Универсальная последовательная шина (USB)

Описание класса устройств, взаимодействующих с человеком (HID)

Спецификация—6/27/01

Версия 1.11

Пожалуйста, присылайте свои комментарии через e-mail:
hidcomments@usb.org

Содержание

1. Предисловие

1.1 Об Интеллектуальной собственности

ЭТА СПЕЦИФИКАЦИЯ ПРЕДОСТАВЛЯЕТСЯ НА УСЛОВИЯХ "КАК ЕСТЬ" БЕЗ КАКИХ-ЛИБО ГАРАНТИЙ, В ТОМ ЧИСЛЕ ГАРАНТИЙ ПРИГОДНОСТИ ДЛЯ ОПРЕДЕЛЕННОЙ ЦЕЛИ.
ЭТА СПЕЦИФИКАЦИЯ ТОЛЬКО ДЛЯ ВНУТРЕННЕГО ИСПОЛЬЗОВАНИЯ.
АВТОРЫ ЭТОЙ СПЕЦИФИКАЦИИ ОТКАЗЫВАЮТСЯ ОТ ВСЕЙ ОТВЕТСТВЕННОСТИ, В ТОМ ЧИСЛЕ ОТВЕТСТВЕННСОТИ ЗА НАРУШЕНИЕ ПРАВ СОБСТВЕННСОТИ, СВЯЗАНЫХ С ОСУЩЕСТВЛЕНИЕМ ИНФОРМАЦИИ В ДАННОЙ СПЕЦИФИКАЦИИ. АВТОРЫ ДАННОЙ СПЕЦИФИКАЦИИ ТАКЖЕ НЕ ГАРАНТИРУЮТ И НЕ ЗАЯВЛЯЮТ, ЧТО НЕ НАРУШАЮТ ТАКИХ ПРАВ.

1.2 Авторы

Хотя многие люди способствовали созданию этого документа, только один участник в списке от каждой организации указан:
Alps Mike Bergman, Cybernet Tom Peurach, DEC Tom Schmidt, Intel Steve McGowan, Key Tronic Corporation Jodi Crowe, LCS/Telegraphics Robert Dezmelyk, Logitech Remy Zimmermann, Microsoft Corporation Mike Van Flandern, NCR Bob Nathan, Sun Microsystems Mike Davis, ThrustMaster Joe Rayhawk.

1.3 Данная версия

Эта версия включает все рассмотренные на момент даты выпуска заявки, соответствующие описанию класса устройств, взаимодействующих с человеком (HID).

1.4 История изменений

Версия	Дата	Описание
1.11	6/27/01	Рассмотрены заявки 39, 53, 60, 61, 62
1.1	4/7/99	Рассмотрены заявки 18, 19, 20, 21, 22, 23, 25, 26, 28, 29, 30, 32, 35 и 52.
1.0	1/30/96	Выпуск

1.5 Условные обозначения

Эта спецификация использует следующие условные обозначения

Пример обозначения	**Описание**
Get_Report, **Repor**t	Слова выделенные жирным текстом описывают определения.
Data, Non-Data	Использование регистра в словах используется для выделения описания типов и категорий
BValue	Выделение курсивом обозначает информацию данную разработчиком
bValue, bcdName, wOther	Приставки ‘b’, ‘bcd’, and ‘w’ используются для обозначения типа информации. Например: b бит или байт; bcd бинарый код.
[bValue]	Элементы в скобках обозначают дополнительные сведения.
{this (0) | that (1)}	Такие скобки с вертикальной линией обознчают выбор между двумя или более элементами

2. Введение

Универсальная последовательная шина (USB) это архитектура связи, дающая персональному компьютеру возможность соединения различных устройств с помощью простого 4-ех проводного кабеля. USB – это двухпроводной последовательный канал связи, работающий на 1,5 или 12 мегабитах в секунду. USB протоколы могут конфигурировать устройства при запуске системы или прямо во время их работы. Эти устройства подразделяется на различные классы утройств. Каждый класс устройств определяет обычное поведение и протоколы для устройств, служащих аналогичным функциям. Некоторые примеры классов USB устройств представлены в таблице:

Класс	**Пример**
Display (отображдение)	Монитор
Communication (связь)	Модем
Audio (аудио)	Колонки
Mass storage (накопитель)	Жесткий диск
Human interface (взаимодействие с человеком)	Перчатки с датчиками (Data glove)

См. также
Для большей информации о терминологии смотрите Приложение H: словарь определений. Далее предполагаетя, что вы прочли информацию и понимаете терминологию, данную в словаре.

2.1 Возможности

Данный документ описывает класс устройства для взаимодейсвия с человеком (Human Interface Device -**HID**) для исользования с USB. Понятия из USB спецификации используются, но не объясняются в данном документе.

См. также
Чтение USB спецификации рекомендуется для понимания содержимого этого документа. См. секцию 2.3: Документы по теме.

HID класс состоит в основном из устрйоств, используемых человеком для контроля компьютерных систем. Характерными примерами **HID** класса являются устройства

- Клавиатуры и указательные устрйоства– например, стандартная мышь, трекболы и джойстики.
- Устройства управление на передняя панели – переключатели, кнопки, ручки, слайдеры(sliders)
- Устрйоства управления, которые можно найти на таких устройствах, как телефоны, пульты дистанционного управления, игровых или симуляторных устройствах - например, перчатки с датчиками, руль и педали.

- Устройства, которые могут не требовать человеческого взаимодействия, но предоставляющие данные в том же формате, что и устройства класса HID – например, ридер штрих кодов, термометры, вольтметры.

Многие типичные устройства HID класса имеют индикаторы, специализированные дисплеи, звуковое сопровождение, силовую или тактильную обратную связь.
Таким образом, определение HID класса включает в себя поддержку различных видов продукции, направленных на конечного пользователя.

Замечание Обратите внимание, что устройства силовой обратной связи с взаимодействием в реальном времени описываются в документе “USB Physical Interface Device (PID) Class.”

См. также
Для более конкретных сведений см. спецификацию USB, глава 9, “USB Device Framework..”.
См. секцию 2.3: Документы по теме.

2.2 Цель

Этот документ предназначен для дополнения USB спецификации и обеспечивания производителей информацией, необходимой для создания USB совместимых устройств. Он также указывает, каким драйвером HID класс должен извлекать данные с устройств USB. Можно руководствоваться следующими целями при создании HID устройств:
- Быть как можно более компактным, чтобы сэкономить место данных устройства.
- Позволить программе пропускать неизвестную информацию.
- Быть расширяемым и надежным.
- Поддерживайте вложенность.
- Описывайте свое приложение для общей доступности

2.3 Документы по теме

Этот документ обращается к следующим документам:

Universal Serial Bus (USB) Specification.

USB Class Specification for Legacy Software

USB HID Usage Supplement

USB Physical Interface Device (PID) Specification

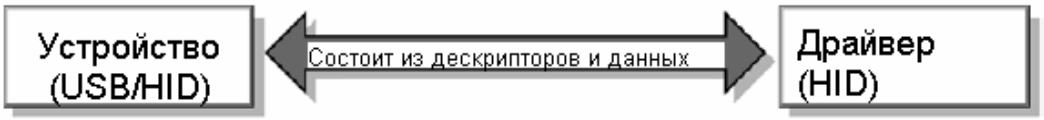
USB Audio Device Class

3. Обзор управления

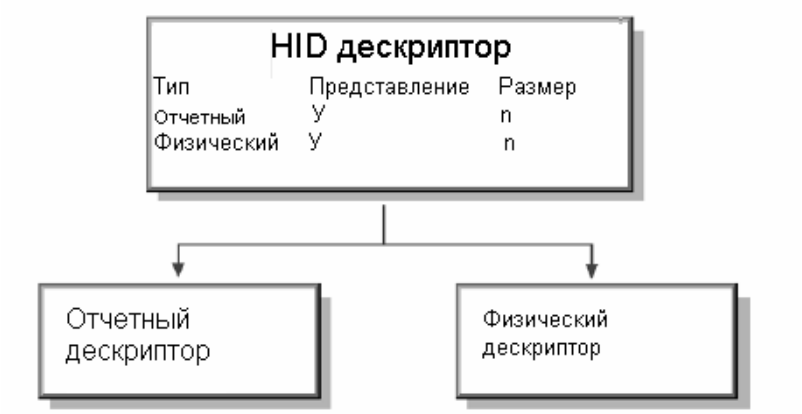
Информация о USB устройстве хранится в сегментах его ROM (памяти только для чтения). Эти сегменты называются дескрипторами (descriptors). Интерфейс дескриптора может определить устройство как принадлежащее к одному из конечного числа классов. **HID** класс будет рассматриваться наиболее подробно в этом документе.

Устройство USB/HID класса использует соответствующий драйвер **HID** класса для извлечения и маршрутизации всех данных.

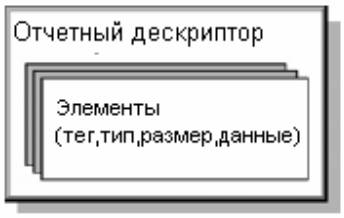
Маршрутизация и извлечение данных достигается путем анализа дескрипторов устройства и данных, которые он предоставляет.



Дескриптор устройства **HID** класса определяет, какие другие дескрипторы HID класса присутствуют, и выводит их размеры. Например**, Отчетный** и **Физический Дескрипторы (Report** and **Physical Descriptors).**



Дескриптор **Отчета** описывает каждый фрагмент данных, которые генерирует устройство и размер данных, которые фактически измеряются.

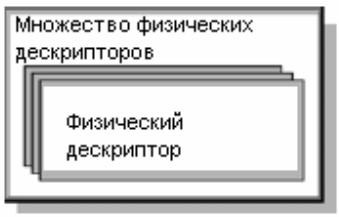


Например, **отчетный** дескриптор определяет элементы, описывающие положение или состояние кнопки. Информация элемента используется для:

- Определение для маршрутизации ввода – например, передать вход мыши или API джойстика.
- Разрешить программному обеспечению назначить функциональность ввода – например, использовать вход джойстика, как позицию танка.

Изучая предметы (под общим названием Отчетный дескриптор), драйвер HID класса может определить размер и состав отчетов с данными от устройства HID класса.

Множества **Физических дескрипторов** являются дополнительными дескрипторами, которые предоставляют информацию о части или частях человеческого тела, используемую для активации управления на устройстве.



Все описанное можно собрать, чтобы проиллюстрировать структуру дескриптора.

Остальная часть данной спецификации – информация о реализации, предостережениях и ограничениях для развития устройств и драйверов **HID** класса.

4. Функциональная характеристика

Данный параграф рассматривает функциональную характеристику **HID**:
- Класс
- Подкласс
- Интерфейсы.

4.1 HID класс

USB устройства разбиты на классы устройств:
- Имеющие схожие требования по перемещению данных
- Делят драйвер одинакового типа.

Например, Устройства класса **Аудио** требуют изохронные каналы данных. Устройства **HID** класса имеют другие, гораздо более простые транспортные требования.

Замечание: USB устройства, чьи требования по пересылке данных находятся за пределами определенных классов, должны сопровождаться их собственной спецификацией класса и драйвера, как указано в USB Specification.

USB устройство может принадлежать к одиночному классу или может состоять из множества классов. Например, каждая сторона при телефонной связи должна использовать возможности классов **HID**, **Аудио** и **Телефонии**. Это возможно, поскольку класс указывается в **Интерфейсном** дескрипторе а не в дескрипторе **Устройства**. Это обсуждается далее в разделе 5.1 Структура дескриптора Устройства.

Спецификация USB Core определяет код класса HID. *bInterfaceClass* – член Интерфейсного дескриптора всегда 3 для устройства HID класса.

> См. также
> Спецификация Аудио класса очень точно определяет транспортные требования. См. секцию 2.3: Документы по теме.

4.2 Подклассы

Во время раннего развития спецификации **HID,** подклассы должны были быть использованы для определения конкретных протоколов различных типов устройств **HID** класса. Хотя вариации этой модели используются в настоящее время в отрасли (все устройства используют протоколы, определенные для одинаковых популярных устройств), быстро стало очевидно, что данный подход был слишком ограниченным. То есть, устройствам нужно было вписаться в узкие подклассы и были не способны обеспечивать функциональность устройства за пределами поддерживаемого подкласса.

HID комитет договорился о неправдоподобии данного подкласса протоколов для всех возможных (и еще не задуманных) устройств, которые могут быть определены. Кроме того, некоторые известные устройства , казалось, оседлали несколько классификаций – например, клавиатуры с локаторами или локаторы с поддержкой клавиш. Следовательно класс **HID** не спользует подклассы для определения большинства протоколов. Вместо этого, усройство **HID** класса идентифицирует свой протокол данных и тип данных, предоставляемых **Отчетным** дескриптором.

Дескриптор отчета загружается и обрабатывается драйвером **HID** класса, как только устройство обнаружено. Протоколы для существующих и новых устройств создаются путем смешивания типов данных внутри отчетного дескриптора.

Замечание: поскольку анализатор **Отчетного** дескриптора представляет собой большой объем кода, простой метод является необходимым для идентификации протокола устройства для устройств, требующих BIOS поддержку. Устройства **HID** класса используют часть Подкласса для обозначения устройств, которые поддерживают предопределение протокола для любого устройства мыши иликлавиатуры (То есть, устройство может использоваться как **Загрузочное устройство**). Загрузочный протокол может быть расширен для включения дополнительных данных, нераспознаваемых BIOS, или устройство может поддерживать второй предпочтительный протокол для использования драйвером **HID** класса.

bInterfaceSubClass заявляет, поддерживает ли устройство загрузочный интерфейс, иначе он равен 0.

Коды подкласса

Код подкласса	**Описание**
0	Нет подкласса
1	Подкласс загрузочный интерфейс
2 - 255	Зарезервированы.

См. также
Загрузочный дескриптор Отчета описан в Приложении B: Дескрипторы с загрузочным интерфейсом. Для HID подкласса и протокола кода смотри Дополнение E: Примеры USB дескрипторов для устройств HID класса.

4.3 Протокол

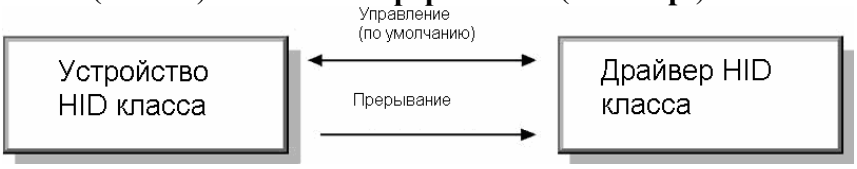
Различные протоколы поддерживаются HID устройствами. bInterfaceProtocol - член Интерфейсного дескриптора, имеет смысл, только если bInterfaceSubClass заявляет, что устройство поддерживает загрузочный интерфейс. иначе он равен 0.

Коды протокола

Код протокола	Описание
0	нет
1	Клавиатура
2	Мышь
3 - 255	Зарезервировано

4.4 Интерфейсы

Устройство **HID** класса общается с драйвером **HID** класса, используя либо канал **Управления (control)** либо канал **Прерывания (Interrupt)**.



Канал управления используется для:

- приема и ответов на запросы о USB управлении и классе данных.
- отправки данных при опросе драйвером **HID** класса (используется запрос **Get_Report**)
- приема данных от хоста.

Канал **прерывания** используется для

- приема асинхронных (незапрашиваемых) данных от устройства
- передачи данных с низкой задержкой на устройство.

Выходящий канал Прерывания (**Interrupt Out** pipe) является дополнительным. Если

устройство объявляет выходящие прерывания конечной точки, тогда выходящие отчеты передаются хостом устройству через выходящие прерывания конечной точки. Иначе отчеты передаются устройству через Управление конечной точки, используя Set_Report(Output) запросы.

Замечание: **Конечная тока (Endpoint) 0** - это канал **управления**, представленный в USB устройстве. Таким образом, только канал Прерывания Внутрь (**Interrupt In** pipe) описан для дескриптора, с использованием дескриптора **Endpoint**. По сути, несколько **Interface** дескрипторов могут делить **Endpoint 0**. Канал **Interrupt Out** дополнительный и требует дополнительного дескриптора **Endpoint** при объявлении.

Канал	Описание	Требование
Control (Endpoint 0)	USB управление, класс запроса кодов и данных опроса (сообщение)	Y
Interrupt In	Информация в, т.е., данные от устройства (потоковые данные)	Y
Interrupt Out	Данные из, т.е., данные в устройство (потоковые данные).	N

См. также
для деталей о канале **Control**, смотри USB спецификацию.

4.5 Ограничения устройства

Эта спецификация относится как к высокоскоростному, так и к низкоскоростному классу **HID** устройств. Каждый тип устройства имеет различные ограничения, как сказано в главе 5 USB спецификации.

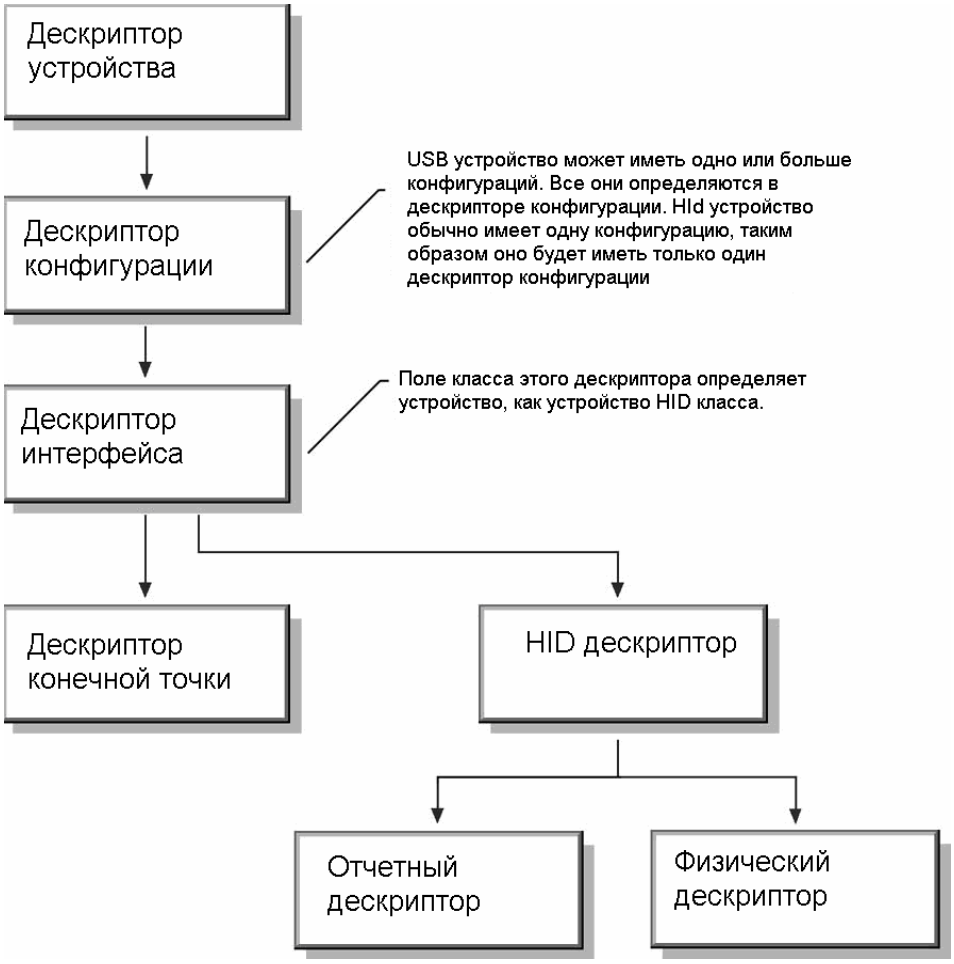
5. Операционная модель

Этот параграф описывает основные оперативные модели **HID** устройств. Блок-схема элементов представляет собой таблицы информации с прошивкой.

5.1 Структура дескриптора устройства

На самом верхнем уровне, дескриптор включает две таблицы информации, указанной в качестве дескриптора устройства в String дескрипторе. Стандартный дескриптор USB устройства определяет ID продукта и другую информацию о устройстве. Например, поля дескриптора в основном содержат::

- класс
- подкласс
- вендор (поставщик)
- продукт
- версия



Для устройств HID класса:

- Тип класса не определен на уровне дескриптора **Устройства**. Тип класса HID устройства определяется Интерфейсным дескриптором.

- Поле подкласса используется для определения **Загрузочных Устройств**

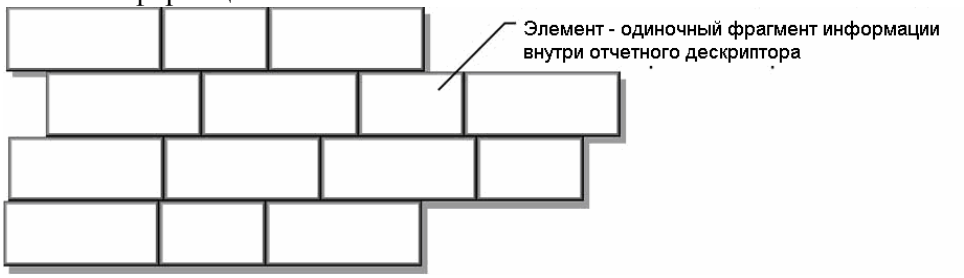
Примечание: поля bDeviceClass и bDeviceSubClass в дескрипторе Устройства не должны использоваться для идентификации устройства, как принадлежащего к HID классу. Вместо них

используйте bInterfaceClass и bInterfaceSubClass поля в дескрипторе интерфейса.

См. также
Драйвер HID класса определяют тип устройства и функции, путем изучения дополнительных класс-дескрипторов. Для большей информации смотрите 6.2 Класс-дескрипторы.

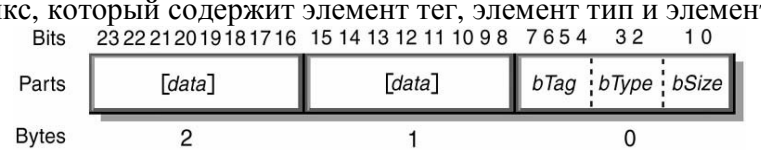
5.2 Отчетный дескриптор

Предшествующие дескрипторы иллюстрируется с помощью блок-схемы элементов, которые представляют таблицы информации. Каждую таблицу информации можно рассматривать как блок данных. **Отчетные** Дескрипторы состоят из фрагментов информации. Каждая часть информации называется **элемент**.

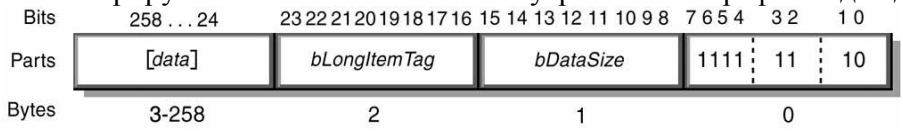


5.3 Общий формат элемента

Элемент – это часть информации об устройстве. Все элементы имеют однобайтовый префикс, который содержит элемент тег, элемент тип и элемент размер.



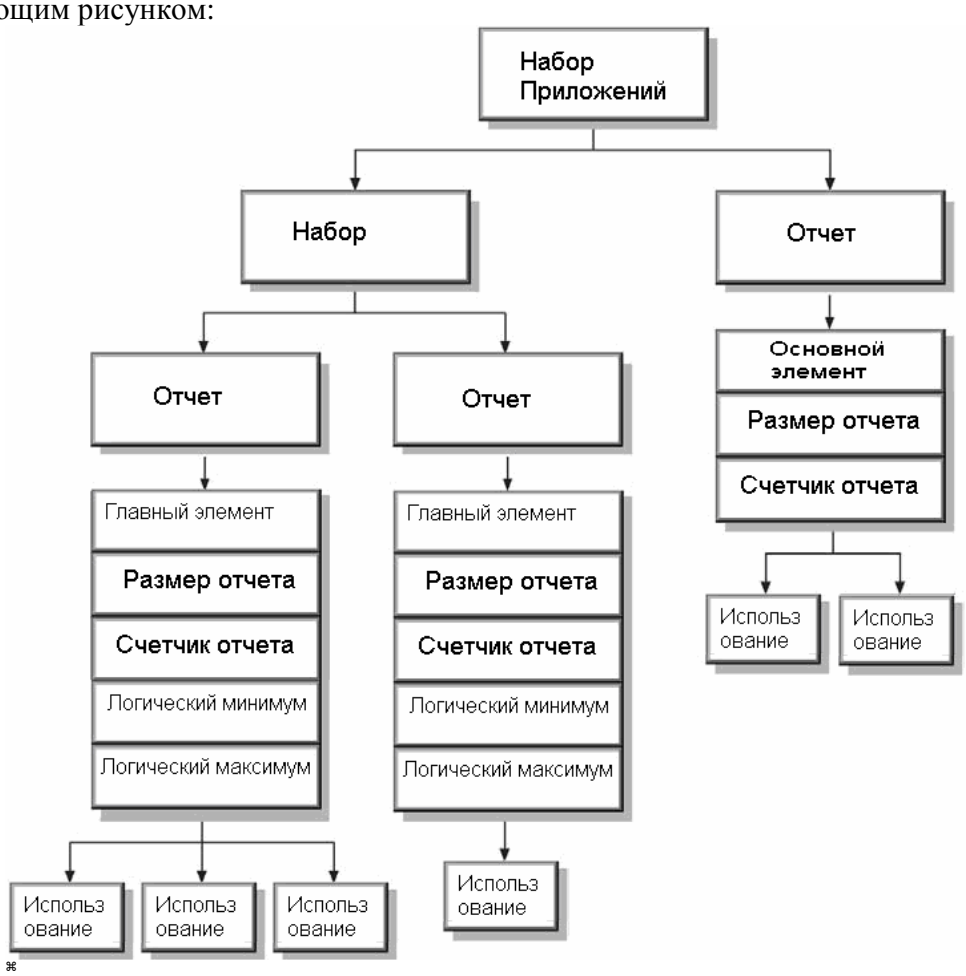
Элемент может включать в себя дополнительный элемент данных. Размер блока данных элемента определяется его основным типом. Существуют два основных типа элементов: короткие и длинные. Если элемент короткий, то его дополнительным объемом данных может быть 0, 1, 2 или 4 байта. Если элемент длинный - его bSize значение всегда 2. Следующий пример иллюстрирует возможные значения внутри 1-байта префикса для длинного элемента.



5.4 Элементный анализатор (Item Parser)

Драйвер **HID** класс содержит синтаксический анализатор, используемый для анализа элементов, обнаруженных в **отчетном** дескрипторе. Парсер извлекает информацию из дескриптора в линейном порядке. Парсер, проходя через дескриптор, собирает состояние каждого известного элемента и сохраняет их в таблицу состояний элементов. Таблица состояния элементов содержит состояние отдельных элементов.

С точки зрения синтаксического анализатора, устройство **HID** класса выглядит следующим рисунком:



Когда некоторый элемент встречается, содержимое таблицы состояний элементов сдвигается. Эти элементы включают в себя все **Main, Push и Pop** элементы.

- Когда **Main** элемент найден, новая структура доклада выделяется и инициализируется с текущей таблицы состояния элементов. Все **Local (местные)** элементы затем удаляется из таблицы состояния элементов, но **Global (глобальные**) элементы остаются. Таким образом, **глобальные** элементы устанавливают значения по умолчанию для последующих новых **Main** элементов. Для устройства с подобным управлением, например, шесть осей, будет необходимо определить **глобальные** элементы только один раз до первого **Main** элемента.

Примечание Main элементы связаны с набором в порядке, в котором они были объявлены. Новая коллекция начинается тогда, когда анализатор достигает **Collection** элемента. Элемент анализатор связывает с набором всех **Главных** элементов, определенных между **Collection** элементом и следующим **End Collection** элементом.

- Когда **Push** элемент встречается, таблица состояния элементов копируется и размещаются в стеке для последующего использования.

- Когда **Pop** элемент найден, таблица состояния элементов заменяется верхней таблицей из стека. Например:

Unit (Meter), Unit Exponent (-3), Push, Unit Exponent (0)

Когда парсер достигает **Push** элемент, он помещает элемент единиц измерения в миллиметрах в стек. Следующий элемент изменяет таблицу состояния элементов на элементы в метрах

Парсер обязан разобрать весь **отчетный** дескриптор, чтобы найти все **Main** элементы. Это необходимо для того, чтобы анализировать отчеты, посланные устройством.

См. также
Для деталей см. параграф 8.

5.5 Использование (Usages)

Использование являются частью **отчетного** дескриптора и поставляется разработчику приложения с информацией о том, какое управление используется. Кроме того, **Usage** тег указывает одобренный поставщиком способ использования для определенного элемента управления или группы элементов управления. Хотя **отчетные** дескрипторы описывают формат данных, например, три 8-битных поля, **Usage** тег определяет, что должно быть сделано с данными, например, ввод x, y, z. Эта функция позволяет поставщику убедиться, что пользователь видит последовательно заданную функцию управления в различных приложениях.

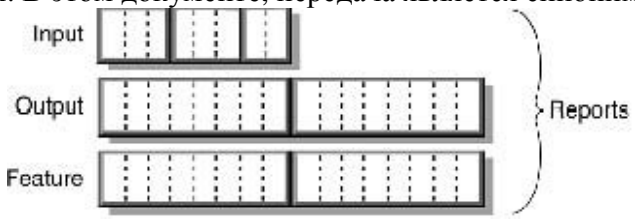
Отчетный дескриптор может иметь несколько **Usage** тегов. Это соответствие один к одному между Usage и Control, одно соответствие определено в дескрипторе. Массив указывает, что каждая область **отчетного** дескриптора представляет несколько физических элементов управления. Каждый элемент управления может иметь атрибуты, закрепленные за ним. Например, массив из четырех кнопок мог бы иметь уникальный **Usage** тег для каждой кнопки.

Usage это 32-битовое беззнаковое целое, где старшие 16 бит определяют **Usage Page** и младшие 16 бит определяют **Usage ID**. Usage ID используется для выбора индивидуального Usage на Usage Page

См. также
Для примера см. приложение E.10: Отчетный дескриптор (мышь)

5.6 Отчеты

Используя USB-терминологию, устройство может отправлять и получать транзакцию каждый USB-кадр (1 мс). Транзакция может быть составлена из нескольких пакетов (маркера, данные, хендшейк), но имеет ограниченный размер до 8 байтов для низкоскоростных устройств и 64 байта для высокоскоростных устройств. Передача одной или нескольких транзакций создает набор данных, которые имеет смысл для устройства, например, **Input, Output и Feature** отчеты. В этом документе, передача является синонимом отчета.



Большинство устройств генерирует отчеты или передачи, возвращая структуру, в которой каждое поле данных последовательно представлены. Тем не менее, некоторые устройства могут иметь несколько отчетных структур на одну конечную тоску, каждая из

которых представляется лишь несколькими полями данных. Например, клавиатура с встроенным указательным устройством может самостоятельно послать "нажатие клавиши" и "указывающие" данные через одинаковую конечную точку. **Report ID** элементы используются для обозначения тех полей данных, которые представленны в каждом докладе структуры. Тег Report ID элемента назначает 1-байт идентификации префикса для каждого отчета передачи. Если ни один тег Report ID элемента не присутствует в отчетном дескрипторе, можно предположить, что только одна **Input**, **Output**, и **Feature** отчетная структура доклада существует, и вместе они представляют все данные устройства.

Замечание: только **Input** отчеты направляются через **Interrupt In** канал. **Feature и Output** отчеты должны быть инициированы хостом через канал контроля или дополнительного **Interupt Out.**

Если устройство имеет несколько отчетных структур, то все передачи данных начинаются с 1-байтового идентификатора - префикса, который указывает, какая структура отчета распространяется на передачу. Это позволяет драйверу класса различать входящие данные от указателя и клавиатуры, путем изучения префикса.

5.7 Строки (Strings)

Набор или поле данных может иметь конкретный указатель (строка индекса) связанный с ним.

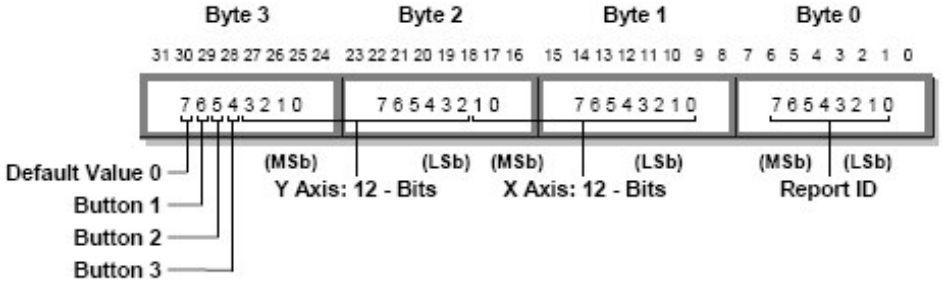
Тег **Usage** элемента не обязательно совпадает со строкой, связанной с **Main** элементом. Тем не менее, строки могут быть полезны, когда требуется определение поставщика. **String** дескриптор содержит список текстовых строк для устройства.

> См. также
> Для деталей смотри дополнение E: Пример USB дескриптора для HID устройства

5.8 Формат многобайтовых числовых значений

Многобайтовых числовых значений в отчетах представлены в прямом порядке байтов, причем младший байт по младшему адресу. Логические Минимальные и Максимальные значения определяют диапазон значений, которые можно найти в отчете. Если Логические Минимальное и Максимальное значения оба положительные, то бит знака не нужен в поле отчета и содержимое поля можно считать значением без знака. В противном случае, все целые значения будут со знаками и представлены в дополнительном коде. Значения с плавающей запятой не допускаются.

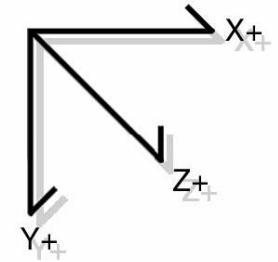
Младший бит в значении хранится в бите 0, следующий, более важный бит в бите 1 и так далее вплоть до размера всего значения. Следующий пример иллюстрирует бит-представление многобайтового числового значения



Byte	Bits
0	0-7
1	8-15
2	16-23
3	24-31

5.9 Ориентация

Устройства **HID** класса, рекомендуется, по возможности, использовать право определенную систему координат. Если пользователь работает с устройством, значения в отчете должны увеличиться как управление, перемещающееся слева направо (X), издалека и ближе (Y) и с высоты в низ (Z).



Управление в отчете в двоичных данных должно использовать: 0 = выкл или ЛОЖЬ, и 1 = включен или ПРАВДА. Примеры такого управления: ключи, кнопки, переключатели питания, датчики близости.

5.10 Значения NULL

Устройства HID класса поддерживают способность игнорировать выбранные поля в отчете во время выполнения. Это достигается путем объявления битового поля в отчете, способного вместить в диапазоне значений больше, чем на самом деле генерирует контроль. Если хост или устройство получает значение, превышающее допустимое, то текущее значение контроля не будет изменено.

Разработчик аппаратного обеспечения должен тщательно оценить управление в каждом отчете, чтобы определить, как приложение на хосте будут их использовать. Если возможна ситуация, в которой приложение не будет изменять соответствующую область каждый раз, когда сообщение передается на устройство, то область должна обеспечить NULL значение. С Null значением хост может инициализировать все поля в отчете, и когда не захочет изменяться Null (вне диапазона) значение, он установит поля, в котором он будет изменяться в своем диапазоне.

Если 8-битовое поле объявлено и диапазон допустимых значений составляет от 0 до 0x7F то любое значение в диапазоне от 0x80 до 0xFF будет рассматриваться как вне диапазона и игнорируется при получении. Инициализации нулевых значений в отчете намного легче, если они все те же.

ПРИМЕЧАНИЕ: Настоятельно рекомендуется, что 0 должен быть включен в набор Null значений, так что отчет буферов может просто быть установлен в ноль, чтобы установить состояние "не волнует" для всех полей.

6. Дескрипторы

6.1 Стандартные дескрипторы

Устройство HID класса использует следующие сандартные USB дескрипторы:

- - **Device (устройства)**
- **- Configuration (конфигурации)**
- **- Interface (интерфейса)**
- **- Endpoint (конечной точки)**
- **- String (строки)**

> См. также
> Для деталей о этих жескрипторах, определенных как устройства HID класса, смотри дополнение E: Пример USB дескриптора для устройсва HID класса. Для общей информациио стандартных USB дескрипторах, смотри часть 9 USB спецификации.

6.2 Классовые дескрипторы

Каждый класс устройства содержит один или более классовых дескрипторов. Эти дескрипторы отличаются от стандартных USB дескрипторов. Устройства HID класса используют следующие классовые дескрипторы:

- **HID**
- **Report (отчетный)**
- **Physical (физический)**

6.2.1 HID дескриптор

Описание

HID дескриптор определяет длину и тип подчиненных дескрипторов для устройства.

Части

Часть	Смещение/Размер (Байт)	Описание
bLength	0/1	Числовое выражение, то есть общий размер дескриптора HID.
bDescriptorType	1/1	Постоянное имя, указывающее тип HID дескриптора.
bcdHID	2/2	Числовое выражение, идентифицирующее спецификацию HID класса.
bCountryCode	4/1	Числовой выражение, определяющее код страны локализирующей оборудование.
bNumDescriptors	5/1	Числовое выражение, определяющее количество классов дескрипторов (всегда, по крайней мере, один, т.е. отчетный дескриптор)
bDescriptorType	6/1	Постоянное имя, идентифицирующее тип класса дескриптора. См. раздел 7.1.2

wDescriptorLength	7/2	Числовое выражение, обозначающее полный размер отчетного дескриптора
[bDescriptorType]...	9/1	Постоянное имя, определяющее тип дополнительного дескриптора
[wDescriptorLength]...	10/2	Числовое выражение, определяющее полный размер дополнительного дескриптора

Замечания

- - Если дополнительный дескриптор указан, соответствующая запись длины должна также быть указана.
- - Нескольк дополнительных дескрипторов и связанные с ними длины могут быть определены до смещения (3 * n) +6 и (3 * n) +7 соответственно.
- - Значение *bNumDescriptors* определяет количество дополнительных классов дескрипторов. Это число должно быть по крайней мере равно одному (1) и **отчетный** дескриптор всегда будет присутствовать. Остальная часть HID дескриптора имеет длину и тип каждого дополнительного класса дескриптора.
- - Значение *bCountryCode* определяет, для какич стран оборудование локализованно. Большинство аппаратного обеспечения не локализовано и, следовательно, это значение будет равно нулю (0). Тем не менее, клавиатуры могут использовать это поле, чтобы указать язык кнопок. Устройства не обязаны размещать значение, отличное от нуля в этой области, но в некоторых операционных средах может потребоваться эта информация. В следующей таблице приведены коды стран.

Код (деятичный)	страна	Код (деятичный)	страна
00	Not Supported	18	Netherlands/Dutch
01	Arabic	19	Norwegian
02	Belgian	20	Persian (Farsi)
03	Canadian-Bilingual	21	Poland
04	Canadian-French	22	Portuguese
05	Czech Republic	23	Russia
06	Danish	24	Slovakia
07	Finnish	25	Spanish
08	French	26	Swedish
09	German	27	Swiss/French
10	Greek	28	Swiss/German
11	Hebrew	29	Switzerland
12	Hungary	30	Taiwan
13	International (ISO)	31	Turkish-Q
14	Italian	32	UK
15	Japan (Katakana)	33	US
16	Korean	34	Yugoslavia
17	Latin American	35	Turkish-F
		36-255	Reserved

6.2.2 Отчетный дескриптор

Отчетный дескриптор отличается от других дескрипторов в том, что это не просто таблица значений. Длина и содержание **Отчетного** дескриптора варьируется в зависимости от количества полей данных, необходимых для отчета устройства или отчетов. **Отчетный** дескриптор состоит из элементов, которые содержат информацию об устройстве. Первая часть элемента содержит три поля: тип элемента, тег элемента, и размер элемента. Вместе эти поля определяют, какую информацию предоставляет элемент.

Вот три типа элементов: **Main**, **Global** и **Local**. Существуют пять тегов **Main** элемента в настоящее время:

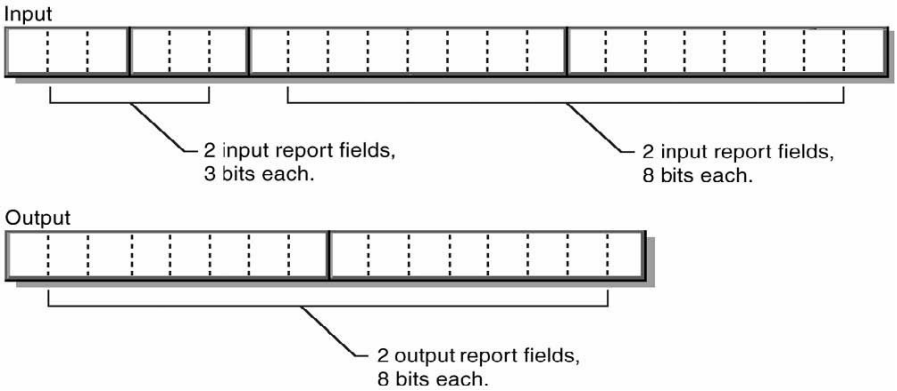
- - Тег **Input** элемента: Относится к данным от одного или нескольких аналогичных элементов управления на устройстве. Например, переменные данные, такие как чтение положения одной оси или группы рычагов или массива данных, например, одной или нескольких кнопок или переключателей.
- - Тег **Output** элемента: Относится к данным одного или нескольких аналогичных элементов управления на устройстве, таким как установка положения одной оси или группы рычагов (переменные данные). Или, он может представлять данные в один или более светодиодов (массив данных).
- - Тег **Feature** элемента: Описывает ввод и вывод устройства, не предназначенные для потребления конечными пользователями, например, функция программного обеспечения или Панель управления переключения.
- - Тег **Collection** элемента: смысловая группировка **Input**, **Output**, и **Feature** элемента - например, мышь, клавиатура, джойстик и указатель.
- - Тег **End Collection** элемента: используется для указания конца набора элементов.

Отчетный дескриптор содержит описание данных, предоставленных каждым элементом управления в устройстве. Каждый тег **Main** элемента определяет размер данных, возвращенных определенным элементом управления, и определяет, являются ли данные абсолютным или относительным, и другую соответствующую информацию. Предшествующие **Локальные и Глобальные** элементы определяют минимальные и максимальные значения данных и так далее. **Отчетный** дескриптор это полный набор всех элементов устройства. Глядя на **отчетный** дескриптор,приложение знает, как обработывать входящие данные, а также то, для чего данные могут быть использованы.

Одно или несколько полей данных от управления определяются **Main** элементом и далее описывается **глобальным и локальным** элементами. **Локальные** элементы описывают только поля данных, определенные следующим **Main** элементом. **Глобальные** элементы становятся атрибутами по умолчанию для всех последующих полей данных в этом дескрипторе. Например, рассмотрим следующее (детали опущены для краткости):

Report Size (3)
Report Count (2)
Input
Report Size (8)
Input
Output

Элементный анализатор интерпретирует **отчетный** дескриптор и создает следующие отчеты:



Отчетный дескриптор может содержать несколько **Main** элементов. **Отчетный** дескриптор должен включать каждый из следующих элементов для описания данных управления (все остальные элементы являются дополнительными):

- **Input (Output or Feature)**
- **Usage**
- **Usage Page**
- **Logical Minimum**
- **Logical Maximum**
- **Report Size**
- **Report Count**

Ниже представлен кодинг-образец элементов, используемых для определения 3-ех кнопочной мыши. В этом случае **Main** элементы предшествуют Global элементам, таким как **Usage, Report Count, Report Size** (каждая линия новый элемент.

```
Usage Page (Generic Desktop),                ;Используйте общую Usage Page
Usage (Mouse),
        Collection (Application),            ;Открыть набо мышь
        Usage (Pointer),
        Collection (Physical),               ;Открыть указательный набор
                Usage Page (Buttons)
                Usage Minimum (1),
                Usage Maximum (3),
                Logical Minimum (0),
                Logical Maximum (1),                ;поля возвращают днные от 0 до 1
                Report Count (3),
                Report Size (1),                    ;создать три однобитных поля (кнопки 1, 2,& 3)
                Input (Data, Variable, Absolute),   ;Создать поле для входных отчетов
                Report Count (1),
                 Report Size (5),                    ;Создать пяти битное поле констант
                Input (Constant),                   ;Добавить поле в входящий отчет
                Usage Page (Generic Desktop),
                Usage (X),
                Usage (Y),
                Logical Minimum (-127),
                Logical Maximum (127),              ;поле возвращает значения от -127 до 127
                Report Size (8),
```

```
        Report Count (2),                ;Создать два 8 битных поля (X & Y)
        Input (Data, Variable, Relative),  ;Добавить поля в входящий отчет
    End Collection,                      ;закрыть набор указателя
End Collection                           ;закрыть набор мыши
```

6.2.2.1 Типы и теги

Все элементы содержат 1-байтный префикс, который определяет основной тип элемента. HID класс определяет два основных формата элементов:

- короткие: 1 – 5 байт общей длины, используются для наиболее часто встречающихся элементов. Содержат 1 или 0 байт дополнителой информации.
- длинные: 3 – 258 байт в длину, используются для элементов, требующих больших объемов данных.

Замечание: Эта спецификация определяет только те элементы, которые используют короткий формат.Двух элементные форматы не следует путать с типами элементов, таких как **Main, Global, Local.**

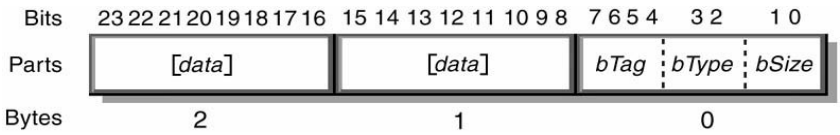
> См. также
> Для дополнительной информации смотри параграф 5.3

6.2.2.2 Короткие элементы

Описание

Короткий формат содержит размер элемента, тип, тег в первом байте. Первый байт может следовать за 0,1,2,4 байтами дополнительных данных, зависящих от размеров данных.

Части



Часть	Описание
bSize	Числовое выражение определяющее размер: 0 = 0 bytes 1 = 1 byte 2 = 2 bytes 3 = 4 bytes
bType	Числовое выражение, определяющее тип данных где: 0 = Main 1 = Global 2 = Local 3 = Reserved
bTag	Числовое выражение, опредляющее функцию элемента.
[data]	Дополнительные данные

Примечения

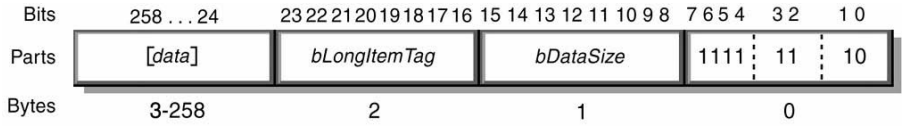
- Теги коротких элементов не имеют явного значения для bSize. Вместо этого значение данных элемента определяют размер элемента. То есть, если элемент может быть представлен в виде одного байта, то данные определяются как 1 байт, хотя это и не требуется.
- Если ожидается большой элемент данных, то он может быть сокращен, если все его старшие байты занулены.
- Есть три категории тегов коротких элементов **Main, Global и Local**. Тип элемента (bType) определяет тег категории и поведение элемента.

6.2.2.3 Длинные элементы

Описание

Как и короткие элементы, длинные содержат размер элемента, тип, тег в первом байте. Длинный формат данных использует специальное значение тега элемента для указания, что это длинный элемент.Размер длинного элемнта и тег длинного элемнта 8-битные величины. Данные элемнта могут содержать до 255 байтов данных.

Части



Часть	**Описание**
bSize	Числовое выражение, определяющее полный размер элемента (2 байта), определяет тип элемента как длинный.
bType	Числовое выражение, определяющее тип элемента (3=reserved)
bTag	Числовое выражение, определяющее функции элемента, всегда 1111
[bDataSize]	Размер элемента
[bLongItemTag]	Тег
[data]	Дополнительные данные

Важно: В данном документе не определяются теги длинных элементов. Эти теги используются для будущего использования. Теги xF0–xFF определяются поставщиком..

6.2.2.4 Главные элементы

Описание

Main элементы используются для определения или группирования некоторых типов полей данных внутри Отчетного дескриптора. Есть два типа Main данных: данные и не данные. Тип данные Main элементы используют для создания полей внутри отчета и включают Input, Output и Feature. Остальные элементы не создают полей и в дальнейшем относятся к типу не данных Main элемета.

Части

Тег Main элемента	1-байтный префикс	Данные	
Input	1000 00 *nn*	Bit 0	{Data (0) | Constant (1)}
		Bit 1	{Array (0) | Variable (1)}
		Bit 2	{Absolute (0) | Relative (1)}
		Bit 3	{No Wrap (0) | Wrap (1)}
		Bit 4	{Linear (0) | Non Linear (1)}
		Bit 5	{Preferred State (0) | No Preferred (1)}
		Bit 6	{No Null position (0) | Null state(1)}
		Bit 7	Reserved (0)
		Bit 8	{Bit Field (0) | Buffered Bytes (1)}
		Bit 31-9	Reserved (0)
Output	1001 00 *nn*	Bit 0	{Data (0) | Constant (1)}
		Bit 1	{Array (0) | Variable (1)}
		Bit 2	{Absolute (0) | Relative (1)}
		Bit 3	{No Wrap (0) | Wrap (1)}
		Bit 4	{Linear (0) | Non Linear (1)}
		Bit 5	{Preferred State (0) | No Preferred (1)}
		Bit 6	{No Null position (0) | Null state(1)}
		Bit 7	{Non Volatile (0) | Volatile (1)}
		Bit 8	{Bit Field (0) | Buffered Bytes (1)}
		Bit 31-9	Reserved (0)
Feature	1011 00 *nn*	Bit 0	{Data (0) | Constant (1)}
		Bit 1	{Array (0) | Variable (1)}
		Bit 2	{Absolute (0) | Relative (1)}
		Bit 3	{No Wrap (0) | Wrap (1)}
		Bit 4	{Linear (0) | Non Linear (1)}
		Bit 5	{Preferred State (0) | No Preferred (1)}
		Bit 6	{No Null position (0) | Null state(1)}
		Bit 7	{Non Volatile (0) | Volatile (1)}
		Bit 8	{Bit Field (0) | Buffered Bytes (1)}
		Bit 31-9	Reserved (0)
Collection	1010 00 *nn*	0x00	Physical (group of axes)
		0x01	Application (mouse, keyboard)
		0x02	Logical (interrelated data)
		0x03	Report
		0x04	Named Array
		0x05	Usage Switch
		0x06	Usage Modifier
		0x07-0x7F	Reserved
		0x80-0xFF	Vendor-defined
End Collection	1100 00 *nn*	Not applicable. Closes an item collection.	
Reserved	1101 00 *nn* to 1111 00 *nn*	Not applicable. Reserved for future items.	

Замечания

- Значения всех **Main** элементов по умолчанию - ноль
- Элемент **Input** могут иметь размер данных 0 байт. В этом случае значение каждого бита данных для элемента предположительно 0. Это идентично использованию тега элемента, который определяет 4-батные данные 4 последовательными нулями.

6.2.2.5 Input, Output и Feature элементы

Описание

Input, Output и Feature элементы используются для создания полей через отчет.

- **Input** элемент описывает инормацию о данных, предоставляемых физическим управлением или управлениями. Приложение может использовать данную информацию для распознавания данных от устройства. Все поля данных определяются в одиночном элементе разделяя идентификационный формат данных.
- **Output** элемент используется для задания исходящих полей данных в отчете. Этот элемент схож с **Input** элементом, но определяет данные для посылки на устройство.
- **Feature** элементы описывают конфигурации устройства, которые могут быть посланы на устройство.

Части

Бит	Часть	Значение	Описание
0	Данные | Константа	0 | 1	Представляет элемент либо как константу, либо как данные Данные показывают элемент, определяющий поле отчета, которые содержат модифицируемые данные. Константа показывает элемент как статически читаемое поле в отчете
1	Массив | переменная	0 | 1	Показывает, что создает элемент - либо поле переменной, либо поля массива данных в отчете В полях переменной, каждое поле представляет данные от физического управления. Число битов зарезервированы для каждого поля, определяется предыдущим Размером Отчета/Счетчиком Отчета элементом. Например, набор из восьми вкл / выкл переключателей могут быть представлены 1 байтом в объявленной переменной. Input элемент, где каждый бит представляет один переключатель, вкл (1) или выкл (0) (Report Size = 1, Report Count = 8) Иначе переменная Input элемента должна добавить 1 отчетный байт, необходимый для предоставления состояния 3-4 положений кнопок, где каждое состояние каждой кнопки представляется 2 битами (Report Size = 2, Report Count = 4). Или один байт переменной Input элемента может представлять X позицию джойстика (Report Size = 8, Report

			Count = 1). Массив обеспечивает альтернативное средство для описания данных, возвращенных от группы кнопок. Массивы более эффективны, хотя и менее гибкие. Вместо возвращения одиночного бита для каждой кнопки, массив возвращает индекс в каждом поле, который отвечает нажатым кнопкам (как скан коды клавиатуры). Переменные и поля массивов вне диапазона отвечают отстутствию контроля. Кнопки и ключи в массиве, которые нажаты одновременно, должны быть отображены множестве полей. Таким образом, количество полей в массиве входящего элемента (Report Count) задает максимальное число одновременных элементов управления, которые можно представить в отчете. Клавиатура может отправить три одновременно нажатых кнопки, используя массив с тремя 8-и битными полями (Report Size = 8, Report Count = 3). логический минимум определяет меньший индекс возвращенный массивом, Логический максимум определяет максимальный. Количество элементов в массиве может быть выведено используя анализ разницы между Logical Minimum и Logical Maximum.
2	Абсолют | Относит	0 | 1	Показывает, данные абсолютные (основанные на фиксированном происхождении) или относительные (с указанием изменений в значении последнего отчета). Например, мышь часто обеспечивается относительными данными.
3	No Wrap | Wrap	0 | 1	Показывает,могут ли данные “rolls over” (зашкаливать) при достижении или принимать экстремальное высокое или низкое значение. Например, диск, свободно вращающийся на 360 градусов может выводить значения от 1 до 10. Если wrap включен, то после 10 позиции увеличивающееся значение будет 0.
4	Линейная | Нелинейная	0 | 1	Показывает, исходные данные от устройства были обработаны в некотором виде и больше не представляют собой линейную зависисмость между тем, что измеряют и тем что сообщают. Ускоряющие кривые и мертвые зоны джойствика пример такого типа данных. Настройки чувствительности будут иметь эффект в элементе, но данные все равно будут линейными.
5	Preferred State | No	0 | 1	Показывает, имеет ли элемент управления предпочтительное состояние, в которое он будет

	Preferred		возвращен, когда юзер физически не взаимоействует с управлением. Примером являются самоцентрируемые джойстики.
6	No Null Position | Null State	0 | 1	Показывает, есть ли у управления положение, в котором он не посылает значащие данные. Один из вариантов использования Null положения для управления, подразумевающего физическое взаимодействие пользователя с управлением с тем чтобы отправлять полезные дынные. Например, некоторые джойстики имеют многонаправленные переключатели (hat переключатели). Когда hat переключатель не нажат он в Null состоянии. В этом случае устройство посылает значение находящееся за пределами логического минимума и максимума (например, -128) .
7	неизменяем | изменяем	0 | 1	Показывает, является ли значение управления Featute или Output изменяемым хостом или нет. Изменяемый Output может меняться и без влияния хоста. Для избежания проблем синхронизации, изменяемое управление должно быть относительным. Иначе при выдаче Output, запрос установит значение любого управления, которого вы не хотите менять на значение за пределами лог мин и макс.
	Reserved	0	Данные бита 7 незаданы для входа и зарезервированы для будущего использования.
8	Bit Field | Buffered Bytes	0 | 1	Отображает, что управление выделяет поток байтов фиксированного значения. Содержимое полей данных определяется приложением. Содержимое буфера не интерпретируется, как одиночное числовое значение. Отчетные данные, определенные буферизаций байтовых элементов, должны быть вровненны по 8 битной границе. Данные, полученные с ридера штрих кодов являются примером
9-31	Зарезерв	0	Для будущего использования.

Замечания

- Если **Input** элемент – массив, то доступны только данные/константа, переменная/массив и абсолютные/относительные атрибуты.

- Число полей данных элемента может быть определенно анализом **Report Size и Report Count** значений. Например, элемент с **Report Size** 8 bits и **Report Count** 3, имеет 3 8-битных поля данных.
- Значение, возвращаемое **Array** элементом – индекс, поэтому рекомендуется
  1) поле массива возвращает 0 при отсутствии управления в утвержденном массиве
  2)The Logical Minimum = 1.
  3) The Logical Maximum = числу элементов в массиве.
- **Input** элементы определяют входные отчеты доступными через канал **Control** с запросом **Get_Report (Input).**
- **Input** тип отчетов также отправляются через **Interrupt In**.
- **данные/константа, переменная/массив, абсолютные/относительные, нелинейн, Wrap, Null состояние Output** элемента идентичны данным для **Input** элемента.
- **output** элементы делают output отчеты доступными через канал **Control** с командой **Get_Report (Output).**
- Тип отчетов **Output** может дополнительно отправляться через Interrupt Out канал.
- И хотя они похожи по функциям, Output и Feature элементы различаются в следующем:
  - **Feature** элемент определяет конфигурации для устройства и устанавливается обычно панелью управления приложения. Так как они влияют на поведение устройства, **Feature** элементы обычно не видимы для программного обеспечения приложения, наоборот **Output** элементы представляют устройство на выход пользователю (например аудио, тактильная отдача, светодиоды). Для программных приложений лучше устанавливать **Output** элементы устройства.
  - **Feature** элементы могут быть атрибутами для других элементов. Например, Origin Reset Feature может обратиться в большое количество **Input** элементов. **Feature** элементы делают Feature отчеты доступными через канал Control с запросом **Get_Report (Feature)** и **Set_Report (Feature).**

6.2.2.6 Collection, End Collection элементы

Описание

Collection элементы определяют отношения между двумя или более данными (**Input, Output или Feature**). Например мышь можно описать, как набор от двух до четырех данных (x, y, кнопка 1, кнопка 2). Когда **Collection** элемент открывает набор данных, **End Collection** элемент его закрывает.

Части

Тип набора	Переменная	Описание
Physical (физическ)	0x00	Физический набор исользуется для задания элементов, которые представляют данные точек набота в виде одной геометорической точки. Это полезно для устройств с датчиками, которые нуждаются в сопоставлении данных измерений с одиночной точкой. Это не показывает, что множество данных пришло с одного устройства, такого как клавиатура. В случае, если устройство отправляет позицию от некоторого числа сенсоров, физическая коллекция служит для отображения, какие данные пришли с какого датчика.
Application	0x01	Группа Main элементов может быть знакома с приложением. Она

(приложен)		может также быть использована для опознавания элементов, сулжащих разным целям в устройстве. Типичными примерами является клавиатура или мышь. Клавиатура с встроенным указующим устройством может быть определена для двух разных наборов приложений. Данные отчета обычно связаны с набором приложения (как минимум один report ID на приложение
Logical (логическ)	0x02	Логический набор используется, когда множество данных элементов имеют стройную структуру. Например, связь между буфером данных и байтом счетчика данных. Набор устанавливает связь между счетчиком и буфером.
Report (отчет)	0x03	Определяет логический набор, обертывающий все поля в отчете. Приложение может легко определить, поддерживает ли устройство определенные функции. Помните, что любой допустимый Report ID может быть объявлен для набора отчета
Named Array	0x04	Именнованый массив это логический набор, содержащий массив выборщика использований. Для данной функции множество выборщиков для схожих устройств может изменяться. Именование полей является обычной практикой при документировании аппаратных регисторов. Для определения, является ли устройство поддерживающим определенный функции (например статуса), приложение должно отправить запрос всем известным выборщикам использований, до того, как определит, является ли устройство поддерживающим данный статус. Использования Именного Массива позволяют поле массива, содержащее выбощиков для названий, таки образом приложению нужно лишь отправить запрос статусному использованию для определения, что устройство поддерживает статусную информацию
Usage Switch	0x05	Usage Switch – логический набор, который модифицирует смысл использований, которых он содержит. Этот тип набора показывет приложению, что использование, найденное в наборе должно быть специально рассмотрено. Например, вместо объявления использования на светодиодной странице для всех возможных функций, отображение использования может быть идентифицированы в качетсве Usage Switch набора и стандартные использования, определенные в наборе, могут быть идентифицированы как индикаторы. Запомните, что данный тип набора не используется для отображения Порядковых наборов - логический набор создан для этого
Usage модификатор	0x06	Модифицирует значение использования, прикрепленного к охватывающему набору. Использование обычно определяет единый режим работы для управления. Usage модификатор

позволяет продлить режим управления. Например, LED обычно либо включена либо нет. Для определенных состояний устройство может генерировать мигания или выбор цвета. Дополнение LED использованя к Usage модификаторной коллекции, будет показывать приложению, что использование поддерживает новую оперативную модель.

Зарезерв	0x07-0x7F	Для будущего использования
	0x80-0xFF	Для определения Vendor

Замечания

- Все **Main** элементы, находящиеся между **Collection** и **End Collection** включены в набор. В набор могут входить и другие вложенные наборы.
- Collection элементы не создают данные. Однако тег **Usage** элемента должен связываться с любым набором (мышь или ручка газа). **Collection** элементы могут быть вложенными и они всегда дополнительные, кроме набора для верхнее уровневого приложения.
- Если встречаются неизвестные Vendor определенные наборы, то приложение игнорирует все главыне элементы в наборе. Запомните, что глобальные элементы в наборе повлияют на таблицу состояний.
- Если к известному набру присоединен неизвестное использование, то содержимое набора будет игнорироваться. Запомните, что глобальные элементы в наборе повлияют на таблицу состояний.
- String и Physical показатели, также как разделители могут быть связаны с коллекцией.

6.2.2.7 Глобальные (Global) элементы

Описание

Global элементы описывают, а не определяют даны от управления. Новый **Main** элемент предполагает характерисики таблицы состояний элементов. **Global** элементы могут менять таблицу состояний. В результате тег **Global** элемента распространяются на все впоследствии определенные элементы, если не переопределяются другим **Global** элементом.

Части

Тег Global элемента	Одно байтовый префикс (nn представляет размер знач)	Описание
Usage Page	0000 01 nn	Безнаковое целое, определяющее текущее Usage Page. С использованием 32 битных значений Usage Page элементы испльзуются для экономии места в отчетном дескрипторе, путем установки 16 старших битов для последующего использования. Любое использование, следующее за ним и определяющееся 16 битами о меншье, считаются Usage ID и соединяются с Usage Page до 32 битов

Logical Minimum	0001 01 nn	Объем значений в логических единицах. Это минимальное значение, которое переменная или массив будет сообщать. Например, мышь сообщающая X координату от 0 до 128 имеет лог минимум 0 и максимум 128
Logical Maximum	0010 01 nn	Объем значений в логических единицах. Это максимальное значение, которое сообщает переменная или массив
Physical Minimum	0011 01 nn	Минимальное значение для физически заданных переменных. Соответствует лог минимуму
Physical Maximum	0100 01 nn	Максимальное значение для физически заданной переменной
Unit Exponent	0101 01 nn	Значение единицы показателя по основанию 10
Unit	0110 01 nn	Unit Значения
Report Size	0111 01 nn	Беззнаковое целое, сообщающее размер в битах поля отчета. Это позволяет парсеру создать карту элементов для отчетного обработчика
Report ID	1000 01nn	Беззнаковое значение указывающее Report ID. Если тег Report ID используется где-нибудь в Отчетном дескрипторе, то все отчеты данных для устройства предшествуют полю с одним байтом ID. Все элементы последующие первому Report ID тегу, но предшествующие второму Report ID тегу, включены в отчет с префиксом в виде 1-байта ID.Все элементы между 2 и 3 Report ID тегами включены в второй отчет с префиксом 2-байта ID, и т.д. Это значение Report ID показывает префикс для кокретного отчета. Например, отчетный дескриптор может определять 3-байтный отчет с Report ID = 01. Это устройсто сгенерирует 4-байта информации, где первый байт 01. Устройство также сгенерирует другие отчеты с уникальными ID. Это позволяет хосту различать различные отчеты, прибывающие через одиночный Interrupt In канал. И позволяет устройству различать разные типы отчетов прибывающих через одиночный канал Interrupt out. Report ID 0 зарезервировно и не должно использоваться.

Report Count	1001 01nn	Беззнаковое целое, указывает число полей данных для элемента. определяет, какое число полей включены в отчет для конкретного элемента.
Push	1010 01nn	Помещает копию глобальной таблицы состояния элементов в стек.
Pop	1011 01nn	Перемещает таблицу состояний элементов с верхней структурой из стека.
Зарезервировано	1100 01nn до 1111 01nn	Промежуток зарезервировани для будущего использования.

См. также
Для списка Usage Page тегов смотри HID Usage Table документ.

Примечания

- Хотя Logical Minimum и Logical Maximum связывают значения, возвращеные устройством, Physical Minimum и Physical Maximum дают смысл этим границам, позволяя значениям в отчете компенсироваться и масштабироваться. Например, термометр должен иметь логические степени от 0 до 999, но физические степени от 32 градусов до 212 градусов. Соотношение может быть определено следующим алгоритмом:

```
if ((Physical Maximum == UNDEFINED)
|| (Physical Minimum == UNDEFINED)
|| ((Physical Maximum == 0) && (Physical Minimum == 0)))
{
Physical Maximum = Logical Maximum;
Physical Minimum = Logical Minimum;
}
If (Unit Exponent == UNDEFINED)
Unit Exponent = 0;
Resolution = (Logical Maximum – Logical Minimum) /
((Physical Maximum – Physical Minimum) *
(10 Unit Exponent))
```

При линейном разборе отчетного дескриптора, глобальные значения Unit Exponent, Physical Minimumи Physical Maximum считаются «неопределенными», пока не будут объявлены Например, мышь с чувствительностью 400dpi может иметь элементы, как показано в след таблице:

Элемент	**Значение**
Logical	Minimum -127
Logical	Maximum 127
Physical	Minimum -3175
Physical	Maximum 3175
Unit Exponent	-4
Unit	Inches

Таким образом, формула для вычисления отношения должна быть
Соотношение = (127-(-127)) / ((3175-(-3175)) * 10^-4) = 400 вычислений на дюйм
Unit элемент квалифицирует значения, как указано в следующей таблице:

Nibble	Система	0x0	0x1	0x2	0x3	0x4
	Степень	0	1	2	3	4
0	System	Нет	SI Linear	SI Rotation	English Linear	English Rotation
1	Length	Нет	Centimeter	Radians	Inch	Degrees
2	Mass	Нет	Gram	Gram	Slug	Slug
3	Time	Нет	Seconds	Seconds	Seconds	Seconds
4	Temperature	Нет	Kelvin	Kelvin	Fahrenheit	Fahrenheit
5	Current	Нет	Ampere	Ampere	Ampere	Ampere
6	Luminous intensity	Нет	Candela	Candela	Candela	Candela
7	Reserved	Нет	Нет	Нет	Нет	Нет

Замечание: для системной части коды 0x5 – 0xE зарезервированы, 0xF определяются поставщиком.

- если оба **Logical Minimum** и **Logical Maximum** определены как положительные значения, тогда поле отчета может значиться как беззнаковое значение. В общем случае все целые знаковые, представленные в дополниельном коде.
- Пока **Physical Minimum** и **Physical Maximum** объявлены в Отчетномдескрипторе, они принимаются HID парсером равными **LogicalMinimum и Logical Maximum,** соответственно. После объявления их так, что они могут применяться к (Input, Output или Feature) Main элемента, они продлеваются до эффекта всех последующих main элементов. Если оба **Minimum** и **Physical Maximum** равны 0, они будут возвращены к значениям по умолчанию.
- коды и экспоненты, не показанные в предыдущей таблице:

Коды	Степень
0x5	5
0x6	6
0x7	7
0x8	-8
0x9	-7
0xA	-6
0xB	-5
0xC	-4
0xD	-3
0xE	-2
0xF	-1

- Большинство сложных элементов может быть выведены из основных единиц длины, массы, времени, температуры, силы тока и света. Например энергия (джоуль) может быть представлена в виде

joule =[mass(grams)][length(centimeters)^2][time(seconds)^-2]

Степень этого элемента будет 7, т.к. джоуль состоит из килограммов и метров. Например, рассмотрим следующее:

Nibble	**Часть**	**Значение**
3	Time	-2
2	Mass	1
1	Length	2
0	System	1

- Части некоторых общепринятых единиц показаны в следующей таблице.

Элемент	**Nibbles**						
	5 (i)	**4 (τ)**	**3 (t)**	**2 (m)**	**1 (l)**	**0 (sys)**	**Code**
Расстояние (cm)	0	0	0	0	1	1	x0011
Масса (g)	0	0	0	1	0	1	x0101
Время (s)	0	0	1	0	0	1	x1001
Скорость (cm/s)	0	0	-1	0	1	1	xF011
Момент	0	0	-1	1	1	1	xF111
Ускорение	0	0	-2	0	1	1	xE011
Сила	0	0	-2	1	1	1	xE111
Энергия	0	0	-2	1	2	1	xE121
Угловое ускорение	0	0	-2	0	1	2	xE012
Напряжение	-1	0	-3	1	2	1	x00F0D121

- В случае массива **Report Count** определяет максимальное число элемента управления, которое может быть включено в отчет и, следовательно, число клавиш и кнопок, которые могут одновременно быть нажаты, также как и размер каждого элемента. Например: массив поддерживает до трех одновременно нажатых клавиш, где каждое поле 1 байт, выглядит следующим образом:

```
...
Report Size (8),
Report Count(3),
...
```

В случае переменного элемента, **Report Count** указывает, как много управлений включены в отчет. Например, 8 кнопок могут выглядеть так:

```
...
Report Size (1),
Report Count (8),
...
```

- Если **Report ID** используется, тогда **Report ID** должен быть объявлен перед первым Input, Output, или Feature Main элементом.
- Одинаковое **Report ID** значение может встречаться более одного раза в отчетном дескрипторе. В последствии заданные Input, Output или Feature main элементы бдут найдены в соответстующих ID/Type (Input, Output или Feature) отчетах.

6.2.2.8 Локальные (Local) элементы

Описание

Теги **Local** элемента определяют характеристику контроля. Эти элементы не переносятся через следующий **Main** элемент. Если **Main** элемент определяет больше чем одно управление, он может предшествовать нескольким схожим тегам **Local** элемента. Например, Input элемент может иметь несколько тегов **Usage**, связанных с ним, один на каждое управление

Части

Тег	Одно байтный префикс	Описание
Usage	0000 10 nn	Usage индекс для использования элемента. представляет предложенное использование для набора или элемента. В случае когда элемент представлет некоторое число управлений, тег Usage может предлагать использование для любой переменной или элемента в массиве
Usage Minimum	0001 10 nn	Задает начальное использование, связанное с массивом или бит-картой
Usage Maximum	0010 10 nn	Задает конечное использование, связанное с массивом или бит-картой
Designator Index	0011 10 nn	Определяет часть тела, используемую для управления. Индексные точки для назначения в физическом дескрипторе
Designator Minimum	0100 10 nn	Определяет индекс начала указателя, связанный с массивом или бит-картой.
Designator Maximum	0101 10 nn	Определяет индекс конца указателя, связанный с массивом или бит-картой.
String Index	0111 10 nn	Стартовый индекс для String дескриптора, позволяет связать строку с конеретным элементом или управлением.
String Minimum	1000 10 nn	Указывает первый строчный индекс при назначении группы последовательных строк для контроля в массиве или бит-карте.
String Maximum	1001 10 nn	Указывает последнийстрочный индекс при назначении группы последовательных строк для контроля в массиве или бит-карте.
Delimiter	1010 10 nn	Указывает начало или конец множества локальных элементов (1 = открыть множество, 0 = закрыть).
Зарезервировано	1010 10 nn to 1111 10 nn	Зарезервировано.

Примечания

- Хотя **Local** элементы не переходят через следующий **Main** элемент, они могут отвечать более чем одному управлению внутри одного элемента. Например, если **Input** элемент определяющий пять управений, предшествует трем **Usage** тегам, три использования будут приписаны соответственно к первым трем управлениям и трете использование

будет также приписано к пятому и четвертому управлениям. Если управлений у элемента нет, тогда тег **Local** элементаприменяется к **Main** элементу.

- Чтобы присвоить уникальные использования к каждому управлению в одиночном **Main** элементы, просто укажите каждый тег **Usage** последовательно (или используйте **Usage min или Usage Max**)
- все Local элементы беззнаковые целые.

замечание: важно использовать **Usag**e правильно. Хотя существуют специфичные использования, эти использования нацелены на распознавания устройствами с специфической программной. Джойстику с общими кнопками никогда не следует назначать приложение с специфическим использованием для любой кнопки. Вместо этого ему следует назначить общее использование, например “кнопка”. Однако устройства могут сами назначать функции для каждого источника информации.

-Также важно знать, что Usage элементы передают информацию о цели использования данных и могут не отвечать тому, что они на самом деле измеряют. Например, джойстик имеет X и Y Usage, связанные с осями данных.

См. также
Для списка частей Usage смотри Дополнение A

- Поскольку массивы и бит-карты кнопок представляют множество кнопок или переключателей с одиночным элементом, может быть полезно назначитьмножество использований для Main элемента. Usage Minimum указывает использование, которое связано с первым несвязанным управлением в массиве или бит-карте. Usage Minimum указывает конец промежутка использования, которое связано элементами элемента. Следующий примерпоказывает, как это может быть использовано для 105 клавишной клавиатуры.

Тег	**Результат**
Report Count (1)	Одно поле будет добавлено в отчет.
Report size (8)	Размер этого поля 1 байт.
Logical Minimum (0)	определяет 0 для нижайшего возможого значения.
Logical Maximum (101)	Определяет 101 для высочайшего возможного значения и устанавливает промежуток 1-101.
Usage Page (0x07)	Выбирает страницу использований клавиатуры.
Usage Minimum (0x00)	Назначает использования младших 101 клавиш.
Usage Maximum (0x65)	Назначает использования старших 101 клавиш.
Input: (Data, Array, Absolute)	Создает и добавляет 1 байтовый массив к входному отчету..

- если **Usage Minimum** объявлен как расширенное использование, тогда соответствующий **Usage Maximum** также должен быть расширенным
- понимание **Usage, Usage Minimum или Usage Maximum** элементов варьируется как функция от bSize поля. Если bSize =3 то элемент понимается как 32 битный беззнаковый, где старшие 16 бит определяют **Usage Page**, а младшие определяют **Usage ID**. 32 разрядные Usage элементы, определяющие Usage Page и Usage ID, называются «расширенными» Usages.
  Если bSize = 1 или 2, тогда **Usage** представляется как беззнаковое значение, выбирающее **Usage ID** на уже заданное **Usage Page**. Когда парсер встречает Main элемент он

объединяет последние заданные Usage Page с Usage и последнее Usage значение. Расширенные использования могут испльзоваться для переопределения уже заданных Usage Page для индивидуальных использований.

- Два или более альтернативных использования могут быть связаны с управлением простым брекетингом с Delimiter элементом. **Delimiter** позволяет псевдонимам быть определенными для управления, так что приложение может достучаться до негоразличными способами. Использования, которые формируют набор разделителей, организованы в порядке предпочтения, где первое использование является более предпочтительным для управления
- HID анализаторы должныобрабатывать **Delimiter**. Тем не менее, поддержка альтернативных использований, которые они определяют, дополнительна. Использования, кроме первого,могут быть недоступны системным программным обеспечением.

6.2.2.9 Заполнения (Padding)

Отчеты могут быть дополнены до байтового поля, путем объявления соответствующего размера Main элемента и не объявления использования для Main элемента.

6.2.3 Физические (Physical) дескрипторы

Физические дескрипторы – это структура данных, которая содержит информацию о конкретной части или частях человеческого тела, которые активируют управление или управления. Например, физический дескриптор может означать, что большой палец правой руки используется для активации кнопки 5. Приложение может использовать эту информацию, чтобы назначить функциональные возможности для контроля работы устройства.

Примечание: Физические Дескрипторы совершенно являются дополнительными. Они добавляют сложности и очень мало предлагают в обмен на большинстве устройств. Тем не менее, используя некоторые устройства, особенно с большим числом одинаковых элементов управления (например, кнопки), вы увидите, что Физические Дескрипторы помогают различным приложениям назначить функциональные возможности для этих элементов управления в более последовательном виде. Пропустите следующий раздел, если вы не планируете поддержку физических дескрипторов.

Подобные **физические Дескрипторы** сгруппированы в множества. **Designator Index** элементы, содержат в **Отчетном** дескрипторе карту элементов (или управлений) для конкретныч **физических дескрипторов**, содержащихся в физическом набор дескрипторов..

Каждое дескрипторное множество состоит из коротких заголовков , следующих за одним или более **физическим дескриптором**. Заголовок определяет **Смещение (Bias)** (если дескрипторный набор ориентирован на праворукого или леворукого) и **Предпочтения (Preference**) множества. Для смещения, поизводитель может определить альтернативные **Физические дескрипторы** (например, праворукий пользователь может держать устройство более чем одним способом, поэтому необходимо переназначение пальцев, которые касаются отдельных элементов).

Каждый Физический дескриптор состоит из следующих трех полей

- **Указатель** – определяет фактическую часть тела, влияющую на элемент, например, рука
- **Квалификатор** – определяет позже указатель (левая или правая рука)
- **Усилие** – значение количественного усилия, для взаимодействия с элементом

Если несколько элементов определяют одну и ту же **Указатель/Квалификатор** комбинацию, то значение **Усилия** может быть использовано для разрешения назначения

функций. Значение **Усилия** = 0 будет использовано для определения использования пальца как кнопки. Значение Усилия увеличиваются, если пальцу необходимо тянуться для получения управления.

Только в одном случае два или более управлений будут иметь одинаковую комбинацию **Указатель/Квалификатор/Усилие**, так как они физически соединены вместе. Хороший пример это длинный тонкий переключатель с + на одном конце и – на другом. Если возможно електически сделать две дискретные кнопки, то возможно иметь двойное нажатие. Если производитель решает, что нажатие + и – одновременно возможно, тогда они будут рассматриваться как две дискретные кнопки с идентичным Физическим дескриптором. Однако, если кнопка называется «громкость», то нажатие обеих кнопок в одно и тоже время бессмысленно, тогда производителю необходимо расписать кнопки как одиночные элементы с тремя возможными статусами: выключен, больше громкость, меньше громкость. В этом случае будет нужен только **физический дескриптор**.

Рассмотрим джойстик с двумя кнопками (A и B) на левой стороне устройства и триггер кнопку (гашетку) на передней стороне рычага, который логически Ored с кнопкой A. Джойстик чаще всего держат в левой руке, манипулируя рычагом правой. Поэтому первый дескриптор будет назначать кнопку Акак
Index Finger, Right, Effort 0
кнопка B как
Thumb, Left, Effort 0

Если джойстик был помещен на стол и левая рука управляет обеими кнопками, то другой дескриптор должен идентифицироваться другой картой для A
Middle Finger, Left, Effort 0
И для B:
Index Finger, Left, Effort 0

Важно: теги **указателя** дополнительные и могут быть предоставлены для всех, некоторых или ни для одного элемента устройства.

Дескрипторное множество 0 это специальное множество, которое определяет количество дополнительных дескрипторных множеств и также количество Физических дескрипторов в каждом множестве.

Часть	**смещение/размер**	**Описание**
bNumber	0/1	Числовое выражение, определяющее число множеств физических дескрипторов. Не включает физический дескриптор 0 в этом числе
bLength	1/2	Числовое выражение, определяющее длину каждого физического дескриптора.

После получения запроса **Get_Descriptor** от хоста, устройство **HID** класса возвращает дескрипторное множество, обозначенное в младшем байте *wValue*. Дескрипторное множесвто состоит из заголовка, следующего за одним или большим количесвтом **Физических Дескрипторов.**

Устройство **HID** класса использует следующий формат для его **Физического** дескриптора.

Часть	**Смещение/Размер**	**Описание**
bPhysicalInfo	0/1	Биты описывающие физическую информацию: 7..5 Bias (смещение) 4..0 Предпочтительные 0 = Наиболее предпочтительный
dPhysical	1/2	Данные физического дескриптора, индекс 1.

dPhysical	3/2	Данные физического дескриптора, индекс 2.
dPhysical	(n*2)-1/2	Данные физического дескриптора, индекс n.

Примечания:

- Поле **Смещения** отображает, какую руку характеризует дескрипторное множество. Может не работать с некоторыми усройствами

Значение смещения	**Описание**
0	Не доступно
1	Правая
2	Левая
3	Обе
4	Любая
5	Резервировано
6	Резервировано
7	Резервировано

Замечание Устройство для использования только в правой руке не вернет дескрипторное множество с **смещением** левой руки.

- Поле **Предпочтения** показывает, содержит ли дескрипторное множество предпочтитеьную или другую указывающую информацию. Производитель укажет значение **Предпочтения** 0 для более предпочтительных физических данных. Старшие значения **Предпочтеняия** отображают менее предпочтительное дескрипторное множество.
- **Физический дескриптор** внутри дескрипторного множества, ссылается благодаря **Designator Index** элементу в отчетном дескрипторе.
- **Физический дескриптор** имеет следующие части:

Часть	**Смещение/Размер**	**Описание**
bDesignator	0/1	Указывающее значение, указывает на часть тела, взаимодействующую с элементом
bFlags	1/1	Определяющие биты флаги: 7..5 Квалификатор 4..0 Усилие

Значение смещения	**Описание**
00	None
01	Hand
02	Eyeball
03	Eyebrow
04	Eyelid
05	Ear
06	Nose
07	Mouth
08	Upper lip
09	Lower lip

0A	Jaw
0B	Neck
0C	Upper arm
0D	Elbow
0E	Forearm
0F	Wrist
10	Palm
11	Thumb
12	Index finger
13	Middle finger
14	Ring finger
15	Little finger
16	Head
17	Shoulder
18	Hip
19	Waist
1A	Thigh
1B	Knee
1C	Calf
1D	Ankle
1E	Foot
1F	Heel
20	Ball of foot
21	Big toe
22	Second toe
23	Third toe
24	Fourth toe
25	Little toe
26	Brow
27	Cheek
28-FF	Reserved

Поле **Квалификатора** отображает руку (или часть тела) с которой указатель взаимодействует. Может не работать для некоторых утсройств

Значение квалификатора	**Описание**
0	Не доступно
1	Правая
2	Левая
3	Обе
4	Любая
5	Центр
6	Резервировано
7	Резервировано

- Поле усилия отображает, насколько легко юзеру производить контроль. Значение 0 означает, что юзер может легко и быстро осуществлять контроль. Чем больше значение, тем сложнее.

7. Запросы

7.1 Стандартные запросы

HID класс использует стандартные запросы **Get_Descriptor,** как описано в USB спецификации. Когда **Get_Descriptor(Configuration)** запрос выдается, возвращается Конфигурационный дескриптор, все **интерфейсные** дескрипторы, все **Endpoint** дескрипторы и все **HID** дескрипторы для каждого интерфейса. Не должен возвращаться **String** дескриптор, **отчетный HID** дескриптор или любой дополнительный **HID** класс дескриптор. HID дескриптор должен чередоваться между **Интерфейсным** и **Endpoint** дескриптором HID интерфейса. Таким образом, порядок должен быть вида:

Configuration descriptor
  Interface descriptor (specifying HID Class)
    HID descriptor (associated with above Interface)
      Endpoint descriptor (for HID Interrupt In Endpoint)
        Optional Endpoint descriptor (for HID Interrupt Out Endpoint)

Замечание: Get_Descriptor может быть использован для получения стандартного, классового и vendor специальных дескрипторов, в зависимотсти от поля **Descriptor Type**

См. также
Для деталей смотри часть 9 USB спецификации

Примечания

Следующая таблица определяет (**Descriptor Type**) **Тип Дескриптора** (старший байт wValue в запросе Get_Descriptor)

Часть	**Описание**
Тип дескриптора	Биты определяющие характеристику типа дескриптора 7 Резервировано (всегда должно быть 0) 6..5 Тип 0 = Стандарт (Standart) 1 = Класс (Class) 2 = Производитель (vendor) 3 = Резервировано 4..0 Дескриптор Смотри стандартный класс или таблицы производителя Типа дескриптора.

Далее определяются доступные типы Class дескриптора

Значение	**Типы Class дескриптора**
0x21	HID
0x22	Отчетный
0x23	Физический дескриптор
0x24 - 0x2F	Резервировано

7.1.1 Get_Descriptor запрос

Описание

Get_Descriptor запрос возвращает дескриптор устройства.

Части

Часть	**Стандартный USB дескриптор**	**HID Класс дескриптор**
bmRequestType	100 xxxxx	10000001
bRequest	GET_DESCRIPTOR (0x06)	GET_DESCRIPTOR (0x06)
wValue	дескриптор Type и дескриптор Index	дескриптор Type и дескриптор Index
wIndex	0 (zero) or Language ID	Interface Number
wLength	дескриптор Length	дескриптор Length
Data	дескриптор	дескриптор

Примечания

- Для стандартного USB дескриптора биты 0-4 bmRequestType показывают с чем связан запрашиваемый дескриптор – с устройством, интерфейсом, конечной точкой или другим
- Поле *wValue* определяет дескриптор типа в старшем байте и **Index дескриптор** в младшем.
- **Index Дескриптор** используется для определения множества для **Физического дескриптора** и сбрасывается в ноль при другом дескрипторе HID класса.
  - Если дескриптор **HID** класса запрашвался, то *wIndex* отображает число HID интерфейсов. Если стандартный дескриптор запрашивался, то *wIndex* оопределяет Языковый ID для строчного дескриптора и сбрасывается в ноль для других стандартных дескрипторов.
  - Запрашивание **Физического дескриптора** множества 0, возвращает специальный дескриптор, определяющий число дескрипторных множеств и их размеры.
  - Запрос **Get_Descriptor** с **Физическим Index** равным 1, вызовет первое множество **Физических дескрипторов**. Устройство может по-своему использовать его элементы. Они могут быть пересчитаны с помощью последующего запроса **Get_Descriptor с** инкрементированным **дескриптором Index**. Устройство возвратит последнее дескрипторное множество на запросы с индексом большим, чем число последне определенных **HID** дескрипторов

7.1.2 Set_Descriptor запрос

Описание

Set_Descriptor запрос позволяет хосту менять дескрипторы в устройстве. Поддержка данного запроса дополнительна**.**

Части

Часть	Стандартный USB дескриптор	HID Класс дескриптор
bmRequestType	00000000	00000001
bRequest	SET_DESCRIPTOR (0x07)	SET_DESCRIPTOR (0x07)
wValue	дескриптор Type(старший) и дескриптор Index (младший)	дескриптор Type и дескриптор Index
wIndex	0 (zero) or Language ID	Interface Number
wLength	дескриптор Length	дескриптор Length
Data	дескриптор	дескриптор

7.2 Классовые (Class-Specific) запросы

Описание

Классовые запросы позволяют хосту узнавать о возможностях и состоянии устройства и устанавливать output и feature элементов. Они сообщаются через **Стандартный** (**Default**) какнал, и, следовательно, в соответствии с форматом канала создают запросы как описано в USB спецификации.

Части

Часть	Смещение/Размер	Описание
bmRequestType	0/1	Биты описывающие характеристику запроса: 7 Направление потока данных 0= с хоста на устройство 1 = С устройства на хост 6..5 Тип 1=Класс 4..0 Получатель 1=Интерфейс
bRequest	1/1	Специальный запрос
wValue	2/2	Числовое выражене, определяющее размер поля
wIndex	4/2	Индекс или смещение, определяющее размер поля
wLength	6/2	Числовое выражене, определяющее число байтов для передачи в фазе данных

Примечания

Следующая таблица определяет доступные значения *bRequest*

Значение	Описание
0x01	GET_REPORT (является обязательным для всех устройств)
0x02	GET_IDLE
0x03	GET_PROTOCOL(обязательный только для загрузочных устройств)
0x04-0x08	Резервировано
0x09	SET_REPORT
0x0A	SET_IDLE
0x0B	SET_PROTOCOL (обязательный только для загрузочных устройств)

7.2.1 Get_Report запрос

Описание

Get_Report запрос позволяет хосту принимать заопросы через канал **Управления** (**Control**).

Части

Part	**Description**
bmRequestType	10100001
bRequest	GET_REPORT
wValue	Тип отчета и Report ID
wIndex	Интерфейс
wLength	Отчет Length
Data	Отчет

Примечания

- Поле *wValue* определяет **Тип отчета** в старшем байте и **Report ID** в младшем.Установите **Report ID** в 0 если **Report ID** не используется. **Тип отчета** определяется следующим образом**:**:

Значение	**Тип отчета**
01	Input
02	Output
03	Feature
04-FF	Reserved

- Этот запрос полезен во время инициальизации для абсолютных элементов для определения состояния Feature элементов. Этот запрос не предназначен для опроса устройства на постоянной основе.
- Канал **Interrupt In** должен использоваться для текущих **Input** отчетов. Ответ **Input** отчета имеет тот же формат, что и отчеты канала **Interrupt.**
- Канал **Interrupt Out** может дополнительно использоваться для малой задержки **Output** отчетов.**Output** отчеты идущие через канал **Interrupt Out** имеют формат, идентичный отчетам через канал Управления, если конечная точка **Interrupt Out** не задана.

7.2.2 Set_Report запрос

Описание

Set_Report запрос позволяет хосту отправлять запрос стройству, по возможноти меняя состояния input, output или feature управлений.

Части

Часть	**Описание**
bmRequestType	00100001
bRequest	SET_REPORT
wValue	Тип отчета and Report ID
wIndex	Интерфейс
wLength	Отчет Length
Data	Отчет

Примечания

- Смысл полей запроса для **Set_Report** идентичный **Get_Report** запросу, однанако, направление данных противоположное и Отчетные Данные посылаются с хоста на устройство.
- Устройство может игнорировать **Set_Report** запросы, как бессмысленные. Иначе эти запросы могут быть использованы для сброса оригинального управления (т.е. текущая позиция должна отсылать ноль). Эффект от отчетов будет также зависеть от получателя управления – относительного или абсолютного.

7.2.3 Get_Idle Запрос

Описание

Get_Idle запрос читает текущий процент бездействия системы для каждого **Input** отчета.

Части

Части	**Описание**
bmRequestType	10100001
bRequest	GET_IDLE
wValue	0 и Report ID
wIndex	Интерфейс
wLength	1
Data	Процент бездействия

Примечания

Для разяснений полей запроса смотри следующий раздел.

7.2.4 Set_Idle Запрос

Описание

Set_Idle запрос не отвечает на отчеты от **Interrupt In** канала, до тех пор, пока не совершиться определенное событие или не пройдет необходимое время.

Части

Часть	**Описание**
bmRequestType	00100001
bRequest	SET_IDLE
wValue	Продолжительность and Report ID
wIndex	Интерфейс
wLength	0
Data	Не доступно

Примечания

Этот запрос нужен для ограничения частоты опроса по прерыванию конечной точки. В частности, этот запрос является причиной значений NAK на все опросы по прерыванию в конечной точке, при неизменном текущем значении. При отсутсвии изменений опросы будут продолжаться пока не примут значение NAK. Этот запрос имеет следуюшие части:

Часть	**Описание**
Продолжительность	Когда старший байт wValue 0, продолжительность не определена. Конечная точка будет препятствовать отчетам всегда, пока не замечено изменение в отчетных данных Когда старший wValue байт не ноль, используется фиксированное значение продолжительности. продолжительность лнейно зависит от значений старшего байта с младшим битом эквивалентным 4 мс. Это позволяет создать промежуток 0.004 до 1.020 секунд с шагом 4 милисекунды. Если продолжительность меньше чем частота опроса, тогда отчеты генерируются с этой частотой. Если данная продолжительность проходит без изменений в отчетных данных, тогда одиночный отчет генерируется конечной точкой и передача замедляется, начиная новое использование предыдущей продолжительности. seconds, with a 4 millisecond resolution. If the duration is less than the device polling rate, then reports are generated at the polling rate.
Report ID	Если младший байт wValue ноль, то частота опроса применяется для всех входных отчетов, создаваемых устройством. Когда младший байт wValue іне ноль, тогда частота простоя применяется к Report ID, определенного значением младшего байта
Точность	Это временная продолжительность должна иметь точность +/-(10% + 2 мс)
Задержка	Новый запрос запустится, как если бы он был отправлен немедленно после предыдущего отчета, если новый запрос принимает хотябы 4 мс перед концом данного периода. Если новый запрос принимается быстрее чем за 4 мс до конца периода, тогда новый запрос не будет иметь эффекта пока не

закончится отчет.

Если текущий период прошел мимо новозаданного временного промежутка, то отчет будет создан немедленно.

Если interrupt in конечная точка обслуживает несколько отчетов, тогда **Set_Idle** запрос может использоваться для влияния на частоту с которой дублирующиеся отчеты генерируются для конкретного **Report ID**. Например, устройство с двумя входными отчетами будет определять частоту простоя 20 мс для report ID 1 и 500 мс для report ID 2.

Рекомендуемая настройка по умолчанию для частоты простоя (частоты, когда устройство инициализируется) - 500 мс для клавиатур и бесконечность для мыши и джойстика.

7.2.5 Get_Protocol запрос

Описание

Get_Protocol запрос читает, какой протокол сейчас активен (либо протокол загрузки либо отчетный протокол).

Части

Часть	**Описание**
bmRequestType	10100001
bRequest	GET_PROTOCOL
wValue	0 (zero)
wIndex	Интерфейс
wLength	1 (one)
Data	0 = Загрузочный протокол 1 = Отчетный протокол

Примечания

Этот запрос поддерживается устройствами **Загрузочного** подкласса. *wValue* поле задает, какой протокол должен использоваться.

7.2.6 Set_Protocol Запрос

Описание

Set_Protocol переключает между загрузочынм протоколом м отчетным пртоколом (или наоборот).

Части

Часть	**Описание**
bmRequestType	00100001
bRequest	SET_PROTOCOL
wValue	0 = Загрузочный протокол 1 = Отчетный протокол
wIndex	Интерфейс
wLength	0
Data	Не применимо

Примечания

Этот Запрос поддерживается устройствами загрузочного подкласса. *wValue* поле задает,

какой протокол должен использоваться.

При инициализации все устройства по умолчанию используют отчетный протокол. Однако хост не должен делать никаких предложений о статусе устройства и должен установить нужный протокол во время инициализации устройства.

8. Отчетный протокол (Report Protocol)

8.1 Отчетные типы

Отчеты содержат данные от одного или нескольких элементов. Потоки данных от устройства к хосту идут через канал **Interrupt In** в ворме отчетов. Отчеты могут также могут быть запрошены (опрошены) и отправлены через канал Управления (**Control)** или отправлены через дополнительный **Interrupt Out** канал. Отчет содержит состояние всех элементов (**Input**, **Output** или **Feature**), принадлежащих конкретному **Report ID.** Программное обеспечение отвечает за извлечение отдельных элементов из отчетов, основанных на **Отчетном дескрипторе.**

Все значения элементов упакованы в битовые границы в отчете (не байтовое или nibble выравнивание). Однако элементы, отправляющие Null или константу могут использовать значения, выровненные до байта, или **Размер отчета** может быть сделан больше, чем нужно для некоторых полей, просто для того чтобы распределить их на границе байта

Длина битов данных элемента получается с помощью **Отчетного** дескриптора (**Report Size** * **Report Count**). Данные элемента упорядочены как и в **Отчетном** дескрипторе. Если тег **Report ID** использовался в **Отчетном** дескрипторе, то все отчеты включают одинобайтовый ID префикс. Если тег **Report ID** не использовался, то все данные возвращаются в одиночном отчете и префикс ID не входит в этот отчет.

8.2 Формат отчета для стандартных элементов

Формат отчета состоит из 8 – битового отчетного идентификатора, следующего за данными, принадлежащими этому отчету

Report ID

Поле **Report ID** 8 бит в длину.Если не используются теги **Report ID** в **Отчетном** дескрипторе,то есть только один отчет и **Report ID** поле опущено.

Report Data

Поле данных это поле переменной длины, которое посылает состояние элемента.

8.3 Формат отчета для Array элементов

Каждой кнопке в массиве отчетов присвоен номер, называемый индекс массива. Он может быть переведен в код клавиши, путем просмотра элементов массива **Usage Page** и **Usage.** Если какая-либо кнопка находится в состоянии перехода между открыто и закрыто, весь список индексов для кнопок закрывается в массиве, перемещенном хосту.

Поскольку только один элемент может быть представлен в поле массива, клавиши-модификаторы должны быть отправлены в виде биткарты.(группы 1-битных переменных полей). Например, клавиши CTRL, SHIFT, ALT и GUI составляет модификационный байт в стандартном отчете клавиатуры. Хотя эти коды использований определены в таблице

использований как E0-E7, использование не отправляется как массив. Байт модификатор определяется следующей таблицей.

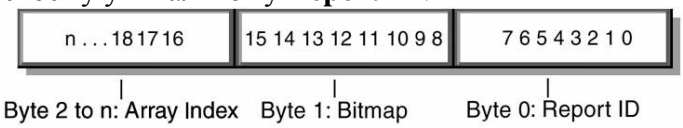
Бит	Клавиша
0	LEFT CTRL
1	LEFT SHIFT
2	LEFT ALT
3	LEFT GUI
4	RIGHT CTRL
5	RIGHT SHIFT
6	RIGHT ALT
7	RIGHT GUI

Следующий пример показывает отчет, созданный пользователем, набравшим ALT+CTRL+DEL, исользуя биткарту для модификаторов и одиночный массив для остальных клавиш.

Переход	**Модификационный байт**	**Байт массива**
LEFT ALT down	00000100	00
RIGHT CTRL down	00010100	00
DEL down	00010100	4C
DEL up	00010100	00
RIGHT CTRL up	00000100	00
LEFT ALT up	00000000	00

См. также
Для списка стандартных клавиатурных кодов клавиш смотри дополнение A

Если существует множество отчетов для устройство, каждый отчет будет предшествовать своему уникальному **Report ID**.



Если набор клавиш и кнопок не исключают друг друга, они должны быть представлены либо как биткарта, либо как некскольо массивов. Например, функционаьные кнопки на 101 клавишной клавиатуре иногда используются как модификаторные клавиши, например F1 A. В этом случае по крайней мере два поля массива должны быть представлены как Array элемент, т.е. **Report Count** (2).

8.4 Ограничения(Constraints) отчетов

- Следующие ограничения распространяются на отчеты и на использование отчетов
- Поле элемента не может занимать больше 4 байт в отчете. Например, 32 разрядный элемент должен стартовать с границы байта, чтобы удовлетворять условию.
- Только один отчет допускаеся в одиночном передатчике USB
- Отчет может охватывать одну или несколько USB транзакций. Например. приложение, имеющее 10-байтный отчет будет охватывать по крайней мере 2 USB транзации на low-speed устройстве.

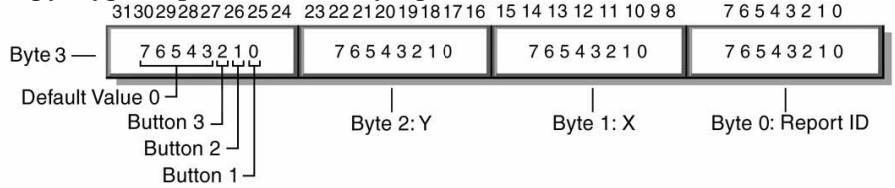
- Все отчеты за исключением превышающих *wMaxPacketSize* для конечной точки, должны завершиться коротким пакетом. Более длинный отчет не удовлетворят короткопакетоному завершителю.
- Каждый верхнеуровневый набор должен быть набором приложения и отчеты не должны охватывать больше чем 1 верхнеуровневый набор.
- Если есть несколько отчетов в верхнеуровневом наборе, тогда все отчеты, за исключением самого длинного должны закончиться коротким пакетом.
- Отчет всегда выровнен по байту. Если необходимо, отчеты дополнятются битами (0) пока не достигнется граница следующего байта,

8.5 Пример Отчета

Следующий отчетный дескриптор определяет элемент с **Input** отчетом

```
Usage Page (Generic Desktop),
Usage (Mouse),
Collection (Application),
        Usage (Pointer),
        Collection (Physical),
                Report ID (0A),                 ;Изменяет report 0A
                Usage (X), Usage (Y),
                Logical Minimum (-127),         ;Значения Report данных от -127
                Logical Maximum (127), ;to 127
                Report Size (8), Report Count (2),
                Input (Data, Variable, Relative),   ;добавляет 2 байта положения (У,Х) в report 0A
                Logical Minimum (0),            ; Значения Report данных от -127
                Logical Maximum (1),            ;до 127
                Report Count (3), Report Size (1),
                Usage Page (Button Page),
                Usage Minimum (1),
                Usage Maximum (3),
                Input (Data, Variable, Absolute),   ;добавляет 2 бита (кнопки 1, 2 и 3) в report 0A
                Report Size (5),
                Input (Constant),               ;добавляет  5 бит дополнения до байта в report 0A
        End Collection,
End Collection
```

Структура Input отчета для устройства выше показана ниже:



Следующая таблица использует клавиатуру с встроенным указывающим утсройством, для демонтрации, как пользоваться двумя отчетами для устройства с одним интерфесом.

Элемент	**Использование**	**Report ID**
Collection (Application)	Клавиатура	
Report ID (00)		00
Input (Variable, Absolute)	Клавиши-модификаторы	
Output (Variable, Absolute)	LED	00

Input (Array, Absolute)	Основные клавиши	00
End Collection		
Collection (Application)	мышь	
Report ID (01)		
Collection (Physical)	Указатель	
Input (Variable Relative),	X, Y	01
Input (Variable, Absolute)	Кнопка	01
End Collection		
End Collection		

Замечание: только **Input, Output и Feature** элементы (не **Collection** элементы) представляют данные в отчете. Этот пример демонстрирует множество отчетов, однако, этот интерфейс не будет оступен для **Загрузочного Устройства** (используйте отдельные интерфейсы для клавиатуры и мыши).

Приложение A: Теги Использований (Usage Tags)

Смотри USB Usage Tables документ для полного списка Usage Pages and **Usage Tags,** включая коды клавиш клавиатуры.

Приложение B: Дескрипторы загрузочного интерфейса

Подкласс HID 1 определяет два дескриптора для загрузочного устройства. Устройства могут добавить дополнительные данные к этим **отчетам загруки**, но первые 8 байт клавиатуры и первые 3 байта мыши **в отчете** должны соответствовать формату**,** определяемому **Boot Report** дескриптором в таком порядке, чтобы данные были правильно интерпретированы BIOS. Отчет не может превышать 8 байт. BIOS будет игнорировать любые расширения для отчетов. Эти дескрипторы описывают отчеты, которые BIOS ожидает увидеть. Однако так как BIOS на самом деле не читал отчетный дескриптор, эти дескрипторы не должны быть жестко запрограммированы в устройстве, если альтернативный отчетный дескриптор предоставляется. Вместо этого, дескрипторы, которые описывают устройство в USB-совместимой операционной системе должны быть включены (они могут или не могут быть одинаковыми). Когда драйвер HID класса загружен, он будет выдавать Change Protocol, меняющийся от загрузочного протокола к отчетному протоколу после прочтения отчетного дескриптора загрузочного интерфейса.

B.1 Протокол 1 (Клавиатура)

Далее представляется отчетный дескриптор для загрузочного интерфейса для клавиатуры

```
Usage Page (Generic Desktop),
Usage (Keyboard),
Collection (Application),
    Report Size (1),
    Report Count (8),
    Usage Page (Key Codes),
    Usage Minimum (224),
    Usage Maximum (231),
    Logical Minimum (0),
    Logical Maximum (1),
     Input (Data, Variable, Absolute),
    Report Count (1),
```

```
Report Size (8),
 Input (Constant),
Report Count (5),
Report Size (1),
Usage Page (LEDs),
Usage Minimum (1),
Usage Maximum (5),
Output (Data, Variable, Absolute),
Report Count (1),
Report Size (3),
 Output (Constant),
Report Count (6),
Report Size (8),
Logical Minimum (0),
Logical Maximum(255),
Usage Page (Key Codes),
Usage Minimum (0),
 Usage Maximum (255),
Input (Data, Array),

End Collection
```

Следующая таблица представляет входной отчет клавиатуры (8 байт)

Байт	**Описание**
0	Клавиши-модификаторы
1	Зарезервировано
2	Код клавиши 1
3	Код клавиши 2
4	Код клавиши 3
5	Код клавиши 4
6	Код клавиши 5
7	Код клавиши 6

Замечание байт 1 этого отчета – константа. Он зарезервирован для OEM использования. BIOS должен игнорировать это поле. Рекомендуется возвращать нули в неиспользуемых полях

Следующая таблица представляет выходной отчет с клавиатуры (1 байт)

Байт	**Описание**
0	NUM LOCK
1	CAPS LOCK
2	SCROLL LOCK
3	COMPOSE
4	KANA
5..7	CONSTANT

Замечание: LED это абсолютные выходные элементы. Это означает, что состояние каждого LED должно входить в выходные отчеты (0 =выкл, 1 = вкл). Относительные элементы будут влиять только на выбранное управление (0 = нет изменения, 1= изменение).

B.2 протокол 2 (Мышь)

Далее иллюстрируется отчетный дескриптор для загрузочного интерфейса мыши.

Usage Page (Generic Desktop),

```
Usage (Mouse),
Collection (Application),
        Usage (Pointer),
        Collection (Physical),
                Report Count (3),
                Report Size (1),
                Usage Page (Buttons),
                Usage Minimum (1),
                Usage Maximum (3),
                Logical Minimum (0),
                Logical Maximum (1),
                Input (Data, Variable, Absolute),
                Report Count (1),
                Report Size (5),
                Input (Constant),
                Report Size (8),
                Report Count (2),
                Usage Page (Generic Desktop),
                Usage (X),
                Usage (Y),
                Logical Minimum (-127),
                Logical Maximum (127),
                Input (Data, Variable, Relative),
        End Collection,
End Collection
```

Байт	**Бит**	**Описание**
0	0	Кнопка 1
	1	Кнопка 2
	2	Кнопка 3
	4 to 7	Спец устройство
1	0 to 7	X расположение
2	0 to 7	Y расположение
3 to n	0 to 7	Спец устройство (дополнительно)

Приложение C: Реализация клавиатуры

Далее требования для USB клавиатур:

- Не клавиши-модификаторы должен быть указаны на вход (Array, Абсолют) элементов. Отчеты должны содержать список ключей, сейчас нажатых клавиш.
- Клавиатура должна поддерживать ожидание запроса.
- Клавиатура должна отправлять данные отчетов на Idle частоте или при получении запроса Get_Report, даже когда нет никаких новых ключевых событий.
- Клавиатура должна сообщать фантомный статус, нумерованный Использованием (ErrorRollOver) во всех полях массива, когда число нажатых клавиш превышает Report Count.. Предел - шесть не-клавиш-модификаторов при использовании клавиатурного дескриптора в Приложении B. Кроме того, клавиатура может сообщить о фантомном состоянии, когда недействительная или неузнаваемымая комбинация клавиш нажата.

- Порядок кодов клавиш в поле массива не имеет значения. Порядок определения на

стороне хоста. Если два или более ключей, отправлены в одном отчете, их порядок не определен.

- "Repeat Rate" и "Delay Before First Repeat" реализуются хостом, а не в клавиатурой (это означает, что BIOS в режиме Legacy). Хост может использовать скорость устройства и число докладов, чтобы определить, как долго нажата клафиша. Кроме того, хост может использовать свои собственные часы для измерения этих возможностей.
- Синхронизация между lED состояниеми и CAPS LOCK, NUM Lock, Scroll Lock, COMPOSE и KANA элементами определяется хостом, а не клавиатурой. При использовании дескриптора из приложения B, LED статус устанавливаются отправкой 5-битный абсолютного отчета клавиатуре через Set_Report (выход) запрос.

- Для Загрузочных Клавиатуры, отчетный индекс для данной кнопки должен быть тот же что и у использования данной кнопки. Это необходимо, поскольку BIOS не будет читать отчетный дескриптор.
- Загрузочные Клавиатуры должены поддерживать протокол загрузки и запрос Set_Protocol.

Событие	**Модификационный байт**	**Array**	**Array**	**Array**	**Коментарий**
None	00000000B	00H	00H	00H	
RALT down	01000000	00	00	00	
None	01000000	00	00	00	Даже если не нажато клавиш, отсылать состояние клавиш .
A down	01000000	04	00	00	
X down	01000000	04	1B	00	
B down	01000000	04	05	1B	Порядок отчета является случайным и не отражает порядок событий
Q down	01000000	01	01	01	Состояние фантом.
A up	01000000	05	14	1B	
B and Q up	01000000	1B	00	00	Несколько событий в отчете, даже если неизвестен порядок.
None	01000000	1B	00	00	
RALT up	00000000	1B	00	00	
X up	00000000	00	00	00	

Замечание: Этот пример использует 4- байтный отчет, поэтому состояние фантом может быть проще показано. Большинство клавиатур должно иметь 8 или более байт в их отчете.

Приложение D: Пример отчетного дескриптора

Следующий пример показывает пример отчетного дескриптора

D.1 Пример дескриптора джойстика

```
Usage Page (Generic Desktop),
Usage (Joystick),
Collection (Application),
    Usage Page (Generic Desktop),
    Usage (Pointer),
     Collection (Physical),
        Logical Minimum (-127),
        Logical Maximum (127),
        Report Size (8),
        Report Count (2),
        Push,
        Usage (X),
        Usage (Y),
        Input (Data, Variable, Absolute),
        Usage (Hat switch),
        Logical Minimum (0),
        Logical Maximum (3),
        Physical Minimum 0),
        Physical Maximum (270),
        Unit (Degrees),
        Report Count (1),
        Report Size (4),
        Input (Data, Variable, Absolute, Null State),
        Logical Minimum (0),
        Logical Maximum (1),
        Report Count (2),
        Report Size (1),
        Usage Page (Buttons),
        Usage Minimum (Button 1),
        Usage Maximum (Button 2),
        Unit (None),
        Input (Data, Variable, Absolute)
     End Collection,
    Usage Minimum (Button 3),
    Usage Minimum (Button 4),
    Input (Data, Variable, Absolute),
    Pop,
    Usage (Throttle),
    Report Count (1),
    Input (Data, Variable, Absolute),
End Collection
```

Байт	**бит**	**Описание**
0	0 to 7	X позиция
1	0 to 7	Y позиция
2	0 to 3	Hat switch
	4	Кнопка 1
	5	Кнопка 2
	6	Кнопка 3
	7	Кнопка 4
3	0 to 7	Газ

Примечание Хотя для hat switch элемента разрешается 3 бита,выделяется 4 бита вотчете для выравнивания в байте.

Приложение E: Пример USB дескриптора для устройства HID класса

Это приложение содержит примерный набор дескрипторов некоего продукта

Внимание! Этот пример предназначен для использования в качестве учебного инструмента. Не передавайте эту информацию дословно, даже в случае аналогичного устройства. Важно понять, почему было выбрано то или иное значение дескриптора и каждое поле. Образец – 105ти клавишная клавиатура с указывающим устройством. Это устройство может быть сделано только на одном интерфейсе. Однако устройство может поодерживать протокол загрузки. Поэтому сдесь указаны по два дескриптора **Interface**, **Endpoint**, **HID** and **Report** для устройства.

E.1 Device Descriptor (дескриптор устройства)

Часть	**Сдвиг/размер**	**Описание**	**Пример знач**
bLength	0/1	Числовое выражение характеризующее размер дескрипт	0x12
bDescriptorType	1/1	Тип дескриптора	0x01
bcdUSB	2/2	USB HID Specification Release 1.0.	0x100
bDeviceClass	4/1	Код класса (определяется USB)	0x00
bDeviceSubClass	5/1	Код подкласса (определяется USB).	0x00
bDeviceProtocol	6/1	Код протокола.	0x00
bMaxPacketSize0	7/1	Максимальный размер пакета (8, 16, 32, or 64 доступны).	0x08
idVendor	8/2	Vendor ID (определяется USB).	0xFFFF
idProduct	10/2	Product ID (определяется производителем).	0x0001
bcdDevice	12/2	Номер релиза (определяется производителем).	0x0100
iManufacturer	14/1	Индекс строчного дескриптора для производителя.	0x04
iProduct	15/1	Индекс строчного дескриптора для продукта.	0x0E
iSerialNumber	16/1	Индекс строчного дескриптора для серийного номера утсройства.	0x30
bNumConfigurations	17/1	Количество возможных конфигураций.	0x01

E.2 Configuration Descriptor (конфигурационный дескриптор)

Часть	**Сдвиг/размер**	**Описание**	**Пример знач**
bLength	0/1	Размер дескриптора в байтах	0x09
bDescriptorType	1/1	Конфигурация (определяется USB).	0x02
wTotalLength	2/2	Полный размер данных, возвращенных для этой конфигурации	0x003B
bNumInterfaces	4/1	Число интерфейсов для данной конфигурации	0x02
bConfigurationValue	5/1	Значение для использования, как аргумента в Set Configuration для выбора данной конфигурации	0x01
iConfiguration	6/1	Индекс строчного дескриптора для описания данной конфигурации	0x00
bmAttributes	7/1	Характеристика конфигурации 7 питание от шины 6 питание от самого устройства 5 удаленное пробуждение 4..0 Reserved (reset to 0)	10100000B
MaxPower	8/1	Максимальное напяжение питания USB устройства от шины	0x32

E.3 Interface Descriptor (Клавиатура)

Часть	Сдвиг/размер	Описание	Пример знач
bLength	0/1	Размер дескриптора в байтах	0x09
bDescriptorType	1/1	Интерфейсный дескриптор .	0x04
bInterfaceNumber	2/1	Номер интерфейса.	0x00
bAlternateSetting	3/1	Используется для назначения альтернативных настроек	0x00
bNumEndpoints	4/1	Число endpoint, используемых этим интерфесом	0x01
bInterfaceClass	5/1	Класс код	0x03
bInterfaceSubClass	6/1	Код подкласса 0 Нет подкласса 1 Загрузочный интерфес	0x01
bInterfaceProtocol	7/1	Код протокола. 0 Нет 1 Клавиатура 2 Мышь	0x01
iInterface	8/1	Индекс строчного дескриптора	0x00

E.4 HID Дескриптор (Клавиатура)

Часть	Сдвиг/размер	Описание	Пример знач
bLength	0/1	Размер дескриптора в байтах	0x09
bDescriptorType	1/1	Тип HID дескриптора (assigned by USB).	0x21
bcdHID	2/2	HID релиз код	0x101
bCountryCode	4/1	Страна для локализации	0x00
bNumDescriptors	5/1	Число HID дескрипторов	0x01
bDescriptorType	6/1	Тип отчетного дескриптора	0x22
wDescriptorLength	7/2	Полный размер отчетного дескриптора	0x3F

E.5 Endpoint Дескриптор (Клавиатура)

Часть	Сдвиг/размер	Описание	Пример знач
bLength	0/1	Размер в байтах	. 0x07
bDescriptorType	1/1	Endpoint дескриптор	. 0x05
bEndpointAddress	2/1	Адрес endpoint на USB устройстве Расшифорвывается следующим образом Bit 0..3 число конечных точек Bit 4..6 резервировано, сброшено в 0 Bit 7 Направление, игнорируемое Control endpoints: 0 - OUT endpoint 1 - IN endpoint	10000001B
bmAttributes	3/1	Это поле описывает endpoint's атрибуты Bit 0..1 Тип трансфера: 00 управление 01 Изохронный 10 масса 11Прерывание	00000011B.
wMaxPacketSize	4/2	Максимальный размер пакета, который эта конечная точка может переслать или принять	0x0008
.			
bInterval	6/1	Интервал обмена endpoint для трансферов	0x0A

E.6 Отчетный Дескиптор (Клавиатура)

Данные		Значение (Hex)
Usage Page (Generic Desktop),		05 01
Usage (Keyboard),		09 06
Collection (Application),		A1 01
Usage Page (Key Codes);		05 07
Usage Minimum (224),		19 E0
Usage Maximum (231),		29 E7
Logical Minimum (0),		15 00
Logical Maximum (1),		25 01
Report Size (1),		75 01
Report Count (8),		95 08
Input (Data, Variable, Absolute),	;Modifier byte	81 02
Report Count (1),		95 01
Report Size (8),		75 08
Input (Constant),	;Reserved byte	81 01
Report Count (5),		95 05
Report Size (1),		75 01
Usage Page (Page# for LEDs),		05 08
Usage Minimum (1),		19 01
Usage Maximum (5),		29 05
Output (Data, Variable, Absolute),	;LED report	91 02
Report Count (1),		95 01
Report Size (3),		75 03
Output (Constant),	;LED report padding	91 01
Report Count (6),		95 06
Report Size (8),		75 08
Logical Minimum (0),		15 00
Logical Maximum(101),		25 65
Usage Page (Key Codes),		05 07
Usage Minimum (0),		19 00
Usage Maximum (101),		29 65
Input (Data, Array),	;Key arrays (6 bytes)	81 00
End Collection		C0

Далее рассматриваются дескрипторы для мыши

E.7 Interface дескриптор (Мышь)

E.8 HID Дескриптор (Мышь)

E.9 Endpoint дескриптор (Мышь)

E.7 Interface дескриптор (Мышь)

Часть	Смещение/размер	Описание	Пример значения
bLength	0/1	Размер дескриптора в байтах.	0x09
bDescriptorType	1/1	Тип интерфейсного дескриптора	0x04
bInterfaceNumber	2/1	Число интерфесов.	0x01
bAlternateSetting	3/1	Значение для выбора альтернативных настроек.	0x00
bNumEndpoints	4/1	Число конечных точек.	0x01
bInterfaceClass	5/1	Класс кодд	0x03
bInterfaceSubClass	6/1	1 = Загрузочный интерфейс.	0x01
bInterfaceProtocol	7/1	2 = Мышь.	0x02
iInterface	8/1	Индекс строчного дескриптора.	0x00

E.8 HID Дескриптор (мышь)

Часть	Смещение/размер	Описание	Пример значения
bLength	0/1	Размер дескриптора в байтах	0x09
bDescriptorType	1/1	Тип HID дескриптора.	0x21
bcdHID	2/2	Номер релиза.	0x101
bCountryCode	4/1	Страна для локализации.	0x00
bNumDescriptors	5/1	Число HID дескрипторов	0x01
bDescriptorType	6/1	Тип Отчетного дескриптора	0x22
wItemLength	7/2	Общая длина отчетного дескриптора	0x32

E.9 Endpoint дескриптор (Мышь)

Часть	Смещение/размер	Описание	Пример значения
bLength	0/1	Размер дескриптора в байтах	0x07
bDescriptorType	1/1	Тип Endpoint дескриптора	0x05
bEndpointAddress	2/1	Адрес endpoint.	10000010B
bmAttributes	3/1	Описывает endpoint атрибуты	00000011B
wMaxPacketSize	4/2	Максимальный размер пакета	0x0008
bInterval	6/1	Интервал для опроса endpoint для трансферов данных.	0x0A

E.10 Отчетный дескриптор (Мышь)

Элемен т	Значение (Hex)
Usage Page (Generic Desktop),	05 01
Usage (Mouse),	09 02
Collection (Application),	A1 01
Usage (Pointer),	09 01
Collection (Physical),	A1 00
Usage Page (Buttons),	05 09
Usage Minimum (01),	19 01

Usage Maximun (03),		29 03
Logical Minimum (0),		15 00
Logical Maximum (1),		25 01
Report Count (3),		95 03
Report Size (1),		75 01
Input (Data, Variable, Absolute),	;3 бита кнопок	81 02
Report Count (1),		95 01
Report Size (5),		75 05
Input (Constant),	;5 бит дополнения	81 01
Usage Page (Generic Desktop),		05 01
Usage (X),		09 30
Usage (Y),		09 31
Logical Minimum (-127),		15 81
Logical Maximum (127),		25 7F
Report Size (8),		75 08
Report Count (2),		95 02
Input (Data, Variable, Relative),	;позиционные биты	81 06
End Collection,		C0
End Collection		C0

E.11 Строчный дескриптор

Часть	**Смещение/размер**	**Описание**	**Пример значения**
bLength	00/01	Длинна с трочного дескриптора в байтах	0x04
bDescriptorType	01/01	Descriptor Type = String	0x03
bString	02/02	Массив LangID кодов (в данном случае 2-байтный код для английского)	0x0009
bLength	04/01	Длина строчного дескриптора	0x0A
bDescriptorType	05/01	Descriptor Type = String	0x03
bString	06/08	Создатель	ACME
bLength	14/01	Длина строчного дескриптора.	0x22
bDescriptorType	15/01	Descriptor Type = String	0x03
bString	16/32	Локализаор клавиатуры	Locator Keyboard
bLength	48/01	Длина строчного дескриптора	0x0E
bDescriptorType	49/01	Descriptor Type = String	0x03
bString	50/12	Серийный номер устройства	ABC123

Замечание В этом примере смещение используется, т.к. смещение всегда меньше 256. Иначе каждая строка могла бы дать последовательный строчный индекс. Оба применения функционально эквиваленты.

Приложение F: Legacy реализация клавиатуры

Протоколы загрузки и наследования для клавиатур USB позволяют системе, которая не поддерживает USB, поддерживать USB HID клавиатуры без полной поддержки всех необходимых элементов USB. Загрузка / Наследования протокола не ограничивает клавиатуры для такого использования. Вместо этого, ожидается, что клавиатура будет поддерживать

полностью HID-совместимые, основанные на элементах протоколы, а также загрузки и унаследованные протоколы.

F.1 Цель

Эта спецификация содержит информацию для руководства дизайнеров клавиатур в создании USB Boot / Legacy клавиатуры. В нем содержится информация для разработчиков системного ROM, чтобы они могли использовать такую клавиатуру без полного разбора HID дескриптора. Мотивацией является то, что в то время как полный класс HID чрезвычайно богатый и сложный, это не представляется возможным осуществить требуемого класс HID драйвера в ROM.

F.2 Обзор управления

Спецификация HID класса предусматривает реализацию собственного описания устройств ввода. HID дескрипторы устройств, в том числе доклад дескриптор, содержат достаточно информации для операционной системы, чтобы понять докладе протокол устройство использует для отправки события, как нажатие клавиш.

Большинство USB-устройств будет работать при поддержке некоторых USB-совместимая операционная система.Операционная система может позволить себе такой уровень сложности. В большинстве систем ПЗУ загрузки системы не может.

Тем не менее, ПЗУ загрузки системы обычно требуется некоторое поддержка клавиатуры, чтобы для настройки системы, отладка и другие функции. Примеры включают BIOS в PC-AT-систем, и IEEE 1275 загрузки прошивки в рабочих станциях. PCaЬ системах под управлением DOS имеют дополнительную проблему в том, что BIOS должен предоставить полную поддержку клавиатуры для DOS, старые приложения, необходимые для настройки системы.

Поэтому необходимо, чтобы система принять ввод с клавиатуры перед загрузкой операционной системы. Вскоре следует, что поддержка мыши также могут быть необходимы. Для облегчения этой задачи на диске разработчика, спецификации HID определяет протокол клавиатуры загрузку и протокол работы с мышью загрузки. Так как эти протоколы являются предопределенными, система может занять 8-байтовые пакеты и декодировать их непосредственно.Загрузка системы не нужно проанализировать доклад дескрипторов понять пакета.

F.3 Требования загрузочной клавиатуры

- Для того, чтобы клавиатура USB Boot, клавиатура должна отвечать следующим требованиям:
- Загрузка Клавиатура представляет доклад ключи в формате, описанном в Приложении B спецификации класса HID.
- Загрузка клавиатура будет поддерживать Set_Idle запросу.
- Загрузка Клавиатура направляет данные отчеты, когда прерывание трубы опрошенных, даже когда Есть никаких новых ключевых событий.Set_Idle просьба изменить это поведение, как описано в спецификации класса HID.
- Загрузка Клавиатура сообщают "Keyboard ErrorRollOver" во всех массив полей, когда количество не клавиши-модификаторы нажатии превышает доклада графа.Предел шесть не-клавиши-модификаторы для загрузки клавиатуры.
- Загрузка Клавиатура сообщают "Keyboard ErrorRollOver" во всех массив полей, когда комбинация клавиш нажата не могут быть точно определены устройства, такие как ключевые призрак или опрокидывания ошибок.

- Загрузки клавиатура не будет содержать CAPS LOCK, NUM Lock, Scroll Lock, сочинять, или КАНА светодиодной государств без явного Set_Report (выход) запросов из системы.
- Загрузка клавиатура будет поддерживать любое использование кодов стандартных 84-ключ
- клавиатуры. (См.: Приложение A.3)
- Загрузка Клавиатура должна поддержать просьбу Set_Protocol.
- Загрузка Клавиатура должна, по сброса, вернуться к не-протокол загрузки, который описан в своем докладе дескриптор. То есть, доклад дескриптор для загрузки клавиатура не обязательно совпадает протокол загрузки.Доклад дескриптор для загрузки клавиатура не-протокол загрузки дескриптор.
- При получении запроса с Get_Descriptor wValue установлен в конфигурации клавиатуры вернемся конфигурации дескриптора, все дескрипторы интерфейсов, все Endpoint дескрипторов и дескриптор HID. Это уже не возвратится дескриптор HID отчета.Дескриптора HID должны чередоваться с интерфейсом и Endpoint дескрипторов, то есть порядка должны быть:

Configuration descriptor (other Interface, Endpoint, and Vendor
Specific descriptors if required)
    Interface descriptor (with Subclass and Protocol specifying BootKeyboard)
    HID descriptor (associated with this Interface)
        Endpoint descriptor (HID Interrupt In Endpoint)
        (other Interface, Endpoint, and Vendor Specificdescriptors if required)

F.4 Клавиатура: Требования для неподдерживающих USB клавиатур

Ниже приведены требования к BIOS, IEEE 1275 загрузки прошивки, или других не-USB знать систему для использования протокола USB загрузки клавиатуры:

- Система должна делать никаких предположений о порядке нажатия клавиш от порядка ключей в одном отчете.Порядок коды клавиш в массиве полей не имеет значения. Заказ определение делается программного обеспечения хоста сравнения содержимого предыдущего доклада текущего отчета. Если два или более ключей, как сообщается в одном отчете, их порядок не определен. Клавиатур, которые могут буфера событий, которые бы в противном случае привело к многочисленным событиям в одном отчете.

- Система осуществляет Typematic скорость повтора и задержки.Загрузка Клавиатура не имеет возможности реализовать Typematic скорость повтора и задержки.Система может использовать скорость устройства отчет и ряд докладов, чтобы определить, как долго ключевых проходит вниз. Кроме того, система может использовать свои собственные часы или Set_Idle просьбой о сроках этих возможностей.

- Система должна поддерживать синхронизацию между государствами светодиодной событий Caps Lock, Num Lock или Scroll Lock.Светодиодная система устанавливает государств, отправив 5bit абсолютной доклад клавиатуры через Set_Report (с указанием выходных доклада) запросу.

- Система выдает запрос Set_Protocol для клавиатуры после настройки клавиатуры устройства.

- Система не обращают внимания на значение второго байта в 8-байтовый пакет данных клавиатуры. Этот байт доступна для конкретной системы расширений, однако нет никакой гарантии, что любое использование второй байт будет переносим на неспецифические системы. Поэтому, вероятно, будет ограниченным в использовании, как особенность ноутбука клавиатура, где вместо клавиатуры специфична для системы и не могут быть перемещены в общих платформ.

F.5 Клавиатура: Использование Boot протокола клавиатуры

В данном разделе описаны некоторые подробности за требований, перечисленных в Приложении G.4.

Чтобы использовать протокол загрузки, система должна выполнить следующие действия:

- Выбор конфигурации, которая включает bInterfaceSubClass 1, "Boot
- Интерфейс Подкласс ", и bInterfaceProtocol 1," Keyboard ".
- У Set_Protocol обеспечить устройство находится в режиме загрузки. По умолчанию, устройство поставляется в не-режиме загрузки (должен прочитать доклад дескриптор знать протокол), так что этот шаг позволяет системе положить устройство в режим предопределенного протокол загрузки.
- По получении 8-байтовый отчет о прерываний В конечной точке, система должна смотреть на клавиш-модификаторов битов (байт 0, биты 7-0), чтобы определить, если любой из SHIFT, CTRL, ALT, или GUI ключи изменилось состояние с момента последнего отчета.Система также должны взглянуть на шесть байт код клавиши, чтобы увидеть, если любой из не-клавиши-модификаторы изменилось состояние с момента последнего отчета.
- Если не-клавиша-модификатор изменилось государство, система должна переводить код клавиши отправлен в Докладе системы признана ключевым событием.

- Это преобразование может быть достигнуто через справочную таблицу.Код клавиши на самом деле индекс, но и для разработчиков систем различие не имеет значения.Значение отправлены в докладе ключ загрузки идентична стоимости в использовании индекса. Например, если отчет содержит следующие то, глядя вверх Использование индекса в таблице Основные использования, 04h является ключом, 3Ah является клавишу F1, и 5Dh является цифровой клавиатуре клавишу 5.

Байт	**Значение**
Byte 0	00000000b
Byte 1	00000000b
Byte 2	04h
Byte 3	3Ah
Byte 4	5Dh
Byte 5	00h
Byte 6	00h
Byte 7	00h

Важно Необходимо подчеркнуть, что это тщательно организовал исключение из правила, что использований не отправляются в докладе HID. В случае загрузки клавиатура, код клавиши таблица была написана специально, чтобы Использование равно Логические индекс которой не сообщается.

Примечание: клавиатура пример ниже должна быть исправлена до 1,0 документ может быть доработан.

Например, предположим, определенных 17-клавиша клавиатуры не использует протокол загрузки. Поэтому он не может объявить себя загрузки клавиатуры. Это может поставить следующие дескриптор доклад, пример не-загрузки 17-клавишная цифровая клавиатура:

```
Usage Page (Generic Desktop),
Usage (Keyboard),
Report Count (0),
Collection (Application),
   Usage Page(Key Codes),
   Usage(0),       ; key null
    Usage Minimum(53h),
    Usage Maximum(63h),
   Logical Minimum (0),
   Logical Maximum (17),
```

```
  Report Size (8),
  Report Count (3)
  Input (Data, Array),
End Collection
```

Использований из той же ключевой страницы Использование кодекса, а потому, что значения логического Минимум, Логические Максимум, Минимум использования и использования Максимальная разные, байт в докладе не совпадать с использований в ключевой страницы Использование Кодекса. Чтобы показать, что клавиатура, '5 'это вниз в этом примере, в докладе от этого устройства будет выглядеть следующим образом.

Байт	Значение
0	0Bh
1	00h
2	00h

0Bh является индекс список использований объявленного вышеде скриптор.Список объявленных использования начинается с 53H, который является использование «Блокировка лавиатуры Num и Clear".Одиннадцатыйэлемент в этом списке "клавиатуры 5", поэтому отчет включает в себя запись c0Bh.

Это два шага де ссылок необходимо для не-загрузочное устройство. В общем случае использования требуется не может начинаться с 1, не может бытьнепрерывной список, и может использовать две или более страницспользования.

Тем не менее, протокол загрузки был разработан как для обеспечения совместимости с части доклада дескриптора HID, а также ликвидироватьдвухступенчатую-де-ссылок для этого особого случая.Операционная система должна прочитать дескриптор HID отчет за устройство протокола.ROM-системаможет использовать протокол загрузки после выдачи запроса Set_Protocol..

Приложение G: HID запрос Support Requirements

Следующая таблица перечисляет запросы, необходимые для поддержки разными типами устройств HID класса.

Тип	Get Report	Set Report	Get Idle	Set Idle	Get Protocol	Set Protocol
Boot Mouse	Обязат	Дополнит	Дополнит	Дополнит	Обязат	Обязат
Non-Boot Mouse	Обязат	Дополнит	Дополнит	Дополнит	Дополнит	Дополнит
Boot Keyboard	Обязат	Дополнит	Обязат	Обязат	Обязат	Обязат
Non-Boot Keyboard	Обязат	Дополнит	Обязат	Обязат	Дополнит	Дополнит
Other Device	Обязат	Дополнит	Дополнит	Дополнит	Дополнит	Дополнит

Приложение H: Глоссарий

Это приложение раскрывает термины, используемые в документе. Для дополнительных терминов, относящихся к USB смотри Chapter 2 “Terms and Abbreviations,” в USB спецификации.

Array (Массив) Набор полей данных, которая содержит индексы, отвечающие к управлению. Наборы кнопок или ключей содержаться в Array элементах.

Boot Device (загрузочное устройство) Устройство, используемое хостом для оказания помощи в системе конфигурации до загрузки операционной системы. Не-загрузочному устройству нет необходимости работать до загрузки.

Button bitmap (биткарта кнопки) Серия 1-битных полей, каждому из которых отвечает вкл/выкл состояние кнопки. Кнопки могут быть представлены в виде массива или биткарты.
Class (класс) USB устройства организованы в классификациях HID, Аудио и др., основанных на особенностях устройства, поддержки запросов, протоколов данных.
Collection (Набор) Набор это группа **Input**, **Output**,и **Feature** элеметов. Например мышь, клавиатура, джойстик, указатель. Набор указателя содержит элементы для x, y позиции и кнопку. **Collection и EndCollection** элементы используются для разграничения наборов.
Control (управление) Приемник или источник данных поля,например, LED это приемник или цель данных поля, а кнопка – пример источника данных.
Control pipe (канал Управления) Канал по умолчанию, созданный для двустороннего сообщения данных.
Data phase (фаза данных) Часть устройства, отвечающая за запорс.
Descriptor (дескриптор) Информация о USB устройстве хранится в сегментах ROM. Эти сегменты называются дескрипторами.
Device class (класс устройства) Метод организации общих функций и протоколов для устройств, служащих одинаковым функциям, например, аудио, отображение и др.
Device descriptor (дескриптор устройства) Пакет с данными, который раскрывает утсройство – производитель, ID продукта, версия и др.
Endpoint descriptor (дескриптор конечной точки) Стандартный дескриптор USB, описывающий тип и возможности USB канала.
Feature control (Feature Управления) Влияет на поведение устройства или посылает состояние устройства. В отличии от входных и выходных данных, особенные данные предназначены для использования устройством, а не приложением. Например, значение скорости повтора конкретной клавиши будет особенностью управления.
Feature item (Feature элемент) Добавляет поля данных в Feature отчет.
Field (поле) Дискретный вид данных внутри отчета.
Frame (кадр) Самый маленький промежуток времени на USB = 1 мс.
HID (Human Interface Device) (устройство для взаимодействия с человеком) Сокращение, описывающее либо специальный класс устройств тип устройства.
HID class (HID класс) Классификация USB устройства, связанного с **HID**.
HID class device (устройство HID класса) Устройство типа: взаимодействующее с человеком и классифицирующееся соответственно.
HID descriptor (HID дескриптор) Информация о USB утсройстве, хранящаяся в ROM.
Host (Хост) Компьютер с USB портом, связанный с устройством через этот порт.
Hub (Хаб) Устройство USB, содержащее несколько портов USB.
Idle rate (процент бездействия системы) Частота, с которой устройство посылает данные, когда нет новых событий в системе.
Input item (Input элемент) Добавляет одно или более плей данных для входящих отчетов.
Interface descriptor (Интерфейсный дескриптор) Поле Класса этого дескриптора определяет, что утсройство - HID устройство.
Interrupt In pipe (Канал Interrupt In) Канал, используемый для передачи незапрашиваемых данных с устройства на хост.
Interrupt Out pipe (Канал Interrupt Out) Канал для передачи данных с небольшой задержкой с хоста на устройство.
Item (Элемент) Компонент отчетного дескриптора, представляющий фрагмент информации о устройстве.Первая часть элемента называется тег, идентифицирует тип информации элемента, ипользуется также для генерации Report элемента, содержащего три категории элементов **Main**, **Global** и **Local.** Каждый тип элемента определяется тегом.
Item parser (элементный анализатор) Часть драйвера **HID** класса, читающая и понимающая информацию в отчетном дескрипторе.
Logical units (Логическая единица) Значение, возвращаемое устройсвом для Логического Максимума и Логического Минимума.
LSB Младший значащий бит
Main item (Главный элемент) Элемент, добавляющий поле в отчет. Например **Input**, **Output** и **Feature** элементы.
Message pipe (Канал Сообщения) Другое название канала Управления.

NAK Значение, которое возвращается, когда на устройство был послан запрос, но устройство не успело подготовиться для ответа.
Nibble Пол байта – 4 бита.
Non-USB aware Операционная система, не знающая интерфейса USB.
Null Отсутсвие значения или ноль, в зависимости от контекста.
Output item (Output элемент) Добавляет одно или более полей данных в выходящий отчет.
Packets (Пакеты) Единица информации в USB. Несколько пакетов составляют танзакцию, несколько транзакций – транспортная передача.
Part (Часть) – Обозначает атрибуты битов
Physical Descriptor(физический дескриптор) Определяет, какая часть тела исползуется для управления или набора Каждый физический дескриптор состоит из следующих трех частей: **Designator(**указатель**)**, **Qualifier (**квалификатор**) и Effort** (Усилие)
Physical units (физические единицы) Логические значения с параметором единицы (см. логические единицы).
Pipes (каналы) Каналы это различные способы передачи данных между драйвером и утсройством. Существуют несколько типов каналов, зависящик от типа кодировки и запросов, которые вы собираетесь делать. Все устройства имеют канал **Управления** по умолчанию. **Interrupt In** канал используется для потокового типа посылки данных с устрйоства. Канал **Interrupt Out** – дополнительный, используется для посылки данных на устройство с небольшой задержкой.
Protocol (протокол) Структура отчета, отличная от структуры, определенной в отчетном дескрипторе.
Report (отчет) Структура данных, возвращаемая устройством хосту (или наоборот). Некоторые утсройства могут иметь несколько отчетных структур, отвечающих только нескольким элементам.Например, клавиатура с встроенным указывающим устройством может посылать данные о кнопке назависимо от указывающего утсройства.
Report descriptor (отчетный дескриптор) Определяет поля данных, передающихся между драйвером и утсройством.
Set (множество) –группа дескрипторов.
Stream pipe (потоковый канал) Изохронный канал для передачи данных.
String descriptor (строчный дескриптор) Таблица текста, используемая одним или большим числом дескрипторов.
Tag (тег) часть отчетного дескртптора, поддерживающая информацию о элементе в соответсвии с использованием.
Terminating items (конечные элемены) Элементы внутри дескриптора.Например **Push**, **Pop**, и **Item.** Когда элементный анализатор внутри драйвера HID класса помещает конечный элемент, содержимое таблицы состояний элементов движется.
Transaction (транзакция) Устройство может посылать и принимать транзакции каждый USB кадр. Она может состоять из несокльких пакетов, но ограничена в размере 8 байт для низкоскоростных устройств и 64 байта для высокоскоростных.
Transfer (трансфер) – одна или несколько транзакций, создающих набор данных, имеющих смысл для устрйоства.
Unknown Usage (Неизветсное использование) Неизвестные использования могут быть стандартными HID использованиями, нераспознанные общим приложением.
Usage (использование) Что конкретно измеряет устройство, в соответствии с указанием производителя.
USB Boot Device (Загрузочное USB устройство) Устройство USB **HID** “Boot/Legacy” совместимое и способное использовать протокол загрузки или формат отчета определено в спецификации HID как Input устройство.
Variable (переменная) Поле данных, содержащее значение для специального управления. Любое управление, посылающее более чем вкл/выкл информацию, должны иметь переменные.
Vendor (производитель) создатель устройства.