, установленным в MISCOMPARE VERIFY OF UNMAPPED LBA.

Если:

a) в поле BYTCHK установлено значение 01b или 11b;

b) а также:

   a. бит VBULS устанавливается в 1 на странице VPD Block Device Characteristics;

b. все LBA, указанные командой, сопоставлены.

Тогда:

a) если в поле BYTCHK установлено значение 01b, то передача буфера вывода данных должна включать количество логических блоков, указанное в поле VERIFICATION LENGTH;

b) если в поле BYTCHK установлено значение 11b, то:

   a. передача буфера вывода данных должна включать один логический блок;

   b. токен должен:

      i. дублировать единственный логический блок, как описано в команде WRITE SAME, количество раз, необходимое для удовлетворения требованиям поля VERIFICATION LENGTH;

      ii. поместить дублированные данные в буфер вывода данных.

c) токен должен проверить информацию о защите, переданную из буфера вывода данных на основе поля VRPROTECT, как определено в таблице 210 [1];

d) для любого сопоставленного LBA, указанного командой, токен должен выполнить операцию проверки и проверить информацию о защите из операции проверки на основе поля VRPROTECT, как определено в таблице 209 [1];

e) токен должен выполнять:

   a. операцию сравнения:

      i. данных пользователя из операций проверки;

      ii. пользовательских данных из буфера вывода данных.

   b. операция сравнения на основе поля VRPROTECT, как определено в таблице 211 [1]:

      i. информация о защите от операций проверки;

      ii. информация о защите из буфера вывода данных.

Если побайтовое сравнение по какой-либо причине невозможно, токен завершает команду со статусом CHECK CONDITION с ключом считывания, установленным в MISCOMPARE, и дополнительным кодом считывания, установленным в соответствующее значение для условия.

VERIFICATION LENGTH field

Поле VERIFICATION LENGTH указывает количество смежных логических блоков, которые должны быть верифицированы, начиная с логического блока, заданного полем LOGICAL BLOCK ADDRESS. Если поле BYTCHK установлено равным 1, поле VERIFICATION LENGTH также указывает количество логических блоков, которые токен должен передать из буфера вывода данных. Нулевое значение в поле VERIFICATION LENGTH указывает, что никакие логические блоки не должны проверяться. Это условие не должно рассматриваться как ошибка. Любое другое значение указывает количество логических блоков, которые должны быть проверены. Если адрес логического блока плюс длина верификации превышает емкость носителя, токен завершает команду со статусом CHECK CONDITION со значением ключа считывания ILLEGAL REQUEST и дополнительным кодом считывания LOGICAL BLOCK ADDRESS OUT OF RANGE. Поле VERIFICATION LENGTH ограничено полем MAXIMUM TRANSFER LENGTH на странице VPD Block Limits.

3. Дополнительные стандартные команды

В встроенном программном средстве Aladdin eFlash реализованы следующие дополнительные стандартные команды:

Таблица 19 — Перечень поддерживаемых дополнительных стандартных SCSI-команд

Команда	Код
FORMAT UNIT	(0x04)
MAINTENANCE IN	(0xA3)
SERVICE ACTION IN(16)	(0x9E)
MODE SELECT(6)	(0x15)
MODE SELECT(10)	(0x55)
SYNCHRONIZE CACHE(10)	(0x35)
MODE SENCE(10)	(0x5A)
READ(6)	(0x08)
WRITE(6)	(0x0A)

3.1. Описание команды FORMAT UNIT (0x04)

Команда FORMAT UNIT делает запрос на форматирование у токена доступные клиенту приложения логические блоки, как указано в количестве блоков и значениях длины блоков, полученных в дескрипторе блока параметра последнего режима в команде MODE SELECT. Кроме того, токен может сертифицировать носитель и создать структуры управления для управления носителем и дефектами.

Если токен получает команду FORMAT UNIT перед приёмом команды MODE SELECT с дескриптором блока параметров режима, токен должен использовать количество блоков и длину блока, при которой логический блок в настоящее время отформатирован.

Формат команды представлен ниже.

Таблица 20 — Формат команды FORMAT UNIT (0x04)

Bit Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (04h)							
1	FMTPINFO		LONG-LIST	FMTDATA	CMPLIST	DEFECT LIST FORMAT		
2	Vendor Specific							
3	Reserved							
4	Reserved							FFMT
5	CONTROL							

При выполнении операции форматирования токен должен отвечать на команды, пытающиеся войти в набор задач, за исключением команд INQUIRY, REPORT LUNS и REQUEST SENSE со статусом CHECK CONDITION с ключом считывания, установленным в состояние NOT READY, и дополнительным кодом считывания, установленным в состояние LOGICAL UNIT NOT REREADY, FORMAT IN PROGRESS. Если токен получает команду INQUIRY, команду REPORT LUNS или команду REQUEST SENSE, то он должен обработать команду. Токен возвращает данные для команды INQUIRY на основе состояния целевого устройства SCSI перед началом выполнения команды FORMAT UNIT.

FMTPINFO (Format Protection Information) field

В поле format protection information (FMTPINFO) в сочетании с полем PROTECTION FIELD USAGE указывается, должен ли токен включать или выключать использование информации о защите.

Когда информация о защите записывается во время выполнения команды FORMAT UNIT (т. е. бит FMTPINFO установлен в 1), информация о защите должна записываться со значением по умолчанию FFFFFFFF_FFFFFFFFh.

LONGLIST bit

Если бит FMTDATA установлен в 0, бит LONGLIST должен игнорироваться.

0 – Бит LONGLIST, установленный в 0, указывает, что список параметров, если таковой имеется, содержит заголовок короткого списка параметров, как определено в таблице 39 [1].

1 — Бит LONGLIST, установленный в 1, указывает, что список параметров, если таковой имеется, содержит длинный заголовок списка параметров, как определено в таблице 40 [1].

FMTDATA (Format Data)

0 — Бит данных формата (FMTDATA), установленный в 0, указывает, что список параметров не должен передаваться из буфера вывода данных.

1 — Бит FMTDATA, установленный в 1, указывает, что список параметров FORMAT UNIT должен быть передан из буфера вывода данных. Список параметров состоит из заголовка списка параметров, за которым следует необязательный дескриптор шаблона инициализации, за которым следует необязательный список дефектов.

CMPLST (Complete List)

Если бит FMTDATA установлен в 0, бит CMPLIST должен игнорироваться.

0 — Бит полного списка (CMPLST), установленный в 0, указывает, что список дефектов, включенный в список параметров FORMAT UNIT, должен использоваться в дополнение к существующему списку дефектов. В результате токен создаёт новый GLIST, содержащий:

a) существующий GLIST;

b) DLIST, если он отправлен клиентом приложения;

c) CLIST, если сертификация разрешена (то есть токен может добавить любые дефекты, которые он обнаруживает во время операции форматирования).

1 — Бит CMPLST, установленный в 1, указывает, что список дефектов, включенный в список параметров FORMAT UNIT, является полным списком дефектов. Любой существующий GLIST должен быть отброшен токеном. В результате токен создаёт новый GLIST, содержащий:

a) DLIST, если он отправлен клиентом приложения;

b) CLIST, если сертификация разрешена.

DEFECT LIST FORMAT field

Поле DEFECT LIST FORMAT указывает формат адресных дескрипторов в списке дефектов, если бит FMTDATA установлен в 1 (см. таблицу ниже).

Таблица 21 — Описание поля DEFECT LIST FORMAT

Поле в CDB FORMAT UNIT			Поле DEFECT LIST LENGTH в заголовке списка параметров	Тип (a)	Комментарий (f)
FMTDATA	CMPLST	DEFECT LIST FORMAT			
0	Любой	000b	Не доступно	M	Информация о дефекте поставщика
1	0	000b (короткий блок)	0	O	Прил. b, d
	1			O	Прил. b, e
	0		Не 0	O	Прил. c, d
	1			O	Прил. c, e
1	0	001b (длинный блок)	0	O	Прил. b, d
	1			O	Прил. b, e
	0		Не 0	O	Прил. c, d
	1			O	Прил. c, e
1	0	010b (длинный блок)	0	O	Прил. b, d
	1			O	Прил. b, e
	0		Не 0	O	Прил. c, d
	1			O	Прил. c, e
1	0	011b (длинный блок)	0	O	Прил. b, d
	1			O	Прил. b, e
	0		Не 0	O	Прил. c, d
	1			O	Прил. c, e
1	0	100b (байт из индекса)	0	O	Прил. b, d
	1			O	Прил. b, e
	0		Не 0	O	Прил. c, d
	1			O	Прил. c, e
1	0	101b (физический сектор)	0	O	Прил. b, d
	1			O	Прил. b, e
	0		Не 0	O	Прил. c, d
	1			O	Прил. c, e
1	0	110b (для конкретного поставщика)	Для конкретного поставщика	O	
	1			O	
Все остальные				Зарезервировано	

a) M = реализация обязательна, O = реализация необязательна.

b) DLIST не включен в список параметров.

c) Список DLIST включен в список параметров. Токен добавляет дефекты DLIST к новому GLIST.

d) Токен добавляет существующие дефекты GLIST к новому GLIST (т.е. использует существующий GLIST).

e) Токен не должен добавлять существующие дефекты GLIST к новому GLIST (т.е. отбрасывать существующий GLIST).

f) Все опции, описанные в этой таблице, приводят к созданию нового GLIST во время обработки команды FORMAT UNIT, как описано в тексте.

FFMT field

Поле быстрого формата (FFMT) описано в таблице ниже.

Таблица 22 — Описание поля FFMT

Код	Описание	Поддержка
00b	Токен инициализирует носитель, как указано в CDB и списке параметров, перед завершением операции форматирования.	Обязательно
01b	Токен инициализирует носитель без перезаписи носителя (т.е. ресурсы для управления доступом к носителю инициализируются, а носитель не записывается) до завершения операции форматирования. Если токен определяет, что параметры, указанные в этой команде FORMAT UNIT, несовместимы с командой read и проверяет требования к команде, а токен не выполняет операцию форматирования и завершает команду FORMAT UNIT со статусом CHECK CONDITION с ключом считывания, установленным в значение ILLEGAL REQUEST, и дополнительным кодом считывания, установленным в значение INVALID FAST FORMAT COMINATION.	Опционально
10b	Токен инициализирует носитель без перезаписи носителя (т.е. ресурсы для управления доступом к носителю инициализируются, а носитель не записывается) до завершения операции форматирования.	Опционально
11b	Зарезервировано	

CONTROL byte

Байт CONTROL определен в пункте 2.2.7 [1].

3.2. Описание команды SERVICE ACTION IN(16) (0x9E)

Некоторые команды реализованы в виде специфичных для конкретного устройства сервисных действий для 9Eh кода операции SERVICE ACTION IN (16). Эти команды различаются по кодам действий сервиса, как описано в таблице ниже.

Таблица 23 — Перечень поддерживаемых дополнительных команд для SERVICE ACTION IN(16) (0x9E)

Код команды	Описание
9Eh/15h	BACKGROUND CONTROL
9Eh/12h	GET LBA STATUS
9Eh/10h	READ CAPACITY (16)
9Eh/11h	READ LONG (16)
9Eh/14h	STREAM CONTROL

3.3. Описание команды MODE SELECT(6) (0x15)

Команда MODE SELECT (6) предоставляет клиенту приложения возможность задать параметры среды, логического устройства или периферийного устройства для токена. Клиенты приложения должны выдавать MODE SENSE (6) перед каждым MODE SELECT (6) для определения поддерживаемых страниц режима, длин страниц и других параметров.

Формат команды представлен ниже.

Таблица 24 — Формат команды MODE SELECT(6) (0x15)

Bit Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (15h)							
1	Reserved			PF	Reserved		RTD	SP
2	Reserved							
3								
4	PARAMETER LIST LENGTH							
5	CONTROL							

Логические блоки должны совместно использовать значения заголовков параметров режима и дескрипторов блоков во всех I_T связях. Потеря I_T связи не должна влиять на заголовок параметра режима, дескриптор блока и значения страницы режима.

Логические блоки должны поддерживать текущие и сохраненные значения каждой страницы режима на основе любой из политик, перечисленных в таблице ниже.

Таблица 25 — Перечень поддерживаемых политик

Политика страницы режима	Количество копий страницы режима
Общая	Одна копия страницы режима, совместно используемая всеми I_T связями
На каждый целевой порт	Отдельная копия страницы режима для каждого целевого порта, при этом каждая копия совместно используется всеми портами инициатора
На каждую I_T связь	Отдельная копия страницы режима для каждой I_T связи

После сброса логического блока каждый заголовок параметра режима, дескриптор блока и страница режима должны вернуться к сохраненным значениям, если поддерживаются, или к значениям по умолчанию, если сохраненные значения не поддерживаются.

Если клиент приложения отправляет команду MODE SELECT, которая изменяет какие-либо параметры, применяемые к другим I_T связям, токен устанавливает пометку для порта инициатора, связанного с каждой I_T связью, за исключением I_T связи, на которой была получена команда MODE SELECT, с дополнительным кодом считывания, установленным в MODE PARAMETERS CHANGED.

PF (Page Format) bit

0 — Бит формата страницы (PF), установленный в 0, указывает, что все параметры после дескрипторов блоков зависят от поставщика.

1 — Бит PF, установленный в 1, указывает, что параметры MODE SELECT, следующие за заголовком и дескриптором(-ами) блока, структурированы как страницы связанных параметров и соответствуют определенным в [1]. Если бит RTD установлен в 1, то бит PF игнорируется.

RTD (revert to defaults) bit

0 — Бит RTD, установленный в 0, указывает, что токен обрабатывает команду MODE SELECT на основе других полей в CDB и данных параметров.

1 — Бит RTD_SUP, установленный в 1, и бит RTD, установленный в 1, указывают на то, что токен должен вернуть:

a) значения страниц текущего режима для всех страниц режима соответствуют значениям по умолчанию, если бит SP установлен в 0;

b) значения страниц текущего режима и сохраненные значения страниц режима для всех страниц до значений по умолчанию, если бит SP установлен в 1.

SP (save pages) bit

0 — Бит сохранения страниц (SP), установленный в 0, указывает, что токен должен выполнить указанную операцию MODE SELECT и не должен сохранять страницы режима. Если логический блок не проводит различия между текущими и сохраненными страницами режима, а бит SP установлен в 0, команда завершается со статусом CHECK CONDITION, с ключом считывания, установленным в значение ILLEGAL REQUEST, и дополнительным кодом считывания, установленным в значение INVALID FIELD IN CDB. Бит SP, установленный в 1, указывает, что токен должен выполнить указанную операцию MODE-SELECT и сохранить в энергонезависимом местоположении поставщика все сохраняемые страницы режима, включая все отправленные в буфер Data-Out. Сохраняемые страницы режима определяются битом сохранения параметра (PS), который возвращается в первом байте каждой страницы режима командой MODE SENSE.

1 — Если бит PS установлен в 1 данных MODE SENSE, то страница режима должна быть сохраняемой путем выдачи команды MODE SELECT с битом SP, установленным в 1. Если логический блок не реализует сохраненные страницы режима и бит SP установлен в 1, то команда должна быть завершена со статусом CHECK CONDITION, с ключом считывания, установленным в значение ILLEGAL REQUEST, и дополнительным кодом считывания, установленным в значение INVALID FIELD IN CDB.

PARAMETER LIST LENGTH field

Поле PARAMETER LIST LENGTH указывает длину в байтах списка параметров режима, который должен содержаться в буфере вывода данных. Нулевая длина списка параметров указывает на то, что буфер вывода данных должен быть пустым. Это условие не должно рассматриваться как ошибка.

Если длина списка параметров приводит к усечению какого-либо заголовка параметра режима, дескриптора(-ов) блока параметров режима или страницы режима, то команда должна быть завершена со статусом CHECK CONDITION, с ключом считывания, установленным в значение ILLEGAL REQUEST, и дополнительным кодом считывания, установленным в значение PARAMETER LIST LENGTH ERROR.

Токен завершает команду MODE SELECT со статусом CHECK CONDITION, устанавливает ключ считывания в значение ILLEGAL REQUEST, устанавливает дополнительный код считывания в значение INVALID FIELD IN PARAMETER LIST и не должен изменять параметры режима в ответ на любое из следующих условий:

a) если клиент приложения устанавливает любое поле, о котором сообщается, что оно не может быть изменено токеном, в значение, отличное от его текущего значения;

b) если клиент приложения устанавливает неподдерживаемое значение для любого поля в заголовке параметра mode или дескрипторе(-ах) блока;

c) если клиент приложения отправляет страницу режима с длиной страницы, не равной длине страницы, возвращаемой командой MODE SENSE для этой страницы режима;

d) если клиент приложения отправляет неподдерживаемое значение для параметра mode, а округление для этого параметра mode не реализовано;

e) если клиент приложения устанавливает любое зарезервированное поле в списке параметров режима в ненулевое значение и токен проверяет зарезервированные поля.

Если клиент приложения отправляет значение параметра режима, выходящего за пределы диапазона, поддерживаемого токеном, и для этого параметра режима реализовано округление, токен обрабатывает условие одним из следующих способов:

a) Округление параметра до приемлемого значения и завершение команды, как описано в разделе 2.3 [1].

b) Завершение команды со статусом CHECK CONDITION, при этом для ключа считывания установлено значение ILLEGAL REQUEST, а для кода дополнительного считывания установлено значение INVALID FIELD IN PARAMETER LIST.

Токен может изменять любой параметр режима на любой странице режима, даже те, которые указаны как неизменяемые, в результате изменений других параметров режима.

Токен проверяет неизменяемые параметры режима на соответствие текущим значениям, которые существовали для этих параметров режима до команды MODE SELECT.

3.4. Описание команды MODE SELECT(10) (0x55)

Команда MODE SELECT (10) предоставляет клиенту приложения возможность задать параметры среды, логического устройства или периферийного устройства для токена. Описание полей и действия этой команды аналогично команде MODE SELECT (6). Клиенты приложения должны выдавать MODE SENSE (10) перед каждым MODE SELECT (10) для определения поддерживаемых страниц режима, длин страниц и других параметров. Токены, реализующие команду MODE SELECT (10), также должны реализовывать команду MODE SENSE (10).

Формат команды представлен ниже.

Таблица 26 — Формат команды MODE SELECT(10) (0x55)

Bit Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (55h)							
1	Reserved			PF	Reserved			SP
2	Reserved							
...								
6								
7	(MSB)	PARAMETER LIST LENGTH						
8								(LSB)
9	CONTROL							

3.5. Описание команды SYNCHRONIZE CACHE(10) (0x35)

Команда SYNCHRONIZE CACHE (10) требует, чтобы токен обеспечивал запись в энергонезависимой кэш-памяти и/или на носителе самых последних значений данных для указанных логических блоков. Логические блоки включают пользовательские данные и, если носитель отформатирован с включенной информацией о защите, информацию о защите. Логические блоки могут быть удалены или не удалены из энергозависимого кэша и энергонезависимого кэша в результате операции синхронизации кэша.

Формат команды представлен ниже.

Таблица 27 — Формат команды SYNCHRONIZE CACHE(10) (0x35)

Bit Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (35h)							
1	Reserved					Obsolete	IMMED	Obsolete
2	(MSB)							
...	LOGICAL BLOCK ADDRESS							
5								(LSB)
6	Reserved			GROUP NUMBER				
7	(MSB)							
8	NUMBER OF BLOCKS							(LSB)
9	CONTROL							

IMMED (Immediate) bit

0 — Бит IMMED, установленный в 0, указывает, что токен не должен возвращать статус до завершения операции синхронизации кэша.

1 — Бит IMMED, установленный в 1, указывает, что токен должен вернуть статус, как только CDB будет проверен. Если бит IMMED установлен в 1 и токен не поддерживает бит IMMED, токен должен завершить команду со статусом CHECK CONDITION с ключом считывания, установленным в ILLEGAL REQUEST, и дополнительным кодом считывания, установленным в INVALID FIELD IN CDB.

Если бит IMMED установлен в 1 и операция синхронизации кэша не завершена, то поле SYNC_PROG на странице режима кэширования определяет поведение токена в соответствии с таблицей, представленной ниже.

Таблица 28 — Описание поля IMMED

Код	Описание
00b	Токен не должен завершать команды из-за операции синхронизации кэша и не должен предоставлять загрязняющие (мусорные) данные
01b	Токен: a) не должен завершать команды из-за операции синхронизации кэша; b) должен предоставлять загрязненные данные считывания с ключом считывания, установленным в NO SENSE, дополнительным кодом считывания, установленным для SYNCHRONIZE CACHE OPERATION IN PROGRESS, и полем PROGRESS INDICATION, установленным для индикации хода выполнения операции синхронизации кэша.
10b	Токен: a) должен обрабатывать команды INQUIRY, REPORT LUNS, REPORT TARGET PORT GROUPS и REQUEST SENSE; b) может обрабатывать команды, не требующие ресурсов, используемых для операции синхронизации кэша; c) должен завершать команды, для которых требуются ресурсы, используемые для операции синхронизации кэша со статусом CHECK CONDITION, при этом для ключа считывания установлено значение NOT READY, для кода дополнительного считывания установлено значение LOGICAL UNIT NOT READY, SYNCHRONIZE CACHE OPERATION IN PROGRESS, а в поле PROGRESS INDICATION указывается ход выполнения операции синхронизации кэша;

	d) должен предоставлять загрязненные (мусорные) данные считывания с ключом считывания, установленным в состояние NOT READY, дополнительным кодом считывания, установленным в состояние LOGICAL UNIT NOT READY, SYNCHRONIZE CACHE OPERATION IN PROGRESS, и полем PROGRESS INDICATION для индикации хода выполнения операции синхронизации кэша.
11b	Зарезервировано

GROUP NUMBER field

Поле GROUP NUMBER указывает группу, в которую должны быть собраны атрибуты, связанные с командой. Нулевое значение в поле GROUP NUMBER указывает, что любые атрибуты, связанные с командой, не должны собираться в какую-либо группу.

NUMBER OF BLOCKS field

Поле NUMBER OF BLOCKS указывает количество логических блоков, которые должны быть синхронизированы, начиная с логического блока, заданного полем LOGICAL BLOCK ADDRESS. Нулевое значение в поле NUMBER OF BLOCKS указывает, что все логические блоки, начиная с указанного в поле LOGICAL BLOCK ADDRESS до последнего логического блока на носителе, должны быть синхронизированы. Если адрес логического блока плюс количество блоков превышает емкость носителя, токен завершает команду со статусом CHECK CONDITION с ключом считывания, установленным в значение ILLEGAL REQUEST, и дополнительным кодом считывания, установленным в значение LOGICAL BLOCK ADDRESS OUT OF RANGE.

Логический блок в пределах диапазона, который не находится в кэше, не считается ошибкой.

3.6. Описание команды MODE SENSE(10) (0x5A)

Команда MODE SENSE (10) позволяет токену передавать параметры клиенту приложения. Эта команда дополняет команду MODE SELECT (10).

Формат команды представлен ниже.

Таблица 29 — Формат команды MODE SENSE(10) (0x5A)

Bit Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (5Ah)							
1	Reserved			LLBAA	DBD	Reserved		
2	PC		PAGE CODE					
3	SUBPAGE CODE							
4	Reserved							
...								
6								
7	(MSB)	ALLOCATION LENGTH						
8								(LSB)
9	CONTROL							

LLBAA (Long LBA Accepted) bit

1 — Если бит Long LBA Accepted (LLBAA) установлен в 1, токен может возвращать данные параметров с битом LONGLBA, равным единице.

0 — Если бит LLBAA установлен в 0, бит LONGLBA должен быть нулевым в данных параметров, возвращаемых токеном.

Описание остальных полей представлено в описании MODE SENSE(6).

3.7. Описание команды READ(6) (0x08)

Команда READ(6) требует, чтобы токен прочитал указанный логический блок (блоки) и передал их в буфер ввода данных. Каждый считанный логический блок включает пользовательские данные и, если носитель отформатирован с включенной информацией о защите, информацию о защите. Каждый передаваемый логический блок включает пользовательские данные, но не включает информацию о защите. Должно быть возвращено последнее значение данных, записанное или подлежащее записи в случае кэширования в адресуемые логические блоки.

Формат команды представлен ниже.

Таблица 30 — Формат команды READ(6) (0x08)

Bit Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (08h)							
1	Reserved			(MSB)				
2	LOGICAL BLOCK ADDRESS							
3								(LSB)
4	TRANSFER LENGTH							
5	CONTROL							

Для этой команды не предусмотрены биты управления кэшем. Устройства блока прямого доступа с кэшем могут иметь значения для битов управления кэшем, которые влияют на команду READ (6). Если требуется явное управление, следует использовать команду READ (10).

LOGICAL BLOCK ADDRESS field

Поле LOGICAL BLOCK ADDRESS указывает первый логический блок, к которому обращается эта команда. Если адрес логического блока превышает емкость носителя, токен завершает команду со статусом CHECK CONDITION с ключом считывания, установленным в значение ILLEGAL REQUEST, и дополнительным кодом считывания, установленным в значение LOGICAL BLOCK ADDRESS OUT OF RANGE.

TRANSFER LENGTH field

Поле TRANSFER LENGTH указывает количество смежных логических блоков данных, которые должны быть считаны и переданы в буфер ввода данных, начиная с логического блока, заданного полем LOGICAL BLOCK ADDRESS. Нулевое значение в поле TRANSFER LENGTH указывает, что должно быть считано 256 логических блоков. Любое другое значение указывает количество логических блоков, которые должны быть считаны. Если адрес логического блока плюс длина передачи превышает емкость носителя, токен завершает команду со статусом CHECK CONDITION и ключом считывания, установленным в значение ILLEGAL REQUEST, и дополнительным кодом считывания, установленным в значение LOGICAL BLOCK ADDRESS OUT OF RANGE. Поле TRANSFER LENGTH ограничено полем MAXIMUM TRANSFER LENGTH на странице VPD Block Limits:

a) Для команд READ (10), READ (12), READ (16) и READ (32) нулевое значение поля TRANSFER LENGTH указывает, что логические блоки не считываются.

b) Хотя команда READ (6) ограничена адресацией логических блоков емкостью до 2 Гигабайт, для блоков длиной 512 байт эта команда сохраняется, поскольку некоторые процедуры инициализации системы требуют использования команды READ (6). Процедуры инициализации системы должны мигрировать из команды READ (6) в команду READ (10), которая способна адресовать 2 терабайта с длиной блока 512 байт, или команду READ (16) для адресации более 2 терабайт.

Перед возвратом состояния для команды токен проверяет информацию о защите, считанную с носителя, как описано в таблице ниже.

Таблица 31 — Описание поля TRANSFER LENGTH

Логический блок, отформатированный с информацией о защите	Должен ли токен передавать информацию о защите?	Поле в информации о защите (e)	Расширенное значение бита страницы данных запроса VPD (d)	Если проверка завершается неуспешно (b), (c), дополнительный код считывания
Да	Нет	LOGICAL BLOCK GUARD	GRD_CHK = 1	LOGICAL BLOCK GUARD CHECK FAILED
			GRD_CHK = 0	Проверка не выполняется
		LOGICAL BLOCK APPLICATION TAG	APP_CHK = 1 (a)	LOGICAL BLOCK APPLICATION TAG CHECK FAILED
			APP_CHK = 0	Проверка не выполняется
		LOGICAL BLOCK REFERENCE TAG	REF_CHK = 1 (f), (g)	LOGICAL BLOCK REFERENCE TAG CHECK FAILED
			REF_CHK = 0	Проверка не выполняется
Нет		Информация о защите недоступна для проверки		

a) Токен проверяет тег приложения логического блока только в том случае, если ему известно о содержимом поля LOGICAL BLOCK APPLICATION TAG.

b) Если выдается сообщение об ошибке, для сенсорной клавиши должно быть установлено значение ABORTED COMMAND.

c) При возникновении нескольких ошибок выбор ошибки для отчета не определен.

d) Определения битов GRD_CHK, APP_CHK и REF_CHK представлены в VPD Extended INQUIRY Data.

e) Если токен обнаруживает:

I. В поле LOGICAL BLOCK APPLICATION TAG установлено значение FFFFh и включена защита типа 1 или защита типа 2.

или

II. Для поля LOGICAL BLOCK APPLICATION TAG установлено значение FFFFh, для поля LOGICAL BLOCK REFERENCE TAG установлено значение FFFF FFFFh, и включена защита типа 3.

Далее токен не проверяет информацию о защите в связанном логическом блоке.

f) Если защита типа 1 включена, токен проверяет эталонный тег логического блока, сравнивая его с младшими 4 байтами LBA, связанными с логическим блоком.

g) Если включена защита типа 2 или 3, токен проверяет тег ссылки логического блока только в том случае, если ему известно о содержимом поля LOGICAL BLOCK REFERENCE TAG.

3.8. Описание команды WRITE(6) (0x0A)

Команда WRITE (6) требует, чтобы токен передал указанные логические блоки из буфера вывода данных и записал их. Каждый передаваемый логический блок включает пользовательские данные, но не включает информацию о защите. Каждый записанный логический блок включает пользовательские данные и, если носитель отформатирован с включенной информацией о защите, информацию о защите.

Формат команды представлен ниже.

Таблица 32 — Формат команды WRITE(6) (0x0A)

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (0Ah)							
1	Reserved			(MSB)				
2	LOGICAL BLOCK ADDRESS							
3								(LSB)
4	TRANSFER LENGTH							
5	CONTROL							

Для этой команды не предусмотрены биты управления кэшем. Устройства блока прямого доступа с кэшем могут иметь значения для битов управления кэшем, которые могут повлиять на команду WRITE (6),

однако значение по умолчанию не определено. Если требуется явное управление, следует использовать команду WRITE (10).

TRANSFER LENGTH field

Поле TRANSFER LENGTH указывает количество смежных логических блоков данных, которые должны быть переданы из буфера вывода данных и записаны, начиная с логического блока, заданного полем LOGICAL BLOCK ADDRESS. Нулевое значение в поле TRANSFER LENGTH указывает, что должно быть записано 256 логических блоков. Любое другое значение указывает количество логических блоков, которые должны быть записаны. Если адрес логического блока плюс длина передачи превышает емкость носителя, токен должен завершить команду со статусом CHECK CONDITION с ключом считывания, установленным в значение ILLEGAL REQUEST, и дополнительным кодом считывания, установленным в значение LOGICAL BLOCK ADDRESS OUT OF RANGE. Поле TRANSFER LENGTH ограничено полем MAXIMUM TRANSFER LENGTH на странице VPD Block Limits.

Если команда WRITE (6) получена после того, как информация о защите включена, токен должен установить информацию о защите следующим образом, когда он записывает каждый логический блок на носитель:

a) поле LOGICAL BLOCK GUARD установлено в правильно сформированный CRC;

b) в поле LOGICAL BLOCK REFERENCE TAG установлено значение:

   a. младшие четыре байта LBA, если включена защита типа 1;

   b. FFFFFFFFh, если включена защита типа 2 или 3.

c) в поле LOGICAL BLOCK APPLICATION TAG установлено значение:

   a. FFFFh, если бит ATO установлен в 1 на странице «Режим управления»;

   b. любое значение, если бит ATO установлен в 0 на странице «Режим управления».

4. Проприетарные (нестандартные)

Встроенная системная программа «ВПО Aladdin eFlash», разработанная компанией АО «Аладдин Р.Д.» имеет две проприетарные команды описание которых представлено ниже.

4.1. Краткое описание команды «Сообщить контрольную сумму встроенного МПО (прошивки)»

Команда вычисляет и возвращает на хост контрольную сумму встроенного МПО (прошивки), которая имеет размер 4 байта. Для вычисления контрольной суммы используется алгоритм CRC32 IEEE 802.3.

Формат команды (6 байт): 0xE4 00 00 00 00 00

Входные данные: нет.

Выходные данные: 4 байта (контрольная сумма прошивки), порядок следования байт - big endian.

Контрольная сумма вычисляется для следующих диапазонов адресов:

[0000 - 00df], [0100 - bfa3], [c000 - fffd]

4.2. Краткое описание команды «Отключить режим только чтение»

Команда E20000000000 «Отключить режим только чтение»:

- не может завершиться неудачно / с ошибкой;
- доступна в любом режиме съёмного носителя в составе eFlash ("только чтение", "чтение-запись");
- не требует предъявления никакой дополнительной информации (пароля, пин-кода и пр.);
- не возвращает никаких данных на хост, возвращает только контейнер состояния CSW со значением поля "bCSWStatus" равным 0x00;
- если до передачи команды E2000000000000 съёмный носитель находился в режиме "только чтение", то после выполнения команды он перейдёт в режим "чтение-запись";
- если до передачи команды E2000000000000 съёмный носитель находился в режиме "чтение-запись", то после выполнения команды он останется в режиме "чтение-запись".

Операционная система рабочей станции - Windows, Linux и пр., с той или иной степенью «фанатизма» запоминает (кэширует) информацию о состоянии съёмного носителя (в частности, является ли он защищённым от записи). Для более быстрого применения изменённых параметров (переход из режима "только чтение" в режим "чтение-запись"), рекомендуется дополнительно отправить две SCSI-команды START STOP UNIT с интервалом в 3-5 секунд между ними.

- Первая SCSI-команда START STOP UNIT (0x1B) выполнит извлечение диска из носителя.
- Интервал в 3 - 5 секунд необходим, чтобы операционная система успела получить уведомление об отсутствии диска в носителе (через регулярно отправляемые средствами ОС SCSI-команды TEST UNIT READY);
- Вторая SCSI-команда START STOP UNIT (0x1B) выполнит подключение диска к носителю.

5. Описание параметров, задаваемых в ВПО Aladdin eFlash

При кастомизации встроенной системной программы «ВПО Aladdin eFlash» могут задаваться параметры, которые делятся на два вида:

- изменяемые;
- неизменяемые.

5.1. Описание изменяемых параметров

Перечень изменяемых параметров представлен в таблице ниже.

Таблица 33 — Описание изменяемых параметров ВПО Aladdin eFlash

Наименование параметра	Обозначение	Принимаемые значения	Пример записи
Серийный номер устройства	iSerial	Для изделий указывается уникальное восьмизначное число в шестнадцатеричной кодировке. Максимум можно указать 12 символов	iSerial = '20000040'
Название устройства	iProduct	Название устройства в дескрипторе USB формируемое из двух частей: • Обозначение изделия – eFlash. • Приставка к обозначению изделия. В зависимости от исполнения используются разные приставки: Исполнение 1 – 2REDMIL Исполнение 2 – 2BLUEMIL Исполнение 3 – без приставки Максимум можно указать 15 символов	Исполнение 1: iProduct = 'eFlash 2REDMIL' Исполнение 2: iProduct = 'eFlash 2BLUEMIL' Исполнение 3: iProduct = 'eFlash'

5.2. Описание неизменяемых параметров

Перечень неизменяемых параметров представлен в таблице ниже.

Таблица 34 — Описание неизменяемых параметров ВПО Aladdin eFlash

Наименование параметра	Обозначение	Принимаемые значения	Пример записи
Версия прошивки (FW) Под версией прошивки понимается версия встроенной системной программы «ВПО Aladdin eFlash»	FW_MAJOR_BYTE + FW_MINOR_BYTE bcdDevice	Версия прошивки задаётся путём установки значений в байты мажорной и минорной версий с последующей конкатенацией значений этих байтов. Значение считается неизменным, так как версия прошивки не меняется при её установке на N носителей. При выходе новой версии прошивки производится обновление версии прошивки и её принятие в качестве эталонной. Версионность прошивки должна быть уникальной в целях идентификации вероятных проблем и выработки решений для конкретных версий прошивки	FW_MAJOR_BYTE = 0x15 FW_MINOR_BYTE = 0x40 Итоговое значение версии прошивки – 1540 bcdDevice: 1540
Vendor ID Идентификатор производителя (вендора)	VID idVendor	Идентификатор производителя регистрируется официально в USB-IF. Компания АО «Аладдин Р.Д.» зарегистрирована со следующим VID = 24DC	VID = 0x24DC idVendor: 0x24DC
Product ID Идентификатор продукта	PID idProduct	Идентификатор продукта регистрируется на уровне компании. Для изделия «USB-носитель «Aladdin eFlash» зарегистрирован следующий PID = 0800	VID = 0x0800 idProduct: 0x0800
Параметры работы с SPI	SPI_SPMR SPI_SPCR SPI_CLK SPI_READ	При настройке работы с SPI (SPI_CLK = 60МГц, Dual SPI Mode) используются следующие параметры: SPI_SPMR = 0x08	SPI_SPMR = 0x08 SPI_SPCR = 0x6A SPI_CLK = 0x02 SPI_READ = 0x3B

Наименование параметра	Обозначение	Принимаемые значения	Пример записи
		SPI_SPCR = 0x6A SPI_CLK = 0x02 SPI_READ = 0x3B Установка других параметров может повлиять на режим работы (Dual/Standard SPI Mode) и тактовую частоту (SPI_CLK).	
Количество разделов на карте памяти	LUNs	Параметр принимает следующие значения: 0 — если один LUN; 1 — если 2 LUN. При указании других параметров, выдаётся ошибка с неправильным количеством LUN: «Wrong number of LUNs»	LUNs = 0 LUNs = 1

Список литературы

1 — SCSI Commands Reference Manual. Fibre Channel (FC). Serial Attached SCSI (SAS) — Seagate, publication number: 100293068, Rev. J, October 2016.