Universal Serial Bus Communications Class Subclass Specification for Asynchronous Transfer Mode Devices

Revision 1.2

February 9, 2007

Revision History

Rev	Date	Filename	Comments
1.2	2/9/07		Final edits as per February 2007 CDC meeting

Please send comments or questions to : cdc@usb.org

Contributors, v1.1

Paul E. Berg	Moore Computer Consultants, Inc.
Chuck Brabenac	Intel Corporation
Shelagh Callahan	Intel Corporation
Paul Chehowski	Mitel Corporation
Joe Decuir	Microsoft Corporation
Stefan Eder	Siemens Semiconductors
Ed Endejan	3Com Corporation
Randy Fehr	Northern Telecom
Diego Friedel	AVM
John Howard	Intel Corporation
Ken Lauffenburger	Efficient Networks, Inc.
Ron Lewis	Rockwell Semiconductors
Dan Moore	Diamond Multimedia Systems
Terry Moore	Moore Computer Consultants, Inc.
Andy Nicholson	Microsoft Corporation
Nathan Peacock	Northern Telecom
Dave Perry	Mitel Corporation
Kenny Richards	Microsoft Corporation
Charlie Tai	Intel Corporation
Mats Webjörn	Universal Access

Contributors, V1.2

Bruce Balden	Belcarra
Jun Guo	Broadcom
CC Hung	Mentor Graphics
Dale Self	Symbian
Janne Rand	Nokia
Joe Decuir	MCCI
Joel Silverman	K-Micro
John Turner	Symbian
Ken Taylor	Motorola
Morten Christiansen	Ericsson AB
Peter FitzRandolph	MCCI
Richard Petrie	Nokia
Saleem Mohammed	Synopsys
Tero Soukko	Nokia

Table of Contents

List of Tables

List of Figures

1 Introduction

1.1 Purpose

The USB Communications Device Class Specification 1.1 contains general Communications Class specifications, and details for seven device subclasses. That specification has been editorially reorganized into a common USB CDC 1.2 specification [USBCDC1.2] and a set of subclass specifications. This should help implementers understand what is necessary for each device subclass and facilitate specification maintenance by the USB Device Working Group.

This document is one of those subclass specifications. It contains material technically identical to that contained in USB CDC 1.1. It is intended to guide implementers of USB-connected devices of the types listed in the following section.

1.2 Scope

This document specifies one device subclass intended for use with Communication devices, based on the Universal Serial Bus Class Definitions for Communication Devices specification [USBCDC1.2]. It supports ATM (Asynchronous Transfer Mode) network terminals, including USB-connected ADSL modems.

The intention of this specification is that all material presented here is technically compatible with the previous version of the USB CDC 1.1 Specification, from which it is derived. Numeric codes are defined for subclass codes, protocol codes, management elements, and notification elements.

In some cases material from [USBCDC1.2] is repeated for clarity. In such cases, [USBCDC1.2] shall be treated as the controlling document.

In this specification, the word 'shall' or 'must' is used for mandatory requirements, the word 'should' is used to express recommendations and the word 'may' is used for options.

1.3 Related Documents

Reference	Description
[USB2.0]	Universal Serial Bus Specification, revision 2.0 (also referred to as the USB Specification). **http://www.usb.org**
[USBCCS1.0]	Universal Serial Bus Common Class Specification, revision 1.0. **http://www.usb.org**
[USBCDC1.2]	Universal Serial Bus Class Definitions for Communication Devices, Version 1.2. **http://www.usb.org**.
[USBECM1.2]	Universal Serial Bus Subclass Specification for Ethernet Control Model Devices, Version 1.1 **http://www.usb.org**
[USBWMC1.1]	Universal Serial Bus Subclass Specification for Wireless Mobile Communications, Version 1.1.**http://www.usb.org**
TR-002	ADSL Forum TR-002: ATM over ADSL Recommendations, 1997, http://www.dslforum.org
ANSI T1.413	Network and Customer Installation Interfaces - Asymmetric Digital Subscriber Line (ADSL) Metallic interface, 1995.
ITU-T I.361	B-ISDN ATM Layer specification, 1995

ITU-T I.363	B-ISDN ATM Adaptation Layer (AAL) Specification, 1993
ITU-T I.413	B-ISDN User-Network Interface, 1993
ITU-T I.610	B-ISDN Operation and Maintenance Principles and Functions, 1993

1.4 Terms and Abbreviations

TERM	Description
AAL	ATM Adaptation Layer
ATM	Asynchronous Transfer Mode
BYTE	For the purposes of this document, the definition of a byte is 8 bits.
COMMUNICATIONS INTERFACE	Refers to a USB interface that identifies itself as using the Communications Class definition.
DATA INTERFACE	Refers to a USB interface that identifies itself as using the Data Class definition.
DEVICE MANAGEMENT	Refers to requests and responses that control and configure the operational state of the device. Device management requires the use of a Communications Class interface.
DESCRIPTOR	Data structure used to describe a USB device capability or characteristic
DEVICE	A logical or physical entity that performs a function. The actual entity described depends on the context of the reference. At the lowest level, device may refer to a single hardware component, as in a memory device. At a higher level, it may refer to a collection of hardware components that perform a particular function, such as a USB interface device. At an even higher level, device may refer to the function performed by an entity attached to the USB; for example, a data/FAX modem device. Devices may be physical, electrical, addressable, and logical. When used as a non-specific reference, a USB device is either a hub or a function.
FUNCTION	Device capabilities exposed over the USB cable.
ITU	International Telecommunications Union (formerly CCITT).
MANAGEMENT ELEMENT	Refers to a type of USB pipe that manages the communications device and its interfaces. Currently, only the Default Pipe is used for this purpose.
MASTER INTERFACE	A Communications Class interface, which has been designated the master of zero or more interfaces that implement a complete function in a USB Communications device. This interface will accept management requests for the union.
MULTIFUNCTION DEVICE	A device of peripheral that exposes one or more functions or services to an end user. Exposed services can but do have to be exposed as USB functions.
NOTIFICATION ELEMENT	Refers to a type of USB pipe. Although a notification element is not required to be an interrupt pipe, a notification element is typically defined in this way.
NOTIFICATION ELEMENT	Refers to a type of USB pipe. Although a notification element is not required to be an interrupt pipe, a notification element is typically defined in this way.
PDU	Protocol Data Unit - A combination of the SDU and the current protocol layer's header and/or trailer.
SDU	Service Data Unit – User data and control information created at the upper protocol layers that is transferred transparently through a primitive by layer (N+1) to layer (N) and subsequently to (N-1).
UNION	A relationship between a collection of one or more interfaces that can be considered to form a functional unit.
UNIT	Entity that provides the basic building blocks to describe a protocol stack.

2 Management Overview

This subclass specification includes specifications for devices that connect to the ATM network, including USB-connected ADSL modems.

Note that for USB-connected ADSL modems that expose PPP-over-Ethernet emulation, the USB Ethernet control model subclass specification is applicable.

Devices of these subclasses must conform to:

- USB Specification [USB2.0]
- Device Framework
- Communications Device Class 1.2 [USBCDC1.2]

3 Functional Overview

3.1 Function Models

[USB2.0] defines "function" as a "USB device that provides a capability to the host, such as an ISDN connection, a digital microphone, or speakers". Further, in section 5.2.3, it says "Multiple functions may be packaged into what appears to be a single physical device…. A device that has multiple interfaces controlled independently of each other is referred to as a composite device." We therefore adopt the term "function" to describe a set of one or more interfaces which taken together provide a capability to the host. .

3.2 USB Networking Models

A USB Networking device has a type of Communications Class Interface that will be presented to the host for configuring and managing the networking device. Networking devices are typically "Always Connected", spending all of their time with the "link up". The Communications Class Interface is primarily used to configure and manage the networking device, not to place calls.

In contrast to a telecommunications device, a networking device will always have at least one associated Data Class interface to exchange network traffic. In a typical host software stack, the same driver that is responsible for configuring and managing the network device is also the client that is a source/sink of networking traffic. An example of such a host resident networking driver is either an ATM (this subclass specification) or Ethernet driver.

Networking devices are differentiated by their SubClass code, which currently are comprised of the Ethernet Networking Control Model [USBECM1.2] and ATM Networking Control Model.

3.3 Common Data Plane Characteristics

The core Data-In/Data-Out pipe mechanism is the same for all networking device models supported by this specification, independent of the media type (e.g., Cable, xDSL, Ethernet) or media data type (e.g., ATM cells, Ethernet frames).

Typical USB-based Networking devices will support bulk transfers as the default configuration to exchange data between a host and the USB device.

While each data packet of a bulk endpoint is limited to the maximum packet size defined in the associated endpoint descriptor, it should be noted that a host might request multiple bulk USB protocol packets within a single USB frame. For maximum throughput, a Networking device must be prepared to transfer multiple bulk packets within a single USB frame.

Some USB-based Networking device implementations may support isochronous data transfers in addition to (or instead of) bulk transfers. Isochronous transfers guarantee data throughput and bounded latency, consistent with the needs of real-time streams (audio, video). Isochronous data errors are reported to receiver, but no data integrity (i.e., retransmission) is provided by the USB link.

The Data Class Interface Descriptor protocol code for all Networking Control Models is 00h.

USB provides no inherent flow control mechanism for isochronous pipes, and this specification defines no higher level mechanism for doing so. Instead, it is assumed that the host software is responsible for doing traffic shaping as necessary to match any end-to-end negotiation. If the networking device is performing traffic shaping, then either a bulk endpoint should be used, or the flow control methods should be provided using vendor-specific methods.

The Data Class interface of a networking device shall have a minimum of two interface settings. The first setting (the default interface setting) includes no endpoints and therefore no networking traffic is exchanged whenever the default interface setting is selected. One or more additional interface settings are used for normal operation, and therefore each includes a pair of endpoints (one IN, and one OUT) to exchange network traffic. The host shall select an alternate interface setting to initialize the network aspects of the device and to enable the exchange of network traffic.

To recover the network aspects of a device to known states, the host will first select the default interface setting (with no endpoints) and then select the appropriate alternate interface setting. In response to this action the device shall flush device buffers, clear any filters or statistics counters and will cause *NetworkConnection* and *ConectionSpeedChange* notifications to be sent to the host. The effect of a "reset" on the device physical layer is media-dependent and beyond the scope of this specification.

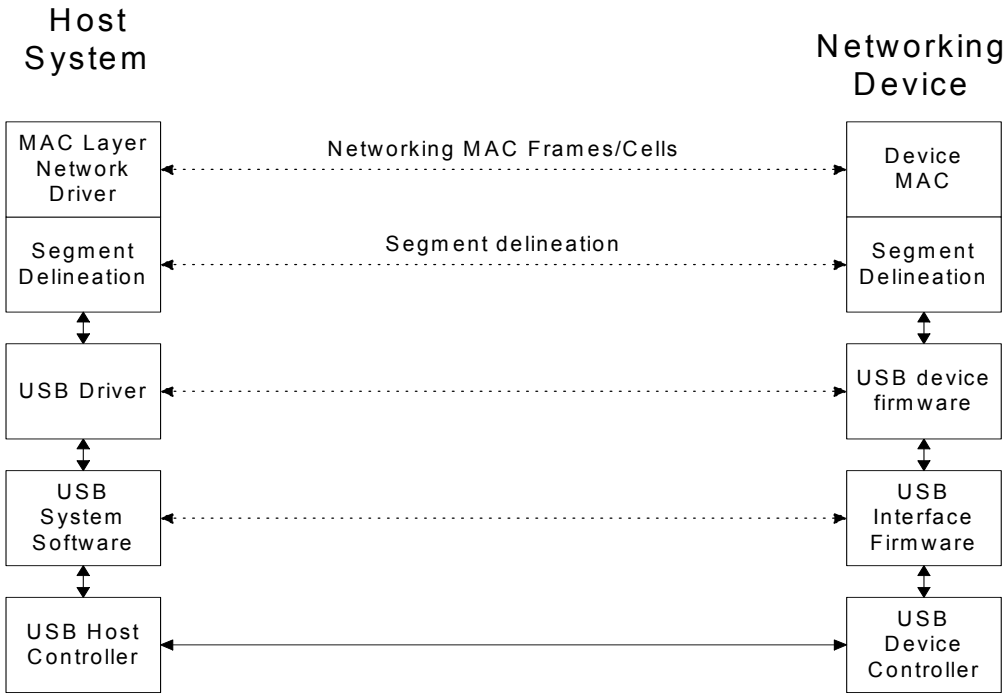

Figure 1 - USB Network Device Example

3.3.1 Segment Delineation

For almost any type of USB attached networking device, a mechanism is needed where both the networking device and the Host can delineate the beginning and ending of a *segment* within the data stream delivered by an endpoint. The meaning and cause of *segment* end is media dependent. Below are some examples:

- The end of an AAL5 ATM SDU
- The end of an AAL5 ATM PDU
- The arrival of a high priority ATM cell

This positive delineation is done using a USB short packet mechanism. When a segment spans *N* USB packets, the first packet through packet *N-1* shall be the maximum packet size defined for the USB endpoint. If the *N*th packet is less than maximum packet size the USB transfer of this short packet will identify the end of the segment. If the *N*th packet is exactly maximum packet size, it shall be followed by a zero-length packet (which is a short packet) to assure the end of segment is properly identified.

When transmitting data to the networking device, it is assumed that the client of the host USB driver takes the appropriate actions to cause a short packet to be sent to the networking device. For segments with lengths that are an even multiple of the pipe's "max packet size", the ability to write a buffer of zero length is required to generate this short packet.

3.3.2 Segment Size

The host and the attached network device must negotiate to establish the maximum segment size. The upper limit for this is usually a function of the buffering capacity of the attached device, but there may be other factors involved as well.

In the case of AAL5 SDU exchanges, the segment size could be up to 64K bytes in length, and is a function of device buffering capacity and the results of end-to-end negotiation with Q.2931.

3.4 ATM Networking Control Model

An ATM USB device is used to move ATM cells or AAL5 SDUs to and from the host. The segmentation and re-assembly (SAR) function in the ATM adaptation layer may be implemented on the host, and not necessarily on the device.

A Communications Class interface is used to configure and manage various ATM functions, where an "ATM Networking Control Model" SubClass code is indicated in the descriptor definition of its Communication Class interface. This Communications Class interface consists of a minimum of two pipes; one is used to implement the management element and the other to implement a notification element. A Data Class interface is used to exchange ATM cells or AAL5 SDUs sent over USB. The host may perform traffic shaping on an aggregated basis according to the upstream speed reported via the Communications Class interface.

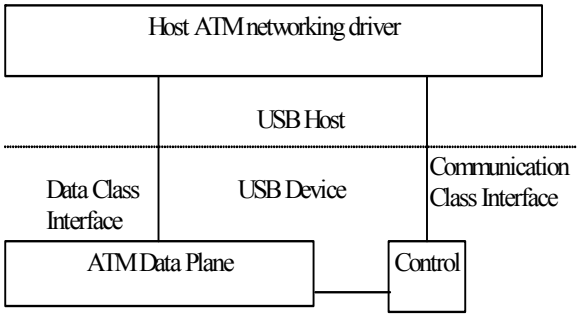

Figure 2: ATM Networking Model

4 Class-Specific Codes

This section lists the codes for the Communications Device Class, Communications Interface Class and Data Interface Class, including subclasses and protocols. These values are used in the *DeviceClass*, *bInterfaceClass*, *bInterfaceSubClass*, and *bInterfaceProtocol* fields of the standard device descriptors as defined in Chapter 9 of [USB2.0].

4.1 Communications Class Code

This is defined in [USBCDC1.2].

4.2 Communications Class Subclass Code

The following table table defines the Communications Subclass code used in this subclass specification:

Table 1 Class Subclass Code

Code	Subclass
07h	ATM

4.3 Communications Class Protocol Code

[USBCDC1.2] defines Communications Class Protocols.

The following table lists the Protocol code used in this subclass specification:

Table 2 Class Protocol Code

Code	Protocol
00h	No class specific protocol required

If a Communications Class interface appears with multiple alternate settings, all alternate settings for that interface shall have the same bInterfaceClass, bInterfaceSubclass and bInterfaceProtocol codes.

5 Descriptors

5.1 Standard USB Descriptor Definitions

Devices that conform to this subclass specification need to implement the standard USB descriptors for the Communications Device Class, Communications Interface Class and Data Interface Class. These are defined in [USBCDC1.2].

5.2 Class-Specific Descriptors

Devices that conform to this subclass specification may need to implement class-specific descriptors for the Communications Interface Class and Data Interface Class. These are defined in section 5.2 of [USBCDC1.2].

5.3 Functional Descriptors

Functional descriptors describe the content of the class-specific information within an Interface descriptor. Functional descriptors all start with a common header descriptor, which allows host software to easily parse the contents of class-specific descriptors. Each class-specific descriptor consists of one or more functional descriptors. Although the Communications Class currently defines class specific descriptor information, the Data Class does not.

[USBCDC1.2] specification describes functional descriptors that may be used in all Communications Subclasses. These include:

- Header Functional Descriptor
- Union Functional Descriptor
- Country Selection Functional Descriptor

The following Functional Descriptors are specific to ATM subclass devices.

5.4 ATM Networking Functional Descriptor

The ATM Networking functional descriptor describes the operational modes supported by the Communications Class interface, as defined in [USBCDC1.2], with the SubClass code of ATM Networking Control. It can only occur within the class-specific portion of an Interface descriptor.

Table 3: ATM Networking Functional Descriptor

Offset	Field	Size	Value	Description
0	*bFunctionLength*	1	Number	Size of this functional descriptor, in bytes.
1	*bDescriptorType*	1	Constant	CS_INTERFACE

Offset	Field	Size	Value	Description
2	*bDescriptorSubtype*	1	Constant	ATM Networking functional descriptor subtype as defined in [USBCDC1.2].
3	*iEndSystemIdentifier*	1	Index	Index of string descriptor. The string descriptor holds the End System Identifier. The first 6 bytes are the unique hardware ID (like a MAC address), and the 7th byte is the end system selector byte. The Unicode representation of End System Identifier is as follows: the first Unicode character represents the high order nibble of the first byte of the Identifier in network byte order. The next character represents the next 4 bits, and so on. The Unicode character is chosen from the set of values 30h through 39h and 41h through 46h (0-9 and A-F). iEndSystemIdentifier can not be zero and the Unicode representation must be 14 characters long. For example, the End System Identifier 0123456789ABCDh is represented as the Unicode string "0123456789ABCD".
4	*bmDataCapabilities*	1	Bitmap	The ATM data types the device supports: D7..D4: RESERVED (Reset to zero) D3: Type 3 -- AAL5 SDU D2: Type 2 -- ATM header template + concatenated ATM cell payloads D1: Type 1 -- Concatenated ATM cells D0: RESERVED (Reset to zero) NOTE: Support for the Type 1 Data Format is mandatory.
5	*bmATMDeviceStatistics*	1	Bitmap	Indicates which optional statistics functions the device collects. If set to 0, the host network driver is expected to keep count. D3 and D4 are only applicable to type 3 devices. If neither D3 nor D4 are set, the type 3 device does not support the SetATMDefaultVC and GetATMVCStatistics requests. D7..D5: RESERVED (Reset to zero) D4: Device counts upstream cells sent on a per VC basis (VC_US_CELLS_ SENT) D3: Device counts downstream cells received on a per VC basis (VC_DS_CELLS_ RECEIVED) D2: Device counts cells with HEC error detected and corrected (DS_CELLS_HEC_ERROR_ CORRECTED) D1: Device counts upstream cells sent (US_CELLS_SENT) D0: Device counts downstream cells received (DS_CELLS_RECEIVED)
6	*wType2MaxSegmentSize*	2	Number	The maximum segment size that the Type 2 device is capable of supporting
8	*wType3MaxSegmentSize*	2	Number	The maximum segment size that the Type 3 device is capable of supporting

Offset	Field	Size	Value	Description
10	*wMaxVC*	2	Number	The maximum number of simultaneous virtual circuits the device is capable of supporting (Type 3 only)

The *wType2MaxSegmentSize* and *wType3MaxSegmentSize* indicate the maximum number of bytes of data in a network segment (i.e., between 2 USB short packets) a device would send to the host via USB for Type 2 and Type 3 devices, respectively. It is expected that a device supporting both Type 2 and Type 3 ATM data formats (as stipulated in the *bmDataCapabilities* field) may employ different buffer management strategies for different ATM data formats, thus may indicate a different maximum segment size for each type. The host driver should allocate buffers accordingly that are large enough to hold the incoming data to prevent any overflow or partial cell/SDU problems. These two fields also serve to inform the host the device buffer capability, and expect the host to transfer network segments no longer than that specified here. Note that this information does not preclude a device or a host to do cut-through forwarding (i.e., start forwarding the portion of the network segment received so far) before receiving a complete network segment.

This maximum segment size does not apply to Type 1 devices, which by definition forward a stream of cells in both directions, and have no concept of network segment size. However, as an implementation note, the host driver should allocate its buffers to contain an integral number of 53-byte cells to prevent partial cells.

6 Communications Class Specific Messages

6.1 Overview

The Communications Interface Class supports the standard requests defined in Chapter 9 of [USB2.0]. In addition, the Communications Interface Class has some class-specific requests and notifications. These are used for device and call management.

Requests for controlling the interface between the USB ATM device are presented in Section 6.2. There are also some additional signals that shall go back to the host as notifications, which are represented in section 6.3. These requests and notifications are transported via the Communications Class interface for the device.

6.2 Management Element Requests

Devices conforming to this subclass specification use the following requests. The only class specific request codes that are valid for a Communications Class interface with a SubClass code of ATM Networking Control Model are listed in the following table. The other class specific requests not listed in that table are inappropriate for an ATM Networking Control Model and must generate a STALL condition if sent to such an interface.

Table 4: Requests — ATM Networking Control Model*

Request	Description	Req'd/Opt	reference
SendEncapsulatedCommand	Issues a command in the format of the supported control protocol. The intent of this mechanism is to support networking devices (*e.g.,* host-based cable modems) that require an additional vendor-defined interface for media specific hardware configuration and management.	Optional	[USBCDC1.2]
GetEncapsulatedResponse	Requests a response in the format of the supported control protocol.	Optional	[USBCDC1.2]
SetATMDataFormat	Chooses which ATM data format will be exchanged between the host and the ATM Networking device.	Required	6.2.1
GetATMDeviceStatistics	Retrieves global statistics from the ATM Networking device.	Required	6.2.2
SetATMDefaultVC	Pre-selects the VPI/VCI value for subsequent GetATMVCStatistics requests	Optional	6.2.3
GetATMVCSTatistics	Retrieves statistics from the ATM Networking device for a particular VPI/VCI.	Optional	6.2.4

* These requests are specific to the Communications Class.

This following describes the request codes that are specific to the Communications Interface Class ATM Subclass.

Table 5: Class-Specific Request Codes for ATM subclass

Request	Value
SET_ATM_DATA_FORMAT	50h
GET_ATM_DEVICE_STATISTICS	51h
SET_ATM_DEFAULT_VC	52h
GET_ATM_VC_STATISTICS	53h
RESERVED (future use)	54h-5Fh

6.2.1 SetATMDataFormat

This request is used to set the data format selected by the host in the *wValue* field.

bmRequestType	bRequestCode	wValue	wIndex	wLength	Data
00100001B	SET_ATM_DATA_ FORMAT	Data Format	Interface	Zero	None

Table 6: ATM Data Format

wValue	Description
1	Type 1 format: concatenated ATM cells
2	Type 2 format: ATM header template + concatenated ATM cell payloads
3	Type 3 format: AAL 5 SDU

The ATM Networking Control Model data format selected specifies how much processing is done in the USB device, versus how much is done in the host. Specifically, the Type 3 data format requires that ATM segmentation and re-assembly (SAR) functions be implemented in the USB device, while Type 1 and Type 2 data formats enable different functional partitioning, and migrate the SAR function to the host. As you move from Type 1 to Type 3, the amount of processing performed in the USB device increases. Support of Type 1 Data Format is mandatory to ensure minimal interoperability. If supported by the device, it is recommended that Type 3 Data Format be chosen by the host whenever possible.

Type 1 Data Format

Type 1 ATM oriented USB devices (e.g., ADSL modems or IEEE 802.14 cable modems) do the least amount of processing on the incoming data and are therefore the simplest types of such devices. The main function of the USB device is to pass the ATM cells from the WAN link to the host, and vice versa. The data format consists of a number of concatenated 53 byte ATM cells. The HEC field in the ATM cell header exists only as a placeholder when transferred over USB, and will be generated and verified by the ATM-oriented device for upstream and downstream traffic, respectively.

For Type 1 devices, all ATM and AAL functions are performed by the host, e.g. AAL layer encapsulation, ATM SAR, and traffic shaping (for upstream direction only). Various AAL types, Operation Administration and Maintenance (OAM) cells, and Resource Management (RM) cells are enabled, if supported by the host.

Type 2 Data Format

The Type 2 ATM-oriented USB device improves USB bus bandwidth efficiency by removing duplicate ATM cell headers prior to transfer over USB. This data format consists of a 4-byte ATM cell header template (excluding the HEC field) followed by a number of 48-byte ATM cell payloads. As with Type 1 devices, the AAL encapsulation, SAR and traffic shaping are all performed by the host.

The Type 2 device needs to assemble cells based on the header template before transmitting them over the WAN link. All the AAL types are supported. OAM cells for flow management are enabled, as are RM cells for the ABR service. A unique requirement for this type of device is that all cell payloads must share the same cell header template. To accomplish this, extra processing rules need to be applied in the host and USB device as follows:

- The 48-octet cell payloads shall be contiguous without additional intervening cell headers, all destined to the same VC, and have the same payload type.

- For AAL5 VCs, the ATM-user-to-ATM-user indication bit in the payload type is used to delimit the AAL5 CPCS PDU. This bit shall be set if the last cell in this segment (i.e., data between 2 consecutive USB short packets) completes an AAL5 CPCS PDU. Note that in this case no other cells can be appended in the same segment after an AAL 5 CPCS PDU has been completed. The device at the other end of the USB bus has to set the ATM-user-to-ATM-user indication bit in the payload type only for the last cell in this segment, but not for any preceding cells, before forwarding them.

- The ATM cell header template transferred between the USB device and host can not include the 8-bit HEC field.

Also, the USB device will have to perform the following functions:

- Generate and insert a HEC field for upstream traffic.

- Verify and remove the HEC field from the downstream traffic.

- Discard cells with HEC errors from the downstream traffic.

For Type 1 and Type 2 ATM oriented USB devices, any number of ATM cells or payloads could be concatenated in an USB buffer as long as it adheres to the rules stipulated above. The general guideline to flush the buffer and send it via USB is when:

- The maximum segment size (*wType2MaxSegmentSize*) is reached (for Type 2 only), or

- Encounter the end of an AAL5 PDU, or

- The timer expires (so you don't hold the cells too long if there are no immediate incoming cells), or

- One or more cells from a low-latency VC are received. NOTE: If a grouping of low latency VC cells are received (e.g., a 20ms audio frame), it is recommended that devices not "flush" (generate a USB short packet) after every ATM cell in this instance to improve link efficiency and reduce host overhead. The methods for accomplishing this are beyond the scope of this document

Type 3 Data Format

Type 3 ATM oriented devices will process the AAL5 SDUs (e.g., Q.2931, ILMI and other SDU's like PPP packets in data VCs) as they arrive from the host, before sending them to the WAN link. This is accomplished by adding AAL5 encapsulation (including the appropriate padding and the CRC generation) to form the AAL5 PDU. The AAL5 PDU formed then goes through the SAR function before being transmitted over the WAN link.

For the ATM cells arriving from the WAN link, the device will reassemble them into AAL5 PDUs, validate the AAL5 CRC (discard the AAL5 PDU if incorrect), strip off the AAL5 encapsulation and transfer AAL5 SDU to the host.

This data format consists of the first 4 bytes of an ATM cell header (excluding the HEC field) followed by a single AAL5 SDU. The VPI/VCI value in the cell header indicates which VC this AAL5 SDU belongs to and the first bit of the PTI field shall be set to 0 for AAL5 SDUs. OAM and RM cells can be supported by sending those cells with the corresponding PTI bits set in the cell header.

6.2.2 GetATMDeviceStatistics

This request is used to retrieve the device statistics based on the feature selector.

BmRequestType	bRequestCode	wValue	wIndex	wLength	Data
10100001B	GET_ATM_DEVICE_ STATISTICS	Feature Selector	Interface	4	32 bit unsigned integer

In Table 7, the terms UPSTREAM(US) and DOWNSTREAM(DS) refer to the direction of the data flow across the connection between the device and the network.

The value returned indicates the number of cells matching the specified statistic that have occurred since the device has been powered on or reset, or since the SET_INTERFACE request has been received (see Section 3.3 for more details). This number is a 32 bit unsigned integer, which is incremented at each occurrence and will be wrapped to zero upon reaching the maximum value.

Table 7: ATM Device Statistics Feature Selector Codes

Feature selector	Code	Targets	Length of Data	Description
RESERVED	00h	None	None	Reserved for future use
US_CELLS_SENT	01h	Interface	4	The number of cells that have been sent upstream to the WAN link by the ATM layer. Support for this statistic by device hardware is optional.
DS_ CELLS_RECEIVED	02h	Interface	4	The number of cells that have been received downstream from the WAN link by the ATM layer. Support for this statistic by device hardware is optional.

Feature selector	Code	Targets	Length of Data	Description
DS_CELLS_USB_CONGESTION	03h	Interface	4	The number of cells that have been received downstream from the WAN link by the ATM layer and discarded due to congestion on the USB link. Support for the feature code by the device is MANDATORY.
DS_CELLS_AAL5_CRC_ERROR	04h	Interface	4	The number of cells that have been received downstream from the WAN link by the ATM layer and discarded due to AAL5 CRC errors. Support for this feature code by Type 3 devices is MANDATORY.
DS_CELLS_HEC_ERROR	05h	Interface	4	The number of cells that have been received downstream from the WAN link and discarded due to HEC errors in the cell header. Support for this statistic by device hardware is MANDATORY.
DS_CELLS_HEC_ERROR_CORRECTED	06h	Interface	4	The number of cells that have been received downstream from the WAN link and have been detected with HEC errors in the cell header and successfully corrected. Support for this statistic by device hardware is optional.

6.2.3 SetATMDefaultVC

This request is used to pre-select the VPI/VCI value for subsequent *GetATMVCStatistics* requests. This request only applies to type 3 devices.

bmRequestType	bRequestCode	wValue	wIndex	wLength	Data
00100001B	SET_ATM_DEFAULT_VC	Zero	Interface	3	1-byte VPI followed by 2-byte VCI value

6.2.4 GetATMVCStatistics

This request is used to retrieve the ATM device statistics based on the feature selector for a pre-selected VPI/VCI as stipulated in latest preceding *SetATMDefaultVC* request. This request only applies to type 3 devices.

bmRequestType	bRequestCode	wValue	wIndex	wLength	Data
10100001B	GET_ATM_VC_ STATISTICS	Feature Selector	Interface	4	32 bit unsigned integer

Table 8: ATM VC Selector Codes

Feature selector	Code	Targets	Length of Data	Description
RESERVED	00h	None	None	Reserved for future use
VC_US_CELLS_SENT	01h	Interface	4	The number of cells that have been sent upstream to the WAN link for the specified VPI/VCI since the device has been powered on or reset. This number is a 32 bit unsigned integer, which is incremented each time a cell is sent, and will be wrapped to 0 if reaching the maximum value. Support for this statistic by the device is optional.
VC_DS_ CELLS_RECEIVED	02h	Interface	4	The number of cells that have been received downstream from the WAN link for the specified VPI/VCI since the device has been powered on or reset. This number is a 32 bit unsigned integer, which is incremented each time a cell is received, and will be wrapped to 0 if reaching the maximum value. Support for this statistic by the device is optional.

6.3 Management Element Notifications

CDC 1.2 defines the common Communications Interface Class notifications that the device uses to notify the host of interface or endpoint events. This section does not contain any additional subclass specific notifications. These requests are sent over the management element and can apply to different device views as defined by the Communications Class interface codes.

The class-specific notification codes valid for a Communications Class interface with a Communications SubClass code of ATM Networking Control Model are listed in the following Table 9, and the class codes are listed in Table 10. The other class specific notifications not listed in this table are inappropriate for a ATM Networking Control Model and shall not be sent by a device.

Table 9: Notifications — ATM Networking Control Model*

Notification	Description	Req'd/Opt	Reference
NetworkConnection	Reports whether or not the physical layer (modem, Ethernet PHY, etc.) link is up.	Required	[USBCDC1.2]
ResponseAvailable	Notification to host to issue a *GetEncapsulatedResponse* request.	Optional	[USBCDC1.2]
ConnectionSpeedChange	Reports a change in upstream or downstream speed of the networking device connection.	Required	[USBCDC1.2]

* These notifications are specific to the Communications Class.

Table 10: Class-Specific Notification Codes for ATM subclass

Request	Value
NETWORK_CONNECTION	00h
RESPONSE_AVAILABLE	01h
CONNECTION_SPEED_CHANGE	2Ah