1301 WILSON DRIVE, WEST CHESTER, PA 19380
Tel: (610) 430-2500 Fax: (610) 430-2694

EBDS
&
EBDS Plus

Extended Bi-Directional Serial Protocol Specification with Extended Note Reporting

Table of Contents

Rev	Date	Description of Change
G1	Jan 5, 2003	Initial Version
G2	Nov 17,2003	This is the first external release.
G3	Feb. 2 2004	Added the Developers Guide information, rearranged the Sections and add clarification.
G4	April 26,2004	Moved the Voucher Information to the CASHFLOW users guide. - Initial Release.
G5	June 19,2004 July 2, 2004	1. Add clarification to extended not reporting sections. No functional changes. 2. Added note indicating the host must remain in extended note mode once established. 3. Added $20 to EUR Index Table
G6	Aug. 9 2004 Aug 25, 2004 Sept, 14, 2004	1. Corrected the “Messages Sent By The Acceptor” table. Escrowed was labeled as a Transient State. 2. Added the following note to the extended note reporting section: This step is not required for the SC66. This step is not required for SC83 with Application software revisions V2.20 (Aug. 2004) or higher. 3. Added SC66 Model 0x4A Special Variant request command. Section 6 sub Type 8 4. Added new question to the FAQ section “How can I Determine the Currency Supported?” 5. Correct the AUD note table 6. Corrected and updated Model Reference
G7	Feb 9, 2006	1. Added the definition of Extended Note Reporting to the Host message summary. 2. Added Escrow timeout message. 3. Edited sections on Message Type 7 and Extended Note Processing Method. Removed instructions to zero all data bytes in extended messages. 4. Change bookmark length to 4 inches. 5. Grammar and spelling corrections throughout document. 6. Extended note bit added to table in Section 4.1.
G8	July 7, 2006	1. Added Set Asset Number message (Extended Message 7, Sub Type 5).

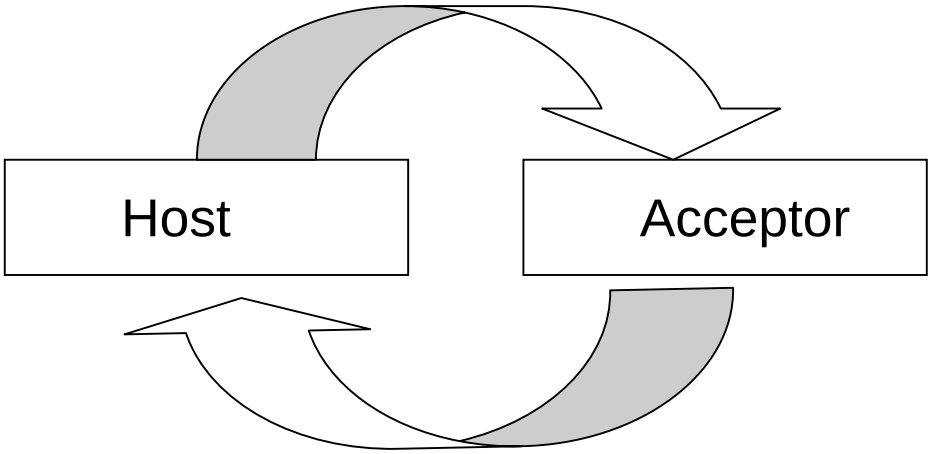
		2. Added Request Value Table (Extended Message 7, Sub Type 6). 3. Clarifications to “Message Type 7” section and “Extended Note Processing Method” section.

1 Overview

The Extended Bi-Directional Serial (EBDS) interface protocol is used to provide a Master (Host) Slave (Acceptor) communication link between an Acceptor and a Host. The Host commands contain information such as when to accept currency, which denominations are to be accepted and when to return a note to the customer. Upon request, the Acceptor communicates its events and states to the Host containing information such as the denominations that are being fed into the Acceptor, when to issue credit for these notes and when it is idling, stacking or returning a note.

The functionality of the Acceptor can be changed in real-time by the Host, while the system remains operational. It is not required to power down the system or manually manipulate dip-switches on the Acceptor to change system configuration settings.

☞ EBDS is a protocol definition. The Host controls the communication session by sending a poll/command to the slave device. The Acceptor will act on the command and send a response message back to the Host for each poll. Each poll/command contains all the possible commands for a specific message type. Likewise, each response for a specific message type has bits representing all the possible states.



1.1 Definitions and Abbreviations

Acceptor	Any Slave device. Acceptor and Slave are used interchangeably throughout this document.
ACK / ACKed	Implies the receiver has followed the ACK requirements.
BA	Bill Acceptor.
Coupon	A bar-coded document with local value only. Not for legal trade. It must be read and validated externally by the network host. Its value is unknown to the BA.
Denomination	Bank Note Value
Document	Either a Bill or a Coupon
Exception	Non planned or expected action of the Acceptor during a particular event sequence or action.
Host	The Master device in an EBDS communication session. It can be a game machine or simply a controller that takes player's documents via the BA. Host and Master are used interchangeably throughout this document.
LRC	Lockable Removable Cassette. This can also refer to the Cash Box, or more simply the cassette attached to the Acceptor for holding processed documents.
Network Host	A remote computer that communicates both ways to the product. It can receive the bar-coded number from a coupon, and lookup its value. The network host may return the coupon's value to the product. It will always return with the instructions to stack or reject the coupon.
Note	Money. A banknote. A document with global value. Legal tender for all debts, public and private, or Barcoded document
Poll	Data string sent by the Host to the slave device.
P O N R	Point of no return. The point of travel where a bill cannot be returned to the user by reversing the transport mechanisms.
Run and Stack	The Acceptor will run the document transport motors inward and perform a staking cycle. This is done to clear the bill path and stacking mechanism in preparation to accept documents. It is a self Check
Session	The time period during which two devices are communicating. A session is considered over when one or both devices are powered down.

2 Overview of EBDS Features

This section provides a high level overview of the features supported by MEI Acceptors running the EBDS protocol. Additional details are provided throughout this specification.

☞ Please reference the "Feature Table" to determine which features are supported by the various MEI models.

2.1 Escrow Mode

When operating in this mode the Acceptor will inform the Host of all uninhibited and validated notes inserted and then wait for the Host to accept or return the notes.

An inserted note will be transported inside the Acceptor until it reaches the "Escrow" position. This is a position where enough information has been received from the note for the Acceptor to make a decision on the validity and denomination of the note. At this point, the entire note is inside the Acceptor and out of the customer's hands, but is still in a position where it can be if needed. With the note in this position a value/data message is reported to the Host, the Acceptor will wait for instructions from the Host to either accept or return the note. If the Host elects to keep the note, the Acceptor completes transportation and stacking of the note and communicates a confirmation message. If the Host elects not to accept the note, the Acceptor returns the note and communicates a rejecting message.

☞ This feature must be used in applications processing Vouchers (barcode coupons).

☞ When this mode is not used (non-escrow mode), all uninhibited and validated notes will automatically be accepted and stacked without waiting for the Host command to accept. The Acceptor will respond with a "stacked" message for each note.

☞ Non-escrow is used in most Drop-Safe type applications were there is no reason to return a note.

2.2 Inhibit Acceptor Feature

This feature allows the Host to inhibit or disable the Acceptor. While the Acceptor is inhibited it will not start the transport motors to draw in a note.

2.3 Barcode Acceptance (Voucher)

A Barcode is a note-sized document with a Barcode printed on the topside. These are used in gaming to replace Coin Pay Out, Promotions and Prizes.

2.4 Bookmark Feature

A bookmark is a piece of paper that is stacked by the Acceptor to mark the location within the cassette (stack of notes) of a special event. For example, a customer may complain that the system has taken a note but credit was not issued. In certain applications (especially those which use Lockable Removable Cassettes) it may be deemed not feasible to immediately gain access to the cassette or magazine to verify a customer complaint. A bookmark can be used at the time of the

event so the customer's complaint can be verified at a later time, when the cassette or magazine is normally removed from the system.

2.5 Special Interrupt Mode

Operating in Special interrupt mode allows the Host to significantly reduce the number of polls sent to the Acceptor. The Acceptor will prompt the Host when a significant event has occurred. For example, when a document is stacked an ENQ is sent to the Host and it will respond with a poll to receive the message.

☞ It is easier to evaluate communications using protocol analyzers in Special Interrupt Mode as this mode generates less message traffic.

2.6 Addressable EBDS (ABDS)

ABDS is an acronym for Addressable Bi-Directional Serial. It is used in applications where an RS-485 multi-drop interface is required. RS-485 allows multiple peripherals to be daisy-chained together, all talking to the same Host, using a single pair of wire. Each Acceptor is configured with a unique address that will be referenced by the Host in its message.

2.7 Extended Information Feature

This feature allows the retrieval of extended information from the Acceptor beyond the standard EBDS status information. A few examples are:

1. Read the Acceptor's software CRC value. The Host can then verify authenticity of the version of software in the Acceptor by comparing it to an acceptable stored value.
2. Reporting the Acceptors supported Bill Set to the Host.
3. Full Note information reporting including currency code, note value, note series, and orientation.
4. Acceptor Performance data (approx. 20 points of data).
5. Inhibit notes by Currency Code, series and value.

☞ Reference Message Type 6 and Message Type 7 Sections for more information on extended information.

2.8 Power Up Features (PUP)

Each Power-Up Process defines the specific rules to follow when power is lost and restored during a note transaction. Power-up policies were introduced for the gaming industries with each policy outlining different Acceptor responses to power loss while processing a note. If a power loss occurs while processing a document, the Host and Acceptor can store information regarding the bill value and bill location. On a successive power up and with the right conditions the Acceptor will either return the document, hold the document in escrow, go out of service, or stack the document.

2.9 Remote Calibration Feature

Most Acceptors support this feature. This feature puts the Acceptor in its "Calibration Mode" were the Acceptors sensors can be calibrated. The calibration process normally requires the insertion of

MEI approved "Calibration Paper". This process is typically done by repair centers after critical parts have been replaced.

2.10 Flash Software Download

The Acceptor's software can be upgraded via the EBDS interface. The Host puts the Acceptor in the download mode and then uses the download protocol to download the new software.

☞ Reference the Flash Download Section.

3 EBDS Feature Operational Considerations

3.1 Acceptor Mode Selection:

Two key mode selections of interface operation are Escrow verses Non Escrow mode and Special interrupt verses polling mode.

3.1.1.1 Escrow Modes

In Escrow Mode all documents that are validated will be held in the escrow position and communicated to the Host. While in the escrow position the Acceptor responds to the controller's poll with its escrow bit set and the value of the document in the bill value field. The controller will then be able to command the Acceptor to stack the document or return it to the customer.

☞ In Non Escrow mode the Acceptor will accept the note (stack) without waiting for the Host's command. In effect, while in Non Escrow mode, the controller will see stacked events from the Acceptor when valid notes are inserted and stacked.

3.1.1.2 Polling Mode

In polling mode the Host send polls out periodically to solicit information from the Acceptor. A polling rate that is to fast will create unnecessarily large amount of transfers between the Host and the Acceptor, while a polling rate that is to slow will unduly delay the reporting of information from the Acceptor. The recommended poll rate is between 200 to 300 ms.

3.1.1.3 Special Interrupt Mode

Operating in Special interrupt mode allows the controller to significantly reduce the number of polls sent to the Acceptor. The Acceptor will prompt the controller, that a significant event has occurred within the Acceptor by sending an ENQ (a single byte, 0x05). The ENQ prompts the Host to poll the Acceptor. The Acceptor will then respond with a new event / state bit change identifying the event or state that has occurred. In some cases the controller will need to respond with additional commands. An example of this is when a document is stacked and an ENQ is sent to the Host; the Host will respond with a poll to receive the message.

☞ It is easier to evaluate communications using protocol analyzers in Special Interrupt Mode as it generates less message traffic

☞ To increase throughput, the ENQ follow up commands should be sent as soon as possible. Once the Host receives a Stacked message it should send a second poll to indicate that the Acceptor's message was received and understood. The Acceptor will assume the message was received after 100ms had elapsed without receiving an ACK or NAK message.

3.2 Inhibiting Denominations:

By inhibiting denominations the Host has control over the note type that the Acceptor will accept.

☞ The Acceptor also evaluates the MEI Configuration Coupon setting and Service Tool setting when deciding if a document is inhibited. All three settings must indicate uninhibited in order to accept a document.

3.3 Inhibit Acceptor Feature

The Host will inhibit all of the notes along with the barcode and bookmark. When inhibited (disabled) the Acceptor will not draw in notes that are presented.

☞ Inhibiting this may be necessary for a Host that wants to prevent the insertion of a document before it is ready to accept it.

☞ Gaming Acceptors will ignore these bits while they are processing notes (from accepting to Stacked). This allows the Host to disable the Acceptor from accepting a second document during the transaction. Otherwise the Acceptor would see all documents inhibited and reject all notes.

3.4 Bookmark Feature

While the acceptance of bookmarks is selected, processing of valid notes will continue as normal. Bookmark acceptance is selected by changing the "Msg Type 3" of Host-to-Acceptor messages. As long as the Msg Type is 3, bookmarks will be accepted. The Host must indicate when this mode is to be terminated by changing to a Msg Type other then 3. It is recommended for the Host to terminate the mode automatically upon receipt of the first bookmark.

Bookmark acceptance is reported to the Host in the same manner as a normal note; except the value field will be set to zero (unknown). If escrow mode is selected, the Host must still indicate whether to accept or return the document.

A bookmark consists of a piece of paper 2.6 inches wide and 4.0 inches in length.

☞ Normally, the Acceptor will not draw notes in with all notes inhibited and Barcodes inhibited. However, this consideration is overridden while Bookmark is selected. This allows only bookmarks to be accepted and reported to the Host if desired (all real notes will be rejected).

! This short length of 4.0 inches was developed with 6-inch note length as a reference. Care should be taken when accepting Multiple Length notes.

3.5 Barcode/Voucher Processing

The processing of barcodes occurs in a similar manner to that of documents. On receipt of a valid barcode, the Acceptor will send a message type 7 escrowed message. The response will include the

barcode data for the Host to validate. The Acceptor will continue to send the escrow message in response to all polls until the Host sends the stack or return message.

3.6 Extended Information Feature

This feature allows the retrieval of extended information from the Acceptor via a message type 6 command. One example of this feature is the Get CRC command that allows the Host to retrieve the CRC value associated with the Acceptor's flash code. The Host can then verify authenticity of the version of software in the Acceptor by comparing it to an acceptable stored value.

Refer to section 6.1 below for more information about retrieving CRC information and other extended information commands.

3.7 Power Up Features

With the addition of the power up policies, which were introduced for the gaming industries, the Acceptor operates differently when addressing power loss situations that occur during a bill transaction. If a power loss occurs while processing a document, the Host and Acceptor can store information regarding the bill value and bill location. On a successive power up and with the right conditions the Acceptor will either return the document, hold the document in escrow, go out of service, or stack the document. The controller will either credit the document or not depending on the Host preference

3.8 Remote Calibration Feature

☞ Most Acceptors support the feature. However, field calibration is not normally required.

Remote calibration is initiated via the Host by setting the message type to 4. Once the Acceptor decodes a message type 4, it will respond by setting its outgoing message type to 4 and immediately or soon after set its Calibration bit (Byte 1 bit 6). After setting its Calibration bit, the Acceptor will wait for the "Calibration Card" or time out and return to Idle if it does not detect the "Calibration Card" within 30 seconds.

When the Host receives a response with the Calibration bit set it can reset the message type to something other than type 4 so that when the Calibration is complete the Acceptor will not reenter the Calibration Mode.

The Calibration Card will be drawn in and then returned. After the Calibration Card has been returned by the Acceptor and removed from the bezel, the Acceptor will resume or continue communications. The Acceptor will indicate an acceptable calibration has been achieved by resetting itself and reporting a "Power Up" event. If calibration fails, the Acceptor remains in Calibration mode until it is manually reset.

☞ Any activity at the front bezel is assumed to be the calibration card while in this mode. When activity is detected at the bezel, the Acceptor will enter Calibration mode and may cease communications. The Acceptor will remain in the Calibration mode until an acceptable calibration is achieved or the Acceptor is manually reset.

☞ Entering Calibration is an EVENT that will generate an ENQ in the Special Interrupt Mode.

3.9 Acceptor State and Event Bits

3.9.1 States vs. Events

Throughout this document, an important distinction is made between "States" and "Events". An Acceptor remains in a certain "state" for varying periods. For example, when idling, the Acceptor continually reports the state in which it currently resides. Generally, only one state can exist at any given time. In contrast, an "event" is an occurrence that is reported to the Host until the Host acknowledges the event. There are two types of events:

1. A one shot event is an event that is transmitted once per occurrence. Example "Stacked".
2. Persistent events are events that will be transmitted continually for some period until the condition is cleared. Example "Cassette Removed".

☞ Whenever a description involves an actual field in a bi-directional serial message, the location of that field within the message is given in parenthesis. This is to facilitate cross-referencing later sections in this document.

3.9.2 Description of Acceptor States

This list contains definitions of the States the Acceptor can pass through while processing a note/barcode document. Generally, only one of the bits in this list will be set at any time in a single message. However, the Acceptor will usually have a particular state bit set across multiple messages.

☞ All states are transient and in some circumstances can be missed by the Host. Detection of transient states must **not** be a critical aspect of the Host implementation. For example, the duration of the Idling State can be very short. Also, some States require an action before they can be exited these are referred to as Persistent States.

3.9.2.1 Idling Bit (Transient)

(Byte 0, Bit 0) - In this state, the Acceptor is not processing a note. The Acceptor is communicating as needed with the Host and monitoring its sensors. This state indicates there are no faults within the Acceptor.

☞ After a document is stacked or returned, the Acceptor will normally return to the idle state and be ready to accept notes unless there was a failure during the stacking or returning process or other exceptions.

3.9.2.2 Accepting Bit (Transient)

(Byte 0, Bit 1) - This State indicates the Acceptor has detected an inserted note and is not inhibited. The Acceptor is attempting to process the note. The Acceptor will continue to report this state until the note reaches the Escrow State and position, it is fully rejected (miss feed of unknown note) or a fault occurs.

! Depending on the Host-polling rate and whether or not the Host is operating in special interrupt mode; the Host may not detect this state.

3.9.2.3 Stacking Bit (Transient)

(Byte 0, Bit 3) - This is a post-escrow state. The Acceptor remains in this state while it is moving a note from the escrow position toward a fully secured position. This state is entered once the Host gives the Stack Command with Escrow Mode selected or automatically after validation in Non-Escrow Mode. The Acceptor will remain in this state until the note is successfully stacked or a jam occurs.

This state bit will also be reported during the initialization cycle (with the Power-Up bit set) if a note is not in the Escrow position or note path.

! The Host may not detect this state depending on the polling rate, whether or not the Host is operating in special interrupt mode or in the event of a power loss.

3.9.2.4 Returning Bit (Transient / Persistent)

(Byte 0, Bit 5) - In this state, the note is being returned to the customer. This state is entered only when the Host issues a Return Command in response to an "Escrowed message".

☞ A distinction is made here between *rejecting* and *returning* a note. A rejected note is returned to the customer based on the Acceptor's validation process. A returned note is one, which passed the Acceptor's checks, but the Host instructed the Acceptor to return the note to the customer.

! The Host may not detect this state depending on the polling rate and whether or not the Host is operating in special interrupt mode.

3.10 Description of One Shot Events

This section contains definitions of the events that occur within the Acceptor and are only reported to the Host once for each occurrence of the event. It is possible for more than one of these bits to be set in the same message. These bits are cleared after a message is received from the Host with a new Msg/Ack # (to be described later) indicating the message containing these bits was received correctly by the Host.

3.10.1.1 Stacked Bit

(Byte 0, Bit 4) - This event is reported after the note has been moved to a secure position within the Acceptor. This is the point at which the Host should issue credit for the inserted note. Even though the Acceptor tells the Host the denomination of a note at the escrow position, the Host should not give credit for the note until the "stacked" message is received. Furthermore, in the case of notes, the Host should only give the amount of credit as indicated in the stacked message. In the case of a barcode, the Host should give the amount of credit based on the data from the extended message previous to receiving the stack message. The note value of the Stacked message for barcodes will be 0. There are some instances where the credit amount told to the Host at escrow will be wiped out in the stacked message (e.g. a CHEAT attempt is detected).

! The Acceptor will assume the Host has acknowledged this event if 100ms have elapsed without a negative acknowledgment and continue to accept notes.

3.10.1.2 Returned Bit

(Byte 0, Bit 6) - This event is reported after a note has been successfully returned to the customer (and removed from the bezel for some models) per the order from the Host following an Escrowed message. The Acceptor had validated the note and the Host commanded the Acceptor to return the note to the customer.

3.10.1.3 Cheated Bit

(Byte 1, Bit 0) - This event is reported if the Acceptor detects an attempt that leads it to believe the note is being manipulated. The customer may or may not be intentionally attempting to fraud the Acceptor. This can result in the note either being returned to the customer or, if the note is too far into the Acceptor to return, being stacked with no credit being given (the note value field will be 0).

3.10.1.4 Rejected Bit

(Byte 1, Bit 1) - This event is reported after the note has been returned to the customer and removed from the bezel because it failed the validation tests or the given denomination is disabled.

3.10.1.5 Power Up Bit

(Byte 2, Bit 0) - This event indicates the Acceptor has experienced a power reset condition. It will continue to be reported until the Acceptor has finished its initialization operation. This event will be reported at least once, even if the initialization is complete, before the first request for status is received.

☞ This bit is a hybrid between a state and an event in that it can be reported more than once but must be reported at least once. During the Acceptor's power up process, the Acceptor will ensure that it has a clear note path by running its Acceptor and stacker motors (unless a note is detected).

3.10.1.6 Invalid Command Bit

(Byte 2, Bit 1) - This event is reported whenever a message containing an unrecognized command code is received from the Host. This condition represents an error in the format of the message rather than in its contents.

3.11 Description Of Persistent States and Events

A persistent event is an event or condition that occurs within the Acceptor that exists for an extended period of time. The State /Event will be reported continuously until it no longer exists.

3.11.1.1 Escrowed Bit (Persistent)

(Byte 0, Bit 2) - Typically, the Acceptor will transport the inserted note to the "Escrow" position. The Host will then chose to either accept or return the note. This bit is set to indicate a note is in the escrow position. The Note Value fields will (Byte 2, Bits 3-5) will be used to indicate the denomination of the note just validated. In case of barcodes, the Acceptor sends a special extended message.

This state will only occur if the Host has selected the Escrow Mode feature by setting the Escrow Mode bit. If the Escrow Mode feature is disabled, a valid document will pass through the Escrow State and proceed directly into the stacking state.

☞ The Acceptor always operates in the Escrow Mode when processing Barcodes. The Acceptor will automatically and temporarily enter the Escrow Mode when it detects a barcode assuming barcodes are enabled. Once the Barcode is processed, the Acceptor will revert to the Host's Escrow setting.

3.11.1.2 Cassette Full Bit (Persistent)

(Byte 1, Bit 3) - In this state, the Acceptor has determined it can not stack any additional documents into the cassette. This indicates the cassette is full and the Acceptor will disable itself.

☞ The Stacked event may occur before or after the State is entered. The Stacked event must be handled separately.

3.11.1.3 Jammed Bit (Persistent)

(Byte 1, Bit 2) - In this state, the Acceptor has determined it cannot successfully finish an operation such as stacking or returning the note. The Acceptor will try to resolve the condition and if successful, will automatically exit this state and indicate the final disposition. If unsuccessful, the Acceptor will remain in this state until serviced.

3.11.1.4 Failure Bit (Persistent)

(Byte 2, Bit 2) - In this state, the Acceptor has discovered a condition, other than those conditions listed above, which does not permit it to continue accepting notes. Typically, the Acceptor has determined, after some attempts at error recovery, it can no longer accept notes.

3.11.1.5 Push / No-Push Bit (Persistent)

(Byte 3, Bit 0) - The Acceptor can operate in either a Push or No Push mode. There are times when the Acceptor has a note, for which it is not going to issue credit, yet the Acceptor cannot return the note to the customer. An example of stacking the note without issuing credit can occur when the Acceptor detects a cheat attempt but cannot return the note.

Some applications require that the Acceptor Stack the unknown notes and remain in service. This is called the Push mode because the note is pushed into the cassette. Other applications require that the Acceptor freeze or go out of service with the note in the elevator as evidence for an on site attendant to verify the customer complaint. This is the No-Push mode.

3.11.1.6 LRC (Lockable Removable Cassette / Cassette) Bit (Persistent)

(Byte 1, bit 4) - This bit is set if the Lockable Removable Cassette is properly attached to the Acceptor. When this bit is clear (indicating the LRC is not attached), the Acceptor will disable itself and not accept any notes regardless of any commands from the Host.

! This bit may be invalid on the first response from the Acceptor following a power up.

3.11.2 Note Value Field

(Byte 2, Bits 3-5) - These three bits are used to encode the value of the note in process within the Acceptor. They are meaningful in combination with:

1. The Escrowed State bit to indicate the note's value at the escrow position.
2. The Stacked event bit to indicate the credit amount.
3. Less important is the Returned event bit to indicate the note value that was returned to the customer. The value of zero indicates unknown denomination.

There are multiple cases where the Acceptor will report the value field set to zero. Below are a few examples:

1. Barcode documents also have a denomination value of zero when stacked.
2. When the Acceptor enters the Accepting State.
3. At power-up if a note is in escrow and PUP B is not selected. After a power reset the Acceptor will report a value of zero if a note is in Escrow (PUP A and PUP C).
4. The Acceptor senses tampering, possibly attempting to cheat the Acceptor. A note value that may have been communicated during escrow may be set to unknown in a subsequent message informing the Host not to issue credit.
5. Bookmarks have a denomination value of zero.
6. If the Cassette is removed and reinserted, the Note Value field may be reset to zero

☞ Under normal circumstance the last value reported would remain until the next note is inserted; however, it should be ignored.

3.11.2.1 Paused Bit (Persistent)

(Byte 1, Bit 5) - This state is entered when the Acceptor detects a second note insertion while it is preparing to stack a note in order to prevent the second from being drawn into the Acceptor. The Acceptor will stop all motion until the second note is removed. In other words, a customer has quickly inserted a second note while the first note is still in motion within the Acceptor. This is known as a "fast feed". In applications where the Host can display messages to the customer, this Pause message can prompt a message to alert the user to remove their note from the Acceptor. The state is exited automatically upon detection of a clear bezel area and processing of the original note being stacked is resumed.

4 Data Fields Formats

All messages sent between the Host and the Acceptor use data fields to store any important information. The following sections define the bits contained in the data fields and how the Host and the Acceptor use them.

☞ There is a maximum of 7 bits (bit 0-bit 6) of information in each byte. The protocol defines the eighth bit (bit 7) as the parity bit (even).

4.1 Messages Sent by the Host

BYTE 0	DESCRIPTION	"1" Indicates	NOTES
BIT 0	Denomination 1 Enable	Denomination 1 uninhibited	1
BIT 1	Denomination 2 Enable	Denomination 2 uninhibited	1
BIT 2	Denomination 3 Enable	Denomination 3 uninhibited	1
BIT 3	Denomination 4 Enable	Denomination 4 uninhibited	1
BIT 4	Denomination 5 Enable	Denomination 5 uninhibited	1
BIT 5	Denomination 6 Enable	Denomination 6 uninhibited	1
BIT 6	Denomination 7 Enable	Denomination 7 uninhibited	1
BYTE 1	**DESCRIPTION**	**"1" Indicates**	**NOTES**
BIT 0	Special Interrupt Mode	Enable the interrupt mode	2
BIT 1	Security	Enable high security	3
BIT 2	Orientation	See Table Below	4
BIT 3			
BIT 4	Escrow Enable	Escrow mode uninhibited	5
BIT 5	Stack	Stack the Note	6
BIT 6	Return	Return the Note	6
BYTE 2	**DESCRIPTION**	**"1" Indicates**	**NOTES**
BIT 0	Push/No Push Mode	In No Push Mode	7
BIT 1	Enable/Disable decoding barcode	Barcode uninhibited	
BIT 2	Enable/Disable PowerUp-B sequence	Uninhibited	8
BIT 3	Enable/Disable PowerUp-C sequence	Uninhibited	8
BIT 4	Extended Note Reporting	Extended Note Reporting enabled	
BIT 5	Reserved – Must be 0	Not Used	
BIT 6	Reserved – Must be 0	Not Used	

Notes:

Note 1 - To inhibit the Acceptor, all of the denomination bits, bookmarks and barcodes must be disabled. While inhibited, the Acceptor will not turn on its motor when a document is presented. If any of the denominations are uninhibited, the Acceptor will draw in all documents presented. After a document is presented, all inhibited denominations will be rejected and all uninhibited denominations, if deemed valid, will be reported to the Host.

☞ See " Denomination Codes" for individual country tables**.**

☞ Also, reference the Configuration Coupon. The configuration can override these settings

☞ When a note is being drawn into the Acceptor (Accepting state) it will ignore note inhibit requests from the Host. This is necessary to perform legacy Host applications that inhibit the Acceptor when it detects the Accepting State. This method is not recommended or required, since the Acceptor can only process one note at a time.

Note 2 - This enables the Special Interrupt Mode. When this bit is set, the Acceptor will send an "ENQ" message whenever there is a state change (e.g. the Acceptor would send a message when it changed from the "Stacking" state to the "Idling" state).

Note 3 - The detailed functionality of the security bit within the Acceptor will change as software revisions are upgraded. It is anticipated some algorithms for fraud detection within the Acceptor may be optionally controlled (bit=1) or inhibited (bit=0) depending on the application. For example, in a highly supervised environment, some anti-fraud algorithms may not be needed and could be disabled in an effort to obtain higher acceptance rates.

Note 4 - Notes will be accepted according to the following table. Barcode acceptance is not selectable and is always set to two-way accept.

Bit 3	Bit 2	Orientation
0	0	One Way Accept - Face up denomination first
0	1	Two Way Accept - Face up both directions
1	N/A	Four Way Accept - All directions

☞ Also, reference the Configuration Coupon. The configuration can override these settings.

Note 5 - Disabling the escrow function (Byte 1, Bit 4 = 0) causes all accepted notes to be stacked without waiting in the escrow position for a STACK command from the Host.

Note 6 - The stack and return bits are used to direct the note/barcode from the escrow position and should only be set when a note is in Escrow. The Acceptor ignores these two bits sent from the Host unless the Acceptor is in the Escrow State. If a note is in Escrow, a message with either bit set will cause the Acceptor to immediately begin the process of stacking or returning. Once a note is stacked or returned. The Host only needs to see the Acceptor's ACK message once and then it can clear it. The Host does not have to send an additional message with the bit set. Therefore, extra "Stack" or "Return" commands from the Host are ignored until the Acceptor is again in the Escrow State.

Note 7 - Setting the Push / No Push message bit to zero will permit the Acceptor to stack (Push) and return to service.

Note 8 - By default this feature is inhibited.

4.2 Messages Sent By The Acceptor

BYTE 0	**Description**	**A "1" Indicates**	NOTES
BIT 0	Idling	Not Processing a Note	5
BIT 1	Accepting	Drawing a note/barcode in	5
BIT 2	Escrowed	Note/barcode now in Escrow	
BIT 3	Stacking	In the Stacking process	5
BIT 4	Stacked	The note/barcode has been Stacked	
BIT 5	Returning	Returning a note/barcode	5
BIT 6	Returned	Note/barcode has been Returned	
BYTE 1	**Description**	**A "1" Indicates**	**NOTES**
BIT 0	Cheated	A Cheat has been detected	
BIT 1	Rejected	A note/barcode was Rejected	
BIT 2	Jammed	The Acceptor is Jammed	
BIT 3	Cassette Full	The Cassette is Full	
BIT 4	LRC Status, 1 = Installed	Lockable Removable Cassette status	
BIT 5	Paused	The Acceptor is paused	
BIT 6	Calibration	Entering Calibration mode	
BYTE 2	**Description**	**A "1" Indicates**	**NOTES**
BIT 0	Power Up	The Acceptor has just been reset	1
BIT 1	Invalid Command	Invalid command was received	
BIT 2	Failure	Requires reset or repair	
BIT 3-5	Note value	See Table below	2
BIT 6	Always 0		
BYTE 3	**Description**	**A "1" Indicates**	**NOTES**
BIT 0	No Push Mode	Acceptor is stalled in the No Push Mode	
BIT 1	Flash Download	Entering Flash Download mode	
BIT 2	Pre-stack	The note is at the point in the stacking process where it can no longer be returned.	
BIT 3-6	Always 0		
BYTE 4	Model #		3
BYTE 5	Revision #		4

NOTE VALUE Table

BIT	5	4	3	NOTE VALUE
	0	0	0	None/Unknown/Bookmark/Barcode
	0	0	1	Denomination 1
	0	1	0	Denomination 2
	0	1	1	Denomination 3
	1	0	0	Denomination 4
	1	0	1	Denomination 5
	1	1	0	Denomination 6
	1	1	1	Denomination 7

Note 1 - The Power-up bit indicates the Acceptor has been reset. The "Power-up" bit will be set (for multiple messages) as long as the Acceptor is in the process of initialization.

☞ Note that this bit may be set along with other bits (e.g. the "Stacking" or "Idling" bits may be set at the same time). It is guaranteed that a reset will cause at least one message to be sent with the "Power-up" bit set. The "Power-up" bit must be clear before the Host can control the operation of the Acceptor.

Note 2 - The "Note Value" field indicates the value of the last note received. It is valid when one of the following are true:

- Note is in Escrow
- Note has just been stacked

☞ The value is reset to zero, unknown note, when the Note Acceptor transmits the accepting message as a result of beginning the next accept cycle.

Note 3 - The Model number is a unique code assigned by MEI at the factory that will identify the Model of Acceptor.

☞ Reference the "Model Number Appendix" for specific details.

Note 4 - The Revision corresponds to the internal software revision. It increments every time the software is updated. When corresponding with MEI technical support, you will need this number to identify the version of the software.

☞ The number is the binary representation of a two-digit revision. For example, software version 2.0 will be transmitted as hexadecimal 0x14. MEI EBDS software will carry a two-digit revision only.

Note 5 – These states are not guaranteed to be reported, thus the Host should not be dependent upon them.

5 General Message Description

Below is a description of the message fields and their meaning. Each data field consists of a single byte.

5.1 Message Format

From the Host

STX (02)	Length (08H)	Msg type and Ack#	Data Byte 0	Data Byte 1	Data Byte 2	ETX (03)	Check Sum

From the Acceptor

STX (02)	Length (0BH)	Msg type and Ack#	Data Byte 0	Data Byte 1	Data Byte 2	Data Byte 3	Data Byte 4	Data Byte 5	ETX (03)	Check Sum

Field Name	Use
STX (0x02)	Indicates the start of a message. Same for all messages.
Length	The number of bytes in the message (in binary), including the STX, ETX, and the Checksum. For ease of future enhancements, this should be treated as a variable and therefore messages treated as variable length messages
Message Type / ACK#	**This byte is broken into two nibbles.** **High nibble = Message Type** Message Type (bits 4, 5, & 6) 0x1n = Standard Host to Acceptor messages 0x2n = Standard Acceptor to Host messages 0x3n = Bookmark selected 0x4n = Calibrate Mode 0x5n = Flash Download 0x6n = Request CRC, Get Cash in Box, Soft Reset … 0x7n = Extended message set **Low nibble = The ACK Number 1 or 0**
Data Fields	The data portion of the message consists of the multiple data fields as described in Format for Data Field section.
EXT (0x03)	Indicates end of information. Same for all messages.
Checksum	The checksum is calculated on all bytes between the STX and the ETX (excluding the STX and the ETX). The calculation is performed by XOR'ing the bytes

5.2 ACK # Considerations

In messages sent by the Host, the ACK# is used to identify the message. Normally, as messages are sent to the Acceptor, the ACK number alternates between 0 and 1. The Host can NAK the Acceptor by not alternating the number. When the Acceptor receives two consecutive messages with the same number, the second message is treated as the re-transmission of the first.

In messages sent by the Acceptor to the Host, the ACK# is used to acknowledge message sent by the Host. When the Acceptor receives a Host's message correctly, the Acceptor's ACK number response is set equal the ACK# of the Host's message. If the Acceptor receives a message incorrectly, it will respond with the previous ACK# (different) or not respond at all.

☞ Host Message Type Change - If a message from the Host is sent with a new Message Type, the Acceptor will ignore the ACK bit and assume the last response was received by the Host and transmit a new response. This has been implemented to address legacy Host compatibility.

5.2.1 ACK# Example (Normal Operation)

The ACK# alternates for each message (re-transmission request do not alter the number). When the Acceptor receives a message successfully, it responds with a message containing an Ack # equal to the Ack # of the received in the Host's message. The byte representing the message type and Ack # during normal transactions appear as follows:

Msg Type & ACK # From HOST	MESSAGE DIRECTION	Msg Type & ACK # From ACCEPTOR	ACTIVITY
1**0**	→		Host Poll
	←	2**0**	Acceptor Responds
1**1**	→		Host Poll
	←	2**1**	Acceptor Responds
1**0**	→		Host Poll

5.2.2 ACK# Example (No Acceptor Response)

If the Acceptor receives a message unsuccessfully (the correct number of bytes is received but an error is detected through parity or the checksum), the Acceptor will respond with the previous Ack #. This should prompt the Host to re-transmit its last message with its "Ack #" unmodified.

If the Acceptor receives a partial message, it does not respond. After the Host times out (time > "Max Line Turn Around Time"), the message is re-transmitted.

Msg Type & ACK # From HOST	MESSAGE DIRECTION	Msg Type & ACK # From ACCEPTOR	ACTIVITY
1**0**	→		Host Poll
	←	2**0**	Acceptor Responds
1**1**	→		Acceptor did not receive the message. It does not respond.
		No Acceptor Response.	
1**1**	→		The Host times out and re-sends the message because it did not receive a response within 35ms.
	←	2**1**	Acceptor Responds

5.3 Message Transfer

The EBDS protocol can be operated in one of two modes: Standard Polling mode or Special Interrupt Mode. In standard polling mode, the Host maintains constant communication with the Acceptor at some constant polling rate. For example, the Host may poll the Acceptor once every 200 milliseconds. In special interrupt mode, there is much less communication between the Acceptor and the Host while the Acceptor is not busy. In special interrupt mode, most communication occurs while something "important" is occurring at the Acceptor (e.g. a note is being fed into the unit).

In polling mode, the Host is the master. The Acceptor is responding to messages initiated by the Host. In special interrupt mode, the Acceptor sends an ENQ to the Host to get the Host to send a polling message. For each message sent by the Host, the Acceptor responds with a message (the Acceptor's message contains both data information and a protocol link response).

5.3.1 Polling Interface Example

(Approximately 300ms poll rate)

	HOST'S POLL								ACCEPTOR RESPONSE											
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**ACTION**
1	02	08	10	1F	14	00	03	13	02	0B	20	**01**	10	**20**	00	01	0A	03	11	Acceptor is IDLING
2	02	08	11	1F	14	00	03	12	02	0B	21	**01**	10	20	00	01	0A	03	10	Acceptor is IDLING
3	02	08	10	1F	14	00	03	13	02	0B	20	**01**	10	20	00	01	0A	03	11	Acceptor is IDLING
4	02	08	11	1F	14	00	03	12	02	0B	21	**01**	10	20	00	01	0A	03	10	Acceptor is IDLING
5	02	08	10	1F	14	00	03	13	02	0B	20	**01**	10	20	00	01	0A	03	11	Acceptor is IDLING
6	02	08	11	1F	14	00	03	12	02	0B	21	**01**	10	20	00	01	0A	03	10	Acceptor is IDLING
7	02	08	10	1F	14	00	03	13	--	--	--	--	--	--	--	--	--	--	--	No Response (drawing in)
8	02	08	10	1F	14	00	03	13	--	--	--	--	--	--	--	--	--	--	--	No Response (drawing in)
9	02	08	10	1F	14	00	03	13	--	--	--	--	--	--	--	--	--	--	--	No Response (drawing in)
10	02	08	10	1F	14	00	03	13	--	--	--	--	--	--	--	--	--	--	--	No Response (drawing in)
11	02	08	10	1F	14	00	03	13	--	--	--	--	--	--	--	--	--	--	--	No Response (drawing in)
12	02	08	10	1F	14	00	03	13	02	0B	20	**02**	10	**00**	00	01	0A	03	32	ACCEPTING (drawing in)
13	02	08	11	1F	14	00	03	12	--	--	--	--	--	--	--	--	--	--	--	No Response (validating)
14	02	08	11	1F	14	00	03	12	--	--	--	--	--	--	--	--	--	--	--	No Response (validating)
15	02	08	11	1F	14	00	03	12	--	--	--	--	--	--	--	--	--	--	--	No Response (validating)
16	02	08	11	1F	14	00	03	12	--	--	--	--	--	--	--	--	--	--	--	No Response (validating)
17	02	08	11	1F	14	00	03	12	--	--	--	--	--	--	--	--	--	--	--	No Response (validating)
18	02	08	11	1F	14	00	03	12	02	0B	21	**04**	10	**18**	00	01	0A	03	2D	ESCROWED $5
19	02	08	10	1F	**20**	00	03	33	02	0B	20	04	10	18	00	01	0A	03	2C	Host requests STACK
20	02	08	11	1F	14	00	03	12	02	0B	21	**08**	10	18	00	01	0A	03	21	Acceptor is STACKING
21	02	08	10	1F	14	00	03	13	--	--	--	--	--	--	--	--	--	--	--	No Response
22	02	08	10	1F	14	00	03	13	02	0B	20	**08**	10	18	00	01	0A	03	20	Acceptor is STACKING
23	02	08	11	1F	14	00	03	12	02	0B	21	**11**	10	18	00	01	0A	03	38	Acceptor has STACKED
24	02	08	10	1F	14	00	03	13	02	0B	20	**01**	10	18	00	01	0A	03	29	Acceptor is IDLING
25	02	08	11	1F	14	00	03	12	02	0B	21	**01**	10	18	00	01	0A	03	28	Acceptor is IDLING
26	02	08	10	1F	14	00	03	13	02	0B	20	**01**	10	18	00	01	0A	03	29	Acceptor is IDLING
27	02	08	11	1F	14	00	03	12	02	0B	21	01	**00**	18	00	01	0A	03	38	IDLING-LRC Removed

5.3.1.1 Explanation Of Polling Interface Example

The Host is polling at about 0.3 second intervals. The faster the control system polls, the more times it will receive a given state (i.e. accepting, stacking...). A poll rate of about once per 0.1 second is recommended. The poll rate of once per 0.3 seconds was selected here to limit the number of messages. In the example the Host has:

- Uninhibited the Polling Interface.
- Uninhibited the Escrow Mode.
- Uninhibited five denominations.
- Uninhibited two-way orientation.

Frame 1-6: The Acceptor is waiting for a note to be inserted. Note that the Value field is still set to the previous $10 that was processed.

Frame 7-11: The Acceptor is drawing a note in (Accepting) and has momentarily stopped communicating.

Frame 12: The Acceptor has reported it is accepting (drawing a note in). Note that the Value field has been reset to zero.

Frame 13-17: The Acceptor is validating the note and has momentarily stopped communicating.

Frame 18: The Acceptor has reported it has a $5 note in Escrow.
Note that the value field has been set to $5.

Frame 19: The Host has instructed the Acceptor to STACK the $5 note.

Frame 20: The Acceptor has reported it is in the Stacking process.

Frame 21: The Acceptor is still in the stacking process and has momentarily stopped communicating.

Frame 22: The Acceptor is still in the stacking process.

Frame 23: The Acceptor has stacked the note (pushed into the cassette).

Frame 24-26: The Acceptor is waiting for a note to be inserted. Note the value field is still set to the previous $5 that was processed.

Frame 27: The Acceptor has reported the Lockable Removable Cassette has been removed. The Acceptor will not accept a note until the LRC is re-installed.

5.3.2 Special Interrupt Mode

(Byte 1, Bit 0) By setting this bit, the Host can establish Special Interrupt Mode. In this mode the Acceptor will send an "ENQ" (05H) message whenever a significant Event or State change occurs. When the Host receives the "ENQ", should send a normal poll sequence.

☞ If the Acceptor does not receive a poll within 100ms after sending an "ENQ" it will send another "ENQ". (Multiple "ENQ's" should not cause problems within the Host).

When using the interrupt mode, the Host should still send un-requested polling messages every 10 to 60 seconds. This should be done as a precautionary measure to recover from Acceptor power fail resets, which will put the Acceptor into the default-polling mode. When the Acceptor is in the default-polling mode it will not send an "ENQ".

5.3.2.1 ENQ Event / State Table

The following table defines STATES or EVENTS that indicate there has been an occurrence Once the Acceptor is assured the Host has received these bits correctly (Host ACK), the Acceptor resets the bits. As an example, for each note that is rejected, the Host will only receive one message with the "Note Rejected" bit set.

EVENTS THAT CAUSE ENQ TO BE SENT FROM THE ACCEPTOR

"STATE"	STATE ENTERED (bit is set)	STATE EXITED (bit is cleared)
Escrowed	ENQ Sent	
Stacked	ENQ Sent	
Returned	ENQ Sent	
Cheated	ENQ Sent	
Rejected	ENQ Sent	
Jammed	ENQ Sent	ENQ Sent
Cassette Full	ENQ Sent	ENQ Sent
LRC Pres.	ENQ Sent	ENQ Sent
Failure	ENQ Sent	ENQ Sent
Calibration	ENQ Sent	

5.3.3 Special Interrupt Mode (Normal Example)

(Approximately 10 second time out)

	HOST'S POLL								ACCEPTOR RESPONSE											
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**ACTION**
1	02	08	10	1F	14	00	03	13	02	0B	20	01	10	**20**	00	01	0A	03	11	
2	02	08	11	1F	**15**	00	03	13	02	0B	21	01	10	20	00	01	0A	03	10	Host selects the INTERRUPT MODE
3									ENQ SENT											Acceptor sends an ENQ
4	02	08	10	1F	15	00	03	12	02	0B	20	**04**	10	**18**	00	01	0A	03	2C	ESCROWED $5
5	02	08	11	1F	**35**	00	03	33	02	0B	21	04	10	18	00	01	0A	03	2D	Host requests STACK
6									ENQ SENT											Acceptor sends an ENQ
7	02	08	10	1F	15	00	03	12	02	0B	20	**11**	10	18	00	01	0A	03	39	Acceptor has STACKED
8									ENQ SENT											Acceptor sends an ENQ
9	02	08	11	1F	15	00	03	13	02	0B	21	01	**00**	18	00	01	0A	03	38	LRC Removed
10									ENQ SENT											Acceptor sends an ENQ
11	02	08	10	1F	15	00	03	12	02	0B	20	01	**10**	18	00	01	0A	03	29	LRC Attached
12	10 seconds elapse.																			No activity
13																				No activity
14																				No activity
15																				No activity
16	Host sends an unrequested POLL																			No activity
17	02	08	11	1F	15	00	03	03	02	0B	21	**01**	10	18	00	01	0A	03	28	Acceptor is IDLING

5.3.4 Explanation Of Special Interrupt Mode Example

Above is an example of the Special Interrupt Interface Mode message sequences. In general the Host only sends a Polling message when it receives an ENQ. In the example the Host has:

- Set the Special Interrupt Mode
- Set the Escrow Mode.
- Uninhibited five denominations.
- Enabled both orientations.

Frame 1-2: The Host has sent two messages to establish the above settings. The Acceptor is waiting for a note to be inserted. Note the Value field is still set to the previous $10 (20h) that was processed before this example.

Frame 3: The Acceptor has sent an interrupt (ENQ) to the Host and is awaiting a Poll from the Host

Frame 4: The Host has responded with a Poll. In the Acceptor's response message the ESCROWED bit and $5 value are communicated.

Frame 5: The Host has instructed the Acceptor to STACK the $5 note.

Frame 6: The Acceptor has sent an interrupt (ENQ) to the Host and is awaiting a Poll from the Host

Frame 7: The Acceptor has stacked the note (pushed it into the LRC). Time has passed and the LRC has been removed.

Frame 8: The Acceptor has sent an interrupt (ENQ) to the Host and is awaiting a Poll from the Host

Frame 9: The Host has responded with a Poll. In the Acceptor's response message the LRC status is cleared indicating the LRC has been removed.

Frame 10: The Acceptor has sent an interrupt (ENQ) to the Host and is awaiting a Poll from the Host.

Frame 11: The Host has responded with a Poll. In the Acceptor's response message the LRC status is set indicating the LRC has been re-installed.

Frame 12-17: The Host has not received a message within 10 seconds and has sent an Un-requested poll to insure the Acceptor is still set as expected. The Host time out interval is user defined.

6 Message Type 6 (Information Query)

Message Type 6 is used to request information from the Acceptor and to send special commands to the Acceptor. This is achieved by setting the Message Type to 6 and setting the data bytes to the specific values listed in this section. The data byte values accumulate to a unique value referred to as a message subtype. Therefore, the Information Query type message can support up to 2^{21} -1 different subcommands. Each Acceptor model may support a different subset of these commands.

Table - Message response summary

Query Command	Sub-type	Acceptor Response	Section
Request Software CRC	00-00-00	Acceptor's Software CRC	6.1
Request Cash Value In Cassette	00-00-01	The total value of all the notes in the cash box.	6.2
Request Number of Acceptor Resets	00-00-02	Total number of Power up/resets the Acceptor has performed	6.3
Clear Cash Value in Cassette	00-00-03	Acceptor clears its note value counter to zero.	1.1
Request Acceptor Type	00-00-04	The Acceptor responds with a non-zero terminated ASCII string.	6.5
Request Acceptor Serial Number	00-00-05	The Acceptor responds with a non-zero terminated ASCII string.	6.6
Request Acceptor Boot Software Version	00-00-06	The Acceptor responds with a non-zero terminated ASCII string.	6.7
Request Acceptor Application Software Version	00-00-07	The Acceptor responds with a non-zero terminated ASCII string.	1.1
Request Acceptor Variant Name	00-00-08 (01-00-08)	The Acceptor responds with a non-zero terminated ASCII string. See special note	6.9
Request Acceptor Variant Version	00-00-09	The Acceptor responds with a non-zero terminated ASCII string.	
Request Acceptor Audit Life Time Totals	00-00-0A	The Acceptor responds with an array of 32 bit integer values.	6.11
Request Acceptor Audit QP Measures	00-00-0B	The Acceptor responds with an array of 16 bit integer values.	1.1
Request Acceptor Audit General Performance Measures*	00-00-0C	The Acceptor responds with an array of 16 bit integer values.	6.13
Request Soft Reset	7F-7F-7F	Acceptor will reset and perform power up initialization.	6.14

Message format from the Host (command)

Field	STX	Length	Msg Type & ACK	Subtype MSB	Subtype	Subtype LSB	ETX	Check Sum
Value in Hex	02	XX	6n	Refer to Section Detail			03	XX

X = Determined by message, **n =** ACK "n" is determined by current ACK / NAK state

6.1 Request Software CRC

This is a message Type 6,

- **Subtype 0 message**.

Retrieval of the embedded software's (flash code) 16-bit CRC with zero seed is accomplished via message type 6, with the 3 data bytes all set to 0.

The message formats for the CRC Message are described below.

Message format from the Host (command)

Field	STX	Length	Msg Type & ACK	Subtype MSB	Subtype	Subtype LSB	ETX	Check Sum
Value in Hex	02	08	6n	00	00	00	03	XX

Message format from the Acceptor (response)

Field	STX	Length	Msg Type & ACK	Data Byte 0	Data Byte1	Data Byte2	Data Byte3	Data Byte4	Data Byte5	ETX	Check Sum
Value in Hex	02	0B	6n	0X	0X	0X	0X	00	00	03	XX

X = The Actual CRC Value (i.e. 0x1, 0x2, 0x3, 0x4 = CRC of 0x1234)

Table of resultant bit values (Values for X within message data bytes)

Data Byte	Bit 3	Bit 2	Bit 1	Bit 0
Byte 0	2^{15}	2^{14}	2^{13}	2^{12}
Byte 1	2^{11}	2^{10}	2^{9}	2^{8}
Byte 2	2^{7}	2^{6}	2^{5}	2^{4}
Byte 3	2^{3}	2^{2}	2^{1}	2^{0}
Byte 4	0	0	0	0
Byte 5	0	0	0	0

6.2 Request Cash Value In Cassette

This is a message type 6

- **Subtype 1 message**.

The Host can query the Acceptor for the total note value in its cassette since the value was last reset. This feature can be used as an aid in reconciling the physical note count.

In the event that a cash box is removed while power is applied to the Acceptor, the “Cash in Box” counter will be automatically cleared after the command is received and the value is sent to the Host.

Message format from the Host (command)

Field	STX	Length	Msg Type & ACK	Subtype MSB	Subtype	Subtype LSB	ETX	Check Sum
Value in Hex	02	08	6n	00	00	01	03	XX

X - Determined by message

Field	STX	Length	Msg Type & ACK	Data Byte 0	Data Byte1	Data Byte2	Data Byte3	Data Byte4	Data Byte5	ETX	Check Sum
Value in Hex	02	0B	6n	0X	0X	0X	0X	0X	0X	03	XX

X = The total value in the Cash Box

The resultant Cash Value in Cassette is a 24-bit number reported as listed below. The value is reported in the lower 4 bits of each message data byte as described in the following table. Data byte 0 contains the resultant MSB(bits) down to data byte 0 which contains the resultant LSB(bits). There are 6 bytes (6 resultant nibbles) transmitted.

Table of resultant bit values (Values for X within message data bytes)

Data Byte	Bit 3	Bit 2	Bit 1	Bit 0
Byte 0	2^{23}	2^{22}	2^{21}	2^{20}
Byte 1	2^{19}	2^{18}	2^{17}	2^{16}
Byte 2	2^{15}	2^{14}	2^{13}	2^{12}
Byte 3	2^{11}	2^{10}	2^{9}	2^{8}
Byte 4	2^{7}	2^{6}	2^{5}	2^{4}
Byte 5	2^{3}	2^{2}	2^{1}	2^{0}

6.3 Request Number of Acceptor Resets

This is a message type 6

- **Subtype 2 message.**

The Host can request the number of times the Note Acceptor has been through the reset process. This includes both soft and hard (i.e. power. cycles) resets. This feature can be used to help diagnose exception faults.

Message format from the Host (command)

Field	STX	Length	Msg Type & ACK	Subtype MSB	Subtype	Subtype LSB	ETX	Check Sum
Value in Hex	02	08	6n	00	00	02	03	XX

X - Determined by message

Field	STX	Length	Msg type 3 Bits	Data Byte 0	Data Byte1	Data Byte2	Data Byte3	Data Byte4	Data Byte5	ETX (03)	Check Sum
Value in Hex	02	0B	6n	0X	0X	0X	0X	0X	0X	03	XX

X = The Number of Resets

Where: The resultant Number of Acceptor Resets is a 24-bit number reported as listed below. The value is reported in the lower 4 bits of each message data byte as described in the following table. Data byte 0 contains the resultant MSB (bits) down to data byte 0 which contains the resultant LSB (bits). There are 6 bytes (6 resultant nibbles) transmitted.

Table of resultant bit values (Values for X within message data bytes)

Data Byte	Bit 3	Bit 2	Bit 1	Bit 0
Byte 0	2^{23}	2^{22}	2^{21}	2^{20}
Byte 1	2^{19}	2^{18}	2^{17}	2^{16}
Byte 2	2^{15}	2^{14}	2^{13}	2^{12}
Byte 3	2^{11}	2^{10}	2^{9}	2^{8}
Byte 4	2^{7}	2^{6}	2^{5}	2^{4}
Byte 5	2^{3}	2^{2}	2^{1}	2^{0}

6.4 Request Clear Cash In Cassette Value

This is a message type 6

- **Subtype 3 message**.

The Host can reset the Cash Value in the Cassette to zero. This feature can be used in the event the cassette is removed while power is removed. To correctly maintain the cash in cassette count it is the controller's responsibility to monitor and track events. For instance, an external button press by an operator could signal the cassette has been emptied.

☞ The Acceptor will respond to this message with a value of 0 (the count after counter has been cleared). The controller should issue Request Cash in Cassette command to obtain the most recent count prior to issuing this command.

Message format from the Host (command)

Field	STX	Length	Msg Type & ACK	Subtype MSB	Subtype	Subtype LSB	ETX	Check Sum
Value in Hex	02	08	6n	00	00	03	03	XX

X - Determined by message

Field	STX	Length	Msg type 3 Bits	Data Byte 0	Data Byte1	Data Byte2	Data Byte3	Data Byte4	Data Byte5	ETX	Check Sum
Value in Hex	02	0B	6n	00	00	00	00	00	00	03	XX

There is no data value to report so all data bytes 0 through 5 are set to 0.

6.5 Request Acceptor Type

This is a message type 6,

- **Subtype 4 message**.

The Host can request the Acceptor type and the response will be a 7-bit ASCII string with the leading character in the data byte 0 position. The terminating character is any non-printable character.

Message format from the Host (command)

Field	STX	Length	Msg Type & ACK	Subtype MSB	Subtype	Subtype LSB	ETX	Check Sum
Value in Hex	02	08	6n	00	00	04	03	XX

X - Determined by message

Field	STX	Length	Msg type 3 Bits	Data Byte 0	Data n . . .	Data Byte 19	ETX	Check Sum
Value in Hex	02	19	6n	XX		XX	03	XX

X = Acceptor Type (Contact MEI for Current list of Acceptor types).

6.6 Request Acceptor Serial Number

This is a message type 6,

- **Subtype 5 message**.

The Host can request the Acceptor Serial Number written into the Acceptor's memory. The response is a 7-bit ASCII string with the leading character in the data byte 0 position. The terminating character is any non-printable character.

Field	STX	Length	Msg Type & ACK	Subtype MSB	Subtype	Subtype LSB	ETX	Check Sum
Value in Hex	02	08	6n	00	00	05	03	XX

X - Determined by message

Field	STX	Length	Msg type 3 Bits	Data Byte 0	Data n . . .	Data Byte 19	ETX	Check Sum
Value in Hex	02	19	6n	XX		XX	03	XX

X = The serial Number. Below is are the fields of the MEI part number

The Serial number is fixed at 11 Chars

WW Y L CC ####

- **WW** = Week of Manufacture (W1 – W54)
- **Y** = Year of Manufacture (Y0 – Y9)
- **L** = Location of Manufacturing
- **CC** = Configuration Code (Build Standard)
- #### = Sequenced MFG number.

6.7 Request Acceptor Boot Software Version

This is a message type 6,

- **Subtype 6 message**.

The Host can request the Acceptor 9 character Boot Software Version written into the Acceptor's memory. The response is a 7-bit ASCII string with the leading character in the data byte 0 position. The response string fills all available data bytes 0 through 8.

Message format from the Host (command)

Field	STX	Length	Msg Type & ACK	Subtype MSB	Subtype	Subtype LSB	ETX	Check Sum
Value in Hex	02	08	6n	00	00	06	03	XX

X - Determined by message

Field	STX	Length	Msg type 3 Bits	Data Byte 0	Data n . . .	Data Byte 8	ETX	Check Sum
Value in Hex	02	0E	6n	XX		XX	03	XX

X = Boot Loader's Software Part Number

MEI Software Part Numbers a fixed at 9 characters.

PPPPP C VVV

- **PPPPP** is the project number
- **C** is the part number Check Digit (Not used external to MEI)
- **VVV** is the revision or Version

Example 28079 8 106

- **28079** is a MEI SC66 Boot Fixed Software Project Number
- **8** is the Check Digit
- **106** is Version 1.06

6.8 Request Acceptor Application Software Version

This is a message type 6

- **Subtype 7 message.**

The Host can request the Acceptor 9 character Application Software Version written into the Acceptor's memory. The response is a 7-bit ASCII string with the leading character in the data byte 0 position. The response string fills all available data bytes 0 through 8.

Message format from the Host (command)

Field	STX	Length	Msg Type & ACK	Subtype MSB	Subtype	Subtype LSB	ETX	Check Sum
Value in Hex	02	08	6n	00	00	07	03	XX

X - Determined by message

Message format from the Acceptor (response)

Field	STX	Length	Msg type 3 Bits	Ack # 4 Bits	Data Byte 0	Data n . . .	Data Byte 8	ETX	Check Sum
Value in Hex	02	0E	6n	X	XX		XX	03	XX

X is the Acceptor's Application Software Version

MEI Software Part Numbers a fixed at 9 characters.

PPPPP C VVV

- **PPPPP** is the project number
- **C** is the part number Check Digit (Not used external to MEI)
- **VVV** is the revision or Version

Example 28142 5 310

- **28142** is the MEI EBDS USD SC66 Fixed Software Project Number
- **5** is the Check Digit
- **310** is Version 3.10

6.9 Request Acceptor Variant Name

This is a message type 6,

- **Subtype 8 message *(Important See SC66 Note below)***

The Host can request the Acceptor Variant Name written into the Acceptor's memory. The response is a 7-bit ASCII character string with the leading character in the data byte 0 position. The response string fills all available data bytes 0 through 31. The terminating character is the first non-printable character within the data.

Message format from the Host (command)

Field	STX	Length	Msg Type & ACK	Subtype MSB	Subtype	Subtype LSB	ETX	Check Sum
Value in Hex	02	08	6n	00	00	08	03	XX

X - Determined by message

Message format from the Acceptor (response)

Field	STX	Length	Msg type 3 Bits	Data Byte 0	Data n . . .	Data Byte 31	ETX	Check Sum
Value in Hex	02	25	6n	XX		XX	03	XX

X Is the Variant Name, generally these are Currency Code(s) such as CAD for Canada.

☞ The Acceptor can and does support mix currency in one variant. In these cases each supported currency will be reported in the same string.

☞ Important Note - SC66 Model type 00X4A supports a special version of this command beginning with versions:

AUD Variant V1.70 (MEI # 281613179)

USD Variant V3.50 (MEI # 281420350).

Future SC66 Model type 00X4A AUD and USD variant releases will support this command.

For SC66 Model type 00X4A AUD and USD set Subtype MSB to a "01"

Special Message format from the Host (command) Special

STX	Length	Msg Type & ACK	Subtype MSB	Subtype	Subtype LSB	ETX	Check Sum
02	08	6n	**01**	00	**08**	03	XX

6.10 Request Acceptor Variant Version

This is a message type 6

- **Subtype 9 message.**

The Host can request the Acceptor 9 character Application Software Version written into the Acceptor's memory. The response is a 7-bit ASCII character string with the leading character in the data byte 0 position. The response string fills all available data bytes 0 through 8.

Message format from the Host (command)

Field	STX	Length	Msg Type & ACK	Subtype MSB	Subtype	Subtype LSB	ETX	Check Sum
Value in Hex	02	08	6n	00	00	09	03	XX

X - Determined by message

Message format from the Acceptor (response)

Field	STX	Length	Msg type 3 Bits	Data Byte 0	Data n . . .	Data Byte 8	ETX	Check Sum
Value in Hex	02	0E	6n	XX		XX	03	XX

X = Boot Loader's Software Part Number

MEI Software Part Numbers a fixed at 9 characters.

PPPPP C VVV

- **PPPPP** is the project number
- **C** is the part number Check Digit (Not used external to MEI)
- **VVV** is the revision or Version

Example 64001 3 100

- **64001** is a Variant Part Number
- **3** is the Check Digit
- **100** is Version 1.01

6.11 Request Acceptor Audit Life Time Totals

This is a message type 6

- **Subtype 10 message.**

The Host can request the Acceptor Audit Lifetime Totals written into the Acceptor's memory. The response is an array of 6 - 32 bit unsigned integers. Each integer is transmitted as the lower nibble within 8 data bytes. The following 6 fields are transmitted:

Performance Data Map ID - Field 0;

Total Operating Hours - Field 1;

Total Motor Starts - Field 2;

Total Documents Reached Escrow Position - Field 3 ;

Total Documents Passed Recognition - Field 4;

Total Documents Passed Validation Field 5;

Message format from the Host (command)

Field	STX	Length	Msg Type & ACK	Subtype MSB	Subtype	Subtype LSB	ETX	Check Sum
Value in Hex	02	08	6n	00	00	0A	03	XX

X - Determined by message

Message format from the Acceptor (response)

Field	STX	Length	Msg type 3 Bits	Data Field 0 8 bytes	Field n . . .	Data Field 5 8 bytes	ETX	Check Sum
Value in Hex	02	35	6n	XX		XX	03	XX

X is the Audit Data

Table of resultant bit values (Values for X within message field data bytes)

Data Field	Bit 3	Bit 2	Bit 1	Bit 0
Field Byte 0	2^{31}	2^{30}	2^{29}	2^{28}
Field Byte 1	2^{27}	2^{26}	2^{25}	2^{24}
Field Byte 2	2^{23}	2^{22}	2^{21}	2^{20}
Field Byte 3	2^{19}	2^{18}	2^{17}	2^{16}
Field Byte 4	2^{15}	2^{14}	2^{13}	2^{12}
Field Byte 5	2^{11}	2^{10}	2^{9}	2^{8}
Field Byte 6	2^{7}	2^{6}	2^{5}	2^{4}
Field Byte 7	2^{3}	2^{2}	2^{1}	2^{0}

6.12 Request Acceptor Audit QP Measures

This is a message type 6

- **Subtype 11 message.**

The Host can request the Acceptor Audit QP (Quotient of Performance) Measures written into the Acceptor's memory. The response is an array of 14 - 16 bit unsigned integers. Each integer is transmitted as the lower nibble within 4 data bytes. The following 14 fields are transmitted:

Last 100 Bills Accept Rate - Field 0;

Total Motor Starts - Field 1;

Total Documents Stacked - Field 2;

Total Documents Reached Escrow Position - Field 3;

Total Documents Passed Recognition - Field 4;

Total Documents Passed Validation - Field 5;

Total Recognition Rejections - Field 6;

Total Security Rejections - Field 7;

Total Orientation Disabled Rejections - Field 8;

Total Document Disabled Rejections - Field 9;

Total Fast Feed Rejection Rejections - Field 10;

Total Documents Inserted while Disabled - Field 11;

Total Host Return Document Rejections - Field 12;

Total Barcodes Decoded - Field 13;

Message format from the Host (command)

Field	STX	Length	Msg type 3 Bits	Msg Type & ACK	Subtype MSB	Subtype	ETX	Check Sum
Value in Hex	02	08	6n	00	00	0B	03	XX

X - Determined by message

Message format from the Acceptor (response)

Field	STX	Length	Msg type 3 Bits	Data Field 0 4 bytes	Data n . . .	Data Field 13 4 bytes	ETX	Check Sum
Value in Hex	02	3D	6n	XX		XX	03	XX

X - Determined by message

Table of resultant bit values (Values for X within message field data

bytes)

Data Field	Bit 3	Bit 2	Bit 1	Bit 0
Field Byte 0	2^{15}	2^{14}	2^{13}	2^{12}
Field Byte 1	2^{11}	2^{10}	2^{9}	2^{8}
Field Byte 2	2^{7}	2^{6}	2^{5}	2^{4}
Field Byte 3	2^{3}	2^{2}	2^{1}	2^{0}

6.13 Request Acceptor Audit General Performance Measures

This is a message type 6

- **Subtype 12 message.**

The Host can request the Acceptor Audit General Performance Measures written into the Acceptor's memory. The response is an array of 17 - 16 bit unsigned integers. Each integer is transmitted as the lower nibble within 4 data bytes. The following 17 fields are transmitted:

Total Cross Channel 0 Rejects - Field 0;

Total Cross Channel 1 Rejects - Field 1;

Total Sum of All Jams - Field 2;

Total Jam Recovery Efforts - Field 3;

Total Reject Attempts Followed by Jam - Field 4;

Total Stacker Jams - Field 5;

Total Jams without Recovery Enabled - Field 6;

Total Out Of Service Conditions - Field 6;

Total Out Of Order Conditions - Field 7;

Total Operating Hours - Field 8;

Total Documents Exceeding Max Length - Field 9;

Total Documents under Min Length - Field 10;

Total Documents Failed To Reach Escrow Position - Field 11;

Total Calibrations - Field 12;

Total Powerups - Field 13;

Total Download Attempts - Field 14;

Total Cassettes Full - Field 15;

Total Cassettes Removed - Field 16;

Message format from the Host (command)

Field	STX	Length	Msg Type & ACK	Subtype MSB	Subtype	Subtype LSB	ETX	Check Sum

Value in Hex	02	08	6n	00	00	0C	03	XX

X - Determined by message

Message format from the Acceptor (response)

Field	STX	Length	Msg type 3 Bits	Data Field 0 4 bytes	Data n . . .	Data Field 16 4 bytes	ETX	Check Sum
Value in Hex	02	49	6n	XX		XX	03	XX

X - Determined by message

Table of resultant bit values (Values for X within message field data bytes)

Data Field	Bit 3	Bit 2	Bit 1	Bit 0
Field Byte 0	2^{15}	2^{14}	2^{13}	2^{12}
Field Byte 1	2^{11}	2^{10}	2^{9}	2^{8}
Field Byte 2	2^{7}	2^{6}	2^{5}	2^{4}
Field Byte 3	2^{3}	2^{2}	2^{1}	2^{0}

6.14 Request Soft Reset

This is a message type 6, subtype 3x 7F message.

The Host can command the Acceptor perform software reset.

Message format from the Host (command)

Field	STX	Length	Msg Type & ACK	Subtype MSB	Subtype	Subtype LSB	ETX	Check Sum
Value in Hex	02	08	6n	7F	7F	7F	03	XX

X - Determined by message

The Acceptor will perform a reset. The Host should immediately begin sending any message that will contain the status bytes (non-information query message) and expect the Acceptor to briefly stop responding while it is resetting. A successful reset will be signaled with the Acceptor sending a response with the power up bit set.

☞ There is no data response.

7 Message Type 7 (Extended EBDS Messaging)

Message Type 7 is used to provide an extended set of messages that can be initiated by the Host. These extended messages are formatted by setting the Message Type to 7 and setting the Subtype byte to the corresponding subtypes shown in the table below.

Extended Messages Supported

Message	Subtype	Section
Barcode Data	01	7.1
Request Supported Note Set	02	7.2
Set Extended Note Inhibits	03	7.3
Set Escrow Timeouts	04	7.4
Set Asset Number	05	7.5
Request Value Table	06	7.6

The following extended message format is used for all extended messaging. Note that the message format is essentially the same as a standard EBDS message with the addition of the subtype and the extended data bytes. The addition of the subtype and extended data bytes shifts the standard command data, ETX, and checksum to the right inside the message.

☞ If no extended data response is required, the Acceptor will respond with a standard response message.

☞ The Host should not send extended messages to the bill acceptor during bill processing.

GENERAL EXTENDED EBDS MESSAGING FORMAT

Message format from the Host (command)

Field	STX	Length	Msg type 3 Bits	Subtype	Cmd Data Byte 0	Cmd Data Byte 1	Cmd Data Byte 2	Extended Data n bytes	ETX	Check Sum
Value in Hex	02	XX	7n	ZZ	XX	XX	XX	ZZ	03	XX

XX – Standard data determined by message

ZZ – Extended data determined by message

Message format from the Acceptor (response)

Field	STX	Length	Msg type 3 Bits	Subtype	Status Data Bytes 0-5	Extended Data n bytes	ETX	Check Sum
Value in Hex	02	XX	7n	ZZ	XX	ZZ	03	XX

XX – Standard data determined by message

ZZ – Extended data determined by message

7.1 Barcode Data

This is a message type 7

- **Subtype 0x01 message.**

The Host must enable the acceptance of Barcodes. Extended message subtype 1 is sent from the Acceptor to the Host and is used to report Barcode data.

☞ The Host must enable Barcode Acceptance by setting bit 1 in data byte 2. If the setting is ON, all Barcodes will be accepted and reported to the Host. If the setting is OFF, the Acceptor will automatically reject all Barcodes and nothing will be reported to the Host.

Message Format from the Acceptor

Field	STX	Length	Msg type 3 Bits	Subtype	Status Data Bytes 0 – 5	Extend Data Bytes 0 – n	ETX	Check Sum
Value in Hex	02	27	7n	01	XX	ZZ	03	XX

XX – Standard data determined by message

ZZ – Extended data determined by message

7.1.1 Extended Data Bytes (From Acceptor)

The extended data field for Barcodes is 28 bytes long and represented in ASCII. The Barcode data is left justified LSC (Least Significant Character) and all unused bytes are filled with 0x28. In other words the Acceptor fills this field with the barcode data starting from position Extended Data Byte 0, ending with Extended Data Byte 27. The first 0x28 indicates the end of valid Barcode data.

Example |0x30|0x31|0x32|0x33|0x34|0x35|0x36|0x37|0x38|0x39|0x28| ---- |0x28|0x28|0x28|0x28|

☞ The Acceptor always operates in Escrow Mode when processing Barcodes. The Acceptor will automatically and temporarily enter Escrow Mode when it detects a barcode assuming barcodes are enabled. Once the Barcode is processed, the Acceptor will revert to the Host's previous Escrow setting.

7.2 Request Supported Note Set

This is a message type 7

- **Subtype 0x02 message.**

This message is sent from the Acceptor to the Host in three different situations.

1. In response to the Host requesting the Note Set (Index byte set by Host).
2. At Escrow to inform the Host of the Note value (Index byte is set to zero).
3. As confirmation that the note has been stacked (Index byte is set to zero).

☞ Refer to Extended Note Processing Method Section for usage details.

Message format from the Host (command)

Field	STX	Length	Msg type 3 Bits	Subtype	Cmd Data Byte 0	Cmd Data Byte 1	Cmd Data Byte 2	Extended Data 1 byte	ETX	Check Sum
Value in Hex	02	0A	7n	02	XX	XX	XX	ZZ	03	XX

XX – Standard data determined by message

ZZ – Note Index value requested by the Host

Message format from the Acceptor (response)

Field	STX	Length	Msg type 3 Bits	Subtype	Status Data Bytes 0 – 5	Extend Data 18 Bytes	ETX	Check Sum
Value in Hex	02	1E	7n	02	XX	ZZ	03	XX

XX – Standard data determined by message

ZZ – Extended data determined by message

7.2.1 Extended Data Bytes (From Acceptor)

Type	Byte	Description
Note Index	0	Note index number reported in binary. Matches the controller note index number. In the Host request, the Host should set this to 1 for the first request and then to 2 and so on until the last notes is reported (Null Note Data). This byte is set to zero for both Escrow and Stacked messages.
ISO Code	1-3	3 character ASCII field indicating the ISO 4217 Currency code. ex. EUR Europe currency. Data Byte 0 = E (ASCII 045h) Data Byte 1 = U (ASCII 055h) Data Byte 2 = R (ASCII 052h) See Appendix A for currency codes
Base Value	4 - 6	3 character ASCII field. ex. 002 Data Byte 0 = 0 (ASCII 030h) Data Byte 1 = 0 (ASCII 030h) Data Byte 2 = 2 (ASCII 032h)
Sign	7	ASCII character indication the sign of the exponent ex: " + " (02Bh) or " - " (02Dh)
Exponent	8, 9	Exponent (Base 10) used to multiply the base field value to get real value. ex. 02 (Multiply base value by 100) Data Byte 0 = 0 (ASCII 030h) Data Byte 1 = 2 (ASCII 032h)
Orientation	10	Single character binary field indicating the inserted note orientation. ex: 00 = RU, 01 = RD, 02 = LU, 03 = LD When transmitting the note table information this field will be zero.
Type	11	This is a documented ASCII letter to define the Note Type. Reference the information supplied with Note Set.
Series	12	This is a documented ASCII letter to define the Note Series. Reference the information supplied with Note Set.
Compatibility	13	MEI specific information Note data / system compatibility
Version	14	MEI specific information Note data version
Expansion Bytes (Spare)	15-17	Reserved and should be set to zero.

7.3 Set Extended Note Inhibits

This is a message type 7

- **Subtype 0x03 message**

The Host must disable any of the 50 extended notes that it does not wish for the Acceptor to accept. Extended message subtype 3 is sent from the Host to the Acceptor and is used to set the 50 extended note inhibits.

☞ Refer to Extended Note Processing Method Section for usage details.

☞ The Host must enable the Acceptor by enabling any of the seven base notes (data byte 0) or barcodes (data byte 2, bit 1) in order to accept extended notes.

Message format from the Host (command)

Field	STX	Length	Msg type 3 Bits	Subtype	Cmd Data Byte 0	Cmd Data Byte 1	Cmd Data Byte 2	Extended Data 8 bytes	ETX	Check Sum
Value in Hex	02	11	7n	03	XX	XX	XX	ZZ	03	XX

XX – Standard data determined by message

ZZ – Extended data determined by message

Message format from the Acceptor (response)

Field	STX	Length	Msg type 3 Bits	Subtype	Status Data Bytes 0 – 5	ETX	Check Sum
Value in Hex	02	0C	7n	03	XX	03	XX

XX – Standard data determined by message

7.3.1 Extended Data Bytes (From Host)

The extended data field in the Host message consists of 8 data bytes, with each bit position corresponding to one of the 50 possible note inhibits (byte 0 bit 0 corresponds to the note inhibit for note index 1). If a bit is logical HI, the note is enabled. Logical LO means the note is inhibited.

Extended Data Byte	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Byte 0	**7**	**6**	**5**	**4**	**3**	**2**	**1**
Byte 1	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Byte 2	**21**	**20**	**19**	**18**	**17**	**16**	**15**
Byte 3	**28**	**27**	**26**	**25**	**24**	**23**	**22**
Byte 4	**35**	**34**	**33**	**32**	**31**	**30**	**29**
Byte 5	**42**	**41**	**40**	**39**	**38**	**37**	**36**
Byte 6	**49**	**48**	**47**	**46**	**45**	**44**	**43**
Byte 7	**NA**	**NA**	**NA**	**NA**	**NA**	**NA**	**50**

NA = Not used and should be set to zero.

7.4 Set Escrow Timeouts

This is a message type 7

- **Subtype 0x04 message**

The Host can set the duration of time the Acceptor will wait for a Stack or Return once a bill is reported in Escrow. Extended message subtype 4 is used to set the Note and Barcode timeout values and is sent from the Acceptor to the Host.

Message format from the Host (command)

Field	STX	Length	Msg type 3 Bits	Subtype	Cmd Data Byte 0	Cmd Data Byte 1	Cmd Data Byte 2	Extended Data Byte 0 Note Timeout	Extended Data Byte 1 Barcode Timeout	ETX	Check Sum
Value in Hex	02	0B	7n	04	XX	XX	XX	ZZ	ZZ	03	XX

XX – Standard data determined by message

ZZ – Extended data determined by message

Message format from the Acceptor (response)

Field	STX	Length	Msg type 3 Bits	Subtype	Status Data Bytes 0 – 5	ETX	Check Sum
Value in Hex	02	0C	7n	04	XX	03	XX

XX – Standard data determined by message

7.4.1 Extended Data Bytes (From Host)

The extended data field in the Host message consists of 2 data bytes. The first extended data byte is used to define the Note timeout value in seconds and the second data byte defines the Barcode timeout value in seconds.

☞ Note and Barcode timeouts values range from 1 to 127 seconds.

☞ A timeout value of “0” equals no timeout (infinite). No timeout is the default setting.

7.5 Set Asset Number

This is a message type 7

- **Subtype 0x05 message**

The Host can set the asset number that will be stored by the Acceptor and used to identify both the Acceptor and the cash box through the use of a RF Tag.

Message format from the Host (command)

Field	STX	Length	Msg type 3 Bits	Subtype	Cmd Data Byte 0	Cmd Data Byte 1	Cmd Data Byte 2	Extended Data 16 Bytes	ETX	Check Sum
Value in Hex	02	19	7n	05	XX	XX	XX	ZZ	03	XX

XX – Standard data determined by message

ZZ – Extended data determined by message

Message format from the Acceptor (response)

Field	STX	Length	Msg type 3 Bits	Subtype	Status Data Bytes 0 – 5	ETX	Check Sum
Value in Hex	02	0C	7n	05	XX	03	XX

XX – Standard data determined by message

7.5.1 Extended Data Bytes (From Host)

The extended data field in the Host message consists of 16 data bytes that make up a unique asset number for each machine.

☞ The asset number is always 16 bytes long.

☞ The asset number can be any combination of alphanumeric characters.

☞ Example 16 byte asset number:

0x30 0x31 0x32 0x33 0x34 0x35 0x36 0x37 0x38 0x39 0x41 0x42 0x43 0x44 0x45 0x46

7.6 Request Value Table

This is a message type 7

- **Subtype 0x06 message.**

This message relates the Note Value Table (Section 4.2) (this is the 'base' index 1-7 reported in the Acceptor's response while a bill is in escrow during non-extended note mode) with the actual value of the note (which is also described in the 'Extended Data Bytes' of Msg Type 7 Sub Type 2 (Section 7.2) – reference the bytes "Base Value", "Sign", and "Exponent".)

☞ This message allows the host to know that when a bill is escrowed of Note Value = 2, that its Value is, for example, a $5 or a $10, depending upon the software loaded in the bill acceptor.

! WARNING: THIS COMMAND IS ONLY VALID FOR BILL SETS WITH ONE COUNTRY OF ORIGIN. FOR EXAMPLE, THE EUR_GBP_ECO_IXP VARIANT (49069XXXX) WILL NOT WORK CORRECTLY WITH THIS COMMAND.

Message format from the Host (command)

Field	STX	Length	Msg type 3 Bits	Subtype	Cmd Data Byte 0	Cmd Data Byte 1	Cmd Data Byte 2	ETX	Check Sum
Value in Hex	02	09	7n	06	XX	XX	XX	03	XX

XX – Standard data determined by message

Message format from the Acceptor (response)

Field	STX	Length	Msg type 3 Bits	Subtype	Status Data Bytes 0 – 5	Extend Data 70 Bytes	ETX	Check Sum
Value in Hex	02	52	7n	06	XX	ZZ	03	XX

XX – Standard data determined by message

ZZ – An array of 7 denomination descriptions (70 bytes). Refer to the following table:

This table describes the “Extended Data” field of the Msg Type 7 Sub Type 6. This table will be reported seven times - once for each Note Value – in the same message response to the host.

Type	Byte	Description
Value Index	0	Value index reported in binary. This field will be ‘1’ for the first iteration of this table, ‘2’ for the second, and so on. Since this table is reported seven times, the last iteration will set this to ‘7’.
ISO Code	1-3	3 character ASCII field indicating the ISO 4217 Currency code. ex. EUR Europe currency. Data Byte 0 = E (ASCII 045h) Data Byte 1 = U (ASCII 055h) Data Byte 2 = R (ASCII 052h)
Base Value	4 - 6	3 character ASCII field. ex. 002 Data Byte 0 = 0 (ASCII 030h) Data Byte 1 = 0 (ASCII 030h) Data Byte 2 = 2 (ASCII 032h)
Sign	7	ASCII character indication the sign of the exponent ex: " + " (02Bh) or " - " (02Dh)
Exponent	8, 9	Exponent (Base 10) used to multiply the base field value to get real value. ex. 02 (Multiply base value by 100) Data Byte 0 = 0 (ASCII 030h) Data Byte 1 = 2 (ASCII 032h)

☞ If a Value Index does not have a corresponding denomination value, then all fields will be 0x00 following the Value Index indicating the unsupported Value Index.

The following is an example of how to describe the following table using this message.

Value Index	Currency
1	$1
2	- Not Supported -
3	$5
4	$10
5	$20
6	$50
7	$100

Note that ASCII is used for all fields EXCEPT the Value Index field. ASCII NULL (0x00) is used for the contents of the unsupported Value Index.

Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8	Byte 9
Value Index	ISO Code	ISO Code	ISO Code	Base Value	Base Value	Base Value	Sign	Exponent	Exponent
0x01	'U'	'S'	'D'	'0'	'0'	'1'	'+'	'0'	'0'

Byte 10	Byte 11	Byte 12	Byte 13	Byte 14	Byte 15	Byte 16	Byte 17	Byte 18	Byte 19
Value Index	ISO Code	ISO Code	ISO Code	Base Value	Base Value	Base Value	Sign	Exponent	Exponent
0x02	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00

Byte 20	Byte 21	Byte 22	Byte 23	Byte 24	Byte 25	Byte 26	Byte 27	Byte 28	Byte 29
Value Index	ISO Code	ISO Code	ISO Code	Base Value	Base Value	Base Value	Sign	Exponent	Exponent
0x03	'U'	'S'	'D'	'0'	'0'	'5'	'+'	'0'	'0'

Byte 30	Byte 31	Byte 32	Byte 33	Byte 34	Byte 35	Byte 36	Byte 37	Byte 38	Byte 39
Value Index	ISO Code	ISO Code	ISO Code	Base Value	Base Value	Base Value	Sign	Exponent	Exponent
0x04	'U'	'S'	'D'	'0'	'0'	'1'	'+'	'0'	'1'

Byte 40	Byte 41	Byte 42	Byte 43	Byte 44	Byte 45	Byte 46	Byte 47	Byte 48	Byte 49
Value Index	ISO Code	ISO Code	ISO Code	Base Value	Base Value	Base Value	Sign	Exponent	Exponent
0x05	'U'	'S'	'D'	'0'	'0'	'2'	'+'	'0'	'1'

Byte 50	Byte 51	Byte 52	Byte 53	Byte 54	Byte 55	Byte 56	Byte 57	Byte 58	Byte 59
Value Index	ISO Code	ISO Code	ISO Code	Base Value	Base Value	Base Value	Sign	Exponent	Exponent
0x06	'U'	'S'	'D'	'0'	'0'	'5'	'+'	'0'	'1'

Byte 60	Byte 61	Byte 62	Byte 63	Byte 64	Byte 65	Byte 66	Byte 67	Byte 68	Byte 69
Value Index	ISO Code	ISO Code	ISO Code	Base Value	Base Value	Base Value	Sign	Exponent	Exponent
0x07	'U'	'S'	'D'	'0'	'0'	'1'	'+'	'0'	'2'

8 Extended Note Processing Method

8.1 Start Up Overview

1. System power up.
2. Communications are established.
3. The Host enables the Acceptor.
4. Host enables the extended note mode. (Host sets Data Byte 2 Bit 4) (Note 2)
5. Host requests the note set from the Acceptor. (series of Msg Type 7, Sub Type 2's) (Note 1)
6. The Host evaluates the note set. (Note 1)
7. The Host communicates the extended note inhibits. (Msg Type 7, Sub Type 3) (Note 3)
8. Ready to accept notes.

☞ Note 1: This step is not required for the SC66. This step is not required for SC83 with Application software revisions V2.20 (Aug. 2004) or higher.

☞ Note 2: Once the Extended Note mode is established, the Host must continue to operate in the Extended Note mode.

☞ Note 3: By default, the Acceptor will enable all extended notes. If the Host does not wish to accept specific extended notes it must communicate this to the Acceptor.

☞ Extended Messages (Message Type 7) should not be sent to the bill acceptor during bill processing.

8.2 Requesting Note Table

When the Acceptor is in the Idle State the Host can request the note table by issuing request note information commands (message type 7, subtype 2) with incrementing note index values. The Acceptor will respond with extended note information for each supported note. The Host will continue to send request note information commands until it receives a null response from the Acceptor indicating the end of the supported note set. The Acceptor will be disabled until the entire note table has been transmitted to the Host.

☞ The Acceptor will only accept the request for note information message when it is in the idle state (not currently accepting a note). This is required to prevent possible confusion if a note is in process and the note information is requested.

☞ This step is not required for the SC66. This step is not required for SC83 with Application software revisions V2.20 (Aug. 2004) or higher.

8.3 Extended Note Processing Overview

1. Host must enable the Acceptor by enabling any one of the seven base notes or barcodes (Host's Data Byte 2, Bit 1).
2. Host must disable extended notes. All extended notes will be enabled by default by the Acceptor. The Host can disable any notes it does not wish to accept. The Host uses Msg Type 7, Sub Type 3 to do this.
3. Note inserted.
4. Acceptor responds with a standard accepting message.
5. Note evaluated by the Acceptor.
6. Acceptor responds with message type 7, subtype 2 reporting a note in Escrow as well as all extended note information.
7. Host evaluates the extended note information.
8. Host issues a Stack command.
9. Acceptor stacks the note and responds with message type 7, subtype 2 to report the stacked note's information.

8.4 Extended Note Message Handling

The Host will enable extended note reporting mode by setting data byte 2 bit 4. This will inform the Acceptor to report all Escrow and Stacked events using the extended note format described in this specification. Data byte 2 bit 4 should remain set in all normal messages to the Acceptor.

8.5 Escrowed Note

While the Acceptor is in the Escrow State, it will respond to all polls with message type 7, subtype 2. The Acceptor will set data byte 0 bit 2 to indicate that a bill has been Escrowed and the extended data bytes will contain the note's information.

8.6 Stacked Note

Once the Acceptor stacks a note, it will respond to the next valid poll with a message type 7, subtype 2. The Acceptor will set data byte 0 bit 3 to indicate that a bill has been Stacked and the extended data bytes will contain the note's information.

8.7 Enable / Disable Acceptor

To disable the bill acceptor (non-extended/extended modes) – Set the Host's Data Byte 0 = 0x00, and the Barcode Bit (Data Byte 2, Bit 1) = 0.

To enable the bill acceptor (non-extended/extended modes) – Set any of the bits used to inhibit the bill acceptor to 1. (If either the Host's Data Byte 0 is non-zero, or the Barcode Bit (Data Byte 2, Bit 1) is non-zero, the bill acceptor will be enabled and will draw in notes.)

To enable/inhibit notes (non-extended mode) – Use the Host's Data Byte 0 only. The Extended Note Inhibits Command (Msg Type 7, Sub Type 3) will not work.

To enable/inhibit notes (extended mode) – Use Extended Bill Inhibits Command (Msg Type 7, Sub Type 3). The Host's Data Byte 0 will not work for bill inhibits.

☞ Enabling/disabling the acceptor and enabling/inhibiting notes are separate steps. If the acceptor is enabled, it will draw notes into the acceptor, but if that note is inhibited, the bill acceptor will reject the note immediately.

8.8 Extended Note Inhibits

The power up default for the Acceptor is all extended notes enabled and they will remain enabled until inhibited by the Host using a Set Extended Note Inhibit message (message type 7, subtype 3). The extended data field in the Host message consists of 8 data bytes, with each bit representing a single entry in the note table. Byte zero bit zero identifies the first table entry. While there are eight bytes containing, 56 bits (bit eight is parity) only 50 are actually required. The extra bits are reserved and should be set to zero. A note is enabled when its corresponding inhibit bit is set to true (logic 1) and inhibited if set to false (logic 0).

8.9 Acceptor Note Report Example

(Message 0x7 Sub Type 0x02)

Below is an example response from the Acceptor. The response format is the same for a Note Table Request, Escrow Message and Stacked Message.

Example using USD $1 with RD orientation.

Field Name	Bytes	Description
Note Index	0	Data Byte 0 = **0** (Binary 00h)
ISO Currency Code	1-3	Data Byte 1 = **U** (ASCII 55h) Data Byte 2 = **S** (ASCII 53h) Data Byte 3 = **D** (ASCII 44h)
Base Value	4 - 6	Data Byte 4 = **0** (ASCII 30h) Data Byte 5 = **0** (ASCII 30h) Data Byte 6 = **1** (ASCII 32h)
Exponent Sign	7	Data Byte 7 = **+** (ASCII 02Bh)
Exponent	8, 9	Data Byte 8 = **0** (ASCII 030h) Data Byte 9 = **2** (ASCII 032h)
Orientation	10	Data Byte 10 = **1** (Binary 00000001)
Type	11	Data Byte 11 = **C** (ASCII 43h)
Series	12	Data Byte 12 = **A** (ASCII 41h)
Compatibility	13	Data Byte 13 = **B** (ASCII 42h)
Version	14	Data Byte 14 = **C** (ASCII 43h)
Spare	15-17	Three character fields all 0.

8.10 ISO Currency Codes 4217

The ISO Currency code list can be found in many locations, one example is listed below.

Source: http://www.wikipedia.org/wiki/ISO_4217

8.11 Unknown Note Reporting

For unknown notes, the Note Acceptor will report a NULL note. A NULL note has all of the extended field set to zero.

9 Flash Download

9.1 Flash Software Download

Most MEI Acceptors can be upgraded with new software via the EBDS interface. The Host will send a special message to the Acceptor that will put the Acceptor in download mode. The Host will then send a binary file that will be loaded into the Acceptor's flash memory. The Acceptor will return to normal operations once the file has been successfully received.

Remote download is initiated via the Host using message type 5. Once the Acceptor decodes a message type 5, it will respond by setting its outgoing message type to 5 and its Flash Download status bit (Byte 3 bit 1). On the next ACKed poll the Acceptor will perform a reset and expect to receive the binary data for programming into its flash memory.

Due to special considerations such as message length and additional information needed by the download process, the message format and transfer specification differs from normal operation.

☞ The Acceptor will not enter Flash Download mode if the Special Interrupt mode is active (Special Interrupt mode is not supported in Download Mode). If the Acceptor is powered down during a flash code transfer, it will return to flash download mode when powered up.

☞ Once the Acceptor has entered flash download mode, it will not accept notes until the programming operation is completed.

Software files are downloaded to the Acceptor using a similar message format as previously described. Differences include the data fields, length of data fields, and information including keeping track of the download process. The STX, Length, Message ACK Number, ETX, and Check Sum bytes retain the same meaning. The Message Type is 5.

The length of the message sent from the Host to the Acceptor is 73 bytes. The STX, Length, Message Type/ACK Number, ETX, and Check Sum consume 5 bytes, while the data and block information bytes consume 64 and 4 bytes respectively.

Software files to be downloaded to the Acceptor are binary files. Each record consists of blocks of 32 bytes. As noted previously, the eighth bit of a transmitted byte is used for parity. Therefore, in order to transmit a byte, the high nibble is transmitted followed by the low nibble; thus 64 bytes are required to transmit one block.

The block information number starts from 0 and extends beyond 1000 blocks. It requires two bytes to store it. Therefore 4 bytes are needed for the respective nibbles for transmission.

The message format from the Acceptor to the Host is identical except the data field is not included. The value of the length byte is then 73 - 64 = 9.

Message format from the Host

STX (02)	**Length** (73)	**Msg Type and Ack#**	**Block Num High Byte High Nibble**	**Block Num High Byte Low Nibble**	**Block Num Low Byte High Nibble**	**Block Num Low Byte Low Nibble**	**Data** 64 bytes	**ETX** (03)	**Check Sum**

Data field

Data Byte n High Nibble	**Data Byte n Low Nibble**	**Data Byte n+1 High Nibble**	**Data Byte n+1 Low Nibble**	**...**	**Data Byte n+31 High Nibble**	**Data Byte n+31 Low Nibble**

The block number is used as an offset, or index, into the file to be downloaded. The index is calculated by adding one to the block number and multiplying by 32.

By setting the Host's message Type to 5, the Acceptor will respond with a message Type 5. The Host continues to poll until the Acceptor responds with its Flash Download bit set (byte 3, bit 1). The Host then sets the block number to 0, reads the block from the file to download and transmits the data.

The Acceptor's response message always contains the last successfully programmed block number. If the block number returned by the Acceptor is "-1" then no blocks have been programmed. This can happen if the first block sent by the Host is NACK'd or if the Acceptor experiences a power reset and needs to restart the download process.

The block number returned by the Acceptor allows the two devices to continue the download process should either experience a power reset:

9.1.1 Acceptor experiences a power reset:

The Acceptor will detect if a Flash Download process had been interrupted and needs to be restarted. The Acceptor will set its message type to 5 and the block number to "-1" and NACK the Host. Since the Host uses the Acceptor's block number to index into the file, it will add 1 to the block number and multiply by 32 to get an index of 0; thus restarting the download process.

9.1.2 Host experiences a power reset:

The Host's software must be able to detect that the Acceptor is responding with a message type 5. The Host should then proceed with the download process by requesting the file name to download and using the Acceptor's block number to index into the file.

9.2 Flash Code Download Example

9.2.1 Example (Normal Download Operation):

	HOST											ACCEPTOR										
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**---**	**72**	**73**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
1	02	08	11	xx	Xx	xx	03	sum				02	0B	21	xx	xx	xx	xx	xx	xx	03	sum
2	02	08	50	xx	Xx	xx	03	sum				02	0B	50	xx	xx	xx	xx	xx	xx	03	sum
3	02	08	51	xx	Xx	xx	03	sum				02	0B	51	xx	xx	xx	02	xx	xx	03	sum
4	02	49	50	00	00	00	00	data	---	03	Sum	02	09	50	00	00	00	00	03	sum		
5	02	49	51	00	00	00	01	data	---	03	Sum	02	09	51	00	00	00	01	03	sum		

Frame 1: Host has sent a normal poll and Acceptor has responded.

Frame 2: Host has set the message type to 5 to request flash download mode. Acceptor has responded by setting its message type to 5.

Frame 3: Host continues to request flash download mode. Acceptor has entered flash download mode by setting its flash download bit (Byte 3, bit 1).

Frame 4: Host sends the first block of data. Acceptor has responded.

Frame 5: Host sends the second block of data. Acceptor has responded.

9.2.2 Example (Acceptor Experiences a Power Reset):

	HOST											ACCEPTOR								
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**		**72**	**73**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
N	02	49	50	00	00	00	**42**	data	---	03	sum	02	09	50	00	00	00	42	03	sum
N+1	02	49	5**1**	00	00	00	**43**	data	---	03	sum	02	09	5**0**	0**F**	0**F**	0**F**	0**F**	03	sum
N+2	02	49	5**1**	00	00	00	**00**	data	---	03	sum	02	09	5**1**	00	00	00	00	03	sum
N+3	02	49	5**0**	00	00	00	**01**	data	---	03	sum	02	09	5**0**	00	00	00	01	03	sum

Frame N: Host has sent block number 42 and Acceptor has responded. The Acceptor experiences a power reset.

Frame N+1: Host has sent block number 43 and Acceptor has NACK'd with block number "-1" (FFFF).

Frame N+2: Host resets the block number to 0 and sends the block.
The Acceptor ACKs.

Frame N+3: Host sends block number 1 and the Acceptor ACKs.

9.2.3 Example (Host Experiences a Power Reset):

	HOST											ACCEPTOR								
	1	2	3	4	5	6	7	8	**---**	72	73	1	2	3	4	5	6	7	8	9
N	02	49	51	00	00	00	**42**	data	---	03	sum	02	09	51	00	00	00	42	03	sum
N+1	02	08	**1**0	xx	xx	xx	03	sum				02	09	**51**	00	00	00	**42**	03	sum
N+2	02	49	**5**0	00	00	00	**43**	data	---	03	sum	02	09	50	00	00	00	43	03	sum

Frame N: Host has sent block number 42 and Acceptor has responded. The Host experiences a power reset.

Frame N+1: Host has sent a normal message and the Acceptor has NACK'd with message type 5 and block number 42.
Host finds the name of the file that was being downloaded.

Frame N+2: Host has set its block number to 42 + 1 = 43 and sent it.
The Acceptor has responded.

10 ABDS Protocol

ABDS is an acronym for Addressable Bi-Directional Serial communication protocol. It is used in places where an RS-485 multi-drop interface is added to the Acceptor. RS-485 allows multiple Acceptors to be daisy-chained together, all talking to the same Host using a single pair of wires. Each Acceptor is configured with a unique address by setting a dip-switch on the Acceptors interface hardware.

The Host is will poll each Acceptor one at a time. All Acceptors will hear all poll messages. To accommodate this multi-drop configuration, an address byte is added to all messages just before the ETX byte. It is always the third to last character in any message, either from the Host or from the Acceptor.

☞ The Messaging format is the exactly the same as a normal Host Poll with the addition of an address byte. The Acceptor's response is the same as normal polling.

ABDS From the Host

STX (02)	**Length** (09H)	**Msg type and Ack#**	**Data Byte 0**	**Data Byte 1**	**Data Byte 2**	**Address**	**ETX** (03)	**Check Sum**

ABDS From the Acceptor

STX (02)	**Length** (0CH)	**Msg type and Ack#**	**Data Byte 0**	**Data Byte 1**	**Data Byte 2**	**Data Byte 3**	**Data Byte 4**	**Data Byte 5**	**Address**	**ETX** (03)	**Check Sum**

The Host must know the addresses of all Acceptors on the multi-drop interface. Second, it must keep track of each Acceptors status and configuration separately. Third, the polling messages must be constructed and transmitted with the right configuration information and the correct address byte inserted into the message.

ABDS messages cannot be used in combination with special interrupt mode. This is because the ENQ message is asynchronous, there is a strong chance that two Acceptors would have their messages collide and become garbled if they tried to transmit at the same time

! Early MEI Acceptors were designed to assume all polls with a length of 9 were ABDS messages as a means of automatically detecting ABDS mode. Newer MEI models will be preset to operate in ABDS mode eliminating this possible conflict.

10.1 ABDS Download

The Host should be designed to discontinue its sequential polling scheme and dedicate all communications towards one Acceptor when it is performing a Flash download. The Acceptors that are not selected will ignore the download information.

11 Messaging Timing and Character Format

The table below contains all of the timing considerations.

11.1 Timing Table

Item	Specification	Comments
Inter Character Timing	0 to 20ms	All messages will be sent in a stream of data without gaps greater then 20ms.
Acceptors Response Time	0 to 35ms	If the Host has not started receiving a response within 35ms, it can assume that a response is not coming and should re-transmit its message.
Host ACK Response Time	0 to 100ms	If the Host does not NAK within 100ms of the Acceptor will assume the messages was received and understood by the Host. If the Host NAKs after 100ms the Acceptor will respond with it's previous message's One Shots still set. The State Bits may change.
Inactive Timing	3 seconds	If a device has been unsuccessfully receiving responses for 3 seconds, it assumes the other device has failed.
Inhibit Acceptance	30 Seconds	If the Acceptor does not receive a poll within 30 seconds it will inhibit until it receives a poll from the Host.
ENQ Response Time	0 to 100ms	If the Acceptor does not receive a poll within 100ms of an ENQs, it will generate a new one.
No Response to ENQ Time-out.	3 seconds	If the Acceptor dose not receive a poll within 3 seconds of ENQs it will inhibit until it receives a poll from the Host.

☞ If the Acceptor does not respond within 35ms the Host can wait its normal poll period and re-send its last message. In the case of Special Interrupt Mode, the Host should re-send its last message some time after the 35ms, there is no real speed requirement except in the case of a an escrow message. In this case it is a through put consideration, the Acceptor is waiting for a Host command to stack or return the document

☞ Note that it is not uncommon for the Acceptor to miss messages (during note processing, the communication channel is frequently turned off for approximately 2 seconds).

11.2 Character format:

- 1 start & 1 stop bit
- 7 data bits (bit 0 = LSB, bit 0 sent first)
- 1 parity bit (bit 7) (even parity)
- NRZ format

11.3 Baud rate:

- Standard: 9600
- Optional: 1200 (Not supported on newer models)

11.4 Duplex:

- Half duplex

☞ The receiver must wait for the sender to complete its message before transmitting. ENQs can be exception as they are asynchronous.

☞ The receiver must always reference the length bytes when evaluating a message.

12 Power Up Policies (PUPs)

12.1 Power Loss During Note Processing

Power Loss Chart – This chart explains what will happen if a power loss happens in the middle of a note transaction. The columns represent the position of the note when power is lost and the cell will indicate the action of the Acceptor when the power is restored.

	Document Position and Action on Power up		
PUP Mode	**Pre-Escrow**	**Escrowed**	**Past PONR***
PUPA	Return Bill	Wait for Host to send Stack or return message	Stack the Document and report stacked Unknown Note
PUPB	Return Bill	Go Out Of Service	Stack the Document with Note Value
PUPC	Return Bill	Go Out Of Service	Stack the Document and report stacked Unknown Note. Uses Pre-stack event during the stacking process.

Definitions:

☞ Pre-Escrow is any time between starting to draw the document in and reporting it to the Host.

☞ Escrowed is any time between reporting the Escrowed document and moving past the point of no return.

☞ PONR (Point of no return). The point of travel where a note cannot be returned to the user by reversing the transport mechanisms. This occurs between the Host's Stack Command and the Acceptors Stacked Response.

12.1.1 Operating under PUP-A:

PUPA is the default power up policy. While operating under PUPA the Acceptor will not store the note value or send special messages to the Host informing it of the note position while processing a document. If power is switched off while a note is in the escrow position and before the note reaches the PONR the Acceptor will hold the note in the escrow position on the successive power up. The Acceptor report Escrowed Unknown Value and will hold the note in the escrow position indefinitely waiting for the Host to send a stack or return command.

12.1.2 Operating under PUP-B:

While operating in PUP-B the Acceptor will store the value of the escrowed note (not Barcodes) being processed in internal non-volatile memory. If power is switched off while the note is escrowed and before it reaches the PONR, the Acceptor will simply go out of service requiring an operator to physically remove the note once power is restored. Once the note has moved passed the PONR and power is switched off the Acceptor will report the note value in its bill value field of the stacked message once power is restored. The Host machine can give credit based on this value. PUPB is the method of choice for Acceptors with internal non-volatile memory.

12.1.3 Power Loss while operating under PUP-C:

If operating in PUP-C and power is switched off while the note is escrowed and before it reaches the PONR, the Acceptor will simply go out of service once power is restored, requiring an operator to physically remove the note. Once the note has moved past the PONR, the Acceptor will report “Pre-Stacked”*. If power is switched off after this event, the Acceptor will Stack the note and report “Stacked Unknown Note” once power is restored. The Host can give credit based on the escrow value or decide not to. It’s the Host’s choice.

13 Data Byte Reference Sheet

HOST CONTROLLER DATA BYTES

BYTE 0	DESCRIPTION	"1" Indicates
BIT 0	Denomination 1 Enable	Denomination 1 enabled
BIT 1	Denomination 2 Enable	Denomination 2 enabled
BIT 2	Denomination 3 Enable	Denomination 3 enabled
BIT 3	Denomination 4 Enable	Denomination 4 enabled
BIT 4	Denomination 5 Enable	Denomination 5 enabled
BIT 5	Denomination 6 Enable	Denomination 6 enabled
BIT 6	Denomination 7 Enable	Denomination 7 enabled

BYTE 1	DESCRIPTION	"1" Indicates
BIT 0	Interrupt Mode	Enable the interrupt mode
BIT 1	Security	Enable high security
BIT 2	Orientation	See Table Below
BIT 3	Orientation	See Table Below
BIT 4	Escrow Enable	Escrow mode enabled
BIT 5	Stack	Stack the Bill
BIT 6	Return	Return the Bill

BYTE 2	DESCRIPTION	"1" Indicates
BIT 0	Push	In No Push Mode
BIT 1	Enable/Disable decoding bar ode	Barcode enabled
BIT 2	Enable/Disable PowerUp-B sequence	Enabled
BIT 3	Enable/Disable PowerUp-C sequence	Enabled
BIT 4	Extended Note Reporting	Enabled
BIT 5	Reserved – Must be 0	Not Used
BIT 6	Reserved – Must be 0	Not Used

3	2	Orientation
0	0	One Way Accept - Face up denomination first
0	1	Two Way Accept - Face up both directions
1	-	Four Way Accept - All directions

ACCEPTOR DATA BYTES

BYTE 0	A "1" Indicates	Description
BIT 0	Idling	Waiting for a bill/bar code
BIT 1	Accepting	Drawing a bill/bar code in
BIT 2	Escrowed	Bill/bar code now in Escrow
BIT 3	Stacking	In the Stacking process
BIT 4	Stacked	The bill/bar code has been Stacked
BIT 5	Returning	Returning a bill/bar code
BIT 6	Returned	Bill/bar code has been Returned

BYTE 1	A "1" Indicates	
BIT 0	Cheated	A Cheat has been detected
BIT 1	Rejected	A bill/bar code was Rejected
BIT 2	Jammed	The Acceptor is Jammed
BIT 3	Cassette Full	The Cassette is Full
BIT 4	LRC Installed	Lockable Removable Cassette status
BIT 5	Paused	The Acceptor is paused
BIT 6	Calibration	Entering Calibration mode

BYTE 2	A "1" Indicates	
BIT 0	Power Up	The Acceptor has just been reset
BIT 1	Invalid Command	Invalid command was received
BIT 2	Failure	Requires reset or repair
BIT 3-5	Bill value	See Bill Value Table below
BIT 6	Always 0	Undefined

BYTE 3	A "1" Indicates	
BIT 0	No Push Mode	Acceptor stalled in no push mode
BIT 1	Flash Download	Entering Flash Download mode
BIT 2	Prestack	The bill is not retrievable.
BIT 3-6	Always 0	Undefined

BYTE 4	Model #	Model # (00 – 07FH)

BYTE 5	Revision	Code Revision (00 - 07FH)

BILL VALUE

BIT	5	4	3	BILL VALUE
	0	0	0	None/Unknown/Bookmark/Bar Code
	0	0	1	Denomination 1
	0	1	0	Denomination 2
	0	1	1	Denomination 3
	1	0	0	Denomination 4
	1	0	1	Denomination 5
	1	1	0	Denomination 6
	1	1	1	Denomination 7

14 Denomination Codes

14.1 United States Currency Value Table

Denomination #	Value in $ US.
1	$1
2	$2
3	$5
4	$10
5	$20
6	$50
7	$100

14.2 Australian Currency Value Table

Denomination #	Value in $ Australian.
1	-
2	-
3	$5
4	$10
5	$20
6	$50
7	$100

14.3 Canadian Currency Value Table

Denomination #	Value in $ Canadian
1	-
2	$5
3	$10
4	$20
5	$50
6	$100
7	-

14.4 Euro Currency Value Table

Denomination #	Value in $ Canadian
1	$5
2	$10
3	$20
4	$50
5	$100
6	$200
7	$500

14.5 UK Currency Value Table

Denomination #	Value in $ Canadian
1	1£
2	5£
3	10£
4	20£
5	
6	
7	

15 Feature Table

This table defines which features of EBDS are available on various models of MEI Note Acceptors

Feature	Type	ZT1200	AE2600	AE2800	Series 3000	CASHFLOW SC
Stacked	Event	✓	✓	✓	✓	✓
Returned	Event	✓	✓	✓	✓	✓
Cheated	Event	✓	✓	✓	✓	✓
Rejected	Event	✓	✓	✓	✓	✓
Power-Up	Event	✓	✓	✓	✓	✓
Invalid Command	Event	✓	✓	✓	✓	✓
Note Value	Persistent Event	✓	✓	✓	✓	✓
Pause Mode	Persistent Event				✓	
Lockable Removable Cassette / Magazine	Persistent Event	✓	✓	✓	✓	✓
Book Mark Uninhibited	Special Msg.	✓		✓	✓	✓
Remote Calibration	Special Msg.	✓	✓	✓	✓	✓
Flash Download	Special Msg.	✓	✓	✓	✓	✓
Request Clear Cash in Box Counter	Special Msg.		✓	✓		
Request Number of Resets	Special Msg.		✓	✓		✓
Request Cash in Box	Special Msg.		✓	✓		
Push - No push	State	✓	✓	✓	✓	
Idling	State	✓	✓	✓	✓	✓
Accepting	State	✓	✓	✓	✓	✓
Escrowed	State	✓	✓	✓	✓	✓
Stacking	State	✓	✓	✓	✓	✓
Returning	State	✓	✓	✓	✓	✓
Jammed	State	✓	✓	✓	✓	✓
Stacker Full	State	✓	✓	✓	✓	✓
Failure	State	✓	✓	✓	✓	✓

Feature	Type	ZT1200	AE2600	AE2800	Series 3000	CASHFLOW SC
Calibration	State	✓	✓	✓	✓	✓
PowerUp B	Command	✓				✓
PowerUp C	Command	✓				✓
1200 Baud	Option	✓			✓	
ABDS capability	Option				✓	✓
Request CRC Value	Special Msg.	✓	✓	✓	✓	✓
Request Acceptor Type	Special Msg.					✓ SC83 Only **See Note 2**
Request Acceptor Serial Number	Special Msg.					✓ SC83 Only **See Note 2**
Request Acceptor Boot Software Version	Special Msg.					✓ SC83 Only **See Note 2**
Request Acceptor Application Software Version	Special Msg.					✓ SC83 Only **See Note 2**
Request Acceptor Variant Name	Special Msg.					✓ SC83 and **See Note 1 and 2**
Request Acceptor Variant Version	Special Msg.					✓ SC83 Only **See Note 2**
Request Acceptor Audit Life Time Totals	Special Msg.					✓ SC83 Only **See Note 2**
Request Acceptor Audit QP Measures	Special Msg.					✓ SC83 Only **See Note 2**
Request Acceptor Audit General Performance Measures	Special Msg.					✓ SC83 Only **See Note 2**
Request Soft Reset	Special Msg.		✓	✓		✓ SC83 Only **See Note 2**

Notes:

1. See command for model type considerations.
2. SC83 Model type 0X54 and SC66 Model type 0X55 both support these commands.

16 Appendix A – CASHFLOW SC Models

16.1 Power Up Mode

Power up configuration (A, B, C) is inhibited until after the power up sequence is done and the Acceptor clears the Power Up bit. The CASHFLOW Acceptors will power up in the configuration set prior to power down.

16.2 Note Inhibits

The Acceptor will ignore the denomination disables while processing a note. The reason is that some OEMs will disable the Acceptor after detecting the Accepting State. When the note arrives at the escrow position and is validated, the Acceptor would find the note disabled and it will be returned.

16.3 LRC Bit

The CASHFLOW Acceptor sets the LRC bit (Installed) starting with the first poll response, if the LRC is attached on power up.

16.4 Returned bit

This event is sent after the note is removed from the Acceptor mouth.

16.5 Note Value

Cassette installed event will clear note value.

16.6 ACK / NAK

New message with ACK bit the same (indicating a NAK), but has a different message type as the previous message is treated as an ACK. It is assumed, by virtue of the new message type that the previous message was received

17 Appendix B – Model Reference Table Appendix

Series 1000	Hex	Product
	0x01	ZT1000 USD (United States)
	0x11	ZT1100 USD (United States)
	0x0C	ZT1107 USD (United States)
	0x14	ZT1200 USD (United States)
	0x0F	ZT1200 AUD (Australia)
	0x16	ZT1200 CAD (Canada)
Series 2000	**Hex**	**Product**
	0x41	AUD (Australia)
	0x43	CAD (Canada)
	0x45	USD (United States) Economy (Generation 1 Platform - VN2500)
	0x46	China
	0x48	USD (United States) Economy (Generation 1 Platform - AE24V)
	0x50	USD (United States) Premium (Generation 2 Platform - AE2600)
	0x51	Philippine
	0x52	USD (United States) Reference (Generation 1 Platform - VN2500)
	0x56	USD (United States) Reference(Generation 2 Platform - VN2500)
	0x57	Brazil (Generation 2 Platform)
	0x58	USD (United States) AE2800 Expanded Premium (Generation 2 Platform)
Series 3000	**Hex**	**Product**
	0x1E	USD (United States) Series 3000 VFX (BDS)
	0x1F	USD (United States) Series 3000 EBDS
CASHFL0W SC	**Hex**	**Product**
	0X4A	CASHFLOW SC 66 (Fixed Width)
	0X54	CASHFLOW SC 83 (Multi Width / Supports Extended Note Reporting)
	0X55	CASHFLOW SC 66 (Supports Extended Note Reporting)

18 Appendix C – Frequently Asked Questions

18.1 What is the difference between BDS, EBDS, and ABDS?

EBDS is an acronym for "**E**xtended **B**i-**D**irectional **S**erial" communications and is often used to imply BDS, EBDS or ABDS.

MEI no longer recommends the use of BDS. Although some MEI legacy products support BDS, newer products do not support the use of this protocol.

BDS and EBDS differ only with the number of data bytes that are sent. EBDS was created to offer new mode selections and increased information from the Acceptor.

ABDS is an addressable version of the protocol that is used in multi drop situations. The Host sends an address byte with the message that is used to identify one of many slaves attached to the network. Only the slave addressed in the message can respond. ENQ mode is not available with ABDS.

18.2 What is an ENQ?

A single character sent by the slave device (Acceptor) to signal a significant event has occurred. The character sent is a '0x05'.

18.3 What is special interrupt mode?

In this mode the Acceptor will signal the Host with an ENQ character when an event has occurred requiring the Host's attention. In this mode message traffic can be significantly reduced when no activity is occurring in the machine which reduces the burden from the Host controller to poll the Acceptor for events.

18.4 What Baud rate should I use?

All MEI products support and recommend the use of 9600 baud. Although some legacy products support 1200 baud, newer products do not support 1200 baud.

18.5 What Poll rate should I use?

For polling mode, an interval of 200 to 300ms is recommended. For special interrupt mode the following poll rates are recommended:

To establish communications a 200 to 300ms poll rate is recommended. Once communications are established and special interrupt mode is set then a 10-second poll rate is recommended. Upon receipt of an ENQ the Host should answer as quickly as possible. If another command is needed, as in a escrow state needing a stack or return command, then 200 to 300ms is recommended.

18.6 What timing considerations are specified?

There are a few important timing considerations. Each is listed below along with the reasoning behind them.

18.6.1 Inter-Character Timing

This is a very useful timing consideration. While receiving a message, if 20ms elapses between characters the receiver should reset its receive engine to expect a start character. This can recover a bad length byte. For example, if the length was intended to be 8 chars but a noise hit made it look like 20 chars the sender will stop after 8 but the receiver will be looking for 20 chars, causing a serious synchronization problem.

Note: One could create receiver software to ignore the length character and just wait for a 20ms gap in receiving bytes. Once the gap is detected, the receiver could then analyze the received packet.

18.6.2 Acceptor Response Time

The Acceptor will respond within 35ms or not respond at all. If the Host does not receive a response interrupt within 36ms it can assume the Acceptor will not respond. The 36ms is to allow a byte transfer and Host interrupt to occur. Assuming 9600 baud and Host is using a UART in an interrupt mode. A byte is roughly 1ms (1200 baud is 4Xs longer).

Note: If the BA does not respond within 35ms the Host can wait its normal poll period and resend its last message. In the case of Special Interrupt Mode, the Host should resend its last message some time after the 35ms, there is no real speed requirement except in the case of a an escrow message. In this case it is a through put consideration, the Acceptor is waiting for a Host command to stack or return the document.

18.6.3 Acceptor Auto Disable Timing

The Acceptor will disable itself after it has transmitted a message with stacked bit set (sending a confirmation message for the document just stacked) until:

- 100ms have expired without a further polling message from the Host. However, the Acceptor must receive an acknowledging message with a new Msg/Ack # for the stacked bit to be cleared.
- an ACK message is received.

18.6.4 Inactive Timing

- HOST
  - If the Host does not receive a message / response within 3 seconds it must assume the other end has failed.
- ACCEPTOR
  - The 3-second time out also applies to the Acceptor's transmission of ENQ's. If the Acceptor does not receive a polling message after 3 seconds of sending ENQ's, it will assume the communications link is lost and automatically inhibit.
  - 30 seconds without a poll from the Host, assuming non-interrupt mode is active.

If a document is in escrow when the communications link is lost, it will be kept in escrow. Consequently, every time the link is reestablished the Host should first verify whether a document is in escrow before sending a "Stack" or "Return" command (if a document is in escrow, byte 0, bit 2) of the Acceptor's messages will =1; see 3.2). The Acceptor will stay in the disabled state until the link is established and a message enabling the acceptance of documents (see 3.1) is received.

Note that it is not uncommon for the Acceptor to miss messages (during document processing, the communication channel is frequently turned off for approximately 1 second).

18.7 Why does it take so long to do a download through EBDS?

The EBDS Download is designed to fit the normal messaging to support legacy systems.

- 9600 Baud (Legacy system speeds)
- 7 Bit data, this requires the bytes to broken into nibbles, doubling the transmission size.
- 32 byte Data payloads. Download files are sent in small chunks amplifying the overhead and turn around times.

18.8 Does EBDS run in full duplex or half duplex?

EBDS is basically a half-duplex protocol that operates as a master slave arrangement with the Host acting as the Master and the Acceptor acting as the Slave. The exception is ENQ's which are asynchronous and sent only by the slave device .

18.9 How does the Acceptor know when to send an ENQ?

The Acceptor will send an ENQ when an event occurs in the Acceptor such as a document has reached the escrow position. While in this mode it is possible that the Host and Acceptor transmit at the same time. The Host may be sending a poll at the same time as the Acceptor is sending an ENQ.

18.9.1 Events That Cause ENQ To Be Sent From The Acceptor

"STATE"	STATE ENTERED (bit is set)	STATE EXITED (bit is cleared)
Escrowed	ENQ Sent	
Stacked	ENQ Sent	
Returned	ENQ Sent	
Cheated	ENQ Sent	
Rejected	ENQ Sent	
Jammed	ENQ Sent	ENQ Sent
Cassette Full	ENQ Sent	ENQ Sent
LRC Pres.	ENQ Sent	ENQ Sent
Failure	ENQ Sent	ENQ Sent
Calibration	ENQ Sent	

18.10 How do I set up Escrow mode?

Escrow mode will be set as long as the Host continues to set the escrow mode bit to logic 1. With the escrow mode bit set low to logic zero it will set the Acceptor for no escrow mode and the Acceptor will begin to stack the document without any interaction from the Host. Although this mode

can be changed from poll to poll it is recommended that this mode bit remain the same throughout each session.

18.11 How do I set up Special Interrupt mode?

Special Interrupt mode will be active for the entire time the Host has the special interrupt bit set to logic 1. With the Special Interrupt mode bit set to logic zero the Acceptor will operate in polling mode. Although this mode can be changed from poll to poll it is recommended that this mode bit remain the same throughout each session.

18.12 What is a Power Up Policy (PUP)?

The power up policy sets the process of handling a bill being accepted during a power loss situation. By setting the policy the Host can control the actions the Acceptor will take on power

18.13 How do I set up the Power Up Policy (PUP)?

The power up policy will be set after the Acceptor has finished initializing and has powered up completely. Although this mode can be changed from poll to poll it is recommended that this mode bit remain the same throughout each session.

18.14 Which Power Up Policy should I use?

This is an OEM decision and not all Acceptors support each PUP process. The gaming Acceptors support all three PUPs.

- If you have an Acceptor without programmable non-volatile memory and you need to control a power loss situation, then you should use PUP B.
- If you have an Acceptor with programmable non-volatile memory and you need to control a power loss situation, then you should use PUP C.
- If you don't need to control a power loss situation then use the default PUPA

18.15 When will the Acceptor set the paused bit?

This is not supported on all Acceptors. The paused bit is used primarily in retail applications to indicate that a document is inserted into the bezel while another one is being processed. In some cases the Acceptor will reject the first bill causing the operator to have to reinsert it. Using the pause mode can allow the operator to feed bills more rapidly.

18.16 When should the controller issue credit?

Credit should be issued when the controller receives the Stacked event bit. Credit should be given immediately following this event. The amount to be credited will be in the document value field when the stacked event bit set. In the case of a barcode/voucher the document value will be zero when the stacked event bit is set.

18.17 Will all Acceptors follow a set sequence of events or states?

No, each Acceptor model will be relatively consistent with the events and states it reports, but when going from one model Acceptor to another the states and events may be reported differently. For

instance, one model Acceptor may set its escrow message sooner or later after the accepting state has begun than another model.

18.18 Why don't the orientation settings work?

The MEI Configuration Coupon and the EBDS Orientation are both considered when the Acceptor is deciding which orientation to accept. The higher of the two is the setting followed. For example, if the interface is set to one way and the coupon is set to four way, all four orientations will be accepted.

The factory default Configuration coupon setting four way accept. So if the Host wants to set less then four it must configure the Acceptor to 1 way accept. This will allow EBDS to control all 3 settings (1, 2 or 4 way acceptance).

18.19 Why don't the Bill Enable settings work?

There are three settings to consider: EBDS, Support Tool and the Configuration Coupon. If any one of the three is set to inhibit the document it will be rejected. In other words, in order to accept all three setting must be set to uninhibited. If the Host wants to control the acceptance of each document the Support Tool settings and the configuration Coupon must be set to uninhibited.

18.20 Can / should I design a state machine around the States?

No need to and this will force unneeded system dependency. This approach has caused most of the system issues we have found. Acceptor States are transient in nature and can be missed even at fast poll rates. Below is an example where the Idle State is missed. One document is stacked and reported by the Acceptor, and then a second document is quickly inserted.

States	
Idle	Waiting for a document
Accepting	Drawing in the document
Escrow	Validated and waiting for Host command
Stacking	Host instructed accept (Stack)
Stacked	The document is stacked.
Idle Missed	A second document is inserted between Host Polls .
Accepting	Drawing in the document

18.21 Are all states transient?

No, there are Sticky States and Temporary States. Detection of transient states must not be a critical aspect of the Host implementation.

18.21.1 Sticky States

These states are dependent on the Host or external actions.

18.21.1.1 Escrow State

The Acceptor will remain in the Escrow State (if enabled) until it receives a Stack or Return command from the Host.

18.21.1.2 Cassette Full State

The Acceptor will remain in the Cassette Full State until the cassette is removed and replaced.

Note: A power cycle may clear this state and allow a few more bills to be stacked before this state is reentered. MEI does not recommend resetting the Acceptor since it increases capacity.

18.21.1.3 Cassette Removed State

The Acceptor will remain in this state until a cassette is installed.

18.21.1.4 Jammed State

The Acceptor will remain in this state until a jam is removed.

18.21.1.5 Failure State

The Acceptor will remain in this state until a failure is corrected.

18.21.2 Temporary States

These states are transient and automatically transition to the next state.

18.21.2.1 Idling

In this state the Acceptor in not processing a document. While the Acceptor is Idle it may be disabled by the Host or due to a full or removed Cassette.

18.21.2.2 Accepting

This state starts when the Acceptor draws in a document and ends either when it Escrows or rejects the note.

NOTE: There is no Rejecting State.

18.21.2.3 Stacking

This states starts when the Acceptor moves the document into the cassette and ends when the document is stacked.

18.21.2.4 Returning

The state starts when the Host instructs the Acceptor to return a document and ends when the document is removed or reaches the rejected position.

18.21.2.5 Paused

Not supported on all Acceptors. Starts when a second document is fed into the Acceptor while it is processing the previous document and ends when it is removed.

18.21.2.6 Power Up

Set when the Acceptor powers up and ends some time after the Acceptor initializes. This is an invalid command.

18.21.3 One Time events

18.21.3.1 Cheated

Not supported on all Acceptors. Set when a cheat is detected. MEI does not offer a method to set / test cheat events.

18.21.3.2 Rejected

Indicates the rejected document has either reached the rejected position or has been removed from the Acceptor.

18.21.3.3 Stacked

Indicates the document is in the cassette and credit can be issued.

18.21.3.4 Returned

Indicates the returned document has either reached the rejected position or has been removed from the Acceptor.

18.22 What is the minimal I need to do to credit notes?

18.22.1 Basic Interface Mode Settings

- Non Escrow Mode
- All Documents Uninhibited
- Four way accept
- Voucher Off
- Set Special Interrupt Mode

  Follow the basic communications and set up the Acceptor, then wait for an ENQ. Once the ENQ is received, poll the Acceptor. Test for the following bits in the response message.

  Note: None of the states are required.
- Stacked – Issue the Value in the Note Value Filed and ACK the Acceptor
- Returned – Do Nothing
- Cheated – Do Nothing
- Rejected– Do Nothing
- Jammed – Do Nothing
- Cassette Full – Do Nothing
- LRC Pres. – Do Nothing

- Failure – Do Nothing

18.22.2 Level Two Mode Settings

Here we are adding Escrow and error reporting. Note that only the Escrow State is used and all other states are not required.

- Set Escrow Mode
- All Documents Uninhibited
- Four way accept
- Voucher Off
- Set Special Interrupt Mode

Follow the basic communications and set up the Acceptor, then wait for an ENQ. Once the ENQ is received, poll the Acceptor. Test for the following bits in the response message.

- Escrow – Decided to Accept or return the Document (Document Value Filed is Valid)
- Stacked – Issue the Value in the Note Value Filed and ACK the Acceptor
- Returned – Do Nothing
- Cheated – Do Nothing
- Rejected– Do Nothing
- Jammed – Indicate service required, if now clear, clear service required indication
- Cassette Full – Indicate service required, if now clear, clear service required indication
- LRC Pres. – Clear service required indication, If zero Indicate service required.
- Failure – Indicates service required.

18.23 Is there general Acceptor power up considerations?

The normal process for an Acceptor during power up is to perform some internal verification and then perform a “Run and Stack”. This activates the motors and positions the mechanism for a bill insertion. Various models of MEI Acceptors have different mechanisms and electronic hardware that may cause it to initialize faster or slower than other Acceptors. Some earlier model Acceptors will wait for the first communication from the Host before performing a “Run and Stack” after a power up.

Regardless of whether an Acceptor physically does a Run and Stack or not, it is guaranteed that the Acceptor will report a “Power Up” (PWR up bit set) event and a Stacked event and enter into the Idle state assuming no other exceptions are present. Not all Acceptor s will have the same timing relative to when these events occur and there is no reason to expect that these events will occur in a particular order. It is adequate just to know that they occurred and in any order.

18.23.1 Handling power up exceptions:

18.23.1.1 Powering up with the cassette removed

The Acceptor will report its LRC bit cleared (cassette removed). When this condition occurs, the Acceptor will disable itself (not draw in documents) until a cassette is inserted. Once a cassette is installed, the Acceptor will perform a run and stack and return to the idle state.

18.23.1.2 Cassette Full

Powering up the Acceptor with a full cassette will cause the Acceptor to report the cassette full event (cassette full bit set to one). When this condition occurs, the Acceptor will disable until a new (empty) cassette is installed. Once a new cassette is installed the Acceptor will perform a run and stack and return to the idle state.

18.23.1.3 Acceptor failure

If a failure occurs within the Acceptor, the Acceptor will report the failure condition by setting the failure bit to a one. When this condition occurs, the Acceptor will be disabled and is in need of operator attention to return the Acceptor to an operable state.

18.24 How can I Determine the Currency Supported?

The SC83 supports a full set of Extended EBDS commands including the reporting of the ISO 3 character currency code. However, the SC66 model 0x4A supports a subset of these commands. Please reference the Feature Table. Also see Section 6 Subtype 8.

Note: The SC66 Model 0x55 will support the full command set once released early 2005.

The current SC66 Model 0x4A supports a special “Variant Request” command. It was added to software versions:

SC66 Model 0x4A AUD Version V1.70 (MEI # 281613179)

SC66 Model 0x4A USD Version V3.50 (MEI # 281420350).

Future SC66 Model type 00X4A AUD and USD releases will support this command.

The SC66 Model 0x4A will report its variant name (either AUD or USD) to a variant request with the MSB set to “01” (i.e. 01-00-08) all other supporting models expect the MSB to be set to “00”.

Suggested Host implantation.

- Test for either Model Type SC66 0x55 or SC83 0x54. If true, the full expanded set and extended note reported is supported.
- Test for SC66 model 0x4A. If true, use the Special Variant request. This assumes the Software loaded in the Acceptor is the listed versions or newer.

☞ The SC66 Model 0x4A will respond to all unsupported commands with its CRC.