Programmer's Reference Manual

P/N 072414

Sabre 1555 RFID & Bar Code Reader

Intellitag RFID by Intermec

Intermec A UNOVA Company

Programmer's Reference Manual

P/N 072414-001

Sabre 1555 PDT Rev. 2.15
RFID Board Rev. 3.10
LCD Board Rev. 1.4

Sabre 1555 RFID & Bar Code Reader

Intellitag
RFID by Intermec

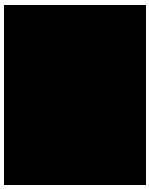
Intermec
A UNOVA Company

Intermec Technologies Corporation
6001 36th Avenue West
Everett, WA 98203-9280

U.S. service and technical support: 1-800-755-5505
U.S. media supplies ordering information: 1-800-227-9947

Canadian service and technical support: 1-800-668-7043
Canadian media supplies ordering information: 1-800-268-6936

Outside U.S.A. and Canada: Contact your local Intermec service supplier.



Contents

Table of Contents

3 Operational Modes 3

4 Interactive Programming Mode 4

List of Figures

List of Tables

1

Before You Begin

This chapter presents the intended audience, topics of discussion, related documentation, and any conventions used in this guide.

Purpose

This programmer's reference manual presents information about configuring and programming the Sabre 1555 RFID & Bar Code Reader. This guide also provides detailed technical specifications of the device.

Intended Audience

This guide was written for people who will be using and programming the 1555 unit. The programming information in Chapter 4, "Interactive Programming Mode," assumes that the user has a working knowledge of computer programming.

Guide Topics

This programmer's reference manual contains the following chapters and appendixes.

Chapter Title	Description
Chapter 1—Before You Begin	Describes the audience, guide topics, and related documentation and software.
Chapter 2—Getting Started	Provides tips for getting started as well as an overview of the functions and features of the 1555 and system configuration.
Chapter 3—Operational Modes	Presents information about the operating modes that the 1555 uses.
Chapter 4—Interactive Programming Mode	Presents the basics for programming the 1555 interactively using a host system.
Appendix A—Acronyms and Glossary	Provides an alphabetical list of abbreviations, acronyms, and terms used in this guide.
Appendix B—Command Value-to-Base 32 Conversion Table	Provides character conversion information.
Appendix C—Data Value-to-Base 64 Conversion Table	Provides character conversion information.
Appendix D—Code 128 ASCII Character-to-Parameter Conversion Table	Provides character conversion information.
Appendix E—System Specifications	Provides technical specifications on the 1555 product.

Related Documentation

The documents that provide operational instructions and to help you get started programming your Sabre 1555 are listed here.

Manual	P/N
Sabre 1555 Laser Scanner & RFID Reader/Programmer Getting Started Guide	3-740049-01
Sabre 1555 Laser Scanner & RFID Reader/Programmer Operator's Guide	3-740049-00

Bar Code Conventions

You can scan the bar codes produced with EasySet configuration software to perform a command. The command bar codes are in the Code 128 symbology. You can also scan bar codes to enter data into your host application. Those can be in any of the supported bar code symbologies. Each bar code includes the name and human-readable interpretation (see Figure 1-1).

Figure 1-1 UPC-A Symbology Used with Sabre 1555 Handheld Unit

Typographical Conventions

Table 1-1 lists typographical conventions that are used in this manual.

Table 1-1 Typographical Conventions

Convention	Indication
	This procedure might cause harm to the equipment and/or the user.
	Concerns about a procedure.
`Code`	Code, including keywords and variables within text and as separate paragraphs, and user-defined program elements within text appear in courier typeface.
Dialog Box Title	Title of a dialog box as it appears on screen.
Function	Start with the characters G4, and are in mixed case with no underscores, and include parentheses after the name, as in G4FunctionName().
Menu Item	Appears on a menu. Capitalization follows the interface.
Note	Auxiliary information that further clarifies the current discussion. These important points require the user's attention. The paragraph is in italics and the word Note is bold.
NUL	Zero-value ASCII character or a zero-value byte.
NULL	Zero-value pointers. Null-terminated string refers to strings of printable ASCII characters with a zero-value byte placed in memory directly after the last printable character of the string.

2

Getting Started

This chapter provides instructions for configuring and using the portable data terminal and standard versions of the Sabre 1555.

1555 Portable Data Terminal and Standard Versions

The 1555 product comes in two versions: portable data terminal (PDT) and standard (Figure 2-1).

Figure 2-1 PDT Version (left) and Standard Version (right)

The PDT version is powered by a rechargeable battery pack. Serial communications for the PDT version is accomplished via a communications cable.

The standard unit requires an external 7.5 ± 0.5 VDC power supply. Serial communications for the standard version is accomplished via a communications cable that branches off from the power cable.

Both versions are available in 915 and 2450 MHz frequencies.

Setting Up the 1555

This section explains the setup for the first use of the 1555.

PDT Version

Before using your 1555 PDT version, you must first charge the battery pack.

Note: *Full charging capacity is reached after three to four uses. A typical charge takes approximately four hours.*

To charge the battery pack

1. Plug the pack into a standard power supply (outlet or power strip). The battery pack light-emitting diode (LED) lights green then changes to red to show that the pack is charging. When the pack is fully charged the LED changes back to green.
2. Unplug the battery pack from its power source and plug it into the base of the 1555 unit.
3. Pull the trigger once. The 1555 emits two audible beeps and displays the current operating mode on its LCD.

   The 1555 is ready to be configured.

Standard Version

To start using your 1555 standard version, plug the power cord into a standard power supply (outlet or power strip) and pull the trigger once. The 1555 emits two audible beeps and displays the current operating mode on its liquid crystal display (LCD). The 1555 is ready to be configured.

Configuring Your Sabre 1555

Before using your Sabre 1555, you must configure it to read from and write to RFID tags and scan laser bar codes. You can set up the 1555 in any of four methods:

- Using EasySet software to print configuration bar codes and read them in bar code read operating mode (referred to as configuration command mode). This method is described in this chapter.
- Using EasySet software to download setup instructions to your 1555 (referred to as configuration command mode). This method is described in this chapter.
- Using combinations of the 1555 mode button, trigger, and LCD screen (referred to as configuration command mode). This method is described in this chapter.
- Controlling the 1555 via a host system (referred to as interactive programming). This method is discussed in detail in Chapter 4.

Configuring the 1555 Using EasySet

Note: *EasySet guides you through a step-by-step approach for printing bar codes or downloading commands to products. Please follow EasySet online instructions for these two configuration methods.*

Starting EasySet

Once you install EasySet System on your PC, you can start EasySet in the following ways:

1. Selecting EasySet from the Start/Programs Menu.
2. Double-clicking on the EasySet icon in the Windows Program Manager.
3. Double-clicking on the *easyset.exe* file in the Windows File Manager.

EasySet starts in general configuration mode. Once you've started EasySet, you're ready to configure your 1555.

1. Move the mouse pointer to the product icon and click to select the 1555 product.
2. Follow the EasySet instructions for printing bar codes or downloading the configuration commands to your 1555.

Configuring the 1555 Using the Mode Button, Trigger, and LCD

You can configure the 1555 using its mode button and trigger (Figure 2-2), and by reading the LCD screen for setup information. This section explains the mode button/trigger combinations and associated LCD menus.

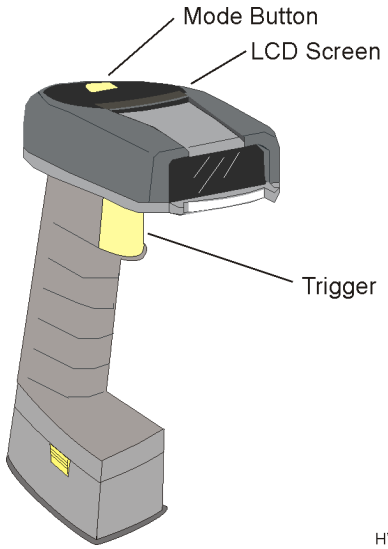


Figure 2-2 Sabre 1555 Features and Components (PDT version shown)

Sabre 1555 Features

This section describes the Sabre 1555 features and how they function in the Setup mode.

Trigger

The trigger has two main functions:

- Activating the currently selected operating mode (e.g., bar code read, tag read)
- Navigating from item to item in the Setup mode

Note: *If you are in Setup mode but not in a setup menu, you can go directly to the Exit Setup option by pressing and holding the trigger until the Exit option displays on the LCD.*

Mode Button

The mode button has four main functions:

- Activating Quick Mode Select to select an operating mode (bar code read, tag read, and other tag options)
- Entering and exiting Setup menus
- Confirming setup choices in the Setup menus, including toggling between deactivate (o) and activate (•) options for some setup commands
- Exiting Setup mode

Mode Button + Trigger

The "mode button + trigger" combination is used to enter Setup mode:

Press and hold the mode button, then press and hold the trigger for two seconds to enter Setup mode.

Note: *If you hold the trigger a second or two longer after releasing the mode button when you enter Setup mode, you will access the Setup menu of the currently selected operating mode, for example, you will access* ***Tag Read settings*** *if the current operating mode is* ***Tag Read****.*

Through the Configuration Inhibit command you can prevent an operator from changing the 1555 configuration (see Table 4-38 on page 4-79).

Quick Mode Select

The Sabre 1555 uses a mode select feature so that you can include only the operating modes you want to use for your application (this simplifies the operation of switching between operating modes).

Note: *Through EasySet you can prevent an operator from entering Quick Mode Setup. By inhibiting LCD setup, you can prevent an operator from viewing any menu except the Data Buffer and View Firmware Version menus.*

Using Quick Mode Select

Use the Quick Mode Select feature to choose an operating mode. You can also customize your Sabre 1555 operating modes by configuring that mode via Setup mode, if the setup is enabled.

To choose an operating mode perform the following steps:

1. Press and hold the Mode button to display the operating modes currently available in Quick Mode Select.

   Note: *If you have a 1555 PDT model and the LCD screen is blank, pull the trigger to activate the product before pressing the Mode button. The Sabre 1555 goes into sleep mode after 40 seconds of inactivity.*

2. Release the Mode button when the operating mode you want to select displays on the LCD screen.

   Note: *If you miss the mode you want to select, hold the Mode button until the desired mode displays again.*

If you cannot find the operating mode you want to select, you will have to add it to Quick Mode Select (see "Customizing Quick Mode Select" on this page).

Customizing Quick Mode Select

You can customize Quick Mode Select to include only the operating modes you want to use for your application (this makes it quicker and easier to switch between operating modes).

1. Press and hold the Mode button, then press and hold the trigger for a second or two until **Setup** displays on the LCD.

   The first Setup option is **Quick Mode Select**. If you do not see this option on the LCD, keep pulling the trigger to cycle through the Setup options until you find **Quick Mode Select**.

2. Press the Mode button to enter the **Quick Mode Select** setup menu.
3. Pull the trigger to cycle through the different operating mode options until you find an item you want to add or remove.
4. Use the Mode button to add or remove the item:

   the • symbol means the operating mode will be enabled to Quick Mode Select

   the o symbol means the operating mode will be disabled from Quick Mode Select

5. Pull the trigger to display the next operating mode and use the Mode button to change the setting if required.
6. When you finish, pull the trigger until you see **Exit** and press the Mode button to exit the **Quick Mode Select** setup menu.

7. Pull the trigger to cycle through the other Setup options or press and hold the trigger to go directly to **Exit Setup**.
8. When you get to **Exit Setup**, press the Mode button two times to exit **Setup**.
9. Press and hold the Mode button to see which operating modes are now available in Quick Mode Select.

Interactive Configuration of the 1555 Using a Host System

With this operational mode, commands pass directly from a host system through the main communications port to the 1555's internal drivers. The operator can control the 1555 remotely from a host system. All generated responses pass directly from the 1555 to the main communications port.

This interactive mode controls the 1555 audiovisual (A/V) indicators, LCD, laser bar code scanner, and RFID reader/programmer module. Figure 2-3 shows an overview of the interactions between the components and the setup and interactive command modes.

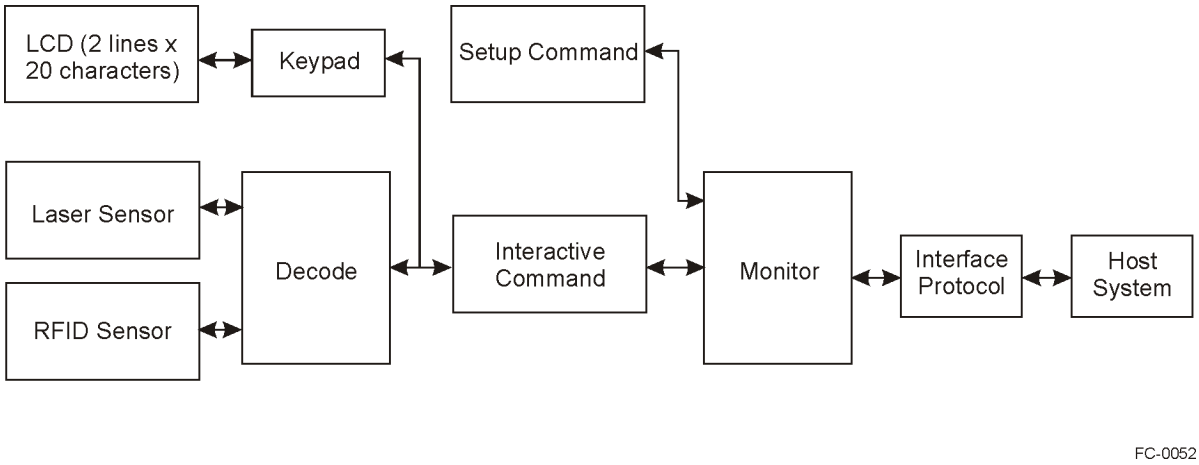


Figure 2-3 Overview of Interactive Operating Mode

Chapter 4, "Interactive Programming Mode," describes the remote control command structures in detail.

3

Operational Modes

This chapter presents information on the operational modes for the Sabre 1555 RFID & Bar Code Reader.

The Sabre 1555 uses two operational modes to set up and control its options: autonomous and interactive. The autonomous mode is discussed in this chapter. The interactive mode is discussed in detail in Chapter 4, "Interactive Programming Mode," of this guide.

Autonomous Mode

You can choose certain setup options that control the Sabre 1555 by using the Autonomous Mode features. You cannot use Autonomous Mode to modify all the setup parameters of the Sabre 1555 (you must use either the EasySet configuration software or setup mode commands (SMC) to access all of the setup options). Table 3-1 lists each Autonomous Mode menu display, its associated option, and a description of that feature/function.

The configurable RFID functions include Tag Identify, Tag Read, Tag Write and Tag Filter. The Tag Read function allows the user to access information previously programmed into the RFID tag storage locations, and then transmit that data to the host system. The Tag Write function allows the user to program the RFID tag with desired information. Tag Identify enables the user to determine how many tags are in the read area of the 1555, and their unique tag IDs are read and can be transmitted to the host system. At any given interrogation point in the customer's RFID tracking process, any single function or a combination of two or three functions may need to be employed. Additionally, the customer may need to employ a "Filter" or "Mask" with the primary RFID function in order to get the desired result. The Filter instructs the 1555 to locate only a desired subset from all the tags in the read area of the 1555, and the Read, Write or Identify function will be performed only with that select subset. The filter or mask functions are primarily used to speed the process of identification of the select tags targeted for interaction with the reader. This speeds the transaction because the filter function effectively eliminates tags that contain data not matching the filter criteria included in the request from the reader. This takes advantage of a function that has been specifically designed into the Intellitag® tag ASIC, which inhibits the tag from participating in the tag sorting process unless data in the tag matches the filter criteria included in the request from the reader.

Table 3-1 Autonomous Mode Options

Setup Screen Display	Setup Menu Option Display	Description
Quick Mode Select	Barcode Read, Tag Identify, Tag Read, Tag Write, Tag RW, Tag Filter, Tag FR, Tag FW, Tag FRW, Exit	Defines which operating modes are available with your Sabre 1555.

Table 3-1 Autonomous Mode Options (continued)

Setup Screen Display	Setup Menu Option Display	Description
Tag Filter Settings	o 1) @ 0 = ???????? o 2) @ 0 = ???????? o 3) @ 0 = ???????? o 4) @ 0 = ???????? Exit	Activates/deactivates from 1 to 4 filter masks (criteria) to isolate desired tags.
Tag Read Settings	• 1) @ 24 Length8 o 2) @ 0 Length0 o 3) @ 0 Length0 o 4) @ 0 Length0 Exit ***Note:*** *The tag read setting for selection 1 @ 24 Length 8 is activated.*	Activates/deactivates from 1 to 4 read fields.
Tag Write Settings	o 1) @ 24 Length0 NO DATA o 2) @ 24 Length0 NO DATA o 3) @ 24 Length0 NO DATA o 4) @ 24 Length0 NO DATA Exit	Activates/deactivates from 1 to 4 write fields.
LCD Settings	Show tag ID: Yes/No Data format: ASCII/hex Contrast + 090% Contrast – 090% Exit	Defines LCD screen options.
Communication	RS-232 (19200,8,N,2) Exit	Displays the desired communication protocol.
Reset	Factory defaults: No/Yes Exit	Global reset of all parameter settings.
Data Buffer	[-----] 000% Store in buffer: No/Yes Send buffer Clear buffer Exit	Data buffer stores tag or bar code ID data in the Sabre 1555. Select enabling or disabling data buffer, send data over serial port, and clearing buffer.
Firmware Versions	*Sabre 1555PDT 2.15* *RFID board 3.10* *LCD board 1.4* Exit	Displays the firmware versions currently loaded in the product.
Exit Setup	Save No save Return to setup	Saves changes and quits Setup mode.

Figure 3-1 shows the options available with the Sabre 1555 reader.

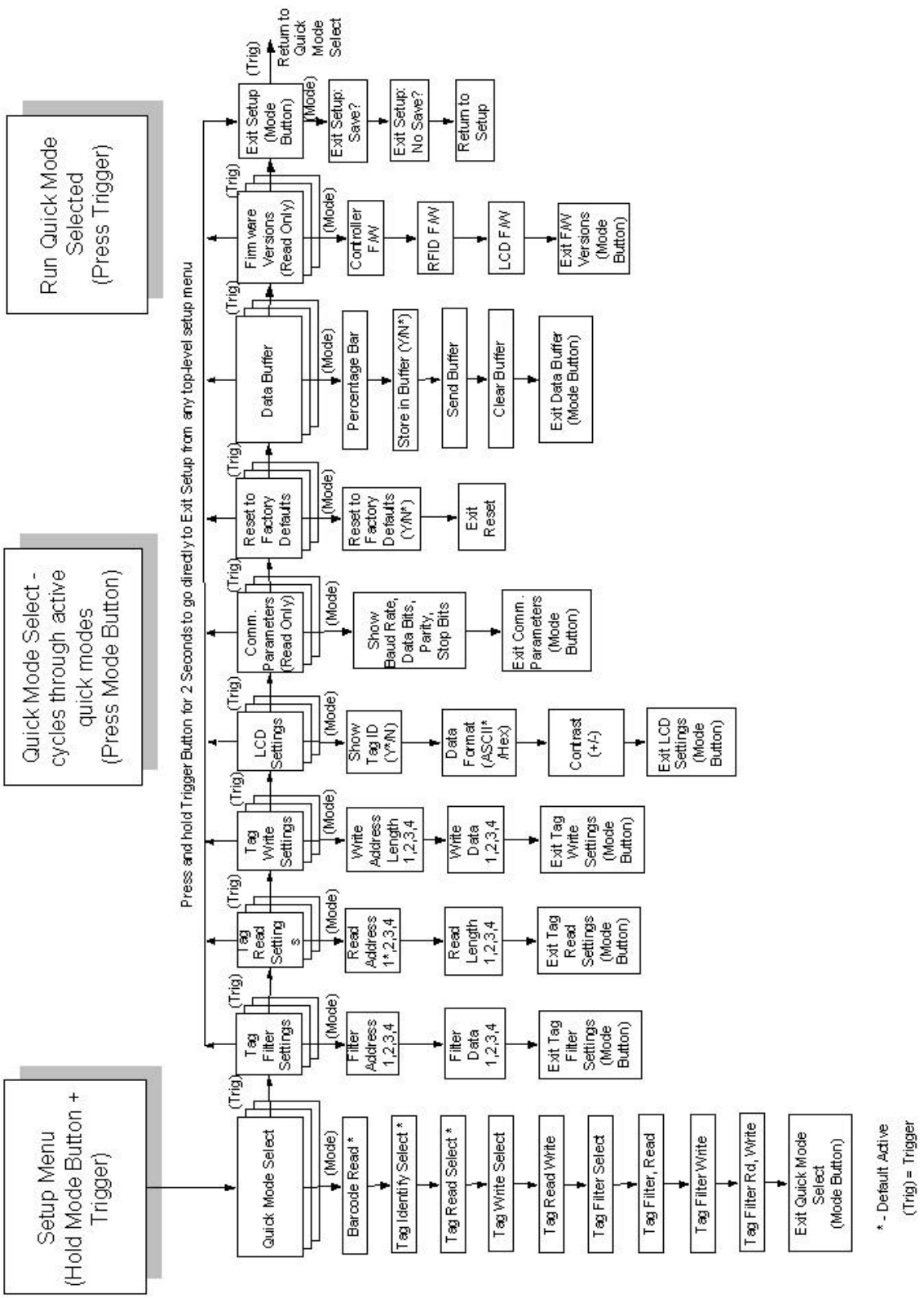


Figure 3-1 Menu Selections Available Through Setup Mode and Quick Mode Select

Quick Mode Select

The Sabre 1555 provides the operating mode selections listed in the following sections.

Bar Code Read

This option reads bar code data.

Tag Identify

This option displays tag IDs of all tags read.

Tag Read

This option reads specified data from tags.

Tag Write

This option writes specified data to tags.

Tag Read/Write (Tag RW)

This option reads specified data from and writes specified data to tags.

Tag Filter

This option enables filter function for tags to be identified.

Tag Filter/Read (Tag FR)

This option enables filter function for tags to be identified and reads specified data from those tags.

Tag Filter/Write (Tag FW)

This option enables filter function for tags to be identified, and writes specified data to those tags.

Tag Filter/Read/Write (Tag FRW)

This option enables filter functions for tags to be identified, and reads specified data from and writes specified data to those tags.

Tag Filter Settings

This feature activates/deactivates from 1 to 4 eight-byte filter masks (filter criteria) to isolate the specific tags you want to identify, read data from, or write data to. The four masks may be set individually or combined with the other masks. For example, you may want to isolate tags with both "Intermec" at position 32 (e.g., mask 1) AND "2000" at position 60 (e.g., mask 4).

Filters are only applied if requested by the selected operating mode. By default, no masks are active (no filtering), and all tags are read.

Use EasySet to define the filter mask parameters for each mask:

- **Start address** — position of the first character the mask applies to (any position in the tag between 0 and 127) (default is 0 for all masks)

- **Comparison operators** — = equal; < > not equal; > greater than; < less than (default is = for all masks)
- **Filter string** — from 1 to 8 characters that you want to match in the tag (maximum is 8 characters depending on the start address, default is ???????? for all masks)

The ? character is an *any character* wildcard in the mask string (one undefined ASCII character is represented by ?, one undefined hex character is represented by ??).

Note: *"Int?rm?c" will match tags with "Int [any_character] rm [any_character] c"*

The filter string can be defined in EasySet as an ASCII string or a hex string. This choice affects how the filter string is displayed in the LCD (single ASCII characters or dual hex values for each character). This filter string display format is independent of the **Data format** selection (ASCII/hex) in the **LCD settings** Setup menu.

Tag Read Settings

This feature activates/deactivates from 1 to 4 read fields (you can combine fields to read from more than one field in the same tag). Field 1 is active by default.

Note: *You can only read from the activated fields if the Tag Read function is activated.*

Use the Tag Filter/Read function if you only want to read from tags that match certain criteria. Use EasySet to define the read parameters for each field:

- **Start address** — position of the first character you want to read (any position in the tag between 0 and 127, default is 24 for field 1 and 0 for the other fields)
- **Field length** — number of characters you want to read (maximum is 32 characters depending on the start address, default is 8 for field 1 and 0 for the other fields)

Tag Write Settings

This feature activates/deactivates from 1 to 4 write fields (you can combine fields to write to more than one field in the same tag). No fields are active by default.

Note: *You can only write to the activated fields if the Tag Write function is activated (see "Autonomous Mode" on page 3-3).*

Use the Tag Filter/Write function if you only want to write to tags that match certain criteria.

Use EasySet to define the write parameters for each field:

- **Start address** — Position of the first character you want to write to (any position in the tag between 24 and 127, default is 24 for all fields)
- **Field length** — Number of characters you want to write (maximum is 32 characters depending on the start address; default is 0 for all fields)

- **Write string** — Characters you want to write in the tag (current maximum length is 32 characters depending on the start address, default length is 0 for all fields and "NO DATA" displayed on the LCD)

LCD Settings

This feature displays the LCD screen options.

- **Show tag ID** — Displays the tag ID of the last tag identified (in all cases, the LCD display indicates the number of tags identified). Default is 'ShowtagID.'
- **Data format** — Displays tag data in ASCII or hex format (ASCII format is more readable but nondisplayable characters are shown as dots (...). Default is ASCII.
- **Contrast** — Feature (**±)** changes the display contrast (default is 90%)

Communication

This feature displays the communication protocol parameters.

Reset

This feature performs a global reset of all parameter settings. It is useful for a first-time setup or if you want to discard previous settings and start a new application.

To reset the Sabre 1555

1. Choose **Yes** from LCD menu.
2. Choose **Exit**. The Sabre 1555 beeps twice to confirm that the reset has been applied.

Note: *The global reset is implemented immediately — you cannot undo it by choosing* ***No Save*** *when you quit Setup mode.*

Reset factory defaults resets the configuration settings to factory settings. After a global reset, you must use EasySet or a custom setup sheet created with EasySet to select the terminal used in your application and customize your setup parameters if required.

Data Buffer

The data buffer is used to store tag or barcode data in the 1555 (mostly used during tetherless batch data collection with the 1555 PDT). It stores all data (including Automatic Identification Manufacturers (AIM) IDs and tag IDs), which would otherwise be sent to the host through a cable. The data buffer capacity depends on the size of the items stored in the buffer — average capacity is around 16,000 items.

The total size for the buffering mode is 256K bytes, but 64K bytes are reserved for buffer management and 192K bytes are used to range the data (tag ID + data). The maximum size of the message entry in the buffer is 32,767 bytes. The total message size (ID + data) must be less than 192K bytes. If message size is greater than this, a full buffer message is displayed.

Data format for each item stored is as follows:

- Bar code scanner: <AIM_ID> <barcode_data>
- RFID reader/programmer: <AIM_ID> <tag_ID> <tag data>

The data buffer commands are as follows:

- **Store in buffer** stores tag or barcode data in the 1555 (if you select **No** and the 1555 is connected to the host with a cable, data will be transmitted directly to the host as it is collected and will not be stored in the data buffer).
- **Clear data buffer** empties the data buffer (reset of the data buffer).
- **Transmit data buffer** sends the data in the data buffer to the host through an RS-232 cable (you will need an adaptor cable (P/N 3634032-01) for PDT models). The buffer remains full until a Clear Buffer command is sent.

For demonstration or test purposes, you can use the terminal window in EasySet to display the contents of the 1555 data buffer.

To display the data buffer contents

1. Connect the 1555 to your PC and set up for online setup (see "Configuring the 1555 Using Easy Set" on page 2-5).
2. In EasySet, select **Terminal** in the **Communication** menu to start the Terminal window.
3. Use the **Send buffer** option in the **Data buffer** menu on the 1555 to copy the contents of the data buffer to the host (when the copy is completed, you can read the contents of the data buffer in the terminal window).

Firmware Versions

This feature lists the various firmware versions that are loaded into your Sabre 1555.

Exit Setup

This feature exits the setup mode and lets you save or discard any changes made during the setup mode interaction.

Main Communications Port Command Set

The 1555 can interpret a considerable number of commands. These commands are received through the unit's main communications port. All commands must use an identical protocol. Figure 3-2 shows the Sabre 1555 operating modes.

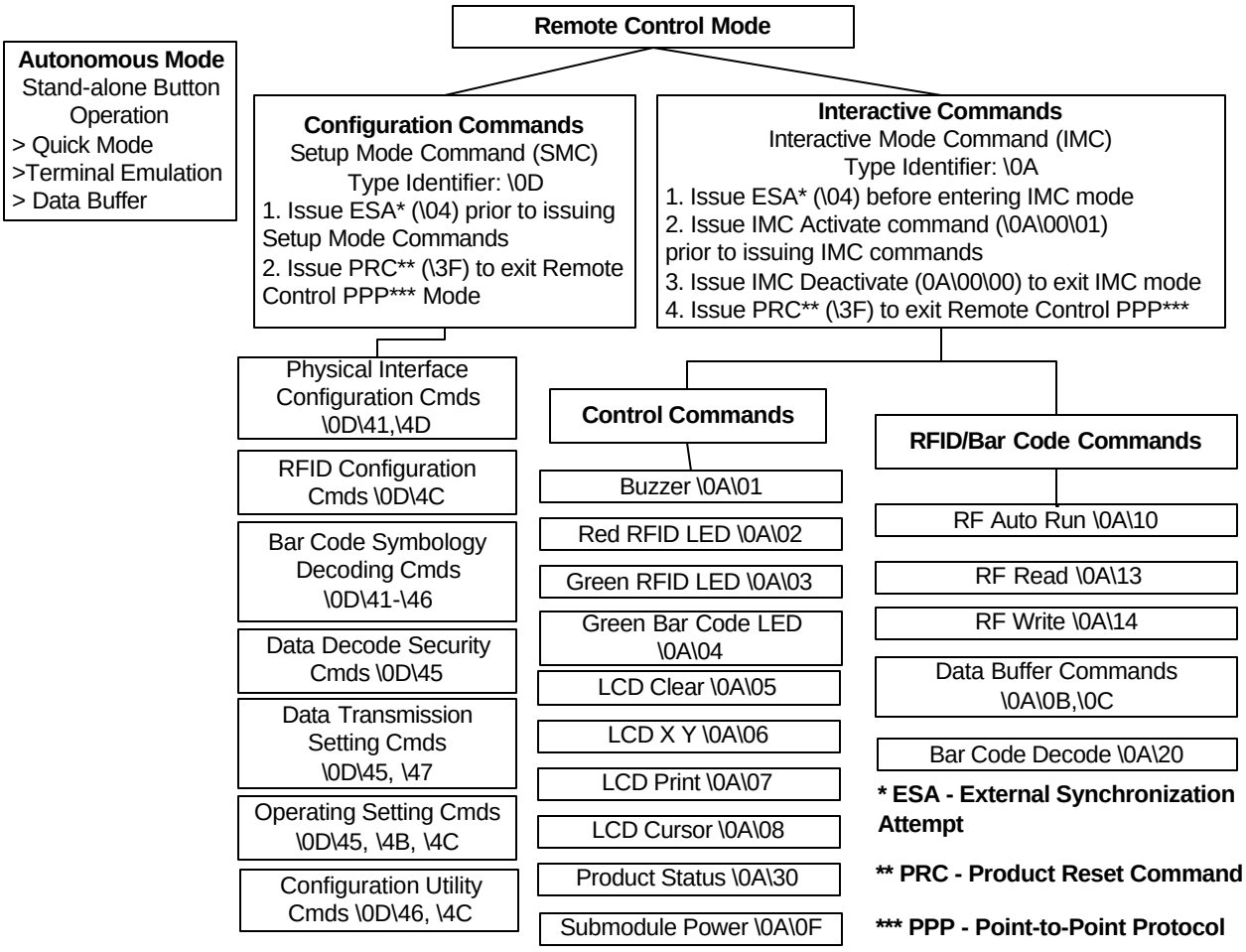


Figure 3-2 Sabre 1555 Operating Modes

Command Types

There are two types of commands based on the command structure used by the Intermec data capture product family. These are configuration commands (those with parameters in nonvolatile memory [NVM]) and interactive mode commands (IMC) (basic functions).

Both types of commands share the same format. The following code shows the required command syntax:

```
<STX><TYPE><CMD/RESP><[PARM/DATA]><Frame_Management><CHK><ETX>
```

"Point-to-Point Protocol" on page 4-5 explains the command formatting in detail.

Table 3-2 shows the command syntax and the code to switch on the 1555's buzzer.

Table 3-2 Command Syntax

Start	Setup Command	Parameters
<IMC>	<buzzer function>	<ON, timeout 100 ms>
0x0A	0x01	0x01, 0x00, 0x64

Configuration Commands

There are currently more than 350 configuration commands for Intermec data capture products (EasySet shows the set of commands for each product, the command string is displayed beneath commands selected for printout).

For the 1555, and in particular for the RFID parameters, additional configuration commands are required (e.g., AIM identifiers for RFID data, custom identifiers for RFID data, Auto Read parameters, and Auto Write parameters).

Interactive Commands

The operator can use the interactive mode to control the 1555 remotely by sending configuration commands and interactive commands through the main communications port.

These commands are used to control the basic functions of the internal peripherals (LCD, buzzer, LED, RAM, and RFID module).

Dialog with the Host System

Commands can be transmitted in different ways according to the context.

The point-to-point protocol (PPP) is an extension of the frame command format ("Frame Format" on page 4-5). This format uses delimiters and a checksum and contains an extended TYPE field for increased application possibilities (e.g., message acknowledgment, message transmission, compressed data, and download applications). Multiple frames are supported for long messages.

Example of a configuration command:

Code 39 Minimum Length configuration (L = 10): \0D\42\5C\0A

Example of an IMC switching the buzzer on:

Buzzer ON for 64 ms: 0A\01\01\00\40

Tag Memory Architecture

The **ASIC** provides 128 bytes of memory, addresses 0-127. The first 12 bytes are factory programmed including an 8-byte unique identification code in address locations 0 through 7 and permanently locked. Future releases of the 1555 product will have the Privileges function enabled, which will occupy locations 12 through 14. The remaining memory can be programmed with customer-specific information. If data is stored in ASCII format, and no other compression technique is employed, then 1 byte of memory will store roughly one alpha-numeric character. Storing in hexadecimal format allows for a greater number of values to be stored since each hexadecimal code can be stored in a single byte. In the majority of applications, most of the data in the reserved locations (including the unique 8-byte Tag ID) is stored in hexadecimal format - as represented with a 2-character nibble per byte. Each byte stored on the ASIC is assigned a location corresponding to the order of the data to be transmitted (locations 0 to 127). These location assignments are used to set up some of the special functions available with the Sabre 1555.

4

Interactive Programming Mode

This chapter presents information for interactive programming and operation of the Sabre 1555 Laser Scanner and RFID Reader/Programmer via a host system.

The information presented in this chapter assumes that the user has a working knowledge of computer programming.

System Requirements

To operate and program the 1555 interactively via a host system, you must have the following equipment:

- Any fixed or handheld computer device with an RS-232 serial port.
- PC (minimum configuration: 166 MHz Pentium, Windows 95, Windows 98, Windows 2000, or Windows NT 4.0 or later; VGA display card; 64 MB of RAM for all operating system; 10 MB or more free hard disk space; and 800 x 600 resolution display) to use with EasySet.
- Sabre 1555 unit (standard or PDT) with power supply.
- RS-232 interface cable (P/N 3-634032-11 for PDT version, P/N 3-614032-00 for standard version).

Communications Interface

The physical interface is the type of cabling used to connect the scanner to a host computer. The interface connection depends on the scanner architecture (RS TTL and RS-232C).

RS-232 TTL Interface for PDT Version

The PDT version of the 1555 product uses a RS TTL (transistor-to-transistor) interface. The parameters for the interface are listed in Table 4-1.

Table 4-1 RS TTL Interface Parameters for PDT Version 9-pin DIN Connector

Pin-out	Signal	Direction	Use
1	VDC	-	5 V
2	Reset	Output to host	Smart cable reset
3	TXD	Output to host	Serial data output
4	RTS	Output to host	Request to send
5	RXD	Input from host	Serial data input

Table 4-1 RS TTL Interface Parameters for PDT Version 9-pin DIN Connector (continued)

Pin-out	Signal	Direction	Use
6	CTS	Input from host	Clear to send
7	Cable detect 1	Input/output	Reserved for software download or cable family detection
8	Cable detect 2	Input/output	Reserved for software download or cable family detection
9	GND	-	Ground

RS TTL Interface for Standard Version

The standard version of the 1555 product uses a RS TTL (transistor-to-transistor) interface. The parameters for the standard version interface are listed in Table 4-2.

Table 4-2 RS TTL Interface Parameters for Standard Version

Pin-out	Signal	Direction	Use
1	VDC	-	5 V
2	-	-	-
3	RESET	Output to host	For example, reset smart cable family
4	TXD	Output to host	Serial data bus to host
5	RTS	Output to host	Request to send
6	RXD	Input from host	Serial data from host
7	CTS	Input from host	Clear to send
8	Cable detect 1	Input/output	Reserved for software download or cable family detection
9	Cable detect 2	Input/output	Reserved for software download or cable family detection
10	GND	-	Ground

These interface connection pin-outs link the product and the cable (the standard port is a 10-position modular plug).

Note: *The connector pin-out to the host depends on the host system hardware interface.*

Baud Rate Parameter

The product input/output (I/O) interface can be configured to operate at 75 to 57600 bps baud rate according to the I/O port characteristics. If you change the current baud

rate by the remote control interface, the new baud rate is applied after you restart the product. The default baud rate for the 1555 is 19200 bps.

Default RS-232 Parameters

The following parameters are factory default settings for the serial interface: 19200 bauds, 8 bits, no parity, 2 stop bits.

Hardware Protocol

This interface can be configured to operate with or without the request to send/clear to send (RTS/CTS) hardware protocol.

The product can send an RTS signal to the host or product and will wait for a CTS from the host or product before sending the data (character by character).

The RTS/CTS protocol can be enabled or disabled. The factory default is disabled.

Point-to-Point Protocol

This section defines the interface command structure, or PPP, which is used to communicate between Intermec products.

Frame Format

The frame format is used with Intermec data capture products to communicate with

- PC applications (e.g., EasySet or customer applications).
- submodules communicating with the main board of the product.

The frame format is defined as follows:

< STX > < TYPE > <CMD/RESP> < [PARM/DATA] > < FRAME_MANAGEMENT > < CHK > < ETX >

Table 4-3 lists the frame format components.

Table 4-3 Frame Format Components

Item	Length	Description
<STX>	1 byte	Delimiter, value of 0x7F
<TYPE>	1 byte	Frame command type
<CMD/RESP> <[PARM/DATA]>	[0 - n] bytes	Setup command or response code, optional parameters or data

Table 4-3 Frame Format Components (continued)

Item	Length	Description
<FRAME_MANAGEMENT >	1 byte	Bits 7, 6 (<More>), indicates that the message is not finished (another frame being sent) Bits 5, 4, and 3 are not used Bit 2 (<AccRequest>), request high-level acknowledgment for the command Bit 1 (<RestartFlag>), product/host restart indicator Bit 0 (<FrameNumber>), frame sequence indicator
< CHK >	2 bytes	The first byte is the most significant byte (MSB) and the second byte is the least significant byte (LSB). Sum with weight calculation (in "Checksum Calculation" on page 4-13)
< ETX >	1 byte	Delimiter, value of 0x7F

Maximum Frame Size

A message consists of one or more frames. For messages with more than one frame, each frame has a maximum size of 255 bytes (STX to ETX).

The maximum size of the message depends on the capacity of the receiver's input buffer.

However, if an application has limited resources, the maximum frame size can be decreased by setup configuration.

Proprietary DLE (Data Link Escape)

To avoid frame desynchronization, proprietary DLE encoding is applied if the following values are encountered inside the formatted frame (not including the STX/ETX delimiters):

STX/ETX (0x7F), DLE (0xEE)

1. The proprietary DLE character (0xEE) is inserted before the ambiguous character.
2. The value of the ambiguous character is increased by 1.

Note: *This method simplifies processing of delimiter localization as the ambiguous character can not be mistaken for a delimiter.*

Example 1 (0x7F in the data)

before:	STX	TYPE	0x70	0x5	0x7F		0x38	CHK	ETX
with DLE:	0x7F	BFD	0x70	0x5	DLE	0x80	0x38	CHK	0x7F

Example 2 (0xEE in the checksum 1)

before:	STX	TYPE	byte1 byte2...byteN	Chk1=0x05		Chk2=0xEE	ETX
with DLE:	STX	TYPE	byte1 byte2...byteN	Chk1=0x05	DLE	Chk2=0xEF	ETX

Frame Fields

STX Delimiter

The PPP is an extension of the protocol used for configuring Intermec data capture products (based on Code 128 Set B and uses the StartB character) bar codes (Table 4-4).

Table 4-4 Start Transmission (STX) Frame Component

Delimiter	Value	Description
STX	0x7F	This character is used as a frame delimiter

TYPE

A value of <TYPE> less than 0x65 corresponds to a setup command (Table 4-5).

Table 4-5 <TYPE> Frame Component (value of 0 to 0x64)

Value of <TYPE>	Size	Description
[0 - 0x64]	1 byte	Indicates the frame type (BFR, SMC, IMC)

Note: *See "<TYPE> Definitions" on page 4-16 for settings.*

<CMD/RESP> <[PARM/DATA]>

Commands and optional parameters/data depending on the associated command (Table 4-6).

Table 4-6 <CMD/RESP> <[PARM/DATA]> Frame Component

Value	Size	Description
[0 - 0xFF]	[0 - n] bytes	Depending on the frame type, <CMD/RESP> <[PARM/DATA]> can be one of the following: • setup commands • response codes • command parameters for setup commands • command parameters for IMCs • data

FRAME_MANAGEMENT (FM)

FM includes a field named <More> that is used to indicate that the message is not finished (another frame is being sent or the message consists of more than one frame). FM also includes a bit field named ACCRequest that is used to request a high-level acknowledgment from the receiver of the frame. The ACCRequest field can be configured in the product, for BFR frames only, by a setup command (Table 4-7).

Table 4-7 FM Component

<FM> byte		
<More>	**Size**	**Description**
b7, b6	2 bits	Used to indicate if more than one frame is expected Value = 0: the message is encoded in a single frame Value = 1: the frame is not the last frame in a multiframe message Value = 2: the frame is the last frame of a multiframe message ***Note:*** *High-level frame acknowledgment is only possible when all of the frames in the message have been transmitted (<More> is set to 0 or 2).*
b5	1 bit	Always set to 0
b4 - b3	2 bits	Reserved
<AccRequest>	**Size**	**Description**
b2	1 bit	Encoded in bit 2 of FM Value = 0: no high-level frame acknowledgment required Value = 1: high-level frame acknowledgment is required from the receiver of the frame (only for ACC and NAC). This does not affect other responses. ***Note:*** *The ACCRequest bit will always be set to 0 in the high-level frame acknowledgment.*
<RestartFlag>	**Size**	**Description**

Table 4-7 FM Component

<FM> byte		
b1	1 bit	Used to indicate a power-on or restart Receiving a frame with <RestartFlag> set to 1 means that the product has just been restarted Examples: • This flag can be used to check if a desynchronization indicated by the FrameNumber bit is due to product/host restart (the FrameNumber returned should be set to the same value as the FrameNumber received, normally 0) • This flag can also be used to inform the host of a restart in IMC for example (any volatile commands may need to be resent)
<FrameNumber>	**Size**	**Description**
b0	1 bit	The value of this bit must alternate between 0 and 1 for different consecutive frames to detect incorrect frame transmission Probable causes of incorrect frame transmission are as follows: • Lost low-level ACK • ACK sent too late (timing error, the time-out needs to be changed by the corresponding setup command) • Desynchronization due to product/host restart (see <RestartFlag>)

Table 4-8 Frame Management

b2 (hi Level Ack Requested)	**b1 (RST)**	**b0 (Fr #)**	
1	0	0	= 04 1st Frame (odd)
1	0	1	= 05 2nd Frame (even)
1	1	0	= 06 Restart.

Example: 06, 05, 04, 05...

CHK (Checksum)

A checksum (CHK) is a calculated value that is used to test data integrity. Table 4-9 lists the CHK components.

Table 4-9 CHK Frame Component

Value	Size	Description
[0 - 0xFFFF]	2 bytes	First byte = b15 - b8 Second byte = b7 - b0 (see explanation in "Checksum Calculation" on page 4-13)

ETX Delimiter

The PPP is an extension of the protocol used for bar code setup of Intermec data capture products (based on Code 128 Set B and uses the Code 128 Stop character). Table 4-10 lists the ETX components.

Table 4-10 End Transmission (ETX) Frame Component

Delimiter	Value	Description
ETX	0x7F	This character is used as a frame delimiter.

Frame Acknowledgment

Each frame received is checked for the following identifiers:

- STX/ETX delimiters
- Frame number
- Frame type
- Checksum

A frame-received acknowledge character (e.g., ACK, NAK) is returned accordingly ().

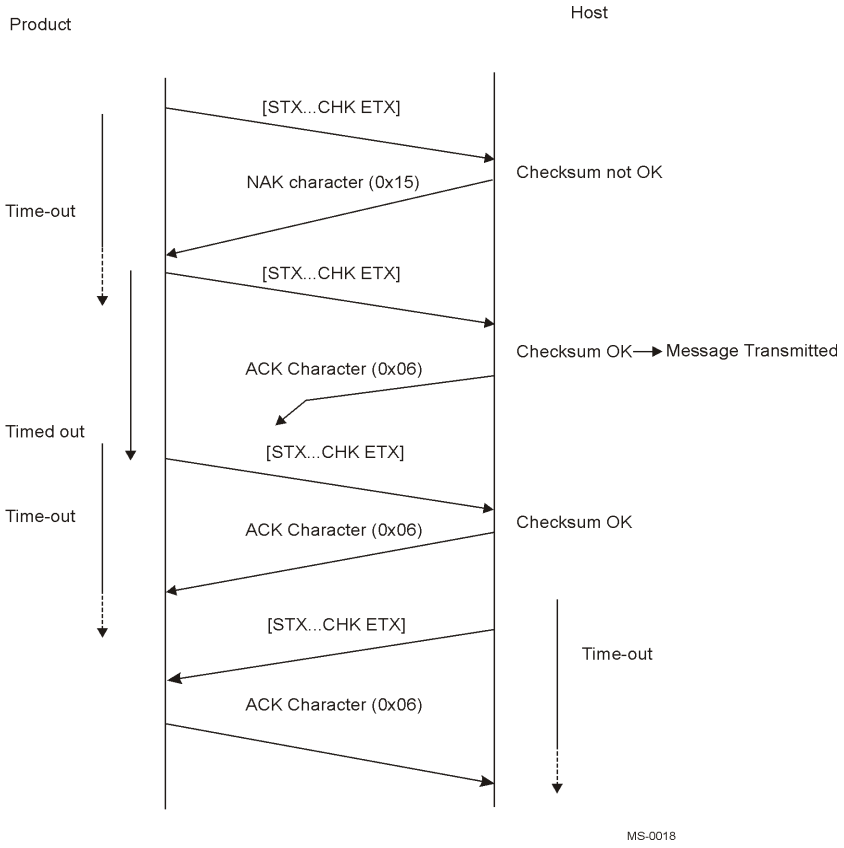


Figure 4-1 Frame-received Acknowledgment Process Flowchart

The frame is sent again if no ACK/NAK character is received before the end of the time-out (100 ms by default, value can be modified by setup command), or if a NAK character is received.

Note: *A maximum of two resend attempts are possible for each frame.*

In the following example, the host sends a *deactivate power-up beeps* setup command to the reader.

```
Reader <== <STX> <SMC> <Power_beeps_off> <FM>   <CHK>     <ETX>
           0x7F  0x0D    0x45, 0x5A     0x00  0x01,0xB7   0x7F
```

The reader has received the command, performed the action, and sent a low-level ACK character, but the ACK character is lost.

```
Host <=X= ACK (ACK character lost)
          0x06
```

The host does not receive the low-level ACK and resends the same frame after the time-out.

```
Reader <== <STX> <SMC> <Power_beeps_off> <FM>    <CHK>     <ETX>
           0x7F  0x0D     0x45, 0x5A    0x00   0x01,0xB7   0x7F
```

The reader has already received this frame (Frame number = 0 in FM byte). The reader sends a low-level ACK and does not repeat its action.

```
Host <== ACK (ACK character received)
```

Implementation Rules

- The first message sent (or when you want to resynchronize after a restart) must begin with the <RestartFlag> set to "1" and the <FrameNumber> set to "0".
- The <FrameNumber> must be incremented each time a **different frame** is transmitted.
- If no low-level acknowledge is received from the product before the low-level timeout expires, the host can resend the same frame up to two more times with the **same frame number**.
- To reduce collisions to a minimum, it is necessary to check before transmission if a frame is being received. In this case, wait until the complete frame is received, acknowledged, and processed before starting the new transmission.
- If the low-level response is NAK, this indicates that the corresponding frame was incorrect:

  1. Bad checksum. This occurs when a character in the frame is lost or if the proprietary DLE mechanism is not integrated in the checksum itself.

  2. Incorrect frame number. If this is the case, you must resend the same frame with the "Restart flag" to 1 to resynchronize the product with the host.

- Host-to-product acknowledgment timeouts:

  The low-level and high-level acknowledgments must be sent before their respective timeouts expire in the product (default timeouts are 100 mS and 500 mS).

- Product-to-host acknowledgment timeouts

  The wait time for the high-level acknowledge from the product (ACC, NAC, ASI . . .) depends on the time required for internal communication with the submodule(s) or execution of the internal setup command task.

  The timeout for the "ACC" response from the product is short (less than 100 mS) for the IMC barcode decode function. It can be considerably longer (up to 5 seconds) for other functions (e.g., IMC "data buffer clear" command).

  The timeout for an ASI response from the RF submodule can be a maximum of 1 second. If there is no response after this period, the ASI response will be sent to the host with parameter 0x00 to indicate "no response from submodule."

  The high-level acknowledge timeout for setup commands can be a maximum of 3 seconds depending on the setup command function.

Checksum Calculation

The checksum is a calculated value that is used to test data integrity. Errors can occur when data is transmitted or when it is written to disk. One means of detecting such errors is the use of a checksum. A value is calculated for a given block of data by sequentially combining all the bytes of data with a series of arithmetic or logical operations. After the data is transmitted or stored, a new checksum is calculated and compared with the original one. If the checksums match, the transmission or storage was ensured to be virtually error free. If they do not match, an error occurred.

To determine the frame's checksum value

1. Calculate the weighted sum of each value in the frame except STX, CHK, and ETX (weight decreases by one for each successive value).
2. Calculate the modulo 65536 on the result of step 1.
3. Assign the hex value obtained (checksum) formatted over two bytes (CHK1 = most significant byte, followed by CHK2 = least significant byte).
4. Apply the proprietary DLE if required on the data, CHK1, and CHK2, but not on the delimiters.

Table 4-11 shows an example for DATA = *12345678*.

Table 4-11 Frame Example with DATA = 12345678

Frame	STX	Type	Data								FM	CHK	ETX
Hex Value	0x7F	0x01	0x31	0x32	0x33	0x34	0x35	0x36	0x37	0x38	0x01	0x8E7	0x7F

Table 4-12 shows the checksum calculation for that data.

Table 4-12 Checksum Calculation for DATA = 12345678

Frame	Type	Data								FM
Hex Value	0x01	0x31	0x32	0x33	0x34	0x35	0x36	0x37	0x38	0x01
Weight	10	9	8	7	6	5	4	3	2	1
Weight * Value	0x0A	0x109	0x190	0x165	0x138	0x109	0xd8	0xa5	0x70	0x01

	CHK1	CHK2
Σ =	**1st byte**	**2nd byte**
0x8E7	0x08	0xE7

The following code describes the checksum calculation:

```
'will calculate the checksum for the data input
'returns the checksum as a 4 digit string (4 nibbles, 16bits)
'sIn is a String (ASCII) representation of the hex value
'The return value is also a string (ASCII) representation of the
hex value


Public Function ChecksumCalc(By Val sIn as string) as string
    Dim cmd as string
    Dim TmpStr  as string
    tmpCMDStr as string
    Dim sCksum As String
    Dim sByte As String
    Dim nWeight As Integer
    Dim i As Integer
    Dim nLen As Integer
    Dim lSum As Long
    ChecksumCalc = ""

‘sIn = the hex string that contains the complete command and
TYPE character.
‘For example, if you want to issue the IMC ON command,
sIn =  “0A00010102”
‘Note the 02 is the frame management byte


    nLen = Len(sIn) / 2
    nWeight = nLen
    lSum = 0
    For i = 0 To nLen - 1
        sByte = Mid(sIn, (i * 2) + 1, 2)
```

```
        lSum = lSum + nWeight * Val("&H" & sByte & "&")
        nWeight = nWeight - 1
    Next i
    sCksum = Hex(lSum)
    If Len(sCksum) > 4 Then
        sCksum = Mid(sCksum, 1, 4)
    End If

    If Len(sCksum) < 4 Then
        If Len(sCksum) = 3 Then sCksum = "0" & sCksum
        If Len(sCksum) = 2 Then sCksum = "00" & sCksum
    End If
    ChecksumCalc = sCksum

‘the correct checksum for the IMC On command is: CheckSumVal  =
”0039”

cmd = “0A00010102” &  CheckSumVal

'check for special DLE characters in command string
TmpStr = ""
tmpCMDStr = ""
For x = 1 To Len(cmd) Step 2
       TmpStr = Mid(cmd, x, 2)
        If TmpStr = "7F" Then tmpCMDStr = tmpCMDStr & "EE80"
        If TmpStr = "EE" Then tmpCMDStr = tmpCMDStr & "EEEF"
        If TmpStr <> "7F" Or TmpStr <> "EE" Then tmpCMDStr =
tmpCMDStr & TmpStr
Next x

cmd = ""
cmd = tmpCMDStr
cmd = "7F" & cmd & "7F"

‘the complete command with checksum and the start and end
 characters added is as follows:
‘cmd = “7F0A0001010200397F”}
```

Note: *Specification AFNOR Z63.300/EN 799/USS describes the checksum calculation method for Code 128.*

PPP Frame Type

This section details the various frame types used in communicating with the 1555.

<TYPE> Definitions

The <TYPE> field indicates the frame type (setup command, binary message, high-level acknowledge, and other parameters). A message consists of one or more frames.

Table 4-13 describes bits b7 through b0.

Table 4-13 <TYPE> Bit, Value, and Description

Bit	Value	Description
b7	0	Always set to 0 for <TYPE> commands
b6	0	Always set to 0 for <TYPE> commands
b5 - b0	[1 - 63]	This value indicates the frame type

In Table 4-14 (frames 1 and 2), the message sent to the host is a Binary Frame Response (BFR) type, where message data = 357 bytes and MFS = 229.

Table 4-14 Binary Frame Data Examples

Frame 1	STX	<Type> (e.g., BFR)	Size 1	Size 2	Data	CHK	ETX
	0x7F	0xC1 (bit 6 set to 1)	0x1	0x85	Byte 1 through byte 233	(CHK)	0x7F
Frame 2	**STX**	**<Type> (e.g., BFR)**	**Data**	**CHK**	**ETX**		
	0x7F	0x81 (bit 6 set to 0)	byte 224 - byte 357	(CHK)	0x7F		

Bits < b5 b4 b3 b2 b1 b0 > are described in Table 4-15.

Table 4-15 <TYPE> Definition Identifiers

Identifier	Value	Description	Page Locator
BFR	0x01	Binary frame response command	Page 4-17
NAD	0x02	Numerical ASCII data command	Page 4-18
ASI	0x03	Accepted PPP command and sending information	Page 4-18
ESA	0x04	External synchronization attempt	Page 4-18
ACC	0x06	Accepted command	Page 4-21
IEC	0x07	IMC error command	Page 4-21
DAP	0x09	Download application program	Page 4-21
IMC	0x0A	Interactive mode command	Page 4-21
SMC	0x0D	Setup mode command	Page 4-21
PEV	0x0E	Product event	Page 4-21
NAC	0x15	Not accepted (command not recognized or not executed)	Page 4-22
PVI	0x16	Product version identification	Page 4-22
RLM	0x19	Resend last message (more than one frame)	Page 4-22
PRC	0x3F	Product reset command	Page 4-23

Binary Frame Response (BFR = 0x01)

BFR commands are used to send decoded bar code data/RFID tag data to the host (ASCII characters and binary data). Each byte has a value of between 0 and 255.

To avoid desynchonization, proprietary data link escape (DLE) encoding is applied if the following values are encountered in the formatted frame (not including the STX/ETX delimiters):

STX/ETX (0x7F), DLE (0xEE)

1. The proprietary character (0xEE) is inserted before the ambiguous character is inserted.
2. The value of the ambiguous character is increased by 1.

Note: *This method simplifies the delimiter localization processing because the ambiguous character cannot be mistaken for a delimiter.*

Example 1 (STX/ETX in the data, and TYPE = BFR)

before:

```
<STX>    <TYPE>     <CMD/RESP> <[PARM/DATA]>  <FM>  <CHK>          <ETX>
0x7F      0x01       0x70, 0x5, 0x7F, 0x38       0xXX, 0xXX        0x7F
```

w/ DLE:

```
<STX> <TYPE> <CMD/RESP><[PARM/DATA] ><DLE> <CMD/RESP><[PARM/DATA]><FM> <CHK> <ETX>
0x7F    0x01          0x70, 0x5        0xEE       0x30*, 0x38       0x00, 0x00  0x7F
```

Note: **CMD/RESP [PARM/DATA] (0x7F) is incremented by 1 to (0x80)*

Example 2 (0xEE in the checksum, TYPE = BFR)

before:

```
<STX>    <TYPE>   byte1, byte2...byteN  <CHK> <FM> <CHK>   <ETX>
0x7F      0x01     0xXX, 0xXX...0xXX     0x05      0xEE    0x7F
```

with DLE:

```
<STX>  <TYPE>  byte1, byte2...byteN <FM> <CHK> DLE  <CHK>  <ETX>
0x7F     0x01     0xXX, 0xXX...0xXX   0x05 0xEE   0xEF*   0x7F
```

Note: **CHK (0xEE) is incremented by 1 to 0xEF*

Numerical ASCII Data (NAD = 0x02)

Compressed numerical ASCII data value (each byte has a numerical ASCII value). This frame type has the same function as BFR.

Accepted PPP Command and Sending Information (ASI = 0x03)

This command indicates that the command has been accepted and is running or has completed. Information concerning the command will be returned.

The second byte (i.e., after the ASI field) in the returned message indicates the initial command request type (refer to ESA command in Table 4-15).

External Synchronization Attempt (ESA = 0x04)

This command is used to synchronize communication with a product and switch to the Remote Control Protocol. The product can then be controlled by the host (IMC and SMC commands). To return to the previous interface, the product must be restarted with a PRC command (ESA is a volatile state).

By using the ESA command, the operator can use EasySet to set up the 1555 directly without having to read a special bar code. EasySet sends a PRC to restart the 1555 when the user quits EasySet (the product reverts to its previous communication parameters if they have not been modified in the EasySet session).

If the product is supplied by a battery with an energy-saver mechanism, this energy-saver mechanism is suspended during the ESA session until the product receives a PRC command.

How to Use the ESA Command

The host must send the ESA command repeatedly with each RS-232 parameter in the order shown in Table 4-16 until the 1555 responds with a low-level ACK character, unless the RS-232 communication parameters in the product are already known. In that case, the host can issue an ESA with those known communication parameters.

Table 4-16 Order of ESA Communication Messages

Baud Rate	Bits	Parity	Stop Bit
57600	8	Even	2
	8	Odd	2
	8	None	2
	7	Even	2
	7	Odd	2
	7	None	2
38400	8	Even	2
	8	Odd	2
	8	None	2
	7	Even	2
	7	Odd	2
	7	None	2
19200	8	Even	2
	8	Odd	2
	8	None	2
	7	Even	2
	7	Odd	2
	7	None	2
9600	8	Even	2
	8	Odd	2
	8	None	2
	7	Even	2
	7	Odd	2
	7	None	2
4800	8	Even	2
	8	Odd	2
	8	None	2
	7	Even	2
	7	Odd	2
	7	None	2
2400	8	Even	2
	8	Odd	2
	8	None	2
	7	Even	2
	7	Odd	2
	7	None	2
...	...	...	...

Two parameters must be set in the ESA command:

- Hardware handshaking protocol if applicable (default: no CTS/RTS); and
- Desired baud rate for communication (e.g., 57600 bauds).

For example, in the ESA command, the host requests communication at 57600 bauds. The product is limited to 19200 bauds and its current speed is 9600 bps, so synchronization occurs at 9600 bauds. As the ASI response returned by the product indicates that it cannot communicate at a baud rate higher than 19200, host and product then switch to 19200 bauds

The syntax without STX and ETX is as follows:

<ESA><HPR><BRR><FM><CHK>

<ESA> External Synchronization Attempt (0x04)

<HPR> Hardware Protocol Requested [0 - 2]
0 = not used;
1 = Host CTS/RTS;
2 = Product CTS/RTS

<BRR> Baud Rate Requested
[0 - 12] = [75, 150, 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200, 230400]

<FM> Frame Marker
bit M = 0;
bit ACK = 0;
bit reset = 1;
frame number = 0

The response is as follows:

<ACK> character

<ASI> <ESA> <HPC> <BRC> <CFG> <FM> <CHK>

<ASI> Accepted PPP Command and Sending Information (0x03)

<ESA> External Synchronization Attempt (0x04)

<HPC> Hardware Protocol Confirm
[0 - 2]
0 = not used;
1 = Host CTS/RTS;
2 = Product CTS/RTS

<BRC> Baud Rate Confirm
[0 - 12] = [75, 150, 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200, 230400]

<CFG> Configuration
b0 = setup parameters in NVM;
b1 = remote communication without beep;
b3, b2 reserved;
b7 - b4 must be set to 0.

<FM> Frame Marker

Once synchronization has been achieved with the product using ESA command, it is necessary for the host to change communications parameters to the following: "Baud_Rate_Requested (BRR), 8, N, 2". In other words, although the baud rate is specified in the ESA command as a parameter, the number of data bits, parity and stop bits

are fixed to 8, N, 2, using the ESA command, regardless of the settings for these parameters prior to issuing the ESA command.

For example: If synchronization between the host and reader was achieved at "9600,7,E,1", but the host issues an ESA command with a requested baud rate of 19200, then it is necessary for the host to change communications parameters to "19200,8,N,2", not "19200,7,E,1". When the host issues the Product Reset Command (PRC) causing the product to exit ESA mode, it is required that the host change the communication parameters back to the values used prior to synchronization, unless, the RS-232 parameters were altered using setup commands while in ESA. If the parameters are altered using a SMC while in synchronization, it is necessary for the host to communicate using these new RS-232 parameters subsequent to exiting ESA mode.

Accepted Command (ACC = 0x06)

Indicates that the command has been accepted and is running or has been completed.

Interactive Mode Command Error Command (IEC = 0x07)

Indicates that the IMC is not active.

Download Application Program (DAP = 0x09)

DAP programs are used to download user application programs from the host to the product and to upload user application programs from the product to the host.

Interactive Mode Command (IMC = 0x0A)

IMC is one of the remote control protocol command types that is used to control the product through the physical interface. A special IMC must be sent to activate this mode. See “Interactive Mode Commands (IMC = 0x0A)” on page 4-30 for detailed IMC structure.

Setup Mode Command (SMC = 0x0D)

The SMC is used to configure the product’s parameters stored in NVM.

Product Event (PEV = 0x0E)

This command is generated by the product if an internal event occurs (e.g., time-out, trigger status, etc.). This command is only returned in interactive mode.

Table 4-17 Product Event Codes and Descriptions

PEV Number	Description
0x10	Battery discharged
0x20	RFID AutoRun function time-out occurred
0x21	Bar Code decoder time-out occurred
0x30	Trigger has been released
0x31	Trigger has been pulled
0x32	Button has been released
0x33	Button has been pushed
0x40	RFID overheated and RF power switched off
0x50	RFID submodule power off
0x60	Reserved for GPIO

Not Accepted Command (NAC = 0x15)

NAC is followed by a one-byte parameter indicating the type of problem encountered (Table 4-18).

Table 4-18 NAC Parameters and Descriptions

Parameter Value	Description
0	Command or message rejected
1	Setup command not recognized
2	Type parameter or setup parameter not recognized
3	No electrical power to submodule (if submodule is destination)
4	Command not applicable in current context (e.g. the submodule is already in use)
5	<TYPE> of the command not recognized

Product Version Identification (PVI = 0x16)

The PVI command is used to obtain the P/N and firmware version of the product or a subpart of the product (Table 4-19).

The syntax is as follows:

<STX> <**PVI**> <**peripheral number**> <FM> <CHK> <ETX>

Table 4-19 Product Version Identification Peripherals

Peripheral Number	Peripheral
0	Main board of the product
1	Bar code engine
2	RFID engine
4 - 63	Extension

The product response is as follows:

<ACK>

<STX> <**ASI**> <PVI> <**"string"** ><FM> <CHK> <ETX>

where "string" format = "board_part_number firmware_revision_number"

For example, "Sabre 1555PDT 2.08C."

A space character (0x20) is used to separate each field. If the peripheral number does not exist, a NAC response is returned.

Resend Last Message (RLM = 0x19)

RLM requests that the last message be resent.

Product Reset Command (PRC = 0x3F)

PRC restarts the product, e.g., to quit temporary remote control mode initiated by the ESA command.

The syntax is as follows:

<STX> <PRC> <FM> <CHK> <ETX>

The response is as follows:

<ACK>

<STX> <ACC> <FM> <CHK> <ETX>

Application Examples

Binary Frame Example

Send binary frame => STX, BFR, 6, 1, 5, 3, 9..., CHK, ETX

Reply <= “frame-received” ACK

Reply <= STX, ACC, CHK, ETX (message data OK)

Send => “frame-received” ACK

Multiple Frame Message

Send 1st frame => STX, BFR, Size 1, Size 2, D_0, D_1, D_2...D_n, More = 1, CHK, ETX

Reply <= ACK character

2nd and last frame => STX, BFR, D_{n+1}, D_{n+2}...D_{n+n}, More = 0, CHK, ETX

Reply <= ACK character

Reply <= STX, ACC, CHK, ETX (message data OK)

Send => ACK character

Typical Host Frame Communication Algorithms

To reduce collisions to a minimum (more efficient), it is necessary to check before transmission if a frame is being received. In this case, wait until the complete frame is received, acknowledged, and processed before continuing with the new transmission. If there is a break in the character flow, the frame will be silently discarded (the time between characters must not exceed the transmission duration of the character).

```
void Host_main ()
 byte Tx_buffer [Tx_MAX]
 byte Rx_buffer [Rx_MAX]
 byte MaxRetry = 3
byte Retry_counter = 0
 RestartFlag set to one
 External_Synchronization attempt () /* try to connect to the product */
 DO
   IF (There_is_data_to_send) THEN
     IF (Host_Txframe (Tx_buffer, Retry_counter)) THEN
       There_is_data_to_send is cleared
         TX_FrameNumber is incremented
         Retry_counter is cleared
       ELSE
           IF (Retry_counter = MaxRetry)
             RestartFlag set to one /* bit of Frame_MANAGEMENT byte */
             Retry_counter is cleared
             External_Synchronization attempt () /* use ESA command */
           END IF
     END IF
   IF (frame_received) THEN
     IF (Host_Rxframe ())
        frame_received bit is cleared
WHILE (TRUE)
```

Sample algorithm for command transmission procedure

```
boolean Host_Txframe(Data_to_send, Retry_counter)
 IF ( no_frame_reception )
   transmit STX character
   WHILE ( last_character_not_transmitted )
       verify DLE mechanism
       SendChar( current character )
       calculate checksum on current character
   END WHILE
   transmit Frame Management character
   calculate checksum on current character
   transmit checksum
   transmit ETX character
   IF ( Host_Wait_ low-level_Acknowledge () )
     RETURN (TRUE)
   ELSE
     Retry_counter is incremented
 END IF
 RETURN (FALSE)
```

Sample algorithm for command reception procedure

```
boolean Host_Rxframe ()
  byte Retry_counter
  byte Buffer[MaxSizeResponse]
   IF (CheckPPP () == checksum_received) THEN
   |  IF ( FrameNumber is expected FrameNumber) THEN
   |  |   SendChar (Low-level ACK)
   |  |   Rx_data = frame data
   |  |  There_is_data_Received = TRUE /* Process received data */
   |  |   IF (Current frame is BFD frame and AccRequest is required) THEN
   |  |   |     Buffer = High-level ACC response
   |  |   |    Retry_counter is cleared
   |  |   |    DO
   |  |   |    |   IF (Host_Txframe (Buffer, Retry_counter))
   |  |   |    |       RETURN (TRUE)
   |  |   |    WHILE (Retry_counter <> MaxRetry)
   |  |   |    RestartFlag set to one
   |  |   |    RETURN (TRUE)
   |  |      END IF
   |  |  ELSE IF (FrameNumber = previous FrameNumber
   |  |   |                        and
   |  |   |          Checksum = previous Checksum)
   |  |   |  SendChar (low-level ACK) /* duplicate frame, do not resend to application */
   |  |   ELSE
   |  |      SendChar (low-level NAK)
   ELSE
      SendChar (Low-level NAK)
   RETURN (FALSE)
```

Sample algorithm for Low-Level acknowledgment procedure

```
boolean Host_Wait_ low-level_Acknowledge ()
        start low-level response timeout
        DO
            CASE (character was received)
              ACK: RETURN (TRUE) /* transmission successful */
              NAK: RETURN (FALSE) /* retry */
        WHILE (timeout not expired)
        RETURN (FALSE) /* frame lost */
```

Lite Acknowledgment Option

This option removes the need to send an ACC upon receiving a BFR command.

For example, for a product communicating with an *intelligent* internal submodule, the internal submodule can be configured using the AccRequest setup command to instruct the product not to send an ACC frame or NAC frame when it receives a BFR frame.

Each frame sent by the submodule must then have the AccRequest bit in the <FM> field set to zero to indicate to the receiver of the message (not the product in the example shown in this section) that the ACC/NAC response is not required.

For example,

The binary frame example (“Application Examples” on page 4-23) would be reduced to sending the BFR frame and only receiving an ACK character.

Send binary frame => STX, BFR, 6, 1, 5, 3, 9..., CHK, ETX

Reply <= ACK character

Note: *If this option is used when sending commands to the product, e.g., setup commands without high-level acknowledge (AccRequest bit = 0 in FRAME_MANAGEMENT field), it may be necessary (depending on the command) to wait a few seconds before sending another command to allow the product to process the previous command.*

Remote Control Interface

This communication interface is used to control the product remotely from the host system. shows the interactions between the components and the setup and interactive command modes.

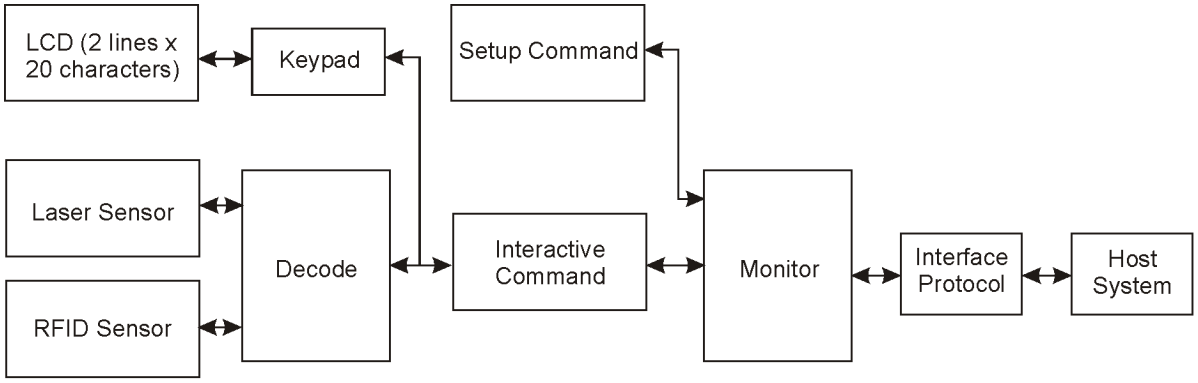


Figure 4-2 Remote Control Command Interactions

When the product is communicating in unpacketed RS232 format, it can receive directly packeted setup commands or the ESA command.

Packeted setup commands can be sent to the product if you know the product's current RS232 communication settings are known. Each setup command is acknowledged by a low-level ACK/NAK followed by a high-level acknowledge ACC/NAC following execution of the command if requested by the host (recommended). However, the results of successful barcode/RFID actions will be sent to the host in unpacketed format without acknowledgment from the host.

The ESA command is used to auto-connect to a product without knowing the current RS232 communication settings (autodetect loop) and switches temporarily to RC Protocol communication (until power off or reset command). In this mode, setup commands can be sent to the product or the product can be controlled by IMC commands (if IMC mode is then activated).

To put the product into RC Protocol mode (the communication is always in packet format), the product must be configured with the special "Remote Control Protocol - framed RC Protocol (19200, 8, N, 2)" command (read setup barcode obtained with EasySet software or send command through RS-232).

Command Types

There are two types of remote control commands:

- **Interactive Mode Commands** — IMCs are used to control the product directly from the host.

Note: *You must send an ESA command and* ***IMC Activation*** *command before performing subsequent IMCs.*

- **Setup Commands** — these commands are used to configure the product directly from the host application (without using EasySet or reading configuration bar codes).

Note: *You must send an ESA command before performing a setup command.*

Interactive Mode Commands (IMC = 0x0A)

Each (IMC has a *function number* parameter corresponding to the desired function, which may have one or more parameters.

Example: < IMC > < buzzer function ><ON, time-out 80 ms>
Command: 0x0A 0x01 0x01, 0x00, 0x10, 0x50

Interactive commands are used to control product functions remotely. Interactive commands can use parameters sent with the command and/or parameters already set in the product (e.g., default settings, remote control setup commands).

Table 4-20 shows the IMC frame examples.

Table 4-20 IMC Frame Examples

Format	<frametype>	<IMCfunction>	[functionparameters]
Example 1	IMC	IMCactivation	ON
Hex Values	0A	00	01
Size	1 byte	1 byte	1 byte
Example 2	IMC	buzzer	ON + 100 ms
Hex Values	0A	01	0064
Size	1 byte	1 byte	1 byte

Note: *Interactive commands are not activated by default, the special IMC activation function must be sent to enable IMC functions.*

Note: *In IMC, your Sabre 1555 remains active, which means it is drawing current. (The current consumption information is listed in Table E-3, "Sabre 1555 Current Use with RF Power On," on page E-4 of this manual).*

Example: The host sends an RF_Auto_Run command with a 10 s time-out parameter (if the product is in RFID Write Mode, the tag write is performed using the write address and write string already programmed in the product, see "RFID Auto Configuration Commands" on page 4-46). Table 4-21 lists the format for the IMCs.

Table 4-21 Interactive Mode Command Format

Command Name	IMC Activation	<TYPE> <CMD>: \0A\00 + Parameters (Prms)
Description	Activate/deactivate IMC mode (IMC must be activated if you want to use IMC commands).	
Parameters	[0 - 1]; 0: Deactivate IMC mode; 1: Activate IMC mode	
Responses	ACC (command accepted and started) NAC (command not recognized)	

The following commands represent the current interactive functions.

Audio/Visual Function

Table 4-22 lists the commands that control audio/visual (A/V) functions interactively.

Table 4-22 A/V Function Commands

Function Name	Buzzer	<TYPE> <CMD>: \0A\01 + Prms
Description	This function activates/deactivates the buzzer with 2 modes: - ON/OFF - switch off after time-out	
Parameters	ON: \01\nn\nn OFF: \00\00\00 Time-out: \01\nn\nn in ms [1 - 1000] (hex format) e.g., Buzzer ON with time-out of 100 ms: \0A\01\01\00\64	
Responses	ACC (command accepted and started) NAC (command not recognized, or parameter out of range)	
Function Name	Red_RFID_LED	<TYPE> <CMD>: \0A\02 + Prms
Description	This function activates/deactivates the red RFID error LED: - ON/OFF - switch off after time-out	
Parameters	ON: Timeout in milliseconds: \01\nn\nn Timeout in seconds: \02\00\nn OFF: \00\00\00 \0A\02\01\nn\nn Timeout in ms [1 - 1000] (hex format) \0A\02\02\00\nn Timeout in seconds [1 - 60] (hex format)	
Responses	IEC (IMC not activated) ACC (command accepted and started) NAC (command not recognized, or parameter out of range)	

Table 4-22 A/V Function Commands (continued)

Function Name	Green_RFID_LED	<TYPE> <CMD>: \0A\03 + Prms
Description	This function activates/deactivates the green RFID success LED: - ON/OFF - switch off after time-out	
Parameters	ON: Timeout in milliseconds: \01\nn\nn Timeout in seconds: \02\00\nn OFF: \00\00\00 \0A\03\01\nn\nn Timeout in ms [1 - 1000] (hex format) 0A\03\02\00\nn Timeout in seconds [1 - 60] (hex format)	
Responses	IEC (IMC not activated) ACC (command accepted and started) NAC (command not recognized, or parameter out of range)	
Function Name	Green_BC_LED	<TYPE> <CMD>: \0A\04 + Prms
Description	This function activates/deactivates the green bar code good read LED: - ON/OFF - switch off after time-out	
Parameters	ON: \01\nn\nn (ms); \02\00\nn (sec) OFF: \00\00\00 Time-out: \0A\04\01\nn\nn in ms [1 - 1000] (hex format) Time-out: \0A\04\02\00\nn in seconds [1 - 60] (hex format)	
Responses	IEC (IMC not activated) ACC (command accepted and started) NAC (command not recognized, or parameter is out of valid range)	
Function Name	LCD_CLEAR	<TYPE> <CMD>: \0A\05
Description	This function clears the product display window	
Parameters	None	
Responses	IEC (IMC not activated) ACC (command accepted and started) NAC (command not recognized)	

Table 4-22 A/V Function Commands (continued)

Function Name	LCD_XY	<TYPE> <CMD>: \0A\06 + Prms
Description	This function sets the cursor to the position X, Y in the product display window	
Parameters	X: is the horizontal position value [1 - 20] Y: is the line number [1 - 2] e.g., XY (10,1) = 0A\06\0A\01	
Responses	IEC (IMC not activated) ACC (command accepted and started) NAC (command not recognized or parameter is out of valid range)	
Function Name	LCD_Print	<TYPE> <CMD>: \0A\07 + Prms
Description	This function inserts a string at the current LCD-XY cursor position in the product display window	
Parameters	Use ASCII characters to create the desired string for a single line. If the string sent goes past the 20th character, only the characters that fit on the line of the display can be displayed	
Responses	IEC (IMC not activated) ACC (command accepted and started) NAC (command not recognized)	
Function Name	LCD_Cursor	<TYPE> <CMD>: \0A\08 + Prms
Description	This function activates/deactivates the type of cursor in the product display window	
Parameters	OFF: \00 Normal cursor: \01 Flashing cursor: \02	
Responses	IEC (IMC not activated) ACC (command accepted and started) NAC (command not recognized)	

Table 4-23 shows the submodule power-on/power-off function command.

Table 4-23 Submodule Power-on/Power-off Function

Function Name	Submodule Power-on/Power-off	<TYPE> <CMD>: \0A\0F + Prms
Description	This function requests power-on/power-off of a product's submodule n if applicable (the product is considered the submodule(s) controller). This function must be sent to the controller module before using an IMC function if the submodule requires activation. When the submodule is powered up by the power-on command, the submodule is initialized.	
Parameters	Peripheral ID number, 1 byte [1]: 1 = RFID submodule Mode [0 - 1]; 1 byte: 0 = deactivate the submodule; 1 = activate the submodule	
Responses	IEC (IMC not activated) ACC (command accepted and started) NAC (command not recognized) ***Note:*** *If the product is powered by battery and a low-battery status is detected, the product will send a PEV (product event) frame:* *PEV frame: <PEV><Event number>, with event number = battery discharged (0x10)*	

RFID Function

RFID function commands configure the RFID portion of the Sabre 1555 parameters. Table 4-24 lists the RFID function commands.

Table 4-24 RFID Function Commands

Function Name	RF_Auto_Run	<TYPE> <CMD>: \0A\10 + Prms
Description	This function launches the auto function(s) with parameters that are already present in the product NVM parameters zone (see "RFID Auto Configuration Commands" on page 4-46). The auto function(s) can be single or multifunction, e.g., identify, read, write, filter, or a combination of these depending on the current operating mode (see "RFID Auto Configuration Commands" on page 4-46).	
Parameters	Mode [0 -1]: 0 = Read until time-out; 1 = Stop after one success or time-out Time-out [1 - 30] e.g., Mode 1, 10 s: \0A\10 \01\00\0A	

Table 4-24 RFID Function Commands (continued)

Responses	**AutoRun ON, continuous operation:** An ASI frame + n BFR frame(s) is sent. - ASI frame format: <ASI><IMC><RF_Auto_Run><current operating mode number><Success flag> 0x03 0x0A 0x10 0xnn partial success = 0; complete success = 1 (current "RFID operating mode selection" and success flag are defined in the "RFID Auto Configuration Commands" on page 4-46). ("Success flag" indicates if the data obtained corresponds to a complete success or partial success with a read/write status to check which data is valid. With complete success, the read/write status of the BFR frame is always set to the value 0xFF) After each success, a BFR frame is sent to the host. - n BFR frames corresponding to the identified tags are sent under the following format each time that a new tag is identified: <BFR><[Preamble]><AIM_ID><Read/Write status><Tag_ID><[Data]><[Postamble]> **AutoRun OFF:** An ACC frame is sent after the AutoRun function is stopped. - ACC frame format: **<**ACC> **AutoRun ON + time-out:** An ASI frame is sent: - ASI frame format: <ASI><IMC><RF_Auto_Run ><current operating mode number ><success flag> After each success, a BFR frame is sent to the host. - n BFR frames corresponding to the tags identified during the session are sent with the following format: <BFR><[Preamble]><AIM_ID><Read/Write status><Tag_ID><[Data]><[Postamble]> When the time-out expires, a PEV frame (Product EVent) is sent. - PEV frame format: <PEV>< Event number> with event number = RF_Auto_Run_Timeout (0x20) **Read status** is represented by the low nibble of the Read/Write status byte [b3 - b0]. Each bit is a flag indicator of successful Auto Read operation, bit 0 corresponds to field 1, bit 3 corresponds to field 4. If a field bit is not set and the corresponding field is activated, this means that the data of this field is incorrect. **Write status** is represented by the high nibble of the Read/Write status byte [b7 - b4]. Each bit is a flag indicator of successful Auto Write operation, bit 4 corresponds to field 1, bit 7 corresponds to field 4. If a field bit is not set and the corresponding field is activated, this means that the data of this field is incorrect. For example, if you have selected 3 read fields (1, 2, 3) and the read status is 0x05 (0101), this indicates that field 2 is invalid (the data for field 2 will be included in the data response but will not be valid).

Table 4-24 RFID Function Commands (continued)

Responses (continued)	***Note:*** *It is important to check the correspondence between the Read/Write status and the activated fields to see if the data is valid.* ***Note:*** *If the product is powered by battery and a low battery status is detected, the product will send the following commands:* *a PEV frame:* *- PEV frame: <PEV><Event number >* *with event number = battery discharged (0x10)* *- IEC (IMC not activated)*

Table 4-24 RFID Function Commands (continued)

Function Name	RF_Read	<TYPE> <CMD>: \0A\13 + Prms
Description	This function reads the data from a specific tag using the Tag ID. The read parameters for this function are included with the command. Unlike the RF_Auto_Run command, the parameters present in the NVM are not used and are not affected with this command.	
Parameters	Byte count [1 - 128] (number of bytes to be read, hex format). Sum of byte count and starting address must not exceed 128. ID [8]: tag ID in hex format Address [0 - 127]: start address to be read in hex format. Sum of byte count and starting address must not exceed 128. For example, 40-byte read (0x28); ID = 00004A4E408C8005; address = byte 24 (0x18); 0A\13\28\00\00\4A\4E\40\8C\80\05\18 0x7F 0x15(NAC)	
Responses	Successful Read: ASI frame (accept the function and sending information) + data frame after a success: <ASI> <IMC> <RF_Read function> <RF_Read_OK> 0x03 0x0A 0x13 0x72 <BFR> <[Preamble]> <[AIM_ID]> <Tag_ID> <Data> <[Postamble]> Unsuccessful Read: 03 0A 14 00 - No response from RF module ASI frame (accept the function and sending information) + error status: <ASI> <IMC> <RF_Read function> <RF_Read_Error> 0x03 0x0A 0x13 0x74 ***Note:*** *If the product is powered by battery and a low-battery status is detected, the product will send a PEV frame.* PEV frame: <PEV> <Event number>, with event number = battery discharged (0x10) IEC (IMC not activated)	

Table 4-24 RFID Function Commands (continued)

Function Name	RF_Write	<TYPE> <CMD>: \0A\14 + Prms
Description	This function causes the reader to write data to a specific tag using the Tag ID. The write parameters for this function are included with the command. Unlike the RF_Auto_Run command, the parameters present in the NVM are not used and are not affected with this command.	
Parameters	Byte Count [1 - 104]: number of bytes to be written (hex format). Sum of byte count and starting address must not exceed 128. ID [8]: Tag ID in hex format. Address [12 - 127]: Start address to be written (hex format). Sum of byte count and starting address must not exceed 128. Data [1 - 32]: in hexadecimal format For example, 10-byte write (49, 6E,...63) (0x0A), ID: 0510107F80AA0102 (DLE must be applied), start at 24 (0x18) data: "Intermec" \0A\14\0A\05\10\10\EE\80\80\AA\01\02\18\49\6E\74\65\72\6D\65\63	
Responses	Successful Write ASI frame (Accept the function and sending information) + success status after a success: <ASI><IMC><RF_Write function><RF_Write_OK> 0x03 0x0A 0x14 0x75 Unsuccessful Write 03 0A 14 00 - No response from RF module. ASI frame (Accept the function and sending information) + error status <ASI><IMC> <RF_Write function> <RF_Write_Error> 0x03 0x0A 0x14 0x76 ***Note:*** *If the product is powered by battery and a low battery status is detected, the product will send a PEV frame.* PEV frame: <PEV> < Event number>, with event number = battery discharged (0x10) IEC (IMC not activated)	

Data Buffer Function

These functions erase and/or transmit the data that is stored in the product's NVM. Table 4-25 lists the data buffer commands.

Table 4-25 Data Buffer Commands

Function Name	Data Buffer Clear	<TYPE> <CMD>: \0A\0B\
Description	This function erases the product data buffer (used in buffering mode, for example). **Note:** It is highly recommended to use the high-level acknowledge (ACC) of the RC Protocol (default setup) for this function due to the long time necessary to clear the buffer in flash memory (seconds). If simplified acknowledgement is selected, it is necessary to wait a few seconds before sending another command.	
Parameters	None	
Responses	IEC (IMC not activated) ACC (command accepted and started) NAC (command not recognized)	
Function Name	Transmit Data Buffer	<TYPE> <CMD>: \0A\0C\
Description	This function transmits data that has been previously stored in the data buffer to a host application or terminal. An ASI response is sent to the host with a word parameter *Number_of_Frames* to indicate the number of data frames that will be sent (each frame corresponding to a successful barcode read or transaction with a tag). The product then sends a quantity of consecutive BFR data records, equaling the *Number_of_Frames* sent in the preceding ASI response.	
Parameters	None	
Responses	<ASI><IMC><Transfer Data Buffer = 0x0C><Number_of_Frames> Number_of_Frames is a 2-byte field indicating the number of BFR frames to be sent to the host. <BFR><[Preamble]><AIM_ID><[Tag_ID]><#1 data record><[Postamble]> . . . <BFR><[Preamble]><AIM_ID><[Tag_ID]><#_of_Frames data record><[Postamble]> or NAC (command not recognized) IEC (IMC not activated)	

Bar Code Function

This bar code function command reads the bar code symbology that has been defined using the Sabre 1555 setup commands. Table 4-26 lists the command parameters.

Table 4-26 Bar Code Function Command

Function Name	BC_Decode	<TYPE> <CMD>: \0A\20 + Prms
Description	This command reads the bar code symbology as defined using setup commands in the product.	
Parameters	ON: \01\00\00 start decode (continuous operation) OFF: \00\00\00 stop decode Time-out: \01\ + hex word, decode with turn-off after success or time-out in seconds [1 - 65535 s] e.g., 10 s 0A\20\ 01\00\0A	
Responses	**<u>Decode ON, Continuous Operation</u>** An ACC frame + n BFR frame(s). ACC frame format: <ACC> 0x06 After each success, a BFR frame is sent to the host. BFR frame format: <BFR> <[Preamble]> <AIM_ID> <barcode_data> <[Postamble]> **<u>Decode OFF</u>** An ACC frame is sent after the function is stopped. ACC frame format: <ACC> **<u>Decode ON with Time-out</u>** An ACC frame is sent. Decode is turned off after a success (BFR frame is sent to the host) and time-out (PEV frame is sent to the host) After a success, a BFR frame is sent to the host. BFR frames: <BFR> <[Preamble]> <AIM_ID> <barcode_data> <[Postamble]> When the time-out expires, a PEV frame is sent. PEV frame format: <PEV><Event number>, with event number = BC_decode_Timeout (0x21) ***Note:*** *If the product is powered by battery and a low battery status is detected, the product will send a PEV frame.* PEV frame: <PEV> < Event number>, with event number = battery discharged (0x10) IEC (IMC not activated)	

User Control Function

A user control function command displays the current status of the Sabre 1555 unit. Table 4-27 lists the function command parameters.

Table 4-27 User Control Function

Function Name	Product_Status	<TYPE> <CMD>: \0A\30
Description	This function gets the current status of the product.	
Parameters	None	
Responses	ASI frame: <ASI><IMC><Product_status function><status> (status = <trigger status><battery status><current RFID operating mode status><buffering mode status> - Trigger status [0 - 3] 0x00: Trigger released 0x03: Trigger still pulled - Battery status [0 or 10] 0x00: battery discharged 0x10: battery OK - Current operating mode status (1 byte). (See “Operating Mode Selection” setup command in “RFID Operating Mode Selection” on page 4-47. These codes are listed below: 0x01 = Tag Identify *default 0x02 = Bar Code 0x05 = Tag Read 0x09 = Tag Write 0x0D = Tag Read Write 0x11 = Tag Filter 0x15 = Tag Filter Read 0x19 = Tag Filter Write 0x1D = Tag Filter Read Write. - Buffering mode status [0..1] (1 byte) 0x00 = Not active 0x01 = Active	

AIM (Automatic Identification Manufacturer's) Identifier

The symbology identifier concept provides a standardized way for a device receiving data from a bar code reader to differentiate between the symbologies.

Its structure is as follows:

The symbology identifier is an ASCII character string prefixed by the reading equipment to the data contained in a bar code symbol or RFID tag.

The structure of the symbology identifier string is as follows:

]cm . . .

where:

] ASCII value 93, represents the symbology identifier flag character

c represents the code character as defined in Table 4-28.

m represents the modifier character(s) as defined for the symbology in question

Example: "] A 0" identifies standard Code 39 without check digit (refer to the official AIM documentation on symbology identifiers for full information on the different processing options supported).

Table 4-28 shows the AIM identifier symbology.

Table 4-28 AIM Identifier Symbology

Symbology	Flag	Symbology_ID	Processing_Option
Codabar	]	F	0, 2, 4
Code 39	]	A	0, 1, 2, 4
Code 93	]	G	0
Code 128/EAN128	]	C	0,1
Interleaved 2 of 5	]	I	0, 1, 2
Matrix 2 of 5	]	X	0,
Standard 2 of 5	]	S	0, 1, 2
MSI Code	]	M	0, 1
Plessey Code	]	P	0
Telepen	]	B	0, 1
UPC/EAN (1)	]	E	0, 3, 4
UPC/EAN (2)	]	X	0
RF Tag	]	Z	2

(1) UPC/EAN *standard* lengths = 8, 13, 15 (add-on 2), 18 (add-on 5) characters
(2) UPC/EAN other lengths (no check digit, . . .)
(3) IMPORTANT: The "symbology_id" letter must be in uppercase for the above definitions.

Setup Commands (SMC = 0x0D)

Setup commands are used to configure the product directly from the host application (without using EasySet or reading configuration bar codes).

Note: *You must send an ESA command before performing a setup command.*

The product parameters are modified and saved in NVM using one of the following methods:

- EasySet system setup software
- Reading individual configuration bar codes
- Remote control interface

These three possibilities are based on the same principle that the frame syntax is a subset of the PPP low-level format. For example:

```
<STX><SetupModeCommand><SetupCommand><[Parameters]><CHK> <ETX>
0x7F  0x0D               0x41           0x07       0xxx 0xXX 0x7F
See the PPP low-level definition for details about the STX, ETX,
and CHK values.
```

See “Frame Fields” on page 4-7 for details regarding STX, ETX, and CHK values.

You can use the EasySet software to generate the desired setup parameter and then read the command value and parameter values under the configuration bar codes and insert them in the setup command frame.

Setup Command Tables

This section lists the setup command tables for the 1555 reader.

Physical Interface Command

Physical interface commands are used to control communications parameters. Table 4-29 shows the baud rate description.

Table 4-29 Baud Rate

Command Name	Baud Rate	<TYPE> <CMD>: \0D\41 + Prms
Description	The baud rate of the UART is changed to the specified parameter value.	
Parameters	[0 - 0x0A] respectively for 75, 150, 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600 bauds	
Responses	ACC (command accepted) NAC (command not recognized)	

Table 4-30 shows the remote control acceptance request description.

Table 4-30 RC_AccRequest

Command Name	RC_AccRequest	<TYPE> <CMD>: 0D\4D\43 + Prms
Description	Selects whether high-level frame acknowledgment is or is not required (only for ACC, NAC type; this does not affect other responses.) This parameter is only applied to the AccRequest bit in **BFR frames**.	
Parameters	[0 - 1]; 0: deactivated; 1: activated (default)	
Responses	ACC (command accepted) NAC (command not recognized)	

Table 4-31 shows the remote control low-level ACK time-out request description.

Table 4-31 RC_lowlevel_ACK_time-out

Command Name	RC_lowlevel_ACK_time-out	<TYPE> <CMD>: 0D\4C\53 + Prms
Description	The time-out determines the wait time for a low-level acknowledgment (ACK/NAK character) before reattempting the transmission.	
Parameters	[10 - 500] ms; see Appendix C to determine the value. The default value is 100 ms. Values $\leq$ 10 msec = 10 msec Values $\geq$ 500 msec = 500 msec	
Responses	ACC (command accepted) NAC (command not recognized)	

Table 4-32 shows the remote control high-level ACC time-out request description.

Table 4-32 RC_highlevel_ACC_time-out

Command Name	RC_highlevel_ACC_time-out	<TYPE> <CMD>: 0D\4C\56 + Prms
Description	The time-out determines the wait time for a high-level acknowledgment (ACC/NAC frame).	
Parameters	[10 - 2500] ms; see Appendix C to determine the value. The default value is 500 ms. Values ≤ 10 msec = 10 msec Values ≥ 2500 msec = 2500 msec	
Responses	ACC (command accepted) NAC (command not recognized)	

RFID Auto Configuration Commands

RFID auto configuration commands set up the Sabre 1555 to work with RFID tags. Table 4-33 lists the RFID autonomous mode configuration commands.

Table 4-33 RFID Auto Configuration Commands

Command Name	Display Tag IDs	<TYPE> <CMD>: \0D\4C\50 + Prms
Description	Displays on the LCD display the tag ID of the last tag read.	
Parameters	- ON: 01 - OFF: 00	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Data Display Format	<TYPE> <CMD>: \0D\4C\5C + Prms
Description	Defines the display format for data displayed in the product display window.	
Parameters	- Hex format: 01 - ASCII format: 00	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Activate RFID Read Fields	<TYPE> <CMD>: \0D\4C\57 + Prms
Description	Activates/deactivates 1 to 4 read fields in the tags.	
Parameters	Set bits to activate fields: b7 b6 b5 b4 — b3 b2 b1 b0 0 0 0 0 = reserved — 0 0 0 1 = field 1 activated 0 0 0 0 = reserved — 0 0 1 0 = field 2 activated 0 0 0 0 = reserved — 0 1 0 0 = field 3 activated 0 0 0 0 = reserved — 1 0 0 0 = field 4 activated 0 0 0 0 = reserved — x x x x = n fields x activated 0 0 0 0 = reserved — 1 1 1 1 = all 4 fields activated Example: 0x03 = fields 1 and 2 activated; 0x05 = fields 1 and 3 activated.	
Responses	ACC (command accepted) NAC (command not recognized)	

Table 4-33 RFID Auto Configuration Commands (continued)

Command Name	Define RFID Auto Read Parameters	<TYPE> <CMD>: \0D\4C\58 + Prms
Description	Defines the auto read parameters in the tags.	
Parameters	Read field [0 - 3], 1 byte, hexadecimal Start address [0 - 127], see Appendix C, base 64 Read size [1 - 32], 2 bytes, see Appendix C, base 64. ***Note:*** *Sum of start address and read size must not exceed 128*	
Responses	ACC (command accepted) NAC (command not recognized, or parameters invalid)	
Command Name	RFID Operating Mode Selection	<TYPE> <CMD>: \0D\4D\40 + Prms
Description	Defines the current operating mode. AutoRun activates RFID operating mode.	
Parameters	Set bits to activate fields: 0x01 = Tag Identify *default 0x05 = Tag Read 0x09 = Tag Write 0x0D = Tag Read Write 0x11 = Tag Filter 0x15 = Tag Filter Read 0x19 = Tag Filter Write 0x1D = Tag Filter Read Write.	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Define RFID Auto Run Response	<TYPE> <CMD>: \0D\4D\42 + Prms
Description	Defines whether the RFID Auto Run Response will contain complete or partial success status. Example: For an RFID Read operation of more than one field, when complete success is activated, the BFR success frame is sent only if all fields selected are read correctly. With partial success, the BFR partial success frame is sent, but the bit status must be checked for each field read. Default is complete success.	
Parameters	[0 - 1]; 0: partial success; 1: complete success	
Responses	ACC (command accepted) NAC (command not recognized)	

Table 4-33 RFID Auto Configuration Commands (continued)

Command Name	Activate RFID Auto Write Fields	<TYPE> <CMD>: \0D\4C\5A + Prms
Description	Activates/deactivates 1 to 4 write fields in the tags.	
Parameters	Set bits to activate fields: b7 b6 b5 b4 — b3 b2 b1 b0 0 0 0 0 = reserved — 0 0 0 1 = field 1 activated 0 0 0 0 = reserved — 0 0 1 0 = field 2 activated 0 0 0 0 = reserved — 0 1 0 0 = field 3 activated 0 0 0 0 = reserved — 1 0 0 0 = field 4 activated 0 0 0 0 = reserved — x x x x = n fields x activated 0 0 0 0 = reserved — 1 1 1 1 = all 4 fields activated Example: 0x03 = fields 1 and 2 activated; 0x05 = fields 1 and 3 activated.	
Responses	ACC (command accepted) NAC (command not recognized, or parameters are invalid)	
Command Name	Define RFID Write Parameters	<TYPE> <CMD>: \0D\4C\5B + Prms
Description	Defines the auto write parameters for the AutoRun command.	
	- write field [0 - 3], 1 byte, hexadecimal - start address [12 - 127], see Appendix C, base 64 - write size [1 - 32], see Appendix C, base 64. Sum of start address and write size must not exceed 128. - write string (compressed with regard to the more limited Code 128 encoding scheme [0 - 127] where a single ASCII character can be encoded as up to 3 bytes, see Appendix D) The write string uses hex values of extended ASCII characters [0 - 55] compressed as follows: 1. The first extended ASCII character in the string is always encoded into 2 bytes (): - the high-level nibble of the hex value of the extended ASCII character is encoded into the low-level nibble of the first byte (the high-level nibble is always equal to 0) - the low-level nibble of the hex value of the extended ASCII character is encoded into the low-level nibble of the second byte (the high-level nibble is always equal to 0)	

Table 4-33 RFID Auto Configuration Commands (continued)

Parameters

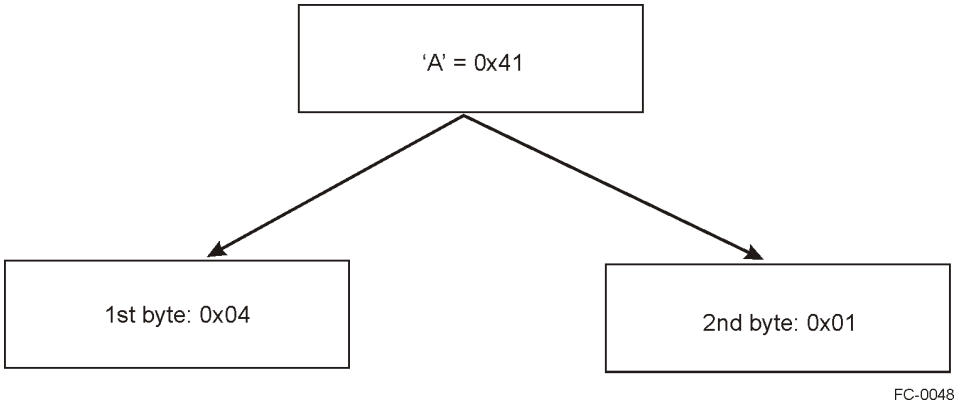


Figure 4-3 First Extended ASCII Character in Write String

2. For the next extended ASCII character following ():

   - if the delta of the hex value of this character with the hex value of the previous character is less than ±16, the new character can be compressed into a single byte as follows:
   - b7 b6 (high-level bits) are always set to 0
   - b5 (compression indicator) is set to 1
   - b4 (sign indicator) is set to:
     - 0 if the delta must be added to the ASCII value of the previous character to obtain the ASCII value of the new character
     - 1 if the delta must be subtracted from the ASCII value of the previous character to obtain the ASCII value of the new character
   - b3 b2 b1 b0 contain the delta value [0 - 15]
   - if not, the new character is encoded like the first character (step 1)

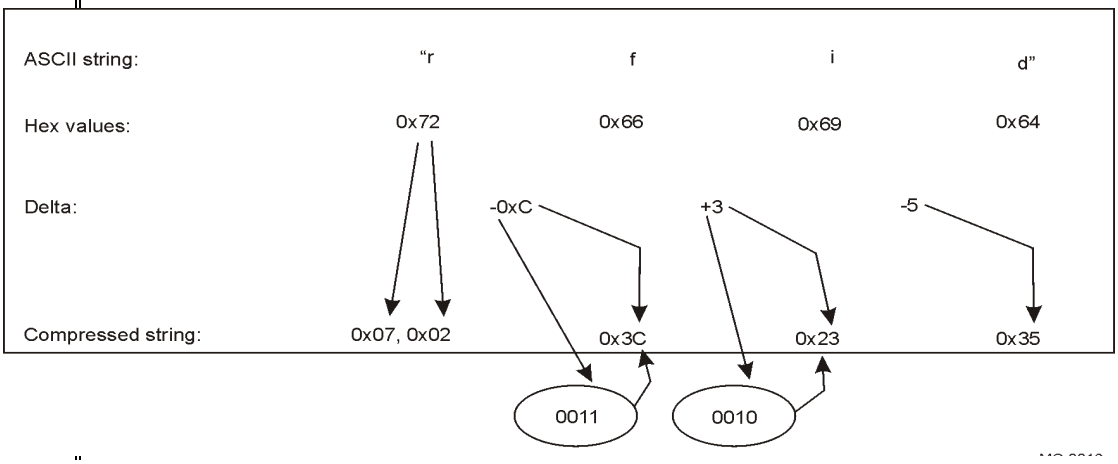


Figure 4-4 Second Through Last Extended ASCII Characters in Write String

Responses	ACC (command accepted) NAC (command not recognized)

Table 4-33 RFID Auto Configuration Commands (continued)

Command Name	Activate RFID Auto Filter Masks	<TYPE> <CMD>: \0D\4C\54 + Prms
Description	Activates/deactivates 1 to 4 filter masks for the tags.	
Parameters	Set bits to activate masks: b7 b6 b5 b4 — b3 b2 b1 b0 0 0 0 0 = reserved — 0 0 0 1 = mask 1 activated 0 0 0 0 = reserved — 0 0 1 0 = mask 2 activated 0 0 0 0 = reserved — 0 1 0 0 = mask 3 activated 0 0 0 0 = reserved — 1 0 0 0 = mask 4 activated 0 0 0 0 = reserved — x x x x = n masks x activated 0 0 0 0 = reserved — 1 1 1 1 = all 4 masks activated Example: 0x03 = fields 1 and 2 activated; 0x05 = fields 1 and 3 activated.	
Responses	ACC (command accepted) NAC (command not recognized)	

Table 4-33 RFID Auto Configuration Commands (continued)

<table>
<tr><td>Command Name</td><td>Define RFID Auto Filter Parameters</td><td><TYPE> <CMD>: \0D\4C\55 + Prms</td></tr>
<tr><td>Description</td><td colspan="2">Defines the Auto Filter parameters for the tags.</td></tr>
<tr><td>Parameters</td><td colspan="2">- filter mask number [0 - 3], 1 byte
- mask criteria, 1 byte:
- b7 to b3 = 0 (reserved)
- b2 = filter display format
0 = ASCII
1 = hex
- b1 b0 = comparison operators
0 0 = EQ
0 1 = NE
1 0 = GT
1 1 = LT
- start address [0 - 120], 2 bytes, see Appendix C
- mask “?” wildcards, 2 bytes:
- 1st byte:
-b7, b6 always set to 0
-b5 b4 b3 b2 reserved and set to 0
-b1, b0 represent the first 2 characters of mask
- 2nd byte:
-b7, b6 always set to 0
-b5 - b0 represent the next 6 characters of the mask</td></tr>
<tr><td></td><td colspan="2"><table>
<tr><td></td><td>b7</td><td>b6</td><td>b5</td><td>b4</td><td>b3</td><td>b2</td><td>b1</td><td>b0</td></tr>
<tr><td>1st wildcard mask byte</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>filter[1]</td><td>filter[2]</td></tr>
<tr><td>2nd wildcard mask byte</td><td>0</td><td>0</td><td>filter[3]</td><td>filter[4]</td><td>filter[5]</td><td>filter[6]</td><td>filter[7]</td><td>filter[8]</td></tr>
</table></td></tr>
<tr><td></td><td colspan="2">- filter string, 8 filter characters (compressed data string, see “RFID Auto Configuration Commands” on page 4-46).</td></tr>
<tr><td>Responses</td><td colspan="2">ACC (command accepted)
NAC (command not recognized)</td></tr>
</table>

Bar Code Decoding Commands

Bar code decoding commands are used to program the bar codes that are to be read by the laser scanner portion of the Sabre 1555 unit. Table 4-34 lists the decoding commands. For defaults and additional information on the bar code setup command, consult the information field in EasySet.

Table 4-34 Bar Code Decoding Commands

Command Name	Disable All Symbologies	<TYPE> <CMD>: \0D\41\4B
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code 39 Active	<TYPE> <CMD>: \0D\41\4C
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code 39 Inactive	<TYPE> <CMD>: \0D\41\4D
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code 2/5 Interleaved Active	<TYPE> <CMD>: \0D\41\4E
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	

Table 4-34 Bar Code Decoding Commands (continued)

Command Name	Code 2/5 Interleaved Inactive	<TYPE> <CMD>: \0D\41\4F
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code 2/5 Standard Active	<TYPE> <CMD>: \0D\41\50
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code 2/5 Standard Inactive	<TYPE> <CMD>: \0D\41\51
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Codabar Active	<TYPE> <CMD>: \0D\41\52
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Codabar Inactive	<TYPE> <CMD>: \0D\41\53
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	

Table 4-34 Bar Code Decoding Commands (continued)

Command Name	Code UPC/EAN Active	<TYPE> <CMD>: \0D\41\56
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code UPC/EAN Inactive	<TYPE> <CMD>: \0D\41\57
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code MSI Active	<TYPE> <CMD>: \0D\41\58
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code MSI Inactive	<TYPE> <CMD>: \0D\41\59
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code 128 Active	<TYPE> <CMD>: \0D\41\5A
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	

Table 4-34 Bar Code Decoding Commands (continued)

Command Name	Code 128 Inactive	<TYPE> <CMD>: \0D\41\5B
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code Plessey Active	<TYPE> <CMD>: \0D\41\5C
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code Plessey Inactive	<TYPE> <CMD>: \0D\41\5D
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code 93 Active	<TYPE> <CMD>: \0D\41\5E
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code 93 Inactive	<TYPE> <CMD>: \0D\41\5F
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	

Table 4-34 Bar Code Decoding Commands (continued)

Command Name	Code Matrix Active	<TYPE> <CMD>: \0D\42\40
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code Matrix Inactive	<TYPE> <CMD>: \0D\42\41
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code Telepen Active	<TYPE> <CMD>: \0D\42\42
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code Telepen Inactive	<TYPE> <CMD>: \0D\42\43
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code 39 Standard 43 Characters	<TYPE> <CMD>: \0D\42\4A
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	

Table 4-34 Bar Code Decoding Commands (continued)

Command Name	Code 39 Full ASCII	<TYPE> <CMD>: \0D\42\4B
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code 39 Start/Stop Transmitted	<TYPE> <CMD>: \0D\42\4C
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code 39 Start/Stop Not Transmitted	<TYPE> <CMD>: \0D\42\4D
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code 39 Start/Stop Accepted Characters	<TYPE> <CMD>: \0D\42\4E + Prms
Description		
Parameters	[1-3] 1: '$', 2: '*', 3: '$' OR '*' accepted	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code 39 Check Digit Not Used	<TYPE> <CMD>: \0D\42\4F
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	

Table 4-34 Bar Code Decoding Commands (continued)

Command Name	Code 39 Check Modulo 43 Checked and Transmitted	<TYPE> <CMD>: \0D\42\50
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code 39 Check Modulo 43 Checked but Not Transmitted	<TYPE> <CMD>: \0D\42\51
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code 39 Check CIP Checked and Transmitted	<TYPE> <CMD>: \0D\42\52
Description	Checksum for French pharmaceutical industry	
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code 39 Check CIP Checked but Not Transmitted	<TYPE> <CMD>: \0D\42\53
Description	Checksum for French pharmaceutical industry	
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code 39 Check CPI Checked and Transmitted	<TYPE> <CMD>: \0D\42\54
Description	Checksum for Italian pharmaceutical industry	
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	

Table 4-34 Bar Code Decoding Commands (continued)

Command Name	Code 39 Check CPI Checked but Not Transmitted	<TYPE> <CMD>: \0D\42\55
Description	Checksum for Italian pharmaceutical industry	
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code 39 Minimum Length	<TYPE> <CMD>: \0D\42\5C + Prms
Description	To optimize decoding performance and increase security select the value as the minimum length in your application.	
Parameters	[0; 3 - 50], hexadecimal, zero is a special value to accept any valid length (3 or more)	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code Interleaved 2/5 Check Mod 10 Checked and Transmitted	<TYPE> <CMD>: \0D\42\5F
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code Interleaved 2/5 Check Mod 10 Checked but Not Transmitted	<TYPE> <CMD>: \0D\43\40
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code Interleaved 2/5 Check CIP HR Checked and Transmitted	<TYPE> <CMD>: \0D\43\41
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	

Table 4-34 Bar Code Decoding Commands (continued)

Command Name	Code Interleaved 2/5 Check CIP HR Checked but Not Transmitted	<TYPE> <CMD>: \0D\43\42
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code Interleaved 2/5 Check Not Active	<TYPE> <CMD>: \0D\43\43
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code Interleaved 2/5 Fixed Lengths	<TYPE> <CMD>: \0D\43\44 + Prms
Description	Choose 1, 2, or 3 fixed lengths provides the best performance and security if your application has fixed lengths (minimum length = 2)	
Parameters	[2 - 50] hexadecimal, each parameter is separated by the value FNC4 (0x64) e.g., two lengths 10, 6 = \0D\43\44\0A\64\06 (length values must be provided in descending order)	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code Interleaved 2/5 Minimum Length	<TYPE> <CMD>: \0D\43\45 + Prms
Description	Choose a minimum length to optimize decoding performance and increase security, select the same length as the minimum length in your application	
Parameters	[0; 2 - 50], hexadecimal, zero is special value to accepts any valid length (2 or more)	
Responses	ACC (command accepted) NAC (command not recognized)	

Table 4-34 Bar Code Decoding Commands (continued)

Command Name	Code Standard 2/5 Computer Identics Format	<TYPE> <CMD>: \0D\43\46
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code Standard 2/5 Identicon Format	<TYPE> <CMD>: \0D\43\47
Description		
Parameters	None	
Responses	ACC if the command is accepted.	
Command Name	Code Standard 2/5 Check Digit Checked and Transmitted	<TYPE> <CMD>: \0D\43\48
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code Standard 2/5 Check Digit Checked but Not Transmitted	<TYPE> <CMD>: \0D\43\49
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code Standard 2/5 Check Digit Not Active	<TYPE> <CMD>: \0D\43\4A
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	

Table 4-34 Bar Code Decoding Commands (continued)

Command Name	Code Standard 2/5 Fixed Lengths	<TYPE> <CMD>: \0D\43\4B + Prms
Description	Choose 1, 2 or 3 fixed lengths provides the best performance and security if your application has fixed lengths (minimum length = 3)	
Parameters	[3 - 50] hexadecimal each parameter is separated by the value FNC4 (0x64) e.g., Two lengths 10, 6 = 0D\43\4B\0A\64\06 (length values must be provided in descending order)	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code Standard 2/5 Minimum Length	<TYPE> <CMD>: \0D\43\4C + Prms
Description	Choose a minimum length to optimize decoding performance and increase security, select the same length as the minimum length in your application	
Parameters	[0; 3 - 50], hexadecimal, zero is a special value to accept any valid length (3 or more)	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code 2/5 Matrix Minimum Length	<TYPE> <CMD>: \0D\46\59 + Prms
Description	Choose a minimum length to optimize decoding performance and increase security, select the same length as the minimum length in your application	
Parameters	[0; 3 - 50], zero is a special value to accept any valid length (3 or more)	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code Codabar Start/Stop Not Transmitted	<TYPE> <CMD>: \0D\43\4D
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code Codabar Start/Stop a,b,c,d	<TYPE> <CMD>: \0D\43\4E
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	

Table 4-34 Bar Code Decoding Commands (continued)

Command Name	Code Codabar Start/Stop A, B, C, D	<TYPE> <CMD>: \0D\43\4F
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code Codabar Start/Stop a,b,c,d/t, n, *, e	<TYPE> <CMD>: \0D\43\50
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code Codabar Start/Stop DC1, DC2, DC3, DC4	<TYPE> <CMD>: \0D\43\51
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code Codabar CLSI Library System Active	<TYPE> <CMD>: \0D\43\52
Description	Spaces inserted after char 1, 5, 10 in the 14 character label. e.g., “39990000192148” is sent as “3 9990 00019 2148”	
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code Codabar CLSI Library System Inactive	<TYPE> <CMD>: \0D\43\53
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	

Table 4-34 Bar Code Decoding Commands (continued)

Command Name	Code Codabar Fixed Lengths	<TYPE> <CMD>: \0D\43\54 + Prms
Description	Choose 1, 2 or 3 fixed lengths provides the best performance and security if your application has fixed lengths (minimum length = 3)	
Parameters	[3 - 50] hexadecimal, each parameter is separated by the value FNC4 (0x64) e.g., Two lengths 10, 6 = 0D\43\4B \0A\64\06 (length values must be provided in descending order)	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code Codabar Minimum Length	<TYPE> <CMD>: \0D\43\55 + Prms
Description	Choose a minimum length to optimize decoding performance and increase security, select the same length as the minimum length in your application	
Parameters	[0; 3 - 50], hexadecimal, zero is a special value to accept any valid length (3 or more)	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Check Digit Codabar Checked and Transmitted	<TYPE> <CMD>: \0D\46\54
Description	AIM recommendation	
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Check Digit Codabar Checked but Not Transmitted	<TYPE> <CMD>: \0D\46\55
Description	AIM recommendation	
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Check Digit Codabar Not Used	<TYPE> <CMD>: \0D\46\56
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	

Table 4-34 Bar Code Decoding Commands (continued)

Command Name	Code Telepen Numeric Mode Active	<TYPE> <CMD>: \0D\47\4B
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code Telepen ASCII Mode Active	<TYPE> <CMD>: \0D\47\4C
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code Telepen Minimum Length	<TYPE> <CMD>: \0D\47\4A + Prms
Description	Choose a minimum length to optimize decoding performance and increase security, select the same length as the minimum length in your application	
Parameters	[0; 1 - 50], hexadecimal, zero is a special value to accept any valid length (1 or more)	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code EAN/UPC All Code Selected	<TYPE> <CMD>: \0D\43\5C
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code UPC A Deactivated	<TYPE> <CMD>: \0D\43\5D
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	

Table 4-34 Bar Code Decoding Commands (continued)

Command Name	Code UPC E Deactivated	<TYPE> <CMD>: \0D\43\5E
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code EAN8 Deactivated	<TYPE> <CMD>: \0D\43\5F
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code EAN13 Deactivated	<TYPE> <CMD>: \0D\44\40
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code EAN/UPC Add-on Required and Transmitted	<TYPE> <CMD>: \0D\44\42
Description	Transmit forces add-ons to be decoded and transmitted for those add-ons that are activated.	
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code EAN/UPC Add-on Not Required but Transmitted if Read	<TYPE> <CMD>: \0D\44\43
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	

Table 4-34 Bar Code Decoding Commands (continued)

Command Name	Add-on 2 Activated	<TYPE> <CMD>: \0D\46\44
Description	Activate the decoding of EAN/UPC add-on 2	
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Add-on 2 Deactivated	<TYPE> <CMD>: \0D\44\41
Description	Deactivate the decoding of EAN/UPC add-on 2	
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Add-on 5 Activated	<TYPE> <CMD>: \0D\46\45
Description	Activate the decoding of EAN/UPC add-on 5	
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Add-on 5 Deactivated	<TYPE> <CMD>: \0D\46\46
Description	Deactivate the decoding of EAN/UPC add-on 5	
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code UPC A Number System Transmitted	<TYPE> <CMD>: \0D\44\44
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	

Table 4-34 Bar Code Decoding Commands (continued)

Command Name	Code UPC A Number System Not Transmitted	<TYPE> <CMD>: \0D\44\45
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code UPC A Check Digit Transmitted	<TYPE> <CMD>: \0D\44\46
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code UPC E Number System Transmitted	<TYPE> <CMD>: \0D\44\48
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code UPC E Number System Not Transmitted	<TYPE> <CMD>: \0D\44\49
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code UPC E Check Digit Transmitted	<TYPE> <CMD>: \0D\44\4A
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	

Table 4-34 Bar Code Decoding Commands (continued)

Command Name	Code UPC E Check Digit Not Transmitted	<TYPE> <CMD>: \0D\44\4B
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code UPC E Transmitted as UPC E	<TYPE> <CMD>: \0D\44\4C
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code UPC E Transmitted as UPC A	<TYPE> <CMD>: \0D\44\4D
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code UPC A Transmitted as EAN 13	<TYPE> <CMD>: \0D\44\4F
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code EAN 8 Transmitted as EAN 8	<TYPE> <CMD>: \0D\44\50
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	

Table 4-34 Bar Code Decoding Commands (continued)

Command Name	Code EAN 8 Transmitted as EAN 13	<TYPE> <CMD>: \0D\44\51
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code 128 Minimum Length	<TYPE> <CMD>: \0D\44\55 + Prms
Description	Choose a minimum length to optimize decoding performance and increase security, select the same length as the minimum length in your application	
Parameters	[0; 1 - 50], zero is a special value to accept any valid length (1 or more)	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code EAN/128 FNC1 Separator Character	<TYPE> <CMD>: \0D\46\52 + Prms
Description	Used as a separator when multiple identifiers and their fields are concatenated. Default FNC1 character is GS function character (ASCII 29)	
Parameters	See Appendix D to convert the character	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code Plessey Check Digit Transmitted	<TYPE> <CMD>: \0D\44\56
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code Plessey Check Digit Not Transmitted	<TYPE> <CMD>: \0D\44\57
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized).	

Table 4-34 Bar Code Decoding Commands (continued)

Command Name	Code Plessey Minimum Length	<TYPE> <CMD>: \0D\44\59 + Prms
Description	Choose a minimum length to optimize decoding performance and increase security, select the same length as the minimum length in your application	
Parameters	[0; 5 - 25], zero is a special value to accept any valid length (5 or more)	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code MSI Check Digit Mod 10 Checked and Transmitted	<TYPE> <CMD>: \0D\44\5A
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code MSI Check Digit DoubleMod 10 Checked and Transmitted	<TYPE> <CMD>: \0D\44\5B
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code MSI Check Digit Mod 10 Checked but Not Transmitted	<TYPE> <CMD>: \0D\44\5C
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code MSI Check Digit DoubleMod 10 Checked but Not Transmitted	<TYPE> <CMD>: \0D\44\5D
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	

Table 4-34 Bar Code Decoding Commands (continued)

Command Name	Code MSI Minimum Length	<TYPE> <CMD>: \0D\44\5F + Prms
Description	Choose a minimum length to optimize decoding performance and increase security, select the same length as the minimum length in your application	
Parameters	[0; 2 - 50], zero is a special value to accept any valid length (2 or more)	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Code 93 Minimum Length	<TYPE> <CMD>: \0D\45\42 + Prms
Description	Choose a minimum length to optimize decoding performance and increase security, select the same length as the minimum length in your application	
Parameters	[0; 1 - 50], zero is a special value to accept any valid length (1 or more)	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Check Digit EAN13 Transmitted	<TYPE> <CMD>: \0D\46\47
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Check Digit EAN13 Not Transmitted	<TYPE> <CMD>: \0D\46\48
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Check Digit EAN8 Transmitted	<TYPE> <CMD>: \0D\46\49
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	

Table 4-34 Bar Code Decoding Commands (continued)

Command Name	Check Digit EAN8 Not Transmitted	<TYPE> <CMD>: \0D\46\4A
Description		
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	

Data Decoding Security Commands

Data decoding security commands are used to verify the information that is read by the laser scanner portion of the Sabre 1555 unit. Table 4-35 lists the decoding commands.

Table 4-35 Data Decoding Security Commands

Command Name	Number of Consecutive Same Reads (data validation)	<TYPE> <CMD>: \0D\45\4B + Prms
Description	Data is only transmitted after repeated reads give the same result.	
Parameters	[1 - 10] 0 is converted to 1 and if value is >10 the value is forced to 10.	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Time-out Between Identical Consecutive Codes ICC	<TYPE> <CMD>: \0D\45\4C + Prms
Description	This delay prevents reading the same bar code more than once.	
Parameters	[1 - 32,000] ms, see Appendix C to determine the value. Default value is 300 ms. If value is $\geq$ 32000 then = 32000.	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Time-out Between Different Consecutive Codes DCC	<TYPE> <CMD>: \0D\45\4D + Prms
Description	This delay prevents unwanted reading of other bar codes on the same label.	
Parameters	[1 - 32,000] ms, see Appendix C to determine the value. Default value is 10 ms. If value is $\geq$ 32000 then = 32000.	
Responses	ACC (command accepted) NAC (command not recognized)	

Table 4-35 Data Decoding Security Commands (continued)

Command Name	Data Decoding Security Level	<TYPE> <CMD>: \0D\45\4E + Prms
Description	Predefined security level. Use medium and high security for poor-quality bar codes or critical applications.	
Parameters	0 = Normal: Single read, ICC = 300ms, DCC = 10 ms 1 = Medium: 2 consecutive same reads, ICC = 300 ms, DCC = 20 ms 2 = High: 4 consecutive same reads, ICC = 350 ms, DCC = 30 ms	
Responses	ACC (command accepted) NAC (command not recognized)	

Data Transmission Setting Commands

Data transmission setting commands identify the characters used in communicating between the bar code and the laser scanner portion of the Sabre 1555 unit. Table 4-36 lists the transmission setting commands.

Table 4-36 Data Transmission Setting Commands

Command Name	Preamble	<TYPE> <CMD>: \0D\45\53 + Prms
Description	This message is sent before the data to the host system.	
Parameters	No preamble: \3E\00 (default setting) See Appendix D to choose another string (Max 20 ASCII characters)	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Postamble	<TYPE> <CMD>: \0D\45\54 + Prms
Description	This message is sent after the data to the host system.	
Parameters	Carriage Return + Line Feed: \3E\0D\3E\0A (default setting) See Appendix D to choose another string (max. 20 ASCII characters)	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Custom Symbology Identifier Transmitted	<TYPE> <CMD>: \0D\45\55
Description	This identifier character is sent after a preamble and before data to the host system.	
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	

Table 4-36 Data Transmission Setting Commands (continued)

<table>
<tr><td>Command Name</td><td>Custom Symbology Identifier Not Transmitted</td><td colspan="2"><TYPE> <CMD>: \0D\45\56</td></tr>
<tr><td>Description</td><td colspan="3">This identifier character is not sent to the host system.</td></tr>
<tr><td>Parameters</td><td colspan="3">None</td></tr>
<tr><td>Responses</td><td colspan="3">ACC (command accepted)
NAC (command not recognized)</td></tr>
<tr><td>Command Name</td><td>Modify the Custom Symbology Identifier</td><td colspan="2"><TYPE> <CMD>: \0D\45\59 + Prms</td></tr>
<tr><td>Description</td><td colspan="3">By default a character identifier exists for each symbology. This command modifies the default character.</td></tr>
<tr><td rowspan="2">Parameters</td><td colspan="3">\06 + "1 Character parameter" (see Appendix C) e.g., Identifier for UPC-E = “E”: \0D\45\59\09\25</td></tr>
<tr><td>
<table>
<tr><th>symbology</th><th>Symbology indicator</th><th>Default Custom symbology_id</th></tr>
<tr><td>Codabar
Code 39
Code 93
Code 128/EAN 128
Interleaved 2 of 5
Matrix 2 of 5
Standard 2 of 5
MSI Code
Plessey Code
Telepen
EAN 8
EAN 13
UPC A
UPC E</td><td>0x04
0x01
0x0D
0x0B
0x02
0x0E
0x03
0x0A
0x0C
0x0F
0x07
0x06
0x08
0x09</td><td>‘D’
‘*’
‘D’
‘D’
‘I’
‘D’
‘D’
‘D’
‘D’
‘*’
“F’, ‘F’
‘F’
‘A’
‘E’</td></tr>
</table>
</td></tr>
<tr><td>Responses</td><td colspan="3">ACC (command accepted)
NAC (command not recognized)</td></tr>
</table>

Table 4-36 Data Transmission Setting Commands (continued)

Command Name	AIM Symbology Identifier Transmitted	<TYPE> <CMD>: \0D\47\58
Description	Activates AIM symbology identifier transmission for all symbologies. This identifier character is sent after a preamble and before data to the host system. e.g., format: "]A0" identifies standard Code 39 without check digit. Refer to the official AIM documentation on symbology identifiers in "AIM (Automatic Identification Manufacturer's) Identifier" on page 4-42 or EasySet for more information.	
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	AIM Symbology Identifier Not Transmitted	<TYPE> <CMD>: \0D\47\59
Description	Deactivates AIM symbology identifier transmission for all symbologies. Refer to the official AIM documentation on symbology identifiers or EasySet for more information.	
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	

Operating Setting Commands

Operating setting commands identify the parameters selected for the A/V settings of the Sabre 1555 unit. Table 4-37 lists the operating setting commands.

Table 4-37 Operating Setting Commands

Command Name	Good Read BC LED	<TYPE> <CMD>: \0D\45\4F + Prms
Description	LED ON for the good bar code read	
Parameters	Duration is fixed to 5 seconds. 0: Deactivates 1:Activates	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Good Read Beep Duration	<TYPE> <CMD>: \0D\45\49 + Prms
Description	Sets duration of “good read” beep in milliseconds.	
Parameters	Duration value in milliseconds (Base 64) see Appendix C; e.g., 80ms \0D\45\49\01\10 = Default	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	No Read Beep	<TYPE> <CMD>: \0D\45\49 + Prms
Description	Configures the reader to inhibit the “good read” beep.	
Parameters	Fixed value set to 0	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Number of Good Read Beeps	<TYPE> <CMD>: \0D\45\4A + Prms
Description	Sets the number of “good read” beeps.	
Parameters	0 = 1 Beep; 1 = 2 Beeps	
Responses	ACC (command accepted) NAC (command not recognized)	

Table 4-37 Operating Setting Commands (continued)

Command Name	Activate Power-up Beeps	<TYPE> <CMD>: \0D\45\5B
Description	Successful power-up: 2 beeps NVM integrity error: 3 long beeps. Note: This function does not apply to PDT version.	
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Deactivate Power-up Beeps	<TYPE> <CMD>: \0D\45\5A
Description	Does not apply to PDT version	
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Beeper Tone Period	<TYPE> <CMD>: \0D\4B\46 + Prms
Description	Beeper tone period in microseconds	
Parameters	200 µsec (03\08) to 10,000 µsec (02\1C\1G) = 100 Hz to 5000 Hz. See Appendix C to calculate base 64 value. Default is 312 µsec = 3205 Hz = 0D\4B\46\04\56. Example: For 100 Hz T = 10,000 µsec = 02\1C\16 = $2(64^2) + 28(64) + 16$.	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Beeps after Setup Command	<TYPE> <CMD>: \0D\4C\59 + Prms
Description	Activate/deactivate the beep when the command is sent by the remote control interface.	
Parameters	0: No beep 1: Beep (default)	
Responses	ACC (command accepted) NAC (command not recognized)	

Configuration Utility Commands

Configuration utility commands set up the operating parameters of the Sabre 1555 unit. Table 4-38 lists the configuration utility commands.

Table 4-38 Configuration Utility Commands

Command Name	Transparent Configuration Mode	<TYPE> <CMD>: \0D\46\41\03
Description	Allows you to use your bar code reader to set up other product families. (Setup commands are transmitted to the other product but do not affect your bar code reader) Remains active until the product is switched off or a PRC (\3F) is issued.	
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Activate the Temporary Configuration mode	<TYPE> <CMD>: \0D\46\41\00
Description	Test new setup without losing current setup.	
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Restore the Current Setup Configuration	<TYPE> <CMD>: \0D\46\41\01
Description	Ignores any temporary configuration actions and quits temporary configuration mode.	
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Update the Current Setup Configuration	<TYPE> <CMD>: \0D\46\41\02
Description	Permanently saves any temporary configuration actions and quits temporary configuration mode.	
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	

Table 4-38 Configuration Utility Commands (continued)

Command Name	Reset Factory Defaults	<TYPE> <CMD>: \0D\46\42
Description	Reset all parameters to their default settings.	
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Product Hardware Reset	<TYPE> <CMD>: \0D\4C\41
Description	This command restarts the product. Simulates cycling power. Product uses configuration parameters stored in NVM. If in temporary configuration, settings are set from those in NVM.	
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Get Version Info	<TYPE> <CMD>: \0D\46\43
Description	Software version	
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Display Data String Mode	<TYPE> <CMD>: \0D\46\4E
Description	Displays ASCII equivalents of data string and checksum values on a terminal screen when you read configuration bar codes without implementing configuration bar code settings. Remains active until the product is switched off or a PRC (\3F) is issued.	
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Configuration Enable	<TYPE> <CMD>: \0D\46\50
Description	Configuration possible all the time (default mode)	
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	

Table 4-38 Configuration Utility Commands (continued)

Command Name	Configuration Inhibit after 1 Minute	<TYPE> <CMD>: \0D\46\51
Description	Protect the product against unwanted bar code configuration* (cycle repeated until no configuration code read within one minute). On PDT, configuration inhibit is active immediately. (In this case, the EasySet setup software or IMC commands can be used to change the setup.) *(Cycle repeated until no configuration code read within 1 minute.)	
Parameters	None	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Inhibit Setup on Product	<TYPE> <CMD>: \0D\4C\52 + Prms
Description	Protect the product against unwanted setup modification in setup mode on the product, but the “Data buffer and Firmware versions” menus remain accessible in setup mode.	
Parameters	[0 1]; 0: inhibit configuration setup; 1: enable configuration setup	
Responses	ACC (command accepted) NAC (command not recognized)	
Command Name	Data Buffer	<TYPE> <CMD>: \0D\4C\5D + Prms
Description	The Data Buffer is used to store tag or bar code data in the NVM memory of the product.	
Parameters	[0 -1]; 0: no data store (transmit data to host); 1: store data in buffer	
Responses	ACC (command accepted) NAC (command not recognized)	

Examples of Interactive Mode and Setup Commands

The following examples show interactive and setup commands.

Note: *The arrows indicate the direction of the command, host to reader (==>) and reader to host (<==).*

Interactive Mode

Enter interactive mode without scanning a bar code

```
==> 0x7F 0x0A 0x00 0x01 0x00 0x00 0x37 0x7F
<== 0x06 //low-level acknowledge
<== 0x7F 0x06 0x01 0x00 0x0D 0x7F
==> 0x06
```

Switch to bar code mode

No action required

Power ON the RFID module

```
// RFID POWER ON
==> 0x7F 0x0A 0x0F 0x01 0x01 0x01 0x00 0x74 0x7F
<== 0x06
<== 0x7F 0x06 0x01 0x00 0x0D 0x7F
==> 0x06
```

```
While (test == Actif) {
    Receive bar code information when user presses trigger
do {
// read trigger status
==> 0x7F 0x0A 0x30 0x01 0x00 0xEE 0x80 0x7F  // DLE on CHK2
<== 0x06
<== 0x7F 0x03 0x0A 0x30 0x01 0x0A 0x01 0x01 0x19 0x7F
==> 0x06
while (TriggerStatus != 0x01 or Triggerstatus != 0x03)
// Read bar code label "4901780190737" for 10s (e.g. EAN13 is
selected by default).
==> 0x7F 0x0A 0x20 0x01 0x00 0x0A 0x01 0x00 0xF5 0x7F
<== 0x06
<== 0x7F 0x06 0x01 0x00 0x6D 0x7F //reply accepted command
==> 0x06
// BFR frame with bar code data
==> 0x7F 0x01 ']"E"0'"4901780190737"'CR"LF'0x01 0x29 0xC7 0x7F
<== 0x06
==> 0x7F 0x06 0x01 0x00 0x0D 0x7F
<== 0x06
...
==> 0x7F 0x0E 0x21 0x00 0x00 0x6C 0x7F  //PEV Bar code time-out
<== 0x06
```

Write the bar code information to a tag at a specified address when user presses trigger

```
do {
// read trigger status
==> 0x7F 0x0A 0x30 0x01 0x00 0xEE 0x80 0x7F  // DLE  on CHK2
<== 0x06
<== 0x7F 0x03 0x0A 0x30 0x01 0x0A 0x01 0x01 0x19 0x7F
==> 0x06
while (TriggerStatus != 0x01 or Triggerstatus != 0x03)
// RFID write with bar code data tagID•004A4E408C8005 @=0x18 L•
// data= "4901780190737"
//Note: With the latest 1555 firmware version, the RF_WRITE
command activates the RF_ON but does not perform RF_OFF after
execution.
```

```
==> 0x7F 0x0A 0x14 0x0D 0x00 0x00 0x4A 0x4E 0x40 0x8C 0x80 0x05
0x18 0x34 0x39 0x30 0x31 0x37 0x38 0x30 0x31 0x39 0x30 0x37 0x33
0x37 0x01 0x3E 0x61 0x7F
<== 0x06
<== 0x7F 0x03 0x0A 0x14 0x75 0x01 0x01 0x5E 0x7F  // ASI command
with successful write status (0x75)
==> 0x06
} //end while
// RFID MODULE POWER OFF
==> 0x7F 0x0A 0x0F 0x01 0x00 0x01 0x00 0x72 0x7F
<== 0x06
<== 0x7F 0x06 0x01 0x00 0x0D 0x7F
==> 0x06

Exit interactive mode without scanning a bar code
==> 0x7F 0x0A 0x00 0x01 0x01 0x00 0x2B 0x7F
<== 0x06
==> 0x7F 0x06 0x01 0x00 0x0D 0x7F
<== 0x06
```

Online Setup

Enter setup mode without scanning a bar code

Define a read address and length.

Example with address 24 and length 10:

Define read parameters - field 1 @24 L10 =

```
==> 0x7F 0x0D 0x4C 0x58 0x00 0x00 0x18 0x00 0x0A 0x00 0x05 0xB1 0x7F
<== 0x06
==> 0x7F 0x06 0x00 0x00 0x0C 0x7F
<== 0x06
```

The reader will send the Tag_ID and data at that address

Activate the read field 1 =

```
==> 0x7F 0x0D 0x4C 0x57 0x01 0x01 0x02 0x79 0x7F
<== 0x06
==> 0x7F 0x06 0x01 0x00 0x0D 0x7F
<== 0x06
```

Exit setup mode without scanning a bar code

Send PRC (\3F)
To execute, pull the trigger of the product to read the tag.

A

Acronyms and Glossary

A

AC alternating current

ACC high-level acknowledge setup command. A command that indicates that a message was received.

ACK low-level acknowledge character. A character that indicates that a message was received.

AIM Automatic Identification Manufacturers

antenna passive device that converts RF energy into magnetic energy (RF signal)

ASCII American Standard Code for Information Interchange. A standard seven bit code almost always transmitted with a parity bit for a total of eight bits per character. The ASCII set consists of both control and printing characters. ASCII was established by the American National Standards Institute to achieve compatibility between various types of data communication equipment. Equivalent to the International ISO 7-bit code. ASCII is the most commonly used code for non-IBM equipment.

ASI Accepted PPP command and sending information

B

bar code An automatic identification technology that encodes information into an array of adjacent parallel rectangular bars and spaces of varying widths.

baud The number of discreet conditions or signal events per second. In RS-232 and RS-422/485 systems, baud rate is the same as bits per second (bps).

BFR binary frame response

bit An abbreviation for binary digit. A binary digit is a single element (0 or 1) in a binary number. Eight bits equal one byte.

bps bits per second. The unit of measure used to describe the rate of data transmission. For example, 1200 bits per second means that there are 1200 data bits transmitted per second.

BRC baud rate confirm

BRR baud rate requested

byte A combination of eight bits in a predetermined pattern, designed to represent a digit or an alphanumeric character.

C

°C	degrees Centigrade
CEE	Communautés Économiques Européennes
CFG	Configuration
checksum	A calculated value that is used to test data integrity. Errors can occur when data is transmitted or when it is written to disk. One means of detecting such errors is the use of a checksum. A value is calculated for a given chunk of data by sequentially combining all the bytes of data with a series of arithmetic or logical operations. After the data is transmitted or stored, a new checksum is calculated and compared with the original one. If the checksums match, the transmission or storage was probably error free. If they do not match, an error occurred.
CHK	checksum
CISPR	International Special Committee on Radio Interference
cm	centimeter(s)
command	Data set that is recognized by the receiving device as intending to elicit a specific response
communications protocol	A set of rules or standards designed to enable computers to connect with each other and exchange data. An example of a communications protocol is Point-to-Point protocol.
configuration command	A configuration command changes the way the terminal or reader operates. You can enter a configuration command by typing on the keypad, by scanning a bar code label, or by sending a command from a device on the network.
CRC	cyclic redundancy check. The CRC is a data block check used to enhance data integrity. A CRC calculation is performed on the contents of each received RF message. If the result of this calculation compares with the received CRC, then the message was received without error.
CTS/RTS	Clear to send/ready to send. A type of hardware flow control. The reader or terminal signals the serial port device when ready to receive data. The reader or device checks with the serial port device when ready to send data.

D

data	Information that is processed by a computing device
DC	direct current
default parameters	A set of configuration parameters that are active when the device is shipped.

display Two-line screen on the control panel that displays messages such as printer status, menus, commands, and errors.

DLE data link escape

E

EasySet Windows-based PC configuration software

ECP error correcting protocol

EEC European Economic Community

EEPROM electrically erasable programmable read-only memory

EMI electromagnetic interference

EPROM Erasable programmable read-only memory. A special type of ROM that can be erased by exposing the chip to ultraviolet light. It can be reprogrammed after it is erased.

ESA external synchronization attempt

ESD electrostatic discharge

F

°F degrees Fahrenheit

FCC Federal Communications Commission. U.S. federal government agency responsible for setting and enforcing regulations associated with telecommunications.

FM Frame_Management

frame Consecutive bits of data in memory that are read and written as a group

frequency bands Range of RF assigned for transmission by an RF device

ft foot or feet

H

hex hexadecimal

hexadecimal base 16 numbering system that uses the characters 0–9 and A–F to represent the digits 0–15

host Device, generally a computer, that is connected to a reader system through the communications port

HPR	hardware protocol requested
hr	hour(s)
Hz	hertz

I

I/O	input/output
ICC	identical consecutive codes
ID	identification, encoded information unique to a particular tag
IEC	Interactive error command
IMC	interactive mode command
in	inch(es)
interactive	A computer session that provides immediate feedback to input or two-way communications between two devices.
interface	connection point for communication with another device
ISO	International Organization for Standardization
IT500	Intellitag 500 series of products manufactured by Intermec

K

k	kilo (10^3)

L

LCD	liquid crystal display
LSB	least significant byte

M

m	meter(s)
mAH	millamp-hour(s)

MB megabyte(s)

mega million (10^6)

message combination of fields, frames, and pages as required by the system to transmit or receive associated command and response data to and from the reader and host

MHz megahertz

mil *see* milli

milli one-thousandth (10^{-3})

mode method of operation

ms millisecond(s)

MSB most significant byte

mW milliwatt(s)

N

NAC high-level negative acknowledge command (command not recognized or executed)

NAD numerical ASCII data

NAK low-level negative acknowledge character (command not recognized or executed)

nibble sequence of four adjacent bits treated as a unit

NiCAD nickel-cadmium

NiMH nickel-metal-hydride

NVM nonvolatile memory

NVRAM nonvolatile RAM used to hold data over power loss

O

oz. ounce(s)

P

page Logical group of bytes representing data stored in a tag. A page is a maximum of 128 bits.

PDT	portable data terminal
PEV	Product event
PLL	phase locked loop
PPP	Communications protocol typically used to connect the reader directly to a computer or terminal. Data sent by the reader is followed by a carriage return and line feed (CR LF). XON/XOFF is supported. Point-to-Point protocol characters cannot be modified; however, the transmission parameters, such as parity and data bits, can be modified.
PRC	Product reset command
protocol	specified convention for the format of data messages communicated between devices
PVI	Product version identification
PWA	printed wiring assembly

R

RAM	random access memory
read	Process of acquiring data from a device, for example, from a tag or from computer memory
reader	Controlled interrogating device capable of acquiring data from a device, for example, acquiring and interrupting data from a tag
read zone	physical area in which a tag can be read by the reader system
RF	radio frequency
RFID	radio frequency identification
RLM	Resend last message
ROM	read-only memory
RS-232	Widely recognized protocol standard for serial binary data interchange. The standard covers the physical, electrical, and functional characteristics of the interface. RS-232 is the standard American format for serial data transmission by cable (that is, from a computer terminal to a modem). RS-232 transmission uses a distinctive 25-pin connector, although in most cases not all of the conductors are used (*see* serial port).
RSS	Reduced Space Symbology

S

s second(s)

SAE Society of Automotive Engineers

serial port A link between a microcomputer and a peripheral device through which data is transmitted.

SMC Setup mode command

SRAM static random access memory

start and stop characters A special bar code character that provides the scanner with start and stop reading instructions as well as a scanning direction indicator. The start character is normally at the left end of a horizontally oriented symbol (bar code label). The stop character is normally at the right end of a horizontally oriented symbol. For Code 39, the asterisk (*) character is used. Depending on the software used, the users may not have to enter the start and stop codes when they create their own bar code labels.

system A reader, RF module, antenna, and tag, all described by their general application and their interfaces with each other and any connected devices that are defined as being outside the system.

T

transponder small, self-contained device acting as an identifying tag

TTL transistor-to-transistor logic

TÜV Technischer Ueberwachungsverein (Technical Control Board)

U

UL Underwriters Laboratories, Incorporated

UPC Universal Product Code

V

V volt(s)

Ver version (software)

W

W watt(s)

write Process of recording data; for example, writing to computer memory or to a tag's memory. Writing erases (writes over) previous data stored at the specified memory locations

X

XOFF Defines a character that disables the transmission event. The receiving device sends XOFF when its receive buffers are nearly full of data. Older Intermec products may not include these protocol characters. Newer generation online reader products implement these protocol acronyms.

XON Enables or disables XON/XOFF flow control and defines a character that reenables transmission when the device can receive data again after an XOFF. Older Intermec products may not include these protocol characters. Newer generation online reader products implement these protocol acronyms.

XON/XOFF A type of software flow control for communication between digital devices. It stops the host from sending data when the device buffer fills up (XOFF) and starts it again when the buffer empties (XON).

B

Command Value-to-Base 32 Conversion Table (b6 = 1)

This appendix presents command value-to-base 32 (b6 = 1) conversions.

Table B-1 is a complete list of conversion values.

Table B-1 Base 10-to-Base 32 Conversion Values (b6 = 1)

base 10 value (decimal)	hex value	base 32 value (decimal, b6 = 1)	command (hex, b6 = 1)
00	0x00	64	0x40
01	0x01	65	0x41
02	0x02	66	0x42
03	0x03	67	0x43
04	0x04	68	0x44
05	0x05	69	0x45
06	0x06	70	0x46
07	0x07	71	0x47
08	0x08	72	0x48
09	0x09	73	0x49
10	0x0A	74	0x4A
11	0x0B	75	0x4B
12	0x0C	76	0x4C
13	0x0D	77	0x4D
14	0x0E	78	0x4E
15	0x0F	79	0x4F
16	0x10	80	0x50
17	0x11	81	0x51
18	0x12	82	0x52
19	0x13	83	0x53
20	0x14	84	0x54
21	0x15	85	0x55
22	0x16	86	0x56
23	0x17	87	0x57
24	0x18	88	0x58

Table B-1 Base 10-to-Base 32 Conversion Values (b6 = 1) (continued)

base 10 value (decimal)	hex value	base 32 value (decimal, b6 = 1)	command (hex, b6 = 1)
25	0x19	89	0x59
26	0x1A	90	0x5A
27	0x1B	91	0x5B
28	0x1C	92	0x5C
29	0x1D	93	0x5D
30	0x1E	94	0x5E
31	0x1F	95	0x5F
32	0x20	65, 64	0x41, 0x40
33	0x21	65, 65	0x41, 0x41
34	0x22	65, 66	0x41, 0x42
35	0x23	65, 67	0x41, 0x43
36	0x24	65, 68	0x41, 0x44
37	0x25	65, 69	0x41, 0x45
38	0x26	65, 70	0x41, 0x46
39	0x27	65, 71	0x41, 0x47
40	0x28	65, 72	0x41, 0x48
41	0x29	65, 73	0x41, 0x49
42	0x2A	65, 74	0x41, 0x4A
43	0x2B	65, 75	0x41, 0x4B
44	0x2C	65, 76	0x41, 0x4C
45	0x2D	65, 77	0x41, 0x4D
46	0x2E	65, 78	0x41, 0x4E
47	0x2F	65, 79	0x41, 0x4F
48	0x30	65, 80	0x41, 0x50
49	0x31	65, 81	0x41, 0x51
50	0x32	65, 82	0x41, 0x52
51	0x33	65, 83	0x41, 0x53
52	0x34	65, 84	0x41, 0x54

Table B-1 Base 10-to-Base 32 Conversion Values (b6 = 1) (continued)

base 10 value (decimal)	hex value	base 32 value (decimal, b6 = 1)	command (hex, b6 = 1)
53	0x35	65, 85	0x41, 0x55
54	0x36	65, 86	0x41, 0x56
55	0x37	65, 87	0x41, 0x57
56	0x38	65, 88	0x41, 0x58
57	0x39	65, 89	0x41, 0x59
58	0x3A	65, 90	0x41, 0x5A
59	0x3B	65, 91	0x41, 0x5B
60	0x3C	65, 92	0x41, 0x5C
61	0x3D	65, 93	0x41, 0x5D
62	0x3E	65, 94	0x41, 0x5E
63	0x3F	65, 95	0x41, 0x5F
64	0x40	66, 64	0x42, 0x40
65	0x41	66, 65	0x42, 0x41
66	0x42	66, 66	0x42, 0x42
67	0x43	66, 67	0x42, 0x43
68	0x44	66, 68	0x42, 0x44
69	0x45	66, 69	0x42, 0x45
70	0x46	66, 70	0x42, 0x46
71	0x47	66, 71	0x42, 0x47
72	0x48	66, 72	0x42, 0x48
73	0x49	66, 73	0x42, 0x49
74	0x4A	66, 74	0x42, 0x4A
75	0x4B	66, 75	0x42, 0x4B
76	0x4C	66, 76	0x42, 0x4C
77	0x4D	66, 77	0x42, 0x4D
78	0x4E	66, 78	0x42, 0x4E
79	0x4F	66, 79	0x42, 0x4F
80	0x50	66, 80	0x42, 0x50

Table B-1 Base 10-to-Base 32 Conversion Values (b6 = 1) (continued)

base 10 value (decimal)	hex value	base 32 value (decimal, b6 = 1)	command (hex, b6 = 1)
81	0x51	66, 81	0x42, 0x51
82	0x52	66, 82	0x42, 0x52
83	0x53	66, 83	0x42, 0x53
84	0x54	66, 84	0x42, 0x54
85	0x55	66, 85	0x42, 0x55
86	0x56	66, 86	0x42, 0x56
87	0x57	66, 87	0x42, 0x57
88	0x58	66, 88	0x42, 0x58
89	0x59	66, 89	0x42, 0x59
90	0x5A	66, 90	0x42, 0x5A
91	0x5B	66, 91	0x42, 0x5B
92	0x5C	66, 92	0x42, 0x5C
93	0x5D	66, 93	0x42, 0x5D
94	0x5E	66, 94	0x42, 0x5E
95	0x5F	66, 95	0x42, 0x5F
96	0x60	67, 64	0x43, 0x40
97	0x61	67, 65	0x43, 0x41
98	0x62	67, 66	0x43, 0x42
99	0x63	67, 67	0x43, 0x43
100	0x64	67, 68	0x43, 0x44
101	0x65	67, 69	0x43, 0x45
102	0x66	67, 70	0x43, 0x46
103	0x67	67, 71	0x43, 0x47
104	0x68	67, 72	0x43, 0x48
105	0x69	67, 73	0x43, 0x49
106	0x6A	67, 74	0x43, 0x4A
107	0x6B	67, 75	0x43, 0x4B
108	0x6C	67, 76	0x43, 0x4C

Table B-1 Base 10-to-Base 32 Conversion Values (b6 = 1) (continued)

base 10 value (decimal)	hex value	base 32 value (decimal, b6 = 1)	command (hex, b6 = 1)
109	0x6D	67, 77	0x43, 0x4D
110	0x6E	67, 78	0x43, 0x4E
111	0x6F	67, 79	0x43, 0x4F
112	0x70	67, 80	0x43, 0x50
113	0x71	67, 81	0x43, 0x51
114	0x72	67, 82	0x43, 0x52
115	0x73	67, 83	0x43, 0x53
116	0x74	67, 84	0x43, 0x54
117	0x75	67, 85	0x43, 0x55
118	0x76	67, 86	0x43, 0x56
119	0x77	67, 87	0x43, 0x57
120	0x78	67, 88	0x43, 0x58
121	0x79	67, 89	0x43, 0x59
122	0x7A	67, 90	0x43, 0x5A
123	0x7B	67, 91	0x43, 0x5B
124	0x7C	67, 92	0x43, 0x5C
125	0x7D	67, 93	0x43, 0x5D
126	0x7E	67, 94	0x43, 0x5E
127	0x7F	67, 95	0x43, 0x5F
128	0x80	68, 64	0x44, 0x40
129	0x81	68, 65	0x44, 0x41
130	0x82	68, 66	0x44, 0x42
131	0x83	68, 67	0x44, 0x43
132	0x84	68, 68	0x44, 0x44
133	0x85	68, 69	0x44, 0x45
134	0x86	68, 70	0x44, 0x46
135	0x87	68, 71	0x44, 0x47
136	0x88	68, 72	0x44, 0x48

Table B-1 Base 10-to-Base 32 Conversion Values (b6 = 1) (continued)

base 10 value (decimal)	hex value	base 32 value (decimal, b6 = 1)	command (hex, b6 = 1)
137	0x89	68, 73	0x44, 0x49
138	0x8A	68, 74	0x44, 0x4A
139	0x8B	68, 75	0x44, 0x4B
140	0x8C	68, 76	0x44, 0x4C
141	0x8D	68, 77	0x44, 0x4D
142	0x8E	68, 78	0x44, 0x4E
143	0x8F	68, 79	0x44, 0x4F
144	0x90	68, 80	0x44, 0x50
145	0x91	68, 81	0x44, 0x51
146	0x92	68, 82	0x44, 0x52
147	0x93	68, 83	0x44, 0x53
148	0x94	68, 84	0x44, 0x54
149	0x95	68, 85	0x44, 0x55
150	0x96	68, 86	0x44, 0x56
151	0x97	68, 87	0x44, 0x57
152	0x98	68, 88	0x44, 0x58
153	0x99	68, 89	0x44, 0x59
154	0x9A	68, 90	0x44, 0x5A
155	0x9B	68, 91	0x44, 0x5B
156	0x9C	68, 92	0x44, 0x5C
157	0x9D	68, 93	0x44, 0x5D
158	0x9E	68, 94	0x44, 0x5E
159	0x9F	68, 95	0x44, 0x5F
160	0xA0	69, 64	0x45, 0x40
161	0xA1	69, 65	0x45, 0x41
162	0xA2	69, 66	0x45, 0x42
163	0xA3	69, 67	0x45, 0x43
164	0xA4	69, 68	0x45, 0x44

Table B-1 Base 10-to-Base 32 Conversion Values (b6 = 1) (continued)

base 10 value (decimal)	hex value	base 32 value (decimal, b6 = 1)	command (hex, b6 = 1)
165	0xA5	69, 69	0x45, 0x45
166	0xA6	69, 70	0x45, 0x46
167	0xA7	69, 71	0x45, 0x47
168	0xA8	69, 72	0x45, 0x48
169	0xA9	69, 73	0x45, 0x49
170	0xAA	69, 74	0x45, 0x4A
171	0xAB	69, 75	0x45, 0x4B
172	0xAC	69, 76	0x45, 0x4C
173	0xAD	69, 77	0x45, 0x4D
174	0xAE	69, 78	0x45, 0x4E
175	0xAF	69, 79	0x45, 0x4F
176	0xB0	69, 80	0x45, 0x50
177	0xB1	69, 81	0x45, 0x51
178	0xB2	69, 82	0x45, 0x52
179	0xB3	69, 83	0x45, 0x53
180	0xB4	69, 84	0x45, 0x54
181	0xB5	69, 85	0x45, 0x55
182	0xB6	69, 86	0x45, 0x56
183	0xB7	69, 87	0x45, 0x57
184	0xB8	69, 88	0x45, 0x58
185	0xB9	69, 89	0x45, 0x59
186	0xBA	69, 90	0x45, 0x5A
187	0xBB	69, 91	0x45, 0x5B
188	0xBC	69, 92	0x45, 0x5C
189	0xBD	69, 93	0x45, 0x5D
190	0xBE	69, 94	0x45, 0x5E
191	0xBF	69, 95	0x45, 0x5F
192	0xC0	70, 64	0x46, 0x40

Table B-1 Base 10-to-Base 32 Conversion Values (b6 = 1) (continued)

base 10 value (decimal)	hex value	base 32 value (decimal, b6 = 1)	command (hex, b6 = 1)
193	0xC1	70, 65	0x46, 0x41
194	0xC2	70, 66	0x46, 0x42
195	0xC3	70, 67	0x46, 0x43
196	0xC4	70, 68	0x46, 0x44
197	0xC5	70, 69	0x46, 0x45
198	0xC6	70, 70	0x46, 0x46
199	0xC7	70, 71	0x46, 0x47
200	0xC8	70, 72	0x46, 0x48
201	0xC9	70, 73	0x46, 0x49
202	0xCA	70, 74	0x46, 0x4A
203	0xCB	70, 75	0x46, 0x4B
204	0xCC	70, 76	0x46, 0x4C
205	0xCD	70, 77	0x46, 0x4D
206	0xCE	70, 78	0x46, 0x4E
207	0xCF	70, 79	0x46, 0x4F
208	0xD0	70, 80	0x46, 0x50
209	0xD1	70, 81	0x46, 0x51
210	0xD2	70, 82	0x46, 0x52
211	0xD3	70, 83	0x46, 0x53
212	0xD4	70, 84	0x46, 0x54
213	0xD5	70, 85	0x46, 0x55
214	0xD6	70, 86	0x46, 0x56
215	0xD7	70, 87	0x46, 0x57
216	0xD8	70, 88	0x46, 0x58
217	0xD9	70, 89	0x46, 0x59
218	0xDA	70, 90	0x46, 0x5A
219	0xDB	70, 91	0x46, 0x5B
220	0xDC	70, 92	0x46, 0x5C

Table B-1 Base 10-to-Base 32 Conversion Values (b6 = 1) (continued)

base 10 value (decimal)	hex value	base 32 value (decimal, b6 = 1)	command (hex, b6 = 1)
221	0xDD	70, 93	0x46, 0x5D
222	0xDE	70, 94	0x46, 0x5E
223	0xDF	70, 95	0x46, 0x5F
224	0xE0	71, 64	0x47, 0x40
225	0xE1	71, 65	0x47, 0x41
226	0xE2	71, 66	0x47, 0x42
227	0xE3	71, 67	0x47, 0x43
228	0xE4	71, 68	0x47, 0x44
229	0xE5	71, 69	0x47, 0x45
230	0xE6	71, 70	0x47, 0x46
231	0xE7	71, 71	0x47, 0x47
232	0xE8	71, 72	0x47, 0x48
233	0xE9	71, 73	0x47, 0x49
234	0xEA	71, 74	0x47, 0x4A
235	0xEB	71, 75	0x47, 0x4B
236	0xEC	71, 76	0x47, 0x4C
237	0xED	71, 77	0x47, 0x4D
238	0xEE	71, 78	0x47, 0x4E
239	0xEF	71, 79	0x47, 0x4F
240	0xF0	71, 80	0x47, 0x50
241	0xF1	71, 81	0x47, 0x51
242	0xF2	71, 82	0x47, 0x52
243	0xF3	71, 83	0x47, 0x53
244	0xF4	71, 84	0x47, 0x54
245	0xF5	71, 85	0x47, 0x55
246	0xF6	71, 86	0x47, 0x56
247	0xF7	71, 87	0x47, 0x57
248	0xF8	71, 88	0x47, 0x58

Table B-1 Base 10-to-Base 32 Conversion Values (b6 = 1) (continued)

base 10 value (decimal)	hex value	base 32 value (decimal, b6 = 1)	command (hex, b6 = 1)
249	0xF9	71, 89	0x47, 0x59
250	0xFA	71, 90	0x47, 0x5A
251	0xFB	71, 91	0x47, 0x5B
252	0xFC	71, 92	0x47, 0x5C
253	0xFD	71, 93	0x47, 0x5D
254	0xFE	71, 94	0x47, 0x5E
255	0xFF	71, 95	0x47, 0x5F
256	0x100	72, 64	0x48, 0x40
257	0x101	72, 65	0x48, 0x41
258	0x102	72, 66	0x48, 0x42
259	0x103	72, 67	0x48, 0x43
260	0x104	72, 68	0x48, 0x44
261	0x105	72, 69	0x48, 0x45
262	0x106	72, 70	0x48, 0x46
263	0x107	72, 71	0x48, 0x47
264	0x108	72, 72	0x48, 0x48
265	0x109	72, 73	0x48, 0x49
266	0x10A	72, 74	0x48, 0x4A
267	0x10B	72, 75	0x48, 0x4B
268	0x10C	72, 76	0x48, 0x4C
269	0x10D	72, 77	0x48, 0x4D
270	0x10E	72, 78	0x48, 0x4E
271	0x10F	72, 79	0x48, 0x4F
272	0x110	72, 80	0x48, 0x50
273	0x111	72, 81	0x48, 0x51
274	0x112	72, 82	0x48, 0x52
275	0x113	72, 83	0x48, 0x53
276	0x114	72, 84	0x48, 0x54

Table B-1 Base 10-to-Base 32 Conversion Values (b6 = 1) (continued)

base 10 value (decimal)	hex value	base 32 value (decimal, b6 = 1)	command (hex, b6 = 1)
277	0x115	72, 85	0x48, 0x55
278	0x116	72, 86	0x48, 0x56
279	0x117	72, 87	0x48, 0x57
280	0x118	72, 88	0x48, 0x58
281	0x119	72, 89	0x48, 0x59
282	0x11A	72, 90	0x48, 0x5A
283	0x11B	72, 91	0x48, 0x5B
284	0x11C	72, 92	0x48, 0x5C
285	0x11D	72, 93	0x48, 0x5D
286	0x11E	72, 94	0x48, 0x5E
287	0x11F	72, 95	0x48, 0x5F

C

Data Value-to-Base 64 Conversion Table

This appendix presents data value-to-base 64 conversions.

Table C-1 is a complete list of conversion values.

Table C-1 Data Value-to-Base 64 Conversion Values

base 10 value (decimal)	hex value	base 64 value (decimal)	base 64 value (hex)
00	0x00	00	0x00
01	0x01	01	0x01
02	0x02	02	0x02
03	0x03	03	0x03
04	0x04	04	0x04
05	0x05	05	0x05
06	0x06	06	0x06
07	0x07	07	0x07
08	0x08	08	0x08
09	0x09	09	0x09
10	0x0A	10	0x0A
11	0x0B	11	0x0B
12	0x0C	12	0x0C
13	0x0D	13	0x0D
14	0x0E	14	0x0E
15	0x0F	15	0x0F
16	0x10	16	0x10
17	0x11	17	0x11
18	0x12	18	0x12
19	0x13	19	0x13
20	0x14	20	0x14
21	0x15	21	0x15
22	0x16	22	0x16
23	0x17	23	0x17

Table C-1 Data Value-to-Base 64 Conversion Values (continued)

base 10 value (decimal)	hex value	base 64 value (decimal)	base 64 value (hex)
24	0x18	24	0x18
25	0x19	25	0x19
26	0x1A	26	0x1A
27	0x1B	27	0x1B
28	0x1C	28	0x1C
29	0x1D	29	0x1D
30	0x1E	30	0x1E
31	0x1F	31	0x1F
32	0x20	32	0x20
33	0x21	33	0x21
34	0x22	34	0x22
35	0x23	35	0x23
36	0x24	36	0x24
37	0x25	37	0x25
38	0x26	38	0x26
39	0x27	39	0x27
40	0x28	40	0x28
41	0x29	41	0x29
42	0x2A	42	0x2A
43	0x2B	43	0x2B
44	0x2C	44	0x2C
45	0x2D	45	0x2D
46	0x2E	46	0x2E
47	0x2F	47	0x2F
48	0x30	48	0x30
49	0x31	49	0x31
50	0x32	50	0x32

Table C-1 Data Value-to-Base 64 Conversion Values (continued)

base 10 value (decimal)	hex value	base 64 value (decimal)	base 64 value (hex)
51	0x33	51	0x33
52	0x34	52	0x34
53	0x35	53	0x35
54	0x36	54	0x36
55	0x37	55	0x37
56	0x38	56	0x38
57	0x39	57	0x39
58	0x3A	58	0x3A
59	0x3B	59	0x3B
60	0x3C	60	0x3C
61	0x3D	61	0x3D
62	0x3E	62	0x3E
63	0x3F	63	0x3F
64	0x40	01, 00	0x01, 0x00
65	0x41	01, 01	0x01, 0x01
66	0x42	01, 02	0x01, 0x02
67	0x43	01, 03	0x01, 0x03
68	0x44	01, 04	0x01, 0x04
69	0x45	01, 05	0x01, 0x05
70	0x46	01, 06	0x01, 0x06
71	0x47	01, 07	0x01, 0x07
72	0x48	01, 08	0x01, 0x08
73	0x49	01, 09	0x01, 0x09
74	0x4A	01, 10	0x01, 0x0A
75	0x4B	01, 11	0x01, 0x0B
76	0x4C	01, 12	0x01, 0x0C
77	0x4D	01, 13	0x01, 0x0D

Table C-1 Data Value-to-Base 64 Conversion Values (continued)

base 10 value (decimal)	hex value	base 64 value (decimal)	base 64 value (hex)
78	0x4E	01, 14	0x01, 0x0E
79	0x4F	01, 15	0x01, 0x0F
80	0x50	01, 16	0x01, 0x10
81	0x51	01, 17	0x01, 0x11
82	0x52	01, 18	0x01, 0x12
83	0x53	01, 19	0x01, 0x13
84	0x54	01, 20	0x01, 0x14
85	0x55	01, 21	0x01, 0x15
86	0x56	01, 22	0x01, 0x16
87	0x57	01, 23	0x01, 0x17
88	0x58	01, 24	0x01, 0x18
89	0x59	01, 25	0x01, 0x19
90	0x5A	01, 26	0x01, 0x1A
91	0x5B	01, 27	0x01, 0x1B
92	0x5C	01, 28	0x01, 0x1C
93	0x5D	01, 29	0x01, 0x1C
94	0x5E	01, 30	0x01, 0x1D
95	0x5F	01, 31	0x01, 0x1F
96	0x60	01, 32	0x01, 0x20
97	0x61	01, 33	0x01, 0x21
98	0x62	01, 34	0x01, 0x22
99	0x63	01, 35	0x01, 0x23
100	0x64	01, 36	0x01, 0x24
101	0x65	01, 37	0x01, 0x25
102	0x66	01, 38	0x01, 0x26
103	0x67	01, 39	0x01, 0x27
104	0x68	01, 40	0x01, 0x28

Table C-1 Data Value-to-Base 64 Conversion Values (continued)

base 10 value (decimal)	hex value	base 64 value (decimal)	base 64 value (hex)
105	0x69	01, 41	0x01, 0x29
106	0x6A	01, 42	0x01, 0x2A
107	0x6B	01, 43	0x01, 0x2B
108	0x6C	01, 44	0x01, 0x2C
109	0x6D	01, 45	0x01, 0x2C
110	0x6E	01, 46	0x01, 0x2D
111	0x6F	01, 47	0x01, 0x2F
112	0x70	01, 48	0x01, 0x30
113	0x71	01, 49	0x01, 0x31
114	0x72	01, 50	0x01, 0x32
115	0x73	01, 51	0x01, 0x33
116	0x74	01, 52	0x01, 0x34
117	0x75	01, 53	0x01, 0x35
118	0x76	01, 54	0x01, 0x36
119	0x77	01, 55	0x01, 0x37
120	0x78	01, 56	0x01, 0x38
121	0x79	01, 57	0x01, 0x39
122	0x7A	01, 58	0x01, 0x3A
123	0x7B	01, 59	0x01, 0x3B
124	0x7C	01, 60	0x01, 0x3C
125	0x7D	01, 61	0x01, 0x3C
126	0x7E	01, 62	0x01, 0x3D
127	0x7F	01, 63	0x01, 0x3F
128	0x80	02, 00	0x02, 0x00
129	0x81	02, 01	0x02, 0x01
130	0x82	02, 02	0x02, 0x02
131	0x83	02, 03	0x02, 0x03

Table C-1 Data Value-to-Base 64 Conversion Values (continued)

base 10 value (decimal)	hex value	base 64 value (decimal)	base 64 value (hex)
132	0x84	02, 04	0x02, 0x04
133	0x85	02, 05	0x02, 0x05
134	0x86	02, 06	0x02, 0x06
135	0x87	02, 07	0x02, 0x07
136	0x88	02, 08	0x02, 0x08
137	0x89	02, 09	0x02, 0x09
138	0x8A	02, 10	0x02, 0x0A
139	0x8B	02, 11	0x02, 0x0B
140	0x8C	02, 12	0x02, 0x0C
141	0x8D	02, 13	0x02, 0x0D
142	0x8E	02, 14	0x02, 0x0E
143	0x8F	02, 15	0x02, 0x0F
144	0x90	02, 16	0x02, 0x10
145	0x91	02, 17	0x02, 0x11
146	0x92	02, 18	0x02, 0x12
147	0x93	02, 19	0x02, 0x13
148	0x94	02, 20	0x02, 0x14
149	0x95	02, 21	0x02, 0x15
150	0x96	02, 22	0x02, 0x16
151	0x97	02, 23	0x02, 0x17
152	0x98	02, 24	0x02, 0x18
153	0x99	02, 25	0x02, 0x19
154	0x9A	02, 26	0x02, 0x1A
155	0x9B	02, 27	0x02, 0x1B
156	0x9C	02, 28	0x02, 0x1C
157	0x9D	02, 29	0x02, 0x1C
158	0x9E	02, 30	0x02, 0x1D

Table C-1 Data Value-to-Base 64 Conversion Values (continued)

base 10 value (decimal)	hex value	base 64 value (decimal)	base 64 value (hex)
159	0x9F	02, 31	0x02, 0x1F
160	0xA0	02, 32	0x02, 0x20
161	0xA1	02, 33	0x02, 0x21
162	0xA2	02, 34	0x02, 0x22
163	0xA3	02, 35	0x02, 0x23
164	0xA4	02, 36	0x02, 0x24
165	0xA5	02, 37	0x02, 0x25
166	0xA6	02, 38	0x02, 0x26
167	0xA7	02, 39	0x02, 0x27
168	0xA8	02, 40	0x02, 0x28
169	0xA9	02, 41	0x02, 0x29
170	0xAA	02, 42	0x02, 0x2A
171	0xAB	02, 43	0x02, 0x2B
172	0xAC	02, 44	0x02, 0x2C
173	0xAD	02, 45	0x02, 0x2C
174	0xAE	02, 46	0x02, 0x2D
175	0xAF	02, 47	0x02, 0x2F
176	0xB0	02, 48	0x02, 0x30
177	0xB1	02, 49	0x02, 0x31
178	0xB2	02, 50	0x02, 0x32
179	0xB3	02, 51	0x02, 0x33
180	0xB4	02, 52	0x02, 0x34
181	0xB5	02, 53	0x02, 0x35
182	0xB6	02, 54	0x02, 0x36
183	0xB7	02, 55	0x02, 0x37
184	0xB8	02, 56	0x02, 0x38
185	0xB9	02, 57	0x02, 0x39

Table C-1 Data Value-to-Base 64 Conversion Values (continued)

base 10 value (decimal)	hex value	base 64 value (decimal)	base 64 value (hex)
186	0xBA	02, 58	0x02, 0x3A
187	0xBB	02, 59	0x02, 0x3B
188	0xBC	02, 60	0x02, 0x3C
189	0xBD	02, 61	0x02, 0x3C
190	0xBE	02, 62	0x02, 0x3D
191	0xBF	02, 63	0x02, 0x3F
192	0xC0	03, 00	0x03, 0x00
193	0xC1	03, 01	0x03, 0x01
194	0xC2	03, 02	0x03, 0x02
195	0xC3	03, 03	0x03, 0x03
196	0xC4	03, 04	0x03, 0x04
197	0xC5	03, 05	0x03, 0x05
198	0xC6	03, 06	0x03, 0x06
199	0xC7	03, 07	0x03, 0x07
200	0xC8	03, 08	0x03, 0x08
201	0xC9	03, 09	0x03, 0x09
202	0xCA	03, 10	0x03, 0x0A
203	0xCB	03, 11	0x03, 0x0B
204	0xCC	03, 12	0x03, 0x0C
205	0xCD	03, 13	0x03, 0x0D
206	0xCE	03, 14	0x03, 0x0E
207	0xCF	03, 15	0x03, 0x0F
208	0xD0	03, 16	0x03, 0x10
209	0xD1	03, 17	0x03, 0x11
210	0xD2	03, 18	0x03, 0x12
211	0xD3	03, 19	0x03, 0x13
212	0xD4	03, 20	0x03, 0x14

Table C-1 Data Value-to-Base 64 Conversion Values (continued)

base 10 value (decimal)	hex value	base 64 value (decimal)	base 64 value (hex)
213	0xD5	03, 21	0x03, 0x15
214	0xD6	03, 22	0x03, 0x16
215	0xD7	03, 23	0x03, 0x17
216	0xD8	03, 24	0x03, 0x18
217	0xD9	03, 25	0x03, 0x19
218	0xDA	03, 26	0x03, 0x1A
219	0xDB	03, 27	0x03, 0x1B
220	0xDC	03, 28	0x03, 0x1C
221	0xDD	03, 29	0x03, 0x1C
222	0xDE	03, 30	0x03, 0x1D
223	0xDF	03, 31	0x03, 0x1F
224	0xE0	03, 32	0x03, 0x20
225	0xE1	03, 33	0x03, 0x21
226	0xE2	03, 34	0x03, 0x22
227	0xE3	03, 35	0x03, 0x23
228	0xE4	03, 36	0x03, 0x24
229	0xE5	03, 37	0x03, 0x25
230	0xE6	03, 38	0x03, 0x26
231	0xE7	03, 39	0x03, 0x27
232	0xE8	03, 40	0x03, 0x28
233	0xE9	03, 41	0x03, 0x29
234	0xEA	03, 42	0x03, 0x2A
235	0xEB	03, 43	0x03, 0x2B
236	0xEC	03, 44	0x03, 0x2C
237	0xED	03, 45	0x03, 0x2C
238	0xEE	03, 46	0x03, 0x2D
239	0xEF	03, 47	0x03, 0x2E

Table C-1 Data Value-to-Base 64 Conversion Values (continued)

base 10 value (decimal)	hex value	base 64 value (decimal)	base 64 value (hex)
240	0xF0	03, 48	0x03, 0x30
241	0xF1	03, 49	0x03, 0x31
242	0xF2	03, 50	0x03, 0x32
243	0xF3	03, 51	0x03, 0x33
244	0xF4	03, 52	0x03, 0x34
245	0xF5	03, 53	0x03, 0x35
246	0xF6	03, 54	0x03, 0x36
247	0xF7	03, 55	0x03, 0x37
248	0xF8	03, 56	0x03, 0x38
249	0xF9	03, 57	0x03, 0x39
250	0xFA	03, 58	0x03, 0x3A
251	0xFB	03, 59	0x03, 0x3B
252	0xFC	03, 60	0x03, 0x3C
253	0xFD	03, 61	0x03, 0x3C
254	0xFE	03, 62	0x03, 0x3D
255	0xFF	03, 63	0x03, 0x3F
256	0x100	04, 00	0x04, 0x00
257	0x101	04, 01	0x04, 0x01
258	0x102	04, 02	0x04, 0x02
259	0x103	04, 03	0x04, 0x03

D

Code 128 ASCII Character-to-Parameter Conversion Table

This appendix presents Code 128 ASCII character-to-parameter conversions.

Table D-1 is a complete list of conversion values.

Table D-1 Code 128 ASCII Character-to-Parameter Conversion Values (continued)

ASCII Character		Parameter	
Decimal	**ASCII**	**Code 128 set B**	**Hex**
0	<NUL>	'^', <SP>	0x3E, 0x00
1	<SOH>	'^', !	0x3E, 0x01
2	<STX>	'^', "	0x3E, 0x02
3	<ETX>	'^', #	0x3E, 0x03
4	<EOT>	'^', $	0x3E, 0x04
5	<ENQ>	'^', %	0x3E, 0x05
6	<ACK>	'^', &	0x3E, 0x06
7	<BEL>	'^', '	0x3E, 0x07
8	<BS>	'^', (	0x3E, 0x08
9	<HT>	'^',)	0x3E, 0x09
10	<LF>	'^', *	0x3E, 0x0A
11	<VT>	'^', +	0x3E, 0x0B
12	<FF>	'^', ,	0x3E, 0x0C
13	<CR>	'^', -	0x3E, 0x0D
14	<SO>	'^', .	0x3E, 0x0E
15	<SI>	'^', /	0x3E, 0x0F
16	<DLE>	'^', 0	0x3E, 0x10
17	<DC1>	'^', 1	0x3E, 0x11
18	<DC2>	'^', 2	0x3E, 0x12
19	<DC3>	'^', 3	0x3E, 0x13
20	<DC4>	'^', 4	0x3E, 0x14
21	<NACK>	'^', 5	0x3E, 0x15
22	<SYN>	'^', 6	0x3E, 0x16
23	<ETB>	'^', 7	0x3E, 0x17

Table D-1 Code 128 ASCII Character-to-Parameter Conversion Values (continued)

ASCII Character		Parameter	
Decimal	**ASCII**	**Code 128 set B**	**Hex**
24	<CAN>	'^', 8	0x3E, 0x18
25	<EM>	'^', 9	0x3E, 0x19
26	<SUB>	'^', :	0x3E, 0x1A
27	<ESC>	'^', ;	0x3E, 0x1B
28	<FS>	'^', <	0x3E, 0x1C
29	<GS>	'^', =	0x3E, 0x1D
30	<RS>	'^', >	0x3E, 0x1E
31	<US>	'^', ?	0x3E, 0x1F
32	<SP>	<SP>	0x00
33	!	!	0x01
34	"	"	0x02
35	#	#	0x03
36	$	'$', '$'	0x04, 0x04
37	%	%	0x05
38	&	&	0x06
39	'	'	0x07
40	(	(	0x08
41	)	)	0x09
42	*	*	0x0A
43	+	+	0x0B
44	,	,	0x0C
45	-	-	0x0D
46	.	.	0x0E
47	'/'	'/', '/'	0x0F, 0x0F
48	0	0	0x10
49	1	1	0x11
50	2	2	0x12

Table D-1 Code 128 ASCII Character-to-Parameter Conversion Values (continued)

ASCII Character		Parameter	
Decimal	**ASCII**	**Code 128 set B**	**Hex**
51	3	3	0x13
52	4	4	0x14
53	5	5	0x15
54	6	6	0x16
55	7	7	0x17
56	8	8	0x18
57	9	9	0x19
58	:	:	0x1A
59	;	;	0x1B
60	<	<	0x1C
61	=	=	0x1D
62	>	>	0x1E
63	?	?	0x1F
64	'@'	'@', '@'	0x20, 0x20
65	A	A	0x21
66	B	B	0x22
67	C	C	0x23
68	D	D	0x24
69	E	E	0x25
70	F	F	0x26
71	G	G	0x27
72	H	H	0x28
73	I	I	0x29
74	J	J	0x2A
75	K	K	0x2B
76	L	L	0x2C
77	M	M	0x2D

Table D-1 Code 128 ASCII Character-to-Parameter Conversion Values (continued)

ASCII Character		Parameter	
Decimal	**ASCII**	**Code 128 set B**	**Hex**
78	N	N	0x2E
79	O	O	0x2F
80	P	P	0x30
81	Q	Q	0x31
82	R	R	0x32
83	S	S	0x33
84	T	T	0x34
85	U	U	0x35
86	V	V	0x36
87	W	W	0x37
88	X	X	0x38
89	Y	Y	0x39
90	Z	Z	0x3A
91	[	[	0x3B
92	\	\	0x3C
93	]	]	0x3D
94	^	'^', '^'	0x3E, 0x3E
95	_	_	0x3F
96	' ` '	$, `	0x4, 0x01, 0x20
97	a	@, A	0x20, 0x21
98	b	@, B	0x20, 0x22
99	c	@, C	0x20, 0x23
100	d	@, D	0x20, 0x24
101	e	@, E	0x20, 0x25
102	f	@, F	0x20, 0x26
103	g	@, G	0x20, 0x27

Table D-1 Code 128 ASCII Character-to-Parameter Conversion Values (continued)

ASCII Character		Parameter	
Decimal	**ASCII**	**Code 128 set B**	**Hex**
104	h	@, H	0x20, 0x28
105	i	@, I	0x20, 0x29
106	j	@, J	0x20, 0x2A
107	k	@, K	0x20, 0x2B
108	l	@, L	0x20, 0x2C
109	m	@, M	0x20, 0x2D
110	n	@, N	0x20, 0x2E
111	o	@, O	0x20, 0x2F
112	p	@, P	0x20, 0x30
113	q	@, Q	0x20, 0x31
114	r	@, R	0x20, 0x32
115	s	@, S	0x20, 0x33
116	t	@, T	0x20, 0x34
117	u	@, U	0x20, 0x35
118	v	@, V	0x20, 0x36
119	w	@, W	0x20, 0x37
120	x	@, X	0x20, 0x38
121	y	@, Y	0x20, 0x39
122	z	@, Z	0x20, 0x3A
123	{	@, [	0x20, 0x3B
124	|	@, \	0x20, 0x3C
125	}	@,]	0x20, 0x3D
126	~	@, ^	0x20, 0x3E
127	<DEL>	@, _	0x20, 0x3F

E

System Specifications

This appendix presents technical specifications for the Sabre 1555 RFID and Bar Code Reader.

Requirements

The standard version of the Sabre 1555 requires 7.5 ± 0.5 VDC.

Battery Pack

The battery pack that is used to power the portable data terminal (PDT) version of the Sabre 1555 requires an 1200 mAH 90-240 VAC, 50Hz/60Hz, lithium ion with integral charger and 4-hour charge time.

Physical Specifications

Table E-1 lists the physical specifications of the Sabre 1555.

Table E-1 Sabre 1555 Physical Specifications

Specification	Value
Scanner length	14.61 cm (5.75 in)
Scanner height	22.23 cm (8.75 in)
Scanner width	8.89 cm (3.5 in)
Scanner weight	600 g (16 oz.) without cable or battery pack
Battery pack weight	176 g (6.23 oz.)
Cable(s) weight	199 to 227 g (7 to 8 oz.)

Environmental Specifications

The Sabre 1555 can withstand the environmental tolerances shown in Table E-2.

Table E-2 Sabre 1555 Environmental Specifications

Specification	Value
Operating temperature	-20° to 50°C (-22° to 122°F) — standard 0° to 50°C (32° to 122°F) — PDT ***Note:*** *for 2450 MHz limited on continuous runtime at ambient temperatures* >25°C (77°F)
Storage temperature (PDT and standard)	-40° to 70°C (-40° to 168°F) — PDT and standard
Battery storage temperature	-20° to 60°C (-4° to 140°F)
Humidity	0-95% noncondensing
Shock	Twenty-six 1.22 m (4-ft) drops to concrete
Vibration	Class 3 SAE off-road vehicle specification
Environmental	Scanner sealed against windblown rain and dust (IEC 529 rated at IP54)
ESD	Conforms to 6 KV rating per IEC 801-2
EMI Emissions	Conforms to Class A FCC Part 15, RSS 210 and CISPR 22 Class B
Approvals	UL & cUL listed, C22.2 No. 950/UL 1950, TÜV Hheinland GS Licensed EN 60950, EN 60825-1, CD marked, C-Tick marked

Power Usage

The current use for the Sabre 1555 unit when operating in interactive mode are listed in Table E-3.

Table E-3 Sabre 1555 Current Use with RF Power On

Version	Parameters	Current Use (mA)
915 MHz	Interactive mode command (IMC) deactivated (1555 in *sleep* mode, no LCD)	<1
	IMC activated	150
	Bar code decode ON	200
	IMC activated, RF submodule ON, no RF commands	350
	IMC activated, RF submodule ON, RF commands	950

Table E-3 Sabre 1555 Current Use with RF Power On (continued)

Version	Parameters	Current Use (mA)
2450 MHz	IMC deactivated (1555 in *sleep* mode, no LCD)	<1
	IMC activated	100
	Bar code decode ON	200
	IMC activated, RF submodule ON, no RF commands	350
	IMC activated, RF submodule ON, RF commands	1000

Communications

The Sabre 1555 uses either 2450- or 915-MHz unlicensed FCC spread-spectrum frequency-hopping.

Host Interfaces

RS-232 C or RS-232 TTL, depending on application or cable.

Software Features

RFID reader/programmer software fully supports ANS NCITS 256-1999 for item management.

Configuration

Configuration setup is done via EasySet configuration software for Windows. Print and scan a bar code menu or download a configuration via online or IMC setup.

Data Rate

The Sabre 1555 can read 8 bytes of data from a tag in less than 12 ms, and can write a single byte of data to a tag in less than 25 ms.

Tag Filter Function

User-specified groups within a population of tags can be selected, read from, and/or written to using the tag filter function commands.

Read/Write Performance

Read/write performance varies with RFID inserts, tags, and frequency configurations.

Read/Write Ranges

Write ranges are approximately 70% of the effective read ranges of the RFID inserts.

Read Range Examples

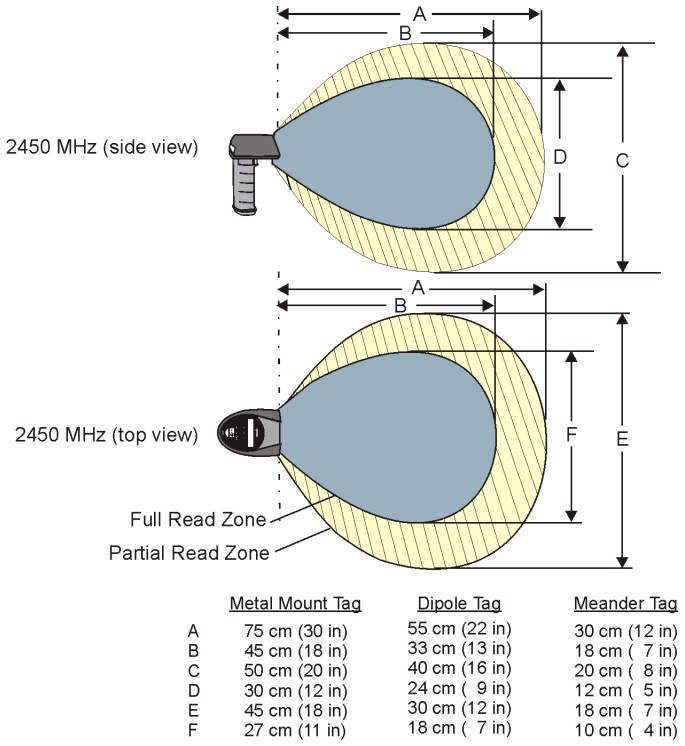
Read ranges are shown in Table E-4 for RFID inserts using frequency-hopping FCC unlicensed operation reader and free space dipole tag.

Table E-4 Read Range Examples

Frequency (MHz)	Specification	Read Range
915	46 x 79 x 0.5 mm (1.81 x 3.13 x 0.02 in) Free Space Sticker Tag (P/N ITTC9153024)	2.5 cm to 137 cm (~1 in to ~54 in)
2450	6 x 30 x 0.5 mm (0.24 x 1.18 x 0.02 in) with Free Space Meander Tag (P/N ITTF2451001)	1.0 cm to 60 cm (0.4 in to ~24 in) for single tag read 1.0 cm to 15 cm (0.4 in to 6 in) for multiple tag read

For both the 2450- and 915-MHz units, the partial read zone is most affected by environmental conditions (i.e., obstructions, metal structures), equipment factors (i.e., type of tag, number of tags), and variation from unit to unit.

The RFID tag read ranges for the 2450-MHz unit are shown in Figure E-1.



	Metal Mount Tag	Dipole Tag	Meander Tag
A	75 cm (30 in)	55 cm (22 in)	30 cm (12 in)
B	45 cm (18 in)	33 cm (13 in)	18 cm (7 in)
C	50 cm (20 in)	40 cm (16 in)	20 cm (8 in)
D	30 cm (12 in)	24 cm (9 in)	12 cm (5 in)
E	45 cm (18 in)	30 cm (12 in)	18 cm (7 in)
F	27 cm (11 in)	18 cm (7 in)	10 cm (4 in)

Figure E-1 Tag Read Patterns for 2450-MHz Version of Sabre 1555

An example of RFID tag read range for the 915-MHz unit is shown in Figure E-2.

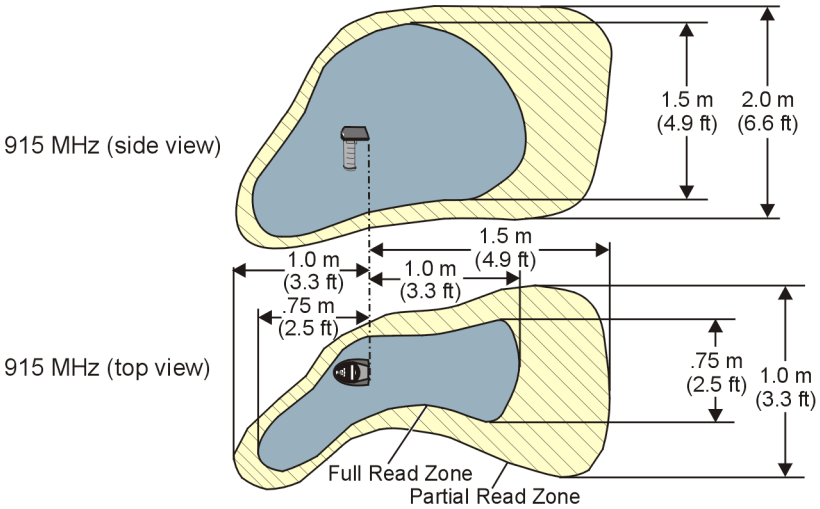


Figure E-2 Tag Read Patterns for 915-MHz Version of Sabre 1555

Note: *The read range shape that extends behind the 1555 is determined by where the user is standing, and in which hand he/she holds the 1555. In Figure E-2, the user operated the 1555 reader in the right hand, and his body blocked the RF. If the 1555 reader is operated in the left hand, the user's body blocks the RF to the right and behind the 1555.*

Optics System

Visible laser diode: 650 nm ±10 nm

USA/Canada Class II

International Class 2

Scan rate: 36 ±3 scans per second

Optical Parameters

Pitch: ±65° from normal

Skew: ±55° from normal:

Table E-5 Depth of Field–Long Range 1555 (1555Exx02040201)*

Specification	Value
10 mil	27.9-98.3 cm (11-19 in)
15 mil	20.3-83.8 cm (8-33 in)
20 mil	20.3-96.5 cm (8-38 in)
40 mil	22.9-190.5 cm (9-75 in)
55 mil	22.9-203 cm (9-80 in)
70 mil (retroflective)	190.5-408.9 cm (75-161 in)
100 mil (retroflective)	210.8-530.9 cm (83-209 in)

Note: **Depth of field measured using Code 39, ANSI Grade A*

Table E-6 Depth of Field–High Visibility (1555Exx01xx0001)*

Specification	Value
5 mil	7.4-11.2 cm (2.9-4.4 in)
7.5 mil	20.3-83.8 cm (8-33 in)
15 mil	20.3-96.5 cm (8-38 in)

Table E-6 Depth of Field–High Visibility (1555Exx01xx0001) (continued)*

20 mil	22.9-190.5 cm (9-75 in)
40 mil	22.9-203 cm (9-80 in)
55 mil	190.5-408.9 cm (75-161 in)

Note: **Depth of field measured using Code 39, ANSI Grade A*

Print Contrast

The minimum print contrast is 40% reflectance difference (long range), and 25% reflectance difference (high visibility).

Ambient Light Immunity

The Sabre 1555 is immune to direct exposure from office level lighting and sunlight.