RC855 IBM 3270 BSC Emulator Reference Manual

RCSL No 42-I 2156

RCSL No: 42-i2156

Edition: November 1982

Author: Pierce C. Hazelton

Title:

RC855 IBM 3270 BSC Emulator
Reference Manual

RCSL No 42-i 2164

A/S REGNECENTRALEN
af 1979

Keywords:

RC855, IBM 3270 BSC Emulator, Reference Manual.

Abstract:

Reference manual for RC855 IBM 3270 BSC Emulator. Describes: operational characteristics of display terminal, keyboard, magnetic-stripe reader, and printer; communication with host computer; commands and responses; character codes; and V.24 connections.

(120 printed pages)

Printed by A/S Regnecentralen af 1979, Copenhagen

RCSL No 42-I 2164

FOREWORD

First edition: RCSL No 42-i1692.

Second edition: RCSL No 42-i2156.
This edition supersedes the first edition. Important changes in the text are indicated by a vertical line in the left-hand margin.

Pierce C. Hazelton
A/S REGNECENTRALEN af 1979, November 1982

TABLE OF CONTENTS PAGE

TABLE OF CONTENTS (continued) PAGE

TABLE OF CONTENTS (continued) PAGE

APPENDICES:

TABLE OF CONTENTS (continued) PAGE

FIGURES PAGE

1. INTRODUCTION 1.

The RC855 Display Terminal is designed for the construction of distributed systems in which each display terminal in a cluster configuration can be used interchangeably for communication with a host computer and as a local microcomputer system.

The RC855 is a member of the RC850 family of intelligent display terminals. The use of a microcomputer as the basis of all RC850 display terminals makes it possible to emulate a number of different terminals on the market as well as to execute a variety of local application programs by means of the same terminal. RC850 display terminals exist accordingly in several different versions.

The RC855 is in principle a soft-programmed version: a versatile work station, which can be alternately loaded with communication programs and programs for local applications from an attached diskette drive.

1.1 RC855 IBM 3270 BSC Emulator 1.1

Inasmuch as the IBM 3270 Information Display System is one of the most widely used terminal systems on the market, an IBM 3270 BSC emulator is included among the communication packages for the RC855. In addition to being available on diskette, the RC855 IBM 3270 BSC Emulator is alternatively available on an internal program memory board for automatic loading when the multifunction capability of the RC855 Work Station is not required.

A 3270 oriented application program in the host computer communicates with the terminal operator through a series of display screen images. An image of the data transmitted from the host is displayed at the terminal. By means of an attached keyboard, the operator can enter, modify, or erase data on the display, and cause the revised data to be returned to the host for storage or further processing. Printed copy of data displayed at a terminal

or transmitted from the host can be provided by an attached printer.

Normally the application program supplies a "form" for the terminal operator to fill in and return. A formatted display screen image consists of protected fields containing guiding text and unprotected fields for operator input.

1.2 RC855 Cluster Concept 1.2

The RC855 terminal cluster does not contain a separate control unit (CU). Instead, one of the terminals in the cluster, the primary terminal, is programmed as a combined IBM 3270 BSC CU/Display Station, whereas the remaining, secondary terminals are programmed as IBM 3270 BSC Display Stations. The CU emulator program in the primary terminal provides compatibility in line discipline with any IBM 3270 BSC control unit which communicates using the EBCDIC character set and supports multiple devices with a buffer of 1920 characters, i.e. appropriate models of the IBM 3271, 3274, or 3276. Note that the term CU is generally used in this manual to refer to the logical, rather than the physical, control unit.

As many as eight RC855 terminals, including the primary, may be grouped for connection to a BSC communication line as an IBM 3270 cluster. The terminals in the cluster are interconnected by means of a simple twisted wire pair, called RC-CIRCUIT.

A printer can be attached to each terminal via a serial V.24 interface. (The printer is not logically connected to the terminal, but is available as a common resource). Thus, with a full complement of terminals and printers, a cluster will contain sixteen devices.

Each terminal in the cluster performs the following functions:

- o Editing of data on the display.
- o Processing of commands from the host.
- o Printing.

The primary terminal (CU emulator program) performs the following additional functions:

- o Remote BSC communication with the host.
- o Routing.
- o Status reporting.
- o Queue administration for all printers.

A secondary terminal can be removed from the cluster at any time, possibly to run another program, but the CU emulator program in the primary terminal must remain running as long as any terminal is operating as an IBM 3270 BSC Display Station.

1.3 RC855 Operation in Brief

1.3

The RC855 terminal is reset when power is applied, and thereafter whenever the RESET button is pushed. On being reset, the RC855 performs an automatic self-test. A program can then be loaded into its random-access memory and executed.

When loaded with the IBM 3270 BSC Emulator, the RC855 is functionally similar to the IBM 3277 Display Station Model 2. The RC855 keyboard, however, has a different layout and certain extensions when compared with the typewriter keyboard of the IBM 3277.

RC855 operator messages are displayed on the bottom line of the (25-line) screen, which is called the status line. Messages from the Emulator are displayed on the status line in inverse video to distinguish them from application-generated messages.

Configuration parameters for the individual RC855 terminal are stored in the terminal's nonvolatile memory. The current values of these parameters can be displayed and modified by means of a configurator program, which the operator can enter as soon as the Emulator is loaded. At any time it is possible to change the values of certain parameters temporarily as well as to enable or disable the reveal mode.

In the reveal mode, the cursor position and the contents of the character position identified by the cursor are displayed on the status line.

The operator can also run one or more diagnostic test programs as soon as the Emulator is loaded. At any time it is possible to perform system monitoring. A primary terminal can capture and display transmission sequences on the BSC communication line, including recognized polling/selection sequences for other control units, while secondary terminals can capture (but not display) sequences on RC-CIRCUIT. The Emulator also collects statistical information on system performance while it is running. This information can be displayed at any time.

A full description of RC855 operation is given in the operating guide for the RC855 IBM 3270 BSC Emulator [1].

2. OPERATIONAL CHARACTERISTICS 2.

The RC855 IBM 3270 BSC Emulator supports format oriented communication between the terminal operator and an application program running on a host computer. The basis of this communication is the transfer of a message from the application toward the terminal (outbound) or from the terminal toward the application (inbound). A message consists of a string of 8-bit characters. A character may be an alphameric or a control character, or it may be binary encoded (e.g. a buffer location) or encoded in subfields (e.g. an attribute character). The EBCDIC code sets (several national variants) used for transmission of data and control characters are given in appendix C.

For each device, i.e. terminal or printer, the Emulator maintains a device buffer containing 1920 character positions. For a terminal the buffer is organized as 24 lines, each containing 80 character positions. For a printer it is also possible to use other line lengths or no fixed line length. The image shown on a display screen as well as the lines printed on a printer are always determined by the contents of the relevant device buffer. Each character in a device buffer is either an attribute character or a data character.

An attribute character indicates the start of a display field extending to the next attribute character, and defines the attributes of the field:

- o alphameric input, numeric input, protected, or automatic skip
- o normal display, cursor selectable, intensified display and cursor selectable, or nondisplay
- o flashing or not flashing
- o modified or not modified

An attribute character occupies the first position of a display field, but appears as a blank position on the screen. The contents of an attribute character can be displayed in the reveal mode. Attribute characters are protected.

A field with the alphameric input or numeric input attribute is unprotected against manual input and therefore called an input field. All character positions following the attribute character in an input field are called input positions. A field which is not an input field is called a protected field. The modified/not modified bit in the attribute character is called the modified data tag, or MDT.

A data character may be any of the following: an alphameric character, null character, DUP character, FM character, or printer control character.

One should bear in mind the difference between the null character and the space character. Space is a well-defined alphameric character, whereas null is an all-binary-0 character. Both occupy a position in the device buffer and are displayed as a blank position, but null is suppressed in the transfer of a message.

The most common outbound messages are write-type commands, which instruct the Emulator to modify the contents of a device buffer in a specified manner. Similarly, most inbound messages are read messages which convey information from the Emulator about the current contents of a device (i.e. terminal) buffer, in particular fields that have been modified by the operator. The part of the message that contains detailed information about the device buffer consists of device buffer orders and device buffer data.

Device buffer orders in outbound messages are used to indicate the buffer locations of the following device buffer data, to set attribute characters, or to request some additional function (e.g. tabulation). In inbound messages, device buffer orders are used exclusively to indicate the buffer locations and attribute characters of transferred fields.

Device buffer data in outbound messages is that which is to be written in the device buffer as data characters. In inbound messages, device buffer data conveys the data characters in transferred fields.

Messages, commands, and orders are further described in chapters 3 and 4.

2.1 Display Terminal

Data Presentation

Data stored in the device (terminal) buffer is displayed on the screen in the form of alphameric characters. The terminal operator can enter alphameric characters into the buffer from the attached keyboard. The application program provides a formatted display by storing attribute characters in the buffer. The attribute characters in turn define the display fields, including those for which the operator can manually enter, modify, or erase data (input fields). As all locations in the buffer can be addressed individually, a display field can start at any character position.

Cursor

The cursor is a unique symbol that identifies a character position on the display screen, usually the location at which the next character to be entered from the keyboard will appear. The form of the cursor is defined by a configuration parameter. The movement of the cursor is described in detail in subsection 2.2.2.1.

Controls and Terminal Status Indication

The operator controls on the RC855 Display Terminal are described in the operating guide [1]. The status of the terminal is indicated by operator messages displayed on the bottom line of the screen. These messages are also described in the operating guide.

Configuration Parameters

The parameters defined by means of the configurator program include, among others, the following: the display device number of the terminal, the printer device number of the terminal's hardcopy printer, the printer device number of a printer physically attached to the terminal, and several parameters pertaining to an

attached printer (see section 2.3). Each secondary terminal, moreover, must be configured with a unique secondary address for RC-CIRCUIT, and the primary terminal of a cluster with a CU number. The configuration parameters are fully described in the operating guide [1].

Picture Timeout

A configuration parameter determines how long the display picture will remain on the screen when neither the host nor the operator is active.

2.2 Keyboard 2.2

The RC855 Display Terminal is provided with a separate, alphameric keyboard, which is attached by means of a signal cable one meter in length. A magnetic-stripe reader is available as optional equipment (see appendix H).

2.2.1 General Description 2.2.1

The keyboard consists of three main sections: a central, typewriter-like keyboard; above that, a row of attention keys; and to the right, a numeric pad and keys for editing data on the display. This general layout has a number of national variants, which are shown in appendix B.

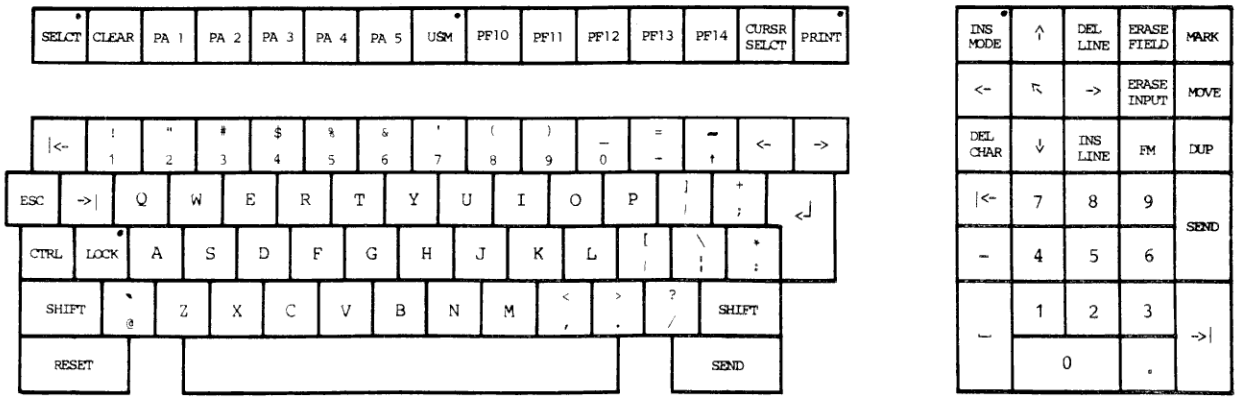


Figure 1: RC855 Keyboard Layout (US English Variant).

All code-generating keys repeat automatically, at the rate of 10 characters a second, when held depressed longer than three-quarters of a second. Some keys are furnished with a LED (light-emitting diode) indicator. The keyboard gives a click when a key is depressed properly. The keyboard also sounds an alarm in certain circumstances. Finally, the use of the keyboard is inhibited under certain conditions, as indicated by the character "x" displayed on the status line.

The keys are classified functionally as alphameric, auxiliary, attention, and editing keys. Alphameric, PA (program access), and PF (program function) keys have a light shade; all other keys have a dark shade.

2.2.2 Alphameric Keys

2.2.2

The alphameric keys generate codes corresponding to the alphabetic, numeric, or special character or characters with which each key is marked, either directly or in conjunction with the SHIFT key (see subsection 2.2.3). Alphameric keys are used to enter data for input fields. When data is entered into an input field, the modified data tag, or MDT, of the field is set.

Nonnumeric Keys

The nonnumeric keys are used to enter data for fields with the alphameric input attribute. If the cursor is positioned at a numeric input or a protected field, an attempt to enter data will cause an alarm. Otherwise the character is entered and the cursor advances (see subsection 2.2.2.1).

Numeric Keys

The numeric keys include the keys marked with the decimal digits, period (.), comma (,), and minus sign (-) and, on the numeric pad, the key marked with a period, which is called the "decimal point" key. These keys are used to enter data for fields with the numeric input or alphameric input attribute. If the cursor is positioned at a protected field, an attempt to enter data will cause an alarm. Otherwise the character is entered and the cursor advances (see subsection 2.2.2.1).

Note that the "decimal point" character can be defined as either the period or the comma character by a configuration parameter. A second parameter then determines whether the character not defined as the "decimal point" character is also a legal numeric character. A third parameter determines whether all characters or only numeric characters are allowed in numeric input fields.

Numeric Pad

The numeric pad contains a space key, keys marked with the decimal digits and the minus sign, and the "decimal point" key. Note that the codes for PF1 to PF9 (see subsection 2.2.4) are generated by pressing the numeric pad keys marked 1 to 9 while holding the SHIFT key (see subsection 2.2.3) depressed.

2.2.2.1 Automatic Cursor Movement 2.2.2.1

The cursor identifies a character position on the display screen, usually that at which the next character to be entered from the keyboard will appear. (Should a character already occupy that position, it will be overwritten). Whenever a character is entered from the keyboard, the cursor advances one character position. If a character is entered at the last position on a line, the cursor advances to the first position on the next lower line, and if a character is entered at the last position on the last line, the cursor advances to the first position on the first line. The cursor is said to wrap.

When a character is entered at the last position of an input field, the cursor advances to the position following the next attribute character, if the latter defines the field as having the alphameric input, numeric input, or protected attribute. If, however, the attribute character defines the field as having the automatic skip attribute (equivalent to protected and numeric), the cursor advances to the position following the next attribute character that defines an input field.

The cursor is unaffected by the transfer of a message to or from the application program (unless, of course, a device buffer order calling for cursor movement is received).

Editing keys that move the cursor are described in subsection 2.2.5.1.

2.2.3 Auxiliary Keys

2.2.3

The auxiliary keys, marked SHIFT, LOCK, and CTRL, perform various auxiliary functions, but do not themselves generate codes.

SHIFT

The SHIFT key can be used in conjunction with any alphameric key on the central keyboard to generate the upper case code for an alphabetic character key or the upper character code for a dual character key. In order to generate the desired code, the operator must press the alphameric key while holding SHIFT depressed. The SHIFT function terminates as soon as the keys are released. Note that the codes for PA6 to PA10 (see subsection 2.2.4) are generated by pressing the keys marked PA1 to PA5 while holding SHIFT depressed. Note also that the codes for PF1 to PF9 (see subsection 2.2.4) are generated by pressing the numeric pad keys marked 1 to 9 while holding SHIFT depressed. Note further that SHIFT is used in conjunction with the RESET key, as described in subsection 2.2.4. Additional uses of the SHIFT key are described in the operating guide [1].

LOCK

The LOCK key is used to place the central keyboard in, and remove it from, the alpha-lock mode (as well as to light and extinguish the indicator on the key). In the alpha-lock mode, the SHIFT function is applied automatically to all alphameric keys marked with (capital) alphabetic characters. The SHIFT key itself can, of course, be used in conjunction with all other alphameric keys while the central keyboard is in the alpha-lock mode. Note that the central keyboard can be placed permanently in the alpha-lock mode by means of a configuration parameter.

CTRL

The CTRL key is used in conjunction with the CLEAR key (see subsection 2.2.4) while the Emulator is running to reset the ter-

minal. This is done by pressing CLEAR while holding CTRL depressed. The CTRL key is also used in conjunction with attention keys for diagnostic purposes, as described in the operating guide [1].

2.2.4 Attention Keys

The depression of an attention-generating key causes an attention to be transferred to the CU emulator program, either via RC-CIRCUIT or internally in a primary terminal. The kind of attention is identified by an attention identification character, or AID character (see subsection 4.1.1.2).

A remote attention is generated when a data attention or a short attention key is pressed. This signals to the CU that a read message is to be transmitted when the terminal is polled (specific or general poll). The read message is said to be pending in the CU until it has been transmitted. It is prepared by executing an implicit Read Modified command (see subsection 4.1.1.4). In the case of a data attention, the CU requests the terminal to execute the Read Modified command, thus constructing a message comprising the relevant AID character and the contents of all display fields in which the MDT is set. This message is called a long read. In the case of a short attention, the message, which is called a short read, contains only the relevant AID character and is generated by the CU without involving the terminal. Read messages are further described in chapters 3 and 4.

A local attention is used to initiate a function that is local to the terminal cluster and does not cause any transmission to the application program.

Note that when an attention key is pressed, the use of the keyboard is inhibited until it is restored by the application program or, in the case of a local attention, by the CU. Keyboard inhibition is indicated by the character "x" displayed on the status line.

CLEAR — Short attention key. Causes the terminal buffer to be cleared to nulls (thus erasing the entire display screen, protected fields included) and the cursor to be moved to the first character position on the first line. Note that CLEAR is used in conjunction with the CTRL key, as described in subsection 2.2.3, while the Emulator is running to reset the terminal.

PA1-PA5 — Short attention keys. Note that the AID codes for PA6 to PA10 are generated by pressing the keys marked PA1 to PA5 while holding the SHIFT key (see subsection 2.2.3) depressed. Each PA, or program access, key can be defined to solicit program action that does not require data to be read from the terminal buffer.

USM — Short attention key. The indicator on the key lights when the terminal receives a USM, or unsolicited message, order in a write-type command. Depression of the key will also extinguish the indicator.

PF10-PF14 — Data attention keys. Note that the AID codes for PF1 to PF9 are generated by pressing the numeric pad keys marked 1 to 9 while holding the SHIFT key (see subsection 2.2.3) depressed. Each PF, or program function, key can be defined to solicit program action that requires data to be read from the terminal buffer.

CURSR SELCT — Data attention key (see appendix H).

PRINT — Local, copy request attention. The terminal is queued for a local print operation on its hard-copy printer. The use of the keyboard is inhibited until the contents of the terminal buffer have been transferred to the printer buffer (see further section 2.3).

RESET — The AID code for RESET can be generated only if RESET is pressed while the SHIFT key (see subsection 2.2.3) is held depressed.

Local, regret attention. If a read message generated by the terminal is pending in the CU, the message is cancelled and the use of the keyboard is restored.

RESET is used analogously to cancel the latest copy request attention (see PRINT, above) and restore the use of the keyboard.

RESET is also used to cancel marks produced by the MARK key (see subsection 2.2.5.2).

SEND — Data attention key. SEND is used in the same manner as a PF key, viz. to solicit program action that requires data to be read from the terminal buffer.

For the uses of SELCT and ESC as well as additional uses of the CLEAR, PA, PF, PRINT, RESET, and SEND keys, e.g. for diagnostic purposes, see the operating guide [1].

2.2.5 Editing Keys 2.2.5

Editing operations can be performed as soon as the Emulator has been loaded, as connection to a host computer is not required. During the entry of data for a formatted display, the operator can edit the data until he presses an attention key (see subsection 2.2.4).

2.2.5.1 Keys to Move the Cursor 2.2.5.1

The keys described in this subsection enable the operator to move the cursor to any character position on the display screen without erasing the characters which the cursor passes. All of the keys repeat automatically as described in subsection 2.2.1.

← Left (Backspace)

Moves the cursor one position left. The cursor may wrap horizontally, e.g. from the first position on the first line to the last position on the last line.

→ Right

Moves the cursor one position right. The cursor may wrap horizontally, e.g. from the last position on the last line to the first position on the first line.

↑ Up

Moves the cursor one position up. The cursor may wrap vertically, e.g. from the first line to the last line (with no horizontal movement).

↓ Down

Moves the cursor one position down. The cursor may wrap vertically, e.g. from the last line to the first line (with no horizontal movement).

↖ Home

Moves the cursor to the first input position of the first input field on the display screen. If no input field is found, the cursor is moved to its home position, i.e. the first position on the first line (character position 0).

|← Field Backward (Backtab)

When the cursor is at the attribute character or the first input position of an input field or at any position of a protected field, the cursor is moved to the first input position of the preceding input field. When the cursor is at any input position other than the first of an input field, the cursor is moved to the first input position of that field. (The cursor may wrap). If no input field is found, the cursor is moved to its home position.

→| Field Forward (Tab)

Moves the cursor to the first input position of the succeeding input field. (The cursor may wrap). If no input field is found, the cursor is moved to its home position.

↵ New Line (Return)

Moves the cursor to the first input position on the next lower or a succeeding lower line. (The cursor may wrap). If no input field is found, the cursor is moved to its home position.

2.2.5.2 Keys to Edit the Data

2.2.5.2

The keys described in this subsection enable the operator to edit the contents of input fields. Note that an attempt to perform an illegal editing operation (e.g. when the cursor is positioned at a protected field or an attribute character) will cause an alarm.

INS MODE Insert Character Mode

Used to place the terminal in, and remove it from, the insert character mode (as well as to light and extinguish the indicator on the key). The operation requires the presence of a null character in the field, either at the cursor position or to the right of it. A character entered from the keyboard will appear at the cursor position, and a nonnull character already occupying that position will be shifted horizontally one position to the right, together with all succeeding characters except a null or characters to the right of a null.

DEL CHAR Delete Character

Erases the character at the cursor position, and shifts all succeeding characters horizontally one position to the left. A null character is inserted at the last position on the line or in the field, whichever is encountered first.

INS LINE Insert Line

Applicable only to fields containing at least one line, i.e. 80 character positions. The 80 positions may begin at the attribute character or constitute an entire display screen line. The operation vertically shifts the line at which the cursor is positioned and all succeeding lines in the field one line down, thus providing a line of nulls at

the cursor position. The last line (80 positions) is deleted.

DEL LINE — Delete Line

Applicable only to fields containing at least one line, i.e. 80 character positions. The 80 positions may begin at the attribute character or constitute an entire display screen line. The operation deletes the line at which the cursor is positioned and vertically shifts all succeeding lines in the field one line up, thus providing a line of nulls as the last line (80 positions).

ERASE FIELD — Erase to End of Field

Erases all characters in the field from and including the character at the cursor position. The cursor is not moved.

ERASE INPUT — Erase Input Fields

Erases all input fields, and moves the cursor to the first input position of the first input field on the display screen. If no input field is found, the cursor is moved to its home position. The MDT of all input fields is reset.

FM — Field Mark

Causes an FM character code to be entered into the terminal buffer. The FM character, which provides a means of informing the application program of the end of a subfield, is displayed (or printed) as the character ";".

DUP — Duplicate

Causes a DUP character code to be entered into the terminal buffer, and moves the cursor to the first input position of the succeeding input field. The DUP character, which provides a means of informing the application program that a "duplicate" operation is indicated for the remainder of the field in which it has been entered, is displayed (or printed) as the character "*".

MARK Mark String

Used to mark the first and the last characters of a string to be moved by means of the MOVE key (see below). The string may not contain an attribute character. When the cursor has been positioned at a character, the depression of the MARK key will cause a frame to appear around the character. Such markings may be cancelled by means of the RESET key (see subsection 2.2.4).

MOVE Move String

Moves a character string marked by means of the MARK key to the display screen location determined by the cursor position. The string is inserted starting at the cursor position, and characters already occupying these positions are overwritten. It is not possible to overwrite an attribute character in this fashion, nor to copy data into a protected field.

2.3 Printers

2.3

A printer can be attached via a serial V.24 interface to each terminal in an RC855 cluster, as described in section 1.2. The parameters defined by means of the configurator program include the following for an attached printer:

- Printer device number (0 when no printer is attached).
- Local device only, i.e. not available and not reported to the host.
- Bit rate of transmission to the printer: 110, 300, 600, 1200, 2400, 4800, or 9600 bps.
- Format of characters transmitted to the printer, e.g. even parity, two stop bits.
- Maximum print line: up to 132 characters in length.
- Printing mode: full-image or compressed, i.e. whether display screen images are to be printed with or without all-null lines and form feed characters.

Note that the above parameters apply to the printer which is physically attached to the terminal, and that the terminal must be running the Emulator in order for the printer to be used (e.g. by another terminal as its hard-copy printer).

A 7-bit code set, derived from ISO 646, is used for printing on printers attached via a serial V.24 interface. The national variants of this code set are given in appendix E. The RC conventions for the use of V.24 signals are given in appendix F.

2.3.1 Print Formats 2.3.1

Two types of print formats can be used:

- Transparent formats with a maximum print line length as preset by the maximum print line, or MPL, parameter. New line and other printer control characters can be inserted in the printer buffer to control formatting.

- Fixed formats with 40, 64, or 80 characters per line. Control characters to cause the printer to move to a new line are generated automatically. Limited support is provided for printer control characters.

Regardless of the type of print format, attribute characters and all characters, including printer control characters, which occur in fields with the nondisplay attribute are treated as nulls during a printing operation, i.e. they are printed as blank positions or completely suppressed. The suppression of all-null lines occurs only when compressed printing is specified by the printing mode, or PM, parameter, which is applicable only to fixed print formats and (remote or local) copying operations.

2.3.2 Printer Control Characters 2.3.2

In conjunction with a Write, Erase/Write, or Copy command (see subsections 4.1.2 and 4.1.3), printer control characters can be

written as data characters in a printer buffer, and stored there until a printing operation begins. Printer control characters are received from the host computer as EBCDIC characters. These characters are: new line (NL), carriage return (CR), vertical tabulation (VT), form feed (FF), and end of message (EM).

The general effect, if any, of a printer control character encountered in the printer buffer during a printing operation is that one or more control characters are generated and sent to the printer. The latter control characters, which are 7-bit ISO characters (see appendix E), are denoted here by the prefix ISO- in order to distinguish them from their EBCDIC counterparts.

The specific effects of printer control characters during a printing operation depend on the print format, as described below.

2.3.2.1 Transparent Print Formats 2.3.2.1

NL Causes the carriage return and line feed characters (ISO-CR and ISO-LF) to be generated. If NL (or CR) is missing, ISO-CR and ISO-LF will be generated automatically when the maximum print line length is attained.

CR Causes the carriage return character (ISO-CR) to be generated.

VT Causes the vertical tabulation character (ISO-VT) to be generated.

FF Causes the form feed character (ISO-FF) to be generated.

EM Causes the carriage return and line feed characters (ISO-CR and ISO-LF) to be generated, unless the EM character immediately follows an NL character. Subsequently terminates the printing operation. If no EM character is present, the printing operation terminates when the end of the buffer is reached. At this time carriage return and line feed charac-

ters (ISO-CR and ISO-LF) are generated as if an EM character had been encountered.

2.3.2.2 Fixed Print Formats 2.3.2.2

NL Treated as null.

CR Treated as null.

VT Causes the vertical tabulation character (ISO-VT) to be generated, provided that the VT character is the first character in a line; otherwise treated as null.

FF Causes the form feed character (ISO-FF) to be generated, provided that the FF character is the first character in a line; otherwise treated as null.

EM Treated as null.

2.3.3 Remote Printing Operations 2.3.3

Printing operations can be initiated by the application program when the start printer bit is set in the write control character (WCC) in a Write or Erase/Write command or the copy control character (CCC) in a Copy command addressed to a printer (see subsections 4.1.2 and 4.1.3). A transparent or a fixed print format is used, as specified by the print format bits in the WCC or the CCC. Compressed printing (PM parameter) may be used in copying operations with a fixed format.

2.3.4 Local Copying 2.3.4

A local print operation is initiated by means of the PRINT key, as described in subsection 2.2.4. The image displayed on the screen is printed on the printer defined by a configuration parameter as the terminal's hard-copy printer. The operation is en-

tirely local, controlled by the CU emulator program in the primary terminal, and connection to a host computer is not required.

A local printing operation always uses a fixed print format with 80 characters per line, so that the printed page corresponds to a display screen image. Compressed printing (PM parameter) may also be used. Spaces or nulls at the end of a line are suppressed. Nulls, when not suppressed, are printed as spaces. The cursor is not printed.

The printer can be busy, printing for another terminal or the application program. Local copy requests are queued accordingly on a FIFO (first-in-first-out) basis. A local copy request can be cancelled by means of the RESET key (see subsection 2.4.4).

2.3.5 Specific Operational Characteristics 2.3.5

The specific operational characteristics of the printers available for the RC855 Display Terminal are described in other RCSL publications. Note that any printer to be used in conjunction with the Emulator must support the ISO control characters CR, LF, VT, and FF.

2.3.6 Résumé 2.3.6

Printing Operation	Print Format	Applicable Parameters MPL	PM	Support of Printer Control Characters
remote (WCC)	transparent	yes	no	full
	fixed	no	no	limited
remote (CCC)	transparent	yes	no	full
	fixed	no	yes	limited
local (PRINT)	fixed (80)	no	yes	not applicable

3. COMMUNICATION WITH THE HOST COMPUTER 3.

The RC855 IBM 3270 BSC Emulator, as its name implies, supports the binary synchronous communication, or BSC, line control protocol for remote communication with a host computer (typically handled by a front-end processor).

A modem can be attached to any RC855 Display Terminal via a serial V.24 interface. This facility is used by the primary terminal, which is programmed as a combined IBM 3270 BSC CU/Display Station, for connection of the device cluster (see section 1.2) to a BSC communication line. The line may be 2-wire or 4-wire (half or full duplex), and the line speed up to 9600 bps. No strapping or reconfiguration is required in the primary terminal to adapt to half or full duplex.

Several remote 3270 clusters can be attached to a host computer by means of a single (multipoint) line, to which other, non-3270 terminal systems may also be connected.

3.1 Communication in General 3.1

The host computer controls the flow of communication between itself and devices on the line by means of a polling/selection system, i.e. the host polls the devices for pending inbound messages and selects a particular device for receipt of an outbound message. A pending inbound message is either a read message (see subsection 2.2.4) or a status message (see section 3.5). Outbound messages are commands (see chapter 4).

Each device on the line has a unique address, which is used in conjunction with polling or selection operations: the CU number, in the range 0 through 31, which identifies the cluster on a multipoint line, concatenated with the device number, in the range 0 through 62, which identifies the device within the cluster.

All data and all control characters are transmitted in EBCDIC code (see appendix C). The line control characters are described in section 3.2.

The polling/selection technique allows the host computer to communicate intermittently with several 3270 control units. The term CU is often used loosely as a synonym for control unit; in the present description, CU refers to the logical unit which communicates with the host computer on the protocol level.

At any point in time, the CU is either engaged in a dialog with the host (IBM text mode) or monitoring the line (IBM control mode) for a polling or a selection sequence. A dialog is an exchange of transmission sequences (see section 3.2). A dialog begins when the CU receives a valid polling or selection sequence from the host. The dialog ends when either party sends an EOT sequence.

All legal dialogs are shown on figs. 2 through 7 in this chapter. Any response from the CU which is not shown on one of these figures is undefined.

The general receive timeout, when either the CU or the host is awaiting a response, is 3 seconds, i.e. the host should not, and the CU will not, request retransmission until at least 3 seconds have elapsed.

Since the terminal system may share the line with other, non-3270 systems, the CU has the capability to discard transparent text (IBM transparent monitor mode) when monitoring the line for polling or selection sequences.

3.2 Transmission Sequences 3.2

Line control characters, as this term is commonly used, comprise two distinct kinds of characters: those which in themselves constitute a transmission sequence in a dialog and those which are merely elements in such sequences. This distinction is reflected in the following description. Note that reference is made here to several concepts which are explained in section 3.3, notably master/slave, text blocking, and transmission control, while status information is explained in section 3.5.

DLE Data Link Escape

The first character in the two-character sequences ACK 0, ACK 1, RVI, and WACK.

ENQ Enquiry

The last character in a polling or a selection sequence. Also sent by the host as the last character in a TTD sequence (see subsection 3.2.3). ENQ also constitutes a sequence in itself (see subsection 3.2.3).

ETB End of Transmission Block

The last character in any but the last block of a text. ETB is followed by the block check sequence and treated as ETX.

ETX End of Text

The last character in the last or only block of a text. ETX indicates the end of the message. ETX is followed by the block check sequence. When the CU receives ETX, a block check is made, and the CU responds accordingly with ACK 0, ACK 1, WACK, NAK, or a text block. When the host receives ETX, a block check is made, and the host responds accordingly with ACK 0, ACK 1, NAK, RVI, WACK, or a write-type or a control-type command.

ITB End of Intermediate Transmission Block

The CU accepts the ITB character, but makes no separate block check on its reception. ITB and the following block check sequence are included in the BCC accumulation, but are not part of the received message. Note that ITB is designated IUS in the tables in appendix C.

SOH Start of Heading

The first character in a text block containing a status message. SOH is the first character in the message header, a three-character sequence used to identify such messages. BCC accumulation begins after SOH.

STX Start of Text

The first character in any text block, unless the text block contains a status message. STX indicates the start of the message. BCC accumulation begins after STX, when STX is the first character in the text block.

3.2.2 Fillers and Delimiters 3.2.2

PAD Pad

Dummy (all-binary-1) character, used to ensure complete transmission and reception of the first and the last character in a transmission sequence.

SYN Synchronous Idle

Every transmission sequence begins with at least two consecutive SYN characters, which are used to achieve and maintain character synchronism. If SYN is used as a time-fill character in a text block, it is not part of the received message.

3.2.3 Sequences 3.2.3

A transmission sequence is a sequence of characters transmitted as an entity by the host toward the CU or by the CU toward the host. All such sequences are delimited by PAD characters and begin with at least two SYN characters. The following transmission sequences can occur in a dialog between the host and the CU:

- polling/selection
- text block
- ACK 0/1, WACK, and RVI
- NAK
- EOT
- ENQ
- TTD

Polling and selection sequences are described in section 3.4, and text block sequences in section 3.3. The remaining sequences are described below.

ACK 0 Even Positive Acknowledgement

Two-character sequence (DLE and 70_{Hex}). Sent by the CU in response to a valid selection sequence, thereby indicating that the addressed device is ready to receive a message. Also sent by the CU in response to a valid, even-numbered text block. Sent by the host in response to a valid, even-numbered text block.

ACK 1 Odd Positive Acknowledgement

Two-character sequence (DLE and 61_{Hex}). Sent by the CU in response to a valid, odd-numbered text block. Sent by the host in response to a valid, odd-numbered text block.

ENQ Enquiry

One-character sequence. Sent by the master to request retransmission of an ACK or a NAK following a timeout. Sent by the CU in response to WACK.

EOT End of Transmission

One-character sequence. Informs the slave of the end of a transmission. EOT ends the dialog. When the CU receives EOT, it returns to monitoring the line. EOT is sent to the host: as the normal conclusion of a read operation when the CU is the master, which applies in particular when a general poll has progressed through all devices attached to a control unit and no pending inbound messages were found; when the CU is the slave and cannot perform the requested operation; or as a response to RVI.

NAK Negative Acknowledgement

One-character sequence. Sent by the CU in response to a text block when: the text block was terminated as a TTD sequence (see below); the block check revealed an error; or STX was missing. Sent by the host to request retransmission of a text block.

RVI Reverse Interrupt

Two-character sequence (DLE and $7C_{Hex}$). Sent by the slave to prevent further transmission from the master. Sent by

the CU in response to a selection sequence when the addressed device has a pending status message. When RVI is received from the host, the CU responds with EOT.

TTD Temporary Text Delay

Sequence comprising at least the two characters STX and ENQ, i.e. any text block terminated by ENQ constitutes a TTD sequence. When the host is unable to transmit a text block within roughly 2 seconds, it can send ENQ in order to maintain the connection with the selected device. ENQ thus prevents the occurrence of a 3-second receive timeout in the CU, which now responds with NAK.

WACK Wait before Transmit

Two-character sequence (DLE and $6B_{Hex}$). Sent by the CU in response to a valid selection sequence to indicate that the selected device is busy and therefore unable to receive a message. Also sent by the CU in response to a valid write-type or control-type command to a printer when the start printer bit in the WCC or CCC (see chapter 4) is set. When WACK is received from the host, the CU responds with ENQ.

3.3 Text Transfer 3.3

All messages, whether inbound or outbound, are transmitted as BSC text. A text is a character sequence which is treated as an entity when it occurs between an STX and an ETX line control character.

3.3.1 Master and Slave 3.3.1

When engaged in a dialog, the host and the CU take turns as the master station and the slave station. The sender of a message is the master; the receiver of the message is the slave.

The CU becomes the master whenever it transmits a message, i.e. in response to a polling sequence or a message comprising a read-

type command. As the master, the CU can send ENQ to request a response after a receiving timeout or to request retransmission of an ACK or a NAK.

The CU becomes the slave whenever it receives a message, i.e. a write-type or a control-type command. As the slave, the CU must respond in a prescribed manner to transmissions from the master.

3.3.2 Text Blocking 3.3.2

Inbound messages are transmitted in text blocks. A block can contain a maximum of 256 characters, including the line control characters (STX, possibly SOH, and ETX or ETB). As the receipt of a text block is acknowledged by the slave station, each block transfer requires two line turnarounds. SF and SBA order sequences (see chapter 4) are not divided between blocks. The maximum length of any text is 3840 characters.

3.3.3 Transmission Control 3.3.3

A cyclic redundancy parity/error check, or CRC, is made on each text block. The block check character, or BCC, which is a 16-bit sequence, is transmitted immediately after the ETX or ETB character. The BCC is not part of the received message, as it is immediately separated from the block.

BCC accumulation takes place at both stations, and the master sends its BCC at the end of the block for comparison with the BCC of the slave. BCC accumulation begins when an SOH or an initial STX character is encountered. The accumulation does not include SOH or an initial STX, but all following characters to and including ETX or ETB.

When a CRC error is detected by the slave, the latter can request retransmission of the block by responding with NAK.

The BSC protocol provides neither cyclic nor longitudinal redundancy checking on polling and selection sequences. Therefore, to ensure data integrity in such sequences, the CU address and the device address are repeated for comparison on reception.

The matters so far discussed are elucidated by the following two figures.

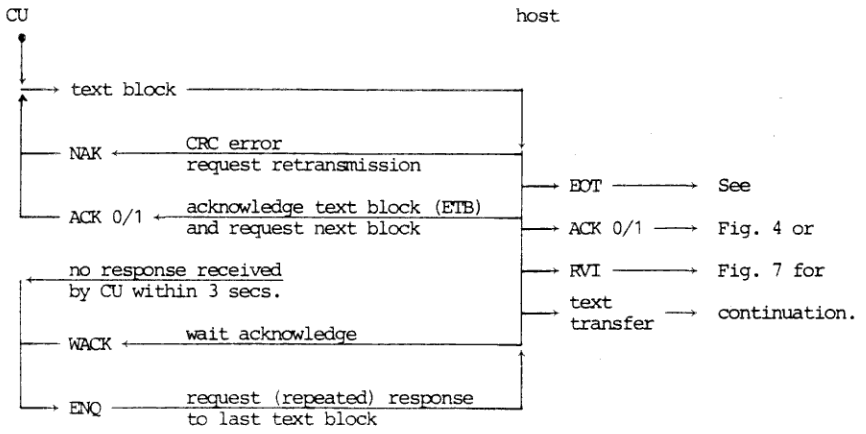


Figure 2: Text Transfer from CU (Master) to Host (Slave).

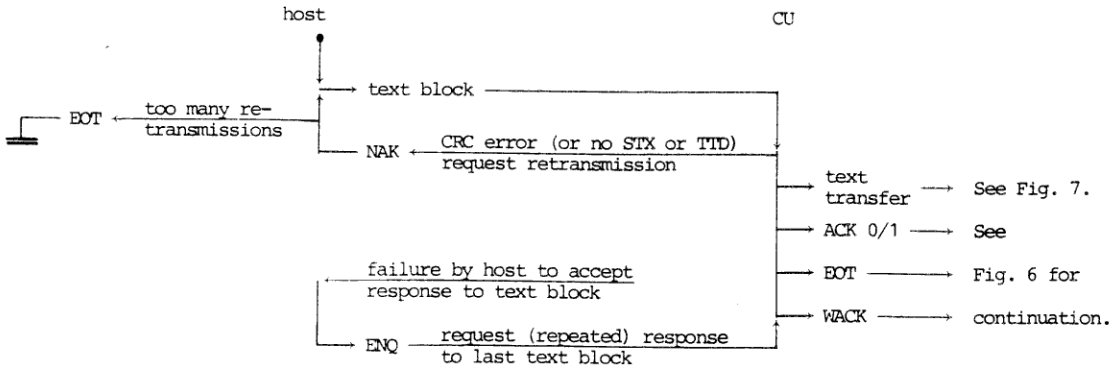


Figure 3: Text Transfer from Host (Master) to CU (Slave).

3.4 Dialog Patterns 3.4

All dialogs between a CU and the host computer are initiated by the host, which either polls a cluster or a specific device for pending inbound messages or selects a device for execution of a command. Depending on the type of polling or selection sequence and on the states of the device or devices involved, the subsequent dialog will follow one of a number of different patterns. These patterns are described in the present section.

3.4.1 Poll-Initiated Dialogs 3.4.1

In a specific poll, the host addresses the control unit and a specific device. In a general poll, the host addresses the control unit, but not a specific device. The control unit address used in a polling sequence is the CU number. The device address used in a specific poll is the device number, whereas the device address used in a general poll is the number 63, which cannot be the number of a device. A polling sequence consists of five characters:

CU CU DEV DEV ENQ

where the CU and DEV characters are derived from the control unit and the device addresses according to the rule described in appendix D. The CU and DEV characters must be identical in pairs, otherwise the control unit will not recognize the polling sequence.

When performing a polling operation on a selected device, the CU will transfer any pending inbound message from the device to the host. A pending inbound message may be a short or a long read message (see subsection 2.2.4) or a status message (see section 3.5). Long read messages are generated at the request of the CU in the individual display terminals, whereas short read messages and status messages are generated internally in the CU. The host, however, is unaware of this division of tasks.

A specific poll causes the CU to select the addressed device and perform a polling operation. A general poll causes the CU to select each device in turn and perform a polling operation. The host, however, can force termination of a general poll at any time by sending RVI instead of ACK 0/1 in response to a message.

A device can remain selected, after transmission of a message in response to a poll, because the host wishes to continue with a chained write-type or control-type command without further addressing. Thus, when the host has received ETX, it sends a command either to the device already selected by a specific poll or to the device currently selected in a general poll. This terminates the polling operation, and any remaining responses to a general poll will not be transmitted until the next poll.

This general outline is amplified by fig. 4, which shows the patterns of dialogs initiated by polling sequences.

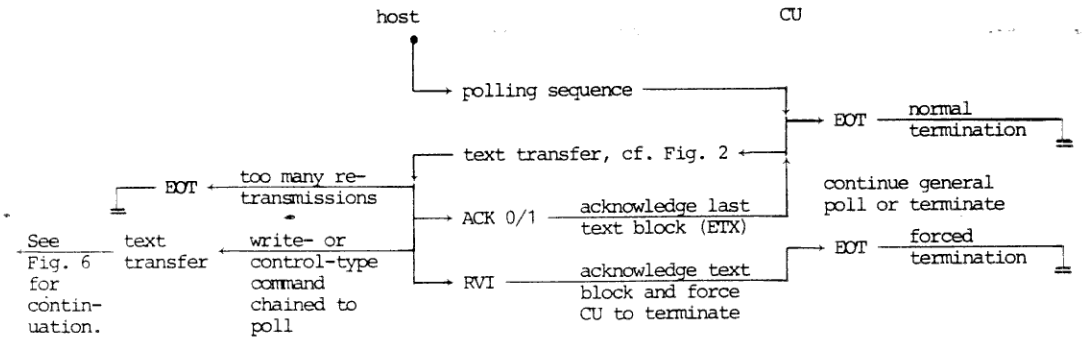


Figure 4: Poll-Initiated Dialogs.

When a pending read message has been transferred, it is no longer pending, and the read operation will not be performed again; this is also true, if the host aborts the message transfer prematurely by responding with EOT, RVI, or a chained command before the final text block (ETX) has been received.

3.4.2 Selection-Initiated Dialogs

An outbound message, which always comprises a command, is normally preceded by a selection sequence allowing the host computer to determine whether the device for which the message is intended is able to receive it. The selection sequence is like the specific polling sequence, except that 32 is added to the control unit address to indicate selection rather than polling.

When a device receives a valid selection sequence, it can respond in one of three ways: RVI indicates that the device is unavailable or offline (see section 3.5) and therefore unable to receive a command; WACK indicates that the device is busy and therefore unable to receive a command; and ACK 0 indicates that the device can receive the command.

On receiving RVI or WACK, the host responds with EOT, thereby ending its attempt to select the device. When the host accepts ACK 0, the device is selected, and the dialog continues with transmission of the message containing the command.

The patterns of dialogs initiated by selection sequences are shown in fig. 5.

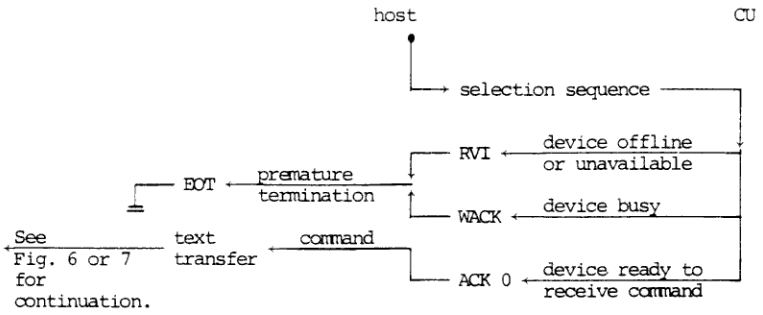


Figure 5: Selection-Initiated Dialogs.

3.4.3 Write-Type and Control-Type Commands 3.4.3

A write-type or a control-type command is normally preceded by a selection sequence, but it can also be chained to a poll or a preceding write-type or control-type command. After a positive acknowledgement from the device, which indicates that the command has been executed, the host computer can either terminate the dialog or send an additional (chained) command to the same device.

The write-type commands are Write and Erase/Write. The control-type commands are Copy and Erase All Unprotected. All commands are fully described in chapter 4.

Fig. 6 shows how write-type and control-type commands are transferred as part of a dialog.

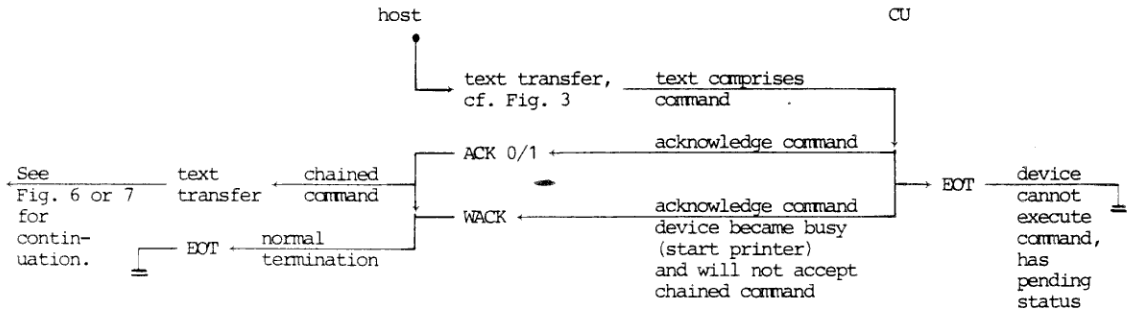


Figure 6: Write-Type or Control-Type Command.

3.4.4 Read-Type Commands 3.4.4

A general or a specific poll, as described in subsection 3.4.1, will initiate a read operation, if the selected device has a pending read message. The host computer can also send an explicit read-type command to a selected device, i.e. following a selection sequence or a write-type or control-type command to which the command is chained. To perform the read operation, the host first transmits the command and then reads the received text (see chapter 4).

The read-type commands are Read Buffer and Read Modified. Both commands are fully described in chapter 4.

Fig. 7 shows the effect of a read-type command in a dialog.

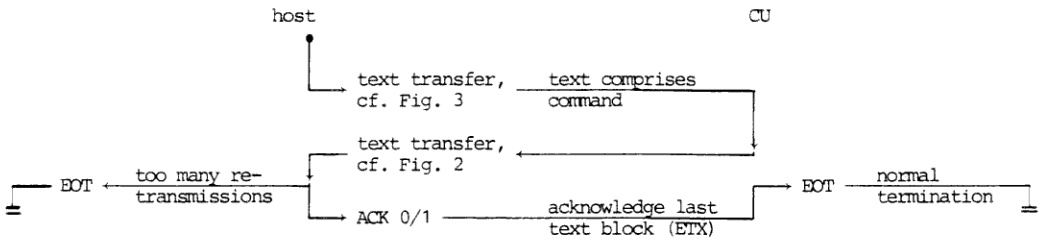


Figure 7: Read-Type Command.

3.4.5 Chained Commands 3.4.5

Command chaining occurs: when two or more commands are sent to a device following a selection sequence; or when the host computer responds to a read message, transferred as a result of a general or specific poll, with a command. Thus the host, for example, can read a message from a polled device and immediately respond with a write-type or a control-type command, followed by another command.

All types of commands can be chained in conjunction with a selection sequence, whereas only write-type and control-type commands can be chained in conjunction with a polling sequence.

All commands save Erase All Unprotected can be chained to a Write, Erase/Write, or Copy command. Note that when a command causes the printer to be started, the device will respond with WACK to indicate that a chained command will not be accepted (cf. fig. 6).

3.5 Status Information 3.5

A number of events pertaining to individual devices within a terminal cluster can cause the CU to transfer a status message in response to a polling sequence. Among these events are the following:

- o A device becomes available.
- o A printer completes a printing operation or becomes ready after having been offline.
- o A printer goes offline during a printing operation.
- o An unavailable device or a printer which is busy or offline is selected.
- o An invalid command is received.

The form of a status message is:

SOH % R STX CU DEV S/0 S/1 ETX

The CU, DEV, S/0, and S/1 characters are all derived from 6-bit quantities in the fashion described in appendix D. These 6-bit quantities are: the CU number, the device number of the device to which the message pertains, status0, and status1.

The bit assignments and definitions for status0 and status1, from the least to the most significant position, are as follows:

status0	status1
0: (not used)	0: OC
1: DE	1: (not used)
2: US	2: (not used)
3: DB	3: (not used)
4: (not used)	4: IR
5: (not used)	5: CR
7-6: See appendix D.	7-6: See appendix D.

The bits are named according to IBM conventions, but only the abbreviations given above are used in this section. The unused bits are always 0. Note that IBM designates the most significant bit position 0.

The types of status messages which can occur depend on whether a general or a specific polling sequence is received.

A status message is transferred in response to a general poll, only if a device has become available, a printer has completed a printing operation initiated by the host, or a printer has become ready after having been offline.

A specific polling sequence addressing a printer is always answered with a status message. A specific polling sequence addressing a display terminal is answered with a status message, only if the device is unavailable or the polling sequence immediately follows an invalid command (answered with EOT; cf. fig. 6) which caused status to be generated for the device.

The various types of status messages which can be sent by the CU are described below. For each type of message, the bits set in status0 and status1 are indicated, together with the reason and the occasion for sending the message.

3.5.1 Device Unavailable Status Message 3.5.1

device unavailable — bit set: IR

Sent when an unavailable device is selected.

When the Emulator is loaded into a secondary terminal, and the latter begins to communicate with the primary terminal on RC-CIRCUIT (see section 1.2), the terminal (3270 display station) and its attached printer, if present, become available devices. When the terminal ceases to communicate on RC-CIRCUIT, it becomes unavailable (as does its attached printer). This normally occurs when the terminal is powered down or a program other than the Emulator is loaded.

3.5.2 Status Messages Used Only for Printers 3.5.2

printer ready and not busy bit set: DE

Sent in response to a general poll when a device has become available or when a printer has successfully completed a printing operation or has become ready after having been offline. Sent in response to a specific poll addressed to a printer when no other status applies.

printer busy bit set: DB

Sent when a printer which is currently performing a printing operation initiated by the host is selected.

printer offline bits set: DE and IR

Sent in response to a general poll when a printer has completed a printing operation initiated by the host, but timed out or went offline (see appendix F) during the operation. Sent in response to a specific poll when the printer is offline. A printer which is performing a local printing operation is considered offline.

3.5.3 Status Messages Resulting from an Invalid Command 3.5.3

The following types of status messages must be fetched by a specific poll immediately following the invalid command. All commands are fully described in chapter 4. If the device is reselected before the status message has been retrieved, the status will be reset.

command error bit set: CR

Occurs when the ESC character is lacking or the CMD character is unknown to the CU.

The remaining status messages, all of which are occasioned by an invalid Copy command, pertain to the device to which the command

was addressed, i.e. the "to" device, even though the status message may refer to the state of the "from" device.

copy busy — bits set: DB and OC

Occurs when the host attempts to copy from the buffer of a busy printer.

copy unavailable — bits set: IR and OC

Occurs when the host attempts to copy from an unavailable device.

copy locked — bits set: US and OC

Occurs when the host attempts to copy from a device with a locked buffer (see subsection 4.1.3.1).

copy command error — bit set: OC

Occurs when the "from" device address or the copy control character is missing.

4. COMMANDS AND RESPONSES 4.

The commands which the host computer can send to a selected device are described in the present chapter. Particular emphasis is placed on the text contained in commands as well as on that contained in messages sent in response to commands. Device buffer orders, which occur principally in write-type commands, are described in detail in section 4.2. Seeing that reference is made throughout the chapter to buffer addresses, it would be well to consider them first.

A buffer address is the number of a position in a device buffer. The positions are numbered from 0 to 1919. For a display terminal, the buffer positions correspond to character positions on the screen in the normal textual ordering, i.e. from left to right within each line, and from top to bottom. Thus the buffer address corresponding to character position (line, column), where line is numbered 0 to 23 from top to bottom, and column is numbered 0 to 79 from left to right, is 80 x line + column.

Buffer addresses are transmitted as part of a command or a read message in several instances (viz. as cursor positions and in SBA, RA, and EUA orders). A buffer address is always transmitted as two characters: ADDR1 and ADDR2. Being in the range 0 through 1919, a buffer address can be represented in binary form using 12 bits. (Actually, 11 bits will suffice, as the high-order bit is always 0). The ADDR1 and ADDR2 characters are obtained by applying the rule described in appendix D to the high-order and the low-order 6 bits, respectively, of this 12-bit representation.

4.1 Commands 4.1

Commands are sent by the host computer to a selected device in order to initiate various operations. The commands are of three types:

- Read-type commands, which transfer read messages to the host, especially messages that contain information about the current contents of the device buffer.

- Write-type commands, which give instructions for modifying the contents of the device buffer in a specified manner.

- Control-type commands, which initiate operations that do not require the transfer of device buffer data between the host and the device.

Commands are transferred from the host to the CU as messages, packaged as BSC text. The general form of a command is:

$$[\text{command message}] = \text{ESC CMD } [\text{command body}]_0^1$$

The command type is identified by the CMD character. The contents of the command body, when present, depend on the command type.

4.1.1 Read-Type Commands

The read-type commands (and their CMD characters) are: Read Buffer ($F2_{Hex}$ corresponding to the graphic "2") and Read Modified ($F6_{Hex}$ corresponding to the graphic "6"). Read Buffer is used to transfer messages containing information about the entire contents of the buffer, whereas Read Modified is used primarily to transfer messages containing information about buffer contents which have been modified.

Read operations are normally performed automatically in conjunction with general and specific polls, but the host can also initiate a read operation by sending an explicit read-type command to a selected device.

The general form of a read message (possibly blocked) transferred in response to a polling sequence or an explicit read-type command is:

$$[\text{read message}] = \text{CU DEV AID } [\text{response body}]_0^1$$

The device in question is identified by the CU and DEV characters, which are formed in the same manner as described for a specific polling sequence (see subsection 3.4.1). The AID character is described in subsection 4.1.1.2. The contents of the response body, which may be absent, depend on the type of read operation.

4.1.1.1 Buffer Addresses 4.1.1.1

The starting position for a read operation is the buffer address at which Read Buffer starts reading and Read Modified starts scanning the buffer. The operation starts at buffer address 0 when the read-type command: immediately follows a selection sequence; is chained to a Copy command; or, chained or not chained, is sent to a terminal with an unformatted display. The operation starts at the current buffer address when the read-type command is chained to a write-type command.

Read Buffer reads to the last position in the buffer, as does Read Modified when the buffer contains no attribute characters. Normally Read Modified scans/reads to the last position in the last display field. The last field will wrap, if the first position on the first line does not contain an attribute character, so that scanning/reading will continue on the first line to the first attribute character.

At the end of a read operation, the buffer address is 0, i.e. the first position on the first line, unless the last field wrapped, in which case the buffer address points to the position occupied by the first attribute character.

4.1.1.2 AID Characters 4.1.1.2

A read message transferred in response to a polling sequence or an explicit read-type command contains an attention identification, or AID, character immediately after the device identification. The AID characters are described in the table in fig. 8.

Note that the AID codes 60_{Hex} and $E8_{Hex}$ are generated when the host sends an explicit read-type command and no attention key has been pressed. For "CURSR SELCT" and "reader", see appendix H.

Source of AID	Hex Code	Occurrence of AID	Response Body
no attention:		only in response to	
terminal	60	explicit	
printer	E8	read-type command	follows AID
data attention:		in response to:	
CURSR SELCT	7D	explicit	
CURSR SELCT	7E	read-type command	follows AID
PF1	F1	or	
PF2	F2		
PF3	F3	poll (long read)	follows AID
PF4	F4		
PF5	F5		
PF6	F6		
PF7	F7		
PF8	F8		
PF9	F9		
PF10	7A		
PF11	7B		
PF12	7C		
PF13	C1		
PF14	C2		
reader	E6		
reader	E7		
SEND	7D		
short attention:		in response to:	
CLEAR	6D	explicit	
PA1	6C	Read Buffer	follows AID
PA2	6E	or	
PA3	6B		
PA4	84	explicit	
PA5	85	Read Modified	absent
PA6	86	or	
PA7	87		
PA8	88	poll (short read)	absent
PA9	89		
PA10	91		
USM	92		

Figure 8: AID Characters.

4.1.1.3 Read Buffer Command 4.1.1.3

Read Buffer causes the transfer of all characters in the buffer, including nulls, from the buffer start address.

The host computer can give a Read Buffer command without regard to the depression of an attention key (which can only change the AID code).

The command message has no command body.

The response body of a message transferred in response to a Read Buffer command has the form:

[response body] = ADDR1 ADDR2 [read buffer string]

$$[\text{read buffer string}] = [\ [\text{start field}]_0^1\ \ [\text{data}]\ ]_1^*$$

[start field] = SF ATR

$$[\text{data}] = [\text{data character}]_0^*$$

The buffer address ADDR1 ADDR2 is the cursor position. The read buffer string contains all characters, including nulls, in the buffer. For a formatted buffer, each attribute character (ATR) is indicated by a start field order.

4.1.1.4 Read Modified Command 4.1.1.4

Read Modified causes the transfer of all modified fields in the buffer, with nulls suppressed, from the buffer start address (long read). If a short attention key has been pressed, Read Modified transfers only the AID character (short read).

The host computer does not normally use the Read Modified command explicitly, since the polling operation initiates the execution of an implicit Read Modified command whenever a display terminal has a pending read message.

The command message has no command body.

The response body of a long read message has the form:

[response body] = ADDR1 ADDR2 [read modified string]

[read modified string] = [[set buffer address]$_0^1$ [data]]$_0^*$

[set buffer address] = SBA ADDR1 ADDR2

[data] = [data character]$_0^*$

The buffer address ADDR1 ADDR2 is the cursor position. The read modified string contains all modified fields in the buffer, with nulls suppressed. Each field is preceded by a set buffer address order, in which the buffer address ADDR1 ADDR2 is the address of the first input position in the field. For an unformatted buffer, the cursor position is followed by a string containing all characters in the buffer, with nulls suppressed.

4.1.2 Write-Type Commands 4.1.2

The write-type commands (and their CMD characters) are: Write ($F1_{Hex}$ corresponding to the graphic "1") and Erase/Write ($F5_{Hex}$ corresponding to the graphic "5"). Write is used to give instructions for modifying the existing contents of the buffer, whereas Erase/Write is used to erase the entire buffer, and then (possibly) to give instructions concerning new buffer contents. The write-type commands can also define operations to be performed at the device by means of a write control character.

4.1.2.1 Buffer Addresses 4.1.2.1

The body of a write-type command can contain device buffer orders and device buffer data. The orders are executed as they occur, whereas the data is written in the buffer as data characters.

For a Write command, the starting position for a write operation is determined by one of the following:

- A set buffer address, or SBA, order.
- The current buffer address, when the command is chained to a read-type command or another Write command.
- The cursor position, when there is no SBA order, when the command is not chained, or when the command is chained to a control-type command (i.e. the current buffer address equals the cursor address).

The data characters are written successively in the following positions until a new SBA order is encountered or all of the characters have been stored. The buffer address is incremented by one for each character written.

For an Erase/Write command, the starting position for a write operation is character position 0, i.e. the first position on the first line, as the buffer address and the cursor address are both reset to 0 after the erase operation.

4.1.2.2 Write Control Character (WCC) 4.1.2.2

The body of a write-type command has, as its first element, a write control character, or WCC, which can define operations to be performed at the device. The bit assignments and definitions for the write control character, from the least to the most significant position, are as follows:

0: Reset MDT

When 1: The MDT in all attribute characters is reset before the write operation is begun.

1: Keyboard Restore

When 1: The use of the keyboard is restored when the write operation is completed.

2: Alarm

When 1: An alarm is sounded when the write operation is completed.

3: Start Printer

When 1: The printer is started when the write operation is completed in the buffer.

5-4: Print Format

When 00: Transparent.

When 01: 40 characters per line.

When 10: 64 characters per line.

When 11: 80 characters per line.

7-6: See appendix D.

Note that IBM designates the most significant bit position 0.

4.1.2.3 Write Command 4.1.2.3

The Write command is used by the host computer to modify the existing contents of the buffer of a selected device.

The command body has the form:

[command body] = WCC [write string]

$$[\text{write string}] = [\ [\text{order}]_0^1\ \ [\text{data}]\ ]_0^*$$

[order] = [start field] | [set buffer address] | [insert cursor]
| [program tabulation] | [repeat to address]
| [erase unprotected to address] | [unsolicited message]

$$[\text{data}] = [\text{data character}]_0^*$$

WCC is a write control character. The write string contains instructions for modifying the contents of the buffer and possibly the address of an associated cursor.

4.1.2.4 Erase/Write Command

The Erase/Write command is used by the host computer to erase the entire contents of the buffer in a selected device, typically prior to the definition of a new format.

Erase/Write replaces all characters in the buffer with nulls, moves the cursor to character position 0, and resets the buffer address to 0. Then, if the command message contained a command body, a write operation is performed as for a Write command.

4.1.3 Control-Type Commands

The control-type commands (and their CMD characters) are: Copy ($F7_{Hex}$ corresponding to the graphic "7") and Erase All Unprotected ($6F_{Hex}$ corresponding to the graphic "?"). The operations initiated by these commands do not require the transfer of device buffer data between the host computer and the selected device.

4.1.3.1 Copy Command

The Copy command is used by the host computer to transfer device buffer data from one device to another device connected to the same control unit. The selected device is the "to" device, i.e. the device to which the data is transferred.

The command body has the form:

[command body] = CCC DEV

CCC is a copy control character (described below). The DEV character (see subsection 3.4.1) specifies the "from" device, i.e. the device from which the data is transferred.

Note that if the DEV character specifies the selected device as the "from" device, an erasing operation, as indicated by bits 1-0 in the copy control character, will be performed in the buffer.

Note also that if the DEV character specifies a "from" device with a locked buffer, the CU will reject the Copy command (see subsection 3.5.3). A locked buffer is obtained by storing an attribute character in which bits 5-4 are 10 (protected attribute) as the first character in the buffer. The attribute character is described in subsection 4.2.1.

Copy Control Character (CCC)

The bit assignments and definitions for the copy control character, from the least to the most significant position, are as follows:

1-0: Copy Type
- When 00: Copy only attribute characters.
- When 01: Copy attribute characters and input fields including nulls; replace the contents of protected fields with nulls.
- When 10: Copy attribute characters and protected fields including nulls; replace the contents of input fields with nulls.
- When 11: Copy the entire contents of the buffer including nulls.

2: Alarm
- When 1: An alarm is sounded at the "to" device when the buffer transfer is completed.

3: Start Printer
- When 1: The printer is started at the "to" device when the buffer transfer is completed.

5-4: Print Format
- When 00: Transparent.
- When 01: 40 characters per line.
- When 10: 64 characters per line.
- When 11: 80 characters per line.

7-6: See appendix D.

Note that IBM designates the most significant bit position 0.

4.1.3.2 Erase All Unprotected Command 4.1.3.2

The Erase All Unprotected command is used by the host computer to perform the following operations at the selected device:

- o Clear all input fields to nulls.
- o Reset the MDT in all attribute characters that specify an input field.
- o Restore the use of the keyboard.
- o Reset the AID code.
- o Move the cursor to the first input position in the first input field.
- o Set the buffer address equal to the cursor address.

If there is no input field, i.e. if the entire buffer is protected, no erasing is performed in the buffer and the MDTs are not reset; the other operations are performed, however, the cursor being moved to character position 0.

The command message has no command body.

4.2 Device Buffer Orders 4.2

In the body of a write-type command, a device buffer order can occur as an element in the write string (see subsection 4.1.2.3). Such orders may or may not be followed by data. The orders themselves are executed as they occur, whereas data following an order is stored in the buffer.

Two of the orders, start field and set buffer address, also occur in the bodies of read messages (see subsections 4.1.1.3 and 4.1.1.4).

A device buffer order consists of a control character, which can be followed by up to three bytes of supplementary information. The device buffer orders are shown in the table in fig. 9.

Device Buffer Order	Control Character	Hex Code	Byte 2	Byte 3	Byte 4
Start Field	SF	1D	ATR		
Set Buffer Address	SBA	11	ADDR1	ADDR2	
Insert Cursor	IC	13			
Program Tabulation	PT	05			
Repeat to Address	RA	3C	ADDR1	ADDR2	CHAR
Erase Unprotected to Address	EUA	12	ADDR1	ADDR2	
Unsolicited Message	USM	2F			

ATR: attribute character
ADDR1 ADDR2: buffer address
CHAR: data character

Figure 9: Device Buffer Orders.

4.2.1 Start Field Order

4.2.1

Two-byte order. The SF character indicates that the next byte is an attribute character. The latter is stored in the buffer at the current buffer address.

Attribute Character (ATR)

The bit assignments and definitions for the attribute character, from the least to the most significant position, are as follows:

0: When 1: Modified field.

1: When 1: Flashing.

3-2: When 00: Normal display.
When 01: Cursor selectable.
When 10: Intensified display and cursor selectable.
When 11: Nondisplay.

5-4: When 00: Alphameric input.
When 01: Numeric input.
When 10: Protected.
When 11: Automatic skip.

7-6: See appendix D.

Note that IBM designates the most significant bit position 0.

The attribute character is protected, and only bit 0, the MDT, can be affected by manual input. Depression of the CLEAR key, however, will clear all characters in the buffer to nulls. The host computer can send an attribute character containing any combination of bits.

4.2.2 Set Buffer Address Order 4.2.2

Three-byte order. The SBA character indicates that the next two bytes constitute the buffer address at which the following data is to be stored.

An SBA order can also precede another order to specify: the address at which an attribute character is to be stored by a subsequent start field order; the address at which the cursor is to be positioned by a subsequent insert cursor order; or the starting address for a subsequent program tabulation, repeat to address, or erase unprotected to address order.

4.2.3 Insert Cursor Order 4.2.3

One-byte order. The IC character causes the cursor to be moved to the character position pointed to by the current buffer address. Neither the contents of this position nor the current buffer address is affected.

4.2.4 Program Tabulation Order

4.2.4

One-byte order. The PT character indicates that a tabulation operation is to be performed in the buffer. The current buffer address is incremented accordingly to the address of the first character position in the next input field. If the PT order is encountered when the current buffer address points to an attribute character that defines an input field, the current buffer address is incremented by one to the address of the first input position.

If the PT order follows a data character in the write string, nulls are inserted in any remaining positions to the end of the field, regardless of whether the latter is protected or unprotected. If the display is unformatted, nulls are inserted in any remaining positions to the end of the screen, and the current buffer address is reset to 0. If the display is formatted, the current buffer address will also be reset to 0, when no input field is found before the end of the screen.

The PT order stops its search for an input field at the last position in the buffer. To continue the search, a second PT order must be given immediately following the first. Since the current buffer address was reset to 0 by the first PT order, the second PT order begins its search at buffer position 0. If the first PT order was still inserting nulls in each character position when it terminated at the last buffer position, the second PT order will continue to insert nulls from buffer position 0 to the end of the current field.

4.2.5 Repeat to Address Order

4.2.5

Four-byte order. The RA character indicates that the data character contained in the fourth byte is to be stored in each position starting at the current buffer address and extending to, but not including, the address specified by the second and third bytes, which will be the current buffer address when the operation is completed.

If the RA order specifies an address which is equal to or lower than the current buffer address, wrapping will take place during the operation. If the address is equal to the current buffer address, the specified data character will be stored in all buffer positions.

Note that an RA order will overwrite an attribute character.

4.2.6 Erase Unprotected to Address Order 4.2.6

Three-byte order. The EUA character indicates that nulls are to be inserted in all unprotected buffer positions starting at the current buffer address and extending to, but not including, the address specified by the second and third bytes, which will be the current buffer address when the operation is completed.

If the EUA order specifies an address which is equal to or lower than the current buffer address, wrapping will take place during the operation. If the address is equal to the current buffer address, nulls will be inserted in all unprotected buffer positions.

Note that an EUA order will not overwrite an attribute character.

4.2.7 Unsolicited Message Order 4.2.7

One-byte order. The USM character in a command to a display terminal causes the indicator on the short attention key marked USM to be lit. When USM occurs in a command to a printer, the latter is reserved by the host computer, until a command that does not contain the USM character is received.

A. REFERENCES A.

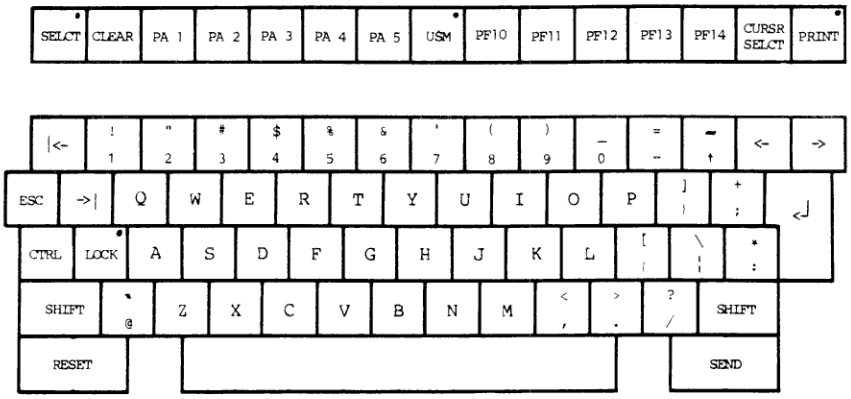
[1] RCSL No 42-i2150:
RC855 IBM 3270 BSC Emulator Operating Guide
Henning Christensen, November 1982
Abstract: Operating guide for the RC855 IBM 3270 BSC Emulator. Describes: keyboard functions; operating procedures; emulator messages; terminal configuration; diagnostics. Covers the specific aspects of operating the emulator, and is meant to be used in conjunction with the terminal operating, software installation, application guidance, and technical reference data documents.

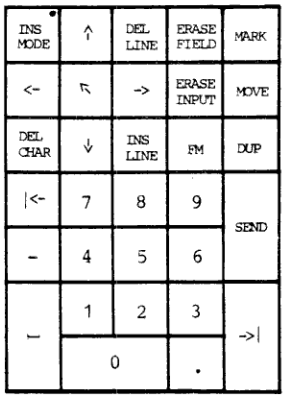
Danish edition:

RCSL Nr. 42-i2151:
RC855 IBM 3270 BSC Emulator Betjeningsvejledning
Henning Christensen, november 1982
Resumé: Betjeningsvejledning for RC855 IBM 3270 BSC Emulator. Beskriver: tastaturets funktioner; betjeningsprocedurer; emulatormeddelelser; terminalkonfigurering; diagnosticering. Omhandler de konkrete aspekter i emulatorbetjeningen og skal benyttes i sammenhæng med dokumentation vedrørende installation af programmel, applikationsbetjening og teknisk referencemateriale.

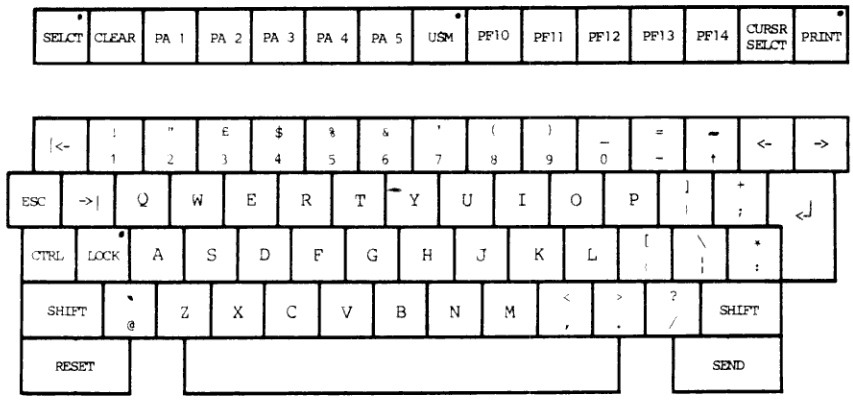
B. RC855 KEYBOARD LAYOUTS

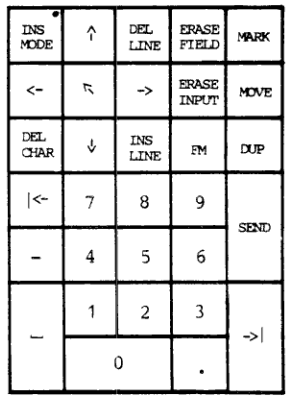
B.1 US English Keyboard Layout



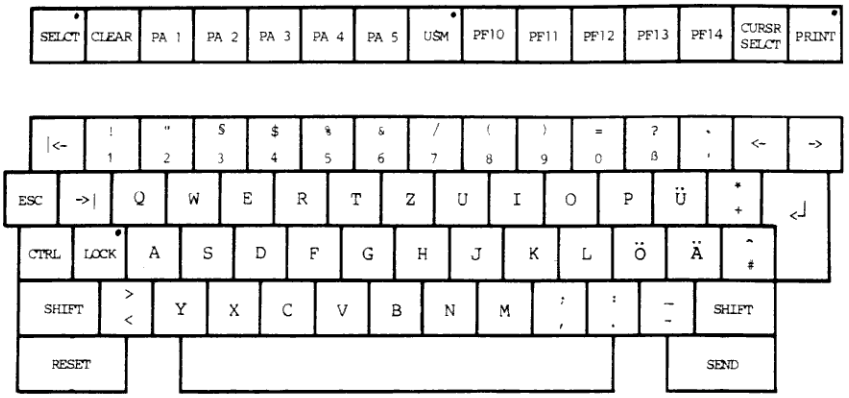


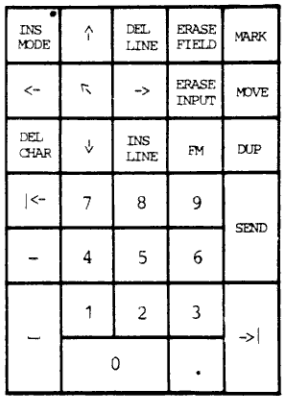
B.2 UK English Keyboard Layout



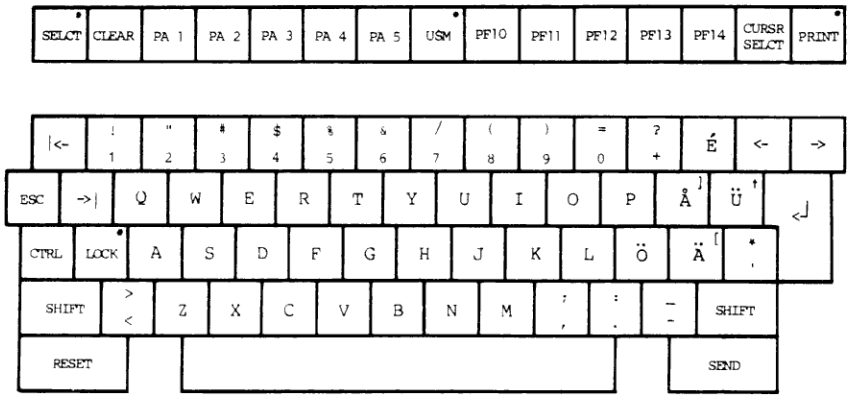


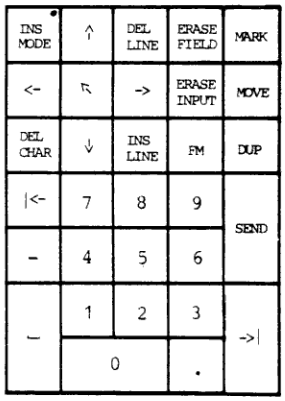
B.3 German Keyboard Layout



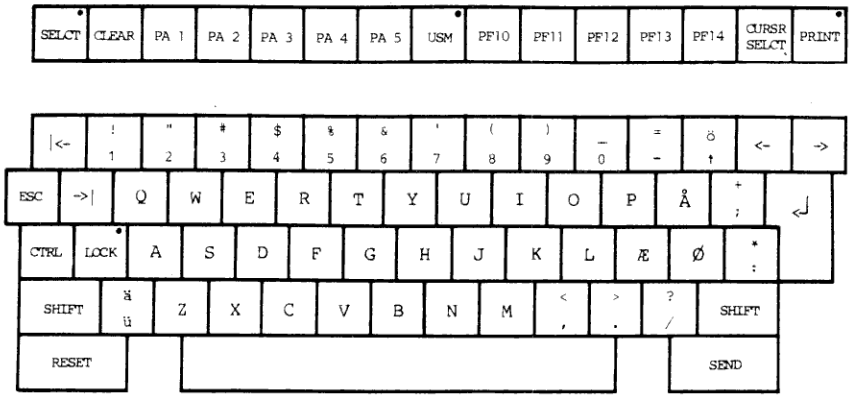


B.4 Swedish Keyboard Layout

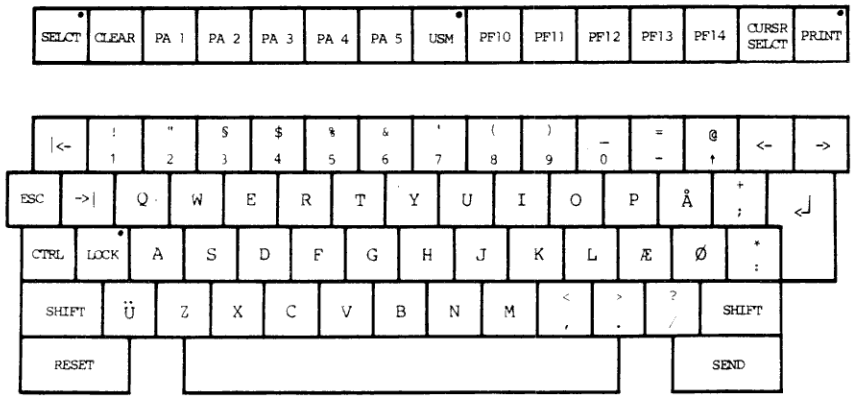




B.5 Danish Standard Keyboard Layout



B.6 Danish Public Sector Keyboard Layout



C. EBCDIC CODE SETS C.

The RC855 IBM 3270 BSC Emulator has a number of national versions. For each national version there are two code sets:

- The EBCDIC code set used for communication with the host computer, including BSC line control characters and control characters used as device buffer orders and printer control characters.

- The 7-bit code set used for printing (see appendix E).

With a few exceptions, which are described in the notes below, each national character set contains 95 graphic characters, including the space (SP) character, with EBCDIC codes defined for all 95 characters.

Note that alternate EBCDIC code sets also exist for the German, Swedish, Danish standard, and Danish public sector versions of the Emulator. A configuration parameter in the primary terminal determines which of the two EBCDIC code sets is to be used.

C.1 US English EBCDIC Character Codes

Bits 76	00				01				10				11			
Bits 54	00	01	10	11	00	01	10	11	00	01	10	11	00	01	10	11
Bits 3210 / Hex 1 / Hex 2	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0000	0	NUL	DLE			SP	&	-						{	}	\	0
0001	1	SOH	SBA					/		a	j	~		A	J		1
0010	2	STX	EUA		SYN					b	k	s		B	K	S	2
0011	3	ETX	IC							c	l	t		C	L	T	3
0100	4									d	m	u		D	M	U	4
0101	5	PT	NL							e	n	v		E	N	V	5
0110	6			ETB						f	o	w		F	O	W	6
0111	7			ESC	EOT					g	p	x		G	P	X	7
1000	8									h	q	y		H	Q	Y	8
1001	9		EM						`	i	r	z		I	R	Z	9
1010	A					]	!	¦	:								
1011	B	VT				.	$	,	#								
1100	C	FF	DUP		RA	<	*	%	@								
1101	D	CR	SF	ENQ	NAK	(	)	_	'								
1110	E		FM			+	;	>	=								
1111	F		IUS	USM		[	↑	?	"								

Note that ITB is designated IUS in the table.

C.2 UK English EBCDIC Character Codes C.2

Bits 3210	Bits 76	00				01				10				11			
	Bits 54	00	01	10	11	00	01	10	11	00	01	10	11	00	01	10	11
	Hex 1 / Hex 2	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0000	0	NUL	DLE			SP	&	-						{	}	\	0
0001	1	SOH	SBA					/		a	j	~		A	J		1
0010	2	STX	EUA		SYN					b	k	s		B	K	S	2
0011	3	ETX	IC							c	l	t		C	L	T	3
0100	4									d	m	u		D	M	U	4
0101	5	PT	NL							e	n	v		E	N	V	5
0110	6			ETB						f	o	w		F	O	W	6
0111	7			ESC	EOT					g	p	x		G	P	X	7
1000	8									h	q	y		H	Q	Y	8
1001	9		EM						`	i	r	z		I	R	Z	9
1010	A					¢	!	¦	:								
1011	B	VT				.	£	,	]								
1100	C	FF	DUP		RA	<	*	%	@								
1101	D	CR	SF	ENQ	NAK	(	)	_	'								
1110	E		FM			+	;	>	=								
1111	F		IUS	USM		[	↑	?	"								

Note that ITB is designated IUS in the table.

C.3 German EBCDIC Character Codes C.3

Bits 76		00				01				10				11			
Bits 54		00	01	10	11	00	01	10	11	00	01	10	11	00	01	10	11
Bits 3210	Hex 1 / Hex 2	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0000	0	NUL	DLE			SP	&	-						ä	ü	Ö	0
0001	1	SOH	SBA					/		a	j	ß		A	J		1
0010	2	STX	EUA		SYN					b	k	s		B	K	S	2
0011	3	ETX	IC							c	l	t		C	L	T	3
0100	4									d	m	u		D	M	U	4
0101	5	PT	NL							e	n	v		E	N	V	5
0110	6			ETB						f	o	w		F	O	W	6
0111	7			ESC	EOT					g	p	x		G	P	X	7
1000	8									h	q	y		H	Q	Y	8
1001	9		EM						`	i	r	z		I	R	Z	9
1010	A					Ä	Ü	ö	:								
1011	B	VT				.	$	,	#								
1100	C	FF	DUP		RA	<	*	%	§								
1101	D	CR	SF	ENQ	NAK	(	)	_	'								
1110	E		FM			+	;	>	=								
1111	F		IUS	USM		!	↑	?	"								

Note that ITB is designated IUS in the table.

C.4 German Alternate EBCDIC Character Codes C.4

Only 90 codes are shown in the table below. The characters "$", "#", "^", "§", and "`" can be entered from the keyboard and stored in the device buffer of a display terminal. However, when the contents of a device buffer are transmitted to the host computer, all of these characters are transmitted as blanks (40_{Hex}).

	Bits 76	00				01				10				11			
	Bits 54	00	01	10	11	00	01	10	11	00	01	10	11	00	01	10	11
Bits 3210	Hex 1 / Hex 2	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0000	0	NUL	DLE			SP	&	-									0
0001	1	SOH	SBA					/		a	j			A	J		1
0010	2	STX	EUA		SYN					b	k	s		B	K	S	2
0011	3	ETX	IC							c	l	t		C	L	T	3
0100	4									d	m	u		D	M	U	4
0101	5	PT	NL							e	n	v		E	N	V	5
0110	6			ETB						f	o	w		F	O	W	6
0111	7			ESC	EOT					g	p	x		G	P	X	7
1000	8									h	q	y		H	Q	Y	8
1001	9		EM							i	r	z		I	R	Z	9
1010	A					.ö	ü	ß	:	←				_			
1011	B	VT				.	Ü	,	Ä								
1100	C	FF	DUP		RA	<	*	%	Ö								
1101	D	CR	SF	ENQ	NAK	(	)	_	'								
1110	E		FM			+	;	>	=								
1111	F		IUS	USM		!	↑	?	.ä								

Note that ITB is designated IUS in the table.

C.5 Swedish EBCDIC Character Codes

	Bits 76	00				01				10				11			
	Bits 54	00	01	10	11	00	01	10	11	00	01	10	11	00	01	10	11
Bits 3210	Hex 1 / Hex 2	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0000	0	NUL	DLE			SP	&	-						ä	å	Å	0
0001	1	SOH	SBA					/		a	j	ü		A	J		1
0010	2	STX	EUA		SYN					b	k	s		B	K	S	2
0011	3	ETX	IC							c	l	t		C	L	T	3
0100	4									d	m	u		D	M	U	4
0101	5	PT	NL							e	n	v		E	N	V	5
0110	6			ETB						f	o	w		F	O	W	6
0111	7			ESC	EOT					g	p	x		G	P	X	7
1000	8									h	q	y		H	Q	Y	8
1001	9		EM						é	i	r	z		I	R	Z	9
1010	A					#	ø	ö	:								
1011	B	VT				.	Å	,	Ä								
1100	C	FF	DUP		RA	<	*	%	Ö								
1101	D	CR	SF	ENQ	NAK	(	)	_	'								
1110	E		FM			+	;	>	=								
1111	F		IUS	USM		!	Ü	¤	"								

Note that ITB is designated IUS in the table.

C.6 Swedish Alternate EBCDIC Character Codes C.6

	Bits 76	00				01				10				11			
	Bits 54	00	01	10	11	00	01	10	11	00	01	10	11	00	01	10	11
Bits 3210	Hex 1 / Hex 2	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0000	0	NUL	DLE			SP	&	-									0
0001	1	SOH	SBA					/		a	j			A	J		1
0010	2	STX	EUA		SYN					b	k	s		B	K	S	2
0011	3	ETX	IC							c	l	t		C	L	T	3
0100	4									d	m	u		D	M	U	4
0101	5	PT	NL							e	n	v		E	N	V	5
0110	6			ETB						f	o	w		F	O	W	6
0111	7			ESC	EOT					g	p	x		G	P	X	7
1000	8					#	ɡ	"	!	h	q	y		H	Q	Y	8
1001	9		EM			é	É	ü	Ü	i	r	z		I	R	Z	9
1010	A					ö	å		:								
1011	B	VT				.	Å	,	Ä								
1100	C	FF	DUP		RA	<	*	§	Ö								
1101	D	CR	SF	ENQ	NAK	(	)	_	'								
1110	E		FM			+	;	>	=								
1111	F		IUS	USM				?	ä								

Note that ITB is designated IUS in the table.

C.7 Danish Standard EBCDIC Character Codes C.7

Bits 3210	Bits 76	00				01				10				11			
	Bits 54	00	01	10	11	00	01	10	11	00	01	10	11	00	01	10	11
	Hex 1 / Hex 2	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0000	0	NUL	DLE			SP	&	-						æ	å	ä	0
0001	1	SOH	SBA					/		a	j	ü		A	J		1
0010	2	STX	EUA		SYN					b	k	s		B	K	S	2
0011	3	ETX	IC							c	l	t		C	L	T	3
0100	4									d	m	u		D	M	U	4
0101	5	PT	NL							e	n	v		E	N	V	5
0110	6			ETB						f	o	w		F	O	W	6
0111	7			ESC	EOT					g	p	x		G	P	X	7
1000	8									h	q	y		H	Q	Y	8
1001	9		EM						ö	i	r	z		I	R	Z	9
1010	A					#	ø	ø	:								
1011	B	VT				.	Å	,	Æ								
1100	C	FF	DUP		RA	<	*	%	Ø								
1101	D	CR	SF	ENQ	NAK	(	)	_	'								
1110	E		FM			+	;	>	=								
1111	F		IUS	USM		!	↑	?	"								

Note that ITB is designated IUS in the table.

C.8 Danish Standard Alternate EBCDIC Character Codes C.8

Only 93 EBCDIC codes are used for graphic characters. The characters "Å" and "$" can both be entered from the keyboard and stored in the device buffer of a display terminal. However, when the contents of a device buffer are transmitted to the host computer, both "Å" and "$" are transmitted as $5B_{Hex}$. When the EBCDIC code $5B_{Hex}$ is received from the host as device buffer data, it is stored in the device buffer as "Å". The characters "Æ" and "#" are similarly treated. Both are transmitted as $7B_{Hex}$. When $7B_{Hex}$ is received, it is stored as "Æ".

	Bits 76	00				01				10				11			
	Bits 54	00	01	10	11	00	01	10	11	00	01	10	11	00	01	10	11
Bits 3210	Hex 1 / Hex 2	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0000	0	NUL	DLE			SP	&	-									0
0001	1	SOH	SBA					/		a	j			A	J		1
0010	2	STX	EUA		SYN					b	k	s		B	K	S	2
0011	3	ETX	IC							c	l	t		C	L	T	3
0100	4									d	m	u		D	M	U	4
0101	5	PT	NL							e	n	v		E	N	V	5
0110	6			ETB						f	o	w		F	O	W	6
0111	7			ESC	EOT					g	p	x		G	P	X	7
1000	8							"	!	h	q	y		H	Q	Y	8
1001	9		EM			ö	ä	ü		i	r	z		I	R	Z	9
1010	A					ø	å		:								
1011	B	VT				.	Åß	,	Æ#								
1100	C	FF	DUP		RA	<	*	%	Ø								
1101	D	CR	SF	ENQ	NAK	(	)	_	'								
1110	E		FM			+	;	>	=								
1111	F		IUS	USM		↑		?	æ								

Note that ITB is designated IUS in the table.

C.9 Danish Public Sector EBCDIC Character Codes C.9

	Bits 76	00				01				10				11			
	Bits 54	00	01	10	11	00	01	10	11	00	01	10	11	00	01	10	11
Bits 3210	Hex 1 / Hex 2	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0000	0	NUL	DLE			SP	&	-						æ	å	Ü	0
0001	1	SOH	SBA					/		a	j	ü		A	J		1
0010	2	STX	EUA		SYN					b	k	s		B	K	S	2
0011	3	ETX	IC							c	l	t		C	L	T	3
0100	4									d	m	u		D	M	U	4
0101	5	PT	NL							e	n	v		E	N	V	5
0110	6			ETB						f	o	w		F	O	W	6
0111	7			ESC	EOT					g	p	x		G	P	X	7
1000	8									h	q	y		H	Q	Y	8
1001	9		EM						@	i	r	z		I	R	Z	9
1010	A					§	Ø	ø	:								
1011	B	VT				.	Å	,	Æ								
1100	C	FF	DUP		RA	<	*	%	Ø								
1101	D	CR	SF	ENQ	NAK	(	)	_	'								
1110	E		FM			+	;	>	=								
1111	F		IUS	USM		!	↑	?	"								

Note that ITB is designated IUS in the table.

C.10 Danish Public Sector Alternate EBCDIC Character Codes C.10

Only 94 EBCDIC codes are used for graphic characters. The characters "Å" and "$" are treated as described for the Danish standard alternate EBCDIC character codes (see section C.8).

	Bits 76	00				01				10				11			
	Bits 54	00	01	10	11	00	01	10	11	00	01	10	11	00	01	10	11
Bits 3210	Hex 1 / Hex 2	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0000	0	NUL	DLE			SP	&	-									0
0001	1	SOH	SBA					/		a	j			A	J		1
0010	2	STX	EUA		SYN					b	k	s		B	K	S	2
0011	3	ETX	IC							c	l	t		C	L	T	3
0100	4									d	m	u		D	M	U	4
0101	5	PT	NL							e	n	v		E	N	V	5
0110	6			ETB						f	o	w		F	O	W	6
0111	7			ESC	EOT					g	p	x		G	P	X	7
1000	8							"	!	h	q	y		H	Q	Y	8
1001	9		EM			@		ü	Ü	i	r	z		I	R	Z	9
1010	A					ø	å		:		§						
1011	B	VT				.	Åå	,	Æ								
1100	C	FF	DUP		RA	<	*	%	Ø								
1101	D	CR	SF	ENQ	NAK	(	)	_	'								
1110	E		FM			+	;	>	=								
1111	F		IUS	USM		↑		?	æ								

Note that ITB is designated IUS in the table.

D. EBCDIC CODES FOR 6-BIT QUANTITIES D.

A number of quantities which are not themselves textual characters can be transmitted as part of a message using the BSC line control protocol. These quantities are either binary encoded in 6 bits or encoded in subfields of a 6-bit field. They include:

- o device buffer addresses (including cursor addresses)
- o CU numbers
- o device numbers
- o status0 and status1
- o attribute characters
- o write control characters
- o copy control characters

The general rule for obtaining the EBCDIC character used for the transmission of a 6-bit quantity is: Place the 6-bit quantity in the least significant bit positions of a character byte, and assign the two high-order bits so that the complete byte value yields a valid EBCDIC graphic character. The set of graphic characters used to represent 6-bit quantities is indicated by the shaded areas in fig. 10.

Bits 76		00				01				10				11			
Bits 54		00	01	10	11	00	01	10	11	00	01	10	11	00	01	10	11
Bits 3210	Hex 1 / Hex 2	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0000	0	NUL	DLE			SP	&	-						{	}	\	0
0001	1	SOH	SBA					/		a	j	~		A	J		1
0010	2	STX	EUA		SYN					b	k	s		B	K	S	2
0011	3	ETX	IC							c	l	t		C	L	T	3
0100	4									d	m	u		D	M	U	4
0101	5	PT	NL							e	n	v		E	N	V	5
0110	6			ETB						f	o	w		F	O	W	6
0111	7			ESC	EOT					g	p	x		G	P	X	7
1000	8									h	q	y		H	Q	Y	8
1001	9		EM						`	i	r	z		I	R	Z	9
1010	A					[	:	¦	:								
1011	B	VT				.	$	,	#								
1100	C	FF	DUP		RA	<	*	%	@								
1101	D	CR	SF	ENQ	NAK	(	)	_	'								
1110	E		FM			+	;	>	=								
1111	F		IUS	USM		]	¬	?	"								

Figure 10: EBCDIC Graphic Characters Used to Represent 6-Bit Quantities.

The EBCDIC equivalent of each 6-bit value is found in the table in fig. 11. The bits are numbered in increasing significance from bit 0. Note that the graphic characters differ according to the national version.

6-Bit Value bit 5 4 3 2 1 0	dec	EBCDIC US English graphic	hex	6-Bit Value bit 5 4 3 2 1 0	dec	EBCDIC US English graphic	hex
0 0 0 0 0 0	0	SP	40	1 0 0 0 0 0	32	-	60
0 0 0 0 0 1	1	A	C1	1 0 0 0 0 1	33	/	61
0 0 0 0 1 0	2	B	C2	1 0 0 0 1 0	34	S	E2
0 0 0 0 1 1	3	C	C3	1 0 0 0 1 1	35	T	E3
0 0 0 1 0 0	4	D	C4	1 0 0 1 0 0	36	U	E4
0 0 0 1 0 1	5	E	C5	1 0 0 1 0 1	37	V	E5
0 0 0 1 1 0	6	F	C6	1 0 0 1 1 0	38	W	E6
0 0 0 1 1 1	7	G	C7	1 0 0 1 1 1	39	X	E7
0 0 1 0 0 0	8	H	C8	1 0 1 0 0 0	40	Y	E8
0 0 1 0 0 1	9	I	C9	1 0 1 0 0 1	41	Z	E9
0 0 1 0 1 0	10	]	4A	1 0 1 0 1 0	42	¦	6A
0 0 1 0 1 1	11	.	4B	1 0 1 0 1 1	43	,	6B
0 0 1 1 0 0	12	<	4C	1 0 1 1 0 0	44	%	6C
0 0 1 1 0 1	13	(	4D	1 0 1 1 0 1	45	_	6D
0 0 1 1 1 0	14	+	4E	1 0 1 1 1 0	46	>	6E
0 0 1 1 1 1	15	[	4F	1 0 1 1 1 1	47	?	6F
0 1 0 0 0 0	16	&	50	1 1 0 0 0 0	48	0	F0
0 1 0 0 0 1	17	J	D1	1 1 0 0 0 1	49	1	F1
0 1 0 0 1 0	18	K	D2	1 1 0 0 1 0	50	2	F2
0 1 0 0 1 1	19	L	D3	1 1 0 0 1 1	51	3	F3
0 1 0 1 0 0	20	M	D4	1 1 0 1 0 0	52	4	F4
0 1 0 1 0 1	21	N	D5	1 1 0 1 0 1	53	5	F5
0 1 0 1 1 0	22	O	D6	1 1 0 1 1 0	54	6	F6
0 1 0 1 1 1	23	P	D7	1 1 0 1 1 1	55	7	F7
0 1 1 0 0 0	24	Q	D8	1 1 1 0 0 0	56	8	F8
0 1 1 0 0 1	25	R	D9	1 1 1 0 0 1	57	9	F9
0 1 1 0 1 0	26	!	5A	1 1 1 0 1 0	58	:	7A
0 1 1 0 1 1	27	$	5B	1 1 1 0 1 1	59	#	7B
0 1 1 1 0 0	28	*	5C	1 1 1 1 0 0	60	@	7C
0 1 1 1 0 1	29	)	5D	1 1 1 1 0 1	61	'	7D
0 1 1 1 1 0	30	;	5E	1 1 1 1 1 0	62	=	7E
0 1 1 1 1 1	31	↑	5F	1 1 1 1 1 1	63	"	7F

Figure 11: Correspondence between 6-Bit Quantity and EBCDIC Code Used for Transmission to/from Host.

For speedy reference, the CU and device numbers and the corresponding hexadecimal addresses (complete byte values) used in polling/selection sequences are given in table form in fig. 12.

CU number			CU poll address	CU selection address
	device number (1)	device number (2)	device address (1)	device address (2)
0	0	32	40	60
1	1	33	C1	61
2	2	34	C2	E2
3	3	35	C3	E3
4	4	36	C4	E4
5	5	37	C5	E5
6	6	38	C6	E6
7	7	39	C7	E7
8	8	40	C8	E8
9	9	41	C9	E9
10	10	42	4A	6A
11	11	43	4B	6B
12	12	44	4C	6C
13	13	45	4D	6D
14	14	46	4E	6E
15	15	47	4F	6F
16	16	48	50	F0
17	17	49	D1	F1
18	18	50	D2	F2
19	19	51	D3	F3
20	20	52	D4	F4
21	21	53	D5	F5
22	22	54	D6	F6
23	23	55	D7	F7
24	24	56	D8	F8
25	25	57	D9	F9
26	26	58	5A	7A
27	27	59	5B	7B
28	28	61	5C	7C
29	29	61	5D	7D
30	30	62	5E	7E
31	31		5F	7F

Figure 12: Correspondence between CU and Device Numbers and Hexadecimal Addresses Used in Polling/Selection Sequences.

E. 7-BIT CODE SETS E.

The RC855 IBM 3270 BSC Emulator has a number of national versions. For each national version there are two code sets:

- The 7-bit code set, derived from ISO 646, used for printing on printers attached via a serial V.24 interface.

- The EBCDIC code set used for communication with the host computer (see appendix C).

Each national character set contains 95 graphic characters, including the space (SP) character, with printer codes defined for all 95 characters.

Each printer alphabet includes the four control characters CR, LF, VT, and FF, which all printers must support.

E.1 US English Printer Character Codes E.1

Bits 3210 / Hex 2	Bits 654 / Hex 1	000 0	001 1	010 2	011 3	100 4	101 5	110 6	111 7
0000	0			SP	0	@	P	`	p
0001	1			!	1	A	Q	a	q
0010	2			"	2	B	R	b	r
0011	3			#	3	C	S	c	s
0100	4			$	4	D	T	d	t
0101	5			%	5	E	U	e	u
0110	6			&	6	F	V	f	v
0111	7			'	7	G	W	g	w
1000	8			(	8	H	X	h	x
1001	9			)	9	I	Y	i	y
1010	A			*	:	J	Z	j	z
1011	B			+	;	K	[	k	{
1100	C			,	<	L	\	l	¦
1101	D			-	=	M	]	m	}
1110	E			.	>	N	↑	n	~
1111	F			/	?	O	_	o	

E.2 UK English Printer Character Codes

E.2

Bits 3210 / Hex 2	Bits 654 / Hex 1	000	001	010	011	100	101	110	111
		0	1	2	3	4	5	6	7
0000	0			SP	0	@	P	`	p
0001	1			!	1	A	Q	a	q
0010	2			"	2	B	R	b	r
0011	3			£	3	C	S	c	s
0100	4			$	4	D	T	d	t
0101	5			%	5	E	U	e	u
0110	6			&	6	F	V	f	v
0111	7			'	7	G	W	g	w
1000	8			(	8	H	X	h	x
1001	9			)	9	I	Y	i	y
1010	A			*	:	J	Z	j	z
1011	B			+	;	K	[	k	{
1100	C			,	<	L	\	l	¦
1101	D			-	=	M	]	m	}
1110	E			.	>	N	↑	n	~
1111	F			/	?	O	_	o	

E.3 German Printer Character Codes E.3

Bits 654	000	001	010	011	100	101	110	111
Bits 3210 / Hex 1 / Hex 2	0	1	2	3	4	5	6	7
0000	0		SP	0	§	P	`	p
0001	1		!	1	A	Q	a	q
0010	2		"	2	B	R	b	r
0011	3		#	3	C	S	c	s
0100	4		$	4	D	T	d	t
0101	5		%	5	E	U	e	u
0110	6		&	6	F	V	f	v
0111	7		'	7	G	W	g	w
1000	8		(	8	H	X	h	x
1001	9		)	9	I	Y	i	y
1010	A		*	:	J	Z	j	z
1011	B		+	;	K	Ä	k	ä
1100	C		,	<	L	Ö	l	ö
1101	D		-	=	M	Ü	m	ü
1110	E		.	>	N	↑	n	ß
1111	F		/	?	O	_	o	

E.4 Swedish Printer Character Codes

E.4

Bits 654	000	001	010	011	100	101	110	111
Bits 3210 / Hex 1 / Hex 2	0	1	2	3	4	5	6	7
0000	0			SP	0	É	P	é	p
0001	1			!	1	A	Q	a	q
0010	2			"	2	B	R	b	r
0011	3			#	3	C	S	c	s
0100	4			¤	4	D	T	d	t
0101	5			%	5	E	U	e	u
0110	6			&	6	F	V	f	v
0111	7			'	7	G	W	g	w
1000	8			(	8	H	X	h	x
1001	9			)	9	I	Y	i	y
1010	A			*	:	J	Z	j	z
1011	B			+	;	K	Ä	k	ä
1100	C			,	<	L	Ö	l	ö
1101	D			-	=	M	Å	m	å
1110	E			.	>	N	Ü	n	ü
1111	F			/	?	O	_	o	

E.5 Danish Standard Printer Character Codes E.5

Bits 654	000	001	010	011	100	101	110	111
Bits 3210 / Hex 1 / Hex 2	0	1	2	3	4	5	6	7
0000	0		SP	0	ü	P	ä	p
0001	1		!	1	A	Q	a	q
0010	2		"	2	B	R	b	r
0011	3		#	3	C	S	c	s
0100	4		¤	4	D	T	d	t
0101	5		%	5	E	U	e	u
0110	6		&	6	F	V	f	v
0111	7		'	7	G	W	g	w
1000	8		(	8	H	X	h	x
1001	9		)	9	I	Y	i	y
1010	A		*	:	J	Z	j	z
1011	B		+	;	K	Æ	k	æ
1100	C		,	<	L	Ø	l	ø
1101	D		-	=	M	Å	m	å
1110	E		.	>	N	↑	n	ö
1111	F		/	?	O	_	o	

E.6 Danish Public Sector Printer Character Codes E.6

Bits 3210 / Hex 2	Bits 654 / Hex 1	000	001	010	011	100	101	110	111
		0	1	2	3	4	5	6	7
0000	0			SP	0	@	P	↑	p
0001	1			!	1	A	Q	a	q
0010	2			"	2	B	R	b	r
0011	3			§	3	C	S	c	s
0100	4			¤	4	D	T	d	t
0101	5			%	5	E	U	e	u
0110	6			&	6	F	V	f	v
0111	7			'	7	G	W	g	w
1000	8			(	8	H	X	h	x
1001	9			)	9	I	Y	i	y
1010	A			*	:	J	Z	j	z
1011	B			+	;	K	Æ	k	æ
1100	C			,	<	L	Ø	l	ø
1101	D			-	=	M	Å	m	å
1110	E			.	>	N	Ü	n	ü
1111	F			/	?	O	_	o	

F. V.24 CONNECTIONS F.

The RC855 Display Terminal has two 25-pin type DB25-S connectors, which are used for the attachment of signal cables to a modem (used for BSC communication with a host computer) and a printer. The connector for the modem is marked LINE I, and the connector for the printer LINE II.

Fig. 13 shows the correspondence between the connector pins and interchange circuits as defined in the CCITT V.24 Recommendation. This correspondence complies with ISO Standard No 2110. The pins not shown in the figure are not used by the Emulator.

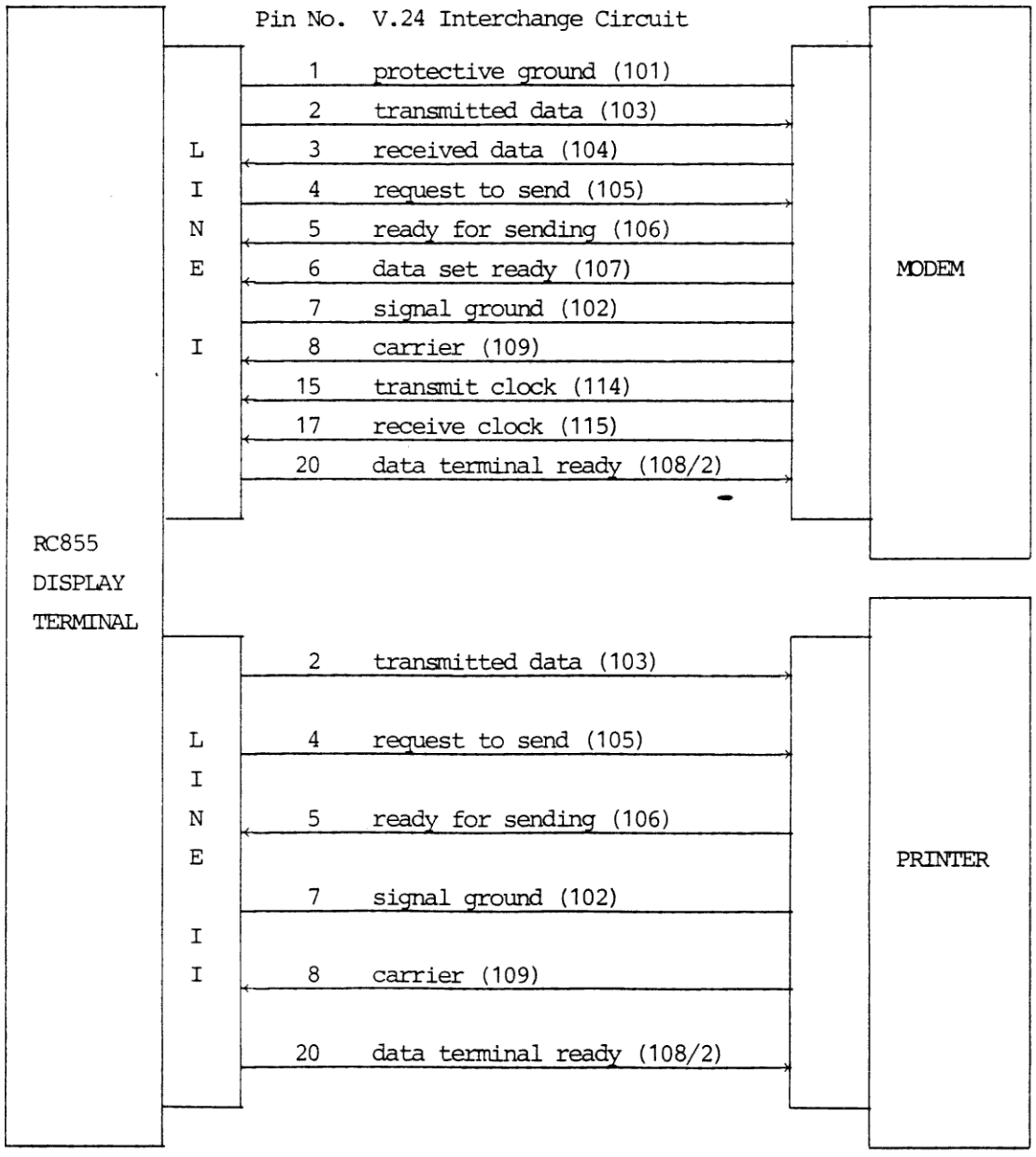


Figure 13: Correspondence between Connector Pins and Interchange Circuits.

The interchange circuits connecting the RC855 to the modem are used in accordance with CCITT V.24.

The use of each of the interchange circuits connecting the RC855 to an attached printer is described below.

transmitted data (103)

Used for transmission of data, i.e. 7-bit ISO-coded control characters and printable characters (see section 2.3 and appendix E).

data terminal ready (108/2)

This circuit is ON whenever the Emulator is running.

request to send (105)

This circuit is ON whenever the Emulator has a character to send to the printer.

ready for sending (106)

The ON condition is interpreted by the Emulator to mean that the printer is ready to receive the next character. When this circuit is OFF, the Emulator will not transmit a character. If the OFF condition persists longer than 20 seconds while circuit 105 is ON, the status printer_offline (see subsection 3.5.2) will be reported to the host, if appropriate, or the operator message "Printer not ready" will be displayed at the terminal from which a hard copy was requested.

carrier (109)

This circuit must be ON whenever the printer is online and operative. When this circuit is OFF, the Emulator assumes that the printer is powered off, is locally selected, or has a paper-out condition. The status printer_offline (see subsection 3.5.2) is reported to the host, if appropriate, and/or the operator message "Printer offline" is displayed at any terminal from which a hard copy is requested.

G. TRANSMISSION SEQUENCES SHOWING HEXADECIMAL CODES G.

This appendix is intended as an aid in the performance of system monitoring (fully described in the operating guide [1]). The primary terminal can capture and display transmission sequences on the BSC communication line, including recognized polling/selection sequences for other control units. The following description is based on the discussion of communication with the host computer found in chapter 3. For each sequence, the corresponding hexadecimal code is shown underscored, e.g. ENQ 2D.

Transmission sequences are delimited by PAD characters:

... PAD <transmission sequence> PAD ...
... FF <transmission sequence> FF ...

<transmission sequence>:

SYN SYN <sequence>	32 32 <sequence>

<sequence>:

<polling/selection sequence>	
<text>	
ACK 0 (i.e. DLE 70)	10 70
ACK 1 (i.e. DLE 61)	10 61
WACK (i.e. DLE 6B)	10 6B
RVI (i.e. DLE 7C)	10 7C
NAK	3D
EOT	37
ENQ	2D
TTD (i.e. STX ... ENQ)	02 ... 2D

<polling/selection sequence>: CU CU DEV DEV ENQ

Examples

General poll,	CU CU DEV DEV ENQ
CU number 0:	40 40 7F 7F 2D

Specific poll,
CU number 0, device number 0:

CU	CU	DEV	DEV	ENQ
40	40	40	40	2D

Selection,
CU number 0, device number 0:

CU	CU	DEV	DEV	ENQ
60	60	40	40	2D

<text>: STX message ETX

Examples

Short read message, CU number 1, device number 5, CLEAR:

STX	CU	DEV	AID	ETX
02	C1	C5	6D	03

Status message, CU number 2, device number 10, printer offline:

SOH	%	R	STX	CU	DEV	S/0	S/1	ETX
01	6C	D9	02	C2	4A	C2	50	03

Command message, Erase All Unprotected:

STX	ESC	CMD	ETX
02	27	6F	03

Message transmitted in three text blocks:

STX	...	ETB	...	ETB	...	ETX
02	...	26	...	26	...	03

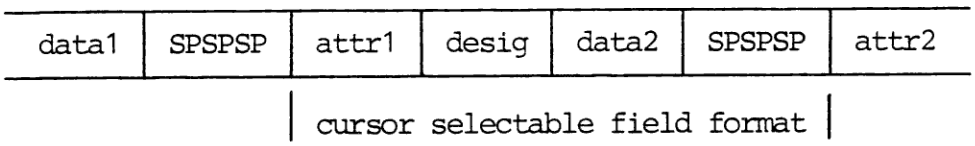
H. CURSOR-SELECT KEY AND MAGNETIC-STRIPE READER OPERATIONS H.

H.1 Cursor-Select Key Operations H.1

The cursor-select key (marked CURSR SELCT) permits the operator to select from a menu of displayed items and then cause a read message containing the selections to be transmitted to the host computer (see subsection 2.2.4). The cursor-select key is operated by positioning the cursor within a field formatted for cursor-select key operations and pressing the CURSR SELCT key.

H.1.1 Cursor Selectable Field Format H.1.1

A field to be used for cursor-select key operations must have the following format:



data1: Data character (preceding field on the same line as the cursor selectable field).

SPSPSP: Space or null characters. Three such characters must precede the attribute character defining the cursor selectable field, unless the attribute character is the first character on the line.

attr1: Attribute character (see subsection 4.2.1). The attribute character defines the field as cursor selectable (normal display) or intensified display and cursor selectable. The field may be an input field or a protected field. A cursor selectable field is the equivalent of an IBM selector-pen detectable (SPD) field.

desig: Designator character (see subsection H.1.2).

data2: Displayed alphameric character(s).

SPSPSP: Space or null characters. Three such characters must precede a new field following on the same line as the cursor selectable field.

attr2: Attribute character (succeeding field on the same line as the cursor selectable field).

The attribute character, designator character, and displayed alphameric character(s) must be on the same line. Should the field extend beyond one line, only that part of the field which is on the same line as the attribute character is cursor selectable. A maximum of 12 cursor selectable fields may precede the last cursor selectable field on a given line.

H.1.2 Designator Characters H.1.2

The effect of cursor selection, i.e. pressing the CURSR SELCT key after having positioned the cursor within a cursor selectable field, is determined by the designator character.

A question mark (?) designator character defines the cursor selectable field as a selection field. Cursor selection of such a field sets the MDT of the field, and changes the displayed designator character to a greater-than sign (>) to indicate successful selection. Cursor selection of the same field again will reset the MDT and change the designator character back to a question mark.

A space or null designator character (displayed as a blank) defines the cursor selectable field as an attention field. Cursor selection of such a field sets the MDT of the field, and generates a data attention (AID code $7E_{Hex}$). The read modified string in the response body of the read message (see subsection 4.1.1.4) will contain only the address of each modified field.

An ampersand (&) designator character defines the cursor selectable field as an attention field of a second type. Cursor selection of such a field sets the MDT of the field, and generates a

data attention (AID code $7D_{Hex}$) as if the SEND key had been pressed (see subsection 2.2.4).

Note that, if none of the above characters is present in the second character position of a cursor-selected field, the keyboard will sound an alarm.

H.2 Magnetic-Stripe Reader Operations H.2

The magnetic-stripe reader reads information encoded on magnetic-striped documents, such as operator identification badges or cards. The recorded information is read from the stripe when the operator inserts the document into the slot of the reader. The reader data is written, as a rule, into the terminal buffer at the location specified by the cursor. Once the information is read, the reader generates a data attention, and the use of the keyboard is inhibited (see subsection 2.2.4).

A configuration parameter determines whether the magnetic-striped document is to be handled according to IBM 3277, IBM 3278, or Alfaskop System 37 specifications.

H.2.1 IBM 3277 Specifications H.2.1

The magnetic-stripe reader reads the 3277-compatible numeric character set shown in figure 14. The reader data is written into the buffer at the location specified by the cursor, but is not displayed on the screen, i.e. the data is automatically secure. The AID code of the reader-generated data attention is $E6_{Hex}$. Once the document is read, it may be removed from the reader.

H.2.1.1 3277-Compatible Numeric Character Set H.2.1.1

The 3277-compatible numeric character set comprises 10 numeric data characters and 4 control characters, including a field separator. Each character consists of a 4-bit pattern plus an odd-parity bit. This bit pattern is recorded with the low-order bit first (i.e. 2^0 2^1 2^2 2^3 P). A longitudinal redundancy check (LRC) character is placed at the end, protected by an odd-parity bit of its own. Characters are recorded, low-order bit first, beginning at the left side of the magnetic stripe.

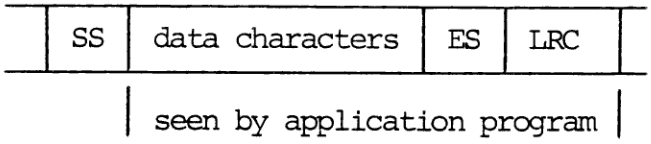
H.2.1.2 Magnetic-Stripe Capacity H.2.1.2

A maximum of 37 data characters can be encoded between the SS (Start Sentinel) and ES (End Sentinel) control characters at a density of 3 bits/mm (75 bits/inch).

At least 7 data characters must be encoded between SS and ES.

H.2.1.3 Magnetic-Stripe Format H.2.1.3

The magnetic-stripe recording format is as follows:



SS: Start Sentinel
ES: End Sentinel
LRC: Longitudinal Redundancy Check

SS (hex B) identifies the beginning and ES (hex F) the end of the data. The LRC character is calculated beginning with the SS and ending with the ES characters. See further subsection H.2.4.

H.2.2 IBM 3278 Specifications H.2.2

The magnetic-stripe reader reads the numeric and alphameric character sets shown in figures 15 and 16. The header of the magnetic-stripe record indicates which of the two character sets was used for recording. The reader data is written into the buffer at the location specified by the cursor, but is not displayed on the screen when secure data is specified by the header. The AID code of the reader-generated data attention is $E7_{Hex}$. Once the document is read, it may be removed from the reader.

H.2.2.1 Numeric and Alphameric Character Sets H.2.2.1

The numeric character set comprises 11 data characters (10 numeric characters and space) and 4 control characters. Each character consists of a 4-bit code plus an odd-parity bit.

The alphameric character set comprises 64 data characters (10 numeric and 54 nonnumeric characters including space) and 4 control characters.

Each nonnumeric character is composed of two hex characters, and each hex character consists of 4 bits plus an odd-parity bit. Viewing this as a paired 4-bit code, the letter M, for example, is recorded as hex D4, hex D being recorded first.

Each numeric character is composed of a single hex character, so that two numeric characters can be recorded in this paired 4-bit code structure. Consequently, there must be an even number of numeric characters in any contiguous string of numeric characters or, if an odd number of numeric characters is recorded, a filler character, hex A, must be added following the odd-numbered numeric character to preserve the paired 4-bit structure.

For both the numeric and the alphameric character sets, hex characters are recorded with the low-order bit first (i.e. 2^0 2^1 2^2 2^3 P).

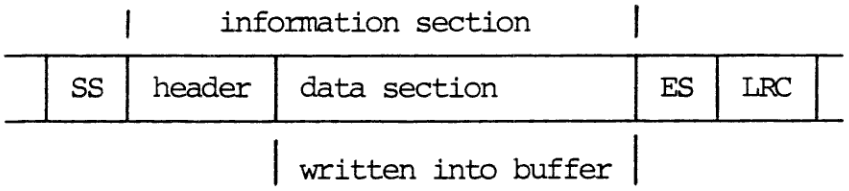
H.2.2.2 Magnetic-Stripe Capacities H.2.2.2

A maximum of 37 numeric data characters (37 hex codes) or 18 non-numeric data characters (36 hex codes) can be encoded between the SS (Start Sentinel) and ES (End Sentinel) control characters at a density of 3 bits/mm (75 bits/inch). If a combination of numeric and nonnumeric characters is recorded, the total number of hex codes must not exceed 37.

At least 7 hex codes must be encoded between SS and ES.

H.2.2.3 Magnetic-Stripe Format H.2.2.3

The magnetic-stripe recording format is as follows:



SS: Start Sentinel
ES: End Sentinel
LRC: Longitudinal Redundancy Check

SS (hex B) identifies the beginning and ES (hex F) the end of the information section. The LRC character is calculated beginning with the SS and ending with the ES characters. See further subsection H.2.4.

The header specifies whether the data section is secure and indicates the character set used in the data section.

A secure data section is specified when hex A immediately follows SS (hex B) as the first (only) character in the header. The character set is indicated by the presence or absence of hex C in the header. There are four possible combinations of secure/nonsecure data and numeric/alphameric character set:

- o secure data, numeric character set:
  hex B followed by hex A immediately followed by a numeric character or the space character (hex D) as the first character of the data section

- o nonsecure data, numeric character set:
  hex B immediately followed by a numeric character or the space character (hex D) as the first character of the data section

- o secure data, alphameric character set:
  hex B followed by hex A immediately followed by hex C in the second hex character position of the header

- o nonsecure data, alphameric character set:
  hex B immediately followed by hex C in the first hex character position and hex A (as a filler character) in the second hex character position of the header

H.2.3 Alfaskop System 37 Specifications H.2.3

The magnetic-stripe reader reads the 3277-compatible numeric character set described in subsection H.2.1. The ES and LRC characters are not seen by the application program. Of the four control characters, only the separation character (hex D) is sent to the host computer, as EBCDIC 40. The reader data is stored in a buffer separate from the terminal buffer. When the document is removed from the reader, the latter generates a second data attention (AID code $E6_{Hex}$). The response body of the read message (see subsection 4.1.1.4) contains only the address of the cursor.

H.2.4 Read Messages H.2.4

Once the magnetic-stripe information is read, the reader generates a data attention (see subsection 2.2.4). The contents of the response body of the read message (see subsection 4.1.1.4) depend on whether the display screen is unformatted or formatted (i.e. has at least one attribute character defined on initial presentation). When the SS (Start Sentinel) control character is

read, and the data is secure, the reader automatically generates an attribute character at the current cursor position (provided that the location is unprotected). This attribute character defines the following field as protected, alphameric, and nondisplay/nonprint. As the data characters are read into the buffer, they are stored starting at the first character position following the attribute character. The cursor does not move, but is repositioned when the document has been read.

If no field is defined by the application program, the response body of the read message will contain:

ADDR1 ADDR2	the address of the cursor on completion of the read operation
SBA ADDR1 ADDR2	a set buffer address order with the address of the first character position in the field defined by the reader-generated attribute character
data	the reader data followed by any other buffer contents

If the application program defines a field for reader data as an unprotected field, with the cursor directly following the attribute character, the response body of the read message will contain:

ADDR1 ADDR2	the address of the cursor on completion of the read operation
SBA ADDR1 ADDR2	a set buffer address order with the address of the first character position in the field defined by the application program, i.e. the address of the attribute character generated by the reader
SBA ADDR1 ADDR2	a set buffer address order with the address of the first character position in the field defined by the reader-generated attribute character

data	the reader data followed by any other data up to the next attribute character

If the application program defines a field for reader data as an unprotected field containing, say, the operator prompt "ENTER ID," followed by the cursor, the response body of the read message will contain:

ADDR1 ADDR2	the address of the cursor on completion of the read operation
SBA ADDR1 ADDR2	a set buffer address order with the address of the first character position in the field defined by the application program
data	ENTER ID
SBA ADDR1 ADDR2	a set buffer address order with the address of the first character position in the field defined by the reader-generated attribute character
data	the reader data followed by any other data up to the next attribute character

Note that the response body of the read message will also include any other fields for which the MDT is set. The position of the reader data in the response body will therefore depend on the position of the reader-data field in the format.

H.2.5 Error Conditions

H.2.5

Reader data is not written into the terminal buffer when any of several different error conditions exists. Thus the reader may detect invalid magnetic-stripe information, for example, or the cursor may be positioned in a protected field. Whenever an error condition arises, the Emulator displays an appropriate message, as described in the operating guide [1].

Character	Bit Pattern 2^0	2^1	2^2	2^3	P	Hex Code	EBCDIC Code
data:							
0	0	0	0	0	1	0	F0
1	1	0	0	0	0	1	F1
2	0	1	0	0	0	2	F2
3	1	1	0	0	1	3	F3
4	0	0	1	0	0	4	F4
5	1	0	1	0	1	5	F5
6	0	1	1	0	1	6	F6
7	1	1	1	0	0	7	F7
8	0	0	0	1	0	8	F8
9	1	0	0	1	1	9	F9
control:							
note 1	0	1	0	1	1	A	7A
note 2	1	1	0	1	0	B	7B
note 3	1	0	1	1	0	D	7D
note 4	1	1	1	1	1	F	7F

Notes:

1. Reserved for operator identification only; must be located in first data character position.
2. SS (Start Sentinel) character.
3. Field separator.
4. ES (End Sentinel) character.

Figure 14: 3277-Compatible Numeric Character Set.

Character	Bit Pattern 2^0	2^1	2^2	2^3	P	Hex Code	EBCDIC Code
data:							
0	0	0	0	0	1	0	F0
1	1	0	0	0	0	1	F1
2	0	1	0	0	0	2	F2
3	1	1	0	0	1	3	F3
4	0	0	1	0	0	4	F4
5	1	0	1	0	1	5	F5
6	0	1	1	0	1	6	F6
7	1	1	1	0	0	7	F7
8	0	0	0	1	0	8	F8
9	1	0	0	1	1	9	F9
space	1	0	1	1	0	D	40
control:							
note 1	0	1	0	1	1	A	not sent
note 2	1	1	0	1	0	B	not sent
note 3	0	0	1	1	1	C	not sent
note 4	1	1	1	1	1	F	not sent

Notes:

1. Secure data; may occur only in header.
2. SS (Start Sentinel) character; must be located in first character position of magnetic-stripe record.
3. Reserved; may occur only in header.
4. ES (End Sentinel) character; occurrence in data section will terminate reading of data section and cause following character to be read as LRC character.

Figure 15: Numeric Character Set.

Character	Bit Pattern 2^0	2^1	2^2	2^3	P	2^0	2^1	2^2	2^2	P	Hex Code	EBCDIC Code
data:												
0	0	0	0	0	1						0	F0
1	1	0	0	0	0						1	F1
2	0	1	0	0	0						2	F2
3	1	1	0	0	1						3	F3
4	0	0	1	0	0						4	F4
5	1	0	1	0	1						5	F5
6	0	1	1	0	1						6	F6
7	1	1	1	0	0						7	F7
8	0	0	0	1	0						8	F8
9	1	0	0	1	1						9	F9
A	0	0	1	1	1	1	0	0	0	0	C1	C1
B	0	0	1	1	1	0	1	0	0	0	C2	C2
C	0	0	1	1	1	1	1	0	0	1	C3	C3
D	0	0	1	1	1	0	0	1	0	0	C4	C4
E	0	0	1	1	1	1	0	1	0	1	C5	C5
F	0	0	1	1	1	0	1	1	0	1	C6	C6
G	0	0	1	1	1	1	1	1	0	0	C7	C7
H	0	0	1	1	1	0	0	0	1	0	C8	C8
I	0	0	1	1	1	1	0	0	1	1	C9	C9
J	1	0	1	1	0	1	0	0	0	0	D1	D1
K	1	0	1	1	0	0	1	0	0	0	D2	D2
L	1	0	1	1	0	1	1	0	0	1	D3	D3
M	1	0	1	1	0	0	0	1	0	0	D4	D4
N	1	0	1	1	0	1	0	1	0	1	D5	D5
O	1	0	1	1	0	0	1	1	0	1	D6	D6
P	1	0	1	1	0	1	1	1	0	0	D7	D7
Q	1	0	1	1	0	0	0	0	1	0	D8	D8
R	1	0	1	1	0	1	0	0	1	1	D9	D9
S	0	1	1	1	0	0	1	0	0	0	E2	E2
T	0	1	1	1	0	1	1	0	0	1	E3	E3
U	0	1	1	1	0	0	0	1	0	0	E4	E4
V	0	1	1	1	0	1	0	1	0	1	E5	E5

Figure 16: Alphameric Character Set (Part 1 of 3).

Character	Bit Pattern 2^0	2^1	2^2	2^3	P	2^0	2^1	2^2	2^3	P	Hex Code	EBCDIC Code
W	0	1	1	1	0	0	1	1	0	1	E6	E6
X	0	1	1	1	0	1	1	1	0	0	E7	E7
Y	0	1	1	1	0	0	0	0	1	0	E8	E8
Z	0	1	1	1	0	1	0	0	1	1	E9	E9
]	0	0	0	0	1	0	0	1	1	1	0C	4A
!	1	0	0	0	0	0	0	1	1	1	1C	5A
:	1	1	0	0	1	0	0	1	1	1	3C	7A
<	0	0	1	0	0	0	0	1	1	1	4C	4C
*	1	0	1	0	1	0	0	1	1	1	5C	5C
%	0	1	1	0	1	0	0	1	1	1	6C	6C
@	1	1	1	0	0	0	0	1	1	1	7C	7C
.	0	0	0	0	1	1	0	1	1	0	0D	4B
$	1	0	0	0	0	1	0	1	1	0	1D	5B
,	0	1	0	0	0	1	0	1	1	0	2D	6B
#	1	1	0	0	1	1	0	1	1	0	3D	7B
(	0	0	1	0	0	1	0	1	1	0	4D	4D
)	1	0	1	0	1	1	0	1	1	0	5D	5D
_	0	1	1	0	1	1	0	1	1	0	6D	6D
'	1	1	1	0	0	1	0	1	1	0	7D	7D
[	0	0	0	0	1	0	1	1	1	0	0E	4F
↑	1	0	0	0	0	0	1	1	1	0	1E	5F
?	0	1	0	0	0	0	1	1	1	0	2E	6F
"	1	1	0	0	1	0	1	1	1	0	3E	7F
+	0	0	1	0	0	0	1	1	1	0	4E	4E
;	1	0	1	0	1	0	1	1	1	0	5E	5E
>	0	1	1	0	1	0	1	1	1	0	6E	6E
=	1	1	1	0	0	0	1	1	1	0	7E	7E
\	0	1	1	1	0	0	0	0	0	1	E0	E0
/	0	1	1	1	0	1	0	0	0	0	E1	61
&	1	0	1	1	0	0	1	0	1	1	DA	50
-	0	1	1	1	0	0	1	0	1	1	EA	60
space	0	0	1	1	1	0	1	0	1	1	CA	40

Figure 16: Alphameric Character Set (Part 2 of 3).

Character	Bit Pattern 2^0 2^1 2^2 2^3 P	2^0 2^1 2^2 2^3 P	Hex Code	EBCDIC Code
control:				
note 1	0 1 0 1 1		A	not sent
note 2	1 1 0 1 0		B	not sent
note 3	0 0 1 1 1		C	not sent
note 4	1 1 1 1 1		F	not sent

Notes:

1. Secure data (header only) or filler character (header or data section).
2. SS (Start Sentinel) character; must be located in first character position of magnetic-stripe record.
3. Alphameric character set; may occur only in header.
4. ES (End Sentinel) character; occurrence in data section will terminate reading of data section and cause following character to be read as LRC character.

Figure 16: Alphameric Character Set (Part 3 of 3).

Note that the characters shown for EBCDIC codes 4A, 5A, 7C, 5B, 7B, 4F, 5F, 7F, and E0 are those of the "US English" EBCDIC code set (see appendix C).

I. INDEX I.

↓ key: see Down key

↖ key: see Home key

|← key: see Field backward key

→| key: see Field forward key

↵ key: see New line key

RETURN LETTER

Title: RC855 IBM 3270 BSC Emulator Reference Manual

RCSL No.: 42-i2156

A/S Regnecentralen af 1979/RC Computer A/S maintains a continual effort to improve the quality and usefulness of its publications. To do this effectively we need user feedback, your critical evaluation of this manual.

Please comment on this manual's completeness, accuracy, organization, usability, and readability:

_____

_____

_____

_____

Do you find errors in this manual? If so, specify by page.

_____

_____

_____

_____

How can this manual be improved?

_____

_____

_____

_____

Other comments?

_____

_____

_____

_____

_____

Name: _____ Title: _____

Company: _____

Address: _____

Date: _____

Thank you

42-i 1288

. . . . . Fold here . . . . .

. . . . . Do not tear - Fold here and staple . . . . .

Affix
postage
here

REGNECENTRALEN
af 1979

Information Department
Lautrupbjerg 1
DK-2750 Ballerup
Denmark