Verifone®

e315/e355 Barcode Application

Programmers Guide

Verifone Part Number DOC087-070-EN-C, Revision C

e315/e355 Barcode Application Programmers Guide

Comments? Please e-mail all comments on this document to your local Verifone Support Team.

Verifone, Inc.
1-800-Verifone
www.verifone.com

Verifone Part Number DOC087-070-EN-C, Revision C

CONTENTS

PREFACE

This programmer's manual describes the following features of the Barcode application:

- Provides descriptions of the message packets
- Provides Barcode commands and examples

This manual contains explicit information regarding use of these barcode commands and their responses to the host.

Organization

This manual is organized as follows:

CHAPTER 1	Presents an overview of the Barcode application, host request and response data packets, barcode application logging and best practices.
CHAPTER 2	Presents application commands.
Appendix A	Presents beep tone definitions.
Appendix B	Presents default parameter values.
Appendix C	Presents an example host header file.

Audience

This document is of interest to application developers creating applications for use on Verix OS or eVo based terminals.

Assumptions About the Reader

It is assumed that the reader:

- understands Linux programming concepts and terminology.
- has access to a PC running Windows XP or Windows 7.
- has installed the Mentor Sourcery CodeBench DTK and Verifone SDK on this machine.
- has access to a running, configured V/OS or eVo terminal with the Barcode application installed.

Conventions and Acronyms

Table 1 Acronyms

Abbreviation	Definition
bps	bits per second
Hz	Hertz
LSB	least-significant bit
MSB	most-significant bit
msec	millisecond
V/OS	Verifone Operating System
VRK	VeriShield Remote Key
VSS	VeriShield Security Scripts

Related Documentation

Beginning Linux Programming (4th Edition) by Neil Matthew and Richard Stones.

Verix Barcode Application

The Host Control Interface defines the message packet format between the host application on the Tablet/Phone and the Verix Barcode application on e315/e355. These message packets reflect an encapsulated data field between the host application and the Verix Barcode application.

Figure 1 illustrates the host request and response packet length and sequence. The maximum size of the host request and response packets are 2048 bytes.

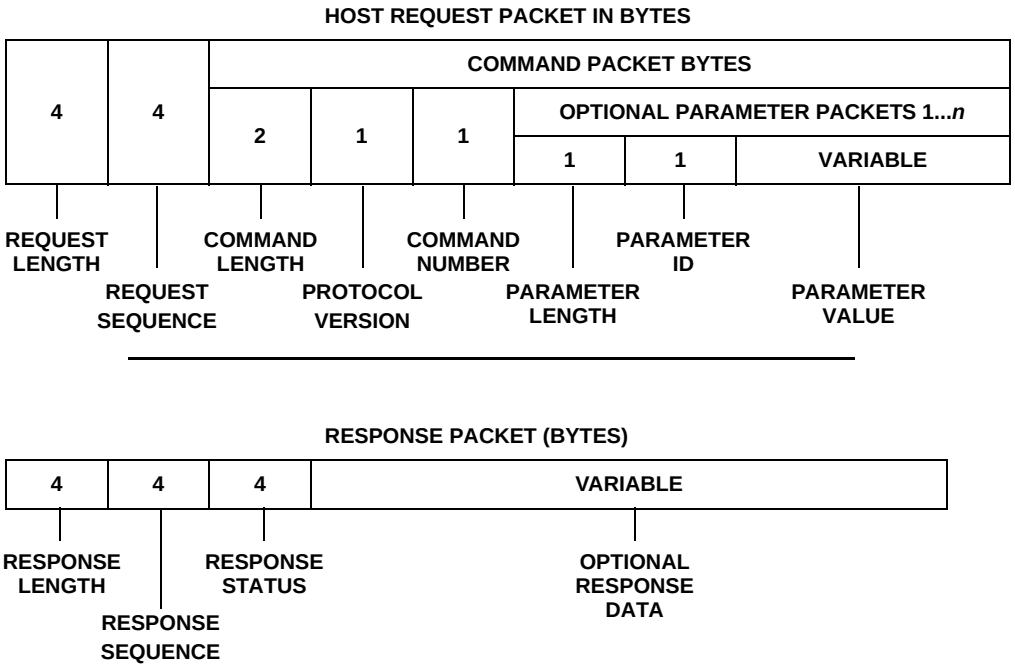

Figure 1 Host Request and Response Packet Format

The sequence number is in the packet header before the payload, and is generated by the host device. It is used for proper handshaking and error handling.

NOTE

Reference this manual with the host header file `Barcode_host.h` provided in Appendix C.

Host Request Packet

As shown in Figure 1, the Host Request packet is comprised of the following data packets:

- Request Length–The total size of the host request packet, including this packet (maximum 2048 bytes).
- Request Sequence–Supplied by the host and unique per command, each response packet sequence number is linked to the sequence number of the Request packet. However, asynchronous Response packets have random sequence numbers. Asynchronous responses are sent after the following events:
  - A successful barcode scan
  - A button change-status notification are sent in edge, level, and soft trigger mode
- Command Packet–This structure comprises the command length, protocol version, command number, and parameter packets.
- Parameter Packet–There are a variable number of Parameter packets in the Command packet. The format of the Parameter packet is parameter length, parameter ID, and parameter value.

  Multiple parameter packets can be sent in the same Host Request packet, which is useful for sending multiple parameters of a symbology in a single Command packet.

Response Packet

The Response packet is comprised of the following data packets:

- Response Length–The total size of host response packet (maximum 2048 bytes).
- Response Sequence–This is usually the sequence number that the host supplied in a Request packet. However, asynchronous responses have a random sequence number, which cannot be related to the request sequence number.
- Response Status–The process status of the previous request:
  - 0x00000000 = Success
  - 0x00000001 = Failure
  - 0x80000000= Asynchronous scanned data
  - 0x80000001= Asynchronous buttons change status

  Sometimes when response data is available immediately, the data and status are sent in a single response packet.
- Response Data (optional)–Some responses have optional data with the status update. For example, scanned data response and failure responses have optional data.

Figure 2 illustrates the data format of asynchronous responses:

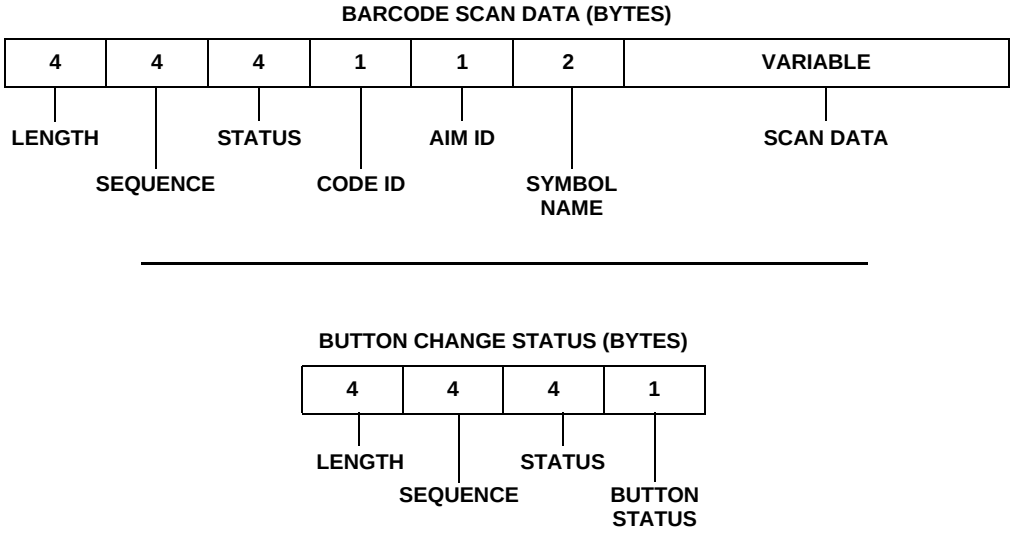

Figure 2 Asynchronous Response Packet Format

Valid Host Requests

The following lists valid host request calls:

- CMD_BAR_DEV_OPEN
- CMD_START_SCAN
- CMD_PASS_THRU
- CMD_SET_TRIG_MODE
- CMD_AUTO_ BEEP_CONFIG
- CMD_BUTTON_STATUS
- CMD_RESTORE_DEFAULTS
- CMD_SCAN_TIMEOUT
- CMD_TIMEOUT_BW_DIFF_SYM
- CMD_EN_UNIQUE_CODE_REPO
- CMD_EN_PREFIX
- CMD_EN_SUFFIX2
- CMD_EN_AIM_PATTERN
- CMD_DISABLE_ALL_SYMB
- CMD_BAR_DEV_CLOSE
- CMD_STOP_SCAN
- CMD_GET_APP_VER
- CMD_BEEP_IMMEDIATE
- CMD_GET_AUTO_BEEP_CONFIG
- CMD_GET_FIRM_VER
- CMD_EN_PICKLIST_MODE
- CMD_TIMEOUT_BW_SAME_SYM
- CMD_EN_CONTINUOUS_RD
- CMD_EN_MOBILE_PH_MODE
- CMD_EN_SUFFIX1
- CMD_EN_XMIT_FMT
- CMD_SYMBOLOGY

CHAPTER 2

Barcode Commands

The following commands are discussed in this chapter:

Barcode Device Open

Powers barcode device and opens the communication port to communicate with the Barcode module. Barcode module takes about 1.2 seconds to power-up and is ready to be used once powered up.

NOTE

This command must be sent before sending any other barcode command. Barcode commands are defined in the host header file as shown in Appendix C.

Command Prototype

Length (4 bytes)	Sequence (4 bytes)	Length (2 bytes)	Protocol Version (1 byte)	Command # (1 byte)
0x0C	-	0x04	0x01	CMD_BAR_DEV_OPEN

Command Example

Raw data in hex format (MSB to LSB): 0x00 00 00 0C 00 00 00 01 00 04 01 1A

Response Prototype

Length (4 bytes)	Sequence (4 bytes)	Status (4 bytes)	Optional Reason (1 byte)
0x0D	-	0x00/0x01	REASON

Response Values

SUCCESS — Response example: 0x00 00 00 0C 00 00 00 01 00 00 00 00

Failure: Failure responses 0x00000001 contains optional reason byte. Possible reason byte values in failure response for Device Open request are:

- REASON = 8 when barcode module cannot be opened due to low battery voltage
- REASON = 35 when there is no response from barcode module after 3 seconds timeout.

Failure Response example: 0x00 00 00 0D 00 00 00 01 00 00 00 01 23

Barcode Device Close

Powers down the barcode module and closes the communication port with module.

NOTE

Barcode module consumes approximately 30 mA when it is powered on (with scanner lights turned off). It is recommended to close barcode device when not in use to save battery power. Barcode Device Open needs to be called after Device Close to communicate with the barcode module again.

Command Prototype

Length (4 bytes)	Sequence (4 bytes)	Length (2 bytes)	Protocol Version (1 byte)	Command # (1 byte)
0x0C	-	0x04	0x01	CMD_BAR_DEV_CLOSE

Command Example

Raw data in hex format (MSB to LSB): 0x00 00 00 0C 00 00 00 01 00 04 01 1B

Response Prototype

Length (4 bytes)	Sequence (4 bytes)	Status (4 bytes)	Optional Reason(1 byte)
0x0D	-	0x00/0x01	REASON

Response Values

SUCCESS — Response example: 0x00 00 00 0C 00 00 00 01 00 00 00 00

Failure: Possible reason byte values in failure response for Device Close request are:

- REASON = 22 when barcode module is already closed or not opened.
- REASON = 35 when there is no response from barcode module after 3 seconds timeout.

Failure response example: 0x00 00 00 0D 00 00 00 01 00 00 00 01 16

Barcode Start Scan

Activates the scan by sending a START_SCAN command to the barcode module. The host initially waits for a Success/Failure response for this command. After receiving the Success response, the host waits for an asynchronous scan data response packet. All trigger modes, such as soft, passive, edge, and level trigger modes require START_SCAN to activate the barcode module with or without using the button(s).

NOTE Success contains status value of 0x00000000 in the response packet. The Failure response contains status value of 0x00000001 in the response packet and reason byte indicating type of failure response.

Command Prototype

Length (4 bytes)	Sequence (4 bytes)	Length (2 bytes)	Protocol Version (1 byte)	Command # (1 byte)
0x0C	-	0x04	0x01	CMD_START_SCAN

Command Example

Raw data in hex format (MSB to LSB): 0x00 00 00 0C 00 00 00 01 00 04 01 01

Response Prototype

Length (4 bytes)	Sequence (4 bytes)	Status (4 bytes)	Optional Reason (1 byte)
0x0D	-	0x00/0x01	REASON

Response Values

SUCCESS — Response example: 0x00 00 00 0C 00 00 00 01 00 00 00 00

Failure: Possible reason byte values in failure response for Start Scan request are:

- REASON = 8 when battery voltage is low.
- REASON = 22 when barcode module is already closed or not opened.
- REASON = 35 when there is no response from barcode module after 3 seconds timeout.
- REASON = 0 when trigger mode is LEVEL/EDGE and battery voltage is low.

Failure response example: 0x00 00 00 0D 00 00 00 01 00 00 00 01 16

Barcode Stop Scan

Deactivates scanner by sending a STOP_SCAN command to the barcode reader. The host receives a Success/Failure response packet for this command.

Command Prototype

Length (4 bytes)	Sequence (4 bytes)	Length (2 bytes)	Protocol Version (1 byte)	Command # (1 byte)
0x0C	-	0x04	0x01	CMD_STOP_SCAN

Command Example

Raw data in hex format (MSB to LSB): 0x00 00 00 0C 00 00 00 01 00 04 01 02

Response Prototype

Length (4 bytes)	Sequence (4 bytes)	Status (4 bytes)	Optional Reason (1 byte)
0x0D	-	0x00/0x01	REASON

Response Values

SUCCESS — Response example: 0x00 00 00 0C 00 00 00 01 00 00 00 00

Failure: Possible reason byte values in failure response for Stop Scan request are:

- REASON = 8 battery voltage is low.
- REASON = 22 when barcode application received an invalid command or when more than one command is seen in a request packet.
- REASON = 35 when there is no response from barcode module after 3 seconds timeout.

Failure response example: 0x00 00 00 0D 00 00 00 01 00 00 00 01 16

Barcode Passthru

Reserved.

Barcode Application Version

Returns a null-terminated 6-byte ASCII version ID for the barcode reader.

Command Prototype

Length (4 bytes)	Sequence (4 bytes)	Length (2 bytes)	Protocol Version (1 byte)	Command # (1 byte)
0x0C	-	0x04	0x01	CMD_SINGLE_GET_APP_VER

Command Example

Raw data in hex format (MSB to LSB): 0x00 00 00 0C 00 00 00 01 00 04 01 04

Response Prototype

Length (4 bytes)	Sequence (4 bytes)	Status (4 bytes)	Optional Reason(1 byte)
0x0D	-	0x00/0x01	REASON

Response Values

SUCCESS — Response example:
0x00 00 00 13 00 00 00 01 00 00 00 00 30 31 30 38 33 32 00

Failure: Possible reason byte values in failure response for Application Version request are:

- REASON = 8 battery voltage is low.
- REASON = 22 when barcode application received an invalid command or when more than one command is seen in a request packet.
- REASON = 35 when there is no response from barcode module after 3 seconds timeout.

Failure response example: 0x00 00 00 0D 00 00 00 01 00 00 00 01 16

Barcode Trigger Mode

Sets the barcode trigger mode using the 1-byte parameter value. The terminal reverts to the default Level trigger mode at every terminal power cycle and restart. The trigger modes are discussed below:

- The EDGE parameter enables Edge Trigger mode, where the scan session starts when the barcode buttons are pressed and released, and stays on until the timeout period expires or the barcode buttons are pressed again and released.
- The LEVEL parameter enables Level Trigger mode, where the scan session starts when the barcode buttons are pressed, and stays on until the buttons are released. The scan session turns off automatically when the specified timeout period expires.
- The SOFT parameter enables Soft Trigger mode, where the host device issues a command to start the scan session, and the barcode buttons have no control over starting the scan session but asynchronous button status response is sent to host whenever buttons were pressed.
- The PASSIVE parameter enables Passive Trigger mode that behaves the same as Soft Trigger mode, except that a BUTTON_STATUS response is NOT sent to host whenever buttons were pressed. Host can obtain button status by sending CMD_BUTTON_STATUS command.

NOTE

In Edge and Level Trigger modes, the START_SCAN command activates barcode buttons, and the STOP_SCAN command deactivates barcode buttons.

- By default, Continuous mode is enabled. When Continuous mode is disabled, a single scan mode is enabled, wherein a scan session is turned off after each successful barcode scan or when scan timeout period expires.

Command Prototype

Length (4 bytes)	Sequence (4 bytes)	Length (2 bytes)	Protocol Version (1 byte)	Command # (1 byte)
0x0F	-	0x07	0x01	CMD_SET_TRIG_MODE

Length (1 byte)	ID (1 byte)	Value (1 byte)
0x03	PID_SET_TRIG_MODE	0x00-0x03

Command Example

Raw data in hex format (MSB to LSB): 0x00 00 00 0F 00 00 00 01 00 07 01 05 03 FE 01

Response Prototype

Length (4 bytes)	Sequence (4 bytes)	Status (4 bytes)	Optional Reason (1 byte)
0x0D	-	0x00/0x01	REASON

Response Values

SUCCESS — Response example: 0x00 00 00 0C 00 00 00 01 00 00 00 00

Failure: Possible reason byte values in failure response for Trigger Mode request are:

- REASON = 8 battery voltage is low.
- REASON = 22 when barcode application received an invalid command or when more than one command is seen in a request packet.
- REASON = 35 when there is no response from barcode module after 3 seconds timeout.
- REASON = 0 when trigger mode is LEVEL/EDGE and battery voltage is low.

Failure response example: 0x00 00 00 0D 00 00 00 01 00 00 00 01 16

Beep Immediate

Provides beep parameters for an immediate beep sequence. Specifies the parameter value in the beep format as shown below. This format requires the number of beeps (maximum of 6) followed by the values in the beep definition. Setting the beep frequency to zero disables beeping. See Appendix A for the supported range of beep frequencies.

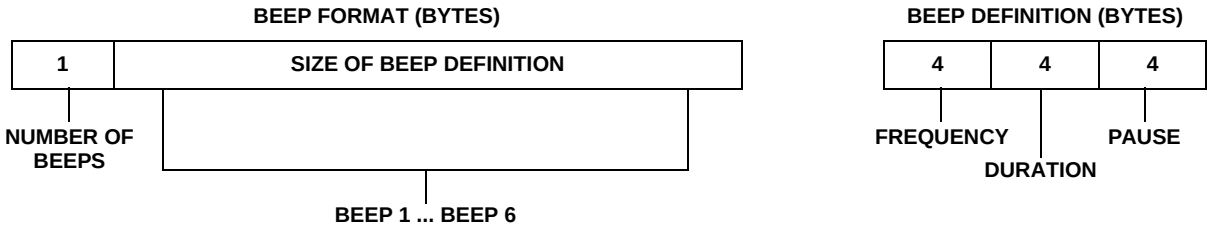

Figure 3 Beep Command Packet

Command Prototype

Length (4 bytes)	Sequence (4 bytes)	Length (2 bytes)	Protocol Version (1 byte)	Command # (1 byte)
0x57	-	0x4F	0x01	CMD_BEEP_IMMEDIATE

Length (1 byte)	ID (1 byte)	Value (size of T_BEEP)
0x4B	PID_BEEP_IMMEDIATE	Value in BEEP_FORMAT

Command Example

Raw data in hex format (MSB to LSB): 0x00 00 00 27 00 00 00 01 00 1F 01 06 1B FD 02 00 00 00 40 00 00 00 32 00 00 00 32 00 00 00 40 00 00 00 32 00 00 00 32

Response Prototype

Length (4 bytes)	Sequence (4 bytes)	Status (4 bytes)
0x0C	-	0x00/0x01

Response Values

SUCCESS Response example: 0x00 00 00 0C 00 00 00 01 00 00 00 00

Failure: No reason byte in failure response for Beep immediate request.
Failure response example: 0x00 00 00 0C 00 00 00 01 00 00 00 01

Auto Beep Configuration

Enables and disables auto beep. When enabled, the terminal beeps after a successful scan and when an error is detected. This command also allows scan beep and error beep parameter configuration.

This API requires up to three input parameters. In sequential order, the first parameter must start with `beep mode`. The second parameter can be either a scan beep or an error beep, depending on how `beep mode` is set. The third parameter becomes an error beep when `beep mode` is set to 3.

Valid values for `beep mode` are 0, 1, 2, and 3:

- 0 = auto beep disabled
- 1 = auto beep enabled; configures scan beeps only
- 2 = auto beep enabled; configures error beeps only
- 3 = auto beep enabled; configures both scan and error beeps

Only one parameter is required in the Command to disable or enable auto beep. Two parameters are required to configure either scan beeps or error beeps. Three parameters are required to configure both scan and error beeps.

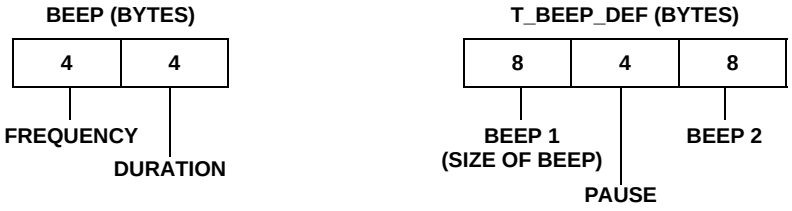

Figure 4 Beep Packet Definitions

NOTE

Default auto beep configuration settings are enforced at terminal power cycle and restart.

Scan Beep	
Frequency	64 (2217 Hz)
Duration	50 msec
Pause	50 msec
Frequency	50 (988 Hz)
Duration	50 msec

Error Beep	
Frequency	58 (1568 Hz)
Duration	100 msec
Pause	75 msec
Frequency	50 (988 Hz)
Duration	100 msec

Command Prototype

Command # (1 byte)	Length (1 byte)	ID (1 byte)	Value (1 byte)
CMD_AUTO_BEEP_CONFIG	0x03	PID_AUTO_BEEP_SCAN	0x00-0x03

Length (1 byte)	ID (1 byte)	Value (1 byte)
0x16	PID_AUTO_BEEP_SCAN	Value in format T_BEEP_DEF

Length (1 byte)	ID (1 byte)	Value (20 bytes)
0x16	PID_AUTO_BEEP_ERROR	Value in format T_DEEP_DEF

Command Example

Raw data in hex format (MSB to LSB): 0x00 00 00 3B 00 00 00 01 00 33 01 1C 2F EC 03 16 EB 00 00 00 40 00 00 00 32 00 00 00 32 00 00 00 32 00 00 00 32 16 EA 00 00 00 3A 00 00 00 64 00 00 00 4B 00 00 00 32 00 00 00 64

Response Prototype

Length (4 bytes)	Sequence (4 bytes)	Status (4 bytes)	Optional Reason (1 byte)
0x0D	-	0x00/0x01	REASON

Response Values

SUCCESS — Response example: 0x00 00 00 0C 00 00 00 01 00 00 00 00

Failure: — Possible reason byte values in failure response for Auto Beep configuration request:

- REASON = 22 when barcode application received an invalid command for Auto Beep configuration

Failure response example: 0x00 00 00 0D 00 00 00 01 00 00 00 01 16

Get Auto Beep Configuration

Retrieves auto beep current configuration. For example, the response has 1 byte of data when the `beep mode` setting is 0, 21 bytes of data when the `beep mode` setting is 1 or 2, and 41 bytes of data when the `beep mode` setting is 3.

Command Prototype

Length (4 bytes)	Sequence (4 bytes)	Length (2 bytes)	Protocol Version (1 byte)	Command # (1 byte)
0x0C	-	0x04	0x01	CMD_GET_AUTO_BEEP_CONFIG

Command Example

Raw data in hex format (MSB to LSB): 0x00 00 00 0C 00 00 00 01 00 04 01 1D

Response Prototype

Length (4 bytes)	Sequence (4 bytes)	Status (4 bytes)
0x0C	-	0x00/0x01

Response includes auto beep mode, scan beep, and error beep configuration.

Response Values

SUCCESS — Response example: 0x00 00 00 35 00 00 00 01 00 00 00 00 03 00 00 00 40 00 00 00 32 00 00 00 32 00 00 00 32 00 00 00 32 00 00 00 3A 00 00 00 64 00 00 00 4B 00 00 00 32 00 00 00 64

Failure: No reason byte values in failure response for Get Auto Beep configuration request

Failure response example: 0x00 00 00 0C 00 00 00 01 00 00 00 01

Barcode Button Status

Reports the current state of the barcode trigger buttons. Button status is reported as a binary mask. Bit '0' is set when button 1 is pressed and bit '1' is set when button 2 is pressed. Each bit is 0 when respective buttons are not pressed.

Command Prototype

Length (4 bytes)	Sequence (4 bytes)	Length (2 bytes)	Protocol Version (1 byte)	Command # (1 byte)
0x0C	-	0x04	0x01	CMD_SINGLE_BUTTON_STATUS

Command Example

Raw data in hex format (MSB to LSB): 0x00 00 00 0C 00 00 00 01 00 04 01 07

Response Prototype

Length (4 bytes)	Sequence (4 bytes)	Status (4 bytes)	Optional Reason (1 byte)
0x0D	-	0x00/0x01	REASON

Response Values

SUCCESS — Response example: 0x00 00 00 0D 00 00 00 01 00 00 00 00 03

Button status response is 1 byte and returned either as value 1 or 2 or 3.

Failure: Possible reason byte values in failure response for Button Status request:

- REASON = 8 battery voltage is low.
- REASON = 22 when barcode application received an invalid command or when more than one command is seen in a request packet.
- REASON = 35 when there is no response from barcode module after 3 seconds timeout.

Failure response example: 0x00 00 00 0D 00 00 00 01 00 00 00 01 16

Barcode Firmware Version

Returns the firmware version number/string(s) of the barcode device.

Command Prototype

Length (4 bytes)	Sequence (4 bytes)	Length (2 bytes)	Protocol Version (1 byte)	Command # (1 byte)
0x0C	-	0x04	0x01	CMD_SINGLE_GET_FIRM_VER

Command Example

Raw data in hex format (MSB to LSB): 0x00 00 00 0C 00 00 00 01 00 04 01 07

Response Prototype

Length (4 bytes)	Sequence (4 bytes)	Status (4 bytes)	Optional Reason (1 byte)
0x0D	-	0x00/0x01	Version/REASON

Response Values

SUCCESS — Response example: 0x00 00 00 25 00 00 00 01 00 00 00 00 50 41 41 42 4C 43 30 35 2D 30 30 32 2D 52 30 31 20 20 20 46 20 55 20 01 01

Failure: Possible reason byte values in failure response for Firmware Version request:

- REASON = 8 battery voltage is low.
- REASON = 22 when barcode application received an invalid command or when more than one command is seen in a request packet.
- REASON = 35 when there is no response from barcode module after 3 seconds timeout.

Failure response example: 0x00 00 00 0D 00 00 00 01 00 00 00 01 16

Barcode Restore Defaults

Sets selected parameters to their default values. See Appendix C for default values of all parameters.

Command Prototype

Length (4 bytes)	Sequence (4 bytes)	Length (2 bytes)	Protocol Version (1 byte)	Command # (1 byte)
0x0C	-	0x04	0x01	CMD_RESTORE_DEFAULTS

Command Example

Raw data in hex format (MSB to LSB): 0x00 00 00 0C 00 00 00 01 00 04 01 0B

Response Prototype

Length (4 bytes)	Sequence (4 bytes)	Status (4 bytes)	Optional Reason (1 byte)
0x0D	-	0x00/0x01	REASON

Response Values

SUCCESS — Response example: 0x00 00 00 0C 00 00 00 01 00 00 00 00

Failure: Possible reason byte values in failure response for Restore Defaults request:

- REASON = 8 battery voltage is low.
- REASON = 22 when barcode application received an invalid command or when more than one command is seen in a request packet.
- REASON = 35 when there is no response from barcode module after 3 seconds timeout.

Failure response example: 0x00 00 00 0D 00 00 00 01 00 00 00 01 16

Barcode Picklist Mode

Enables or disables barcode Picklist mode, where the barcode reader decodes only barcodes that are aligned under the center of the aiming pattern.

Command Prototype

Command # (1 byte)	Length (1 byte)	ID (1 byte)	Value (1 byte)
CMD_EN_PICKLIST_MODE	0x03	PID_PICKLIST_MODE	0x01 or 0x00

Command Example

Raw data in hex format (MSB to LSB): 0x00 00 00 0F 00 00 00 01 00 04 01 0C 03 FB 01

Response Prototype

Length (4 bytes)	Sequence (4 bytes)	Status (4 bytes)	Optional Reason (1 byte)
0x0D	-	0x00/0x01	REASON

Response Values

SUCCESS — Response example: 0x00 00 00 0C 00 00 00 01 00 00 00 00

Failure: Possible reason byte values in failure response for Picklist mode request:

- REASON = 8 battery voltage is low.
- REASON = 22 when barcode application received an invalid command or when more than one command is seen in a request packet.
- REASON = 35 when there is no response from barcode module after 3 seconds timeout.

Failure response example: 0x00 00 00 0D 00 00 00 01 00 00 00 01 16

Barcode Scan TImeout

Times out the active session at the specified time. The timeout value range is 1 to 255 seconds in the following hex format:

- 1 second = 0x01
- 10 seconds = 0x0A
- 255 seconds = 0xFF

The maximum timeout value allowed in Continuous and Single Scan modes are 255 seconds and 10 seconds respectively.

Command Prototype

Command # (1 byte)	Length (1 byte)	ID (1 byte)	Value (1 byte)
CMD_SCAN_TIMEOUT	0x03	GEN_PID_SCAN_TIMEOUT	0x01 or 0xFF

Command Example

Raw data in hex format (MSB to LSB): 0x00 00 00 0F 00 00 00 01 00 07 01 0D 03 FA FF

Response Prototype

Length (4 bytes)	Sequence (4 bytes)	Status (4 bytes)	Optional Reason (1 byte)
0x0D	-	0x00/0x01	REASON

Response Values

SUCCESS — Response example: 0x00 00 00 0C 00 00 00 01 00 00 00 00

Failure: Possible reason byte values in failure response for Scan Timeout request:

- REASON = 8 battery voltage is low.
- REASON = 22 when barcode application received an invalid command or when more than one command is seen in a request packet.
- REASON = 35 when there is no response from barcode module after 3 seconds timeout.

Failure response example: 0x00 00 00 0D 00 00 00 01 00 00 00 01 16

Barcode Timeout – Same Symbol Decodes

Use this command in Continuous mode to prevent multiple reads of a symbol left in the decoder's field of view. The timeout begins when a symbol is removed from the field of view. The default value is 0.6 seconds.

Command Prototype

Command # (1 byte)	Length (1 byte)	ID (1 byte)	Value (1 byte)
CMD_TIMEOUT_BW_SAME SYM	0x03	GEN_PID_TIMEOUT_ BW_SAME_SYM	0x00 or 0x63

Command Example

Raw data in hex format (MSB to LSB): 0x00 00 00 0F 00 00 00 01 00 07 01 0E 03 F9 63

Response Prototype

Length (4 bytes)	Sequence (4 bytes)	Status (4 bytes)	Optional Reason (1 byte)
0x0D	-	0x00/0x01	REASON

Response Values

SUCCESS — Response example: 0x00 00 00 0C 00 00 00 01 00 00 00 00

Failure: Possible reason byte values in failure response for Timeout Same Symbol request:

- REASON = 8 battery voltage is low.
- REASON = 22 when barcode application received an invalid command or when more than one command is seen in a request packet.
- REASON = 35 when there is no response from barcode module after 3 seconds timeout.

Failure response example: 0x00 00 00 0D 00 00 00 01 00 00 00 01 16

Barcode Timeout – Different Symbol Decodes

Use this command in Continuous mode to control the time the scanner is inactive between decoding different symbols. The default value is 0.2 seconds.

Command Prototype

Command # (1 byte)	Length (1 byte)	ID (1 byte)	Value (1 byte)
CMD_TIMEOUT_BW_DIFF_SYM	0x03	GEN_PID_TIMEOUT_BW_DIFF_SYM	0x01 or 0x63

Command Example

Raw data in hex format (MSB to LSB): 0x00 00 00 0F 00 00 00 01 00 07 01 0F 03 F8 63

Response Prototype

Length (4 bytes)	Sequence (4 bytes)	Status (4 bytes)	Optional Reason (1 byte)
0x0D	-	0x00/0x01	REASON

Response Values

SUCCESS — Response example: 0x00 00 00 0C 00 00 00 01 00 00 00 00

Failure: Possible reason byte values in failure response for Timeout Different Symbol request:

- REASON = 8 battery voltage is low.
- REASON = 22 when barcode application received an invalid command or when more than one command is seen in a request packet.
- REASON = 35 when there is no response from barcode module after 3 seconds timeout.

Failure response example: 0x00 00 00 0D 00 00 00 01 00 00 00 01 16

Barcode Continuous Mode

Enables or disables Continuous mode, where the barcode laser scans barcode continuously one after the other. Normally, the laser shuts off and the scanning session ends after one successful scan/decode. Continuous mode can be enabled with any of the Trigger modes. By default, Continuous mode is enabled.

Command Prototype

Command # (1 byte)	Length (1 byte)	ID (1 byte)	Value (1 byte)
CMD_EN_CONTINUOUS_RD	0x03	GEN_PID_CONTINUOUS_READ	0x01 or 0x00

Command Example

Raw data in hex format (MSB to LSB): 0x00 00 00 0F 00 00 00 01 00 07 01 10 03 F7 63

Response Prototype

Length (4 bytes)	Sequence (4 bytes)	Status (4 bytes)	Optional Reason (1 byte)
0x0D	-	0x00/0x01	REASON

Response Values

SUCCESS — Response example: 0x00 00 00 0C 00 00 00 01 00 00 00 00

Failure: Possible reason byte values in failure response for Continuous mode request:

- REASON = 8 battery voltage is low.
- REASON = 22 when barcode application received an invalid command or when more than one command is seen in a request packet.
- REASON = 35 when there is no response from barcode module after 3 seconds timeout.

Failure response example: 0x00 00 00 0D 00 00 00 01 00 00 00 01 16

Barcode Unique Code Report

Use this command in Continuous mode to report only unique bar codes during a scan session.

Command Prototype

Command # (1 byte)	Length (1 byte)	ID (1 byte)	Value (1 byte)
CMD_EN_UNIQUE_CODE_ REPO	0x03	GEN_PID_UNIQUE_CODE_ REPORT	0x01 or 0x00

Command Example

Raw data in hex format (MSB to LSB): 0x00 00 00 0F 00 00 00 01 00 07 01 11 03 F6 01

Response Prototype

Length (4 bytes)	Sequence (4 bytes)	Status (4 bytes)	Optional Reason (1 byte)
0x0D	-	0x00/0x01	REASON

Response Values

SUCCESS — Response example: 0x00 00 00 0C 00 00 00 01 00 00 00 00

Failure: — Possible reason byte values in failure response for Unique Code request:

- REASON = 8 battery voltage is low.
- REASON = 22 when barcode application received an invalid command or when more than one command is seen in a request packet.
- REASON = 35 when there is no response from barcode module after 3 seconds timeout.

Failure response example: 0x00 00 00 0D 00 00 00 01 00 00 00 01 16

Barcode Mobile Phone/Display Mode

Enables or disables mobile phone/display mode, which improves bar code reading performance with target barcodes displayed on mobile phones and electronic displays.

Command Prototype

Command # (1 byte)	Length (1 byte)	ID (1 byte)	Value (1 byte)
CMD_EN_MOBILE_PH_ MODE	0x03	GEN_PID_MOBILE_PH_ MODE	0x01 or 0x00

Command Example

Raw data in hex format (MSB to LSB): 0x00 00 00 0F 00 00 00 01 00 07 01 12 03 F5 01

Response Prototype

Length (4 bytes)	Sequence (4 bytes)	Status (4 bytes)	Optional Reason (1 byte)
0x0D	-	0x00/0x01	REASON

Response Values

SUCCESS — Response example: 0x00 00 00 0C 00 00 00 01 00 00 00 00

Failure: Possible reason byte values in failure response for Mobile Phone/ Display mode request:

- REASON = 8 battery voltage is low.
- REASON = 22 when barcode application received an invalid command or when more than one command is seen in a request packet.
- REASON = 35 when there is no response from barcode module after 3 seconds timeout.

Failure response example: 0x00 00 00 0D 00 00 00 01 00 00 00 01 16

Barcode Scan Data Prefix

This command allows user to set the prefix value which is transmitted as part of scanned data when the scan data transmission format command ‘prefix + scan data’ is selected.

Command Prototype

Command # (1 byte)	Length (1 byte)	ID (1 byte)	Value (1 byte)
CMD_PREFIX_VAL	0x03	GEN_PID_PREFIX_KEY	0x01

Length (1 byte)	ID (1 byte)	Value (1 byte)
0x03	GEN_PID_PREFIX_VAL	0x00-0xFF

Command Example

Raw data in hex format (MSB to LSB): 0x00 00 00 12 00 00 00 01 00 0A 01 13 03 F4 01 03 F3 0D

Response Prototype

Length (4 bytes)	Sequence (4 bytes)	Status (4 bytes)	Optional Reason (1 byte)
0x0D	-	0x00/0x01	REASON

Response Values

SUCCESS — Response example: 0x00 00 00 0C 00 00 00 01 00 00 00 00

Failure: Possible reason byte values in failure response for Scan Data Prefix request:

- REASON = 8 battery voltage is low.
- REASON = 22 when barcode application received an invalid command or when more than one command is seen in a request packet.
- REASON = 35 when there is no response from barcode module after 3 seconds timeout.

Failure response example: 0x00 00 00 0D 00 00 00 01 00 00 00 01 16

Barcode Scan Data Suffix1

This command allows user to set the suffix1 value which is transmitted as part of scanned data when the scan data transmission format command 'suffix1 + scan data' is selected.

Command Prototype

Command # (1 byte)	Length (1 byte)	ID (1 byte)	Value (1 byte)
CMD_SUFFIX1_VAL	0x03	GEN_PID_SUFFIX1_KEY	0x01

Length (1 byte)	ID (1 byte)	Value (1 byte)
0x03	GEN_PID_SUFFIX1_VAL	0x00-0xFF

Command Example

Raw data in hex format (MSB to LSB): 0x00 00 00 12 00 00 00 01 00 0A 01 14 03 F2 01 03 F1 0D

Response Prototype

Length (4 bytes)	Sequence (4 bytes)	Status (4 bytes)	Optional Reason (1 byte)
0x0D	-	0x00/0x01	REASON

Response Values

SUCCESS — Response example: 0x00 00 00 0C 00 00 00 01 00 00 00 00

Failure: Possible reason byte values in failure response for Scan Data Prefix request:

- REASON = 8 battery voltage is low.
- REASON = 22 when barcode application received an invalid command or when more than one command is seen in a request packet.
- REASON = 35 when there is no response from barcode module after 3 seconds timeout.

Failure response example: 0x00 00 00 0D 00 00 00 01 00 00 00 01 16

Barcode Scan Data Suffix2

This command allows user to set the suffix1 value which is transmitted as part of scanned data when the scan data transmission format command ‘suffix2 + scan data’ is selected.

Command Prototype

Command # (1 byte)	Length (1 byte)	ID (1 byte)	Value (1 byte)
CMD_SUFFIX2_VAL	0x03	GEN_PID_SUFFIX2_KEY	0x01

Length (1 byte)	ID (1 byte)	Value (1 byte)
0x03	GEN_PID_SUFFIX2_VAL	0x00-0xFF

Command Example

Raw data in hex format (MSB to LSB): 0x00 00 00 12 00 00 00 01 00 0A 01 15 03 F0 01 03 EF 0D

Response Prototype

Length (4 bytes)	Sequence (4 bytes)	Status (4 bytes)	Optional Reason (1 byte)
0x0D	-	0x00/0x01	REASON

Response Values

SUCCESS — Response example: 0x00 00 00 0C 00 00 00 01 00 00 00 00

Failure: — Possible reason byte values in failure response for Scan Data Suffix2 request:

- REASON = 8 battery voltage is low.
- REASON = 22 when barcode application received an invalid command or when more than one command is seen in a request packet.
- REASON = 35 when there is no response from barcode module after 3 seconds timeout.

Failure response example: 0x00 00 00 0D 00 00 00 01 00 00 00 01 16

Barcode Scan Data Transmit Format

Changes the scan data transit format to have a custom prefix and/or suffix1 and/or suffix2 value. The default transmit format is set to “scan data as is”.

Command Prototype

Command # (1 byte)	Length (1 byte)	ID (1 byte)	Value (1 byte)
CMD_EN_XMIT_FMT	0x03	GEN_PID_SCAN_DATA_XMIT_FMT	0x00-0x07

Parameter Values

0x00	Selects scan data transmit format as “Scan data as is”
0x01	Selects scan data transmit format as “Scan data + Suffix1 + Suffix2”
0x02	Selects scan data transmit format as “Scan data + Suffix 2”
0x03	Selects scan data transmit format as “Scan data + Suffix1 + Suffix2”
0x04	Selects scan data transmit format as “Prefix + Scan data”
0x05	Selects scan data transmit format as “Prefix + Scan data + Suffix1 data”
0x06	Selects scan data transmit format as “Prefix + Scan data + Suffix2 data”
0x07	Selects scan data transmit format as “Prefix + Scan data + Suffix1 + Suffix2 data”

Command Example

Raw data in hex format (MSB to LSB): 0x00 00 00 0F 00 00 00 01 00 07 01 16 03 EE 00

Response Prototype

Length (4 bytes)	Sequence (4 bytes)	Status (4 bytes)	Optional Reason (1 byte)
0x0D	-	0x00/0x01	REASON

Response Values

SUCCESS — Response example: 0x00 00 00 0C 00 00 00 01 00 00 00 00

Failure: Possible reason byte values in failure response for Scan Data Transmit Format request:

- REASON = 8 battery voltage is low.
- REASON = 22 when barcode application received an invalid command or when more than one command is seen in a request packet.
- REASON = 35 when there is no response from barcode module after 3 seconds timeout.

Failure response example: 0x00 00 00 0D 00 00 00 01 00 00 00 01 16

Barcode AIM Pattern

Enables or disables AIM pattern. When enabled, the cross-hair pattern for the laser is used during barcode capture. In e355, AIM pattern is shown as LED illumination instead of cross-hair pattern.

Command Prototype

Command # (1 byte)	Length (1 byte)	ID (1 byte)	Value (1 byte)
CMD_EN_AIM_PATTERN	0x03	GEN_PID_EN_AIM_ PATTERN	0x01

Command Example

Raw data in hex format (MSB to LSB): 0x00 00 00 0F 00 00 00 01 00 07 01 17 03 ED 01

Response Prototype

Length (4 bytes)	Sequence (4 bytes)	Status (4 bytes)	Optional Reason (1 byte)
0x0D	-	0x00/0x01	REASON

Response Values

SUCCESS — Response example: 0x00 00 00 0C 00 00 00 01 00 00 00 00

Failure: Possible reason byte values in failure response for AIM Pattern request:

- REASON = 8 battery voltage is low.
- REASON = 22 when barcode application received an invalid command or when more than one command is seen in a request packet.
- REASON = 35 when there is no response from barcode module after 3 seconds timeout.

Failure response example: 0x00 00 00 0D 00 00 00 01 00 00 00 01 16

Barcode Symbology

Configures any supported symbology, including enabling or disabling a symbology and setting symbology parameters (for example, to set length, format, and supplements of a symbology).

Command Prototype

Command for multiple parameters of same symbology

Command # (1 byte)	Length (1 byte)	ID (1 byte)	Value (1 byte)
CMD_SYMBOLOGY	0x03	GEN_PID_EN_CODE128	0x01, 0x00

Length (1 byte)	ID (1 byte)
0x02	SYM_PID_SETLEN_ANY_C128

Command for multiple symbologies and their parameters

Command # (1 byte)	Length (1 byte)	ID (1 byte)	Value (1 byte)
CMD_SYMBOLOGY	0x03	GEN_PID_EN_INTER2OF5	0x01

Length (1 byte)	ID (1 byte)	Value (1 byte)	Length (1 byte)
0x03	SYM_PID_SETLEN_1DISCRETE_I2OF5	0x0E	0x03

ID (1 byte)	Value (1 byte)	Length (1 byte)	ID (1 byte)
SYM_PID_I2OF5_CHECK_DIGIT	0x01	0x03	SYM_PID_EN_CODE93

Value (1 byte)	Length (1 byte)	ID (1 byte)	Value (2 bytes)
0x01	0x04	SYM_PID_SETLEN_RANGE_C93	0x0437

Command Example

Raw data in hex format (MSB to LSB): 0x00 00 00 11 00 00 00 01 00 09 01 18 03 00 01 02 04

or

0x00 00 00 1C 00 00 00 01 00 14 01 18 03 2D 01 03 2E 0E 03 32 01 03 3B 01 04 3E 04 37

Response Prototype

Length (4 bytes)	Sequence (4 bytes)	Status (4 bytes)	Optional Reason (1 byte)
0x0D	-	0x00/0x01	REASON

Response Values

SUCCESS — Response example: 0x00 00 00 0C 00 00 00 01 00 00 00 00

Failure: — Possible reason byte values in failure response for Symbology request:

- REASON = 8 battery voltage is low.
- REASON = 22 when barcode application received an invalid command or when more than one command is seen in a request packet.
- REASON = 35 when there is no response from barcode module after 3 seconds timeout.

Failure response example: 0x00 00 00 0D 00 00 00 01 00 00 00 01 16

Barcode Disable All Symbologies

Disables all symbologies.

Command Prototype

Length (1 byte)	Sequence (4 bytes)	Length (2 bytes)	Protocol Version (1 byte)	Command # (1 byte)
0x0C	-	0x04	0x01	CMD_DISABLE_ALL_SYMB

Command Example

Raw data in hex format (MSB to LSB): 0x00 00 00 0C 00 00 00 01 00 04 01 19

Response Prototype

Length (4 bytes)	Sequence (4 bytes)	Status (4 bytes)	Optional Reason (1 byte)
0x0D	-	0x00/0x01	REASON

Response Values

SUCCESS — Response example: 0x00 00 00 0C 00 00 00 01 00 00 00 00

Failure: Possible reason byte values in failure response for Disable All Symbologies request:

- REASON = 8 battery voltage is low.
- REASON = 22 when barcode application received an invalid command or when more than one command is seen in a request packet.
- REASON = 35 when there is no response from barcode module after 3 seconds timeout.

Failure response example: 0x00 00 00 0D 00 00 00 01 00 00 00 01 16

CHAPTER 3

Barcode Application Logging

By default, Barcode application outputs the basic and error message logging. To enable detailed logging and to look at data packets in Hex, set the configuration variable BARCODEDB=1 in *VTM->Edit Parameters*.

Message Logging

In e315, e335, and e315M terminals, application level logging is enabled by choosing one of these logging options: *DEBUG=4 (OR) *LOG=1024 in *VTM->Edit Parameters* menu (OR) by sending a G31 control command.

In e355 terminal, application level logging is enabled by choosing one of these logging options: *DEBUG=4 (OR) *DEBUG=1 (OR) *LOG=1024 in *VTM->Edit Parameters* menu (OR) by sending a G31 control command.

Barcode application logs starts with string “---BCS:”. For example: “---BCS:Start scan.

CHAPTER 4

Barcode Best Practices

Note the following important information.

- The barcode device is closed by default. To start communications with the barcode module, Barcode device open command needs to be sent first.
- To save battery power, power down the barcode module using command Barcode device close when using barcode module is done.
- The default scan trigger mode is LEVEL. Trigger mode resets to LEVEL whenever terminal restarts.
- By default, continuous scan mode is enabled. Single scan mode is enabled automatically when continuous scan mode is disabled. In single scan mode, scanner shuts off scan session after successful scan of barcode.
- All barcode parameters are restored to their default values when the restore defaults command is called.
- Barcode application expects one command at a time. Next command can't be sent to Barcode application until Success/Failure response of previous command is received.
- Barcode application has a 3 second timeout for each command and returns failure response after expiry of this timeout.
- Barcode module has a power up delay of 1.2 seconds after device open and communications with module are not possible during this delay.
- Barcode commands sequence for turning on the scanner:

  1. Barcode Device Open.
  2. Barcode Trigger mode (for other than LEVEL).
  3. Barcode Start Scan.
     - Scan a barcode
  4. Barcode Stop Scan.
  5. Repeat steps 3 & 4 for multiple transactions.
  6. Barcode Device Close (Applications can choose to not close Barcode device between transactions for avoiding to deal with 1.2 seconds power up delay).

Beeper Tone Definitions

Allows the beeper to generate one of the 96 standard tones at the specified time. The beeper device supports 96 distinct tones designed to approximate eight octaves of the equal tempered musical scale of standard international pitch, with “treble A” having a frequency of 440 Hz. Actual frequencies generated are shown in Table 2 along with the corresponding musical notes and variations. Table 2 reflects a system frequency of 200 MHz, the maximum duration in 10,000 ms or 10 seconds.

The column labels indicate the following characteristics for each of the 96 notes:

- Note – standard “do-re-mi” designation for the musical note.
- N# – the note number, used as a parameter to the sound() function.
- Nominal – frequency in Hertz for the standard musical note.

Table 2 Beeper Tones

Note	N#	Nominal	Note	N#	Nominal
A	0	55.00	A	48	880
A#	1	58.27	A#	49	932
B	2	61.74	B	50	988
C	3	65.41	C	51	1047
C#	4	69.30	C#	52	1109
D	5	73.42	D	53	1175
D#	6	77.78	D#	54	1245
E	7	82.41	E	55	1319
F	8	87.31	F	56	1397
F#	9	92.50	F#	57	1480
G	10	98.00	G	58	1568
G#	11	103.83	G#	59	1661
A	12	110.00	A	60	1760
A#	13	116.54	A#	61	1865
B	14	123.47	B	62	1976
C	15	130.81	C	63	2093
C#	16	138.59	C#	64	2217
D	17	146.83	D	65	2349
D#	18	155.56	D#	66	2489
E	19	164.81	E	67	2637

Table 2 Beeper Tones (continued)

Note	N#	Nominal	Note	N#	Nominal
F	20	174.61	F	68	2794
F#	21	185.00	F#	69	2960
G	22	196.00	G	70	3136
G#	23	207.65	G#	71	3322
A	24	220.00	A	72	3520
A#	25	233.08	A#	73	3729
B	26	246.94	B	74	3951
C	27	261.63	C	75	4186
C#	28	277.18	C#	76	4435
D	29	293.66	D	77	4699
D#	30	311.13	D#	78	4978
E	31	329.63	E	79	5274
F	32	349.23	F	80	5588
F#	33	369.99	F#	81	5920
G	34	392.00	G	82	6272
G#	35	415.30	G#	83	6645
A	36	440.00	A	84	7040
A#	37	466.16	A#	85	7459
B	38	493.88	B	86	7902
C	39	523.25	C	87	8372
C#	40	554.37	C#	88	8870
D	41	587.33	D	89	9397
D#	42	622.25	D#	90	9956
E	43	659.26	E	91	10548
F	44	698.46	F	92	11175
F#	45	739.99	F#	93	11840
G	46	783.99	G	94	12544
G#	47	830.61	G#	95	13290

APPENDIX B

Default Parameter Values

This appendix presents the default parameter values.

Parameter	Define in header	Default
General Scanner Parameters		
Set Trigger Mode	GEN_PID_SET_TRIG_MODE	Level (0x01)
Picklist Mode	GEN_PID_PICKLIST_MODE	Disabled (0x00)
Scan Session Timeout	GEN_PID_SCAN_TIMEOUT	60 seconds in continuous mode (0x3C) 10 seconds in single scan mode (0x0A)
Timeout between scans, same symbol	GEN_PID_TIMEOUT_BW_SAME_SYM	0.6 second (0x06)
Timeout between scans, different symbols	GEN_PID_TIMEOUT_BW_DIFF_SYM	0.2 second (0x02)
Continuous bar code read	GEN_PID_CONTINUOUS_READ	Enabled (0x01)
Unique bar code report	GEN_PID_UNIQUE_CODE_REPORT	Disabled (0x00)
Mobile Phone/Display Mode	GEN_PID_MOBILE_PHONE_MODE	Disabled (0x00)
Prefix key category	GEN_PID_PREFIX_KEY	1 (0x01)
Prefix Value	GEN_PID_PREFIX_VAL	<CR> (0x0D)
Suffix1 key category	GEN_PID_SUFFIX1_KEY	1 (0x01)
Suffix1 Value	GEN_PID_SUFFIX1_VAL	<CR> (0x0D)
Suffix2 key category	GEN_PID_SUFFIX2_KEY	1 (0x01)
Suffix2 Value	GEN_PID_SUFFIX2_VAL	<CR> (0x0D)
Scan data transmission format	GEN_PID_SCAN_DATA_XMIT_FMT	Data as is (0x01)
AIM pattern	GEN_PID_AIM_PATTERN_EN	Enabled (0x01)
Auto beep	GEN_PID_AUTO_BEEP_MODE	Disabled (0x00)
Scan beep	GEN_PID_AUTO_BEEP_SCAN	• Frequency 1–64 msec • Duration 1–50 msec • Pause –50 msec • Frequency 2–50 msec • Duration 2–100 msec
Error beep	GEN_PID_AUTO_BEEP_ERROR	• Frequency 1–58 msec • Duration 1–100 msec • Pause–75 msec • Frequency 2–50 msec • Duration 2–100 msec.

Parameter	Define in header	Default
Symbology Parameters		
Code 128		
Code 128	SYM_PID_EN_CODE128	Enable (0x01)
Set Length(s)	SYM_PID_SETLEN_X_C128	Any Length
GS1-128 (formerly UCC/EAN-128)	SYM_PID_EN_GS1128	Enable (0x01)
ISBT 128	SYM_PID_EN_ISBT	Enable (0x01)
ISBT concatenation	SYM_PID_ISBT_CONCATE	Disable (0x00)
Check ISBT table	SYM_PID_CHECK_ISBT_TABLE	Enable (0x01)
ISBT concatenation redundancy	SYM_PID_ISBT_CONCATE_REDUN	10 (0x0A)
UPC/EAN		
UPC-A	SYM_PID_EN_UPCA	Enable (0x01)
UPC-E	SYM_PID_EN_UPCE	Enable (0x01)
UPC-E1	SYM_PID_EN_UPCE1	Disable (0x00)
EAN-8/JAN 8	SYM_PID_EN_EAN8_JAN8	Enable (0x01)
EAN-13/JAN 13	SYM_PID_EN_EAN13_JAN13	Enable (0x01)
Bookland EAN	SYM_PID_EN_BOOKLAND_EAN	Enable (0x01)
Bookland ISBN format	SYM_PID_EN_ISBN_FORMAT	ISBN-10 (0x00)
Decode UPC/EAN/JAN supplements	SYM_PID_SUPPLIMENTS_UPC_EAN	Disable (0x00)
User-Programmable supplement1	SYM_PID_USER_PROG_SUPP1	N/A
User-Programmable supplement2	SYM_PID_USER_PROG_SUPP2	N/A
UPC/EAN/JAN supplemental redundancy	SYM_PID_UPC_EAN_JAN_SUPP_REDUN	10 (0x0A)
Transmit UPC-A Check Digit	SYM_PID_XMIT_UPCA_CHECK_DIGIT	Enable (0x01)
Transmit UPC-E Check Digit	SYM_PID_XMIT_UPCE_CHECK_DIGIT	Enable (0x01)
Transmit UPC-E1 Check Digit	SYM_PID_XMIT_UPCE1_CHECK_DIGIT	Enable (0x01)
UPC-A Preamble	SYM_PID_XMIT_UPCA_PREAMBLE	System Character (0x01)
UPC-E Preamble	SYM_PID_XMIT_UPCE_PREAMBLE	System Character (0x01)
UPC-E1 Preamble	SYM_PID_XMIT_UPCE1_PREAMBLE	System Character (0x01)
Convert UPC-E to A	SYM_PID_CONVERT_UPCE_2_UPCA	Disable (0x00)
Convert UPC-E1 to A	SYM_PID_CONVERT_UPCE1_2_UPCA	Disable (0x00)
EAN-8/JAN-8 Extend	SYM_PID_EAN8_JAN8_EXTEND	Disable (0x00)
UCC Coupon Extended Code	SYM_PID_UCC_COUPON_EXTEND	Disable (0x00)
Coupon Report	SYM_PID_COUPON_REPORT	New Coupon Symbols (0x01)
ISSN EAN	SYM_PID_ISSN_EAN	Disable (0x00)

Parameter	Define in header	Default
Code 39		
Code 39	SYM_PID_EN_CODE39	Enable (0x01)
Trioptic Code 39	SYM_PID_EN_TRIOPTIC_CODE39	Disable (0x00)
Convert Code 39 to Code 32	SYM_PID_CONV_CODE39_2_CODE32	Disable (0x00)
Code 32 Prefix	SYM_PID_CODE32_PREFIX	Disable (0x00)
Set Length(s)	SYM_PID_SETLEN_X_C39	Length range (0x02–0x37)
Code 39 Check Digit Verification	SYM_PID_CODE39_CHK_DIGIT	Disable (0x00)
Transmit Code 39 Check Digit	SYM_PID_XMIT_CODE39_CHK_DIGIT	Disable (0x00)
Code 39 Full ASCII Conversion	SYM_PID_CODE39_FULL_ASCII	Disable (0x00)
Buffer Code 39	SYM_PID_CODE39_BUFFERING	Disable (0x00)
Interleaved 2 of 5		
Interleaved 2 of 5	SYM_PID_EN_INTER2OF5	Disable (0x00)
Set Length(s)	SYM_PID_SETLEN_X_I2OF5	One discrete length (0x0E)
I 2 of 5 Check Digit Verification	SYM_PID_I2OF5_CHECK_DIGIT	Disable (0x00)
Transmit I 2 of 5 Check Digit	SYM_PID_XMIT_I2OF5_CHECK_DIGIT	Disable (0x00)
Convert I 2 of 5 to EAN 13	SYM_PID_CONV_I2OF5_EAN13	Disable (0x00)
2D, QR Code		
QR Code	SYM_PID_EN_QR_CODE	Enable (0x01)
2D, QR Inverse		
QR Inverse	SYM_PID_EN_QR_INVERSE	Regular (0x00)
2D, MicroQR		
MicroQR	SYM_PID_EN_MICRO_QR	Enable (0x01)
2D, Data Matrix		
Data Matrix	SYM_PID_EN_DATA_MATRIX	Enable (0x01)
Data Matrix Inverse	SYM_PID_EN_DATA_MATRIX_INVERSE	Regular (0x00)
Decode Mirror Images	SYM_PID_EN_MIRROR_IMAGES	Auto (0x02)
Code 93		
Code 93	SYM_PID_EN_CODE93	Disable (0x00)
Set Length(s)	SYM_PID_SETLEN_X_C93	Length range (0x04–0x37)
Code		
Code 11	SYM_PID_EN_CODE11	Disable (0x00)
Set Length(s)	SYM_PID_SETLEN_X_C11	Length range (0x04 – 0x37)
Code 11 check digit verification	SYM_PID_CODE11_CHK_DIGIT	Disable (0x00)
Transmit Code 11 Check Digit(s)	SYM_PID_XMIT_CODE11_CHK_DIGIT	Disable (0x00)

Parameter	Define in header	Default
Discrete 2 of 5		
Discrete 2 of 5	SYM_PID_EN_DISC2OF5	Disable (0x00)
Set Length(s)	SYM_PID_SETLEN_X_D2OF5	One discrete length (0x0C)
Codabar		
Codabar	SYM_PID_EN_CODABAR	Disable (0x00)
Set Length(s)	SYM_PID_SETLEN_X_CBAR	Length range (0x05 – 0x37)
CLSI Editing	SYM_PID_EN_CLSI	Disable (0x00)
NOTIS Editing	SYM_PID_EN_NOTIS	Disable (0x00)
Upper or Lower case Start/Stop char	SYM_PID_START_STOP_CASE	Upper Case (0x00)
MSI		
MSI	SYM_PID_EN_MSI	Disable (0x00)
Set Length(s)	SYM_PID_SETLEN_X_MSI	Length range (0x04 – 0x37)
MSI Check Digits	SYM_PID_MSI_CHK_DIGIT	One (0x00)
Transmit MSI Check Digit	SYM_PID_XMIT_MSI_CHK_DIGIT	Disable (0x00)
MSI Check Digit Algorithm	SYM_PID_MSI_CHK_DIGIT_ALGOR	Mod 10/Mod 10 (0x00)
Chinese 2 of 5		
Chinese 2 of 5	SYM_PID_EN_CHINESE2OF5	Disable (0x00)
MATRIX 2 of 5		
Matrix 2 of 5	SYM_PID_EN_MATRIX2OF5	Disable (0x00)
Set Length(s)	SYM_PID_SETLEN_X_M2OF5	One discrete length (0x0E)
Matrix 2 of 5 Check Digit	SYM_PID_M2OF5_CHK_DIGIT	Disable (0x00)
Transmit Matrix 2 of 5 Check Digit	SYM_PID_XMIT_M2OF5_CHK_DIGIT	Disable (0x00)
KOREAN 3 of 5		
Korean 3 of 5	SYM_PID_EN_KOREAN3OF5	Disable (0x00)
Inverse 1D		
Inverse 1D	SYM_PID_EN_INVERSE1D	Regular (0x00)
Postal Codes		
US Postnet	SYM_PID_EN_US_POSTNET	Disable (0x00)
US Planet	SYM_PID_EN_US_PLANET	Disable (0x00)
Transmit US Postal Check Digit	SYM_PID_XMIT_US_POST_CHK_DIGIT	Enable (0x01)
UK Postal	SYM_PID_EN_UK_POSTAL	Disable (0x00)
Transmit UK Postal Check Digit	SYM_PID_XMIT_UK_POST_CHK_DIGIT	Enable (0x01)
Japan Postal	SYM_PID_EN_JAPAN_POSTAL	Disable (0x00)
Australia Post	SYM_PID_EN_AUSTRALIA_POST	Disable (0x00)
Australia Post Format	SYM_PID_EN_AUST_POST_FMT	Auto discriminate (0x00)

Parameter	Define in header	Default
Netherlands KIX Code	SYM_PID_EN_NETHERLANDS_KIX	Disable (0x00)
USPS 4CB/One Code/Intelligent Mail	SYM_PID_EN_USPS_4CB	Disable (0x00)
UPU FICS Postal	SYM_PID_EN_UPU_FICS	Disable (0x00)
GS1 DataBar		
GS1 DataBar (omni directional, truncated, stacked, stacked omni directional)	SYM_PID_EN_GS1_DATABAR	Enable (0x01)
GS1 DataBar Limited	SYM_PID_EN_GS1_LIMITED	Disable (0x00)
GS1 DataBar Limited Security Level	SYM_PID_LTD_SECURITY	3 (0x03)
GS1 Databar Expanded	SYM_PID_EN_GS1_EXPANDED	Enable (0x01)
Convert GS1 DataBar to UPC/EAN	SYM_PID_EN_CONV_UPC_EAN	Disable (0x00)
Composite		
Composite CC-C	SYM_PID_EN_COMP_CC_C	Disable (0x00)
Composite CC-A/B	SYM_PID_EN_COMP_CC_A_B	Disable (0x00)
Composite TLC-39	SYM_PID_EN_COMP_TLC_39	Disable (0x00)
UPC Composite Mode	SYM_PID_UPC_COMP_MODE	UPC always linked (0x01)
GS1-128 Emulation Mode for UCC/EAN	SYM_PID_EN_GS1_128_EMULATION	Disable (0x00)
Composite codes		
2D, PDF417		
PDF417	SYM_PID_EN_PDF417	Enable (0x01)
2D, MicroPDF417		
MicroPDF417	SYM_PID_EN_MICRO_PDF417	Disable (0x00)
2D, Maxicode		
Maxicode	SYM_PID_EN_MAXICODE	Disable (0x00)
2D, Aztec		
Aztec	SYM_PID_EN_AZTEC	Enable (0x01)
2D, Aztec Inverse		
Aztec Inverse	SYM_PID_EN_AZTEC_INVERSE	Inverse Auto detect (0x02)

Host Header File

The following is an example of a header file.

```
// Beep definitions
typedef struct {
    int freq;
    int dur;
}T_BEEP;
typedef struct {
    T_BEEP b1;
    int bPause;
    T_BEEP b2;
}T_BEEP_DEF;
typedef struct {
    T_BEEP b1;
    int bPause;
}T_BEEP_PAUSE;

///
// Host command IDs
// command IDs are used in host request packets
///
#define CMD_BAR_DEV_OPEN                0x1A
#define CMD_BAR_DEV_CLOSE               0x1B
#define CMD_START_SCAN                  0x01
#define CMD_STOP_SCAN                   0x02
#define CMD_PASS_THRU                   0x03
#define CMD_SINGLE_GET_APP_VER          0x04
#define CMD_SET_TRIG_MODE               0x05
#define CMD_BEEP_IMMEDIATE              0x06
#define CMD_SINGLE_BUTTON_STATUS        0x07
#define CMD_SINGLE_GET_FIRM_VER         0x08
#define CMD_SINGLE_GET_DEVICE_ID        0x09
#define CMD_MULTI_SCAN                  0x0A
#define CMD_RESTORE_DEFAULTS            0x0B
#define CMD_EN_PICKLIST_MODE            0x0C
#define CMD_SCAN_TIMEOUT                0x0D
#define CMD_TIMEOUT_BW_SAME_SYM         0x0E
#define CMD_TIMEOUT_BW_DIFF_SYM         0x0F
#define CMD_EN_CONTINUOUS_RD            0x10
```

```
#define CMD_EN_UNIQUE_CODE_REPO         0x11
#define CMD_EN_MOBILE_PH_MODE           0x12
#define CMD_PREFIX_VAL                  0x13
#define CMD_SUFFIX1_VAL                 0x14
#define CMD_SUFFIX2_VAL                 0x15
#define CMD_EN_XMIT_FMT                 0x16
#define CMD_EN_AIM_PATTERN              0x17
#define CMD_SYMBOLOGY                   0x18
#define CMD_DISABLE_ALL_SYMB            0x19
#define CMD_AUTO_BEEP_CONFIG            0x1C
#define CMD_GET_AUTO_BEEP_CONFIG        0x1D

///
// Parameter IDs
// asterisk (*) in the comments indicate default value
///

//code 128
#define SYM_PID_EN_CODE128              0x00      //value 1 for *enable, 0 disable
#define SYM_PID_SETLEN_1DISCRETE_C128   0x01      //one byte input value
#define SYM_PID_SETLEN_2DISCRETE_C128   0x02      //two bytes input value
#define SYM_PID_SETLEN_RANGE_C128       0x03      //Range is set by two byte input value
#define SYM_PID_SETLEN_ANY_C128         0x04      //*No input value expected for this parameter
#define SYM_PID_EN_GS1128               0x05      //value 1 for *enable, 0 disable
#define SYM_PID_EN_ISBT                 0x06      //value 1 for *enable, 0 disable
#define SYM_PID_ISBT_CONCATE            0x07      //value 1 for enable, 0 *disable, 2 auto
#define SYM_PID_CHECK_ISBT_TABLE        0x08      //value 1 for *enable, 0 disable
#define SYM_PID_ISBT_CONCATE_REDUN      0x09      //value range is 2 to 20, *0x0A
//code UPC
#define SYM_PID_EN_UPCA                           0x0A      //value 1 for *enable, 0 disable
#define SYM_PID_EN_UPCE                           0x0B      //value 1 for *enable, 0 disable
#define SYM_PID_EN_UPCE1                          0x0C      //value 1 for enable,  0 *disable
#define SYM_PID_EN_EAN8_JAN8                      0x0D      //value 1 for *enable, 0 disable
#define SYM_PID_EN_EAN13_JAN13                    0x0E      //value 1 for *enable, 0 disable
#define SYM_PID_EN_BOOKLAND_EAN                   0x0F      //value 1 for *enable, 0 disable
#define SYM_PID_EN_ISBN_FORMAT                    0x10      //value 1 for enabling ISBN-13 &
                                                              0 *ISBN-10
#define SYM_PID_SUPPLIMENTS_UPC_EAN               0x11      //possible values of this parameter
                                                              are given below
   #define IGNORE_UPC_EAN_W_SUPPLIMENTS           *0x00
   #define DECODE_UPC_EAN_W_SUPPLIMENTS           0x01
   #define AUTODESCRIMINATE_SUPPLIMENTS           0x02
   #define ENABLE_378_379_SUPPLIMENTS             0x04
   #define ENABLE_978_979_SUPPLIMENTS             0x05
   #define ENABLE_977_SUPPLIMENT                  0x07
   #define ENABLE_414_419_434_439                 0x06
```

```
    #define ENABLE_491_SUPPLIMENTS              0x08
    #define ENABLE_SMART_SUPPLIMENT             0x03
    #define SUPPLIMENT_USER_PROG1               0x09
    #define SUPPLIMENT_USER_PROG1_2             0x0A
    #define SMART_SUPPLIMENT_USER_PROG1         0x0B
    #define SMART_SUPPLIMENT_USER_PROG1_2       0x0C
#define SYM_PID_USER_PROG_SUPP1                 0x12    //3 digit value input
#define SYM_PID_USER_PROG_SUPP2                 0x13    //3 digit value input
#define SYM_PID_UPC_EAN_JAN_SUPP_REDUN          0x14    //range is 2 to 30, *0x0A
#define SYM_PID_XMIT_UPCA_CHECK_DIGIT           0x15    //value 1 for *enable,0 disable
#define SYM_PID_XMIT_UPCE_CHECK_DIGIT           0x16    //value 1 for *enable,0 disable
#define SYM_PID_XMIT_UPCE1_CHECK_DIGIT          0x17    //value 1 for *enable,0 disable
#define SYM_PID_XMIT_UPCA_PREAMBLE              0x18    //value 0 for No Preamble,1 *system
                                                          char & 2 country code & system char
#define SYM_PID_XMIT_UPCE_PREAMBLE              0x19    //value 0 for No Preamble,1 *system
                                                          char & 2 country code & system char
#define SYM_PID_XMIT_UPCE1_PREAMBLE             0x1A    //value 0 for No Preamble,1 *system
                                                          char & 2 country code & system char
#define SYM_PID_CONVERT_UPCE_2_UPCA             0x1B    //value 1 for enable,0 *disable
#define SYM_PID_CONVERT_UPCE1_2_UPCA            0x1C    //value 1 for enable,0 *disable
#define SYM_PID_EAN8_JAN8_EXTEND                0x1D    //value 1 for enable,0 *disable
#define SYM_PID_UCC_COUPON_EXTEND               0x1E    //value 1 for enable,0 *disable
#define SYM_PID_COUPON_REPORT                   0x1F    //value 0 for old coupon symbols,1
                                                          *new coupon symbol,2 both coupons
#define SYM_PID_ISSN_EAN                        0x20    //value 1 for enable,0 disable
//code 39
#define SYM_PID_EN_CODE39                       0x21    //value 1 for *enable,0 disable
#define SYM_PID_EN_TRIOPTIC_CODE39              0x22    //value 1 for enable,0 *disable
#define SYM_PID_CONV_CODE39_2_CODE32            0x23    //value 1 for enable,0 *disable
#define SYM_PID_CODE32_PREFIX                   0x24    //value 1 for enable,0 *disable
#define SYM_PID_SETLEN_1DISCRETE_C39            0x25    //one byte input value
#define SYM_PID_SETLEN_2DISCRETE_C39            0x26    //two bytes input value
#define SYM_PID_SETLEN_RANGE_C39                0x27    //*Range is set by two byte input
                                                          value,*0x02-0x37
#define SYM_PID_SETLEN_ANY_C39                  0x28    //No input value expected for this
                                                          parameter
#define SYM_PID_CODE39_CHK_DIGIT                0x29    //value 1 for enable,0 *disable
#define SYM_PID_XMIT_CODE39_CHK_DIGIT           0x2A    //value 1 for enable,0 *disable
#define SYM_PID_CODE39_FULL_ASCII               0x2B    //value 1 for enable,0 *disable
#define SYM_PID_CODE39_BUFFERING                0x2C    //value 1 for enable,0 *disable
//Interleaved 2 of 5
#define SYM_PID_EN_INTER2OF5                    0x2D    //value 1 for enable,0 *disable
#define SYM_PID_SETLEN_1DISCRETE_I2OF5          0x2E    //*one byte input value, *0x0E
#define SYM_PID_SETLEN_2DISCRETE_I2OF5          0x2F    //two bytes input value
#define SYM_PID_SETLEN_RANGE_I2OF5              0x30    //Range is set by two byte input
                                                          value
#define SYM_PID_SETLEN_ANY_I2OF5                0x31    //No input value expected for this
                                                          parameter
```

```
#define SYM_PID_I2OF5_CHECK_DIGIT                0x32      //value 1 for enable,0 *disable
#define SYM_PID_XMIT_I2OF5_CHECK_DIGIT           0x33      //value 1 for enable,0 *disable
#define SYM_PID_CONV_I2OF5_EAN13                 0x34      //value 1 for enable,0 *disable
//2D,QR code
#define SYM_PID_EN_QR_CODE                       0x35      //value 1 for *enable,0 disable
//2D,QR Inverse
#define SYM_PID_EN_QR_INVERSE                    0x36      //value 0 *regular,1 Inverse,2 Auto
//2D, MicroQR
#define SYM_PID_EN_MICRO_QR                      0x37      //value 1 for *enable,0 disable
//2D,Data Matrix
#define SYM_PID_EN_DATA_MATRIX                   0x38      //value 1 for *enable,0 disable
#define SYM_PID_EN_DATA_MATRIX_INVERSE           0x39      //value 0 *regular,1 Inverse,2 Auto
#define SYM_PID_EN_MIRROR_IMAGES                 0x3A      //value 0 regular,1 Inverse, 2 *Auto
//Code 93
#define SYM_PID_EN_CODE93                        0x3B      //value 1 for enable, 0 *disable
#define SYM_PID_SETLEN_1DISCRETE_C93             0x3C      //one byte input value
#define SYM_PID_SETLEN_2DISCRETE_C93             0x3D      //two bytes input value
#define SYM_PID_SETLEN_RANGE_C93                 0x3E      //*Range is set by two byte input
                                                             value, *0x04-0x37
#define SYM_PID_SETLEN_ANY_C93                   0x3F      //No input value expected for this
                                                             parameter
//Code 11
#define SYM_PID_EN_CODE11                        0x40      //value 1 for enable,0 *disable
#define SYM_PID_SETLEN_1DISCRETE_C11             0x41      //one byte input value
#define SYM_PID_SETLEN_2DISCRETE_C11             0x42      //two bytes input value
#define SYM_PID_SETLEN_RANGE_C11                 0x43      //Range is set by two byte input
                                                             value,*0x04-0x37
#define SYM_PID_SETLEN_ANY_C11                   0x44      //No input value expected for this
                                                             parameter
#define SYM_PID_CODE11_CHK_DIGIT                 0x45      //value 0 for *disable,1 one check
                                                             digit & 2 two check digits
#define SYM_PID_XMIT_CODE11_CHK_DIGIT            0x46      //value 1 for enable,0 *disable
//Discrete 2 of 5
#define SYM_PID_EN_DISC2OF5                      0x47      //value 1 for enable,0 *disable
#define SYM_PID_SETLEN_1DISCRETE_D2OF5           0x48      //*one byte input value, 0x0C
#define SYM_PID_SETLEN_2DISCRETE_D2OF5           0x49      //two bytes input value
#define SYM_PID_SETLEN_RANGE_D2OF5               0x4A      //Range is set by two byte input
                                                             value
#define SYM_PID_SETLEN_ANY_D2OF5                 0x4B      //No input value expected for this
                                                             parameter
//Codabar
#define SYM_PID_EN_CODABAR                       0x4C      //value 1 for enable,0 disable
#define SYM_PID_SETLEN_1DISCRETE_CBAR            0x4D      //one byte input value
#define SYM_PID_SETLEN_2DISCRETE_CBAR            0x4E      //two bytes input value
#define SYM_PID_SETLEN_RANGE_CBAR                0x4F      //*Range is set by two byte input
                                                             value,*0x05-0x37
#define SYM_PID_SETLEN_ANY_CBAR                  0x50      //No input value expected for this
                                                             parameter
```

```
#define SYM_PID_EN_CLSI                         0x51      //value 1 for enable,0 *disable
#define SYM_PID_EN_NOTIS                        0x52      //value 1 for enable,0 *disable
#define SYM_PID_START_STOP_CASE                 0x53      //value 1 for lower case,0 *upper case
//MSI
#define SYM_PID_EN_MSI                          0x54      //value 1 for enable,0 *disable
#define SYM_PID_SETLEN_1DISCRETE_MSI            0x55      //one byte input value
#define SYM_PID_SETLEN_2DISCRETE_MSI            0x56      //two bytes input value
#define SYM_PID_SETLEN_RANGE_MSI                0x57      //*Range is set by two byte input
                                                            value,*0x04-0x37
#define SYM_PID_SETLEN_ANY_MSI                  0x58      //No input value expected for this
                                                            parameter
#define SYM_PID_MSI_CHK_DIGIT                   0x59      //value 0 for *one check digit,1 two
                                                            check digits
#define SYM_PID_XMIT_MSI_CHK_DIGIT              0x5A      //value 1 for enable,0 *disable
#define SYM_PID_MSI_CHK_DIGIT_ALGOR             0x5B      //value 0 for MOD 10/11 algorithm,1
                                                            *MOD10/MOD10 algorithm
//CHINESE 2 of 5
#define SYM_PID_EN_CHINESE2OF5                  0x5C      //value 1 for enable,0 *disable
//MATRIX 2 of 5
#define SYM_PID_EN_MATRIX2OF5                   0x5D      //value 1 for enable,0 *disable
#define SYM_PID_SETLEN_1DISCRETE_M2OF5          0x5E      //*one byte input value,*0x0E
#define SYM_PID_SETLEN_2DISCRETE_M2OF5          0x5F      //two bytes input value
#define SYM_PID_SETLEN_RANGE_M2OF5              0x60      //Range is set by two byte input value
#define SYM_PID_SETLEN_ANY_M2OF5                0x61      //No input value expected for this
                                                            parameter
#define SYM_PID_M2OF5_CHK_DIGIT                 0x62      //value 1 for enable,0 *disable
#define SYM_PID_XMIT_M2OF5_CHK_DIGIT            0x63      //value 1 for enable,0 *disable
//KOREAN 3 of 5
#define SYM_PID_EN_KOREAN3OF5                   0x64      //value 1 for enable,0 *disable
//INVERSE 1D
#define SYM_PID_EN_INVERSE1D                    0x65      //value 0 for *regular,1 inverse & 2
                                                            inverse auto detect
//POSTAL CODES
#define SYM_PID_EN_US_POSTNET                   0x66      //value 1 for enable,0 *disable
#define SYM_PID_EN_US_PLANET                    0x67      //value 1 for enable,0 *disable
#define SYM_PID_XMIT_US_POST_CHK_DIGIT          0x68      //value 1 for *enable,0 disable
#define SYM_PID_EN_UK_POSTAL                    0x69      //value 1 for enable,0 *disable
#define SYM_PID_XMIT_UK_POST_CHK_DIGIT          0x6A      //value 1 for *enable,0 disable
#define SYM_PID_EN_JAPAN_POSTAL                 0x6B      //value 1 for enable,0 *disable
#define SYM_PID_EN_AUSTRALIA_POST               0x6C      //value 1 for enable,0 *disable
#define SYM_PID_EN_AUST_POST_FMT                0x6D      //value 0 for *auto,1 raw format,2
                                                            alphanum enc,3 num enc
#define SYM_PID_EN_NETHERLANDS_KIX              0x6E      //value 1 for enable,0 *disable
#define SYM_PID_EN_USPS_4CB                     0x6F      //value 1 for enable,0 *disable
#define SYM_PID_EN_UPU_FICS                     0x70      //value 1 for enable,0 *disable
//GS1 DataBar
#define SYM_PID_EN_GS1_DATABAR                  0x71      //value 1 for *enable,0 disable
```

```
#define SYM_PID_EN_GS1_LIMITED                0x72      //value 1 for enable,0 *disable
#define SYM_PID_LTD_SECURITY                  0x73      //value 1 for security level 1,2
                                                          level2,3 *level3,4 level4
#define SYM_PID_EN_GS1_EXPANDED               0x74      //value 1 for *enable,0 disable
#define SYM_PID_EN_CONV_UPC_EAN               0x75      //value 1 for enable,0 *disable
//COMPOSITE
#define SYM_PID_EN_COMP_CC_C                  0x76      //value 1 for enable,0 *disable
#define SYM_PID_EN_COMP_CC_A_B                0x77      //value 1 for enable,0 *disable
#define SYM_PID_EN_COMP_TLC_39                0x78      //value 1 for enable,0 *disable
#define SYM_PID_UPC_COMP_MODE                 0x79      //value 0 for UPC never linked, 1
                                                          *always linked,2 auto
#define SYM_PID_EN_GS1_128_EMULATION          0x7A      //value 1 for enable,0 *disable
//2D, PDF417
#define SYM_PID_EN_PDF417                     0x7B      //value 1 for *enable,0 disable
//2D, MICRO PDF417
#define SYM_PID_EN_MICRO_PDF417               0x7C      //value 1 for enable,0 *disable
//2D, Maxicode
#define SYM_PID_EN_MAXICODE                   0x7D      //value 1 for enable,0 *disable
//2D, Aztec
#define SYM_PID_EN_AZTEC                      0x7E      //value 1 for *enable,0 disable
//2D, Aztec Inverse
#define SYM_PID_EN_AZTEC_INVERSE              0x7F      //value 0 for regular,1 inverse,
                                                          2 *inverse auto detect


//general parameter IDs
// asterisk (*) in the comments indicate default value
#define GEN_PID_PASS_THRU                   0xFF      //reserved value
#define GEN_PID_SET_TRIG_MODE               0xFE      //8 bit value,0 edge,1 *level,2 soft,
                                                        3 passive
#define GEN_PID_BEEP_IMMEDIATE              0xFD      //value starts with no of beeps followed no.
                                                        of T_BEEP_PAUSE
#define GEN_PID_RESTORE_DEFAULTS            0xFC      //value is not required and reserved for
                                                        future
#define GEN_PID_PICKLIST_MODE               0xFB      //value 0 *disabled,1 enabled
#define GEN_PID_SCAN_TIMEOUT                0xFA      //value range 1 to 255 secs,*60 sec in
                                                        continuous,*10 sec in single scan
#define GEN_PID_TIMEOUT_BW_SAME_SYM         0xF9      //value range 0 to 9.9 sec (99 decimal),
                                                        *0.6 sec(0x06)
#define GEN_PID_TIMEOUT_BW_DIFF_SYM         0xF8      //value range 1 to 9.9 sec (99 decimal),
                                                        *0.2 sec(0x02)
#define GEN_PID_CONTINUOUS_READ             0xF7      //value 0 disabled,1 *enabled
#define GEN_PID_UNIQUE_CODE_REPORT          0xF6      //value 0 *disabled,1 enabled
#define GEN_PID_MOBILE_PHONE_MODE           0xF5      //value 0 *disabled,1 enabled
#define GEN_PID_PREFIX_KEY                  0xF4      //value is 1 when host sends prefix value
#define GEN_PID_PREFIX_VAL                  0xF3      //value any 3 digit number
                                                        0-255,*<CR>(0x0D)
#define GEN_PID_SUFFIX1_KEY                 0xF2      //value is 1 when host sends prefix value
```

```
#define GEN_PID_SUFFIX1_VAL                0xF1      //value any 3 digit number
                                                       0-255,*<CR>(0x0D)
#define GEN_PID_SUFFIX2_KEY                0xF0      //value is 1 when host sends prefix value
#define GEN_PID_SUFFIX2_VAL                0xEF      //value any 3 digit number
                                                       0-255,*<CR>(0x0D)
#define GEN_PID_SCAN_DATA_XMIT_FMT         0xEE      //value 0 *data as is, 1 sufix1&2,
                                                       2 sufix2,3 sufix1&2,4 prefix,
                                                       5 Prefix&sufix1,6 prefix&sufix2,
                                                       7 prefix&suffixes
#define GEN_PID_AIM_PATTERN_EN             0xED      //value 0 disabled,1 *enabled
#define GEN_PID_AUTO_BEEP_MODE             0xEC      //value 0 *disabled,1 config scan beep
                                                       only,2 config error beep only,
                                                       3 config both
#define GEN_PID_AUTO_BEEP_SCAN             0xEB      //scan beep value in format T_BEEP_DEF
#define GEN_PID_AUTO_BEEP_ERROR            0xEA      //error beep value in format T_BEEP_DEF

///
// Verix device response to Host
///
#define RESP_ACK                           0x00000000
#define RESP_NAK                           0x00000001
#define RESP_BARCODE_DATA                  0x80000000
#define RESP_BUTTON_STATUS                 0x80000001

///
// Barcode scanned data header definition
///
typedef struct {
char codeID;
char AIMID;
unsigned short int symbology;
}SYMB_INFO;

///
// Code ID value in header of barcode scanned data response
///
#define CODE_ID_UPC_EA              0x01
#define CODE_ID_CODE39_32           0x02
#define CODE_ID_CODABAR             0x03
#define CODE_ID_CODE128_ISBT        0x04
#define CODE_ID_CODE93              0x05
#define CODE_ID_INTL2OF5            0x06
#define CODE_ID_DISC2OF5            0x07
#define CODE_ID_CODE11              0x08
#define CODE_ID_MSI                 0x09
#define CODE_ID_GSI128              0x0A
#define CODE_ID_BOOKLAND_EAN        0x0B
```

```
#define CODE_ID_TRIOPTIC39          0x0C
#define CODE_ID_COUPONCODE          0x0D
#define CODE_ID_GS1DATABAR          0x0E
#define CODE_ID_MATRIX2OF5          0x0F
#define CODE_ID_UCCCOMPOS           0x10
#define CODE_ID_CHINESE2OF5         0x11
#define CODE_ID_KOREAN3OF5          0x12
#define CODE_ID_PDF417_ISSNEAN      0x13
#define CODE_ID_AZTEC_RUNE          0x14
#define CODE_ID_DATA_MATRIX         0x15
#define CODE_ID_QRCODE_MICRO        0x16
#define CODE_ID_MAXICODE            0x17
#define CODE_ID_US_POSTNET          0x18
#define CODE_ID_US_PLANET           0x19
#define CODE_ID_JAPAN_POSTAL        0x1A
#define CODE_ID_UK_POSTAL           0x1B
#define CODE_ID_POSTBAR_CA          0x1C
#define CODE_ID_NETH_KIX            0x1D
#define CODE_ID_AUS_POST            0x1E
#define CODE_ID_USPS_4CB            0x1F
#define CODE_ID_UPU_FICS            0x20
#define CODE_ID_SCANLET_WEB         0x21
#define CODE_ID_CUECAT              0x22


///
//look-up table to translate to code ID character (barcode's industry standard) from given Code ID
//value. Application can choose to convert code ID value to code ID character in front of scanned
//data (OR) can choose to ignore Code ID value and strip it in header of scanned data response
///
const char* symb_code [] = {
 "NA",                                  //0x00
 "A",   //UPC/EAN                       //0x01
 "B",   //Code 39,Code 32               //0x02
 "C",   //Codabar                       //0x03
 "D",   //Code 128, ISBT 12             //0x04
 "E",   //Code 93                       //0x05
 "F",   //Interleaved 2 of 5            //0x06
 "G",   //Discrete 2 of 5               //0x07
 "H",   //Code 11                       //0x08
 "J",   //MSI                           //0x09
 "K",   //GS1-128                       //0x0A
 "L",   //Bookland EAN                  //0x0B
 "M",   //Trioptic Code 39              //0x0C
 "N",   //Coupon Code                   //0x0D
 "R",   //GS1 DataBar Family            //0x0E
 "S",   //Matrix 2 of 5                 //0x0F
```

```
 "T",   //UCC Composite, TLC 39            //0x10
 "U",   //Chinese 2 of 5                   //0x11
 "V",   //Korean 3 of 5                    //0x12
 "X",   //ISSN EAN,PDF417,MacroPDF417      //0x13
 "z",   //Aztec, Azec Rune                 //0x14
 "P00", //Data Matrix                         //0x15
 "P01", //QR Code, MicroQR                    //0x16
 "P02", //Maxicode                            //0x17
 "P03", //US Postnet                          //0x18
 "P04", //US Planet                           //0x19
 "P05", //Japan Postal                        //0x1A
 "P06", //UK Postal                           //0x1B
 "P07", //Postbar CA                          //0x1C
 "P08", //Netherlands KIX code                //0x1D
 "P09", //Australia Post                      //0x1E
 "P0A", //USPS 4CB                            //0x1F
 "P0B", //UPU FICS Postal                     //0x20
 "W",   //scanlet webcode                     //0x21
 "Q "//Cue CAT code                           //0x22
};

///
// AIM ID value in header of barcode scanned data response
///
#define AIM_ID_CODE39_32            0x01
#define AIM_ID_CODE128_ISBT_GS1     0x02
#define AIM_ID_DATAMATRIX           0x03
#define AIM_ID_UPC_EAN_COUPON       0x04
#define AIM_ID_GS1DATABAR           0x05
#define AIM_ID_CODEBAR              0x06
#define AIM_ID_CODE93               0x07
#define AIM_ID_CODE11               0x08
#define AIM_ID_INTL2OF5             0x09
#define AIM_ID_PDF417               0x0A
#define AIM_ID_TLC39                0x0B
#define AIM_ID_MSI                  0x0C
#define AIM_ID_QRCODE_MICROQR       0x0D
#define AIM_ID_DISC2OF5             0x0E
#define AIM_ID_MAXICODE             0x0F
#define AIM_ID_AZTEC_RUNE           0x10
#define AIM_ID_X                    0x11
#define AIM_ID_COMP_EC              0x12
#define AIM_ID_COMP_EE              0x13
#define AIM_ID_COMP_RS              0x14
```

```
///
//look-up table to translate to AIM ID character (barcode's industry standard) from given AIM
//ID value. Application can choose to convert AIM ID value to AIM ID character in front of
//scanned data (OR) can choose to ignore AIM ID value and strip it in header of scanned data
response
///
const char* symb_AIM [] = {
 "NA",
 "A",
 "C",
 "d",
 "E",
 "e",
 "F",
 "G",
 "H",
 "I",
 "L",
 "L2",
 "M",
 "Q",
 "S",
 "U",
 "z",
 "X",
 "E+C",
 "E+E",
 "RS   "
};

///
// 16-bit symbology value in header of barcode scanned data response
///
#define SYM_ID_CODE39                   0x0001
#define SYM_ID_CODABAR                  0x0002
#define SYM_ID_CODE128                  0x0003
#define SYM_ID_D25                      0x0004
#define SYM_ID_IATA                     0x0005
#define SYM_ID_ITF\                     0x0006
#define SYM_ID_CODE93                   0x0007
#define SYM_ID_UPCA                     0x0008
#define SYM_ID_UPCE                     0x0009
#define SYM_ID_EAN8                     0x000A
#define SYM_ID_EAN13                    0x000B
#define SYM_ID_CODE11                   0x000C
#define SYM_ID_MSI                      0x000D
```

```
#define SYM_ID_EAN128                   0x000E
#define SYM_ID_UPCE1                    0x000F
#define SYM_ID_PDF417                   0x0010
#define SYM_ID_CODE39FULL               0x0011
#define SYM_ID_TRIOPTIC                 0x0012
#define SYM_ID_BOOKLAND                 0x0013
#define SYM_ID_COUPONCODE               0x0014
#define SYM_ID_ISBT128                  0x0015
#define SYM_ID_MICROPDF                 0x0016
#define SYM_ID_DATAMATRIX           0x0017
#define SYM_ID_QRCODE               0x0018
#define SYM_ID_POSTNETUS            0x0019
#define SYM_ID_PLANETUS             0x001A
#define SYM_ID_CODE32               0x001B
#define SYM_ID_ISBT128CONC          0x001C
#define SYM_ID_POSTALJAPAN          0x001D
#define SYM_ID_POSTALAUST           0x001E
#define SYM_ID_POSTALDUTCH          0x001F
#define SYM_ID_MAXICODE             0x0020
#define SYM_ID_POSTBARCA            0x0021
#define SYM_ID_POSTALUK             0x0022
#define SYM_ID_MACROPDF417          0x0023
#define SYM_ID_RSS14                0x0024
#define SYM_ID_RSSLIMIT             0x0025
#define SYM_ID_RSSEXPAND            0x0026
#define SYM_ID_SCANLETWEB           0x0027
#define SYM_ID_CUECAT               0x0028
#define SYM_ID_UPCA_2               0x0029
#define SYM_ID_UPCE_2               0x002A
#define SYM_ID_EAN8_2               0x002B
#define SYM_ID_EAN13_2              0x002C
#define SYM_ID_UPCE1_2              0x002D
#define SYM_ID_CCA_EAN128           0x002E
#define SYM_ID_CCA_EAN13            0x002F
#define SYM_ID_CCA_EAN8             0x0030
#define SYM_ID_CCA_RSSEXPAND        0x0031
#define SYM_ID_CCA_RSSLIMIT         0x0032
#define SYM_ID_CCA_RSS14            0x0033
#define SYM_ID_CCA_UPCA             0x0034
#define SYM_ID_CCA_UPCE             0x0035
#define SYM_ID_CCC_EAN128           0x0036
#define SYM_ID_TLC39                0x0037
#define SYM_ID_CCB_EAN128           0x0038
#define SYM_ID_CCB_EAN13            0x0039
#define SYM_ID_CCB_EAN8             0x003A
#define SYM_ID_CCB_RSSEXPAND        0x003B
```

```
#define SYM_ID_CCB_RSSLIMIT          0x003C
#define SYM_ID_CCB_RSS14             0x003D
#define SYM_ID_CCB_UPCA              0x003E
#define SYM_ID_CCB_UPCE              0x003F
#define SYM_ID_KOR3OF5               0x0040
#define SYM_ID_UPCA_5                0x0041
#define SYM_ID_UPCE_5                0x0042
#define SYM_ID_EAN8_5                0x0043
#define SYM_ID_EAN13_5               0x0044
#define SYM_ID_UPCE1_5               0x0045
#define SYM_ID_MACROPDF              0x0046

///
//look-up table to translate to symbology name from given values of symbology name.
//Application can choose to convert symbology name value to symbology name in front
//of scanned data (OR) can choose to ignore symbology name value and strip it in
//header of scanned data response
///
const char* symbology[] = {
 "NA",
 "Code 39",
 "Codabar",
 "Code 128",
 "D25",
 "IATA",
 "ITF",
 "Code 93",
 "UPCA",
 "UPCE",
 "EAN-8",
 "EAN-13",
 "Code 11",
 "MSI",
 "EAN-128",
 "UPCE1",
 "PDF-417",
 "Code 39 FULL ASCII",
 "Trioptic",
 "Bookland",
 "Coupon Code",
 "ISBT-128",
 "Micro PDF",
 "Data Matrix",
 "QR Code",
 "PostnetUS",
 "PlanetUS",
```

```
 "Code 32",
 "ISBT-128 Concat",
 "PostalJapan",
 "PostalAustralia",
 "PostalDutch",
 "Maxicode",
 "PostbarCA",
 "PostalUK",
 "MacroPDF-417",
 "RSS-14",
 "RSS Limited",
 "RSS Expanded",
 "ScanletWebcode",
 "CueCATCode",
 "UPCA+2",
 "UPCE+2",
 "EAN8+2",
 "EAN13+2",
 "UPCE1+2",
 "CompositeCCA+EAN128",
 "CompositeCCA+EAN13",
 "CompositeCCA+EAN8",
 "CompositeCCA+RSSExpand",
 "CompositeCCA+RSSLimit",
 "CompositeCCA+RSS14",
 "CompositeCCA+UPCA",
 "CompositeCCA+UPCE",
 "CompositeCCC+EAN128",
 "TLC39",
 "CompositeCCB+EAN128",
 "CompositeCCB+EAN13",
 "CompositeCCB+EAN8",
 "CompositeCCB+RSSExpand"
 "CompositeCCB+RSSLimit",
 "CompositeCCB+RSS14",
 "CompositeCCB+UPCA",
 "CompositeCCB+UPCE",
 "KOREAN3of5",
 "UPCA+5",
 "UPCE+5",
 "EAN8+5",
 "EAN13+5",
 "UPCE1+5",
 "MacroMicroPDF"
};
```

Verifone, Inc.
1-800-Verifone
www.verifone.com

e315/e355 Barcode Application

Programmers Guide

Verifone Part Number DOC087-070-EN-C, Revision C