Wireless Universal Serial Bus Specification
Errata on Revision 1.0 as of December 2006

Agere Systems

Hewlett-Packard

Intel

Microsoft

NEC

Philips

Samsung

December 2006

Check the http://www.usb.org website for the latest errata.

Chapter 4

Erratum Title: Self Beaconing devices can maintain 2 DevAddrs

Background: Self Beaconing devices can generate a MAC layer DevAddr in the range specified in the MAC Layer specification. Additionally they need to maintain a private DevAddr that is assigned by the WUSB host. For all WUSB transactions the SBD uses the WUSB assigned DevAddr, while for beaconing purposes, the SBD can use the derived DevAddr.

Change: p28, Section 4.3.7.1, add the following sentences after the first sentence of the first paragraph:

Additionally, a Self beaconing device can maintain a 16-bit MAC layer DevAddr that is derived from a 48 bit MAC Address, which it uses when transmitting beacons. This is in addition to the WUSB host-assigned DevAddr that the device uses in all its WUSB transactions.

Change: p35, Table 4.1, first row, Explanation column, must read as follows:

Wireless USB Hosts must have a full 48-bit MAC Address from which a 16-bit DevAddr is generated. The MAC Layer also assigns addresses in the upper portion of this range to Multicast and Broadcast DevAddrs. Self Beaconing devices that generate a 16-bit DevAddr must also have a full 48-bit MAC Address.

Erratum Title: No Wireless USB commands during Capture/Count Packet operation.

Background: A Wireless USB host must not send Wireless USB requests to a Directed Beaconing device while the device is in the middle of performing a Capture or Count Packet operation. This is due to the fact that while the device is performing a capture or count packet operation (between the Receive Start Time and Receive End Time) it is not capable of responding to any other Wireless USB requests that are sent to any of its endpoints. Therefore a host must only issue a command (like ClearFeature) after the Receive End Time or before the Receive Start Time.

Change: p29, Section 4.3.7.2.2, add the following sentences before the last sentence of the third paragraph:

A host is not allowed to send any Wireless USB request after the Receive Start Time and before the Receive End Time when the device is enabled for performing a Count Packet operation.

Change: p30, Section 4.3.7.2.3, add the following sentences before the last sentence of the third paragraph:

A host is not allowed to send any Wireless USB request after the Receive Start Time and before the Receive End Time when the device is enabled for performing a Capture Packet operation.

Change: p143, Section 7.3.1.1, add the following sentences after table 7-5:

A host is not allowed to send the ClearFeature request after the Receive Start Time and before the Receive End Time when the device is enabled for performing a Count or Capture Packet operation.

Erratum Title: Incorrect reference to SetWUSBData function

Background: N/A.

Change: p29, Section 4.3.7.2.1, first sentence, second paragraph must read as follows:

The host provides the complete packet data that the device transmits using the SetWUSBData(Transmit Data) standard device request (see Section 7.3.1.6).

Change: p29, Section 4.3.7.2.1, sixth sentence, third paragraph must read as follows:

The host provides the necessary time values using the SetWUSBData(Transmit Params) standard device request (see Section 7.3.1.6).

Change: p29, Section 4.3.7.2.2, last sentence, second paragraph must read as follows:

See Section 7.3.1.6 for details on this standard request and its parameters.

Change: p30, Section 4.3.7.2.3, last but one sentence, second paragraph must read as follows:

See Section 7.3.1.6 for details on this standard request and its parameters.

Erratum Title: Directed beaconing setup exercise and DBDs going to sleep

Background: The specification requires that DBD devices after waking from sleep need to resume transmitting packets if they were enabled to do so prior to going to sleep. However this might lead to DBDs transmitting the wrong information as the channel state might have changed while the DBD was asleep. This erratum addresses this issue.

Change: p29, Section 4.3.7.2.1, third sentence, fourth paragraph must read as follows:

Devices are disabled for directed packet transmission after reset, after wakeup and when they receive a ClearFeature (DEV_XMIT_PACKET) request.

Change: p29, Section 4.3.7.2.2, third sentence, third paragraph must read as follows:

Devices are disabled for counting packets after reset, after wakeup, when they have completed a count packet function, when the capture buffer is full, or when they receive a ClearFeature (COUNT PACKETS) request.

Change: p30, Section 4.3.7.2.3, third sentence, third paragraph must read as follows:

Devices are disabled for capturing a packet after reset, after wakeup, when they have captured a packet that matches the filter criteria, when the receive period has terminated, or when they receive a ClearFeature (CAPTURE PACKET) request.

Change: p82, Section 4.16.1.1, add the following paragraph at the end of the section:

Note that a host needs to go through the directed beaconing setup exercise as described in Section 4.3.7 once the device wakes up and re-establishes the connection.

Erratum Title: Incorrect reference to directed packet transmission

Background: N/A.

Change: p29, Section 4.3.7.2.2, second sentence, third paragraph must read as follows:

Hosts are responsible for appropriately initializing the device with a SetWUSBData(Receive Params) request before enabling a device for counting packets.

Erratum Title: Transmit Packet function repetition rate

Background: The specification states that after transmitting the first packet, the device will continue to transmit the packet every 64K + Transmit Adjustment microseconds. The Transmit Adjustment is calculated by the host based on the drift between its clock and the clock of the slowest beaconing device in the neighborhood. There needs to be a clarification that the 64K counting by the device needs to be synchronized to the WUSB channel time. Otherwise, the packet transmission may drift over time relative to the slowest beaconing device in the neighborhood (if the device clock is faster or slower then the host clock).

Change: p29, Section 4.3.7.2.1, add the following sentences after the fifth sentence of the third paragraph:

The counting process of the 64K + Transmit adjustment must be synchronized to the WUSB channel time, in order to prevent the packet transmission from drifting over time. One way to achieve that is by always matching transmission time to the WUSB channel time. As mentioned earlier, the first packet transmission happens when the WUSB channel time matches the time specified by the host. After transmitting the packet for the first time, the device adds 64K + Transmit Adjustment to the time specified by the host and then waits for WUSB channel time to match that value to transmit the packet again, and so on.

Erratum Title: Transaction groups must fit within a contiguous block of MAS

Background: Even though figures 4-4 on p22 and 4-5 on p23 imply that a transaction group must fit within a contiguous block of MAS, the text is not clear about this requirement.

Change: p32, Section 4.3.8.2, third sentence must read as follows:

The host must ensure that it does not schedule Wireless USB channel communications (i.e. any part of a transaction group) to cross the boundary of a permitted MAC Layer channel access period (i.e. DRP reservation).

Erratum Title: Isochronous Packet Discard IE and OUT transfers.

Background: The third paragraph is Section 4.7.5 details the host behavior when it must discard an Isochronous packet to an OUT endpoint. However the paragraph incorrectly states that a host must schedule a W_{DT}CTA to the Isochronous IN endpoint and not the Isochronous OUT endpoint.

Change: p47, Section 4.7.5, eighth sentence, third paragraph must read as follows:

The host must include at least one W_{DT}CTA for the isochronous OUT function endpoint in the same MMC.

Erratum Title: Nominal Transmit Power Levels.

Background: In Table 4-4, Power Control Setting 7 has a nominal value of TFI_BASE_14 when it should be TFI_BASE - 14.

Change: p52, Table 4-4, entry for Power Control Setting 7 must be changed as follows:

7	TFI_BASE – 14	TFI_BASE – (11.2 to 16.8) dB	FFI_BASE – 14 dB	FFI_BASE – (11.2 to 16.8) dB

Erratum Title: Last Packet Flag in a burst

Background: This erratum is being made so as to make it easier for device and host implementations to handle bursts.

Change: p54, Section 4.10.2, the following sentence must be added to the first paragraph:

Note that a transmitter can only set the *bmStatus.Flags.Last Packet Flag* field in the Wireless USB data header of a packet in a burst if it is the last packet in that burst.

Erratum Title: Keep Alive IEs and DN_Alive Notifications.

Background: To make Host and device implementations easier and more robust with regard to Keep Alive IEs sent by a host and DN_Alive notifications sent by a device the following changes need to be made to the specification.

Change: p79, Section 4.14, last sentence in paragraph 7 must read as follows:

Similarly, a device will continue to transmit *DN_Alive* or equivalent notifications until the host removes the device address from subsequent Keepalive IEs or the device observes an MMC without a Keepalive IE or it experiences a *TrustTimeout*.

Change: p182, Section 7.5.9, last sentence in the first paragraph must read as follows:

This IE must be included in every MMC until host software removes it.

Change: p191, Section 7.6.7, first sentence in the Stop Retransmission Condition must read as follows:

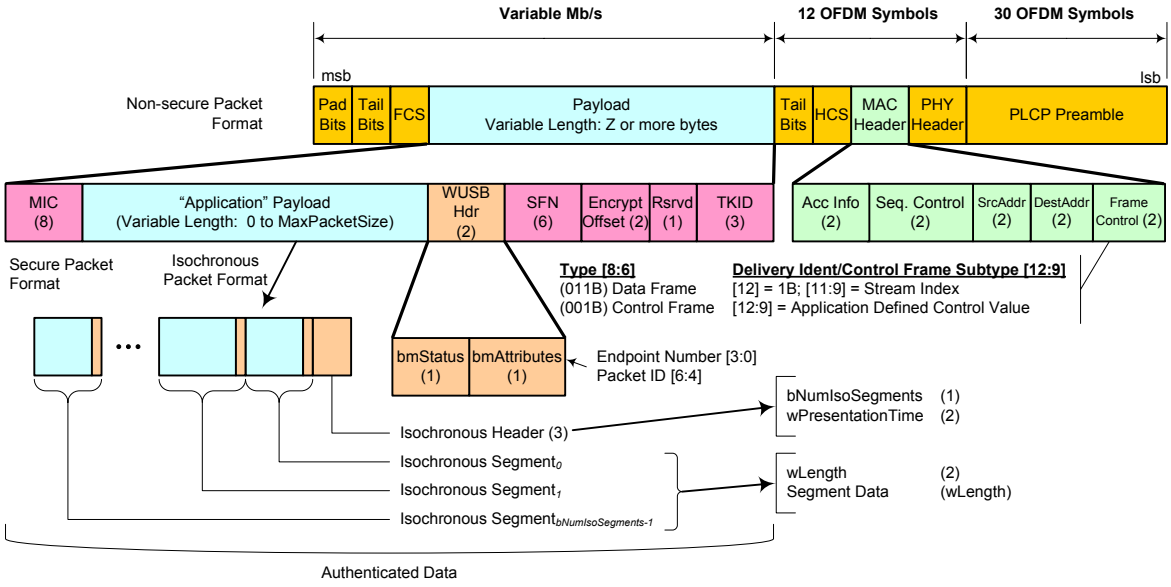
If in response to a Keepalive_IE, a device will cease transmitting *DN_Alive* notifications when it no longer observes a Keepalive IE that includes its device address, or it observes an MMC without a Keepalive_IE in it.

Chapter 5

Erratum Title: Figure 5-1 incorrectly marks Authenticated data as “encrypted” data

Background: N/A.

Change: p87, Section 5.1, Figure 5-1 must look like:



Erratum Title: Incorrect Cross reference Number of W_XCTAs in a CTA IE

Background: The reference is intended to direct the reader to the section(s) where device notifications are explained in detail. The correct reference for this description is to Section 5.5.3. In addition there is a cross reference to Section 5.3 that is not required.

Change: p93, Section 5.1, first paragraph after Table 5-4 must read as follows:

- A device notification packet. Device notification packets are encoded as *Data Frame* in the MAC Header. Device notification packets can only be transmitted by a device, during a W_{DNTS}CTA time slot (see Sections 5.2.1). Device notification packets are transmitted using secure packet encapsulation unless explicitly specified otherwise. Device notification packets must include the Wireless USB Header with the *bmAttributes.PID* field set to DN and the *bmAttributes.EndpointDirection* field must be set to 0B by the device and may be ignored by the host. Note that some device notifications are transmitted without secure packet encapsulation because they are transmitted outside of the secure relationship (i.e. like the Connect notification). Refer to Section 5.5.3 for details about device notifications. The data payload portion of the packet is used to convey specific notification information from the device to the host. When secure packet encapsulation is present, the Encryption Offset field in the Security Header is set to the length of the Wireless USB header plus the length of the notification payload. In short, the entire packet is transmitted in plain text in form similar to an MMC (see above).

Erratum Title: Number of W_XCTAs in a CTA IE

Background: The Wireless USB Specification 1.0 states that a CTA IE consists of one or more W_XCTAs. However, a CTA IE must contain at least two CTAs - one W_XCTA and the other an End-Of-List CTA.

Change: p93, Section 5.2.1, paragraph after Figure 5-3 must read as follows:

The *bLength* field value includes the total length of the Wireless USB Channel Time Allocation information element, including the *bLength* field. A Wireless USB Channel Time Allocation IE is comprised of two or more W_XCTA blocks (Wireless USB Channel Time Allocation blocks). W_XCTA blocks describe a time slot allocation relative to the MMC. The general structure of a W_XCTA and the relationship between its information and the described time slot is illustrated in Figure 5-4.

Erratum Title: Transmit power for handshake packets

Background: p97 and p98, Section 5.2.1.2, the specification only specifies the transmit power for data packets, but it does not specify the transmit power for handshake packets.

Change: p97, Table 5-7, entry for the *Transmit Power* field must be changed as follows:

4	*bmTXAttributes*	4	Bitmap	These sub-fields indicate the use parameters of the data phase time slot: **Bit** — **Description** 23:21 — **Transmit Power.** The value of this field selects the transmit power level the device must use to transmit data/handshake packets during the data/handshake phase protocol time slot.

Change: p98, Section 5.2.1.2, the third paragraph from the top of the page must read as follows:

The *bmTXAttributes.Transmit Power* field is used to specify the transmit power level the device must use to transmit all of the data/handshake packets during the associated data/handshake phase protocol time slot. In general, a value of zero (0) selects the highest power setting and a value of seven (7) selects the lowest. Refer to Section 4.10.1 for details.

Erratum Title: wStart time in a blank W_XCTA

Background: In order for devices to be able to detect a blank W_XCTA it is necessary for the wStart times in these CTAs to be set to the same value as that in the next CTA in the CTA IE.

Change: p109, Section 5.5, the last sentence in the last second paragraph must read as follows:

A *blank* W_{DT}CTA is one that allocates no channel time with the *wStart* field in this CTA set to same value as that in the next W_XCTA in this CTA IE. It serves only to acknowledge the packets received in the previous data phase.

Change: p110, Section 5.5, the second sentence in the last paragraph must read as follows:

A *blank* W_{DR}CTA is a W_{DR}CTA with no channel time allocation with the *wStart* field in this CTA set to the same value as that in the next W_XCTA in this CTA IE.

Erratum Title: Device receive window in response to blank W_{DR}CTA

Background: A blank W_{DR}CTA requires special attention by a device, since the device behaves as if it received a blank packet with the next sequence number. This erratum is intended to bring attention to that case.

Change: p110, Section 5.5, the third sentence in the last paragraph must read as follows:

The receiving device behaves as if an actual zero-length transfer has occurred by advancing it's receive window to acknowledge seeing the next sequence number in the transfer.

Erratum Title: Correction on setting of the Endpoint Direction field in the Wireless USB header when transmitting handshake packets during a control transfer.

Background: N/A.

Change: p113, Section 5.5.2, the fourth sentence in the first paragraph must read as follows:

The device must set the *rWUSBHeader.Endpoint Direction* field to a 1B for a handshake packet transmitted during the status stage of a control transfer.

Erratum Title: Handshake packet returned from a previously flow controlled OUT endpoint.

Background: The specification does not clearly specify the requirements on a handshake packet returned by an OUT endpoint (which previously returned a flow control response) when the host only requests a handshake packet via a W_{DT}CTA. If an ACK packet with the bvAckCode field set to zero is returned even though the endpoint previously returned an NAK, the host may incorrectly determine that all data packets were transferred successfully in the previous transaction.

Change: p115, the following paragraph must be added at the end of Section 5.5.4:

A host may at any time request the device to return its buffer availability by scheduling a W_{DT}CTA after an OUT function endpoint returns a flow control response. If the function endpoint does not have any buffer space available for at least one data packet, then it must respond with the same handshake value it used for the previous flow control response. If it does have buffer space available, then it must return an ACK handshake packet with a non-zero bvAckCode indicating the current buffering availability.

Chapter 6

Erratum Title: Correction on public key encryption methods used.

Background: The information about public key encryption in this chapter has been superseded by the association supplement.

Change: p121, Section 6.2.4, remove second, third, and fourth paragraphs.

Erratum Title: PK Association Key and Symmetric Association Key no longer relevant.

Background: The information about the PK Association Key and Symmetric Association Key in the specified sections has been superseded by the association supplement.

Change: p121, Section 6.2.6.1.1, replace entire section text with the following text:

"The information in this section has been superseded by the association models supplement."

Change: p121, Section 6.2.6.1.2, replace the entire section text with the following text:

"The information in this section has been superseded by the association models supplement."

Erratum Title: Text ambiguous as to which key to be used to perform Handshake 3 during the 4-Way Handshake.

Background: N/A.

Change: p126, Section 6.2.10.9.1, 3rd paragraph must read as follows:

The host initiates Phase 2 of the 4-way handshake by asking the device for *DNonce*. The device returns the TKID and *DNonce*. It also adds a third item to the returned data. It uses KCK to compute a message integrity code (MIC) over the entire packet payload. The format of the packet payload is described in Table 7-22.

Change: p126, Section 6.2.10.9.1, 5th paragraph must read as follows:

At the end of Phase 2, the host and device have both derived initial session keys and the host has proof that the device holds the correct CK. Phase 3 is used to provide proof to the device that the host also holds the correct CK and to instruct the device to install the derived key. The host initiates Phase 3 by constructing a message containing the TKID, *HNonce*, and a MIC computed over the entire packet payload with the KCK. It sends this message to the device. The device validates the received MIC by also performing the MIC computation. If the MICs do not match, the device silently disconnects. If the MICs match, the device installs the derived session key. The Host is allowed to install the derived session key after successful completion of Phase 2 but must be prepared to remove it if Phase 4 does not complete successfully.

Erratum Title: Typo in parameters passed to PRF-256 function.

Background: The first parameter specified in section 6.5.1 is incorrectly specified.

Change: p135, Section 6.5.1. The first bullet must read as follows:

"K – The master key (MK) being used for derivation"

Erratum Title: Random numbers must be based on sources of physical entropy.

Background: To avoid any ambiguity, an explicit reference to physical noise sources is being added to this section.

Change: p136, Section 6.5.3. Insert the text below after this sentence: "It recommends collecting random samples from multiple sources followed by conditioning with PRF.":

The randomness samples must be derived from a physical entropy source, such as RF noise, thermal noise, or other unpredictable physical sources of entropy.

Erratum Title: Clarification of values used in example RNG

Background: The explanation of nonce construction in the example is not clear.

Change: p136, Section 6.5.3. Append the following paragraph:

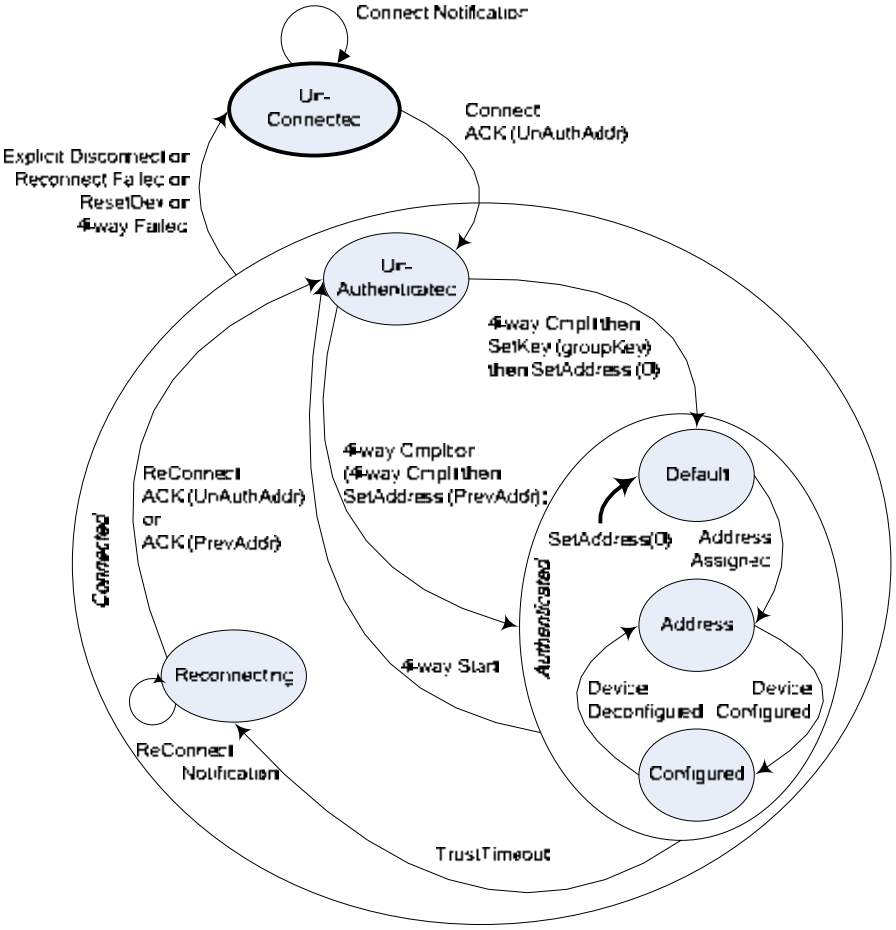
The construction of N in this example differs from the nonce construction for transmitted data because this is an internal re-use of the CCM logic. The initial value of N in this example could be 0, but adding a non-zero component serves to bind the results to this specific usage. This example uses two copies of DevAddr for this purpose because it is typically readily available on all devices before a connection is made. The initial value for N in a true implementation is at the discretion of the designer.

Chapter 7

Erratum Title: Wireless USB Device States and 4-Way Handshakes.

Background: 4-Way handshakes are always performed in clear text (i.e. they are unencrypted) and a host may choose to re-authenticate a previously authenticated device at any time. However there is currently no state transition from the **Authenticated** state the **Un-Authenticated** state when a 4-way handshake is initiated.

Change: p137, Section 7.1, Figure 7-1 must look like:



Change: p139, Section 7.1.3, third paragraph must read as follows:

Note that the host may initiate a 4-way handshake at any time with the device, including while it is in any substate of the **Authenticated** device state. The control transfers used to conduct the 4-way handshake do not use secure data encapsulation during the data and handshake phases of the control transfers. A device must transition to the **UnAuthenticated** state when the host starts a 4-way handshake and all the keys the device has must be discarded.

Change: p139, Section 7.1.3, add the following bullet point under the conditions for exiting the **Authenticated** state:

- **Reception of Set Handshake 1**. The device must transition to the **UnAuthenticated** state when the host starts a 4-way handshake.

Erratum Title: Standard Request availability in Wireless USB Device States.

Background: The Set and Clear Feature commands are marked as available in the **UnAuthenticated** state in Table 7-1 however the sections that gives details on these commands clearly states that they are not available in the **UnAuthenticated** state. This erratum addresses this issue.

Change: p140 and p141, Table 7-1, entry for CLEAR_FEATURE and SET_FEATURE must be changed as follows:

CLEAR_FEATURE	No	
SET_FEATURE	No	

Erratum Title: The data format used to set the DRPIE Information when performing a Set WUSB must be updated to match the WiMedia MAC Specification.

Background: Since the host can set multiple DRP IEs using this command and each DRPIE may be safe or unsafe and the conflict tie-breaker may be different from one DRPIE to the next the command was changed to send the actual DRP IE to the device.

Change: p150, Section 7.3.1.6, text under the DRPIE Information heading must read as follows:

DRPIE Information

A self-beaconing device uses this data in conjunction with data it derives from its host's Wireless USB channel to construct the DRP IE(s) that it transmits in its beacon. The data contains a variable size array containing one or more DRP IEs. A host may issue this command regardless of the current setting of the *TX DRP IE* feature. The device must include the new data within the next 2 Beacon transmissions if the *TX DRP IE* feature is enabled. The Self Beaconing device should not modify the contents of the received DRP IE(s) as long as it can keep sending them, however it may modify the DRP IE(s) if necessary, e.g. beacon size will exceed due to addition/modification of other IEs.

Table 7-15. DRP IE WUSB Data Format

Offset	Field	Size	Value	Description
0	*DRP IE 0*	Var0	Bitmap	This is the first DRP IE transmitted by the device (including the Element ID and Length). See Table 7-64 for full layout of a cluster member DRP IE layout.
…				
Var	*DRP IE N*	VarN	Bitmap	This is the last DRP IE transmitted by the device.

Note, if the Self Beaconing device does not have an existing DRP IE for this Wireless USB channel, it simply adds the received DRP IE to its beacon. If the device has an existing DRP IE for this Wireless USB channel, then it must replace all the existing DRP IEs (for this Wireless USB channel) with the new DRP IE(s) provided in this command payload.

Erratum Title: Cross-reference to the Data Loopback commands incorrect.

Background: In Section 7.3.1.7 and 7.3.1.8, the cross reference points to the Isochronous Endpoint Host System Admission Decisions and not the Data loopback commands section.

Change: p152, Section 7.3.1.7, second paragraph after the table describing the command must read as follows:

For full requirements for the Data Loopback Write command and device behavior with stored loopback data refer to Section 4.8.4.

Change: p152, Section 7.3.1.8, second paragraph after the tables describing the command must read as follows:

For full requirements for the Data Loopback Write command and device behavior with stored loopback data refer to Section 4.8.4.

Erratum Title: Clarification on Set Encryption and UNSECURE encryption mode.

Background: The specification does not clearly state what a device must do with the keys that it might have when it receives a Set Encryption. In addition, the specification allows a device to be set to the UNSECURE mode, which was useful only when we needed to support public key and private key authentication scenarios. These authentication methods are now obsolete and hence the ability to set the device to the UNSECURE encryption mode is no longer required.

Change: p154, Section 7.3.2.2, second paragraph after the table describing the command must read as follows:

Encryption Value comes from one of the Encryption Type descriptors contained in the Security Descriptor. A value of Zero in this field is undefined. The request is valid for any device in the **Connected** device state. All the keys that the device has remain valid until the device transitions to the **UnAuthenticated** state. The device must continue to use the encryption type that is set via this command until the device transitions to the **UnConnected** state.

Erratum Title: The Request Error conditions for Get Encryption are incorrect

Background: N/A

Change: p154, Section 7.3.2.3, the text after the table describing the command must read as follows:

Encryption Value comes from one of the Encryption Type descriptors contained in the Security Descriptor. A value of Zero always represents UNSECURE or no encryption.

A wired/wireless device always returns WIRED if it connected with a cable.

It is a Request Error if *wValue* or *wIndex* are other than as specified above.

UnAuthenticated State: This is a valid request when the device is in the **UnAuthenticated** state.

Default State: This is a valid request when the device is in the **Default** state.

Address State: This is a valid request when the device is in the **Address** state.

Configured State: This is a valid request when the device is in the **Configured** state.

Erratum Title: Cross-reference to the endpoint companion descriptor is incorrect.

Background: N/A.

Change: p161, Table 7-24, entry for ENDPOINT must be changed as follows:

ENDPOINT	Changes	Updates required for Wireless USB. Also require extensions to endpoint capabilities which utilize a companion descriptor, see Section 7.4.4.

Erratum Title: Removal of RSA_1 as a valid Encrytion Type.

Background: N/A.

Change: p174, Table 7-35, entry for RSA_1 must be changed as follows:

Reserved	3	Reserved

Erratum Title: Minimum number of IEs required in an MMC packet.

Background: In Table 7-37, it is required that an MMC packet must contain at least one IE, however it would easier for host implementations to able to send MMCs with no IEs so that they can properly schedule any IEs included added via Add MMC IE commands (see Section 8.5.3.1) at the specified interval and schedule Device Notification Time Slots with the interval specified via a Set Num DNTS Slots command (see Section 8.5.3.10). Moreover an MMC packet by itself is meaningful for maintaining clock synchronization between host and devices.

Change: p175, Table 7-37, entry for IE must be changed as follows:

10	*IE[0 to n-1]*	Var	MMC IE	Array of information elements (if present).

Erratum Title: Cross-reference to the rules for generating a CHID is incorrect.

Background: In Section 7.5.2, the cross reference points to the Replay Prevention section and not one that defines a CHID or the rules for generating one.

Change: p178, Section 7.5.2, first paragraph after the table must read as follows:

Note, connect includes any *DN_Connect* device notification where the *New Connection* bit is a zero. The guidelines for generating a unique CHID are detailed in Section 4.15.2.1.

Erratum Title: Start time of Wireless USB Release Channel IE.

Background: The specification incorrectly states that the wStart field in Wireless USB Release Channel IE gives the Wireless USB Channel time at which the addressed device must transmit a UDR packet. The wStart time is actually specified as an offset from the beginning of the MMC that contains this IE. This erratum addresses this issue.

Change: p180, Table 7-44, entry for wStart must be changed as follows:

0	*wStart*	2	Number	The units of this field are in micro-seconds. The value is measured from the beginning of the preamble of the MMC packet where this Wireless USB Release Channel IE is transmitted. The addressed device must transmit a UDR control packet at this time.

Erratum Title: Device Notifications and Priority.

Background: DN_MASAvailabilityChanged has higher priority than DN_EPRdy and must be sent first, however the specification states otherwise.

Change: p185, Section 7.6, last sentence on page must read as follows:

For example, if a device has simultaneous pending *DN_EPRdy* and *DN_MASAvailableChanged* notifications, it must send the *DN_MASAvailableChanged* message first because it has a higher priority. Note that a device must never send more than one notification message per DNTS period.

Erratum Title: DN_EPRdy Maximum Retransmit Rate and Stop Transmission Condition.

Background: The change to the Maximum Retransmit Rate is required to stop a flood of DN_EPRdy notification from endpoints that sent a flow control response at some earlier time but the host has no data or buffer space available when the DN_EPRdy is received. The change to the Stop Transmission condition is being made to make it easier for device implementations.

Change: p189, Section 7.6.3, the Maximum Retransmit Rate and Stop Retransmission Condition must read as follows:

- **Maximum Retransmit Rate:** a device should retransmit this notification at every DNTS opportunity provided by the host with a 50 millisecond duty cycle i.e. the device must send at every DNTS opportunity for the first 50 milliseconds and then should not send this notification for the next 50 milliseconds (and so-on).
- **Stop Retransmission Condition:** a device will cease retransmission attempts of this notification when it observes the host has resumed transactions to any of the listed function endpoint(s) or the host has not responded to the *DN_EPRdy* for an endpoint for *TrustTimeout* seconds.

  Note that if the host resumes transactions to some of the listed function endpoints (but not all) in this DN_EPRdy notification, then it is acceptable for the device to transmit a new DN_EPRdy notification that only includes the endpoints that are still flow controlled.

Erratum Title: DRP Control field must be updated to match the WiMedia MAC Specification.

Background: N/A.

Change: p192 and p193, Section 7.7, Table 7-63, the note after it and Table 7-64 must read as follows:

Table 7-63 Host Wireless USB MAC Layer DRP IE Settings

Offset	Field	Size	Value	Description
0	*Element ID*	1	Constant	Distributed Reservation Protocol IE (0x09)
1	*Length*	1	Number	This field contains the length of this descriptor, not including the Element ID and Length (4+4×N), where N is the number of DRP allocation blocks.
2	*DRP Control*	2	Bitmap	**Bits** / **Name** / **Value** 2:0 Reservation Type 011B (Private) 5:3 Stream Index Assigned 8:6 Reason Code 000B 9 Status 1B 10 Owner 1B 11 Conflict Tie-breaker Assigned 12 Unsafe Variable, Note 1 15:13 Reserved 000B
4	*DevAddr*	2	Number	Wireless USB Broadcast Cluster ID
6	*DRP Allocation*	Var	Record	Array of DRP Allocations. Refer to reference [3] for details.

Note 1. The value for the *Unsafe* field must be set to an appropriate value in compliance with the reservation policy defined by [3].

Table 7-64 Wireless USB Cluster Member MAC Layer DRP IE Settings

Offset	Field	Size	Value	Description
0	*Element ID*	1	Constant	Distributed Reservation Protocol IE (0x09)
1	*Length*	1	Number	This field contains the length of this descriptor, not including the Element ID and Length (4+4×N), where N is the number of DRP allocation blocks.
2	*DRP Control*	2	Bitmap	**Bits** / **Name** / **Value** 2:0 Reservation Type 011B (Private) 5:3 Stream Index Same as host 8:6 Reason Code 000B 9 Status 1B 10 Owner 0B 11 Conflict Tie-breaker Same as host 12 Unsafe Same as host 15:13 Reserved 000B
4	*DevAddr*	2	Number	Host DevAddr
6	*DRP Allocation*	Var	Record	Set up by host, see Section 4.3.8.4.

Chapter 8

Erratum Title: DWAs and Sleep requirements.

Background: In order to reduce the amount of polling a host performs and since there is currently no way for host software to tell a device to go to sleep, a DWA must go to sleep when no devices are connected to its downstream ports or when all devices connected on its downstream ports are suspended.

Change: p202, Section 8.1.10.1, the following text must be added to the end of the first paragraph:

Note that a DWA (once it is configured and all its downstream ports are powered) must go to sleep state if either of the following conditions occurs:

- All of its downstream ports are in the unconnected state
- All of its downstream ports that have devices connected to them are suspended

Erratum Title: Clarification of Wire Adapter status after a successful WIRE_ADAPTER_RESET.

Background: N/A.

Change: p204, Section 8.1.11, last paragraph must read as follows:

To reset just the host controller in a Wire Adapter, the host issues the class specific Set Feature(WIRE_ADAPTER_RESET) request. On reception of this command, the Wire Adapter must terminate any transfers intended for any downstream endpoints and all RPipes must transition to the unconfigured state. The Wire Adapter must return power on default values for the RPipe descriptors if queried after the Reset completes. Any data to be transferred on its upstream endpoints must be discarded and the data sequence as well as buffer availability values on its upstream endpoints must return to their initial configured values. Further, in the case of a DWA, all the downstream ports must transition to their power on default state. In the case of an HWA, all security keys and device information buffer contents set by host software must be discarded. After the reset completes, the WA must transition to the **Disabled** state.

Erratum Title: The RPipe State diagram does not include all the states and state transitions possible.

Background: The following state transitions were not present in the RPipe State Diagram:

- The RPipe does not change state when it is reconfigured via a Set RPipe Descriptor request
- An RPipe in an HWA must transition to the **Paused** state on reception of a flow control response
- An RPipe in an HWA must transition to the **Active** state from the **Paused** state on reception of an EP Ready notification from a previous flow controlled endpoint
- A Wire Adapter must transition the RPipe state to **Stall** when a transfer is completed due to an Error response from the targeted endpoint
- A Wire Adapter must transition the RPipe state to **Idle** when it receives a Clear Feature: Stall request

Change: p206, Table 8-6 must be changed as follows:

Table 8-6 Wire Adapter Class Feature Selector

Feature Selector	Recipient	Value
WIRE_ADAPTER _ENABLE	Wire Adapter Device	1
WIRE_ADAPTER _RESET	Wire Adapter Device	2
RPIPE_PAUSE	RPipe	1
RPIPE_STALL	RPipe	2

Change: p207, Section 8.3.1.2, the second paragraph after the table describing the command must read as follows:

The Wire Adapter must transition the state of the RPipe from its current state to the new state (see Figure 8-6) depending on the Feature being cleared; see Table 8-6 for the feature selector definitions that apply to an RPipe as a recipient. Features that can be cleared with this request are:

- RPIPE_PAUSE
- RPIPE_STALL

Change: p208, Table 8-7 must be changed as follows:

Table 8-7 RPipe State Report

Offset	Field	Size	Value	Description
0	*RPipeState*	1	Bitmap	State of this RPipe: **Bit** — **Description** 0 — 1 = Idle, 0 = Active 1 — 1 = Paused, 0 = Not Paused 2 — 1 = Configured 0 = UnConfigured 3 — 1 = Stalled, 0 = Not Stalled 7:4 — Reserved

Once an RPipe is configured then that RPipe can only be in of three states: **Paused**, **Stalled** or **Not Paused**

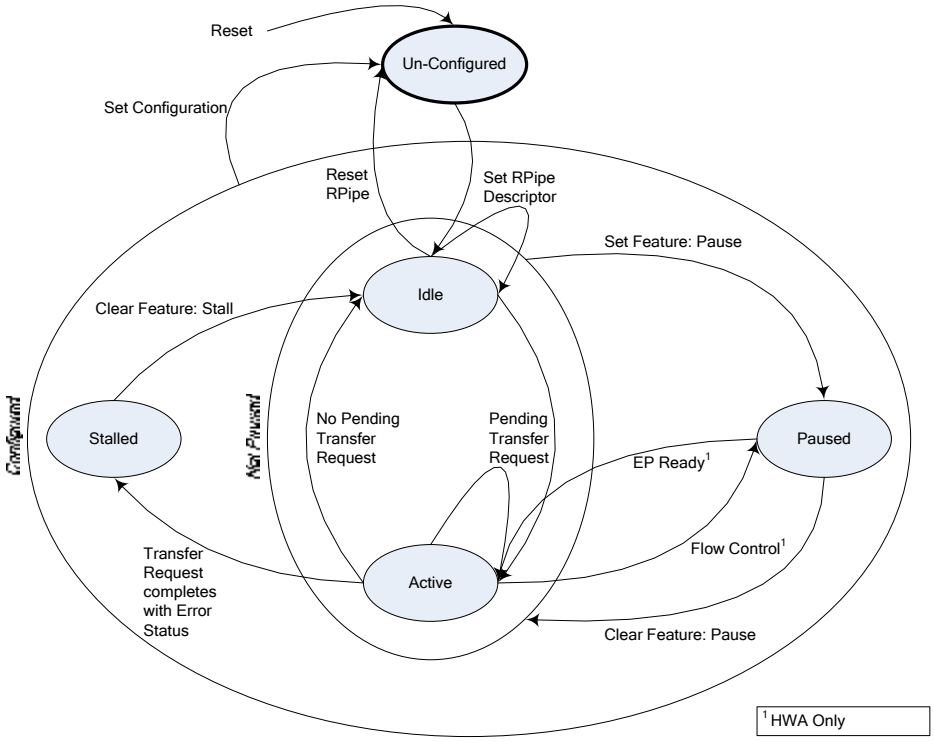
A Wire Adapter must not perform any transactions to the endpoint that an RPipe is targeted at if that RPipe is in the **Paused** state. The RPipe must be transitioned out of the **Paused** state and into one of the two substates of the **Not Paused** state when the Wire Adapter receives a Clear Feature: RPIPE_PAUSE command.

However if an RPipe in an HWA was transitioned to the **Paused** state due to the reception of a flow control response from the endpoint targeted by that RPipe, then that RPipe must be transitioned back to one of the two substates of the **Not Paused** state when the HWA receives an DN_EPRdy from that endpoint.

Similarly, a Wire Adapter must not perform any transactions to the endpoint that an RPipe is targeted at if that RPipe is in the **Stalled** state. The RPipe must be transitioned to this state when a transfer completes with an error condition (e.g. STALL response from the targeted endpoint or maximum number or retries is exceeded for a transaction etc).

It is the responsibility of the WA driver to abort any pending transfer requests if necessary, perform any operation to clear the error condition on the targeted endpoint and finally transition that RPipe back to the **Idle** state by sending the Wire Adapter a Clear Feature: RPIPE_STALL command.

Change: p209, Section 8.3.1.5, Figure 8-6 must look like:



Erratum Title: Cleared up the location of the note in Table 8-15. Added a new Transfer Error Status for discarded packets.

Background: N/A.

Change: p215 and p216, Table 8-15 must be changed as follows:

Bit	Description
5:0	**Status Value** — **Description** 0 — TRANSFER_STATUS_SUCCESS The transfer completed successfully. Bit 6 and 7 are set to zero. 1 — TRANSFER_STATUS_HALTED This means that the endpoint that this transfer was attempted on is currently halted. 2 — TRANSFER_STATUS_DATA_BUFFER_ERROR There was a data buffer under/over run. 3 — TRANSFER_STATUS_BABBLE A babble was detected on the transfer. This could be either Frame babble or Packet babble or both. 4 — Reserved 5 — TRANSFER_STATUS_NOT_FOUND Returned as a response to an Abort Transfer request that has an invalid or already completed TransferID. 6 — TRANSFER_STATUS_INSUFFICIENT_RESOURCE Returned in the transfer result when the Wire Adapter could not get enough resources to complete a previously accepted transfer request.

Table 8-15. Transfer/Packet Status (cont.)

Bit	Description
5:0	7 TRANSFER_STATUS_TRANSACTION_ERROR Returned in the transfer result when the Wire Adapter encountered a transaction error while performing this transfer. **Bits** **Description** 7:6 Indicates whether this was an error or a warning. **Value** **Meaning** 00B Undefined 01B The transfer completed successfully but transaction errors occurred which were successfully retried. 10B The transaction failed after the number of retry attempts specified in *bmRetryOptions* field of the RPipe descriptor. 11B Undefined Timeout, Bad PID, CRC error are examples of DWA transaction errors. Timeout, Bad PID, FCS error, Bad sequence number are examples of HWA transaction errors. 8 TRANSFER_STATUS_ABORTED The transfer was aborted by an Abort Transfer Request or by an AbortRPipe command. 9 TRANSFER_STATUS_RPIPE_NOT_READY The transfer request was sent to an unconfigured RPipe. 10 INVALID_REQUEST_FORMAT This status may be sent back for one of two reasons: • The transfer request length was not equal to the length field for the specified request type • The request type was unknown. 11 UNEXPECTED_SEGMENT_NUMBER The transfer request segment numbers were not received in incrementing order starting with zero. 12 TRANSFER_STATUS_RPIPE_TYPE_MISMATCH The transfer type in the transfer request did not match the transfer type that the RPipe was previously configured to. 13 TRANSFER_STATUS_PACKET_DISCARDED This indicates that the HWA was unable to transmit an Isochronous packet by its presentation time. This error code must only be set in the Packet Status for the packet that was discarded. If all the packets in the transfer request were discarded then the transfer status must be set to this value as well. Note that this error code must only be used in the result field if this was the result of an Isochronous OUT Transfer request. 14-63 Reserved Note unless mentioned otherwise, bit 7 is set for all status values.
6	**Warning** This bit is set when the status is warning.
7	**Error** This bit is set when the status is error

Erratum Title: Class, Subclass and Protocol values in the Device Descriptor of a DWA that does not support Isochronous streaming.

Background: N/A

Change: p218, Section 8.4.3.1, last paragraph must read as follows:

If the Device Wire Adapter does not support downstream Isochronous endpoints then *bDeviceClass*, *bDeviceSubClass* and *bDeviceProtocol* must be set to zero in the device descriptor.

Erratum Title: Legal values for the number of Isochronous endpoints supported by a DWA.

Background: The total number of endpoints that a Wireless USB device may have is limited to 31. In a DWA 4 of these are already used and hence only 27 are available to support Isochronous streaming. The specification incorrectly states that 28 are available.

Change: p225, Table 8-28, entry for bNumEndpoints must be changed as follows:

4	*bNumEndpoints*	1	Number	Number of endpoints used by this interface. Legal values are 1 through 27.

Erratum Title: Clarification on bOverTheAirInterval Description for Isochronous Streaming IN and OUT endpoints in a DWA.

Background: N/A.

Change: p226 and p227, Table 8-30 and Table 8-32, entry for bOverTheAirInterval must be changed as follows:

8	*bOverTheAirInterval*	1	00H	Interval for polling endpoint for data transfers. This is a dynamic switching capable endpoint and hence the value must be set to zero. See Section 7.4.4 for details.

Erratum Title: Muli-TT hub support behind a DWA.

Background: There is currently no way for a DWA to determine if a full or low speed endpoint that is a target of an RPipe is connected behind a high speed hub with Multi-TT support enabled.

Change: p221, Table 8-21, entry for bmAttributes must be changed as follows:

5	*bmAttributes*	1	Bitmap	**Bit**	**Description**
				0	Logical Power Switching Mode 0: Ganged power switching (all ports' power at once) 1: Individual port power switching
				1	Over-current Protection Mode 0: Global Over-current Protection. The Wire Adapter reports over-current as a summation of all ports' current draw, without a breakdown of individual port over-current status. 1: Individual Port Over-current Protection. The Wire Adapter reports over-current on a per-port basis. Each port has an over-current status.
				2	Port Indicators Supported 0: Port Indicators are not supported on its downstream facing ports and the PORT_INDICATOR request has no effect. 1: Port Indicators are supported on its downstream facing ports and the PORT_INDICATOR request controls the indicators.
				3	Multi-TT Supported 0: Multi-TT is not supported on its downstream. 1: Multi-TT is supported on its downstream.
				7:4	Reserved

In addition, p229, Table 8-33, entry for bmAttribute must be changed as follows:

23	*bmAttribute*	1	Bitmap	**Bit**	**Description**
				1:0	**Value** / **Transfer Type** 00B Control 01B Isochronous 10B Bulk 11B Interrupt
				5:2	If the transfer type is Isochronous, then this field indicates the associated upstream Isochronous endpoint on the DWA. This field is reserved and must be set to zero for all other transfer types.
				6	Reserved
				7	If this bit is set to one then the endpoint that is the target of this RPipe is connected behind a hub that has multi-tt support enabled. This field is only valid if the *bHSHubAddress* field is not set to zero and the *bSpeed* field is not set to **High-Speed**. Host software must not set this bit to one if the *bmAttributes.Multi-TT Supported* field is set to zero in the Wire Adapter Class Descriptor.

Erratum Title: Cross-reference to Feature Selector Table is incorrect.

Background: In Section 8.4.4.1, the feature selectors referred to are not for Ports but for Wire Adapters and RPIpes.

Change: p231, Section 8.4.4.1, second paragraph from the top must read as follows:

It is a Request Error if *wValue* is not a feature selector listed in Table 11-17 of the USB 2.0 Specification, if *wIndex* specifies a port that does not exist, or if *wLength* is not as specified above.

Erratum Title: Sending of Remote Wake notification by a DWA.

Background: The specification states that the DWA “should” send a Remote Wake notification when it should state that the DWA “must” send a Remote Wake notification.

Change: p232, Section 8.4.5.1, first paragraph must read as follows:

When the Device Wire Adapter detects a remote wake from any of its downstream connected devices and it is armed for remote wake then it must send a Remote Wake notification to the host. The format of this notification is shown in Table 8-37.

Erratum Title: HWA MaxPacketSize setting for the default endpoint.

Background: N/A.

Change: p236, Table 8-39, entry for bMaxPacketSize0 must be changed as follows:

7	*bMaxPacketSize0*	1	40H	Maximum packet size for endpoint zero

Erratum Title: An HWA must support remote wakeup.

Background: Bit 5 in the bmAttributes of an HWA's Configuration descriptor must be set to one due to the fact that all HWAs must support remote wakeup.

Change: p238, Table 8-41, entry for bmAttributes must be changed as follows:

7	*bmAttributes*	1	Bitmap	Configuration characteristics D7: Reserved (set to one) D6: Self-powered D5: Remote Wakeup (must be set to one) D4...0: Reserved (reset to zero)

Erratum Title: Ordering of IEs that are added via the Add MMC IE Command is not specified.

Background: When an IE block is added it is not clear where in the MMC generated by the HWA that this IE Block must be added. In addition it is not clear that an IE Block may contain one or more IEs.

Change: p245, Section 8.5.3.1, the text in this section must be replaced with the following text:

This request is used to add/modify Information Elements to be sent by the HWA in subsequent MMCs.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
00100001B	ADD_MMC_IE	Interval and Repeat Count	IE Handle and Interface Number	IE Length	IE Block

Upon receipt of this request, the HWA will store the Information Element (IE) block of data internally. An IE block may contain one or more IEs. The upper byte of the *wValue* field (Interval) specifies the rate at which the HWA must include this IE block in an MMC. This field is expressed in milliseconds. The lower byte of *wValue* (Repeat Count) specifies the number of consecutive MMCs that this IE block must be sent in during each interval. The HWA operational requirements when it gets this request are given in Table 8-53.

Table 8-53 WA Add MMC IE Request Operational Requirements

Interval	Repeat Count	Operational Requirement
0	Number	The HWA must send this IE block in every MMC. The value in the Repeat Count field must be ignored.
1-254	Number	The HWA must send this IE block every *Interval* period. The *Repeat Count* field specifies the number of consecutive MMCs that this IE block must be sent in per Interval.
255	Number	This value must be treated as an infinite period. The HWA must only send this IE block the number of times specified in the *Repeat Count* field.

The upper byte of *wIndex* field uniquely identifies an IE block. This is the handle to be used by the host software when it needs to modify or remove this IE block from subsequent MMCs. The total number of IE blocks supported by the HWA is specified by *bNumMMCIEs* specified in the HWA's Wire Adapter descriptor.

The HWA must add the IE blocks that are to be sent in the MMC (based on the *Repeat Count* and *Interval* values specified when the IE block was added) after all the WxCTA IEs in ascending IE Handle order.

The lower byte of *wIndex* specifies the target interface number.

It is a Request Error if *wIndex* specifies an IE Handle that is greater than (*bNumMMCIEs* – 1).

If the interface specified does not exist, then the device responds with a Request Error.

If the HWA is not configured, the HWA's response to this request is undefined.

Erratum Title: Redundant Device Address information in HWA RPipe Descriptor.

Background: The device address information is stored in the RPipe descriptor as well as the Device Information Buffer and the specification does not state which one an HWA should use when performing transfers to the endpoint targeted by that RPipe. To avoid confusion when the two values are different, the device address field in an HWA RPipe descriptor is being changed to an unused Reserved field.

Change: p241, Table 8-49, the entry for bDeviceAddress must be changed as follows:

13	*bReserved*	1	Number	This field is reserved and must be set to zero.

Erratum Title: The text above and below the table in Section 8.5.3.3 contradicts itself.

Background: N/A.

Change: p246, Section 8.5.3.3, the first paragraph after the table describing the command must read as follows:

The lower byte of *wIndex* specifies the target interface number. The returned value gives the 24-bit WUSB channel time value at the Beacon Period Start Time (BPST) of the next superframe.

Erratum Title: Typo in Set WUSB MAS Command.

Background: N/A.

Change: p250, Section 8.3.5.12, the table describing this command must be changed as follows:

bmRequestType	bRequest	wValue	wIndex	wLength	Data
00100001B	SET_WUSB_MAS	Zero	Interface Number	32	WUSB MAS

Erratum Title: Clarification on DN Received Notification returned by an HWA.

Background: The Wireless USB Header of the device notification must not be returned as part of the raw notification data. This is not clear in the specification and this erratum addresses this issue.

Change: p251, Table 8-56, the entry for NotificationSpecific must be changed as follows:

3	*NotificationSpecific*	Variable	Raw Data	The device notification received. The HWA is responsible for decrypting the notification if it was a secure frame. See Section 7-6 for the various notifications that an HWA may receive. Note that the HWA must only send back the *bType* and *NotificationSpecific* data as part of this notification and must not include the Wireless USB header.

Appendix A

Erratum Title: A.1 Incorrect value for CK

Background: The erratum issued in July 2005 to resolve the test vector missing a byte had the wrong value for the CK. This erratum is being issued to address the same.

Change: p.277, Section A.1. The table must be replaced with the following:

Inputs	
Host Address	9876
Device Address	00BE
TKID	019876
Host Nonce	10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F
Device Nonce	20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F
CK	F0 E1 D2 C3 B4 A5 96 87 78 69 5A 4B 3C 2D 1E 0F
Results	
KCK	4B 79 A3 CF E5 53 23 9D D7 C1 6D 1C 2D AB 6D 3F
PTK	C8 70 62 82 B6 7C E9 06 7B C5 25 69 F2 36 61 2D

Association Supplement Section 2

Erratum Title: Add reference to Microsoft Locale ID system

Background: The original published version of the supplement ambiguously referred to “UNICODE language ID”.

Change: p.7, section 2.2: Insert the following line after the “FIPS198A” line:

[LCID] List of Locale ID (LCID) Values as Assigned by Microsoft

Erratum Title: Add references to Microsoft Locale ID system

Background: The original published version of the supplement ambiguously referred to “UNICODE language ID”.

Change: p.11, table 3-1: Change “Description” column for “LangID” to “Locale ID language code (LCID)”

Change: p.16, table 4-7: Change “Data” column for “LangID” to “Locale ID language code (LCID) for the HostFriendlyName field”

Change: p.17, table 4-8: Change “Data” column for “LangID” to “Locale ID language code (LCID) for the DeviceFriendlyName field”

Association Supplement Section 4

Erratum Title: Correct UNICODE encoding to match USB 2.0 standard.

Background: The original published version of the supplement incorrectly specified the UNICODE encoding. The correct encoding is UNICODE UTF-16LE. The original published version of the supplement also used null-terminated strings and specified a length field. The redundant null-termination requirement has been removed.

Change: p.16, table 4-7: Change "Data" for the "HostFriendlyName" entry to "UNICODE string (UNICODE UTF-16LE encoding) used to hold the host-assigned host friendly name. Attribute length must be between 0000H and 0040H."

Change: p.17, table 4-8: Change "Data" for the "DeviceFriendlyName" entry to "UNICODE string (UNICODE UTF-16LE encoding) used to hold the device-assigned device friendly name. Attribute length must be between 0000H and 0040H."

Change: p.15, table 4-6: Change "Data" for "Offset 21: AssociationTypeInfoSize" entry to "00000038H".

Erratum Title: Incorrect description for BandGroups in DEVICE_INFO

Background: N/A

Change: p.17, Table 4-8, offset 28 must be replaced with the following:

28	BandGroups	1004H	0002H	This bitmap reports the UWB band groups that are supported by the device. See [WUSB1] section 7.4.1.

Association Supplement Section 5

Erratum Title: Update section title for section 5.3.2

Background: N/A

Change: p.21, section 5.3.2: Change the heading title to read as follows:

"Host Retrieves Hash Commitment and Friendly Name from Device"

Erratum Title: Add DeviceFriendlyName to Table 5-2

Background: The device now provides its friendly name to the host during the initial phase of the numeric model so that the user can optionally be presented with the information in order to make a decision about whether to accept the association or not.

Change: p.21, replace table 5-2 with the following table:

Offset	Field	Size	Value	Description
0	Version	1	Constant	01H
1	LangId	2	Number	Locale ID language code (LCID) for the DeviceFriendlyName field
3	DeviceFriendlyNameLen	1	Number	Length in bytes of the DeviceFriendlyName field
4	SHA-256(M_3)	32	Number	SHA-256 hash of the M_3 data structure.
36	DeviceFriendlyName	Variable	String	UNICODE string (UNICODE UTF-16LE encoding) used to hold the device-assigned friendly name. Total length of string must be between 0000H and 0040H.

Erratum Title: Update section title for section 5.3.3

Background: N/A

Change: p.21, section 5.3.3: Change the heading title to read as follows:

"Host Sends Its Public Key and Friendly Name to Device"

Erratum Title: Add HostFriendlyName to Table 5-3

Background: The host now provides its friendly name to the device during the initial phases of the numeric model so that the user can optionally be presented with the information in order to make a decision about whether to accept the association or not.

Change: p.22, replace table 5-3 with the following table:

Offset	Field	Size	Value	Description
0	Version	1	Constant	01H
1	LangId	2	Number	Locale ID language code (LCID) for the HostFriendlyName field
3	HostFriendlyNameLen	1	Number	Length in bytes of the HostFriendlyName field
4	PK_H	384	Number	Host's Diffie-Hellman public key
388	HostFriendlyName	Variable	String	UNICODE string (UNICODE UTF-16LE encoding) used to hold the host-assigned friendly name. Total length of string must be between 0000H and 0040H.

Erratum Title: Remove HostFriendlyName from M4 data structure (table 5-6)

Background: The HostFriendlyName has been moved to the M2 data structure.

Change: p.24, section 5.3.8: Remove the last two rows (offsets 74 and 80) from table 5-6. After removing the rows, the last entry in the resulting table is BandGroups.

Erratum Title: Incorrect reference to DHKey

Background: N/A

Change: p.27, section 5.4.5: Replace the text "DHKey from section 5.4.4 and" with the following:

"This section uses PK_D and PK_H from section 5.4.2."